STABLE AND EFFICIENT COMPUTATION OF GENERALIZED POLAR DECOMPOSITIONS

PETER BENNER*, YUJI NAKATSUKASA†, AND CAROLIN PENKE*

Abstract. We present methods for computing the generalized polar decomposition of a matrix based on the dynamically weighted Halley (DWH) iteration. This method is well established for computing the standard polar decomposition and a stable implementation is available, where matrix inversion is avoided and QR decompositions are used instead. We establish a natural generalization of this approach for computing generalized polar decompositions with respect to signature matrices. Again, the inverse can be avoided by using a generalized QR decomposition called hyperbolic QR decomposition. However, this decomposition does not show the same favorable stability properties as its orthogonal counterpart. We overcome the numerical difficulties by generalizing the CholeskyQR2 method. This method computes the standard QR factorization in a stable way via two successive Cholesky factorizations. An even better numerical stability is achieved by employing permuted graph bases, yielding residuals of order 10^{-14} even for badly conditioned matrices, where other methods fail.

Key words. Generalized Polar Decomposition, Dynamically Weighted Halley Iteration, Matrix Sign Function, LDL^{T} Factorization, Hyperbolic QR Decomposition, Indefinite QR Decomposition, Permuted Graph Basis

1. Introduction.

For $\mathbb{K} = \mathbb{C}$ or $\mathbb{K} = \mathbb{R}$, the polar decomposition of a matrix $A \in \mathbb{K}^{m\times n}$, $m \geq n$, is given as

$$A = UH, \quad U^*U = I, \quad H = H^* \geq 0 \tag{1.1}$$

where $U \in \mathbb{K}^{m\times n}$ has unitary columns and $H \in \mathbb{K}^{n\times n}$ is positive semidefinite. $\cdot^*$ is a placeholder for the transpose $\cdot^{\mathsf{T}}$ or conjugate transpose $\cdot^{\mathsf{H}}$. Applications of the polar decompositions arise from looking at the following key aspects.

(a) The polar decomposition and the singular value decomposition (SVD) are intimately connected.
(b) The factors of the polar decomposition possess particular best-approximation qualities and can be used to solve the orthogonal Procrustes problem (see [28, Chapter 8]).
(c) The unitary polar factor U of a Hermitian matrix $A = UH$ coincides with its matrix sign function $\operatorname{sign}(A) = U$ [28].

Elaborating on aspect (a), the polar decomposition is classically computed by dropping the middle term in the SVD, but recently, more efficiently alternatives have been developed [36, 41, 42, 43] and rather, the polar decomposition can now be seen as a first step for computing the SVD [51].

The matrix sign function, referred in aspect (c), is a widely used tool for acquiring invariant subspaces of a matrix. This property is used to solve matrix equations [8, 47] and develop parallelizable algorithms for solving eigenvalue problems [3, 52]. Therefore, efficient iterations for computing the polar decomposition, such as the QDWH iteration [43] and its successor based on Zolotarev's functions [42], can be used to improve these methods for Hermitian matrices.

The concept of polar decompositions can be generalized in terms of non-standard inner product spaces. The papers [11, 12, 39] treat inner products induced by Hermit-

*Max Planck Institute for Dynamics of Complex Technical Systems, Sandtorstr. 1, 39106 Magdeburg, Germany (benner@mpi-magdeburg.mpg.de, penke@mpi-magdeburg.mpg.de)
†Mathematical Institute, University of Oxford, Oxford, OX2 6GG, UK (nakatsukasa@maths.ox.ac.uk)

ian matrices, while [29, 30] provide a more general treatment. The group theoretical foundations are laid out in [24] and used to provide a systematic derivation of a large class of matrix decompositions. Let $A \in \mathbb{K}^{m\times n}$, and $M \in \mathbb{K}^{m\times m}$, $N \in \mathbb{K}^{n\times n}$ be nonsingular. Under certain assumptions on A, M and N (see [30]), A has a *(canonical) generalized polar decomposition* with respect to the inner products induced by M and N:

$$A = WS, \tag{1.2}$$

where $W \in \mathbb{K}^{m\times n}$ is a partial (M,N)-isometry, i.e. $AA^{\star_{M,N}}A = A$. The notation $A^{\star_{M,N}}$ refers to a generalized transpose with respect to the two inner products. $S \in \mathbb{K}^{n\times n}$ is self-adjoint with respect to N and its nonzero eigenvalues are contained in the open right half plane. Details are given in Section 2.

The standard polar decomposition (1.1) can be used to solve the orthogonal Procrustes problem, arising in fields such as marketing in the context of multidimensional scaling [13]. A generalized polar decomposition can be used as a tool to solve the nonorthogonal variant [34].

In analogy with the standard setting, the factor W of the generalized polar decomposition (1.2) coincides with the matrix sign function of a square matrix A if A is self-adjoint with respect to the defining inner product. This is shown in Section 2 of this paper. Finding efficient iterations for computing the generalized polar decomposition can therefore lead to new methods for matrix equations and eigenvalue problems involving self-adjoint matrices.

In this work, we present some results on how generalized polar decompositions can be computed based on the dynamically weighted Halley (DWH) iteration. This iteration is successful in computing the standard polar decomposition in an efficient and stable way [43]. We focus on the important subclass of inner products induced by signature matrices, i.e. diagonal matrices with $+1$ and -1 as diagonal values, denoted by Σ throughout the paper. Self-adjoint matrices with respect to Σ are called pseudosymmetric. They show up in the field of computational quantum physics [20, 44], from which our main motivation is drawn. Discretizations of differential equations often lead to structured eigenvalue problems of very large size. Consider, e.g., the Bethe-Salpeter eigenvalue problem [45], which aims to find eigenvalues and eigenvectors of a block matrix

$$H_{BS} = \begin{bmatrix} A & B \\ -\bar{B} & -\bar{A} \end{bmatrix} = \begin{bmatrix} A & B \\ -B^{\mathsf{H}} & -A^{\mathsf{T}} \end{bmatrix}, \quad A = A^{\mathsf{H}}, \quad B = B^{\mathsf{T}} \in \mathbb{C}^{n\times n}.$$

These are used to determine optical properties of crystalline systems [48] or molecules [9]. H_{BS} has the additional property, coming from physical constraints of the original problem, that ΣH_{BS} is positive definite for $\Sigma = \operatorname{diag}(I_n, -I_n)$. Similar structures arise in different contexts of electronic structure theory [5, 22, 38]. We call pseudosymmetric matrices with this property definite pseudosymmetric matrices. For these matrices in particular, the convergence behaviour of our proposed method will turn out to be as good as in the standard setting defined by the Euclidean inner product. Pseudosymmetric matrices also play a role in describing damped oscillations of linear systems. See [53], where they are called J-Hermitian and definite pseudosymmetric matrices are called J-positive.

The remainder of this paper is structured as follows. Section 2 fixes the notation on inner products and related aspects which form basic concepts used throughout the remaining paper. Section 3 clarifies how generalizations of the QR factorization can

be used to compute matrices that are orthogonal with respect to non-standard inner products. In Section 4, we recapitulate the central ideas of the QDWH algorithm. Section 5 shows how they can be applied in order to compute a generalized polar decomposition. We show general results and then restrict ourselves to inner products induced by signature matrices. Here, inverses can be avoided by using the decompositions presented earlier in Section 3. The introduction of permuted graph bases can improve the stability of the computation of the generalized polar factor. Details are found in Section 6. Section 7 gives numerical results on the questions of stability and convergence. Conclusions and further research directions are given in Section 8.

2. Preliminaries. Following [29] and [37], we provide basic notation regarding inner products needed for the generalized polar decomposition. A nonsingular matrix M defines an *inner product* on $\mathbb{K}^n$ (where $\mathbb{K} \in \{\mathbb{C}, \mathbb{R}\}$), which is a bilinear or sesquilinear form $\langle ., . \rangle_M$, given by

$$\langle x, y \rangle_M = \begin{cases} x^{\mathsf{T}} M y \text{ for bilinear forms,} \\ x^H M y \text{ for sesquilinear forms,} \end{cases}$$

for $x, y \in \mathbb{K}^n$. We use $\cdot^*$ throughout the paper to indicate transposition $\cdot^{\mathsf{T}}$ or conjugated transposition $\cdot^{\mathsf{H}}$, depending on whether a bilinear or sesquilinear form is given. We overline a quantity to denote complex conjugation.

For a matrix $A \in \mathbb{K}^{m \times n}$, the matrix $A^{\star_{M,N}} \in \mathbb{K}^{n \times m}$ denotes the adjoint with respect to the inner products defined by the nonsingular matrices $M \in \mathbb{K}^{m \times m}$ and $N \in \mathbb{K}^{n \times n}$. This matrix is uniquely defined by satisfying the identity

$$\langle Ax, y \rangle_M = \langle x, A^{\star_{M,N}} y \rangle_N$$

for all $x \in \mathbb{K}^n, y \in \mathbb{K}^m$. The matrix $A^{\star_{M,N}}$ is called the (M, N)-*adjoint* of A and fulfills

$$A^{\star_{M,N}} = N^{-1} A^* M. \tag{2.1}$$

A is (M, N)-*orthogonal* if $A^{\star_{M,N}} A = I_n$ and this notion is generalized in the form of partial (M, N)-isometries. A matrix A is called a *partial* (M, N)-*isometry* when $AA^{\star_{M,N}} A = A$.

If A is square and $M = N$, the notation simplifies and the M-*adjoint* is given by $A^{\star_M} := A^{\star_{M,M}}$. We call a square matrix A an *(M-)automorphism* if $A^{\star_M} = A^{-1}$ (given the inverse exists), and *(M-)self-adjoint* if $A = A^{\star_M}$.

In the following, we give basic results regarding the generalized polar decomposition (1.2) that can be found in [29] or [30]. The canonical generalized polar decomposition can be defined if $M \in \mathbb{K}^{m \times m}$ and $N \in \mathbb{K}^{n \times n}$ form an *orthosymmetric pair*, i.e. if they satisfy

(a) $M^{\mathsf{T}} = \beta M$, $N^{\mathsf{T}} = \beta N$, $\beta = \pm 1$ for bilinear forms,
(b) $M^{\mathsf{H}} = \alpha M$, $N^{\mathsf{H}} = \alpha N$, $|\alpha| = 1$ for sesquilinear forms.

Definition 2.1 (Definition 3.6 in [30]). *A matrix $A \in \mathbb{K}^{m \times n}$ has a* canonical generalized polar decomposition *with respect to an orthosymmetric pair of matrices $M \in \mathbb{K}^{m \times m}$ and $N \in \mathbb{K}^{n \times n}$, if there exists a partial (M, N)-isometry W and an N-self-adjoint matrix S, whose eigenvalues all have positive real parts, s.t.*

$$A = WS,$$

and $\operatorname{range}(W^{\star_{M,N}}) = \operatorname{range}(S)$.

If A has full column rank, W is (M,N)-orthogonal and, if additionally A is square and $M = N$, then W is an N-automorphism.

In contrast to the standard polar decomposition (see Theorem 8.1 in [28]), the existence of the (canonical) generalized polar decomposition can in general not be guaranteed. The following theorem clarifies this issue.

THEOREM 2.2 (Existence of the canonical generalized polar decomposition, Theorem 3.9 in [30]). *A matrix $A \in \mathbb{K}^{m\times n}$ has a unique canonical generalized polar decomposition with respect to the orthosymmetric pair M, N if and only if*

1. *$A^{\star_M} A$ has no eigenvalues on the negative real axis,*
2. *if zero is an eigenvalue of $A^{\star_{M,N}} A$, then it is semisimple and*
3. *$\ker(A^{\star_{M,N}}) = \ker(A)$.*

In case of existence, we have $S = (A^{*_{M,N}} A)^{\frac{1}{2}}$ and $W^{\star_{M,N}} WS = S$. Just as the standard polar decomposition, the generalized polar decomposition is related to the matrix sign function, which is a generalization of the scalar sign function

$$\operatorname{sign}(z) = \begin{cases} 1, & \operatorname{Re}(z) > 0, \\ -1, & \operatorname{Re}(z) < 0, \end{cases} \qquad z \in \mathbb{C},\ z \notin \mathrm{i}\mathbb{R},$$

applied to matrices. For a detailed treatment see [28], Chapter 5. Let a square matrix A without purely imaginary eigenvalues have a Jordan decomposition $A = Z \operatorname{diag}(J_+, J_-)\, Z^{-1}$, where $J_+ \in \mathbb{K}^{n_+\times n_+}$ contains all Jordan blocks associated with eigenvalues with positive real part and $J_- \in \mathbb{K}^{n_-\times n_-}$ contains Jordan blocks associated with eigenvalues with negative real part. Then the matrix sign function is defined as

$$\operatorname{sign}(A) := Z \operatorname{diag}\big(I_{n_+}, -I_{n_-}\big)\, Z^{-1}.$$

THEOREM 2.3. *Let M be a nonsingular matrix and $A \in \mathbb{K}^{n\times n}$ be self-adjoint with respect to the inner product induced by M. If A has no purely imaginary eigenvalues or eigenvalues equal to zero,* $\operatorname{sign}(A)$ *and the canonical generalized polar decomposition (with respect to M) $A = WS$ are well-defined and they are connected in form of*

$$\operatorname{sign}(A) = W.$$

Proof. The matrix sign function can be expressed as [28]

$$\operatorname{sign}(A) = A(A^2)^{-1/2}.$$

The generalized polar decomposition $A = WS$ is well-defined with a unique self-adjoint factor S if all eigenvalues of $M^{-1}A^*MA$ are positive real. For self-adjoint matrices we have $M^{-1}A^*M = A$, so $M^{-1}A^*MA = A^2$ can only have negative real eigenvalues if A has purely imaginary eigenvalues. A^2 can only have an eigenvalue equal to zero if A has an eigenvalue equal to zero. So $A = WS$ is well-defined because A has only real non-zero eigenvalues. The principal matrix square root $S = (M^{-1}A^*MA)^{\frac{1}{2}}$ exists and is uniquely defined because all eigenvalues of $M^{-1}A^*MA$ are positive real. W is given as

$$W = A(M^{-1}A^*MA)^{-1/2} = A(A^2)^{-1/2} = \operatorname{sign}(A)\,. \qquad \square$$

3. Connections between LDL^{T} factorizations and hyperbolic QR decompositions.

3.1. The hyperbolic QR factorization. A matrix $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)$, where $\sigma_i \in \{+1, -1\}$ for $i = 1, \ldots, n$, is called a *signature matrix*. We search for a way to compute $(\Sigma, \hat{\Sigma})$-orthogonal bases, which span a given subspace. While Σ is a given signature matrix, $\hat{\Sigma}$ can be another arbitrary signature matrix. Matrices that are $(\Sigma, \hat{\Sigma})$-orthogonal are also called hyperexchange matrices [27] and can be used to solve indefinite least square problems [10].

The methods presented in this section take a rectangular matrix $A \in \mathbb{K}^{m\times n}$ and signature matrix Σ as input and deliver two outputs. These are another signature matrix $\hat{\Sigma}$, and $H \in \mathbb{K}^{m\times n}$, which spans the same subspace as the column space of A and is $(\Sigma, \hat{\Sigma})$-orthogonal. Subspace representations of this kind will be used in the computation of generalized polar decompositions (Section 5). A classic method for computing such a subspace representation uses the hyperbolic QR decomposition.

THEOREM 3.1 (The hyperbolic QR decomposition [17]). *Let $\Sigma \in \mathbb{R}^{m\times m}$ be a signature matrix, $A \in \mathbb{K}^{m\times n}$, $m \geq n$. Suppose all the leading principal submatrices of $A^*\Sigma A$ are nonsingular. Then there exists a signature matrix $\hat{\Sigma} = P^{\mathsf{T}}\Sigma P$, where P is a permutation, a $(\Sigma, \hat{\Sigma})$-orthogonal matrix $H \in \mathbb{K}^{m\times m}$ (i.e. $H^*\Sigma H = \hat{\Sigma}$), and an upper triangular matrix $R \in \mathbb{R}^{n\times n}$, such that*

$$A = H \begin{bmatrix} R \\ 0 \end{bmatrix}.$$

The hyperbolic QR decomposition is unique when the diagonal values of R are restricted to be positive real [50].

Remark 3.2. The hyperbolic QR decomposition can be truncated to form a thin hyperbolic QR decomposition

$$A = H_0 R,\ H_0 \in \mathbb{K}^{m\times n},\ R \in \mathbb{K}^{n\times n},\ H_0^*\Sigma H_0 = \hat{\Sigma}_0.$$

H_0 contains the first n columns of H and $\hat{\Sigma}_0$ contains the $n \times n$ leading submatrix of $\hat{\Sigma}$, where H and $\hat{\Sigma}$ are given in Theorem 3.1.

The hyperbolic QR decomposition can be computed by accumulating transformations that introduce zeros below the diagonal, similar to the standard QR decomposition. We give a quick idea on how these elimination matrices are computed. For a more formal treatment, see e.g. [54]. For a given vector x and a given signature matrix Σ, we look for a transformation H such that $H^{-1}x = de_1$, where $d \in \mathbb{K}$, e_1 denotes the first unit vector and $H^{\mathsf{H}}\Sigma H = \hat{\Sigma}$ is another signature matrix. The two kinds of transformations used are orthogonal Householder transformations and hyperbolic Givens rotations. For illustrative purposes suppose $x \in \mathbb{C}^{2n}$ and $\Sigma = \mathrm{diag}(I_n, -I_n)$. Let

$$H_1 = \begin{bmatrix} H_+ & \\ & H_- \end{bmatrix},$$

where H_+ and H_- are Householder transformations of dimension $n \times n$, such that $H_1^{-1}x = ae_1 + be_{n+1}$, where $a, b \in \mathbb{K}$. We have $H_1^{\mathsf{H}}\Sigma H_1 = \Sigma$. The b entry in position $n + 1$ is annihilated by a hyperbolic Givens rotation acting on row 1 and $n + 1$. We

achieve $G^{-1}\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} d \\ 0 \end{bmatrix}$ by defining

$$G^{-1} = \begin{bmatrix} c & -s \\ -\overline{s} & c \end{bmatrix},$$

$$\text{where } \begin{cases} c = |a|/\sqrt{|a|^2 - |b|^2},\ s = e^{i\phi}|b|/\sqrt{|a|^2 - |b|^2} & \text{if } |a| > |b|, \\ c = |a|/\sqrt{|b|^2 - |a|^2},\ s = e^{i\phi}|b|/\sqrt{|b|^2 - |a|^2} & \text{if } |a| < |b|, \end{cases}$$

with $\phi = \arg a - \arg b$.

G is given as $G = \begin{bmatrix} c & \overline{s} \\ s & c \end{bmatrix}$. For the $|a| > |b|$ case we have

$$G^{\mathsf{H}} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} G = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{3.2}$$

For $|a| < |b|$ there is a sign switch in the signature matrix,

$$G^{\mathsf{H}} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} G = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{3.3}$$

If a and b are real then G is also real. Embedding G into a larger matrix H_2 (equal to the identity except in rows and columns 1 and $n+1$) gives the sought-after transformation $H = H_1 H_2$, such that $H^{\mathsf{H}} \Sigma H = \hat{\Sigma}$ is another signature matrix, in which $+1$ at diagonal position 1 and -1 at diagonal position $n+1$ have been interchanged if (3.3) takes effect. If (3.2) takes effect, the signature matrix does not change: $\Sigma = \hat{\Sigma}$.

The presented method works not only for the specific signature matrix $\Sigma = \operatorname{diag}(I_n, -I_n)$. For an arbitrary signature matrix Σ, a matrix taking over the role of H_+ acts on the rows corresponding to positive entries of Σ. A matrix in the role of H_- acts on the remaining rows. A first full-size Householder transformation (H_1 in the example above) is set up as a combination of the two smaller matrices. A matrix, embedding a Givens rotation, (given as H_2 in the example above) then acts on the remaining two entries and may or may not introduce a sign switch in the signature matrix. In (3.1), the case $|a| = |b|$ is not covered and in this case no suitable matrix G exists. The assumptions in Theorem 3.1 prevent this from happening, but, if a and b are close, G becomes ill-conditioned. This can lead to an instability in algorithms employing this kind of column elimination.

In order to overcome these potential instabilities, we once again take a look at the standard QR decomposition. Here, we can find Cholesky-QR as an alternative computational approach, explained below. It has been rarely considered because its unmodified variant is less stable than the classical approach using Householder transformations.

The orthogonal QR decomposition is connected to a Cholesky factorization in the following way [55]. If $A = QR$ is a QR decomposition, then $A^*A = R^*R$ is a Cholesky factorization of A^*A. Conversely, if the Cholesky factorization $A^*A = R^*R$ with nonsingular R is given, $Q = AR^{-1}$ is the orthogonal factor of the QR decomposition.

In the indefinite setting an analogous connection exists between the hyperbolic QR factorization (Theorem 3.1) and a scaled variant of the LDL^{T} factorization given in [25, Thm. 4.1.3].

LEMMA 3.3. *Let $A \in \mathbb{K}^{m\times n}$ have a decomposition $A = HR$, $H \in \mathbb{K}^{m\times n}$, $R \in \mathbb{K}^{n\times n}$. Then*

$$H^*\Sigma H = \hat{\Sigma} \qquad \Leftrightarrow \qquad A^*\Sigma A = R^*\hat{\Sigma}R. \tag{3.4}$$

Remark 3.4. If the right side of the equivalence in (3.4) with nonsingular R is given, $H = AR^{-1}$ can be recovered from A and R. In the case of signature matrices, the right side can be computed from an LDL^T decomposition $A^*\Sigma A = LDL^*$, where L is unit lower triangular, D is real diagonal. Then $R := |D|^{\frac{1}{2}}L^*$ and $\hat{\Sigma} := \operatorname{sign}(D)$ (containing the signs of the diagonal values in D) fulfill $A^*\Sigma A = R^*\hat{\Sigma}R$.

The LDL^T factorization with a strictly diagonal D is typically not used in modern algorithms, as it becomes unstable when small diagonal values appear [2]. Instead, D is allowed to be block-diagonal with 1×1 and 2×2 blocks, and a pivoting scheme is employed [16]. This yields an LDL^T factorization $A = PLDL^*P^\mathsf{T}$, where P is a permutation matrix, L is unit lower triangular and D is block-diagonal. We call this factorization "LDL^T factorization with pivoting" or "block LDL^T factorization" in order to distinguish it from the "diagonal LDL^T factorization". The additional degrees of freedom destroy the uniqueness property, but allow for a more stable computation. Several backward stable algorithms have been developed (see [14, 15]) and well-established implementations are available in software packages such as LAPACK and MATLAB [2, 23]. In the latter, the implementation is given as the `ldl` command.

Remark 3.4 points out how the hyperbolic QR decomposition can be computed from the diagonal LDL^T decomposition. If instead the LDL^T decomposition with pivoting is used, one obtains the (thin) indefinite QR decomposition, which is not unique anymore.

THEOREM 3.5 ((Thin) indefinite QR decomposition [50]). *Let $\Sigma \in \mathbb{K}^{m\times m}$ be a signature matrix, $A \in \mathbb{K}^{m\times n}$, $m \geq n$. Suppose $A^*\Sigma A$ is nonsingular. Then there exists a decomposition*

$$A = HRP^\mathsf{T}, \qquad H \in \mathbb{K}^{m\times n},\ R \in \mathbb{K}^{n\times n},\ P \in \mathbb{R}^{n\times n},$$

where P is a permutation matrix. There exists $P_\Sigma \in \mathbb{R}^{m\times n}$ which contains n columns of an $m\times m$ permutation matrix and defines the signature matrix $\hat{\Sigma} = P_\Sigma^\mathsf{T}\Sigma P_\Sigma$. The matrix H is $(\Sigma, \hat{\Sigma})$-orthogonal (i.e. $H^*\Sigma H = \hat{\Sigma}$), and R is block-upper triangular with blocks of size 1×1 and 2×2.

The difference between the indefinite QR decomposition (Theorem 3.5) and the hyperbolic QR decomposition (Theorem 3.1) is that pivoting is introduced, which results in the second permutation matrix P. Blocks of size 2×2 appear on the diagonal of R, and the assumption on $A^*\Sigma A$ is weaker. This decomposition can be computed via the successive use of transformation matrices, as described in [49], similar to the hyperbolic QR decomposition (Theorem 3.1).

A perturbation analysis for the computation of the hyperbolic QR factorization (Theorem 3.1), i.e. the triangular case of the indefinite QR factorization in Theorem 3.5, is given in [50] and more recently in [35].

Computing the indefinite QR factorization via the LDL^T factorization proceeds as follows.

1. Compute an LDL^T factorization $A^*\Sigma A = PLDL^*P^\mathsf{T}$, where D is block-diagonal.
2. Diagonalize D, i.e., compute unitary V, diagonal Λ, s.t. $V\Lambda V^* = D$. V has the same block-diagonal structure as D.

3. Set $R = |\Lambda|^{\frac{1}{2}} V^* L^*$, $H = APR^{-1}$.

3.2. LDLIQR2: Computing the indefinite QR factorization via two LDL^T decompositions. Up to now, we explored how the indefinite QR decomposition is computed from an LDL^T factorization when exact arithmetic is assumed. Due to rounding errors, this method may not yield results with a satisfying accuracy. In this section, we describe an additional step serving as a remedy in this case.

In [55], the CholeskyQR2 algorithm is formulated and following these ideas we derive the indefinite variant (see also [7]). We call the algorithm LDLIQR2, standing for **LDL**$^\mathsf{T}$-based computation of the **I**ndefinte **QR** decomposition, applied **twice**. It computes a $(\Sigma, \hat{\Sigma})$-orthogonal basis of the subspace spanned by a matrix A. A signature matrix is given as Σ and $\hat{\Sigma}$ is another signature matrix determined by the algorithm. It starts by computing the indefinite QR factorization $A = H_1 R_1 P_1^\mathsf{T}$ via the LDL^T factorization with pivoting as described in the previous section. Then as a second step, the indefinite QR decomposition $H_1 = H R_2 P_2^\mathsf{T}$ is computed using the same method. This yields a factorization

$$A = H R_2 P_2^\mathsf{T} R_1 P_1^\mathsf{T}, \text{ with } R_1,\ R_2 \text{ upper triangular}, \quad P_1,\ P_2 \text{ permutation matrices}. \tag{3.5}$$

In exact arithmetic, the second step is redundant, as the hyperbolic QR decomposition of a $(\Sigma, \hat{\Sigma})$-orthogonal H is $H = HI$. In floating point arithmetic, however, performing the second step provides improvements regarding the accuracy of the computed factorization. P_2 will in practice often be the identity matrix. In this case, we have computed an instance of the Indefinite QR factorization given in Theorem 3.5 with $R := R_2 R_1$ and $P := P_1$. For our application we are just interested in a $(\Sigma, \hat{\Sigma})$-orthogonal basis, so the exact shape of R in a decomposition $A = HR$ does not matter. The method is formulated in Algorithm 3.1.

Algorithm 3.1 LDLIQR2: Compute $(\Sigma, \hat{\Sigma})$-orthogonal basis via double LDL^T factorization with pivoting.

Input: $A \in \mathbb{K}^{m \times n}$, with full column rank, $\Sigma \in \mathbb{R}^{n \times n}$ is a signature matrix.
Output: $(\Sigma, \hat{\Sigma})$-orthogonal $H \in \mathbb{K}^{m \times n}$ and R_1, P_1, R_2, $P_2 \in \mathbb{K}^{n \times n}$ as in (3.5).

// **First pass:**
1: $[L_1, D_1, P_1] \leftarrow \texttt{ldl}(A^* \Sigma A)$
2: $[V_1, \Lambda_1] \leftarrow \texttt{eig}(D)$ ▷ V_1 is block-diagonal.
3: $H_1 \leftarrow A P_1 L_1^{-*} V_1 |\Lambda_1|^{-\frac{1}{2}}$
4: $R_1 \leftarrow |\Lambda_1|^{\frac{1}{2}} V_1^* L_1^*$
// **Second pass:**
5: $[L_2, D_2, P_2] \leftarrow \texttt{ldl}(H^* \Sigma H)$
6: $[V_2, \Lambda_2] \leftarrow \texttt{eig}(D)$ ▷ V_2 is block-diagonal.
7: $H \leftarrow H_1 P_2 L_2^{-*} V_2 |\Lambda_2|^{-\frac{1}{2}}$
8: $R_2 \leftarrow |\Lambda_2|^{\frac{1}{2}} V_2^* L_2^*$
// **Compute new signature matrix:**
9: $\hat{\Sigma} \leftarrow \Lambda_2 |\Lambda_2|^{-1}$

If one is only interested in computing H and $\hat{\Sigma}$, then Steps 4 and 8, computing R_1 and R_2, can be omitted.

4. The QDWH algorithm for computing the standard polar decomposition. Methods for the computation of the polar decomposition of a matrix $A = UH$ (1.1) have been studied extensively in recent years. Once the orthogonal polar factor is computed, the symmetric factor can be recovered via $H = U^*A$. In floating point arithmetic, the symmetric factor is not symmetric due to rounding errors and thus $H := (H + H^*)/2$ can be performed to guarantee numerical symmetry.

A current state-of-the-art iterative method for computing the polar factor is the QDWH algorithm [41]. It is based on the well-known Halley iteration which is a member of the Padé family of iterations [33]. The Dynamically Weighted Halley (DWH) iteration introduces the weights a_k, b_k, $c_k \in \mathbb{R}^+$ and is given as

$$X_{k+1} = X_k(a_k I + b_k X_k^* X_k)(I + c_k X_k^* X_k)^{-1}, \quad X_0 = \frac{1}{\|A\|_2} A. \tag{4.1}$$

Convergence is globally guaranteed with an asymptotic cubic rate, provided A has full column rank. In order to choose the weights in an optimal fashion, Iteration (4.1) is understood as an iteration acting on the singular values of the iterate X_k. Let $X_k = U_S \Sigma_k V_S{}^*$ be the SVD of X_k. Then one step of Iteration (4.1) yields

$$X_{k+1} = U_S g_k(\Sigma_k) V_S{}^*, \tag{4.2}$$

where

$$g_k(x) = x \frac{a_k + b_k x^2}{1 + c_k x^2}. \tag{4.3}$$

The singular value $\sigma_{i,k+1}$ of X_{k+1} is hence given by a rational function acting on the singular value $\sigma_{i,k}$ of X_k,

$$\sigma_{i,k+1} = g_k(\sigma_{i,k}). \tag{4.4}$$

The singular values converge to 1 as X_k approaches the polar factor. Let $\ell (=: \ell_0)$ be a lower bound to the singular values of X_0. Due to the initial scaling with $1/\|A\|_2$ the singular values of X_0 lie between 0 and 1. A successful strategy for accelerating convergence can be developed by minimizing the distance of ℓ_k, a lower bound on the singular values of X_k, to 1 in each iteration. This line of thoughts leads to weights chosen as

$$a_k = h(\ell_k), \quad b_k = (a_k - 1)^2/4, \quad c_k = a_k + b_k - 1, \quad \ell_{k+1} = g_k(\ell_k), \tag{4.5}$$

where

$$h(\ell) = \sqrt{1+d} + \frac{1}{2}\sqrt{8 - 4d + \frac{8(2-\ell^2)}{\ell^2\sqrt{1+d}}}, \quad d = \sqrt[3]{\frac{4(1-\ell^2)}{\ell^4}}. \tag{4.6}$$

The weights in (4.5) are the solutions of an optimization problem, which is how they were introduced in [41]. Another derivation considers the best rank-(3,2) rational approximation of the sign function, leading to the same weights given in (4.5). The latter approach can be extended to rational approximations of higher order (Zolotarev's functions), see [42].

For matrices A with condition number $\kappa_2(A) < 10^{16}$, convergence within 6 iterations can be guaranteed using these weights [41]. A simple rewrite of the iteration

(4.1)

$$
\begin{aligned}
& X_k(a_k I + b_k X_k^* X_k)(I + c_k X_k^* X_k)^{-1} \\
= \;& \frac{b_k}{c_k} X_k + \left(a_k - \frac{b_k}{c_k}\right) X_k (I + c_k X_k^* X_k)^{-1}
\end{aligned} \tag{4.7}
$$

leads to two distinct implementation variants: $(I + c_k X_k^* X_k)$ is a symmetric positive definite matrix and its linear solve can be done using a Cholesky factorization.

$$
\begin{cases}
Z_k = I + c_k X_k^* X_k, \quad W_k = \mathrm{chol}(Z_k), \\
X_{k+1} = \frac{b_k}{c_k} X_k + \left(a_k - \frac{b_k}{c_k}\right) X_k W_k^{-1} W_k^{-*}.
\end{cases} \tag{4.8}
$$

It can also be shown that $X_k(I + c_k X_k^{\mathsf{T}} X_k)^{-1}$ is equivalently computed via a QR decomposition, which leads to the actual QR-based Dynamically Weighted Halley (QDWH) iteration

$$
\begin{cases}
\begin{bmatrix} \sqrt{c_k} X_k \\ I \end{bmatrix} = \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} R, \\
X_{k+1} = \frac{b_k}{c_k} X_k + \frac{1}{\sqrt{c_k}} \left(a_k - \frac{b_k}{c_k}\right) Q_1 Q_2^{\mathsf{T}}.
\end{cases} \tag{4.9}
$$

This variant entirely avoids inversion and is proven to be backward stable [43], but has a higher operation count than the Cholesky variant (4.8). This is why in practice the algorithm carries out the QR-based variant (4.9) in the first iterations and switches to the Cholesky variant (4.8) as soon as a reasonably conditioned iterate X_k is guaranteed. This way, numerical stability of the iteration is not compromised.

The two forms of the iteration represent the connection between the QR decomposition and the Cholesky factorization described in the previous section. They are two sides of the same coin. Either the QR decomposition of $A = \begin{bmatrix} \sqrt{c_k} X \\ I \end{bmatrix}$ is computed (leading to Iteration (4.9)), or the Cholesky factorization of $A^* A = I + c_k X^* X$ (Iteration (4.8)) is computed and used for a linear solve.

5. Generalized polar decompositions.

5.1. The generalized QDWH algorithm. Iterative methods for computing the generalized polar factor can be constructed from a connection to the matrix sign function.

THEOREM 5.1 (Computation of the canonical generalized polar decomposition, Theorem 5.1 in [29]). *Let $A = WS$ be a matrix with an existing canonical generalized polar decomposition with respect to the orthosymmetric pair M, N. Let*

$$
X_{k+1} = g(X_k) = X_k h(X_k^2) \tag{5.1}
$$

be an iteration that converges to $\mathrm{sign}(X_0)$*, assuming it exists. $g(\cdot)$ and $h(\cdot)$ are matrix functions. Let $g(0) = 0$ and for sesquilinear forms assume that $g(X^{\star_N}) = g(X)^{\star_N}$ holds for all X in the domain of g. Then the iteration*

$$
Y_{k+1} = Y_k h(Y_k^{\star_{M,N}} Y_k), \qquad Y_0 = A,
$$

converges to W with the same order of convergence as iteration (5.1) converges to $\mathrm{sign}(X_0)$.

Iterations for the matrix sign function of the form (5.1) are very common and well-studied [28, Ch. 5]. They include the class of Padé iterations devised in [32]. Here, the iteration is given as a rational function of the form

$$X_{k+1} = X_k p_{lm}(I - X_k^2) q_{lm}(I - X_k^2)^{-1}, \qquad X_0 = A,$$

where $p_{lm}(\cdot)$ and $q_{lm}(\cdot)$ are explicitly given polynomials, yielding the Padé approximant of degree (l, m).

Choosing $l = m = 1$ leads to the Halley iteration, which also forms the basis of the QDWH algorithm presented in Section 4. In the context of the generalized polar decomposition, the dynamically weighted Halley iteration follows from applying Theorem 5.1 and is given as

$$X_{k+1} = X_k(a_k I + b_k X_k^{\star_{M,N}} X_k)(I + c_k X_k^{\star_{M,N}} X_k)^{-1}, \quad X_0 = sA, \tag{5.2}$$

where $s \in \mathbb{R}$ is an arbitrary scaling factor, as any sA has the same polar factor W. A discussion on how to choose a beneficial s follows later. More explicitly, using (2.1), Iteration (5.2) is given as

$$X_{k+1} = X_k(a_k I + b_k N^{-1} X_k^* M X_k)(I + c_k N^{-1} X_k^* M X_k)^{-1}, \quad X_0 = sA.$$

The generalization of the DWH algorithm given in the previous paragraphs is straightforward. We now investigate whether this iteration has attractive numerical properties and under which circumstances it can lead to an accelerated convergence. The key observation in the standard setting is that one iteration step acts as a rational function on the singular values of the iterate X_k (see Equations (4.2) to (4.4)). A similar observation helps in the indefinite setting.

COROLLARY 5.2. *Let the canonical generalized polar decomposition $A = WS$ exist and be computed via an iteration $X_{k+1} = X_k h(X_k^{\star_{M,N}} X_k)$, $X_0 = A$, as given in Theorem 5.1. Then X_k has a canonical generalized polar decomposition*

$$X_k = WS_k.$$

The series of self-adjoint factors S_k satisfies

$$S_{k+1} = S_k h(S_k^2). \tag{5.3}$$

Proof. See proof of Theorem 5.1 in [30]. ∎

Using the Jordan canonical form $S = ZJZ^{-1}$, we see that (5.3) is equivalent to

$$S_{k+1} = Z g(J_k) Z^{-1} = Z J_{k+1} Z^{-1},$$

with $g(x) = xh(x^2)$. Essentially, one iteration step for computing the generalized polar decomposition acts as a rational function on the eigenvalues of the self-adjoint factor S, such that they converge towards 1 (or stay 0 in the rank-deficient case). Note that all non-zero eigenvalues of S have positive real part and $S = (A^{\star_{M,N}} A)^{\frac{1}{2}}$ by definition.

In the standard setting outlined in Section 4, i.e. the case $M = I_m$, $N = I_n$, the matrix S is symmetric (respectively Hermitian) and has only real eigenvalues. These eigenvalues are the singular values of A. This property does not hold in the general case. Only the convergence of the real eigenvalues of S is guaranteed to benefit from choosing the weighting parameters as in the standard case. Complex eigenvalues also

converge (see the numerical results of Example 2 in Section 7 in Table 1) but a concise description of the convergence behaviour is left as future research.

The reason we are interested in developing this method further, lies in its possible applications laid out in Section 1. In the application in quantum physics, the relevant eigenvalues are in fact often real. This follows from physical constraints and does not follow directly from the given matrix structure. To be more specific, ΣA is Hermitian and positive definite in this case. We call a matrix with this property a definite pseudosymmetric matrix. This property leads to A having only real eigevalues (see e.g. [6, Thm. 5]), such that the pseudosymmetric polar factor has only positive real eigenvalues. In this case, we expect great benefits from choosing the weighting parameters as in (4.5) and (4.6).

The scaling factor s in (5.2) should be chosen in the following way. Let $sA = WS_s$ be the generalized polar decomposition of $X_0 = sA$. The polar factor W is the same as for A. The pseudosymmetric factor $S_s = sS$ is the scaled pseudosymmetric factor of $A = WS$ and s should be chosen such that its eigenvalues lie between 0 and 1, i.e.

$$s \leq (\lambda_{\max}(S))^{-1} = (\lambda_{\max}((\Sigma A^* \Sigma A)^{\frac{1}{2}}))^{-1}. \tag{5.4}$$

The value ℓ should be a lower bound on the smallest eigenvalue of S_s, i.e.

$$\ell \leq \lambda_{\min}(S_s) = s\lambda_{\min}((\Sigma A^* \Sigma A)^{\frac{1}{2}}). \tag{5.5}$$

Computing values fulfilling (5.4) and (5.5) seems non-trivial, as computing S (after computing W via the iteration) is the goal of the algorithm and S is not known a-priori. The following lemma gives a remedy for square matrices.

LEMMA 5.3. *Let $A \in \mathbb{K}^{n\times n}$ and Q_1, Q_2 be unitary. Then*

$$|\lambda_{max}((Q_1 A^* Q_2 A)^{\frac{1}{2}})| \leq \sigma_{max}(A), \qquad |\lambda_{min}((Q_1 A^* Q_2 A)^{\frac{1}{2}})| \geq \sigma_{min}(A).$$

Proof. Because the spectral norm is submultiplicative, we have

$$\begin{aligned} |\lambda_{\max}(Q_1 A^* Q_2 A)| &\leq \sigma_{\max}(Q_1 A^* Q_2 A) \leq \sigma_{\max}(Q_1 A^* Q_2)\sigma_{\max}(A) = \sigma_{\max}(A)^2, \\ |\lambda_{\min}(Q_1 A^* Q_2 A)| &\geq \sigma_{\min}(Q_1 A^* Q_2 A) \geq \sigma_{\min}(Q_1 A^* Q_2)\sigma_{\min}(A) = \sigma_{\min}(A)^2. \end{aligned}$$

The lemma follows immediately. ∎

Lemma 5.3 for $Q_1 = Q_2 = \Sigma$ implies that s and ℓ_0 can be chosen as

$$s \approx 1/\sigma_{\max}(A), \qquad \ell_0 \approx s\sigma_{\min}(A) = 1/\mathrm{cond}_2(A) \tag{5.6}$$

in order to fulfill (5.4) and (5.5) in the case of square matrices.

Additionally to favorable convergence properties guaranteed for certain matrices, generalizing the ideas from QDWH leads to a new class of inverse-free iterations for computing the generalized polar factor. In the case of self-adjoint matrices, this polar factor coincides with the matrix sign function, which is relevant in many application areas. Avoiding the inverse opens up the possibility of more stable methods. How exactly this is done is described in the following.

Here, the role of the orthogonal representations in QDWH is played by (M, N)-orthogonal matrices defined via two inner products given by two matrices M and N. The following lemma provides a tool for substituting the inverse $(I + c_k X_k^{\star_{M,N}} X_k)^{-1}$ in Iteration (5.2).

LEMMA 5.4. *Let $M \in \mathbb{K}^{m\times m}$, $N \in \mathbb{K}^{n\times n}$ be nonsingular, and $M_2 := \begin{bmatrix} M & \\ & N \end{bmatrix}$. For $X \in \mathbb{K}^{m\times n}$, $\eta \in \mathbb{K}$, let $\begin{bmatrix} \eta X \\ I \end{bmatrix} = VR$ with $V = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \in \mathbb{K}^{(m+n)\times n}$, $R \in \mathbb{K}^{n\times n}$ nonsingular, be a decomposition. Then*

$$\eta X(I + |\eta|^2 X^{\star_{M,N}} X)^{-1} = V_1 (V^{\star_{M_2,N}} V)^{-1} V_2^{\star_N}.$$

Proof. We have

$$\begin{aligned}\eta X(I + |\eta|^2 X^{\star_{M,N}} X)^{-1} &= \eta X \left(\begin{bmatrix} \eta X \\ I \end{bmatrix}^{\star_{M_2,N}} \begin{bmatrix} \eta X \\ I \end{bmatrix} \right)^{-1} \\ &= V_1((VR)^{\star_{M_2,N}} V)^{-1} = V_1 (V^{\star_{M_2,N}} V)^{-1} V_2^{\star_N}.\end{aligned}$$

In the last step we used $V_2 = R^{-1}$. ∎

For $M = I_m$, $N = I_n$ and V with orthogonal or unitary columns, we have the known result

$$\eta X(I + |\eta|^2 X^* X)^{-1} = V_1 V_2^*,$$

given for example as Theorem 4.1 in [41]. The original QDWH algorithm is based on this result. A straightforward idea to generalize this approach would be to choose V to be (M_2, N)-orthogonal, i.e. $V^{\star_{M_2,N}} V = I$. The next lemma shows how we can relax this condition, while keeping the inverse easy to compute.

LEMMA 5.5. *Let $M \in \mathbb{K}^{m\times m}$ and $N \in \mathbb{K}^{n\times n}$ both be nonsingular, $M_2 = \begin{bmatrix} M & \\ & N \end{bmatrix}$, and $V = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \in \mathbb{K}^{(m+n)\times n}$ be $(M_2, \hat{N})$-orthogonal for a matrix $\hat{N} \in \mathbb{K}^{n\times n}$, i.e. $V^* M_2 V = \hat{N}$. Then*

$$V_1 (V^{\star_{M_2,N}} V)^{-1} V_2^{\star_N} = V_1 V_2^{\star_{N,\hat{N}}}.$$

Proof. From $V^* M_2 V = \hat{N}$, it follows that $V^{\star_{M_2,N}} V = N^{-1}\hat{N}$ and therefore

$$V_1 (V^{\star_{M_2,N}} V)^{-1} V_2^{\star_N} = V_1 \hat{N}^{-1} V_2^* N = V_1 V_2^{\star_{N,\hat{N}}}.$$ ∎

5.2. Realizing the ΣDWH iteration. When a practical method for computing the $(M_2, \hat{N})$-orthogonal matrices in Lemma 5.5 is available, we can formulate a generalized QDWH algorithm. If N^{-1} is trivial to compute, this leads to an inverse-free computation, if the computation of the $(M_2, \hat{N})$-orthogonal matrix avoids inversion. We now leave the general framework and restrict ourselves to inner products induced by signature matrices.

Section 3 laid the groundwork for several options in the algorithm design realizing the iteration for the canonical generalized polar decomposition of $A \in \mathbb{K}^{m\times n}$ (5.2) with respect to the signature matrices Σ_m and Σ_n. As signature matrices are involutory, the iteration is given as

$$X_{k+1} = X_k(a_k I + b_k \Sigma_n X_k^* \Sigma_m X_k)(I + c_k \Sigma_n X_k^* \Sigma_m X_k)^{-1}, \quad X_0 = sA. \tag{5.7}$$

We call (5.7) the ΣDWH iteration. The naive approach is to implement the iteration straightforward, using a linear solve employing the MATLAB backslash operator.

However, there is a better way to exploit the structure at hand. To see this, we rewrite (5.7) as

$$\begin{aligned} & X_k(a_k I + b_k \Sigma_n X_k^* \Sigma_m X_k)(I + c_k \Sigma_n X_k^* \Sigma_m X_k)^{-1} \\ = \quad & \frac{b_k}{c_k} X_k + (a_k - \frac{b_k}{c_k}) X_k (\Sigma_n + c_k X_k^* \Sigma_m X_k)^{-1} \Sigma_n. \end{aligned}$$

This is the indefinite analogue to (4.7). In the standard case, the Cholesky factorization is employed to exploit the symmetric structure in Iteration (4.8). In the indefinite case, this role is played by the pivoted LDL^T factorization. Analogous to (4.8), Iteration (5.7) is equivalently given as

$$\left\{ \begin{array}{l} Z_k = \Sigma_n + c_k X_k^* \Sigma_m X_k, \quad [L_k, D_k, P_k] = \mathrm{ldl}(Z_k), \\ X_{k+1} = \frac{b_k}{c_k} X_k + \left(a_k - \frac{b_k}{c_k}\right) X_k P_k L_k^{-*} D_k^{-1} L_k^{-1} P_k^\mathsf{T} \Sigma. \end{array} \right. \tag{5.8}$$

This approach is already more promising than the naive one because the structure of the involved matrices is exploited. This way, less computational work is needed and we may expect better accuracy. We employ Lemma 5.4 and Lemma 5.5 to find an equivalent formulation of the DWH iteration (5.7), which in principle does not rely on computing inverses. The role of $\hat{N}$ in Lemma 5.5 is played by another signature matrix $\hat{\Sigma}_n$ of size $n \times n$. The formulation

$$\left\{ \begin{array}{l} \begin{bmatrix} \sqrt{c_k} X_k \\ I \end{bmatrix} = \begin{bmatrix} H_1 \\ H_2 \end{bmatrix} R, \text{ where } \begin{bmatrix} H_1 \\ H_2 \end{bmatrix}^* \begin{bmatrix} \Sigma_m & \\ & \Sigma_n \end{bmatrix} \begin{bmatrix} H_1 \\ H_2 \end{bmatrix} = \hat{\Sigma}_n, \\ X_{k+1} = \frac{b_k}{c_k} X_k + \frac{1}{\sqrt{c_k}} \left(a_k - \frac{b_k}{c_k}\right) H_1 \hat{\Sigma}_n H_2^* \Sigma_n \end{array} \right. \tag{5.9}$$

is the analogue to the QR-based iteration (4.9) in the standard case. Instead of an orthogonal basis (using the QR decomposition), a $\left(\begin{bmatrix} \Sigma_m & \\ & \Sigma_n \end{bmatrix}, \hat{\Sigma}_n\right)$-orthogonal basis is computed. This can be done by computing the hyperbolic QR decomposition (Theorem 3.1) or the indefinite QR decomposition (Theorem 3.5). Here, methods exist that are based on successive column elimination and do not perform any matrix inversions. Computing the indefinite QR decomposition via an LDL^T factorization (i.e. employing Lemma 3.3) gives exactly the LDL^T based iteration (5.8).

In exact arithmetic, there would be no difference between the versions, but LDLIQR2 might be helpful in avoiding rounding errors in floating point arithmetic. The resulting stability for an iteration employing these different approaches is examined experimentally in the numerical experiments of Section 7.

6. Subspaces in the ΣDWH iteration.

6.1. Permuted graph bases for general matrices. Looking at Lemma 5.4, we see that the factor R of the VR decomposition is in fact not referenced in order to rewrite part of the DWH iteration. This suggests the idea to employ a well-conditioned basis of the subspace spanned by $\begin{bmatrix} \sqrt{c_k} X \\ I_n \end{bmatrix}$. The linear solve in one iteration step is not avoided completely but we hope to invert a better-conditioned matrix.

In the following we use $A \sim B$ to indicate that the columns of the two matrices A and B span the same subspace. A good candidate for providing a basis with desirable properties are permuted graph bases. An n-dimensional subspace $\mathcal{U}$ is said to be

represented in a *permuted graph basis*, determined by a permutation P and a matrix $X \in \mathbb{K}^{(m-n)\times n}$, if

$$\mathcal{U} = \operatorname{colspan}\left(P^{\mathsf{T}} \begin{bmatrix} I_n \\ X \end{bmatrix} \right). \tag{6.1}$$

It is shown in [40] that, for any $\mathcal{U}$, a permutation P exists, such that the entries of X are all smaller than 1. This leads to much better numerical properties when using this representation in numerical algorithms.

The actual computation of the entry-bound representations (6.1) is an NP-hard problem. However, in [40] heuristic methods are presented that compute representations, where, for a given threshold value $\tau > 1$, the matrix entries of X fulfill $|x_{i,j}| < \tau$. This can be done with a reasonable amount of computational effort, which in the worst case is $\mathcal{O}(n^3 \log_\tau n)$ [46]. In practice, it is typically much lower, in particular when good starting guesses for P are available.

The following lemma is a reformulation of Lemma 5.4, where $M = \Sigma_m$ and $N = \Sigma_n$ are signature matrices and V is attained via representation (6.1).

LEMMA 6.1. *Let $\Sigma_m \in \mathbb{R}^{m\times m}$, $\Sigma_n \in \mathbb{R}^{n\times n}$ be signature matrices. For $X \in \mathbb{K}^{n\times n}$, $\eta \in \mathbb{K}$ let $\begin{bmatrix} I \\ \eta X \end{bmatrix} \sim V = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = P^{\mathsf{T}} \begin{bmatrix} I \\ \hat{X} \end{bmatrix} \in \mathbb{K}^{2n\times n}$, where P is a permutation. Let*

$$P \begin{bmatrix} \Sigma_n & \\ & \Sigma_m \end{bmatrix} P^{\mathsf{T}} = \begin{bmatrix} \hat{\Sigma}_n & \\ & \hat{\Sigma}_m \end{bmatrix}.$$

Then

$$\eta X (I + |\eta|^2 \Sigma_n X^* \Sigma_m X)^{-1} = V_2 (\hat{\Sigma}_n + \hat{X}^* \hat{\Sigma}_m \hat{X})^{-1} V_1^* \Sigma_n.$$

Proof. Let $\Sigma_2 := \begin{bmatrix} \Sigma_n & \\ & \Sigma_m \end{bmatrix}$. We follow the lines of the proof of Lemma 5.4. As $\begin{bmatrix} I \\ \eta X \end{bmatrix}$ and V span the same subspace, there exists a nonsingular matrix R such that

$$\begin{bmatrix} I \\ \eta X \end{bmatrix} = V R.$$

Exactly as in the proof of Lemma 5.4 (with the roles of V_1 and V_2 switched) it can be shown that

$$\begin{aligned} \eta X (I + |\eta|^2 \Sigma_n X^* \Sigma_m X)^{-1} &= V_2 (V^{\star_{\Sigma_2,\Sigma_n}} V)^{-1} V_1^{\star_{\Sigma_n}} \\ &= V_2 (\hat{\Sigma}_n + \hat{X}^* \hat{\Sigma}_m \hat{X})^{-1} V_1^* \Sigma_n. \end{aligned}$$

∎

Algorithm 6.1 presents the details on how permuted graph bases can be used in the computation of generalized polar decomposition via the dynamically weighted Halley iteration.

Lemma 6.1 states that in exact arithmetic the iterates computed by Algorithm 6.1 are the same as in the original iteration (5.7). As with the iteration employing LDLIQR2, the algorithm is an alternative attempt to improve floating point accuracy.

Algorithm 6.1 Compute the generalized polar decomposition with respect to signature matrices, using permuted graph bases.

Input: $A \in \mathbb{K}^{m\times n}$,
$\Sigma_m \in \mathbb{R}^{m\times m}$ $\Sigma_n \in \mathbb{R}^{n\times n}$: signature matrices, s.t. the canonical generalized polar decomposition of A exists (according to Theorem 5.1),
s: estimate on $|\lambda_{\max}((\Sigma_n A^*\Sigma_m A)^{\frac{1}{2}})|^{-1}$,
ℓ: estimate on $s|\lambda_{\min}(\Sigma_n A^*\Sigma_m A)^{\frac{1}{2}}|$,
$\tau > 1$: threshold value for permuted graph basis.

Output: $A = WS$ is the canonical generalized polar decomposition with respect to Σ_m and Σ_n.

1: $W \leftarrow sA$.
2: **for** $k = 1, 2, \ldots$ **do**
3: Compute weighting parameters a, b, c and update ℓ from equations (4.5) and (4.6).
4: Compute entry-bound permuted graph bases of $\operatorname{colspan}\left(\begin{bmatrix} I \\ \sqrt{c}W \end{bmatrix}\right)$, i.e.

$$\begin{bmatrix} I \\ \sqrt{c}W \end{bmatrix} \sim P^{\mathsf{T}} \begin{bmatrix} I \\ \hat{W} \end{bmatrix} =: \begin{bmatrix} V_1 \\ V_2 \end{bmatrix},$$
$$|\hat{W}_{ij}| < \tau \text{ for } i \in \{1, \ldots, m\},\ j \in \{1, \ldots, n\}.$$

5: $\begin{bmatrix} \hat{\Sigma}_n & \\ & \hat{\Sigma}_m \end{bmatrix} \leftarrow P \begin{bmatrix} \Sigma_n & \\ & \Sigma_m \end{bmatrix} P^{\mathsf{T}}$
6: Compute LDL^{T} factorization $\hat{\Sigma}_n + \hat{W}^*\hat{\Sigma}_m\hat{W} = PLDL^*P^{\mathsf{T}}$.
7: $W \leftarrow \frac{b}{c}W + (a - \frac{b}{c})V_2 PL^{-*}D^{-1}L^{-1}P^{\mathsf{T}}V_1^*\Sigma_n$
8: **end for**
9: Compute pseudosymmetric factor and ensure pseudosymmetry numerically
$S \leftarrow \Sigma_n W^*\Sigma_m A, \qquad S \leftarrow (S + \Sigma_n S^*\Sigma_n)/2.$

6.2. Permuted Lagrangian graph bases for pseudosymmetric matrices. As pointed out in Section 1, we are in particular interested in computing the generalized polar decomposition (with respect to a signature matrix) of pseudosymmetric matrices. A way to exploit this structure in the iteration can be found by considering Lagrangian subspaces, to which pseudosymmetric matrices can be linked.

A subspace $\mathcal{U} = \operatorname{colspan}(U)$, $U \in \mathbb{K}^{2n\times n}$, is called *Lagrangian* if $U^*JU = 0$, where $J = \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix}$.

A Lagrangian subspace can be represented by a *permuted Lagrangian graph basis*

$$\mathcal{U} = \operatorname{colspan}\left(\Pi^{\mathsf{T}} \begin{bmatrix} I \\ X \end{bmatrix}\right), \tag{6.2}$$

where $X = X^*$. Π denotes a symplectic swap matrix [4]. A symplectic swap matrix is defined by a Boolean vector $v \in \{0,1\}^n$ and its complement $\hat{v} \in \{0,1\}^n$, where $\hat{v}_i = 1 - v_i$, as

$$\Pi_v = \begin{bmatrix} \operatorname{diag}(v) & \operatorname{diag}(\hat{v}) \\ -\operatorname{diag}(\hat{v}) & \operatorname{diag}(v) \end{bmatrix}. \tag{6.3}$$

It is shown in [40] that each Lagrangian subspace admits a representation (6.2), where X has no entries with modulus larger than $\sqrt{2}$.

As for general subspaces, there exist heuristics for computing a basis, such that the entries of X are bounded, within a reasonable amount of time. In this case $|x_{i,j}| < \tau$, where $\tau > \sqrt{2}$ is a given threshold value.

A Lagrangian subspace could of course be treated as a general subspace and admits a representation (6.1), with even smaller entries than in (6.2). However, the structural property, i.e. the subspace being Lagrangian, is not encoded anymore in this representation. It is encoded in the symmetry of X, which can easily be enforced and preserved in the course of computations. This has numerical benefits, which typically outweigh the slightly larger entries in X.

The following lemma draws a connection between self-adjoint matrices and Lagrangian subspaces.

LEMMA 6.2. *Let $M \in \mathbb{K}^{n\times n}$, $M = M^*$ be a nonsingular matrix. Let $X \in \mathbb{K}^{n\times n}$ be self-adjoint with respect to the inner product induced by M. Then $\begin{bmatrix} M \\ X \end{bmatrix}$ spans a Lagrangian subspace.*

The following lemma is a variant of Lemma 5.4 applied to square matrices, where the positions of the two matrix blocks are switched. The goal is to get to a formulation, in which the subspace given in Lemma 6.2 appears.

LEMMA 6.3. *Let $M, N \in \mathbb{K}^{n\times n}$ be nonsingular, N be M-orthogonal, i.e. $N^{\star_M} N = I$. $M_2 := \begin{bmatrix} M & \\ & M \end{bmatrix}$ and $X \in \mathbb{K}^{n\times n}$. Let $\begin{bmatrix} N \\ \eta X \end{bmatrix} = VR$ with $V = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \in \mathbb{K}^{2n\times n}$, $R \in \mathbb{K}^{n\times n}$ nonsingular be a decomposition. Then*

$$\eta X(I + |\eta|^2 X^{\star_M} X)^{-1} = V_2 (V^{\star_{M_2,M}} V)^{-1} V_1^{\star_M} N.$$

Proof. We observe

$$\begin{bmatrix} N \\ \eta X \end{bmatrix}^{\star_{M_2,M}} \begin{bmatrix} N \\ \eta X \end{bmatrix} = N^{\star_M} N + |\eta|^2 X^{\star_M} X = I + |\eta|^2 X^{\star_M} X.$$

Following the proof of Lemma 5.4, we get

$$\eta X(I + |\eta|^2 X^{\star_M} X)^{-1} = V_2 (V^{\star_{M_2,M}} V)^{-1} (R^{-1})^{\star_M} = V_2 (V^{\star_{M_2,M}} V)^{-1} V_1^{\star_M} N.$$

In the last step we used $R^{-1} = N^{-1} V_1 = N^{\star_M} V_1$. ∎

Let us go back to the specific case of an inner product induced by a signature matrix, i.e. $M := \Sigma$. In this case, Lemma 6.2 and Lemma 6.3 come together. Σ is symmetric, so Lemma 6.2 holds. So does Lemma 6.3 by setting $N := \Sigma$. The subspace in question can be represented by permuted Lagrangian graph bases. The situation is summarized in the following lemma.

LEMMA 6.4. *Let $\Sigma \in \mathbb{R}^{n\times n}$ be a signature matrix. Let $\Sigma_2 := \begin{bmatrix} \Sigma & \\ & \Sigma \end{bmatrix}$, and let $X \in \mathbb{K}^{n\times n}$ be self-adjoint with respect to the inner product induced by Σ and $\eta \in \mathbb{K}$. Let*

$$\begin{bmatrix} \Sigma \\ \eta X \end{bmatrix} \sim \Pi^{\mathsf{T}} \begin{bmatrix} I \\ \hat{X} \end{bmatrix} =: V = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix}$$

be a permuted Lagrangian graph basis, i.e. Π *is a symplectic swap matrix and* $\hat{X} = \hat{X}^*$. *Then*

$$\eta X(I + |\eta|^2 \Sigma X^* \Sigma X)^{-1} = V_2(\Sigma + \hat{X}\Sigma\hat{X})^{-1}V_1^*.$$

Proof. Note that

$$V^{\star_{\Sigma_2,\Sigma}}V = \Sigma \begin{bmatrix} I_{2n} & \hat{X}^\mathsf{T} \end{bmatrix} \Pi\Sigma_2\Pi^\mathsf{T} \begin{bmatrix} I_{2n} \\ \hat{X} \end{bmatrix} = I_{2n} + \Sigma\hat{X}^*\Sigma\hat{X} = I_{2n} + \Sigma\hat{X}\Sigma\hat{X}.$$

We have used $\Pi\Sigma_2\Pi^\mathsf{T} = \Sigma_2$, which holds because $\Pi = \begin{bmatrix} V & \hat{V} \\ -\hat{V} & V \end{bmatrix}$ is a symplectic swap matrix as given in (6.3):

$$\Pi\Sigma_2\Pi^\mathsf{T} = \begin{bmatrix} V & \hat{V} \\ -\hat{V} & V \end{bmatrix} \begin{bmatrix} \Sigma & \\ & \Sigma \end{bmatrix} \begin{bmatrix} V & \hat{V} \\ -\hat{V} & V \end{bmatrix}^\mathsf{T} = \begin{bmatrix} V\Sigma V + \hat{V}\Sigma\hat{V} & -V\Sigma\hat{V} + \hat{V}\Sigma V \\ -\hat{V}\Sigma V + V\Sigma\hat{V} & \hat{V}\Sigma\hat{V} + V\Sigma V \end{bmatrix} = \Sigma_2.$$

The equalities $V\Sigma V + \hat{V}\Sigma\hat{V} = \Sigma$ and $-V\Sigma\hat{V} + \hat{V}\Sigma V = 0$ hold because V and $\hat{V}$ pick up complementing rows and columns of Σ. Now, applying Lemma 6.3 gives

$$\eta X(I + |\eta|^2 X^{\star_\Sigma} X)^{-1} = V_2(\Sigma + \hat{X}^*\Sigma\hat{X})^{-1}V_1^*. \qquad \square$$

Algorithm 6.2 is a variant of Algorithm 6.1 using permuted Lagrangian graph bases. It computes the generalized polar decomposition of a pseudosymmetric matrix with respect to its defining signature matrix.

In the update step (Step 7 in Algorithm 6.1 and Step 6 in Algorithm 6.2), the structure of V_1 and V_2 should be taken into account for an efficient implementation. The rows of the identity matrix are distributed in V_1 and V_2 according to the permutation P or the symplectic swap Π. The remaining columns are given by $\hat{W}$. If this is taken care of, the matrix representing the subspace $V = \Pi^\mathsf{T} \begin{bmatrix} I \\ \hat{U} \end{bmatrix}$ never has to actually be formed. We can directly work on the matrices W and $\hat{W}$.

However, we may need to form a $n \times 2n$ matrix if a good starting guess for the permutation in the computation of the permuted graph basis is desired. For this task, a heuristic is proposed in [40] that includes a modified version of the QR factorization with column pivoting of an $n \times 2n$ matrix.

7. Numerical results. In this paper, we have developed several variants of the ΣDWH iteration to compute the canonical generalized polar decomposition of a matrix with respect to signature matrices.

In general, the existence of the decomposition is not guaranteed, which is why we first examine pseudosymmetric matrices with respect to Σ. For these matrices, the generalized polar decomposition exists if and only if A has no purely imaginary eigenvalues (note that this is also required for $\operatorname{sign}(A)$ to exist). For randomly generated matrices this is typically the case, which is why we observe convergence most times. Pseudosymmetric matrices represent an important class of matrices regarding the application potential of the developed methods, as pointed out in Section 1. For other matrices, which are not pseudosymmetric but yield a generalized polar decomposition with respect to Σ, similar results were observed in further tests. All experiments were performed in MATLAB R2017a.

Algorithm 6.2 Compute the generalized polar decomposition of a pseudosymmetric matrix with respect to a signature matrix, using permuted Lagrangian graph bases.

Input: Signature matrix $\Sigma \in \mathbb{K}^{n\times n}$, $A = \Sigma A^*\Sigma \in \mathbb{K}^{n\times n}$, s.t. A has no purely imaginary eigenvalues,
s: estimate on $|\lambda_{\max}((\Sigma A^*\Sigma A)^{\frac{1}{2}})|^{-1}$,
ℓ: estimate on the norm of the smallest eigenvalue of $s(\Sigma A^*\Sigma A)^{\frac{1}{2}}$,
$\tau > \sqrt{2}$: threshold value for permuted Lagrangian graph bases.

Output: $A = WS$ is the generalized polar decomposition with respect to Σ.

1: $W \leftarrow sA$.
2: **for** $k = 1, 2, \ldots$ **do**
3: Compute weighting parameters a, b, c and update ℓ from equations (4.5) and (4.6).
4: Compute entry-bound permuted Lagrangian graph bases of $\operatorname{colspan}\left(\begin{bmatrix}\Sigma\\ \sqrt{c}W\end{bmatrix}\right)$, i.e.

$$\begin{bmatrix}\Sigma\\ \sqrt{c}W\end{bmatrix} \sim \Pi^{\mathsf{T}}\begin{bmatrix}I\\ \hat{W}\end{bmatrix} =: \begin{bmatrix}V_1\\ V_2\end{bmatrix}, \quad |\hat{W}_{ij}| < \tau \text{ for } i,j \in \{1,\ldots,n\}.$$

5: Compute LDL^{T} factorization $\Sigma + \hat{W}^*\Sigma\hat{W} = PLDL^*P^{\mathsf{T}}$.
6: $W \leftarrow \frac{b}{c}W + (a - \frac{b}{c})V_2PL^{-*}D^{-1}L^{-1}P^{\mathsf{T}}V_1^*$
7: **end for**
8: Compute pseudosymmetric factor and ensure pseudosymmetry numerically
$S \leftarrow \Sigma W^*\Sigma A, \qquad S \leftarrow (S + \Sigma S^*\Sigma)/2.$

In light of the asymptotic cubic convergence of the iteration (see [28, Sec. 4.9.2]) we use the stopping criterion

$$\|X_{k+1} - X_k\|_F \leq (5\epsilon)^{\frac{1}{3}}, \tag{7.1}$$

where ϵ is the machine precision.

We take the same values for s and ℓ as in the QDWH algorithm [41], which are given in (5.6). As explained there, this makes sense for definite pseudosymmmetric matrices. The resulting convergence behavior is the same as in the standard setting. Further investigation of the convergence behavior is needed to devise sensible values for s and ℓ in the general case. Here, the iteration may act on complex values. This consideration goes beyond the scope of this paper, and we use the same values as in the definite case even when they are not completely justified.

We first compare the algorithms in terms of their achieved residual for badly conditioned matrices. We consider square matrices and their generalized polar decomposition for a given signature matrix $\Sigma := \begin{bmatrix}I_n & \\ & -I_n\end{bmatrix}$ ($M = N = \Sigma$ in Definition 2.1).

Example 1. A real pseudosymmetric matrix with a condition number $\kappa = 10^k$ is generated as $A := \Sigma Q D Q^{\mathsf{T}}$. The matrix Q is orthogonal and generated randomly (`orth(rand(2*n))`), and D is a diagonal matrix containing equally distributed values between 1 and 10^k, with alternating signs. A polar decomposition $A \approx WS$ is computed and the resulting residual $\|WS - A\|_F/\|A\|_F$ for matrices of size 200×200 ($n = 100$) is given in Figure 1. The residuals were averaged over 10 runs with different

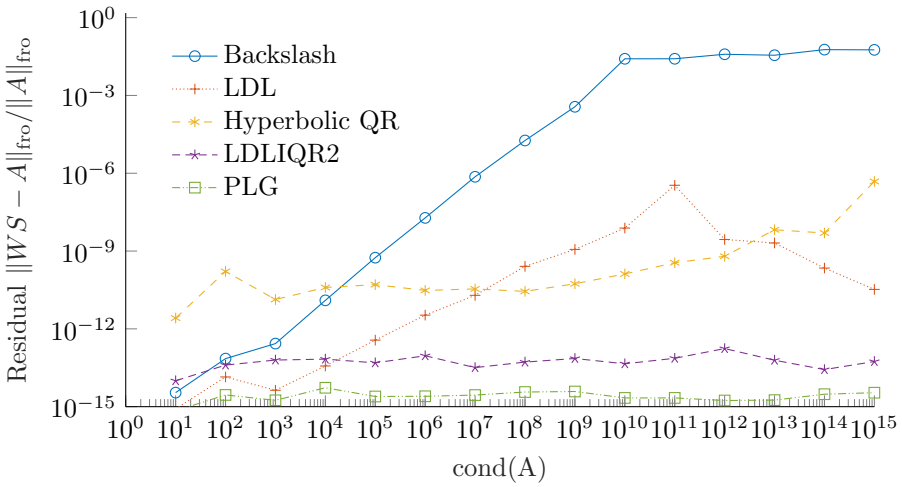

FIG. 1. *Residuals for different iterations for computing the generalized polar decomposition of pseudosymmetric matrices $A \in \mathbb{R}^{200\times 200}$ with a certain condition number. "Backslash" refers to the naive implementation, "LDL" refers to iteration (5.8), "Hyperbolic QR" and "LDLIQR2" refer to the variants of iteration 5.9. "PLG" refers to the variant using permuted Lagrangian graph bases described in Algorithm 6.2.*

randomly generated matrices.

We see that a naive implementation of the ΣDWH iteration (5.7) leads to a highly unstable method. The accuracy improves as the iteration is rewritten to employ the LDL^{T} decomposition (see (5.8)). This can be interpreted as exploiting structure that is hidden and ignored in the original formulation. Again, the accuracy deteriorates as the matrix becomes ill-conditioned. Surprisingly, for matrices with a condition number higher than 10^{11}, this trend is reversed and the method performs quite well for extremely ill-conditioned matrices. A possible explanation is that MATLAB function `ldl` estimates the condition number of the input and acts differently, in our case preferably, for ill-conditioned matrices. The LDL^{T}-based iteration can be read as an iteration based on the indefinite QR decomposition (see Theorem 3.5 and Iteration (5.9)), that has been computed via the pivoted LDL^{T} decomposition. For computing a hyperbolic QR decomposition directly, using a column elimination approach, we used available MATLAB code [31], based on the works [1, 21, 26]. In our setting, this does not perform well. For well-conditioned matrices, this approach delivers the worst accuracy. For ill-conditioned matrices it yields better results than the naive implementation, but is still highly dependent on the condition number. The two remaining methods use the indefinite QR decomposition via a double LDL^{T} decomposition (LDLIQR2) and permuted Lagrangian graph bases (PLG). These give high accuracy, which is independent of the condition number. For well-conditioned matrices, LDLIQR2 does not seem to be preferable, as it yields a higher residual than even the naive implementation. However, the residual stays at a consistently low order of magnitude as the condition number increases. Using PLGs consistently delivers the best results regarding accuracy, in the well-conditioned as well as in the ill-conditioned setting. The disadvantage of the PLG approach is that it relies on very recently developed, fine-grained algorithms. Therefore, no optimized implementations are available yet and the runtimes resulting from a prototype MATLAB implementation are very

high. Formulating the computation of PLGs in a way that exploits current computer architectures is a challenge not yet addressed. This method would need to be block-based in order to exploit the memory hierarchy, be parallelizable and avoid communication. The LDLIQR2 approach on the other side is easily implemented and only relies on the LDL^{T} factorization for which highly optimized implementations are available. However, both approaches rely on pivoting strategies, implying a considerable cost for communication if they are to be deployed in a massively parallel setup.

In a practical implementation, a combination of the LDL^{T}, LDLIQR2 and PLG approach should be considered, as it is possible for each iteration step to be performed by a different method. For badly conditioned matrices, the first steps could be performed via PLG. As soon as the condition number of the iterate has improved, another method could be employed, which shows better performance.

We now compare the developed algorithms with other available methods, in particular concerning convergence properties. A standard approach for computing (generalized) polar decompositions is the scaled Newton iteration (see e.g. [28]). For a given signature matrix Σ, it is given as

$$X_{k+1} = \frac{1}{2}(\mu_k X_k + \mu_k^{-1}\Sigma X_k^{-*}\Sigma), \qquad X_0 = A. \tag{7.2}$$

It is called the Newton iteration as it represents the Newton method for solving $A^\star A = I$. See also [27] for details. For the DWH iteration, we have shown in Corollary 5.2 that the iteration acts as a matrix sign function iteration on the self-adjoint factor of the decomposition. This observation also holds for the Newton iteration. Let $X_k = WS_k$ be a generalized polar decomposition of the iterate, then (7.2) is equivalent to

$$X_{k+1} = W\left(\frac{1}{2}(\mu_k S_k + \mu_k^{-1}S_k^{-1})\right), \qquad X_0 = A.$$

The part in large parentheses is the Newton iteration for the matrix sign function acting on S_k. In the standard setting, the self-adjoint factor is Hermitian and its eigenvalues are real. This is exploited to devise scaled iterations which drive these values closer to one and therefore accelerate convergence (see [28, 19, 41]). For the generalized polar decomposition, the values are not necessarily real. In this case, we can fall back on scaling strategies for the matrix sign function which show good convergence properties in practice. In particular, we consider determinantal scaling [18], where

$$\mu_k := |\det S_k|^{-\frac{1}{n}} = |\det X_k|^{-\frac{1}{n}}.$$

The computation via the iterate X_k becomes possible because signature matrices and automorphisms with respect to them have a determinant of ± 1. Its computation is cheap as it can be computed from the diagonal values of the LU factorization, which is used to compute X_k^{-*}. For the next numerical example, we generate matrices for which the generalized polar decomposition with respect to Σ is guaranteed to exist, but where the eigenvalues of the self-adjoint factor are all complex.

Example 2. For the generalized polar decomposition $A = WS$, we prescribe the self-adjoint factor S with a condition number $\kappa = 10^k$. The absolute values r_j of the eigenvalues $\lambda_j = r_j \exp(\mathrm{i}\phi_j)$ of H are uniformly distributed between $10^{-\lfloor k/2 \rfloor}$ and $10^{\lceil k/2 \rceil}$. The angles ϕ_j are uniformly distributed between $-\pi/2$ and $\pi/2$, i.e. all

eigenvalues lie in the right half plane. S is generated using two random orthogonal matrices $Q_1, Q_2 \in \mathbb{R}^{n\times n}$, forming $Q = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix}$, as

$$S := Q^{\mathsf{T}} \begin{bmatrix} \mathrm{Re}(\lambda_1) & & & -\mathrm{Im}(\lambda_1) & & \\ & \ddots & & & \ddots & \\ & & \mathrm{Re}(\lambda_n) & & & -\mathrm{Im}(\lambda_n) \\ \mathrm{Im}(\lambda_1) & & & \mathrm{Re}(\lambda_1) & & \\ & \ddots & & & \ddots & \\ & & \mathrm{Im}(\lambda_n) & & & \mathrm{Re}(\lambda_n) \end{bmatrix} Q.$$

The polar factor W is prescribed as

$$W := \begin{bmatrix} Q_3 & \\ & Q_4 \end{bmatrix} \begin{bmatrix} C_W & S_W \\ S_W & C_W \end{bmatrix},$$

where Q_3 and Q_4 are random orthogonal matrices. The matrix $\begin{bmatrix} C_W & S_W \\ S_W & C_W \end{bmatrix}$ describes a series of hyperbolic Givens rotations, i.e.

$$C_W = \mathrm{diag}(\cosh\omega_1, \ldots, \cosh\omega_{2n}), \quad S_W = \mathrm{diag}(\sinh\omega_1, \ldots, \sinh\omega_{2n}),$$

where $\omega_1, \ldots, \omega_{2n}$ are uniformly distributed angles between 0 and $\frac{1}{4}\pi$. Averaged results for 20 matrices of size 200×200 ($n = 100$) are given in Table 1.

For the Newton iteration, we use the stopping criterion given in [28], Chapter 8:

$$\|X_{k++1} - X_k\|_F \leqq (2\epsilon)^{\frac{1}{2}}, \tag{7.3}$$

where ϵ denotes the machine precision.

For the ΣDWH iteration, we employ permuted graph bases (Algorithm 6.1), available in the `pgdoubling` package associated with [40]. It is compared to the Newton iteration with determinantal scaling (DN) and the Newton iteration with sub-optimal scaling [19] (SON). We generate 20 different random matrices and report the average number of iterations and the resulting residual $\|A - \tilde{W}\tilde{S}\|_F / \|A\|_F$, where $\tilde{W}$ and $\tilde{S}$ are the computed polar factors. We influence the condition number of A indirectly via $\kappa = \mathrm{cond}(S)$. It is about twice as high as κ because of the used hyperbolic Givens rotations.

In the standard setting, DWH and SON converge in 6 [41], respectively 9 [19], steps. Here, the iterations act as scalar iterations on the eigenvalues of the self-adjoint factor, which happen to be real in the standard case, but not in the indefinite setting. Still, we can observe that they converge significantly faster than the Newton iteration with determinantal scaling, in particular for ill-conditioned matrices. ΣDWH generally seems to need about 2/3 as many iteration steps as SON. Whether the cost per iteration is comparable, depends on the chosen implementation method for the DWH iteration. The simplest method is based on one LDL^{T} decomposition (5.8) and the main cost is a symmetric matrix inversion, just as in the Newton variants. If higher stability is needed in the case of badly conditioned matrices, it can be obtained at the expense of a higher costs per iteration. This can be done by employing the LDLIQR2 iteration or by improving the corresponding subspace via Lagrangian graph bases (Algorithm 6.1).

TABLE 1

Convergence behavior for different methods computing the generalized polar decomposition with respect to Σ of a 200×200 matrix (Example 2).

	κ	10	10^5	10^{10}	10^{15}
	cond(A)	2.15e+01	1.98e+05	1.98e+10	2.02e+15
# iterations	ΣDWH	8.70	9.70	10.65	10.60
	DN	12.30	20.00	32.95	44.13 [1]
	SON	14.05	15.45	16.45	16.74 [2]
residual	ΣDWH	5.06e-15	7.68e-15	9.88e-15	3.00e-15
	DN	2.98e-15	2.98e-15	2.93e-15	2.96e-15
	SON	3.00e-15	2.98e-15	2.96e-15	2.89e-15
rel. error W	ΣDWH	1.35e-14	9.45e-12	5.35e-08	8.01e-03
	DN	1.18e-14	3.96e-11	1.53e-06	7.83e-02
	SON	1.32e-14	3.12e-11	2.24e-07	5.22e-03
rel. error S	ΣDWH	1.05e-14	2.76e-14	3.51e-14	4.51e-14
	DN	9.98e-15	9.00e-12	8.52e-07	6.65e-02
	SON	1.43e-14	2.42e-14	2.33e-14	2.83e-14
$\|\Sigma W^\mathsf{T}\Sigma W - I\|_F$	ΣDWH	1.16e-15	1.23e-15	1.07e-15	1.25e-15
	DN	3.19e-15	3.19e-15	3.13e-15	3.12e-15
	SON	3.21e-15	3.18e-15	3.16e-15	3.09e-15

[1] 5 out of 20 runs did not converge. [2] 1 out of 20 runs did not converge.

ΣDWH displays the lowest backward error for the Σ-orthogonal factor W, which deteriorates for all methods as matrices become ill-conditioned. All methods yield a factor W that shows a good Σ-orthogonality. SON and ΣDWH both do a much better job than DN at recovering the self-adjoint factor S with backward errors of order 10^{-14} instead of 10^{-2}. DN and SON sometimes fail to converge for badly conditioned matrices.

We see that ΣDWH can compete with standard methods, even if no definite pseudosymmetric structure is given. Note that ΣDWH is the only one of the three methods that can directly be applied to non-square matrices, in order to compute the canonical generalized polar decomposition.

The results of Example 2 should be seen as preliminary, as the scaling factors and the stopping criterion (7.1) are not completely justified in the non-definite case. They do, however, motivate further research to devise iterations based on rational functions acting on complex values.

Example 3. We generate pseudosymmetric matrices as in Example 1, but additionally ensure the definiteness of ΣA by choosing only positive values for D. We compare the same methods as in Example 2 with respect to convergence properties. 20 matrices were generated and averaged results are reported in Table 2.

As expected, we see the convergence of ΣDWH and of the Newton iteration with suboptimal scaling within 6, respectively 9, iterations.

8. Conclusions. In this paper, we have presented a generalization of the QDWH method to compute the canonical generalized polar decomposition of a matrix with respect to a signature matrix Σ. If Σ is chosen as the identity, the hyperbolic QR decomposition becomes the standard QR decomposition and can safely be computed

TABLE 2

Convergence behavior for different methods computing the generalized polar decomposition with respect to Σ of definite pseudosymmetric matrices of size 200×200 (Example 3).

	κ	10	10^5	10^{10}	10^{15}
# iterations	ΣDWH	4.00	5.00	6.00	6.00
	DN	6.00	15.10	30.50	44.50
	SON	6.00	7.00	8.00	9.00
residual	ΣDWH	1.38e-15	4.47e-14	2.34e-14	2.85e-14
	DN	8.11e-16	2.46e-14	5.30e-14	1.05e-14
	SON	8.14e-16	3.20e-14	3.03e-14	1.04e-14
$\|\Sigma W^\mathsf{T}\Sigma W - I\|_F$	ΣDWH	1.26e-15	1.95e-13	2.03e-13	6.92e-14
	DN	7.31e-16	6.87e-14	5.66e-14	3.13e-14
	SON	7.16e-16	6.94e-14	5.64e-14	3.09e-14

with the column elimination approach. This yields the well-known QDWH iteration.

Several options were provided on how to realize the iterations. While the column elimination based hyperbolic QR decomposition forms the most natural generalization of QDWH, it does not yield the best results regarding stability. LDLIQR2 (Section 3.2) or employing permuted (Lagrangian) graph bases (Algorithm 6.1 and 6.2) perform better in this regard.

Using these variants, a stability similar to Newton methods can be observed, but fewer iterations are needed. For the important class of definite pseudosymmetric matrices, the convergence behavior corresponds to the standard QDWH method. Convergence up to machine precision can be guaranteed in 6 steps for reasonably conditioned matrices.

The theoretical results we gave, in particular Lemma 5.4, provide a greater flexibility in the algorithmic design for DWH-based iterations, which might be utilized further than the scope of this paper permits. Other methods for computing well-conditioned bases could also yield good results. Being more flexible in algorithmic design becomes increasingly important in view of modern computer architectures. In general these become more heterogeneous. They employ different levels of parallelism on various scales and have restrictions on available memory or use numerous accelerators and GPUs. Our framework provides the flexibility to find solutions, which could exploit the architecture at hand to its full potential.

Our main motivation came from computing the matrix sign function of large definite pseudosymmetric matrices. Here, the iteration acts as a rational function on what can be understood as generalized singular values. Hence, further developments using ideas from [42] are possible. Using Zolotarev's functions as best-approximations to the sign function of higher degree, yields an iteration that converges in two steps. The individual steps take more work but are embarrassingly parallel and well-suited for large-scale high performance computations. In the field of computational quantum physics this is exactly what is needed, making this research direction promising.

For matrices which are not definite, our numerical experiments have shown that the proposed methods still converge but no theoretical convergence results are given yet. Further research in this direction may close this gap.

Computing the hyperbolic QR decomposition is useful in many applications, which could benefit from the analysis given in Section 3. In particular the LDLIQR2

method (Algorithm 3.1) is a promising technique to tackle problems associated with the stability of the hyperbolic or indefinte QR decomposition.

REFERENCES

[1] E. Anderson. Discontinuous plane rotations and the symmetric eigenvalue problem. LAPACK Working Note 150, 2000. URL: http://www.netlib.org/lapack/lawnspdf/lawn150.pdf.

[2] C. Ashcraft, R. G. Grimes, and J. G. Lewis. Accurate symmetric indefinite linear equation solvers. *SIAM J. Matrix Anal. Appl.*, 20(2):513–561, 1999. doi:10.1137/S0895479896296921.

[3] Zhaojun Bai and James W. Demmel. Design of a parallel nonsymmetric eigenroutine toolbox, part i. Technical Report UCB/CSD-92-718, EECS Department, University of California, Berkeley, Feb 1993. URL: http://www2.eecs.berkeley.edu/Pubs/TechRpts/1993/6014.html.

[4] P. Benner. Symplectic balancing of Hamiltonian matrices. *SIAM J. Sci. Comput.*, 22(5):1885–1904, 2001. doi:10.1137/S1064827500367993.

[5] P. Benner, V. Khoromskaia, and B: N. Khoromskij. A reduced basis approach for calculation of the Bethe-Salpeter excitation energies using low-rank tensor factorizations. *Mol. Phys.*, 114(7–8):1148–1161, 2016. doi:10.1080/00268976.2016.1149241.

[6] P. Benner and C. Penke. Efficient and accurate algorithms for solving the Bethe-Salpeter eigenvalue problem for crystalline systems. *J. Comput. Appl. Math.*, 2021. URL: 10.1016/j.cam.2021.113650.

[7] P. Benner and C. Penke. GR decompositions and their relations to Cholesky-like factorizations. *Proc. Appl. Math. Mech.*, 20(1):e202000065, 2021. doi:10.1002/pamm.202000065.

[8] P. Benner and E. S. Quintana-Ortí. Solving stable generalized Lyapunov equations with the matrix sign function. *Numer. Algorithms*, 20(1):75–100, 1999. doi:10.1023/A:1019191431273.

[9] X. Blase, I. Duchemin, D. Jacquemin, and P.-F. Loos. The Bethe-Salpeter equation formalism: From physics to chemistry. *J. Phys. Chem. Lett.*, 11(17):7371–7382, 2020. doi:10.1021/acs.jpclett.0c01875.

[10] A. Bojanczyk, N. J. Higham, and H. Patel. Solving the indefinite least squares problem by hyperbolic QR factorization. *SIAM J. Matrix Anal. Appl.*, 24(4):914–931, 2003. doi:10.1137/S0895479802401497.

[11] Y. Bolshakov and B. Reichstein. Unitary equivalence in an indefinite scalar product: an analogue of singular-value decomposition. *Linear Algebra Appl.*, 222:155–226, 1995. doi:10.1016/0024-3795(93)00295-B.

[12] Y. Bolshakov, C. V. M. van der Mee, A. C. M. Ran, B. Reichstein, and L. Rodman. Polar decompositions in finite-dimensional indefinite scalar product spaces: general theory. *Linear Algebra Appl.*, 261:91–141, 1997. doi:10.1016/S0024-3795(96)00317-5.

[13] I. Borg and P. J. F. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer-Verlag, Berlin Heidelberg, 2005. doi:10.1007/0-387-28981-X.

[14] J. R. Bunch and L. Kaufman. Some stable methods for calculating inertia and solving symmetric linear systems. *Math. Comp.*, 31(137):163–179, 1977. doi:10.1090/S0025-5718-1977-0428694-0.

[15] J. R. Bunch, L. Kaufman, and B. Parlett. Decomposition of a symmetric matrix. *Numer. Math.*, 27:95–109, 1976. doi:10.1007/BF01399088.

[16] J. R. Bunch and B. N. Parlett. Direct methods for solving symmetric indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 8:639–655, 1971. doi:10.1137/0708060.

[17] W. Bunse and A. Bunse-Gerstner. *Numerische Lineare Algebra*. Teubner, Stuttgart, 1985.

[18] R. Byers. Solving the algebraic Riccati equation with the matrix sign function. *Linear Algebra Appl.*, 85:267–279, 1987.

[19] R. Byers and H. Xu. A new scaling for Newton's iteration for the polar decomposition and its backward stability. *SIAM J. Matrix Anal. Appl.*, 30(2):822–843, 2008. doi:10.1137/070699895.

[20] M. Casida. Time-dependent density functional response theory for molecules. In *Recent Advances in Density Functional Methods*, pages 155–192. World Scientific, 1995. doi:10.1142/9789812830586_0005.

[21] S. Chandrasekaran and A. H. Sayed. Stabilizing the generalized Schur algorithm. *SIAM J. Matrix Anal. Appl.*, 17(4):950–983, 1996. doi:10.1137/S0895479895287419.

[22] J. J. Dongarra, J. R. Gabriel, D. D. Koelling, and J. H. Wilkinson. The eigenvalue problem for Hermitian matrices with time reversal symmetry. *Linear Algebra Appl.*, 60:27–42, 1984. doi:10.1016/0024-3795(84)90068-5.

[23] I. S. Duff. MA57—a code for the solution of sparse symmetric definite and indefinite systems. *ACM Trans. Math. Software*, 30(2):118–144, 2004. doi:10.1145/992200.992202.

[24] A. Edelman and S. Jeong. Fifty three matrix factorizations: A systematic approach, 2021. arXiv:2104.08669.

[25] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, fourth edition, 2013.

[26] D. Henrion and P. Hippe. Hyperbolic QR factorization for J-spectral factorization of polynomial matrices. In *42nd IEEE International Conference on Decision and Control*, volume 4, pages 3479.–3484, 2003. doi:10.1109/CDC.2003.1271685.

[27] N. J. Higham. J-orthogonal matrices: properties and generation. *SIAM Rev.*, 45(3):504–519, 2003. doi:10.1137/S0036144502414930.

[28] N. J. Higham. *Functions of Matrices: Theory and Computation*. Applied Mathematics. SIAM, Philadelphia, PA, 2008. doi:10.1137/1.9780898717778.

[29] N. J. Higham, D. Mackey, N. Mackey, and F. Tisseur. Functions preserving matrix groups and iterations for the matrix square root. *SIAM J. Matrix Anal. Appl.*, 26(3):849–877, 2005. doi:10.1137/S0895479804442218.

[30] N. J. Higham, C. Mehl, and F. Tisseur. The canonical generalized polar decomposition. *SIAM J. Matrix Anal. Appl.*, 31(4):2163–2180, 2010. doi:10.1137/090765018.

[31] I. Houtzager. JQR/JRQ/JQL/JLQ factorizations. *MATLAB Central File Exchange*, 2015. Retrieved February 12, 2020. URL: https://www.mathworks.com/matlabcentral/fileexchange/50329-jqr-jrq-jql-jlq-factorizations.

[32] C. Kenney and A. J. Laub. Rational iterative methods for the matrix sign function. *SIAM J. Matrix Anal. Appl.*, 12:273–291, 1991. doi:10.1137/0612020.

[33] C. Kenney and A. J. Laub. On scaling Newton's method for polar decomposition and the matrix sign function. *SIAM J. Matrix Anal. Appl.*, 13:688–706, 1992. doi:10.1137/0613044.

[34] U. Kintzel. Procrustes problems in finite dimensional indefinite scalar product spaces. *Linear Algebra Appl.*, 402:1–28, 2005. doi:10.1016/j.laa.2005.01.004.

[35] H. Li, H. Yang, and H. Shao. Perturbation analysis for the hyperbolic QR factorization. *Comput. Math. Appl.*, 63(12):1607–1620, 2012. doi:10.1016/j.camwa.2012.03.036.

[36] H. Ltaief, D. Sukkari, A. Esposito, Y. Nakatsukasa, and D. Keyes. Massively parallel polar decomposition on distributed-memory systems. *ACM Trans. Parallel Comput.*, 6(1), 2019. doi:10.1145/3328723.

[37] D. S. Mackey, N. Mackey, and F. Tisseur. Structured factorizations in scalar product spaces. *SIAM J. Matrix Anal. Appl.*, 27(3):821–850, 2005. doi:10.1137/040619363.

[38] C. Mehl, V. Mehrmann, and H. Xu. On doubly structured matrices and pencils that arise in linear response theory. *Linear Algebra Appl.*, 380:3–51, 2004. doi:10.1016/S0024-3795(02)00455-X.

[39] C. Mehl, A. C. M. Ran, and L. Rodman. Polar decompositions of normal operators in indefinite inner product spaces. In *Operator theory in Krein spaces and nonlinear eigenvalue problems*, volume 162 of *Oper. Theory Adv. Appl.*, pages 277–292. Birkhäuser, Basel, 2006. doi:10.1007/3-7643-7453-5_15.

[40] V. Mehrmann and F. Poloni. Doubling algorithms with permuted Lagrangian graph bases. *SIAM J. Matrix Anal. Appl.*, 33(3):780–805, 2012. doi:10.1137/110850773.

[41] Y. Nakatsukasa, Z. Bai, and F. Gygi. Optimizing Halley's iteration for computing the matrix polar decomposition. *SIAM J. Matrix Anal. Appl.*, 31(5):2700–2720, 2010. doi:10.1137/090774999.

[42] Y. Nakatsukasa and R. W. Freund. Computing fundamental matrix decompositions accurately via the matrix sign function in two iterations: the power of Zolotarev's functions. *SIAM Rev.*, 58(3):461–493, 2016. doi:10.1137/140990334.

[43] Y. Nakatsukasa and N. J. Higham. Backward stability of iterations for computing the polar decomposition. *SIAM J. Matrix Anal. Appl.*, 33(2):460–479, 2012. doi:10.1137/110857544.

[44] G. Onida, L. Reining, and A. Rubio. Electronic excitations: density-functional versus many-body Green's-function approaches. *Rev. Mod. Phys.*, 74:601–659, Jun 2002. doi:10.1103/RevModPhys.74.601.

[45] C. Penke, A. Marek, C. Vorwerk, C. Draxl, and P. Benner. High performance solution of skew-symmetric eigenvalue problems with applications in solving the Bethe-Salpeter eigenvalue problem. *Parallel Comput.*, 96:102639, 2020. doi:10.1016/j.parco.2020.102639.

[46] Federico Poloni. *Permuted Graph Matrices and Their Applications*, pages 107–129. Springer International Publishing, Cham, 2015. doi:10.1007/978-3-319-15260-8_5.

[47] J. D. Roberts. Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. *Internat. J. Control*, 32(4):677–687, 1980. doi:10.1080/00207178008922881.

[48] M Shao, F. H. da Jornada, C. Yang, J. Deslippe, and S. G. Louie. Structure preserving parallel algorithms for solving the Bethe-Salpeter eigenvalue problem. *Linear Algebra Appl.*, 488:148–167, 2016. doi:10.1016/j.laa.2015.09.036.

[49] S. Singer. Indefinite QR factorization. *BIT Numer. Math.*, 46(1):141–161, 2006. doi:10.1007/s10543-006-0044-5.

[50] S Singer and S. Singer. Rounding-error and perturbation bounds for the indefinite QR factorization. In *Proceedings of the International Workshop on Accurate Solution of Eigenvalue Problems (University Park, PA, 1998)*, volume 309, pages 103–119, 2000. doi:10.1016/S0024-3795(99)00156-1.

[51] D. Sukkari, H. Ltaief, A. Esposito, and D. Keyes. A QDWH-based SVD software framework on distributed-memory manycore systems. *ACM Trans. Math. Software*, 45(2):Art. 18, 21, 2019. doi:10.1145/3309548.

[52] X. Sun and E. S. Quintana-Ortí. Spectral division methods for block generalized Schur decompositions. *Math. Comp.*, 73(248):1827–1847, 2004. doi:10.1090/S0025-5718-04-01667-9.

[53] K. Veselić. *Damped oscillations of linear systems*, volume 2023 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin Heidelberg, 2011. doi:10.1007/978-3-642-21335-9.

[54] D. Watkins. *The Matrix Eigenvalue Problem*. SIAM, 2007. doi:10.1137/1.9780898717808.

[55] Y. Yamamoto, Y. Nakatsukasa, Y. Yanagisawa, and T. Fukaya. Roundoff error analysis of the Cholesky QR2 algorithm. *ETNA*, 44:306–326, 2015.