Balancing Skew-Hamiltonian/Hamiltonian Pencils
With Applications in Control Engineering

Vasile Sima
National Institute for Research & Development in Informatics, 8–10 Bd. Mareşal Averescu, Bucharest, Romania

Keywords: Deflating Subspaces, Generalized Eigenvalues, Numerical Methods, Skew-Hamiltonian/Hamiltonian Matrix Pencil, Software, Structure-preservation.

Abstract: Badly-scaled matrix pencils could reduce the reliability and accuracy of computed results for many numerical problems, including computation of eigenvalues and deflating subspaces, which are needed in many key procedures for optimal and H_∞ control, model reduction, spectral factorization, and so on. Standard balancing techniques can improve the results in many cases, but there are situations when the solution of the scaled problem is much worse than that for the unscaled problem. This paper presents a new structure-preserving balancing technique for skew-Hamiltonian/Hamiltonian matrix pencils, and illustrates its good performance in solving eigenvalue problems and algebraic Riccati equations for large sets of examples from well-known benchmark collections with difficult examples.

1 INTRODUCTION

Computing eigenvalues and bases of certain associated invariant or deflating subspaces of matrices or matrix pencils is central to various numerical techniques in control engineering and other domains. When the corresponding eigenproblems have special structure, implying structural properties of their spectra, it is important to use structure-preserving and/or structure-exploiting algorithms. General numerical algorithms cannot ensure that the theoretical properties are preserved during computations (Paige and Van Loan, 1981; Van Loan, 1984; Kressner, 2005). Common special structures are Hamiltonian and symplectic matrices or matrix pencils, which are encountered in optimal or H_∞ control (e.g., solution of algebraic Riccati equations, or evaluation of the H_∞- and L_∞-norms of linear dynamic systems), spectral factorization, model reduction, etc., see, for instance, (Bruinsma and Steinbuch, 1990; Laub, 1979; Mehrmann, 1991; Sima, 1996). While structured matrices are encountered for standard linear dynamic systems, generalized and descriptor systems may involve structured matrix pencils. Often such pencils can be recast as skew-Hamiltonian/Hamiltonian pencils, for which dedicated algorithms have been developed, see e.g., (Benner et al., 2007; Benner et al., 2002; Benner et al., 2005; Benner et al., 2012a; Benner et al., 2012b; Benner et al., 2013a; Benner et al., 2013b; Jiang and Voigt, 2013), and the references therein.

Quite often, the pencil matrices have large norms and elements with highly different magnitude. Such pencils imply potential numerical difficulties for software implementations of eigensolvers, with negative consequences on the reliability and accuracy of the results, see, e.g., (Sima and Benner, 2015a). Balancing procedures can be used to improve the numerical behavior. Ward (1981) proposed a balancing technique for general matrix pencils, which has been incorporated in state-of-the-art software packages, such as LAPACK (Anderson et al., 1999). (This will be referred below as *standard balancing*.) The data matrices are preprocessed by equivalence transformations, in two optional stages: the first stage uses permutations to find isolated eigenvalues (which are available by inspection, with no rounding errors), and the second stage uses diagonal scaling transformations to make the row and corresponding column 1-norms as close as possible. Balancing may reduce the 1-norm of the scaled matrices, but there is no guarantee in general. This is the reason why full balancing is either avoided or provided as an option in the LAPACK subroutines; some expert driver routines, such as DGEES, DGEESX, DGGES, DGGESX, and DGGEV use permutations only, while other drivers, e.g., DGEEVX and DGGEVX, have an input argument allowing either permutations, scaling, both permutations and scaling, or no balancing at all.

Sima, V.
Balancing Skew-Hamiltonian/Hamiltonian Pencils - With Applications in Control Engineering.
DOI: 10.5220/0005981201770184
In *Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2016) - Volume 1*, pages 177-184
ISBN: 978-989-758-198-4

A structure-preserving balancing technique for Hamiltonian matrices has been developed in (Benner, 2001), but no special procedure has been available for skew-Hamiltonian/Hamiltonian matrix pencils.

This paper presents a new, structure-preserving balancing procedure for skew-Hamiltonian/Hamiltonian pencils, and illustrates its performance for some control engineering applications, in comparison with the general approach, and with the case when no balancing is used. This procedure includes several optional enhancements which allow to get meaningful results when standard balancing (possibly even a structured variant) fails, as shown for some numerical examples.

2 BALANCING SKEW-HAMILTONIAN/ HAMILTONIAN PENCILS

Let $\alpha\mathcal{S} - \beta\mathcal{H}$ be a complex skew-Hamiltonian/ Hamiltonian pencil, with α, $\beta \in \mathbf{C}$, $\mathcal{S} \in \mathbf{C}^{2n\times 2n}$ a skew-Hamiltonian matrix and $\mathcal{H} \in \mathbf{C}^{2n\times 2n}$ a Hamiltonian matrix, i.e.,

$$(\mathcal{S}\mathcal{J})^H = -\mathcal{S}\mathcal{J},\ (\mathcal{H}\mathcal{J})^H = \mathcal{H}\mathcal{J},\ \mathcal{J} := \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix}, \tag{1}$$

where M^H denotes the conjugate transpose of a matrix M, $M^H = \bar{M}^T$, the overbar denotes the complex conjugate, M^T denotes the transpose of M, and I_n is the identity matrix of order n. These definitions imply the following structure for $\mathcal{S}$ and $\mathcal{H}$:

$$\mathcal{S} = \begin{bmatrix} A & D \\ E & A^H \end{bmatrix}, \quad \mathcal{H} = \begin{bmatrix} C & V \\ W & -C^H \end{bmatrix}, \tag{2}$$

where D and E are skew-Hermitian, i.e., $D^H = -D$, $E^H = -E$, and V and W are Hermitian matrices. When $\mathcal{S}$ and $\mathcal{H}$ are real matrices, then D and E are skew-symmetric, V and W are symmetric, the H superscript is replaced by T, and some derivations below simplify.

When the matrices $\mathcal{S}$ and $\mathcal{H}$ are badly scaled, e.g., when their elements have moduli with highly different magnitude, the accuracy of computed solutions of related eigenproblems can be very poor. The accuracy may be improved by using a balancing procedure, as described in (Ward, 1981) for general matrix pencils, and implemented in the LAPACK package (Anderson et al., 1999). Similarly to the techniques in (Anderson et al., 1999; Benner, 2001; Ward, 1981), a structure-exploiting balancing procedure for the structured pencil matrices in (2) would involve, in a first step, permuting $\alpha\mathcal{S} - \beta\mathcal{H}$ by a symplectic equivalence transformation to isolate eigenvalues in the elements $1:\ell-1$ and $n+1:n+\ell-1$ on the diagonals of $\mathcal{S}$ and $\mathcal{H}$, and in a second step, applying a diagonal equivalence transformation to rows and columns $\ell:n$ and $n+\ell:2n$, to make the rows and columns as close in 1-norm as possible. (A MATLAB-style notation for array indexing (MATLAB, 2016) is used.) Both steps above are optional. Balancing may reduce the 1-norms of the matrices $\mathcal{S}$ and $\mathcal{H}$.

Note that it is enough to equilibrate the 1-norms of the rows and columns 1:n of the matrices $\mathcal{S}$ and $\mathcal{H}$, since $\|\mathcal{S}_{i+n,:}\|_p = \|\mathcal{S}_{:,i}\|_p$ and $\|\mathcal{S}_{:,i+n}\|_p = \|\mathcal{S}_{i,:}\|_p$, for any p-norm, for $p \geq 1$, and similarly for $\mathcal{H}$ (see (Benner, 2001)). The balancing procedure performs the following transformation:

$$\widetilde{\mathcal{S}} = \mathcal{L}\mathcal{S}\mathcal{R}, \quad \widetilde{\mathcal{H}} = \mathcal{L}\mathcal{H}\mathcal{R}, \tag{3}$$

where

$$\mathcal{L} = \mathcal{L}_n \cdots \mathcal{L}_\ell \mathcal{P}_{\ell-1}^T \cdots \mathcal{P}_1^T,\ \mathcal{R} = \mathcal{P}_1 \cdots \mathcal{P}_{\ell-1}\mathcal{R}_\ell \cdots \mathcal{R}_n, \tag{4}$$

where $\mathcal{P}_k$, $k = 1:\ell-1$, are symplectic permutation or $\mathcal{J}$-permutation matrices (Benner, 2001), and $\mathcal{L}_k$ and $\mathcal{R}_k$ are left and right diagonal scaling matrices (with $\mathcal{L}_k(k,k)$, $\mathcal{L}_k(k+n,k+n)$, $\mathcal{R}_k(k,k)$, and $\mathcal{R}_k(k+n,k+n)$ the only diagonal elements different from 1), chosen to equilibrate the 1-norms of the k-th row and column of $\mathcal{S}$ and $\mathcal{H}$, for $k = \ell,\cdots,n$. A permutation matrix has exactly one nonzero entry, which is 1, in each row and column. A $\mathcal{J}$-permutation matrix has a similar structure, but the nonzero entries may also be -1. When possible, the permutations are chosen in the form

$$\mathcal{P} = \begin{bmatrix} P & 0 \\ 0 & P \end{bmatrix}, \tag{5}$$

where P is an $n\times n$ permutation matrix (hence, $P^TP = PP^T = I_n$). Clearly, $\mathcal{P}$ is symplectic, since $\mathcal{P}\mathcal{J}\mathcal{P}^T = \mathcal{J}$. As shown in (Benner, 2001), it is generally not possible to preserve the (skew-)Hamiltonian structure using symplectic permutations only, but symplectic $\mathcal{J}$-permutations are also needed.

Applying a permutation (5) in (3) is easy. Specifically, the transformed matrices become P^TAP, P^TDP, P^TEP, P^TCP, P^TVP, and P^TWP.

Consider applying a $\mathcal{J}$-permutation, $\mathcal{P}$, which has the following structure, defined by an index i, $i \leq n$,

$$\mathcal{P} = \begin{bmatrix} I_{i-1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & I_{n-i} & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{i-1} & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{n-i} \end{bmatrix}, \tag{6}$$

to the skew-Hamiltonian matrix $\mathcal{S}$, and Hamiltonian matrix $\mathcal{H}$. Using the partition of $\mathcal{H}$ in (2), the transformed matrix $\mathcal{P}^T\mathcal{H}\mathcal{P}$ is

$$\mathcal{P}^T\mathcal{H}\mathcal{P} = \left[\begin{array}{ccc} \star & -v_{1:i-1,i} & \star \\ -w_{i,1:i-1} & -\bar{c}_{ii} & -w^H_{i+1:n,i} \\ \star & -v^H_{i,i+1:n} & \star \\ \star & c^H_{i,1:i-1} & \star \\ c_{i,1:i-1} & -v_{ii} & c_{i,i+1:n} \\ \star & c^H_{i,i+1:n} & \star \end{array}\right. \left.\begin{array}{ccc} \star & c_{1:i-1,i} & \star \\ c^H_{1:i-1,i} & -w_{ii} & c^H_{i+1:n,i} \\ \star & c_{i+1:n,i} & \star \\ \star & w^H_{i,1:i-1} & \star \\ v^H_{1:i-1,i} & c_{ii} & v_{i,i+1:n} \\ \star & w_{i+1:n,i} & \star \end{array}\right], \quad (7)$$

where $\star$ denotes submatrices which do not change. Matrix $\mathcal{S}$ is transformed similarly.

The permutation procedure first scans the columns $1:n$ and then the rows $1:n$ of $\mathcal{S}$ and $\mathcal{H}$, and uses row and column permutations with matrices of the form (5) and (6) so that $\widetilde{\mathcal{S}}$ and $\widetilde{\mathcal{H}}$ finally have the first $\ell-1$ columns upper triangular, i.e., with zeros below the diagonals. This way, the first $\ell-1$ eigenvalues of the pencil are defined by the diagonal elements $1:\ell-1$ in $\widetilde{\mathcal{S}}$ and $\widetilde{\mathcal{H}}$. The upper triangular structure is built column by column, starting from the first. Therefore, the permutations above can take advantage of the zero subdiagonal part in the already processed columns. For instance, when a $\mathcal{J}$-permutation (6) is formed and applied, the vectors $c_{i,1:i-1}$, and $w_{i,1:i-1}$ in (7) (and similarly for $a_{i,1:i-1}$, $e_{i,1:i-1}$ in $\mathcal{S}$), as well as all $\star$ vectors under the diagonals in the first $i-1$ columns are already zero.

The balancing procedure for (skew-)Hamiltonian matrices must use *symplectic scaling* matrices as well, in order to preserve the structure (Benner, 2001). Fortunately, this is not needed for skew-Hamiltonian/Hamiltonian pencils, if diagonal scaling is used, as mentioned above. Indeed, without loss of generality, consider that permutations are not wanted, and only scaling should be applied. Let L and R be the real diagonal matrices which scale the first n rows and the first n columns, respectively, of $\mathcal{S}$ and $\mathcal{H}$. Since R and L will be the scaling matrices for the last n rows and columns, respectively, the global scaling transformation is

$$\begin{aligned} \widetilde{\mathcal{S}} &= \begin{bmatrix} L & 0 \\ 0 & R \end{bmatrix} \begin{bmatrix} A & D \\ E & A^H \end{bmatrix} \begin{bmatrix} R & 0 \\ 0 & L \end{bmatrix}, \\ \widetilde{\mathcal{H}} &= \begin{bmatrix} L & 0 \\ 0 & R \end{bmatrix} \begin{bmatrix} C & V \\ W & -C^H \end{bmatrix} \begin{bmatrix} R & 0 \\ 0 & L \end{bmatrix}. \quad (8) \end{aligned}$$

Then,

$$\begin{aligned} \widetilde{\mathcal{S}}\mathcal{J} &= \begin{bmatrix} LAR & LDL \\ RER & RA^HL \end{bmatrix} \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix} \\ &= \begin{bmatrix} -LDL & LAR \\ -RA^HL & RER \end{bmatrix} = -(\widetilde{\mathcal{S}}\mathcal{J})^H, \quad (9) \end{aligned}$$

since $D=-D^H$, $E=-E^H$. Similarly,

$$\begin{aligned} \widetilde{\mathcal{H}}\mathcal{J} &= \begin{bmatrix} LCR & LVL \\ RWR & -RC^HL \end{bmatrix} \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix} \\ &= \begin{bmatrix} -LVL & LCR \\ RC^HL & RWR \end{bmatrix} = (\widetilde{\mathcal{H}}\mathcal{J})^H, \quad (10) \end{aligned}$$

since $V=V^H$ and $W=W^H$. Therefore, the structure of $\mathcal{S}$ and $\mathcal{H}$ is preserved for the transformation in (8). Note that the left and right transformations are not symplectic if $L \neq R^{-1}$, but they are structure-preserving, for diagonal L and R. The permutations used for isolating the eigenvalues must, however, be symplectic.

As described above, the elementary scaling matrices in (4) are, for $k=\ell,\cdots,n$,

$$\begin{aligned} \mathcal{L}_k &= \mathrm{bl\_diag}(I_{k-1}, l_k, I_{n-1}, r_k, I_{n-k}), \\ \mathcal{R}_k &= \mathrm{bl\_diag}(I_{k-1}, r_k, I_{n-1}, l_k, I_{n-k}), \quad (11) \end{aligned}$$

where bl_diag$(X,Y,\ldots)$ denotes a bloc-diagonal matrix with diagonal blocks X, Y, etc. Note that

$$\mathcal{L}^{-1} = \mathcal{P}_1 \cdots \mathcal{P}_{\ell-1}\mathcal{L}_\ell^{-1} \cdots \mathcal{L}_n^{-1}. \quad (12)$$

Formula (12) can be used to apply, from the left, the back balancing transformations to a given $2n \times m$ matrix. Since $\mathcal{L}^{-1}$ has a formula similar to $\mathcal{R}$, to obtain $\mathcal{R}$, one can apply to I_{2n} the transformations in (12), with scaling factors l_k and r_k replaced by $1/r_k$ and $1/l_k$, respectively. There is no need for an algorithm to apply the right back transformation.

A special case of application of the balancing transformations appears when computing the solution of algebraic Riccati equations (AREs) using the skew-Hamiltonian/Hamiltonian approach, see (Benner et al., 2013a), and the references therein. This approach uses a basis for the stable right deflating subspace of the matrix pencil $\alpha\mathcal{S}-\beta\mathcal{H}$, where $\mathcal{S}$ and $\mathcal{H}$ are suitably defined in terms of the dynamic system and performance index matrices. For convenience, assume that $\mathcal{S}=I_{2n}$, which is the case for standard continuous-time dynamic systems, e.g., when the control weighting matrix of the performance index is well-conditioned. (The general case will be briefly described below.) Let $\widetilde{\mathcal{U}} = \begin{bmatrix} \widetilde{U}_1^T & \widetilde{U}_2^T \end{bmatrix}^T$ be a basis of the stable right deflating subspace of $\alpha\widetilde{\mathcal{S}}-\beta\widetilde{\mathcal{H}}$, corresponding to the balanced pencil. The stabilizing solution of the balanced ARE is given by $\widetilde{X}=\widetilde{U}_2\widetilde{U}_1^{-1}$.

Note that $\widetilde{\mathcal{U}}$ is related to a basis, $\mathcal{U}$, of the stable right deflating subspace of the original pencil, $\alpha\mathcal{S}-\beta\mathcal{H}$, by the transformation $\widetilde{\mathcal{U}} = \mathrm{bl\_diag}(R^{-1},L^{-1})\mathcal{U}$, hence $\left[\begin{array}{cc} U_1^T & U_2^T \end{array}\right]^T := \mathcal{U} = \mathrm{bl\_diag}(R,L)\widetilde{\mathcal{U}}$. Therefore, the stabilizing solution of the original ARE can be computed as follows

$$X = U_2U_1^{-1} = L\widetilde{U}_2\widetilde{U}_1^{-1}R^{-1}. \qquad (13)$$

Formula (13) allows to represent and use the solution X in a factored form, which may be useful for numerical reasons. If balancing also involves symplectic permutations of the form (5) only, then the right hand side in (13) becomes $PL\widetilde{U}_2\widetilde{U}_1^{-1}R^{-1}P^T$, where P denotes here the product of the $\ell-1$ permutations performed. Such factorization, is, however, not possible if $\mathcal{J}$-permutations are also needed, in which case it is necessary to apply the balancing transformations before solving the set of linear systems giving $\widetilde{X}$.

If $\mathcal{S}$ is a general matrix, the order of $\mathcal{S}$ and $\mathcal{H}$ is $2(n+p)$, where n is the dimension of the state vector, and p may be chosen as $p=\lceil m/2\rceil$ (i.e., $p=m/2$, if m is even, and $p=(m+1)/2$, otherwise), with m the number of system inputs. In this case, the computations are similar, but $\widetilde{U}_i$ and U_i, $i=1,2$, refer to the lines $1:n$ and $n+p+1:2n+p$ of the matrix bases $\widetilde{\mathcal{U}}$ and $\mathcal{U}$, respectively.

3 IMPLEMENTATION ISSUES

The developed balancing algorithm operates only on the matrices A, D, E, C, V, and W, and preserves the pencil structure. Moreover, the pairs of skew-Hermitian matrices, D and E, and Hermitian matrices, V and W, are stored compactly in two $n\times(n+1)$ arrays, `DE` and `VW`. Specifically, the lower triangle of E and the upper triangle of D are concatenated along their main diagonals, and similarly for W and V:

$$\begin{array}{rcll} \mathtt{DE}_{ij} &=& e_{ij}, & j=1:n, \quad i\geq j,\\ \mathtt{DE}_{i,j+1} &=& d_{ij}, & j=1:n, \quad i\leq j,\\ \mathtt{VW}_{ij} &=& w_{ij}, & j=1:n, \quad i\geq j,\\ \mathtt{VW}_{i,j+1} &=& v_{ij}, & j=1:n, \quad i\leq j. \end{array} \qquad (14)$$

This way, the storage requirements are reduced by $4n^2-2n$ memory locations, in comparison with the general algorithm in (Anderson et al., 1999), which needs $8n^2$ storage space for $\mathcal{S}$ and $\mathcal{H}$. Note that in the real case, the diagonal elements of D and E, d_{ii} and e_{ii}, $i=1:n$, which should be zero, by definition, are not stored and not used. However, in the complex case, d_{ii} and e_{ii}, $i=1:n$, should have zero real parts, while the imaginary parts might be nonzero; moreover, the diagonal elements of V and W should have zero imaginary parts, but possibly nonzero real parts. The implementation takes care of the compact storage scheme when computing $P^TDP, \ldots, P^TWP$.

Consider a structured pencil, $(\mathcal{S},\mathcal{H})$, and assume that $\mathcal{S}=I_{2n}$. The order of magnitude of the differences in size of the nonzero elements in the $\mathcal{H}$ matrix can be huge. For instance, for the CM6 and CM6_IS examples from the COMPl$_e$ib collection (Leibfritz and Lipinski, 2003), the maximum and minimum absolute values of the nonzero entries are about $2.53\cdot10^5$ and $4.9407\cdot10^{-324}$, hence their ratio is not representable in the usual double precision representation, and it is evaluated as Inf (∞). The standard balancing algorithm implemented in the LAPACK Library subroutine `DGGBAL` (for matrix pencils) returns scaling factors covering a very large range of magnitudes, namely with maximum and minimum values 10^{159} and 10^{-47}, respectively, for both left and right scaling factors. No eigenvalue can be isolated. The scaling transformation matrices, $\mathcal{L}$ and $\mathcal{R}$, have condition numbers with values 10^{206}. The 1-norms of scaled matrices, $\widetilde{\mathcal{H}}$ and $\widetilde{\mathcal{S}}$, are about 10^{228} and 10^{184}, respectively, while the 1-norms of the original $\mathcal{H}$ and $\mathcal{S}$ are $6.9123\cdot10^5$ and 1. Computing the eigenvalues or deflating subspaces using the balanced matrices would return results very far from the true ones. CM6 and CM6_IS are not the only examples in the COMPl$_e$ib collection which produce numerical troubles for eigenproblem-related computations. The CM5 and CM5_IS examples have also a huge ratio, 10^{279}, between the magnitudes of their elements. Other very large ratios are 10^{141}, for CM4 and CM4_IS, 10^{72}, for CM3 and CM3_IS, or 10^{37}, for CM2 and CM2_IS.

The proposed algorithm uses an adaptation of the basic LAPACK procedure for finding the scaling factors, but optionally limits the range of their variation, possibly via an outer loop. Specifically, the user can set a threshold value, τ; if $\tau\geq0$, the entries whose absolute values are smaller than τM_0, where $M_0=\max(\|\mathcal{H}(s,s)\|_1,\|\mathcal{S}(s,s)\|_1)$, with $s:=\ell:n\cup n+\ell:2n$, are considered negligible, and do not count for computing the scaling factors. (For the CM6 and CM6_IS examples, $\ell=1$, and setting $\tau=10^{-20}$, all entries with magnitude less than about $7\cdot10^{-15}$ will be taken as zero by the procedure.) If $\tau<0$ on entry, an outer loop over a set of values $\tau_i>0$ will enable to select a set of scaling factors which, if possible, ensure the reduction of a desired norm-related measure for the scaled matrices. For $\tau=-1$, this measure is the minimum of $\max_i(\|\mathcal{H}_i(s,s)\|_1/\|\mathcal{S}_i(s,s)\|_1,\|\mathcal{S}_i(s,s)\|_1/\|\mathcal{H}_i(s,s)\|_1)$ where $\mathcal{H}_i(s,s)$ and $\mathcal{S}_i(s,s)$ are the scaled submatrices corresponding to the threshold τ_i. This strategy tries to balance $\mathcal{H}$ and $\mathcal{S}$, but also have comparable 1-norms. For $\tau = -2$, the same measure is used, but if $\max(\|\widetilde{\mathcal{H}}(s,s)\|_1, \|\widetilde{\mathcal{S}}(s,s)\|_1) > cM_0$ and $t > T$, where c and T are given constants (c possibly larger than 1), and t is the maximum ratio of the scaling factors found (the maximum of the condition numbers of $\mathcal{L}$ and $\mathcal{R}$), then the scaling factors are set to 1 and a warning indicator is set; here, the matrices with tilde accent are the solution of the above norm ratio reduction problem. This approach avoids to obtain scaled matrices with too large norms, compared to the given ones, and also limits the range of the scaling factors. For $\tau = -3$, the measure used is the smallest product of norms, $\min_i(\|\mathcal{H}_i(s,s)\|_1\|\mathcal{S}_i(s,s)\|_1)$, over the sequence of τ_i values tried, while for $\tau = -4$, the condition numbers of the scaling transformations are additionally supervised, and the scaling factors are set to 1 if the "optimal" scaling has a condition number larger than T. This tends to reduce the 1-norms of both matrices. Finally, if $\tau = -10^k$, the condition numbers of the acceptable scaling matrices are bounded by 10^k.

4 NUMERICAL RESULTS

An extensive testing has been performed to assess the performance of the new balancing solver for skew-Hamiltonian/Hamiltonian matrix pencils. Computation of eigenvalues, as well as solution of AREs, have been considered as main applications. This section summarizes part of the results.

The COMPl$_e$ib collection (Leibfritz and Lipinski, 2003) is taken here as a benchmark for eigenvalue computations. All 151 problems with $n < 2000$ have been tried. For testing purposes, the balancing algorithms have been used in combination with the MATLAB eigensolvers `eig`($\mathcal{H}$), when $\mathcal{S} = I_{2n}$, and `eig`($\mathcal{H},\mathcal{S}$), when $\mathcal{S}$ is general, or, for small-size problems, with the symbolic solvers `eig(vpa(`$\mathcal{H}$`))` (with default number of digits, i.e., 32), or `eig(sym(`$\mathcal{H}$`))`. Usually, `eig(vpa(`$\mathcal{H}$`))` and `eig(sym(`$\mathcal{H}$`))` have been used for examples with $n \leq 200$ and $n \leq 30$, respectively. Examples with larger orders need significant CPU times for symbolic solvers, e.g., over one hour for CM4 or CM4_IS ($n = 240$) using `vpa`, or much longer for `sym`. But a comparison between the eigenvalues computed by `eig(vpa(`$\mathcal{H}$`))` and `eig(sym(`$\mathcal{H}$`))` has shown a very good agreement between their results. The maximum absolute difference between the relative errors of `eig`($\mathcal{H}$) in comparison with `eig(vpa(`$\mathcal{H}$`))` and `eig(sym(`$\mathcal{H}$`))` for 100 COMPl$_e$ib examples with $n \leq 30$ is $3.0292 \cdot 10^{-28}$. (Examples JE2 and JE3, with $n = 21$ and $n = 24$, respectively, have been excluded, since `eig(sym(`$\mathcal{H}$`))` needed unreasonably large CPU time.) However, the relative errors of `eig`($\mathcal{H},\mathcal{S}$), when $\mathcal{S} = I_{2n}$, compared to `eig(vpa(`$\mathcal{H}$`))` and `eig(sym(`$\mathcal{H}$`))`, for example PAS have been $1.2798 \cdot 10^{-6}$, and $1.4432 \cdot 10^{-6}$, respectively. Omitting PAS, the norm of the differences between relative errors of `eig`($\mathcal{H},\mathcal{S}$) compared to `eig(vpa(`$\mathcal{H}$`))` and `eig(sym(`$\mathcal{H}$`))` was $5.8491 \cdot 10^{-25}$.

There are 18 examples with ratios between the magnitude of the largest and smallest moduli of their elements larger than 10^{16}. Besides examples CM2* – CM6*, this set also includes AC10, BDT2, PAS, TL, CDP, NN16, ISS1, and ISS2 examples. Most of these are difficult examples for any balancing algorithm, and for generalized eigensolvers using standard balancing. For instance, for CDP example, even the structure-exploiting skew-Hamiltonian/Hamiltonian solver with standard balancing, for $\mathcal{S} = I_{2n}$, returns eigenvalues with a relative error of 0.19172 (when compared to `eig`($\mathcal{H}$)), and of 0.24788 (when compared to `eig(vpa(`$\mathcal{H}$`))`), while the same solver without scaling delivers relative errors $1.0476 \cdot 10^{-14}$ and $1.3064 \cdot 10^{-14}$, respectively. These values are close to the error obtained using `eig`($\mathcal{H}$), $4.9248 \cdot 10^{-15}$, compared to `eig(vpa(`$\mathcal{H}$`))`. The structure-exploiting solver with balancing option $\tau < 0$ returns a relative error of $5.4838 \cdot 10^{-15}$ (when compared to `eig`($\mathcal{H}$)), and of $3.9091 \cdot 10^{-15}$ (when compared to `eig(vpa(`$\mathcal{H}$`))`), even more accurate than `eig`($\mathcal{H}$).

Figure 1 shows the relative errors of eigenvalues computed using `eig`(Hl,Sl), where Hl := $\widetilde{\mathcal{H}}$, Sl := $\widetilde{\mathcal{S}}$, with LAPACK balancing (using DGGBAL) and `eig`($\mathcal{H},\mathcal{S}$), in comparison with `eig`($\mathcal{H}$) for 151 COMPl$_e$ib examples. The error is infinite for CDP, CM3, CM4, CM3_IS, and CM4_IS examples. There are also several large errors using LAPACK balancing. However, for many examples, balancing improves the accuracy of the computed eigenvalues. Similar results have been obtained using the structured balancing solver with option $\tau = 0$. Specifically, except for five examples, the differences between the relative errors of eigenvalues computed using LAPACK balancing and the new balancing solver with option $\tau = 0$ have been less than about $3.64 \cdot 10^{-3}$.

Figure 2 shows the relative errors of eigenvalues computed using `eig`(Hl,Sl) with the new balancing solver with option $\tau = -1$ and `eig`($\mathcal{H},\mathcal{S}$), in comparison with `eig`($\mathcal{H}$) for 151 COMPl$_e$ib examples. There are no infinite errors. The errors are smaller for many examples than when using LAPACK balancing. Note that the ordinate axes have different scales. Note also that `eig`($\mathcal{H},\mathcal{S}$) has often larger errors than `eig`(Hl,Sl) with the new solver. The behavior for $\tau = -3$ has been

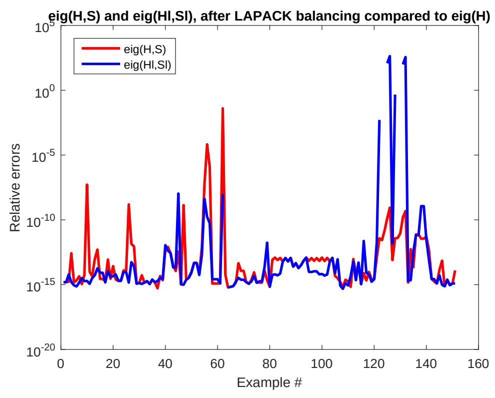

Figure 1: Relative errors of eigenvalues computed using eig(Hl,Sl) with LAPACK balancing and eig(H,S), compared to eig(H), for COMPl$_e$ib examples.

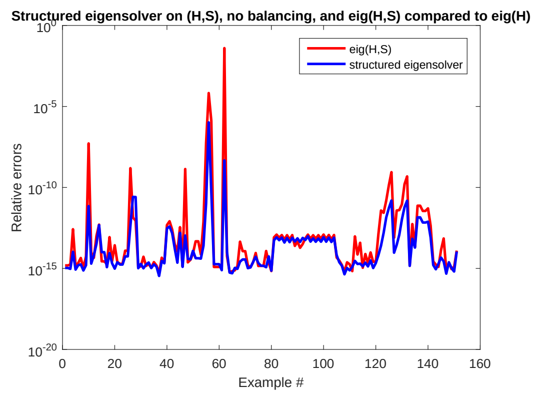

Figure 3: Relative errors of eigenvalues computed by SLICOT structured eigensolver without balancing and eig(H,S), compared to eig(H), for COMPl$_e$ib examples.

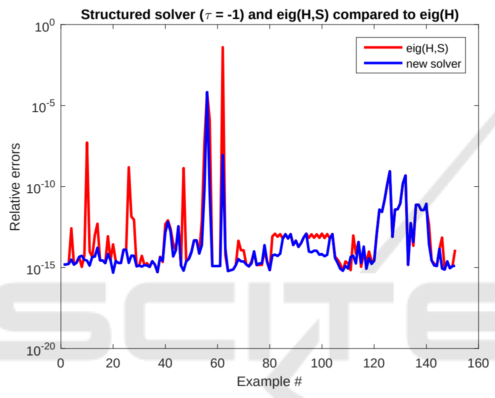

Figure 2: Relative errors of eigenvalues computed using eig(Hl,Sl) with new balancing solver, $\tau = -1$, and eig(H,S), compared to eig(H), for COMPl$_e$ib examples.

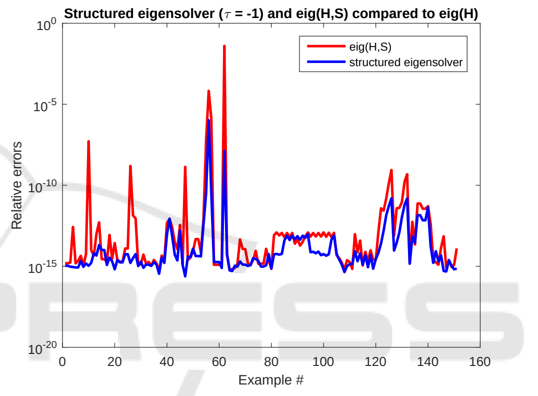

Figure 4: Relative errors of eigenvalues computed by SLICOT structured eigensolver with balancing, $\tau = -1$, and eig(H,S), compared to eig(H), for COMPl$_e$ib examples.

almost the same (the maximum difference between errors for the two options has been of order 10^{-9}).

Figure 3 illustrates the relative errors of eigenvalues of $\alpha\mathcal{S} + \beta\mathcal{H}$, computed using eig and the SLICOT structured eigensolver MB04BD (see www.slicot.org), without balancing, in comparison with eig($\mathcal{H}$). The structured eigensolver is more accurate than eig for almost all examples and comparable with eig for the remaining examples. Figure 4 uses similarly the structured eigensolver applied to $\widetilde{\mathcal{S}}$ and $\widetilde{\mathcal{H}}$, obtained with balancing option $\tau = -1$. It can provide even more accurate results than eig($\mathcal{H},\mathcal{S}$), see Fig. 2. For comparison, Fig. 5 shows results for option $\tau = 0$. Clearly, standard balancing does not provide good results in all cases, even using a structured solver.

The remaining of this section considers the performance of the new balancing solver, in combination with the structured SLICOT routine MB03LD, to compute the deflating subspaces of skew-Hamiltonian/Hamiltonian pencils, for solving AREs

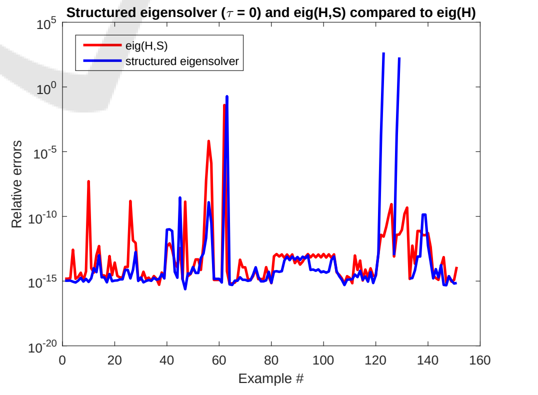

Figure 5: Relative errors of eigenvalues computed by SLICOT structured eigensolver with balancing, $\tau = 0$, and eig(H,S), compared to eig(H), for COMPl$_e$ib examples.

from the SLICOT CAREX collection (Abels and Benner, 1999). This combination is referred to as skHH in the legends of the following figures. Both pencils of order $2n$ (with $\mathcal{S} = I_{2n}$) and extended pencils of order $2(n+p)$ have been tried. The com-

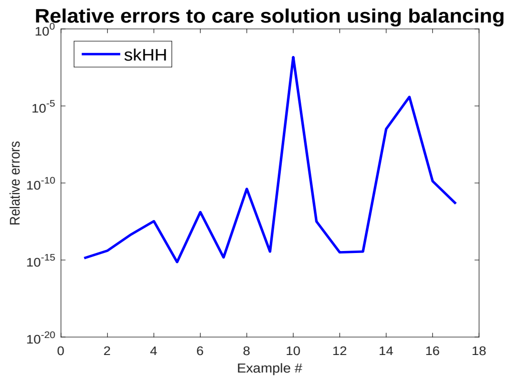

Figure 6: Relative errors of the stabilizing ARE solutions computed by SLICOT structured eigensolver (for extended pencil), with balancing option $\tau = 0$, compared to `care`, for examples from the SLICOT CAREX collection.

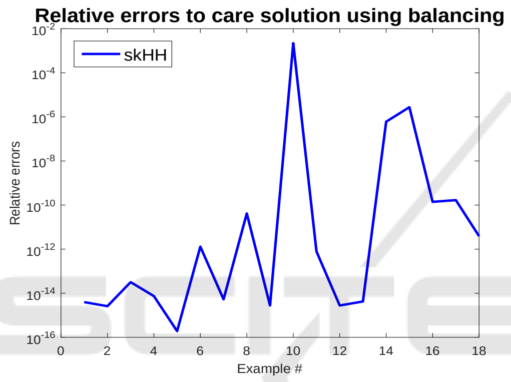

Figure 7: Relative errors of the stabilizing ARE solutions computed by SLICOT structured eigensolver (for $2n$-order pencil), with option $\tau = -1$, compared to `care`, for examples from the SLICOT CAREX collection.

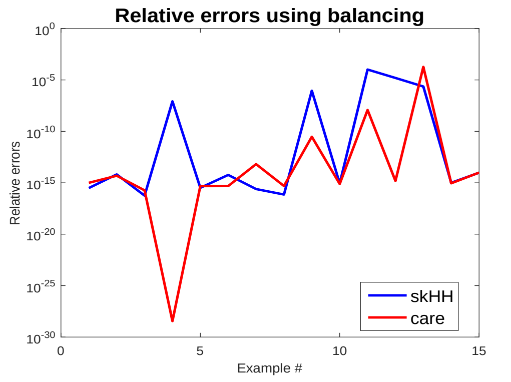

Figure 8: Relative errors of the stabilizing ARE solutions computed by `care` and SLICOT structured eigensolver (for extended pencil), with option $\tau = 0$, compared to exact solution, for examples from the SLICOT CAREX collection.

puted solutions are compared with the exact ones, when known, or with the solutions returned by the MATLAB function `care`. Note that `care` uses a special balancing procedure, and Hamiltonian matrices, instead of pencils, for all examples, except for CAREX example 2.2 with default parameter, for which the control weighting matrix tends to be singular. This creates an advantage for `care` over the new solver. However, both solvers produce comparable results, when the balancing option $\tau = -1$ is used. See (Sima and Benner, 2015b) for a comparison between a Hamiltonian-based solver and `care` for CAREX examples.

Figure 6 shows the relative errors of the stabilizing solution of the AREs computed using SLICOT structured eigensolver (for extended pencil), with balancing option $\tau = 0$, in comparison with `care`, while Fig. 7 plots similarly the relative errors when option $\tau = -1$ is used. Moreover, Fig. 8 and Fig. 9 present in the same manner the relative errors compared to the known, exact solutions. The balancing option $\tau = -1$ ensures better results than the standard balancing (for $\tau = 0$). Finally, Fig. 10 plots the relative residuals for `care` and the structured solver.

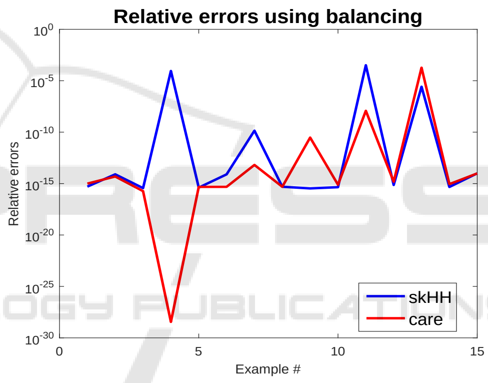

Figure 9: Relative errors of the stabilizing ARE solutions computed by `care` and SLICOT structured eigensolver (for $2n$-order pencil), with option $\tau = -1$, compared to exact solution, for examples from the SLICOT CAREX collection.

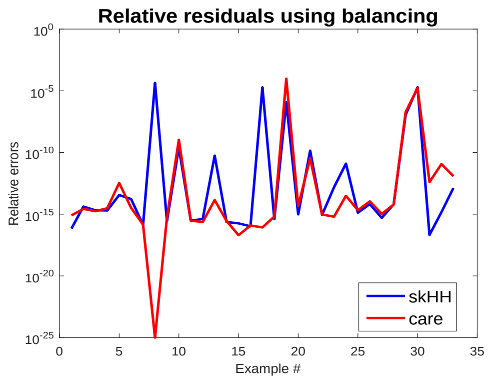

Figure 10: Relative residuals of the stabilizing ARE solutions computed by `care` and SLICOT structured eigensolver (for $2n$-order pencil), with option $\tau = -1$, for examples from the SLICOT CAREX collection.

5 CONCLUSIONS

A new structure-preserving balancing technique for skew-Hamiltonian/Hamiltonian matrix pencils is presented. Symplectic ($\mathcal{J}$-)permutations and equivalence scaling transformations are used. Several enhancements are described, which avoid a large increase of the norms of the pencil matrices, and/or of the condition numbers of the scaling transformations, which can appear when using the standard balancing procedure. The numerical results show a good performance of the new technique in comparison with state-of-the-art solvers. Tens of examples from well-known benchmark collections have been investigated.

ACKNOWLEDGEMENTS

The NICONET support is highly acknowledged.

REFERENCES

Abels, J. and Benner, P. (1999). CAREX—A collection of benchmark examples for continuous-time algebraic Riccati equations (Version 2.0). SLICOT Working Note 1999-14.

Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., and Sorensen, D. (1999). *LAPACK Users' Guide: Third Edition*. SIAM, Philadelphia.

Benner, P. (2001). Symplectic balancing of Hamiltonian matrices. *SIAM J. Sci. Comput.*, 22(5):1885–1904.

Benner, P., Byers, R., Losse, P., Mehrmann, V., and Xu, H. (2007). Numerical solution of real skew-Hamiltonian/Hamiltonian eigenproblems. Technical report, Technische Universität Chemnitz, Chemnitz.

Benner, P., Byers, R., Mehrmann, V., and Xu, H. (2002). Numerical computation of deflating subspaces of skew Hamiltonian/Hamiltonian pencils. *SIAM J. Matrix Anal. Appl.*, 24(1):165–190.

Benner, P., Kressner, D., and Mehrmann, V. (2005). Skew-Hamiltonian and Hamiltonian eigenvalue problems: Theory, algorithms and applications. In *Proceedings of the Conference on Applied Mathematics and Scientific Computing*, 3–39. Springer-Verlag, Dordrecht.

Benner, P., Sima, V., and Voigt, M. (2012a). $\mathcal{L}_\infty$-norm computation for continuous-time descriptor systems using structured matrix pencils. *IEEE Trans. Automat. Contr.*, AC-57(1):233–238.

Benner, P., Sima, V., and Voigt, M. (2012b). Robust and efficient algorithms for $\mathcal{L}_\infty$-norm computations for descriptor systems. In *7th IFAC Symposium on Robust Control Design (ROCOND'12)*, 189–194. IFAC.

Benner, P., Sima, V., and Voigt, M. (2013a). FORTRAN 77 subroutines for the solution of skew-Hamiltonian/Hamiltonian eigenproblems. Part I: Algorithms and applications. SLICOT Working Note 2013-1.

Benner, P., Sima, V., and Voigt, M. (2013b). FORTRAN 77 subroutines for the solution of skew-Hamiltonian/Hamiltonian eigenproblems. Part II: Implementation and numerical results. SLICOT Working Note 2013-2.

Bruinsma, N. A. and Steinbuch, M. (1990). A fast algorithm to compute the H_∞-norm of a transfer function. *Systems Control Lett.*, 14(4):287–293.

Jiang, P. and Voigt, M. (2013). MB04BV — A FORTRAN 77 subroutine to compute the eigenvectors associated to the purely imaginary eigenvalues of skew-Hamiltonian/Hamiltonian matrix pencils. SLICOT Working Note 2013-3.

Kressner, D. (2005). *Numerical Methods for General and Structured Eigenvalue Problems*. Springer-Verlag, Berlin.

Laub, A. J. (1979). A Schur method for solving algebraic Riccati equations. *IEEE Trans. Automat. Contr.*, AC–24(6):913–921.

Leibfritz, F. and Lipinski, W. (2003). Description of the benchmark examples in *COMPl$_e$ib*. Technical report, Department of Mathematics, University of Trier, D–54286 Trier, Germany.

MATLAB (2016). *MATLAB® Primer. R2016a*. The MathWorks, Inc., 3 Apple Hill Drive, Natick, MA.

Mehrmann, V. (1991). *The Autonomous Linear Quadratic Control Problem. Theory and Numerical Solution*. Springer-Verlag, Berlin.

Paige, C. and Van Loan, C. F. (1981). A Schur decomposition for Hamiltonian matrices. *Lin. Alg. Appl.*, 41:11–32.

Sima, V. (1996). *Algorithms for Linear-Quadratic Optimization*. Marcel Dekker, Inc., New York.

Sima, V. and Benner, P. (2015a). Pitfalls when solving eigenproblems with applications in control engineering. In *Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics* (ICINCO-2015), 21-23 July, 2015, Colmar, France, volume 1, 171–178. SciTePress.

Sima, V. and Benner, P. (2015b). Solving SLICOT benchmarks for continuous-time algebraic Riccati equations by Hamiltonian solvers. In *Proceedings of the 2015 19th International Conference on System Theory, Control and Computing* (ICSTCC 2015), October 14-16, 2015, Cheile Gradistei, Romania, 1–6. IEEE.

Van Loan, C. F. (1984). A symplectic method for approximating all the eigenvalues of a Hamiltonian matrix. *Lin. Alg. Appl.*, 61:233–251.

Ward, R. C. (1981). Balancing the generalized eigenvalue problem. *SIAM J. Sci. Stat. Comput.*, 2:141–152.