Permuted graph matrices and their applications

Federico Poloni

Abstract A *permuted graph matrix* is a matrix $V \in \mathbb{C}^{(m+n)\times m}$ such that every row of the $m \times m$ identity matrix I_m appears at least once as a row of V. Permuted graph matrices can be used in some contexts in place of orthogonal matrices, for instance when giving a basis for a subspace $\mathscr{U} \subseteq \mathbb{C}^{m+n}$, or to normalize matrix pencils in a suitable sense. In these applications the permuted graph matrix can be chosen with bounded entries, which is useful for stability reasons; several algorithms can be formulated with numerical advantage with permuted graph matrices. We present the basic theory and review some applications from optimization or in control theory.

1 Introduction

A *graph matrix* is a matrix of the form

$$\mathscr{G}(X) := \begin{bmatrix} I_m \\ X \end{bmatrix}, \quad X \in \mathbb{C}^{n\times m},$$

where I_m is the $m \times m$ identity matrix. The name comes from the set-theoretical definition of graph of a function f as the set of pairs $(x, f(x))$. The image $\operatorname{im}\mathscr{G}(X)$ of a graph matrix is sometimes called *graph subspace*; however, this is improper, since "graph-ness" is a property of the basis matrix $\mathscr{G}(X)$, not of the subspace. Indeed, almost every subspace is a graph subspace: let

$$U = \begin{bmatrix} E \\ A \end{bmatrix}, \quad E \in \mathbb{C}^{m\times m}, \quad A \in \mathbb{C}^{n\times m} \tag{1}$$

Federico Poloni
Dipartimento di Informatica, Università di Pisa, Largo B. Pontecorvo, 3, I–56127 Pisa, Italy, e-mail: fpoloni@di.unipi.it

be any matrix with full column rank; whenever E is invertible, we have $U = \mathscr{G}(AE^{-1})E$, and hence $\operatorname{im} U = \operatorname{im} \mathscr{G}(AE^{-1})$, because post-multiplying by E^{-1} does not change the column space.

It will be useful to introduce a notation that avoids the dependence on the change of basis matrix E. Given $U, V \in \mathbb{C}^{(m+n)\times m}$ with full column rank, we write $U \sim V$ to mean that there exists an invertible $S \in \mathbb{C}^{m\times m}$ such that $U = VS$. In other words, this is the equivalence relation "U has the same column space as V".

Note that, in the above setting, given a generic matrix U, computing $X = AE^{-1}$ is not a good idea numerically, since its top $m \times m$ block E could be ill-conditioned or even singular. A modification of this approach is the following: instead of requiring an identity submatrix in the top block, we can ask for a subset of the rows that, when taken in some order, forms an identity matrix. More formally, we call a matrix $V \in \mathbb{C}^{(m+n)\times m}$ a *permuted graph matrix* if there exist $X \in \mathbb{C}^{n\times m}$ and a permutation matrix $P \in \mathbb{R}^{(m+n)\times(m+n)}$ such that $V = P\mathscr{G}(X)$. This is equivalent to requiring that every row of I_m occurs at least once as a row of V.

It is easy to prove that every subspace is spanned by the columns of a permuted graph matrix. Indeed, let the columns of $U \in \mathbb{C}^{(m+n)\times m}$ form a basis for a given subspace; U must then have full column rank, that is, it must contain an invertible submatrix $E \in \mathbb{C}^{m\times m}$. We can choose a permutation matrix P such that $U = P\begin{bmatrix} E \\ A \end{bmatrix}$, with $A \in \mathbb{C}^{n\times m}$. Then, $U \sim P\mathscr{G}(X)$ with $X = AE^{-1}$. We call $P\mathscr{G}(X)$ a *permuted graph representation* of U, or of its column space.

A more interesting result is the following, which shows that we can always find a basis matrix in the form $P\mathscr{G}(X)$ with the additional property that X is bounded in a suitable sense.

Theorem 1 ([22, 31]). *Let $U \in \mathbb{C}^{(m+n)\times m}$ be a matrix with full column rank. Then, there exist $X \in \mathbb{C}^{n\times m}$ and a permutation matrix $P \in \mathbb{R}^{(m+n)\times(m+n)}$ such that $U \sim P\mathscr{G}(X)$ and $\|X\|_{\max} \leq 1$.*

We have used the notation $\|X\|_{\max} := \max_{i,j}|x_{ij}|$, where x_{ij} are the entries of the matrix X; essentially, the theorem states that all the entries of X are bounded in modulus by 1.

In this chapter, we focus on Theorem 1, its extension to Lagrangian subspaces, and the applications of these two results. There are several contexts in numerical linear algebra and in control theory in which it is useful to work with the pair (P, X) as a representation of the subspace $\operatorname{im} U$; we review briefly these applications and the underlying theory.

The chapter is organized as follows. We describe an efficient algorithm for the computation of a permuted graph matrix $P\mathscr{G}(X) \sim U$ in Section 2; in Section 3, we present a result regarding their conditioning and introduce two different applications of these matrices in optimization. Another application, skeleton approximation of large-scale matrices, is discussed in Section 4. A structured version of this technique is presented in Section 5; in Section 6 we show how permuted graph representations can be used to work with matrix pencils. In section 7 we introduce briefly numerical methods for a standard problem in control theory, constant-coefficient linear-quadratic control, and in Section 8 and 9 we show how two of these algo-

rithms can be improved with the use of permuted graph matrices. Lastly, Section 10 discusses open issues and research problems.

2 Computing permuted graph bases

We start our review by discussing the computation of permuted graph bases in practice. The idea of the proof of Theorem 1 is the following. Choose as E the $m \times m$ submatrix of U that maximizes $|\det E|$ (*maximum-volume submatrix*). Using Cramer's rule on the linear system $XE = A$, one can write $|x_{ij}| = \frac{|\det E'|}{|\det E|}$, for a suitable $m \times m$ submatrix E' of U (depending on i, j), hence the result follows.

Unfortunately, finding an E with this maximizing property is an NP-hard problem [13], so this construction is not useful computationally. We can use instead an iterative algorithm that resembles a lot the so-called "canonical tableaux" implementation of the simplex algorithm [14], in that we update at each step an active set of rows containing an identity submatrix. This procedure is described in [22, 31, 35]; we present it as Algorithm 1. The method produces a permuted graph representation in which each entry of X is bounded in magnitude by a parameter τ. It is advised to choose $\tau > 1$ (for instance $\tau = 2$), to avoid numerical troubles with entries that are exactly 1 and to get faster convergence.

Input: $U \in \mathbb{C}^{(m+n)\times m}$ with full column rank; a threshold value $\tau \geq 1$; an initial permutation P_0 such that the top m rows of $P_0^T U$ form an invertible matrix
Output: A permutation matrix $P \in \mathbb{C}^{(m+n)\times(m+n)}$ and $X \in \mathbb{C}^{n\times m}$ such that $U \sim P\mathscr{G}(X)$ and $\|X\|_{\max} \leq \tau$
Let $P = P_0$, $\left[\begin{smallmatrix} E \\ A \end{smallmatrix}\right] = P^T U$, and $X = AE^{-1}$;
while $\|X\|_{\max} > \tau$ **do**
  take a pair (i, j) such that $|x_{ij}| > \tau$;
  let $P' = P\Pi$, where $\Pi \in \mathbb{C}^{(m+n)\times(m+n)}$ is the permutation that exchanges j and $m+i$;
  find $X' \in \mathbb{C}^{n\times m}$ such that $P\mathscr{G}(X) \sim P'\mathscr{G}(X')$;
  replace (X, P) with (X', P') and continue;
end

Algorithm 1: Obtaining a permuted graph representation with $\|X\|_{\max} \leq \tau$ [22, 31, 35]

In practice, the permutation P can be stored as a sequence of $m+n$ integers, and all the needed operations on it can be performed on a computer in $O(m+n)$ time and space.

The computation of X' in Algorithm 1 can be performed efficiently as well. Here and in the following we use the notation $X_{\mathcal{I}\mathcal{J}}$ to denote the submatrix of $X \in \mathbb{C}^{n\times m}$ containing the rows with indices $\mathcal{I}$ and columns with indices $\mathcal{J}$, where $\mathcal{I}$ (resp. $\mathcal{J}$) is a tuple of distinct indices in $\{1, 2, \ldots, n\}$ (resp. $\{1, 2, \ldots, m\}$). Moreover, we denote by $\mathcal{I}^c$ a tuple composed of all the row (or column) indices that do not belong to $\mathcal{I}$,

and with a colon ‘: ’ (as in many computer languages) the whole set of admissible row/column indices.

Lemma 1 ([35, Lemma 4.1]). *Let a permutation matrix $P \in \mathbb{C}^{(m+n)\times(m+n)}$ and $X \in \mathbb{C}^{n\times m}$ be given. Let $\mathcal{I} = (i_1, i_2, \dots, i_\ell)$ be distinct elements of $\{1,2,\dots,n\}$ and $\mathcal{J} = (j_1, j_2, \dots, j_\ell)$ be distinct elements of $\{1,2,\dots,m\}$. Let $P' = P\Pi$, where Π is the permutation that swaps j_k with $m+i_k$, for each $k = 1,2,\dots,\ell$, and leaves everything else unchanged. A matrix $X' \in \mathbb{C}^{n\times m}$ such that $P\mathcal{G}(X) \sim P'\mathcal{G}(X')$ exists if and only if $X_{\mathcal{I}\mathcal{J}}$ is invertible, and in that case it is given by*

$$X' = \begin{bmatrix} X'_{\mathcal{I}\mathcal{J}} & X'_{\mathcal{I}\mathcal{J}^c} \\ X'_{\mathcal{I}^c\mathcal{J}} & X'_{\mathcal{I}^c\mathcal{J}^c} \end{bmatrix} = \begin{bmatrix} (X_{\mathcal{I}\mathcal{J}})^{-1} & -(X_{\mathcal{I}\mathcal{J}})^{-1}X_{\mathcal{I}\mathcal{J}^c} \\ X_{\mathcal{I}^c\mathcal{J}}(X_{\mathcal{I}\mathcal{J}})^{-1} & X_{\mathcal{I}^c\mathcal{J}^c} - X_{\mathcal{I}^c\mathcal{J}}(X_{\mathcal{I}\mathcal{J}})^{-1}X_{\mathcal{I}\mathcal{J}^c} \end{bmatrix}.$$

Lemma 1 shows how to update a permuted graph representation $P\mathcal{G}(X)$ when we change the set of rows where the identity submatrix is located. The operation needed in Algorithm 1 corresponds to the case in which $\mathcal{I} = \{i\}$ and $\mathcal{J} = \{j\}$ have one element.

The map $X \mapsto X'$ appears in other applications as well and is known as *principal pivot transform* [47].

As an initial permutation, in absence of better guesses, one can take the permutation P produced by a rank-revealing QR factorization $U^H = QRP$ [21, Section 5.4.1]. With this choice, one can prove (when $\tau > 1$) that the algorithm terminates in $O(n\log_\tau n)$ steps, with a total cost of $O(n^3\log_\tau n)$ floating point operations, and converges to a *local* maximizer of $|\det E|$ (that is, a submatrix E such that $|\det E| \geq |\det E'|$ for each other submatrix E' differing from E only by a single row). Moreover, the determinant of the top $m \times m$ submatrix of $P^T U$ increases by a factor greater than τ at each step. In practice, the number of steps is often much lower than the bound, and in many small-scale cases the P coming from rank-revealing QR already gives an X with $\|X\|_{\max} \leq 1$. Indeed, finding the submatrix E with maximum volume $|\det E|$ is a problem that can be explicitly related to the computation of rank-revealing factorizations [42].

Another area of mathematics where these submatrix determinants appear is algebraic geometry: given $U \in \mathbb{C}^{(m+n)\times n}$ with full column rank, the determinants of all possible $\binom{m+n}{m}$ subset of rows (each subset ordered, for instance, in increasing order) are called *Plücker (projective) coordinates* of the subspace $\operatorname{im} U$. Indeed, if we have two matrices spanning the same subspace, U and $V = UE \sim U$, their Plücker coordinates differ only by a common factor $\det E$, and one can show that the converse holds, that is, matrices with the same Plücker coordinates up to a common multiple are equivalent according to $\sim$ and span the same subspace.

3 Conditioning of subspaces and applications in optimization

Given a matrix U, its condition number $\kappa(U) := \frac{\sigma_{\max}(U)}{\sigma_{\min}(U)}$ (where $\sigma_{\min}(U)$ and $\sigma_{\max}(U)$ are its minimum and maximum singular value, respectively) measures the

sensitivity of its column space $\operatorname{im} U$ with respect to perturbations [46, Page 154]. Hence, if we wish to perform computations involving a subspace, a natural way to operate on it is through an orthogonal basis U_O, for which $\kappa(U_O) = 1$. Suppose that we decide instead to use a basis $U_G = P\mathscr{G}(X)$ which is a permuted graph matrix. How well does it fare with respect to this measure? The answer is given by the following result.

Theorem 2 ([35]). *Let $U_G = P\mathscr{G}(X) \in \mathbb{C}^{(m+n)\times m}$, where the elements x_{ij} of X satisfy the inequality $|x_{ij}| \leq \tau$ for a certain $\tau \geq 1$. Then, $\kappa(U_G) \leq \sqrt{1+mn\tau^2}$.*

Proof. Because of the identity submatrix, $\|U_G v\|_2 \geq \|v\|_2$ for each $v \in \mathbb{C}^m$, hence $\sigma_{\min}(U_G) \geq 1$. On the other hand,

$$\sigma_{\max}(P\mathscr{G}(X)) = \|P\mathscr{G}(X)\|_2 = \|\mathscr{G}(X)\|_2 \leq \sqrt{\|I_m\|_2^2 + \|X\|_2^2} \leq \sqrt{1+mn\tau^2}. \quad \square \tag{2}$$

The condition number $\kappa(U_G)$ is not as small as the perfect $\kappa(U_O) = 1$ of an orthogonal basis, but still it can be explicitly bounded by a linear function of the dimensions and of the chosen threshold τ. A permuted graph basis can hence be used to represent a subspace in a stable way and to operate on it.

Are there contexts in which there is an advantage in using a permuted graph basis U_G rather than an orthogonal one U_O? We sketch two applications here, taken from [48] and [3], respectively. More examples will appear in the next sections.

A problem encountered in optimization is the maximization (or minimization) of functions on the *Grassmann manifold* [1, 48], i.e., the set of m-dimensional subspaces of $\mathbb{C}^{m+n}$. In practice, this means maximizing a given function $f : \mathbb{C}^{(m+n)\times m} \mapsto \mathbb{R}$ such that $f(U) = f(V)$ whenever $U \sim V$. Working with orthogonal bases may lead to some difficulties. First, the parametrization of the Grassmann manifold via orthogonal matrices is ambiguous, since the relation between an orthogonal matrix $U \in \mathbb{C}^{(m+n)\times m}$ and its spanned subspace is not one-to-one. As a consequence, the gradient ∇f is always zero in some directions, and the optimization problem in this formulation is never strictly convex. Moreover, in most iterative algorithms, it is difficult to enforce orthogonality of the next iterate explicitly, so a typical algorithm will make an update in a general direction in $\mathbb{C}^{(m+n)\times m}$ and then restore orthogonality at a later time via projection.

Neither of these problems is unsolvable, and there are now mature algorithms for optimization on matrix manifolds [1]. Nevertheless, using permuted graph bases rather than orthogonal bases allows for a simplification of the problem. The maps $g_P(X) := X \mapsto P\mathscr{G}(X)$ are one-to-one local charts and together constitute an atlas of the Grassmann manifold, so they can be used to reduce the problem to a standard multivariate optimization problem on the space $\mathbb{C}^{nm}$. In practice, one defines for each permutation matrix the auxiliary map $f_P : \mathbb{C}^{n\times m} \to \mathbb{R}$ as $f_P(X) := f(P\mathscr{G}(X))$, and uses a traditional multivariate optimization algorithm on it. We sketch a method, originally from [48], in Algorithm 2: at each step, we check if the entries of the current iterate X have magnitude greater than τ, and if so, we update the permutation. Changing the permutation P with Algorithm 1 is needed in few iterations only, and

the previous value of P is a good initial guess, so its cost is typically much less than the generic $O(n^3 \log_\tau n)$.

Input: A function $f : \mathbb{C}^{(m+n)\times m} \to \mathbb{R}$ such that $f(U) = f(V)$ whenever $U \sim V$; an initial value $U \in \mathbb{C}^{(m+n)\times m}$; a threshold $\tau > 1$
Output: A (possibly local) minimum of f
Find a permuted graph representation $U \sim P\mathscr{G}(X)$ (with threshold τ);
while *X is not a local minimum of f_P* **do**
  apply one step of a multivariate optimization algorithm (gradient descent, Newton, BFGS...) to f_P, starting from X, obtaining a new point X';
  if $\|X'\|_{\max} > \tau$ **then**
    Use Algorithm 1 to find a permuted graph representation $P''\mathscr{G}(X'')$ of $P\mathscr{G}(X')$, with threshold τ;
    replace (X,P) with (X'',P'') and continue;
  else
    replace X with X' and continue;
  end
end

Algorithm 2: Optimization on the Grassmann manifold [48]

A different context in optimization in which suitable permutations and graph forms have appeared recently is the preconditioning and solution of large-scale saddle-point problems [3, 15, 16]. We present here the preconditioner for least-squares problems appearing in [3]. A least-squares problem $\min_{x\in\mathbb{C}^m} \|Ux - b\|$, for $U \in \mathbb{C}^{(m+n)\times m}$, can be reformulated as the augmented system

$$\begin{bmatrix} I_{m+n} & U \\ U^H & 0 \end{bmatrix} \begin{bmatrix} r \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}.$$

Let us take a permuted graph basis $U = P\begin{bmatrix} E \\ A \end{bmatrix} \sim P\mathscr{G}(X)$, with $X = AE^{-1}$; permuting the first $m+n$ entries and partitioning $Pr = \begin{bmatrix} r_E \\ r_A \end{bmatrix}$ and $Pb = \begin{bmatrix} b_E \\ b_A \end{bmatrix}$ conformably with $\begin{bmatrix} E \\ A \end{bmatrix}$, we get the equivalent system

$$\begin{bmatrix} I_m & 0 & E \\ 0 & I_n & A \\ E^H & A^H & 0 \end{bmatrix} \begin{bmatrix} r_E \\ r_A \\ x \end{bmatrix} = \begin{bmatrix} b_E \\ b_A \\ 0 \end{bmatrix}.$$

Finally, eliminating the variables r_E from this system and multiplying by $Q = \begin{bmatrix} I_n & 0 \\ 0 & E^{-H} \end{bmatrix}$ and Q^H on the two sides, one gets the equivalent reduced system

$$\begin{bmatrix} I_n & X \\ X^H & -I_m \end{bmatrix} \begin{bmatrix} r_A \\ Ex \end{bmatrix} = \begin{bmatrix} b_A \\ -b_E \end{bmatrix}.$$

The condition number of this linear system equals $\kappa(P\mathscr{G}(X))$ (see [3, p. 4]), and thus it can be bounded using Theorem 2. In practice, the matrix Q above is applied as a preconditioner; hence, to get faster convergence of preconditioned iterative meth-

ods, it is useful to choose a permuted graph basis with a small $\kappa(P\mathscr{G}(X))$. The authors in [3] suggest useful heuristic methods to find one for a large and sparse U.

4 Skeleton approximation

In this section, we consider the problem of finding or approximating $\|M\|_{\max}$ for a large-scale matrix M, not necessarily sparse.

Let $M \in \mathbb{C}^{n\times m}$, and $\mathcal{I},\mathcal{J}$ be two tuples of ℓ pairwise distinct row and column indices respectively. If $M_{\mathcal{I}\mathcal{J}}$ is invertible, the matrix

$$M_S = M_{:\mathcal{J}} M_{\mathcal{I}\mathcal{J}}^{-1} M_{\mathcal{I}:} \tag{3}$$

is called *skeleton approximation* of M along $(\mathcal{I},\mathcal{J})$ [23], and has the same entries as M on the rows belonging to $\mathcal{I}$ and the columns belonging to $\mathcal{J}$. Moreover, whenever $\operatorname{rank} M \leq \ell$ we have $M_S = M$. If $\mathcal{I}$ and $\mathcal{J}$ are chosen so that $|\det M_{\mathcal{I}\mathcal{J}}|$ is the maximum over all $\ell\times\ell$ submatrices, then one can prove specific approximation properties for the extremal values of M.

Theorem 3 ([22, 23]). *Let $M \in \mathbb{C}^{n\times m}$ and $\ell \leq \min(m,n)$ be given; let $\mathcal{I},\mathcal{J}$ be ℓ-tuples of pairwise distinct indices chosen so that $|\det M_{\mathcal{I}\mathcal{J}}|$ is maximal and M_S be the skeleton approximation* (3)*. Then,*

1. $\|M - M_S\|_{\max} \leq (\ell+1)\sigma_{\ell+1}$*, where $\sigma_{\ell+1}$ is the $(\ell+1)$st singular value of M;*
2. $\|M_{\mathcal{I}\mathcal{J}}\|_{\max} \geq \|M\|_{\max}/(2\ell^2+\ell)$.

As stated above, finding a maximum-volume submatrix is an NP-complete problem already in the case $\ell = m$, so in practice one must resort to heuristics and approximations. A possible implementation, using alternating optimization on $\mathcal{I}$ and $\mathcal{J}$, is given in Algorithm 3. As in Algorithm 1, termination is ensured by the fact that

Input: A matrix $M \in \mathbb{C}^{n\times m}$, possibly sparse or given implicitly as a procedure that returns single entries, rows of columns; an initial guess $\mathcal{J}$; a threshold $\tau \geq 1$
Output: ℓ-tuples of row and column indices $\mathcal{I},\mathcal{J}$ such that $\|M_{:\mathcal{J}}(M_{\mathcal{I}\mathcal{J}})^{-1}\|_{\max} \leq \tau$ and $\|(M_{\mathcal{I}\mathcal{J}})^{-1}M_{\mathcal{I}:}\|_{\max} \leq \tau$
repeat
  apply Algorithm 1 to $M_{:\,\mathcal{J}}$, producing a new index set $\mathcal{I}$ that maximizes $|\det M_{\mathcal{I}\mathcal{J}}|$;
  apply Algorithm 1 to $M_{\mathcal{I}:}^H$, producing a new index set $\mathcal{J}'$ that maximizes $|\det M_{\mathcal{I}\mathcal{J}'}|$;
  replace $\mathcal{J}$ with $\mathcal{J}'$ and continue;
until *$\mathcal{I}$ and $\mathcal{J}$ stop changing*;

Algorithm 3: Alternating optimization algorithm for skeleton approximation.

$|\det M_{\mathcal{I}\mathcal{J}}|$ increases monotonically by a factor larger than τ. As initial guess, one can take for instance a random $\mathcal{J}$, or start with a permuted graph representation of MV for a suitably chosen random full-rank $V \in \mathbb{C}^{m\times\ell}$.

Note that the procedure works on few rows and columns of M at a time, and in fact typically it will not even access many of its entries. Nevertheless, in practice $\|M_{\mathcal{I}\mathcal{J}}\|_{\max}$ is a good approximation of $\|M\|_{\max}$ in many real-world cases where the singular values of M decay sufficiently fast [22]. This method has been used in cases in which the entries of M can be efficiently generated one-by-one, or one row/column at a time; for instance, they might be the values of a bivariate function $f(x,y)$ on a huge grid. Generalizations to problems in more than two variables and tensor approximations can be devised using the same ideas; see, e.g., [41, 45]. This method, in combination with efficient tensor storage techniques, allows for the treatment of massively large maximization/minimization problems, with applications to many computationally challenging problems in quantum physics, computational chemistry and biology.

5 Permuted graph bases for Lagrangian subspaces

An n-dimensional subspace $\mathscr{U}$ of $\mathbb{C}^{2n}$ is called *Lagrangian* if $u^H J_{2n} v = 0$ for every $u, v \in \mathscr{U}$, where $J_{2n} := \left[\begin{smallmatrix} 0 & I_n \\ -I_n & 0 \end{smallmatrix}\right]$. Lagrangian subspaces appear naturally in systems and control theory, as we discuss later in Section 7.

Given $U \in \mathbb{C}^{2n\times n}$ of full column rank, $\operatorname{im} U$ is Lagrangian if and only if $U^H J_{2n} U = 0$. When $U = \mathscr{G}(X)$ is a graph basis, this expands to $X = X^H$, that is, $\operatorname{im}\mathscr{G}(X)$ is Lagrangian if and only if X is Hermitian. The same property does not hold for *permuted* graph bases, though; to recover it, we have to alter the definition to adapt it to this structured case. For $i = 1, 2, \ldots, n$, let

$$S_i = \left[\begin{array}{ccc|ccc} I_{i-1} & & & 0_{i-1} & & \\ & 0 & & & -1 & \\ & & I_{n-i} & & & 0_{n-i} \\ \hline 0_{i-1} & & & I_{i-1} & & \\ & 1 & & & 0 & \\ & & 0_{n-i} & & & I_{n-i} \end{array}\right],$$

where 0_k denotes the zero matrix of size $k \times k$; that is, S_i is the $2n \times 2n$ matrix that acts as $-J_2$ on the i-th and $n+i$-th component of a vector (swapping them and changing sign to one of them) and as the identity matrix on all other components. Clearly the S_i all commute. Let us consider the set of all 2^n possible products that we can build by taking a (possibly empty) subset of them,

$$\mathscr{S}_{2n} := \{S_{i_1} S_{i_2} \cdots S_{i_\ell} \mid 1 \le i_1 < i_2 < \ldots < i_\ell \le n\}, \quad \ell \in \{0, 1, \ldots, n\}.$$

The identity matrix I_{2n} and $-J_{2n}$ are contained in the set, corresponding to the trivial subsets. All the matrices in $\mathscr{S}_{2n}$ are orthogonal and *symplectic* (i.e., they satisfy $S^H J_{2n} S = J_{2n}$), and they are up to sign changes a subgroup of the permutation ma-

trices (the one generated by the transpositions $(i, n+i)$). We call these matrices *symplectic swaps*.

A symplectic swap can be stored as the subset $\{i_1, i_2, \ldots, i_\ell\}$, memorized for instance as a length-n binary vector; all the operations that we need can be easily performed on a computer in $O(n)$ space and time.

Using these matrices in place of the permutations, we can build an analogue of the theory of permuted graph bases for symplectic subspaces. Given $U \in \mathbb{C}^{2n\times n}$ and a symplectic swap $S \in \mathscr{S}_{2n}$, whenever the top $n \times n$ submatrix E of $S^T U = \left[\begin{smallmatrix} E \\ A \end{smallmatrix}\right]$ is nonsingular, we can form $X = AE^{-1}$ so that $U \sim S\mathscr{G}(X)$. Using the symplecticity of S, it is easy to check that $\operatorname{im} U$ is Lagrangian if and only if $X = X^H$; hence, if $X = X^H$ for some choice of $S \in \mathscr{S}_{2n}$, then the same property holds for all possible choices.

Since there are only 2^n symplectic swaps, less than the number of essentially different $n \times n$ submatrices of U, it is already nontrivial to see that for any $U \in \mathbb{C}^{2n\times n}$ with full column rank there exists at least one choice of S that gives an invertible E, let alone one with bounded X. Nevertheless, the following result holds.

Theorem 4 ([17, 35]). *Let $U \in \mathbb{C}^{2n\times n}$ have full column rank and satisfy $U^H J_{2n} U = 0$ (i.e., $\operatorname{im} U$ is Lagrangian). Then,*

1. *There exists $S \in \mathscr{S}_{2n}$ so that the top $n \times n$ submatrix E of $S^T U = \left[\begin{smallmatrix} E \\ A \end{smallmatrix}\right]$ is nonsingular, and hence $U \sim S\mathscr{G}(X)$ with $X = X^H = AE^{-1}$.*
2. *There exists $S \in \mathscr{S}_{2n}$ so that the above property holds, and $\|X\|_{\max} \le \sqrt{2}$.*

Item 1 appeared in [17], and Item 2 in [35]; indeed, one can find X with $|x_{ij}| \le 1$ when $i = j$ and $|x_{ij}| \le \sqrt{2}$ otherwise, which is a slightly stronger condition.

The proof of Item 2 is similar to the one for unstructured case: one looks for the symplectic swap S that maximizes $|\det E|$, where $S^T U = \left[\begin{smallmatrix} E \\ A \end{smallmatrix}\right]$. Similarly, for each $\tau \ge \sqrt{2}$ there is an iterative optimization algorithm with complexity $O(n^3 \log_{(\tau^2-1)^{1/2}} n)$ flops which produces a permuted Lagrangian graph representation $U \sim S\mathscr{G}(X)$ with $\|X\|_{\max} = \tau$. As a starting permutation, one can take the S originating from a variant of the rank-revealing QR factorization in which the third term is a symplectic swap rather than a permutation. The proof and the algorithm use ideas similar to the ones in the unstructured case; we refer the reader to [35] for more detail. Here we report only the analogue of Lemma 1, which gives an interesting symmetric variant of the principal pivot transform.

Lemma 2. *Let $S \in \mathscr{S}_{2n}$ be a symplectic swap and $X = X^H \in \mathbb{C}^{n\times n}$. Let $\mathcal{I} = (i_1, i_2, \ldots, i_\ell)$ be given, where the i_k are distinct elements of $\{1, 2, \ldots, n\}$. Define $S' = S S_{i_1} S_{i_2} \cdots S_{i_\ell}$. Then there exists a matrix $X' = (X')^H \in \mathbb{C}^{n\times n}$ such that $S\mathscr{G}(X) \sim S'\mathscr{G}(X')$ if and only if $X_{\mathcal{I}\mathcal{I}}$ is invertible, and in that case it is given by*

$$X' = \begin{bmatrix} X'_{\mathcal{I}\mathcal{I}} & X'_{\mathcal{I}\mathcal{I}^c} \\ X'_{\mathcal{I}^c\mathcal{I}} & X'_{\mathcal{I}^c\mathcal{I}^c} \end{bmatrix} = \begin{bmatrix} -(X_{\mathcal{I}\mathcal{I}})^{-1} & (X_{\mathcal{I}\mathcal{I}})^{-1} X_{\mathcal{I}\mathcal{I}^c} \\ X_{\mathcal{I}^c\mathcal{I}}(X_{\mathcal{I}\mathcal{I}})^{-1} & X_{\mathcal{I}^c\mathcal{I}^c} - X_{\mathcal{I}^c\mathcal{I}}(X_{\mathcal{I}\mathcal{I}})^{-1} X_{\mathcal{I}\mathcal{I}^c} \end{bmatrix}.$$

Some additional sign book-keeping is needed in addition to the above formula if we wish to get a representation with a symplectic swap as in Theorem 4: indeed, if

for a $k \in \{1,2,\dots,\ell\}$ the symplectic swap S already contains the factor S_{i_k}, then the product S' includes $S_{i_k}^2$, which acts as $-I_2$ on the i_kth and $(n+i_k)$th entry of a vector. Hence $S' \notin \mathscr{S}_{2n}$; to get back a symplectic swap we need to correct some signs in S' and X'. This is just a technical issue; a MATLAB function that deals with it and produces $S' \in \mathscr{S}_{2n}$ is in [43, file `private/updateSymBasis.m`].

The statement and proof of Theorem 2 hold for permuted Lagrangian graph bases as well, by simply changing P to S. Hence, permuted Lagrangian graph bases $U \sim S\mathscr{G}(X)$ provide a reasonably well-conditioned way to represent a Lagrangian subspace on a computer and perform computational work with it. This time, we have a distinct advantage with respect to orthogonal bases: the fact that the subspace is Lagrangian is equivalent to $X = X^H$, a property which is easy to enforce and deal with computationally. On the other hand, when working with orthogonal bases, it is well possible that a subspace "drifts away" from the manifold of Lagrangian subspaces due to the accumulation of numerical errors. Structure preservation in permuted Lagrangian graph bases will be crucial in Section 9.

6 Representation of pencils

A *matrix pencil* is a degree-1 matrix polynomial, i.e., an expression of the form $L(z) = L_1 z + L_0$, with $L_0, L_1 \in \mathbb{C}^{n\times m}$ and z an indeterminate. We call a pencil *row-reduced* if $\begin{bmatrix} L_1 & L_0 \end{bmatrix}$ has full row rank, i.e., if there exists no nonzero $v \in \mathbb{C}^n$ such that $v^H(L_1\lambda + L_0) = 0$ for all $\lambda \in \mathbb{C}$. We call a pencil *regular* if $m = n$ and $\det L(z)$ is not the zero polynomial. For a regular $L(z)$, the roots of $\det L(z)$ are called *eigenvalues*, and a vector $v \neq 0$ such that $L(\lambda)v = 0$ is called *(right) eigenvector* relative to the eigenvalue λ. We say that ∞ is an eigenvalue of $L(z)$ (with eigenvector v) whenever 0 is an eigenvalue of $L_0 z + L_1$ (with eigenvector v). A full theory of eigenvalues and eigenvectors of (non necessarily regular) matrix pencils, including an extension of the Jordan canonical form, can be found in the classical book [19].

In this section and the next ones, we focus on eigenvalue and eigenvector problems for pencils; therefore, we are free to replace a pencil with another one having the same eigenvalues and eigenvectors. We say that two matrix pencils $L(z), M(z) \in \mathbb{C}[z]^{n\times n}$ are *left equivalent* (and we write $L(z) \sim M(z)$) if there is an invertible matrix $N \in \mathbb{C}^{n\times n}$ (not depending on z) such that $L(z) = NM(z)$. When this property holds and $L(z)$ and $M(z)$ are regular, clearly they have the same eigenvalues and right eigenvectors. The symbol $\sim$ is the same that we have used for matrices spanning the same subspace, and indeed these two equivalence relations are intimately connected: given $L(z) = L_1 z + L_0$ and $M(z) = M_1 z + M_0$, we have $L(z) \sim M(z)$ if and only if $\begin{bmatrix} L_1 & L_0 \end{bmatrix}^H \sim \begin{bmatrix} M_1 & M_0 \end{bmatrix}^H$.

In the previous sections, we have focused our efforts on finding P and a bounded X so that $U \sim P\mathscr{G}(X)$, for a given matrix U. In view of the above connection, this translates immediately to a result on pencils.

Theorem 5 ([35]). *Let $L(z) = L_1 z + L_0 \in \mathbb{C}[z]^{n\times n}$ be a row-reduced matrix pencil. Then, there exists another matrix pencil $M(z) = M_1 z + M_0 \in \mathbb{C}[z]^{n\times n}$ such that $L(z) \sim M(z)$ and $\begin{bmatrix} M_1 & M_0 \end{bmatrix}^H = P\mathscr{G}(X)$, for a suitable permutation matrix $P \in \mathbb{R}^{2n\times 2n}$ and $X \in \mathbb{C}^{n\times n}$ with $\|X\|_{\max} \leq 1$.*

In other words, each column of I_n appears at least once among the columns of M_1 and M_0, and all the entries of these two matrices are bounded by 1.

Similarly, the results of Section 5 can be used to obtain pencils that are left equivalent to some with special structures. A row-reduced pencil $L(z) = L_1 z + L_0 \in \mathbb{C}^{2n\times 2n}$ is called *Hamiltonian* if $L_1 J_{2n} L_0^H + L_0 J_{2n} L_1^H = 0$; see [32, 40]. Simple manipulations show that this holds if and only if $U = \begin{bmatrix} L_1 & J_{2n} L_0 \end{bmatrix}^H \in \mathbb{C}^{4n\times 2n}$ satisfies $U^H J_{2n} U = 0$, i.e., $\operatorname{im} U$ is Lagrangian. Hence we can reduce to the setting of Theorem 4, obtaining the following result.

Theorem 6 ([35]). *Let $L(z) = L_1 z + L_0 \in \mathbb{C}[z]^{2n\times 2n}$ be a row-reduced Hamiltonian pencil. Then, there exist $S \in \mathscr{S}_{4n}$ and $X = X^H \in \mathbb{C}^{2n\times 2n}$ with $\|X\|_{\max} \leq \sqrt{2}$ so that $L(z) \sim M(z)$, with $M(z)$ defined by $\begin{bmatrix} M_1 & M_0 J_{2n} \end{bmatrix}^H = S\mathscr{G}(X)$.*

Notice the structure of $M(z)$: for each $i = 1, 2, \ldots, 2n$, either the ith column of M_1 or the $n \pm i$th column of M_0 is (modulo signs) equal to the ith column of the identity matrix I_{2n}.

It is common in the literature to represent a Hamiltonian pencil with no infinite eigenvalues as $L(z) \sim I_{2n} z - H$, where H is a *Hamiltonian matrix*, i.e., a matrix such that HJ_{2n} is Hermitian: this corresponds to the case $S = I_{4n}$ of Theorem 6. Introducing column swaps in the picture allows us to find a representation that has bounded entries and works without constraints on the eigenvalues.

Another structure that we can deal with is the following. A row-reduced pencil $L(z) = L_1 z + L_0 \in \mathbb{C}[z]^{2n\times 2n}$ is called *symplectic* if $L_1 J_{2n} L_1^H - L_0 J_{2n} L_0^H = 0$; see [32, 40]. If one partitions $L_1 = \begin{bmatrix} L_{10} & L_{11} \end{bmatrix}$, $L_0 = \begin{bmatrix} L_{00} & L_{01} \end{bmatrix}$, with all blocks $2n \times n$, this is equivalent to $U = \begin{bmatrix} L_{10} & L_{01} & L_{11} & L_{00} \end{bmatrix}^H$ spanning a Lagrangian subspace. Note that symplectic swaps act separately on the two blocks composing L_1 and on the two composing L_0. Keeping track of this, one can decompose $S \in \mathscr{S}_{4n}$ into two smaller symplectic swaps, and obtain a simpler statement for the analogue of Theorem 6 for symplectic pencils.

Theorem 7 ([35]). *Let $L(z) = L_1 z + L_0 \in \mathbb{C}[z]^{2n\times 2n}$ be a row-reduced symplectic pencil. Then, there exist two symplectic swaps $S_1, S_2 \in \mathscr{S}_{2n}$ and $X = X^H \in \mathbb{C}^{2n\times 2n}$ with $\|X\|_{\max} \leq \sqrt{2}$ so that $L(z) \sim M(z)$, with $M(z)$ defined by*

$$M(z) = \begin{bmatrix} I_n & X_{11} \\ 0 & X_{12}^H \end{bmatrix} S_1 z - \begin{bmatrix} X_{12} & 0 \\ X_{22} & I_n \end{bmatrix} S_2, \quad X = \begin{bmatrix} X_{11} & X_{12} \\ X_{12}^H & X_{22} \end{bmatrix}.$$

Again, the representation with $S_1 = S_2 = I_{2n}$ is widely used [11, 18, 34].

The main advantage of these forms is that we can represent on a computer pencils that are symplectic or Lagrangian, not up to numerical errors but exactly, and at the same time we do not have to deal with the numerical troubles of unduly large entries.

Lemma 1 bounds the quantity $\kappa(\begin{bmatrix}M_1 & M_0\end{bmatrix}^H)$ for these "permuted graph pencils". Using standard properties of the singular values, one can see that the inverse of this quantity is the relative distance (in the Euclidean norm) to the closest non-row-reduced pencil, i.e.,

$$\kappa(\begin{bmatrix}M_1 & M_0\end{bmatrix})^{-1} = \frac{\min_{\tilde{M}_1,\tilde{M}_0} \left\| \begin{bmatrix}\tilde{M}_1 & \tilde{M}_0\end{bmatrix} - \begin{bmatrix}M_1 & M_0\end{bmatrix} \right\|_2}{\left\| \begin{bmatrix}M_1 & M_0\end{bmatrix} \right\|_2},$$

where the minimum is taken over all the pencils $\tilde{M}_1 z + \tilde{M}_0$ that are not row-reduced. While having a small $\kappa(\begin{bmatrix}M_1 & M_0\end{bmatrix})$ seems desirable, because it means that $M(z)$ is far away from a variety of ill-posed problems, it is not clear what exactly this quantity represents in terms of perturbation theory. It is not a condition number for the eigenvalues, nor the distance from the closest singular (i.e., non-regular) pencil. Indeed, all non-row-reduced pencils are singular, but the converse does not hold (see for instance (7) in the following for a counterexample).

Hence, from the point of view of perturbation theory and numerical stability, the effectiveness of these special forms can currently only be justified by heuristic reasons.

7 Numerical methods for linear-quadratic optimal control

Systems and control theory is a branch of engineering and mathematics that leads to an abundance of linear algebra applications. Here we focus on a simple version of the linear-quadratic optimal control problem [34]. The reader will find several chapters in this book dedicated to control theory, but we give a quick introduction to the numerical methods directly here to keep this chapter self-contained and introduce a notation consistent with our exposition.

Given matrices $A \in \mathbb{R}^{n\times n}$, $B, S \in \mathbb{R}^{n\times m}$, $Q = Q^T \in \mathbb{R}^{n\times n}$, $R = R^T \in \mathbb{R}^{m\times m}$, one looks for vector-valued functions $x, \mu : \mathbb{R}_+ \to \mathbb{R}^n$, $u : \mathbb{R}_+ \to \mathbb{R}^m$ such that

$$\begin{bmatrix} 0 & I_n & 0 \\ -I_n & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{d}{dt} \begin{bmatrix} \mu(t) \\ x(t) \\ u(t) \end{bmatrix} = \begin{bmatrix} 0 & A & B \\ A^T & Q & S \\ B^T & S^T & R \end{bmatrix} \begin{bmatrix} \mu(t) \\ x(t) \\ u(t) \end{bmatrix}, \quad x(0) = x_0, \lim_{t\to\infty} \begin{bmatrix} \mu(t) \\ x(t) \\ u(t) \end{bmatrix} = 0. \quad (4)$$

The textbook solution to this problem goes as follows. First, assuming $R > 0$, one eliminates $u(t)$ and swaps the two remaining equations, obtaining

$$\frac{d}{dt} \begin{bmatrix} x(t) \\ \mu(t) \end{bmatrix} = H \begin{bmatrix} x(t) \\ \mu(t) \end{bmatrix}, \quad H = -J_{2n} M, \quad M = \begin{bmatrix} Q & A^T \\ A & 0 \end{bmatrix} - \begin{bmatrix} S \\ B \end{bmatrix} R^{-1} \begin{bmatrix} S^T & B^T \end{bmatrix}. \quad (5)$$

One can prove under mild assumptions that H has n eigenvalues with negative real part and n with positive real part (counted with multiplicities); hence there exists a unique n-dimensional subspace $\mathscr{U}$ such that $H\mathscr{U} \subseteq \mathscr{U}$, and the restriction of H to $\mathscr{U}$ has only eigenvalues with negative real part. A stable solution to the system

of ordinary differential equations (5) is obtained if and only if $\begin{bmatrix} x(0) \\ \mu(0) \end{bmatrix} \in \mathscr{U}$, and if this happens then $\begin{bmatrix} x(t) \\ \mu(t) \end{bmatrix} \in \mathscr{U}$ for all $t > 0$. How does one determine $\mathscr{U}$? The traditional approach is looking for a graph basis $U = \mathscr{G}(X)$, with $X \in \mathbb{R}^{n \times n}$, which exists under additional assumptions on the problem (typically satisfied). Then the condition $H\mathscr{U} \subseteq \mathscr{U}$ becomes $HU = UT$, for some matrix $T \in \mathbb{C}^{n \times n}$ with all its eigenvalues in the left half-plane; expanding out the products gives

$$\begin{cases} M_{11} + M_{12}X + XM_{21} + XM_{22}X = 0, \\ T = M_{21} + M_{22}X, \end{cases} \quad \text{with } M = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}. \tag{6}$$

The first equation in X alone is called *algebraic Riccati equation*; several solution methods exist. Once X is determined, thanks to the previous observation, we have $\mu(t) = Xx(t)$ for each t, and hence some manipulations give $u(t) = Kx(t)$ with $K = -R^{-1}(B^T X + S^T)$, and $x(t) = \exp((A + BK)t)x_0$.

Although one can prove that $\mathscr{U}$ admits a graph basis, this does not mean that it is a good idea to compute it numerically. The corresponding X might have very large elements. An alternative strategy is computing an orthogonal basis instead. Given any basis $U = \begin{bmatrix} E \\ A \end{bmatrix}$ for $\mathscr{U}$, we can reduce the problem to solving an initial-value ODE problem for $w(t) : \mathbb{R}_+ \to \mathbb{R}^n$ such that $\begin{bmatrix} x(t) \\ \mu(t) \end{bmatrix} = Uw(t)$. A necessary step is computing $w(0) = w_0$, which might still be troublesome numerically if E is ill-conditioned, but all the other steps, notably the eigenvalue computation, benefit from the additional stability associated with working with orthogonal matrices. If needed, the solution X of the Riccati equation can be obtained as well as AE^{-1}.

One can apply this approach of computing an invariant subspace directly to (4) as well. Let us call $\mathcal{E}$ and $\mathcal{A}$ the two block-3×3 matrices appearing in the left- and right-hand side of the leftmost equation in (4), respectively. This time, $\mathcal{E}$ is singular, but one can generalize the concept of invariant subspaces to matrix pencils. Given a regular pencil $\mathcal{E}z - \mathcal{A} \in \mathbb{C}[z]^{k \times k}$, we say that the image of a $U \in \mathbb{C}^{k \times \ell}$ with full column rank is a *deflating subspace* if there are $V \in \mathbb{C}^{k \times \ell}$, $E, A \in \mathbb{C}^{\ell \times \ell}$ such that $(\mathcal{E}z - \mathcal{A})U = V(Ez - A)$. The eigenvalues of $Ez - A$ are a subset of those of $\mathcal{E}z - \mathcal{A}$, and are called *associated* with the deflating subspace $\operatorname{im} U$. Under the same assumptions that we have made above, $\mathcal{E}z - \mathcal{A}$ has m eigenvalues equal to ∞, n with positive real part and n with negative real part; the finite eigenvalues coincide with those of H. One can solve a generalized eigenvalue problem [21, Section 7.7] to determine the invariant subspace associated with the last ones, and proceed similarly.

Several more general settings exist, most notably *finite-horizon problems* in which the boundary condition at ∞ in (4) is replaced by one at a time $t_f > 0$, or problems in which R is not invertible and the assumptions that we made on the location of eigenvalues are not respected. Large-scale problems with pencils exhibiting the same structure appear for instance in model reduction.

In the numerical solution of linear-quadratic control problems, matrix structures play a crucial role. The pencil $J_{2n}z - M$ is Hamiltonian, as well as the matrix H, and

the pencil $\mathcal{E}z-\mathcal{A}$ is *even*, i.e., $\mathcal{E}=-\mathcal{E}^H$ and $\mathcal{A}=\mathcal{A}^H$. These pencils and matrices have a distinguishing pairing of eigenvalues; namely, for each eigenvalue λ, one has that $-\bar{\lambda}$ is an eigenvalue as well. For Hamiltonian problems, moreover, $\mathscr{U}$ is Lagrangian, and hence $X=X^T$. For even problems in general this latter property does not hold (although a similar property does hold for the pencil $\mathcal{E}z-\mathcal{A}$ defined in (4)).

Numerical methods that exploit these structures are preferable, not only for speed, but especially for accuracy: in an ill-conditioned problem, for instance, an unstructured eigensolver might detect numerically $n+1$ eigenvalues with negative real part and $n-1$ with positive real part, a situation which is impossible under the structural constraints, and hence fail to identify correctly the unique n-dimensional invariant subspace. Even when this does not happen, it is a sounder theoretical guarantee to have a low *structured* backward error, that is, to be able to guarantee that the computed solution is the exact solution of a nearby problem respecting the same structure.

There has been extensive numerical research on how to accurately solve Hamiltonian and even eigenvalue problems; countless methods have been suggested [9, 34]: for instance, focusing only on the small-case dense case, there are the Newton method for algebraic Riccati equations [5, 24, 30], QR-type algorithms based on reduction to Hamiltonian or symplectic and Schur forms [10, 18], and structure-preserving versions of matrix iterations [2, 11, 20]. In the next sections, we describe two improvements that can be obtained by using permuted graph bases.

8 Permuted graph bases for the deflation of control problems

A first area where we can see an improvement by judiciously using permuted graph bases is transforming (4) into the form (5). Indeed, consider the following formulation of this deflation process. We premultiply $\mathcal{E}z-\mathcal{A}$ by a suitable matrix to obtain an identity submatrix I_{2n+m} in the first $2n$ columns of $\mathcal{E}$ and the last m columns of $\mathcal{A}$, that is,

$$\mathcal{E}z-\mathcal{A}\sim\begin{bmatrix}0 & I_n & B\\ -I_n & 0 & S\\ 0 & 0 & R\end{bmatrix}^{-1}(\mathcal{E}z-\mathcal{A})=\begin{bmatrix}I_n & 0 & 0\\ 0 & I_n & 0\\ 0 & 0 & 0\end{bmatrix}z-\begin{bmatrix}H_{22} & H_{21} & 0\\ H_{11} & H_{12} & 0\\ R^{-1}B^T & R^{-1}S^T & I\end{bmatrix},$$

where one can see that the H_{ii} are exactly the blocks of H defined in (5), albeit swapped. The rightmost pencil is block-triangular with a leading diagonal block of size $2n\times 2n$ and a trailing one of size $m\times m$; its eigenvalues are given by the union of the eigenvalues of these two diagonal blocks. The trailing $m\times m$ block contains the m infinite eigenvalues, and the leading $2n\times 2n$ block contains the $2n$ finite eigenvalues that coincide with the eigenvalues of H. The eigenvectors and deflating subspaces can be related as well; we do not go through the details. This construction shows that the process of reducing (4) to (5) can be interpreted as per-

forming an equivalence transformation of $\mathcal{E}z-\mathcal{A}$ that enforces an identity submatrix and then deflating the resulting block-triangular pencil. In view of our previous discussion, it looks natural to try to enforce an identity submatrix in a different choice of columns. A special choice of swap matrices is needed to ensure that the deflated pencil is Hamiltonian. The following result can be obtained extending the previous theory to this particular problem.

Theorem 8 ([36]). *Let $\mathcal{E}z-\mathcal{A}$ be a row-reduced pencil with $\mathcal{E}$, $\mathcal{A}$ the two matrices in* (4)*. There exist matrices M_{ij} such that*

$$\mathcal{E}z-\mathcal{A}\sim\begin{bmatrix}M_{11}&M_{12}&0\\M_{12}&M_{22}&0\\M_{31}&M_{32}&0\end{bmatrix}z+\begin{bmatrix}M_{13}&M_{14}&0\\M_{23}&M_{24}&0\\M_{33}&M_{34}&I_m\end{bmatrix},$$

where

$$\begin{bmatrix}0&I_n&A&0\\-I_n&0&Q&-A^T\\0&0&S^T&-B^T\end{bmatrix}^H\sim\begin{bmatrix}M_{11}&M_{12}&-M_{14}&M_{13}\\M_{21}&M_{22}&-M_{24}&M_{23}\\M_{31}&M_{32}&-M_{34}&M_{33}\end{bmatrix}^H=S\begin{bmatrix}I_n&0&X_{11}&X_{12}\\0&I_n&X_{21}&X_{22}\\0&0&X_{31}&X_{32}\end{bmatrix}^H,$$

for some $S\in\mathscr{S}_{2n}$, and $X=\begin{bmatrix}X_{11}&X_{12}\\X_{21}&X_{22}\end{bmatrix}$ symmetric and such that $\|X\|_{\max}\leq 1$.

An explicit algorithm to obtain X with $\|X\|_{\max}\leq\tau$ for each $\tau\geq 1$ and an initial permutation heuristic inspired by the rank-revealing QRP factorization are provided in [36].

If one performs deflation in this form, $\begin{bmatrix}M_{11}&M_{12}\\M_{21}&M_{22}\end{bmatrix}z-\begin{bmatrix}M_{13}&M_{14}\\M_{23}&M_{24}\end{bmatrix}$ is a Hamiltonian pencil left equivalent to $I_{2n}z-H$, already in the format given by Theorem 6.

We report an example with a pencil that is particularly troublesome for most numerical methods. Let $m=n=1$, and

$$\mathcal{E}z-\mathcal{A}=\begin{bmatrix}0&1&0\\-1&0&0\\0&0&0\end{bmatrix}z-\begin{bmatrix}0&0&1\\0&0&0\\1&0&\varepsilon\end{bmatrix}. \tag{7}$$

Then,

$$\mathcal{E}z-\mathcal{A}\sim\begin{bmatrix}0&0&-1\\-1&0&0\\0&-1&-\varepsilon\end{bmatrix}^{-1}(\mathcal{E}z-\mathcal{A})=\begin{bmatrix}1&0&0\\0&\varepsilon&0\\0&-1&0\end{bmatrix}z+\begin{bmatrix}0&0&0\\1&0&0\\0&0&1\end{bmatrix}.$$

The deflated pencil is $\begin{bmatrix}1&0\\0&\varepsilon\end{bmatrix}z+\begin{bmatrix}0&0\\1&0\end{bmatrix}$, which is in the form of Theorem 6 with $S=S_2$ and $X=\begin{bmatrix}0&0\\0&-\varepsilon\end{bmatrix}$. Note that the procedure can be performed without trouble even if ε is very small or zero. Several methods for the deflation of an even problem (4) to a Hamiltonian one have appeared in literature [27, 28, 34, 44, 49]; in most of them, it is required either that R is nonsingular, or that the kernel of R (and possibly further kernels) are determined accurately. Rank decisions on R have often been considered a crucial part of the deflation procedure; the method outlined here shows instead

that it is not the case, and that a Hamiltonian problem can be produced in a stable way without worrying about its singularity (or closeness to singularity).

If $\varepsilon = 0$, then the pencil (7) is singular, although both $\mathcal{E}z - \mathcal{A}$ and its transpose are row-reduced; correspondingly, the deflated Hamiltonian pencil is singular, too. So, from a computational point of view, we did not eliminate the problem of singularity, but simply push it to a later stage. Numerical methods for Hamiltonian eigenproblems that do not break down for singular (and close-to-singular) pencils are then required; as far as we know they have not yet appeared in the literature.

9 Hamiltonian pencils and the doubling algorithm

A second application of permuted graph bases comes from solving Hamiltonian invariant subspace problems. The starting point for our algorithm is the following result.

Theorem 9 ([6]). *Let $L(z) = L_1 z + L_0 \in \mathbb{C}[z]^{n\times n}$ be a regular pencil, and let $M_0, M_1 \in \mathbb{C}^{n\times n}$ be such that $\begin{bmatrix}-M_0 & M_1\end{bmatrix}$ has full row rank and $\begin{bmatrix}-M_0 & M_1\end{bmatrix}\begin{bmatrix}L_1\\ L_0\end{bmatrix} = 0$. If $v \in \mathbb{C}^n$ is an eigenvector of $L(z)$ with eigenvalue λ, then it is also an eigenvector of the pencil*

$$N(z) = N_1 z + N_0 = M_0 L_1 z + \frac{1}{2}(M_1 L_1 + M_0 L_0) \tag{8}$$

with eigenvalue $f(\lambda)$, where $f(z) = \frac{1}{2}(z + z^{-1})$.

If we denote by $f^{(k)}$ the composition of f with itself k times and by $\Re z$ the real part of z, we have

$$\lim_{k\to\infty} f^{(k)}(z) = \begin{cases} 1 & \text{if } \Re z > 0, \\ -1 & \text{if } \Re z < 0 \end{cases}$$

(the iteration in this form breaks down if $\Re z = 0$, but as we see in the following this will not be a concern). Hence, if we start from a pencil $L(z)$ with no eigenvalues on the imaginary axis, repeating the transformation $L(z) \mapsto N(z)$, we converge (in a suitable sense) to a pencil $L_\infty(z)$ with eigenvalues 1 and -1 only, from which one can recover the invariant subspace associated with the eigenvalues having negative real part. This iteration is essentially a pencil version of the matrix sign iteration [26, Chapter 5].

Note that one can replace $M(z) = M_1 z + M_0$ with any pencil $M'(z) \sim M(z)$, obtaining then a different $N'(z) \sim N(z)$; so there is some arbitrariness in how to perform the iteration. Some form of normalization needs to be enforced, otherwise N_1 and N_0 could both converge to zero, or diverge, or (even worse) converge to matrices with the same left kernel, giving a non-row-reduced $L_\infty(z)$. Hence one can see a role for permuted graph representations in this setting. A second point in which this technique helps is in computing the kernel $\begin{bmatrix}-M_0 & M_1\end{bmatrix}$. Indeed, the following result is easy to verify.

Lemma 3. *Let $U \sim P\mathscr{G}(X)$ be the permuted graph representation of a matrix $U \in \mathbb{C}^{(m+n)\times m}$ with full column rank. Then, $W = P\begin{bmatrix} -X^H \\ I_n \end{bmatrix} \in \mathbb{C}^{(m+n)\times n}$ is such that $W^H U = 0$. Moreover, the matrix W has full column rank and spans the kernel of U^H.*

Hence, given a permuted graph basis for a subspace, we can determine a permuted graph basis for its left kernel with basically no computational effort.

Another important observation is that in Theorem 9 whenever $L(z)$ is Hamiltonian, then $N(z)$ is Hamiltonian, too, so we can compute at each step a permuted Lagrangian graph representation as normalization. Putting everything together, we get Algorithm 4. The bulk of the computational cost consists in computing per-

Input: A Hamiltonian pencil $L(z) = L_1 z + L_0 \in \mathbb{C}[z]^{2n\times 2n}$ without eigenvalues on the imaginary axis; a threshold $\tau > \sqrt{2}$
Output: A basis for the invariant subspace $\mathscr{U}$ of $L(z)$ associated with the eigenvalues in the left half-plane
repeat
  compute a permutation matrix $P \in \mathbb{C}^{4n\times 4n}$ and $X \in \mathbb{C}^{2n\times 2n}$ such that $P\mathscr{G}(X) \sim \begin{bmatrix} L_1 \\ L_0 \end{bmatrix}$ and $\|X\|_{\max} \le \tau$, using Algorithm 1;
  let $\begin{bmatrix} -M_0 & M_1 \end{bmatrix} = \begin{bmatrix} -X & I \end{bmatrix} P^H$;
  compute $N(z)$ as in (8);
  compute $S \in \mathscr{S}_{4n}, Y = Y^H \in \mathbb{C}^{2n\times 2n}$ such that $\begin{bmatrix} N_1 & N_0 J_{2n} \end{bmatrix}^H \sim S\mathscr{G}(Y)$ and $\|Y\|_{\max} \le \tau$, using the symplectic analogue of Algorithm 1 (see Theorem 6 and Section 5);
  replace $L(z)$ with $N(z)$ and continue;
until *Y converges*;
Find the kernel of $L_1 + L_0$, which is $\mathscr{U}$;

Algorithm 4: Inverse-free sign algorithm with permuted graph bases [36]

muted graph bases for $\begin{bmatrix} L_1 \\ L_0 \end{bmatrix}$ and permuted Lagrangian graph bases for $\begin{bmatrix} N_1 & N_0 J_{2n} \end{bmatrix}^H$, alternately, together with the matrix products that appear in (8). At each step after the first, P and S from the previous steps typically work well as initial guesses; recomputing X and Y from the permutation at the end of Algorithm 1 might be needed for better accuracy.

The algorithm converges quadratically; one can relax the assumptions, allowing for eigenvalues on the imaginary axis; in this case, the algorithm can be proved to converge in every problem for which there exists a Lagrangian deflating subspace [35, 36], but the convergence rate turns to linear. (Actually, Theorem 9 and our analysis above are slightly incomplete even in the case with no eigenvalues on the imaginary axis, because we do not consider what happens to multiple eigenvalues; we refer the reader to [35, 36] for full detail.)

There are essentially two versions of this algorithm; one is as described above; the other one works by first converting $L_1 z + L_0$ to a symplectic pencil via the transformation $L(z) \mapsto (L_1 + L_0)z + (L_0 - L_1)$ (known as *Cayley transform*), and then applying a transformation analogous to (8), that is,

$$N(z) = M_1 L_1 z - M_0 L_0, \tag{9}$$

which transforms the eigenvalues according to $g(z) = z^2$, and for which

$$\lim_{k\to\infty} g^{(k)}(z) = \begin{cases} \infty & \text{if } |z| > 1, \\ 0 & \text{if } |z| < 1. \end{cases}$$

In this second case, the last line of Algorithm 4 changes to computing the kernel of L_0. The work [6] contains a general theory of operations with matrix pencils that describes how to produce "matrix pencil versions" of rational functions, such as (8) and (9) for $f(z)$ and $g(z)$.

This modified version is called *doubling algorithm*; it was introduced (with a different derivation) for unstructured invariant subspace problems without the use of permuted graph bases in [4, 33], and for symplectic problems with the special choice $S = I$ (graph basis without permutation) in [2, 11, 12, 29], and then generalized to make full use of permuted graph bases in [35]. The algorithm that we described first is known as *inverse-free sign method*; it appeared without the use of permuted graph bases in [6], then with $S = I$ in [20], and with permuted graph bases in [36]. Permuted graph bases are important here because they ensure that the iterative procedure produces an exactly Hamiltonian (or symplectic) pencil at each step and steers clear of numerically singular pencils.

A basic implementation in the MATLAB language of Algorithm 4 and its doubling variant is available on [43]; the library also includes Algorithm 1 and several functions to compute permuted graph representations of subspaces and pencils, both in the unstructured and the Lagrangian case.

How well do these algorithms fare in practice, compared to their many competitors and variants that do not make use of permuted graph bases? The work [35] reports computational results on a set of 33 small-scale problems (the same test problems used in [10]) obtained from the benchmark set [7]. This is a benchmark set containing examples of linear-quadratic control problems; it contains both examples from real-life applications and challenging problems created ad-hoc to be difficult to solve. As far as we know, the algorithm in [35] (Algorithm 4 in the variant with transformation (9)) is the first numerical algorithm to obtain completely satisfying results in all 33 problems on both these grounds:

- small subspace residual, that is, $\frac{\|(I-UU^T)HU\|}{\|H\|}$ of the order of machine precision for the computed subspace U;
- exact preservation of the Lagrangian structure, that is, $U^T J_{2n} U$ either zero or of the order of machine precision.

Algorithm 4 in the variant presented here was tested on another challenging application ($\mathscr{H}_\infty$ control, an optimization procedure which requires solving one after another a set of close-to-unsolvable Riccati equations) in [36]; the results suggest that the variant (8) is more stable than (9), because it avoids the initial Cayley transform. This is why we chose to highlight (8) in this presentation.

Explicit theoretical results proving stability of the algorithm are still an open issue, though. For methods based on orthogonal transformations and reduction to Schur form, the standard technique is a Wilkinson-style backward stability proof ([50, Chapter 3] and [25, Section 19.3]); however, a counterexample in [35, p. 798] shows that the simplest version of a backward stability proof using this technique would not work for doubling-type algorithms. As far as we know, the only stability proof for an algorithm of this family is given in [4], for a doubling algorithm for unstructured pencils based on orthogonal bases. It can be adapted to the other variants; however, it is not completely satisfying as the error growth coefficient is bounded by $1/d^2$, where d is the distance to the closest ill-posed problem, instead of the more natural $1/d$ which constitutes the condition number of the problem. We are not aware of an example in which this larger factor shows up in practice. Another related result is the work in [38, 39], which shows that for Hermitian matrices a carefully chosen variant of doubling achieves a mixed variant of backward and forward stability. Nevertheless, for nonsymmetric problems, proving the stability of doubling-type algorithms is still an open problem, both in the structured and non-structured case.

10 Research directions

There are many possible improvements and open problems related to these topics; some of them have already been presented in our exposition, and we collect here a few more.

An interesting issue is the significance of $\kappa(\left[\begin{smallmatrix} L_1 \\ L_0 \end{smallmatrix}\right])$ for a pencil $L_1 z + L_0$: what is its role in the stability and perturbation theory of the stable invariant subspace problem?

Another research direction is extending the methods presented in Sections 8 and 9 to deal with so-called *descriptor systems*, a common variant of the control theory setting described above. The matrix pencils appearing in such problems are similar to the one in (4), but the leading submatrix J_{2n} of $\mathcal{E}$ is replaced by $\left[\begin{smallmatrix} 0 & E \\ -E^T & 0 \end{smallmatrix}\right]$, for a matrix $E \in \mathbb{C}^{n\times n}$ that may be singular. With this modification, only part of the structure is preserved: the resulting pencil $\mathcal{E}z - \mathcal{A}$ is still even, but the deflated problem is not Hamiltonian and its stable deflating subspace $\mathscr{U}$ is not Lagrangian. The algorithms that we have presented rely in an essential way on these structures, so modifying them to work with descriptor systems will probably require major changes.

A first attempt to use permuted graph bases in large-scale control problems is in the recent preprint [37]; the underlying algorithm is not a doubling algorithm but low-rank ADI [8], and inverse-free techniques and permuted graph bases are used to adapt it to an invariant subspace formulation. One of the most interesting observations in that work is that Lemma 3, which we have used here only in the case $m = n$, is noteworthy also when $m \ll n$, since it allows one to compute an

explicit basis for a large n-dimensional subspace of $\mathbb{C}^{m+n}$ defined as a kernel with basically no effort.

Overall, in control theory there are many possibilities for further applications; many problems can be reduced to the computation of some Lagrangian invariant subspace, and permuted graph bases are a natural choice for this task.

11 Conclusions

In this chapter we have presented the basic theory of permuted graph matrices and bases, and shown how techniques based on them are useful in a variety of applications. We believe that they are an interesting alternative, in selected problems, to the ubiquitous orthogonal matrices. They have led to the construction of several efficient algorithms for various tasks, and, as always in research, there is plenty of open problems and possibilities for improvement.

Acknowledgements The author is grateful to C. Mehl, B. Meini and N. Strabič for their useful comments on an early version of this chapter.

References

1. P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, Princeton, NJ, 2008.
2. B. Anderson. Second-order convergent algorithms for the steady-state Riccati equation. *Internat. J. Control*, 28(2):295–306, 1978.
3. M. Arioli and I. S. Duff. Preconditioning of linear least-squares problems by identifying basic variables. Technical Report RAL-P-2014-007s, Rutherford Appleton Laboratory, 2014.
4. Z. Bai, J. Demmel, and M. Gu. An inverse free parallel spectral divide and conquer algorithm for nonsymmetric eigenproblems. *Numer. Math.*, 76(3):279–308, 1997.
5. P. Benner and R. Byers. An exact line search method for solving generalized continuous-time algebraic Riccati equations. *IEEE Trans. Automat. Control*, 43(1):101–107, 1998.
6. P. Benner and R. Byers. An arithmetic for matrix pencils: theory and new algorithms. *Numer. Math.*, 103(4):539–573, 2006.
7. P. Benner, A. Laub, and V. Mehrmann. A collection of benchmark examples for the numerical solution of algebraic Riccati equations I: the continuous-time case. Technical Report SPC 95-22, Forschergruppe 'Scientific Parallel Computing', Fakultät für Mathematik, TU Chemnitz-Zwickau, 1995. Version dated February 28, 1996.
8. P. Benner, J.-R. Li, and T. Penzl. Numerical solution of large-scale Lyapunov equations, Riccati equations, and linear-quadratic optimal control problems. *Numer. Linear Algebra Appl.*, 15(9):755–777, 2008.
9. D. A. Bini, B. Iannazzo, and B. Meini. *Numerical solution of algebraic Riccati equations*, volume 9 of *Fundamentals of Algorithms*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2012.
10. D. Chu, X. Liu, and V. Mehrmann. A numerical method for computing the Hamiltonian Schur form. *Numer. Math.*, 105(3):375–412, 2007.
11. E. K.-W. Chu, H.-Y. Fan, and W.-W. Lin. A structure-preserving doubling algorithm for continuous-time algebraic Riccati equations. *Linear Algebra Appl.*, 396:55–80, 2005.

12. E. K.-W. Chu, H.-Y. Fan, W.-W. Lin, and C.-S. Wang. Structure-preserving algorithms for periodic discrete-time algebraic Riccati equations. *Internat. J. Control*, 77(8):767–788, 2004.
13. A. Çivril and M. Magdon-Ismail. On selecting a maximum volume sub-matrix of a matrix and related problems. *Theoret. Comput. Sci.*, 410(47-49):4801–4811, 2009.
14. G. Dantzig and M. Thapa. *Linear programming. 1*. Springer Series in Operations Research. Springer-Verlag, New York, 1997.
15. H. S. Dollar. Constraint-style preconditioners for regularized saddle point problems. *SIAM J. Matrix Anal. Appl.*, 29(2):672–684, 2007.
16. H. S. Dollar and A. J. Wathen. Approximate factorization constraint preconditioners for saddle-point matrices. *SIAM J. Sci. Comput.*, 27(5):1555–1572, 2006.
17. F. Dopico and C. Johnson. Complementary bases in symplectic matrices and a proof that their determinant is one. *Linear Algebra Appl.*, 419(2-3):772–778, 2006.
18. H. Fassbender. *Symplectic methods for the symplectic eigenproblem*. Kluwer Academic/Plenum Publishers, New York, 2000.
19. F. R. Gantmacher. *The theory of matrices. Vols. 1, 2*. Translated by K. A. Hirsch. Chelsea Publishing Co., New York, 1959.
20. J. D. Gardiner and A. J. Laub. A generalization of the matrix-sign-function solution for algebraic Riccati equations. *International Journal of Control*, 44(3):823–832, 1986.
21. G. Golub and C. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, third edition, 1996.
22. S. A. Goreinov, I. V. Oseledets, D. V. Savostyanov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. How to find a good submatrix. In *Matrix methods: theory, algorithms and applications*, pages 247–256. World Sci. Publ., Hackensack, NJ, 2010.
23. S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin. A theory of pseudoskeleton approximations. *Linear Algebra Appl.*, 261:1–21, 1997.
24. C.-H. Guo and P. Lancaster. Analysis and modification of Newton's method for algebraic Riccati equations. *Math. Comp.*, 67(223):1089–1105, 1998.
25. N. Higham. *Accuracy and stability of numerical algorithms*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, second edition, 2002.
26. N. J. Higham. *Functions of matrices. Theory and computation*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2008.
27. V. Ionescu, C. Oară, and M. Weiss. *Generalized Riccati theory and robust control. A Popov function approach*. John Wiley & Sons Ltd., Chichester, 1999.
28. D. Jacobson and J. Speyer. Necessary and sufficient conditions for optimality for singular control problems; a transformation approach. *J. Math. Anal. Appl.*, 33:163–187, 1971.
29. M. Kimura. Convergence of the doubling algorithm for the discrete-time algebraic Riccati equation. *Internat. J. Systems Sci.*, 19(5):701–711, 1988.
30. D. Kleinman. On an iterative technique for Riccati equation computations. *Automatic Control, IEEE Transactions on*, 13(1):114–115, Feb 1968.
31. D. E. Knuth. Semioptimal bases for linear dependencies. *Linear and Multilinear Algebra*, 17(1):1–4, 1985.
32. W.-W. Lin, V. Mehrmann, and H. Xu. Canonical forms for Hamiltonian and symplectic matrices and pencils. *Linear Algebra Appl.*, 302/303:469–533, 1999. Special issue dedicated to Hans Schneider (Madison, WI, 1998).
33. A. N. Malyshev. Calculation of invariant subspaces of a regular linear matrix pencil. *Sibirsk. Mat. Zh.*, 30(4):76–86, 217, 1989.
34. V. Mehrmann. *The autonomous linear quadratic control problem*, volume 163 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, Berlin, 1991. Theory and numerical solution.
35. V. Mehrmann and F. Poloni. Doubling algorithms with permuted Lagrangian graph bases. *SIAM J. Matrix Anal. Appl.*, 33(3):780–805, 2012.
36. V. Mehrmann and F. Poloni. Using permuted graph bases in $\mathscr{H}_\infty$ control. *Automatica J. IFAC*, 49(6):1790–1797, 2013.

37. V. Mehrmann and F. Poloni. An inverse-free ADI algorithm for computing Lagrangian invariant subspaces. Technical Report 14-2014, Institute of Mathematics, Technische Universität Berlin, 2014.
38. 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.
39. Y. Nakatsukasa and N. J. Higham. Stable and efficient spectral divide and conquer algorithms for the symmetric eigenvalue decomposition and the SVD. *SIAM J. Sci. Comput.*, 35(3):A1325–A1349, 2013.
40. V. Noferini and F. Poloni. Duality of matrix pencils, Wong chains and linearizations. MIMS E-prints 2013.17, The University of Manchester, 2014. Version deposited on 7 August 2014.
41. I. V. Oseledets, D. V. Savostyanov, and E. E. Tyrtyshnikov. Cross approximation in tensor electron density computations. *Numer. Linear Algebra Appl.*, 17(6):935–952, 2010.
42. C.-T. Pan. On the existence and computation of rank-revealing *LU* factorizations. *Linear Algebra Appl.*, 316(1-3):199–222, 2000. Conference Celebrating the 60th Birthday of Robert J. Plemmons (Winston-Salem, NC, 1999).
43. F. Poloni. Pgdoubling, 2012. MATLAB library. Available online on https://bitbucket.org/fph/pgdoubling.
44. F. Poloni and T. Reis. A deflation approach for large-scale Lur'e equations. *SIAM J. Matrix Anal. Appl.*, 33(4):1339–1368, 2012.
45. D. V. Savostyanov. Quasioptimality of maximum-volume cross interpolation of tensors. Technical Report arXiv:1305.1818, arXiv.org, 2013.
46. G. W. Stewart and J. G. Sun. *Matrix perturbation theory*. Computer Science and Scientific Computing. Academic Press, Inc., Boston, MA, 1990.
47. M. Tsatsomeros. Principal pivot transforms: properties and applications. *Linear Algebra Appl.*, 307(1-3):151–165, 2000.
48. K. Usevich and I. Markovsky. Optimization on a Grassmann manifold with application to system identification. *Automatica J. IFAC*, 50(6):1656–1662, 2014.
49. H. Weiss, Q. Wang, and J. Speyer. System characterization of positive real conditions. *IEEE Trans. Automat. Control*, 39(3):540–544, 1994.
50. J. H. Wilkinson. *The algebraic eigenvalue problem*. Monographs on Numerical Analysis. The Clarendon Press, Oxford University Press, New York, 1988. Oxford Science Publications.