Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

New Hamiltonian Eigensolvers with Applications in Control

Peter Benner and Daniel Kressner

***Abstract*— A new MATLAB toolbox for computing eigenvalues and invariant subspaces of Hamiltonian and skew-Hamiltonian matrices is described. Based on orthogonal symplectic decompositions, the implemented algorithms are both numerically backward stable and structure-preserving. It will be demonstrated how this toolbox can be used to address a number of tasks in systems and control theory, including some model reduction methods and the computation of the H_∞ norm.**

I. INTRODUCTION

The need for solving Hamiltonian eigenvalue problems arises from a variety of applications in systems and control, including linear-quadratic optimal control, stability radius and H_∞-norm computations, robust control design, as well as model reduction, see, e.g., [1] and the references therein. A real $2n \times 2n$ matrix H is called *Hamiltonian* if it takes the form

$$H = \begin{bmatrix} A & G \\ Q & -A^T \end{bmatrix}, \quad G = G^T, \quad Q = Q^T, \qquad (1)$$

where A, G and Q are $n \times n$ matrices. Hamiltonian matrices have several important properties that distinguish them from general, unstructured matrices. For example, if λ is an eigenvalue of H then also $-\lambda, \bar{\lambda}, -\bar{\lambda}$ are eigenvalues of H. In other words, the spectrum of H is symmetric with respect to the real and imaginary axes.

If one attempts to compute the eigenvalues of a Hamiltonian matrix H by a standard method, such as the QR or Arnoldi algorithm [2] (which correspond to the MATLAB commands `eig` and `eigs`), then the computed eigenvalues will lose this symmetry property due to the influence of roundoff errors. This also makes it difficult to identify eigenvalues of H that have zero or negative real part. However, a proper identification of those eigenvalues is often crucial in applications. For example, deciding whether a certain Hamiltonian matrix has purely imaginary eigenvalues is the most critical step in many algorithms for computing the H_∞-norm and in H_∞ design [3], [4]. Also, checking the stability of a gyroscopic system depends on such a decision [5].

These observations have been the major source of motivation for developing Hamiltonian eigenvalue solvers that are both numerically sound (i.e., backward stable) *and* capable to preserve the eigenvalue symmetries in finite-precision arithmetic. An algorithm that satisfies these criteria was presented in [6], [7]. This algorithm has been the basis for HAPACK [8], a comprehensive Fortran 77 library for solving Hamiltonian and skew-Hamiltonian eigenvalue problems. However, HAPACK contains further algorithmic improvements, including symplectic balancing [9] and block algorithms [10], making it competitive to the general-purpose eigenvalue solvers implemented in state-of-the-art software libraries, such as LAPACK [11].

This work was partially supported by the DFG Research Center "Mathematics for Key technolgies" (FZT86) in Berlin.

P. Benner is with the Fakultät für Mathematik, TU Chemnitz, 09107 Chemnitz, Germany. benner@mathematik.tu-chemnitz.de

D. Kressner is with the Department of Mathematics, University of Zagreb, Bijenička 30, 10000 Zagreb, Croatia. kressner@math.hr

The software toolbox presented in this paper is mainly built on MEX interfaces, which provide the complete functionality of HAPACK in a convenient way to MATLAB users. Section II summarizes the contents of this toolbox as well as its usage for solving Hamiltonian eigenvalue problems and related tasks. In Section III, the following applications will be addressed: solution of linear-quadratic optimal control problems, stability radius and H_∞-norm computations, as well as the solution of gyroscopic eigenvalue problems.

II. THE HAPACK MATLAB TOOLBOX

Its MATLAB functions for solving Hamiltonian eigenvalue problems constitute the core of the presented toolbox. They are built on the MEX gateway `hapack_haeig.f`, which bundles all the relevant functionality of HAPACK, see also Fig. 1. Among these functions, `haeig` is used for computing the eigenvalues and `hastab` for computing the stable invariant subspace of a Hamiltonian matrix.

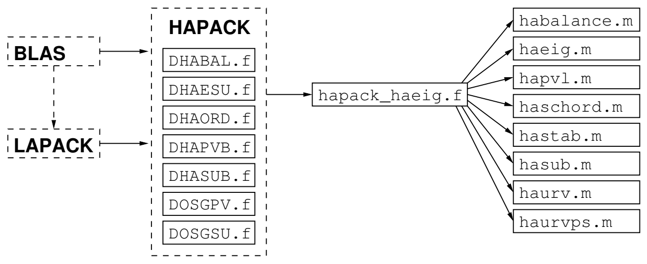

Fig. 1. Dependencies between HAPACK Fortran routines, MATLAB functions and the MEX gateway `hapack_haeig.f`.

A. `haeig`

The MATLAB command `e = haeig(H)` returns the eigenvalues of a $2n \times 2n$ Hamiltonian matrix $H = \begin{bmatrix} A & G \\ Q & -A^T \end{bmatrix}$. The eigenvalues are contained in a vector e of length $2n$, ordered such that $e_i = -e_{n+i}$ and the leading n elements of e have nonpositive real part.

The numerical method behind `haeig` is based on transformations with matrices that are both orthogonal ($Q^TQ = I_{2n}$ with the $2n \times 2n$ identity matrix I_{2n}) and symplectic ($Q^TJQ = J$ with $J = \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix}$). The *symplectic URV* *decomposition* introduced in [7] states that there exist two such orthogonal symplectic matrices U and V so that

$$U^T H V = \left[\begin{array}{cc} R_{11} & R_{12} \\ R_{21} & R_{22} \end{array}\right] = \left[\begin{array}{cc} \text{◹} & \square \\ & \text{⧅} \end{array}\right], \qquad (2)$$

i.e., the matrix $R_{21} \in \mathbb{R}^{n\times n}$ is zero, $R_{11} \in \mathbb{R}^{n\times n}$ is upper triangular and $R_{22} \in \mathbb{R}^{n\times n}$ is lower Hessenberg. A simple calculation reveals

$$U^T H^2 U = \left[\begin{array}{cc} -R_{11}R_{22}^T & R_{11}R_{12}^T - R_{12}R_{11}^T \\ 0 & -R_{22}R_{11}^T \end{array}\right],$$

showing that the eigenvalues of H are the square roots of the eigenvalues of the upper Hessenberg matrix $-R_{11}R_{22}^T$. In a second step, the periodic QR algorithm [12], [13], [14] is applied to compute the eigenvalues of this matrix product in a numerically backward stable manner. Both steps can also be called individually via the MATLAB functions `haurv` and `haurvps`.

Symplectic balancing [9] often positively influences the accuracy of the computed eigenvalues and is consequently used by default in `haeig`. In exceptional cases, however, it may have a negative influence on the accuracy, in which case it can be beneficial to turn balancing off: `e = haeig(H,'nobalance')`.

A $2n \times 2n$ Hamiltonian matrix of the form (1) can be completely represented by the $n \times n$ matrix A and the lower/upper triangular parts of the symmetric matrices Q and G. To account for this fact, we offer users the option to employ the packed storage layout proposed in [15]. While A is stored in a separate variable, the symmetric matrices G and Q are stored in an $n \times (n+1)$ array QG as follows:

$$\mathsf{QG} = \left[\begin{array}{ccccc} q_{11} & g_{11} & g_{12} & g_{13} & \cdots \\ q_{21} & q_{22} & g_{22} & g_{23} & \cdots \\ q_{31} & q_{32} & q_{33} & g_{33} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array}\right].$$

The function `haconv` can be used to convert between different storage layouts. To employ the packed storage layout in eigenvalue computations, one simply calls `e = haeig(A,QG)` or `e = haeig(A,QG,'nobalance')`.

B. `hastab`

Assuming that the $2n \times 2n$ Hamiltonian matrix H has no eigenvalues on the imaginary axis, there are precisely n eigenvalues with negative real part. The invariant subspace $\mathcal{X}$ belonging to these eigenvalues is called *stable*. An important property of this subspace is its isotropy, i.e., $x^T J y = 0$ for all $x, y \in \mathcal{X}$ with, as above, $J = \left[\begin{smallmatrix} 0 & I_n \\ -I_n & 0 \end{smallmatrix}\right]$.

The function `X = hastab(H)` returns an orthonormal basis $X \in \mathbb{R}^{2n\times n}$ for $\mathcal{X}$. It is based on the method described in [6], which extracts X from a symplectic URV decomposition (2). Occasionally, the results returned by `hastab` can be unsatisfactorily inaccurate, in the sense that one of the relations $X^T J X = 0$ (corresponding to the isoptropy of $\mathcal{X}$) or $X^T J H X = 0$ (corresponding to the fact that $\mathcal{X}$ is an invariant subspace) is severely violated. In such cases, it can be beneficial to call `X = hastab(H,3)`, which uses a slightly improved but more expensive method also described in [6]. Using this option, `hastab` is capable to compute X in a satisfactory and numerically backward stable manner for all Hamiltonian matrices from the benchmark collection [16], see also the numerical results in [8].

`[X,Y,e] = hastab(H)` also computes an orthonormal basis Y for the unstable invariant subspace, i.e., the invariant subspace belonging to the eigenvalues with positive real part. Moreover, the eigenvalues of H are contained in a vector e, in the same order as returned by `haeig`. Invariant subspaces belonging to other eigenvalues can be computed with the MATLAB function `hasub`.

Again, the packed storage layout can be employed for invariant subspace computations by using two input arguments `A,QG` instead of `H`.

C. `sheig`

Skew-Hamiltonian matrices are closely related to Hamiltonian matrices and take the form

$$W = \left[\begin{array}{cc} A & G \\ Q & A^T \end{array}\right], \quad G = -G^T, \quad Q = -Q^T, \qquad (3)$$

where A, G and Q are $n \times n$ matrices. The eigenvalues of W have even algebraic multiplicity; an algorithm that preserves this property has been developed by Van Loan [17] and is implemented in the MATLAB function `sheig` built on the MEX gateway `hapack_sheig.f` to HAPACK. The calling syntax of `sheig` is identical to the syntax of `haeig`.

We now briefly explain how `sheig` can be used to address complex Hamiltonian eigenvalue problems. A *complex Hamiltonian matrix* $H \in \mathbb{C}^{2n\times 2n}$ is defined by the property $(JH)^\star = JH$, with the matrix J as above. Similar to the real case, the eigenvalues of H come in pairs $\{\lambda, -\bar{\lambda}\}$; `sheig` enables us to preserve these eigenvalue pairings in finite-precision arithmetic.

To see this, let us decompose $H = H_R + \imath H_I$ such that $H_R, H_I \in \mathbb{R}^{2n\times 2n}$. Then the relationship $(JH)^\star = JH$ implies that the two matrices H_R and H_I have the following structure:

$$H_R = \left[\begin{array}{cc} A_R & G_R \\ Q_R & -A_R^T \end{array}\right], \quad H_I = \left[\begin{array}{cc} A_I & G_I \\ Q_I & A_I^T \end{array}\right],$$

where $G_R = G_R^T, Q_R = Q_R^T, G_I = -G_I^T, Q_I = -Q_I^T$, i.e., the matrix H_R is Hamiltonian and the matrix H_I is skew-Hamiltonian. The $4n \times 4n$ skew-Hamiltonian matrix

$$W = \left[\begin{array}{cc|cc} A_I & A_R & G_I & G_R \\ -A_R & A_I & -G_R & G_I \\ \hline Q_I & Q_R & A_I^T & -A_R^T \\ -Q_R & Q_I & A_R^T & A_I^T \end{array}\right] \qquad (4)$$

is permutationally similar to the matrix $\left[\begin{smallmatrix} H_I & H_R \\ -H_R & H_I \end{smallmatrix}\right]$, which implies that λ is an eigenvalue of H if and only if $\{\imath\lambda, -\imath\bar{\lambda}\}$ is an eigenvalue pair of W. Since `sheig` preserves the even eigenvalue multiplicities of W, it is sufficient to consider the

$2n$ (generically different) eigenvalues of W. These eigenvalues come in pairs $\{\mu, \bar{\mu}\}$, each of which corresponds to an eigenvalue pair $\{\imath\mu, \imath\bar{\mu}\}$ of H.

An embedding of the form (4) can also be used to obtain invariant subspaces of complex Hamiltonian matrices, see [18].

D. `sympqr`

A decomposition closely related to Hamiltonian eigenvalue problems is the *symplectic QR decomposition*, defined as follows. Let $X \in \mathbb{R}^{2n\times k}$ with $n \geq k$, then there exists an orthogonal symplectic matrix Q so that $X = QR$ and

$$R = \left[\begin{array}{c} R_{11} \\ R_{21} \end{array}\right], \; R_{11} = \left[\begin{array}{c} \diagdown\!\!| \\ 0 \end{array}\right], \; R_{21} = \left[\begin{array}{c} {}^{0}\diagdown\!\!|\,{}_{0} \\ 0 \end{array}\right], \qquad (5)$$

i.e., the matrix $R_{11} \in \mathbb{R}^{m\times n}$ is upper triangular and $R_{21} \in \mathbb{R}^{m\times n}$ is strictly upper triangular [19]. The MATLAB function `sympqr`, which computes such a decomposition, can be called via `[Q,R] = sympqr(X)` or `[Q,R] = sympqr(X,0)`. The latter call results in an "economy size" decomposition, i.e., only columns $1, \ldots, k$ and $n+1, \ldots, n+k$ of Q are computed.

The symplectic QR decomposition can be used for the computation of nearby isotropic subspaces [1] or for the symplectic integration of Hamiltonian systems [20].

III. Applications and numerical examples

In the following, we consider a *linear continuous-time system* with constant coefficients in state-space form,

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \quad x(0) = x_0, && (6) \\ y(t) &= Cx(t) + Du(t), && (7) \end{aligned}$$

where $x(t) \in \mathbb{R}^n$ is the vector of *states*, $u(t) \in \mathbb{R}^m$ the vector of *inputs* (or *controls*) and $y(t) \in \mathbb{R}^r$ the vector of *outputs* at time $t \in [0, \infty)$. The system is described by the *state matrix* $A \in \mathbb{R}^{n\times n}$, the *input (control) matrix* $B \in \mathbb{R}^{n\times m}$, the *output matrix* $C \in \mathbb{R}^{r\times n}$, and the *feedthrough matrix* $D \in \mathbb{R}^{r\times m}$. To simplify the presentation we restrict ourselves in Sections A and C to $D = 0$; all results can be extended to the case $D \neq 0$ without any greater difficulties.

A. Linear-quadratic optimal control problems

First, we consider the linear-quadratic L_2 optimal control problem (LQR) for a linear continuous-time system, which is to minimize an energy functional

$$\mathcal{S}(x, u) = \int_{t_0}^{\infty} \left(x(t)^T Q x(t) + u(t)^T R u(t)\right) \, \mathrm{d}t \qquad (8)$$

subject to the control equation (6), where $Q \in \mathbb{R}^{n\times n}$ is symmetric positive semi-definite matrix, $R \in \mathbb{R}^{m\times m}$ is a symmetric positive definite matrix; and it is required that the solution associated with the optimal control is asymptotically stable.

The Hamiltonian matrix

$$H = \left[\begin{array}{cc} A & -BR^{-1}B^T \\ -Q & -A^T \end{array}\right],$$

is closely related to (8). If the columns of $X = \left[\begin{smallmatrix} X_1 \\ X_2 \end{smallmatrix}\right]$ with $X_1, X_2 \in \mathbb{R}^{n\times n}$ span the stable invariant subspace of X, then, under some mild assumptions, $F = -X_2X_1^{-1}$ is the stabilizing solution of the algebraic Riccati equation

$$0 = -Q + A^T F + FA + FBR^{-1}B^T F,$$

see, e.g., [21], [22], [23]. In turn, the linear state feedback

$$u(t) = B^T F u(t)$$

solves (8). The following piece of MATLAB code computes the matrix F based on `hastab`:

```
RR = chol(R); B = B / RR;
H = [ A  -B*B'; -Q -A'];
X = hastab(H);
F = X(n+1:2*n,:) / X(1:n,:);
```

For numerical examples and a comparison to other methods for computing F, we refer to [8].

B. Stability Radius Computation

A linear system of the form (6) is called (asymptotically) *stable* if all eigenvalues $\lambda(A)$ of the state matrix A are in $\mathbb{C}^-$, the open left half complex plane. It is often important to know how near the system is to an unstable one, i.e., what is the smallest perturbation $E \in \mathbb{C}^{n\times n}$ so that $\lambda(A+E) \not\subset \mathbb{C}^-$. This corresponds to the computation of the *stability radius* of A, which is defined as

$$\gamma(A) := \min\{\|E\|_2 \, : \, \lambda(A+E) \cap \imath\mathbb{R} \neq \emptyset\}.$$

A bisection method for measuring $\gamma(A)$ can be based on the following observation [24], [25]: if $\alpha \geq 0$, then the Hamiltonian matrix

$$H(\alpha) = \left[\begin{array}{cc} A & -\alpha I_n \\ \alpha I_n & -A^T \end{array}\right]$$

has an eigenvalue on the imaginary axis if and only if $\alpha \geq \gamma(A)$. This suggests the following simple algorithm. Start with a lower bound $\beta \geq 0$ and an upper bound $\delta > \gamma(A)$ (an easy-to-compute upper bound is $\|A + A^T\|_F/2$ [26]). Then in each step, set $\alpha := (\beta + \delta)/2$ and compute $\lambda(H(\alpha))$. If there is an eigenvalue on the imaginary axis, choose $\delta = \alpha$, otherwise, set $\beta = \alpha$. A MATLAB implementation of this method based on the function `haeig` could be as follows:

```
nA = norm(A+A','fro') / 2;
beta  = 0;
delta = nA;
e = ones(n,1);
QG = full(spdiags([e -e], 0:1, n, n+1));
while (delta-beta) > 100*eps*nA,
   e = haeig(A,(beta + delta)/2 * QG);
   if isempty( find(real(e)==0) ),
      beta  = (beta + delta)/2;
   else
      delta = (beta + delta)/2;
   end
end
```

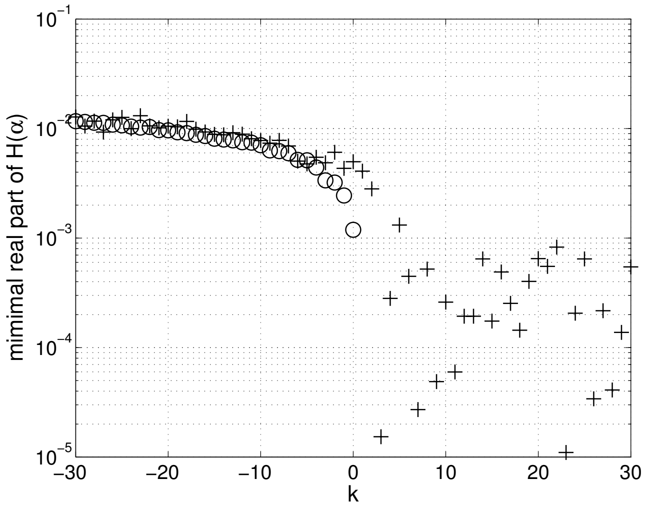

Fig. 2. Minimum absolute values for the real parts of the eigenvalues of $H(\alpha)$ computed by `eig` ('+') and `haeig` ('o') with the matrix A as in (9) and $\alpha = 1.005^k \times \gamma(A)$, $k \in [-30, 30]$. For $k > 0$, `haeig` computes exact zeroes which are not shown in the figure.

The correct decision whether $H(\alpha)$ has eigenvalues on the imaginary axis is crucial for the success of the bisection method. Byers [24] showed that if the eigenvalues of $H(\alpha)$ are computed by a strongly backward stable method, then the computed $\gamma(A)$ will be within a distance of $\mathcal{O}(\mathbf{u}) \times \|A\|_2$ to the exact stability radius, where $\mathbf{u}$ is the machine precision.

As an example, let us consider a variation of Demmel's matrix [27]:

$$A = Q^T \begin{bmatrix} \lambda & -10 & -10^2 & -10^3 & -10^4 \\ 0 & \lambda & -10 & -10^2 & -10^3 \\ 0 & 0 & \lambda & -10 & -10^2 \\ 0 & 0 & 0 & \lambda & -10 \\ 0 & 0 & 0 & 0 & \lambda \end{bmatrix} Q, \quad (9)$$

where $\lambda = -0.05$ and Q is a randomly generated orthogonal matrix. The stability radius of this matrix is much smaller than the minimal distance of the eigenvalues to the imaginary axis, $\gamma(A) \approx 3.6 \times 10^{-11}$. Figure 2 shows that `haeig` correctly identifies the purely imaginary eigenvalues of $H(\alpha)$ for $\alpha > \gamma(A)$. On the other hand, the eigenvalues computed by MATLAB's `eig` all have non-negligible real parts no matter whether $\alpha > \gamma(A)$ or $\alpha \leq \gamma(A)$. Hence, for this case, `haeig` leads to much more reliable decisions within the bisection method, which in turn improves the reliability of the method itself.

C. H_∞ Norm Computation

A problem very similar to the stability radius calculation is the computation of the H_∞ norm of a stable system. Consider the transfer function $\mathcal{G}(s)$ of a stable system of the form (6)–(7),

$$\mathcal{G}(s) = C(sI - A)^{-1}B + D,$$

then

$$\|\mathcal{G}\|_{H_\infty} = \operatorname{esssup}\{\|\mathcal{G}(\imath\omega)\|_2 \ : \ \omega \in \mathbb{R}\}.$$

is the H_∞ norm of $\mathcal{G}$.

Let α be a positive real number and consider the parameter-dependent Hamiltonian matrix

$$H(\alpha) = \begin{bmatrix} A & -\frac{1}{\alpha^2}BB^T \\ C^TC & -A^T \end{bmatrix},$$

The following result can be used to approximate $\|\mathcal{G}\|_{H_\infty}$, see, e.g., [23]:

$$\|\mathcal{G}\|_{H_\infty} < \alpha \quad \Leftrightarrow \quad \lambda(H(\alpha)) \cap \imath\mathbb{R} = \emptyset.$$

Using this fact, a bisection algorithm analogous to the stability radius computation can be formulated, starting with a lower bound $\beta > 0$ and an upper bound $\delta > \|\mathcal{G}\|_{H_\infty}$, see [28] for details. Again, the bisection algorithm benefits if the decisions are based on a symmetry-preserving eigenvalue solver as implemented in `haeig`.

Faster convergent versions of this algorithm, which may also involve the eigenvectors of $H(\alpha)$, can be found in [29], [30], [31].

D. Gyroscopic Eigenvalue Problems

The quadratic eigenvalue problem (QEP) is to find scalars λ and nonzero vectors x satisfying

$$(\lambda^2 M + \lambda G + K)x = 0, \quad (10)$$

where $M, G, K \in \mathbb{R}^{n \times n}$. It arises, for example, from linear systems that are governed by second order differential equations, see [5]. Gyroscopic systems yield QEPs with symmetric positive definite M, skew-symmetric G and symmetric K. In this case, the eigenvalues of (10) have the same symmetries as in the Hamiltonian eigenvalue problem, i.e., if λ is an eigenvalue then $-\lambda, \bar{\lambda}$ and $-\bar{\lambda}$ are also eigenvalues.

Assuming we have a such a gyroscopic eigenvalue problem, let R be the Cholesky factor of M, i.e., R is an upper triangular matrix and $M = R^TR$. Then the matrix

$$H = \begin{bmatrix} -\frac{1}{2}R^{-T}GR^{-1} & \frac{1}{4}(R^{-T}GR)^2 - K \\ I & -\frac{1}{2}R^{-T}GR^{-1} \end{bmatrix}$$

is Hamiltonian and has the same eigenvalues as the QEP (10). Hence, `haeig` applied to H preserves the eigenvalue pairings of (10). This is particularly important for testing the stability of the underlying gyroscopic system, which amounts to checking whether all eigenvalues of (10) are on the imaginary axis, see e.g. [5, Sec.5.3]. The corresponding MATLAB code reads as follows:

```
R = chol(M);
G = - ( R' \  G ) / R / 2;
K = ( R' \  K ) / R;
K = [ zeros(n,1), triu(G^2)-triu(K) ];
K(1:n+1:n*n) = 1;
e = haeig(G,K);
```

Let us consider the following QEP [5], which stems from the model of a shaft rotating with angular velocity Ω, containing a mass m and four springs with stiffness k_x, k_y:

$$\lambda^2 I + \lambda \begin{bmatrix} 0 & -2\Omega \\ 2\Omega & 0 \end{bmatrix} + \begin{bmatrix} \frac{k_x}{m} - \Omega^2 & 0 \\ 0 & \frac{k_y}{m} - \Omega^2 \end{bmatrix}. \quad (11)$$

For $k_x = 1, k_y = 3, m = 5, \Omega = 1/\sqrt{5} - 10^{-14}$, the underlying gyroscopic system is known to be stable implying that all eigenvalues of (11) are on the imaginary axis. The following table shows the eigenvalues of (11) computed by MATLAB's polynomial eigenvalue solver `polyeig` and by the above approach based on `haeig`.

`polyeig`	`haeig`
$-2.44 \times 10^{-17} - 1.10\imath$	$-1.10\imath$
$-2.48 \times 10^{-17} + 1.10\imath$	$+1.10\imath$
$-4.27 \times 10^{-14} - 5.47 \times 10^{-8}\imath$	$-5.47 \times 10^{-8}\imath$
$+4.26 \times 10^{-14} + 5.47 \times 10^{-8}\imath$	$+5.47 \times 10^{-8}\imath$

It can be seen that `haeig` correctly identifies all eigenvalues to be purely imaginary while the use of `polyeig` results in non-zero (although almost negligible) real parts.

It should be noted that the described approach is only feasible if M is sufficiently well-conditioned. Otherwise, it is advisable to use linearizations to structured generalized eigenvalue problems as proposed in [32], [5].

E. Model Reduction

Several balancing-related methods for model reduction of linear continuous-time systems as in (6)–(7) with $m = r$ and D nonsingular require the solution of algebraic Riccati equations [33], [34]. Thus, similar to the LQR problem, the function `hastab` can be employed here.

Model reduction methods based on balanced truncation in a first step require the computation of two Gramians of the system. These are positive (semi-)definite matrices P, Q which carry certain system information. In the classical balanced truncation method, P, Q are the controllability and observabilty Gramians of the system which can be obtained as solutions to a pair of dual Lyapunov equations. Algebraic Riccati equations show up, for instance, in *balanced stochastic truncation (BST)* and *positive real balanced truncation (PRBT)*. For BST, we have to solve the algebraic Riccati equation

$$0 = \hat{A}^T Q + Q\hat{A} + Q\hat{B}\hat{R}^{-1}\hat{B}^T Q + C^T \hat{R}^{-1} C, \qquad (12)$$

where

$$\begin{aligned} \hat{R} &= DD^T, \\ \hat{B} &= BD^T + PC^T, \\ \hat{A} &= A - \hat{B}\hat{R}^{-1}C, \end{aligned}$$

and P is the controllability Gramian of (6). One way to solve (12) in MATLAB is as follows:

```
[Q,R] = qr(D');
B = (B*D' + P*C') \ R;
C = R' \ C;
A = A - B*C;
H = [ A  -B*B'; C*C' -A'];
X = hastab(H);
Q = X(n+1:2*n,:) / X(1:n,:);
```

The dual algebraic Riccati equations arising in positive real balancing,

$$\begin{aligned} 0 &= \hat{A}P + P\hat{A}^T + PC^T R^{-1} CP + BR^{-1}B^T, \\ 0 &= \hat{A}^T Q + Q\hat{A} + QBR^{-1}B^T Q + C^T R^{-1} C, \end{aligned}$$

where

$$\begin{aligned} \hat{R} &= D + D^T, \\ \hat{A} &= A - \hat{B}\hat{R}^{-1}C, \end{aligned}$$

can be solved in an analogous way employing `hastab`. The duality of the two algebraic Riccati equations can be employed by noting that the Hamiltonian matrix H corresponding to the "Q"-equation is the transpose of that of the "P"-equation and that the unstable invariant subspace Y of H yields the stable one of H^T via $-JY$. By observing that for positive real balancing $D + D^T$ must be assumed to be positive definite, we get the following code fragment to solve the positive real algebraic Riccati equations:

```
R = chol(D+D');
B = B / R;
C = R' \ C;
A = A - B*C;
H = [ A  -B*B'; C*C' -A'];
[X,Y,e] = hastab(H);
P = -Y(1:n,:) / Y(n+1:2*n,:);
Q = -X(n+1:2*n,:) / X(1:n,:);
```

Here we employ the feature of `hastab` to also return unstable invariant subspaces.

IV. Conclusions

We have presented a MATLAB toolbox for solving Hamiltonian eigenvalue problems and demonstrated its advantages for solving eigenvalue problems arising from several applications in systems and control theory. This toolbox as well as MATLAB files for addressing the applications considered in this paper are available from `http://www.tu-chemnitz.de/mathematik/hapack/`. Future work will aim at providing similar functionality for generalized Hamiltonian eigenvalue problems based on the algorithms presented in [7], [35], which will further enhance the reliability and accuracy of HAPACK.

References

[1] P. Benner, D. Kressner, and V. Mehrmann, "Skew-Hamiltonian and Hamiltonian eigenvalue problems: Theory, algorithms and applications," in *Proceedings of the Conference on Applied Mathematics and Scientific Computing, Brijuni (Croatia), June 23-27, 2003*, Z. Drmač, M. Marušić, and Z. Tutek, Eds. Springer-Verlag, 2005.

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

[3] P. Benner, R. Byers, V. Mehrmann, and H. Xu, "Robust numerical methods for robust control," Institut für Mathematik, TU Berlin," Preprint 06-2004, 2004.

[4] N. J. Higham, M. Konstantinov, V. Mehrmann, and P. Petkov, "The sensitivity of computational control problems," *IEEE Control Syst. Mag.*, vol. 24, no. 1, pp. 28–43, 2004.

[5] F. Tisseur and K. Meerbergen, "The quadratic eigenvalue problem," *SIAM Rev.*, vol. 43, no. 2, pp. 235–286, 2001.

[6] P. Benner, V. Mehrmann, and H. Xu, "A new method for computing the stable invariant subspace of a real Hamiltonian matrix," *J. Comput. Appl. Math.*, vol. 86, pp. 17–43, 1997.

[7] ——, "A numerically stable, structure preserving method for computing the eigenvalues of real Hamiltonian or symplectic pencils," *Numerische Mathematik*, vol. 78, no. 3, pp. 329–358, 1998.

[8] P. Benner and D. Kressner, "Fortran 77 subroutines for computing the eigenvalues of Hamiltonian matrices II," *ACM Trans. Math. Software*, to appear, available from http://www.tu-chemnitz.de/mathematik/hapack/.

[9] P. Benner, "Symplectic balancing of Hamiltonian matrices," *SIAM J. Sci. Comput.*, vol. 22, no. 5, pp. 1885–1904, 2000.

[10] D. Kressner, "Block algorithms for orthogonal symplectic factorizations," *BIT*, vol. 43, no. 4, pp. 775–790, 2003.

[11] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. W. Demmel, J. J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. C. Sorensen, *LAPACK Users' Guide*, 3rd ed. Philadelphia, PA: SIAM, 1999.

[12] A. Bojanczyk, G. H. Golub, and P. Van Dooren, "The periodic Schur decomposition; algorithm and applications," in *Proc. SPIE Conference*, vol. 1770, 1992, pp. 31–42.

[13] J. J. Hench and A. J. Laub, "Numerical solution of the discrete-time periodic Riccati equation," *IEEE Trans. Automat. Control*, vol. 39, no. 6, pp. 1197–1210, 1994.

[14] C. F. Van Loan, "A general matrix eigenvalue algorithm," *SIAM J. Numer. Anal.*, vol. 12, no. 6, pp. 819–834, 1975.

[15] P. Benner, R. Byers, and E. Barth, "Algorithm 800: Fortran 77 subroutines for computing the eigenvalues of Hamiltonian matrices I: The square-reduced method," *ACM Trans. Math. Software*, vol. 26, pp. 49–77, 2000.

[16] J. Abels and P. Benner, "CAREX - a collection of benchmark examples for continuous-time algebraic Riccati equations (version 2.0)," WGS," SLICOT Working Note 1999-14, 1999, available online from http://www.win.tue.nl/niconet/.

[17] C. F. Van Loan, "A symplectic method for approximating all the eigenvalues of a Hamiltonian matrix," *Linear Algebra Appl.*, vol. 61, pp. 233–251, 1984.

[18] P. Benner, V. Mehrmann, and H. Xu, "A note on the numerical solution of complex Hamiltonian and skew-Hamiltonian eigenvalue problems," *Electron. Trans. Numer. Anal.*, vol. 8, pp. 115–126, 1999.

[19] A. Bunse-Gerstner, "Matrix factorizations for symplectic QR-like methods," *Linear Algebra Appl.*, vol. 83, pp. 49–77, 1986.

[20] B. J. Leimkuhler and E. S. Van Vleck, "Orthosymplectic integration of linear Hamiltonian systems," *Numer. Math.*, vol. 77, no. 2, pp. 269–282, 1997.

[21] B. Anderson and J. Moore, *Optimal Control – Linear Quadratic Methods*. Englewood Cliffs, NJ: Prentice-Hall, 1990.

[22] M. Green and D. J. N. Limebeer, *Linear Robust Control*. Englewood Cliffs, NJ: Prentice-Hall, 1995.

[23] K. Zhou, J. C. Doyle, and K. Glover, *Robust and Optimal Control*. Upper Saddle River, NJ: Prentice-Hall, 1996.

[24] R. Byers, "A bisection method for measuring the distance of a stable to unstable matrices," *SIAM J. Sci. Statist. Comput.*, vol. 9, pp. 875–881, 1988.

[25] D. Hinrichsen and A. J. Pritchard, "Stability radii of linear systems," *Systems Control Lett.*, vol. 7, no. 1, pp. 1–10, 1986.

[26] C. F. Van Loan, "How near is a matrix to an unstable matrix?" *Lin. Alg. and its Role in Systems Theory*, vol. 47, pp. 465–479, 1984.

[27] J. W. Demmel, "A counterexample for two conjectures about stability," *IEEE Trans. Automat. Control*, vol. 32, pp. 340–342, 1987.

[28] S. Boyd, V. Balakrishnan, and P. Kabamba, "A bisection method for computing the $\mathcal{H}_\infty$ norm of a transfer matrix and related problems," *Math. Control, Signals, Sys.*, vol. 2, pp. 207–219, 1989.

[29] S. Boyd and V. Balakrishnan, "A regularity result for the singular values of a transfer matrix and a quadratically convergent algorithm for computing its $\mathbf{L}_\infty$-norm," *Systems Control Lett.*, vol. 15, no. 1, pp. 1–7, 1990.

[30] N. Bruinsma and M. Steinbuch, "A fast algorithm to compute the H_∞-norm of a transfer function matrix," *Sys. Control Lett.*, vol. 14, no. 4, pp. 287–293, 1990.

[31] Y. Genin, P. Van Dooren, and V. Vermaut, "Convergence of the calculation of H_∞ norms and related questions," in *Proceedings of the Conference on the Mathematical Theory of Networks and Systems, MTNS '98*, A. Beghi, L. Finesso, and G. Picci, Eds., 1998, pp. 429–432.

[32] V. Mehrmann and D. S. Watkins, "Structure-preserving methods for computing eigenpairs of large sparse skew-Hamiltonian/Hamiltonian pencils," *SIAM J. Sci. Comput.*, vol. 22, no. 6, pp. 1905–1925, 2000.

[33] A. Antoulas, *Lectures on the Approximation of Large-Scale Dynamical Systems*. Philadelphia, PA: SIAM Publications, 2005.

[34] G. Obinata and B. Anderson, *Model Reduction for Control System Design*, ser. Communications and Control Engineering Series. London, UK: Springer-Verlag, 2001.

[35] P. Benner, R. Byers, V. Mehrmann, and H. Xu, "Numerical computation of deflating subspaces of skew-Hamiltonian/Hamiltonian pencils," *SIAM J. Matrix Anal. Appl.*, vol. 24, no. 1, 2002.