Computing Optimal Hankel Norm Approximations of Large-Scale Systems*

Peter Benner† Enrique S. Quintana-Ortí‡ Gregorio Quintana-Ortí§

April 14, 2004

Abstract

We discuss an efficient algorithm for optimal Hankel norm approximation of large-scale systems and an implementation which allows to reduce models of order up to $\mathcal{O}(10^4)$ using parallel computing techniques. The major computational tasks in this approach are the computation of a minimal balanced realization, involving the solution of two Lyapunov equations, and the additive decomposition of a transfer function via block diagonalization. We will illustrate that these computational tasks can all be performed using iterative schemes for the matrix sign function. Numerical experiments on a cluster of Linux PCs show the efficiency of our methods.

1 Introduction

Consider the transfer function matrix (TFM) $G(s) = C(sI - A)^{-1}B + D$, and the associated stable, but not necessarily minimal, realization of a linear time-invariant (LTI) system,

$$\begin{array}{rcll} \dot{x}(t) & = & Ax(t) + Bu(t), & t > 0, \\ y(t) & = & Cx(t) + Du(t), & t \geq 0 \end{array} \tag{1}$$

with $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{p \times n}$, and $D \in \mathbb{R}^{p \times m}$. We assume that A is stable, i.e., the spectrum of A, denoted by $\Lambda(A)$, is contained in the open left half plane. This implies that the system (1) is stable, that is, all the poles of $G(s)$ have strictly negative real parts. The number of state variables n is known as the order of the system.

We are interested in finding a reduced-order LTI system,

$$\begin{array}{rcll} \dot{\hat{x}}(t) & = & \hat{A}\hat{x}(t) + \hat{B}\hat{u}(t), & t > 0, \\ \hat{y}(t) & = & \hat{C}\hat{x}(t) + \hat{D}\hat{u}(t), & t \geq 0 \end{array} \tag{2}$$

of order r, $r \ll n$, such that the TFM $\hat{G}(s) = \hat{C}(sI - \hat{A})^{-1}\hat{B} + \hat{D}$ approximates $G(s)$.

*This work was partially supported by the DFG Research Center "Mathematics for Key technolgies" (FZT86) in Berlin, the Spanish MCyT project TIC2002-04400-C03-01 and FEDER.

†Fakultät für Mathematik, TU Chemnitz, 09107 Chemnitz, Germany. benner@mathematik.tu-chemnitz.de

‡Departamento de Ingeniería y Ciencia de Computadores, Universidad Jaume I, 12.071 Castellón, Spain. quintana@icc.uji.es

§Departamento de Ingeniería y Ciencia de Computadores, Universidad Jaume I, 12.071 Castellón, Spain. gquintan@icc.uji.es

A well-known approach for model reduction is based on state-space truncation or projected dynamics; see, e.g., the recent monographs [1, 2]. Methods based on truncated state-space transformations usually differ in the measurement of the approximation error and the way they attempt to minimize this error. Balanced truncation (BT) methods [3, 4, 5, 6], singular perturbation approximation (SPA) methods [7], and Hankel norm approximation (HNA) methods [8] all belong to the family of absolute error methods, which try to minimize $\|\Delta_a\| = \|G - \hat{G}\|$ for some system norm $\|\,.\,\|$. BT and SPA model reduction methods aim at minimizing the $\mathcal{H}_\infty$-norm norm of the error system $G - \hat{G}$, defined as

$$\|G\|_\infty = \operatorname*{ess\,sup}_{\omega \in \mathbb{R}} \sigma_{\max}(G(\jmath\omega)), \tag{3}$$

where $\jmath := \sqrt{-1}$ and $\sigma_{\max}(M)$ is the largest singular value of the matrix M. However, they usually do not succeed in finding an optimal approximation; see [9]. A different option is to use the *Hankel norm* of a stable rational transfer function, defined by

$$\|G\|_H := \sigma_1(G), \tag{4}$$

where $\sigma_1(G)$ is the largest Hankel singular value of G. Note that $\|G\|_H$ is only a semi-norm on the Hardy space H_∞ as $\|G\|_H = 0$ does not imply $G \equiv 0$. However, semi-norms are often easier to minimize than norms. In particular, using the Hankel norm it is possible to compute a best order-r approximation to a given transfer function in H_∞. It is shown in [8] that a reduced-order transfer function $\hat{G}$ of order r can be computed that minimizes the Hankel norm of the approximation error in the following sense:

$$\|G - \hat{G}\|_H = \sigma_{r+1} \leq \|G - \tilde{G}\|_H$$

for all stable transfer functions $\tilde{G}$ of McMillan degree less than or equal to r. Moreover, there are explicit formulae to compute such a realization of $\hat{G}$.

The derivation of a realization of $\hat{G}$ is quite involved. Due to space limitations, we refer the reader to [8, 10]. In the next section, we only describe the essential computational tools required in an implementation of the HNA method. We will discuss an efficient implementation of this algorithm that allows to compute an optimal HNA for large-scale systems, of order up to $\mathcal{O}(10^4)$, using parallel computing techniques. The major computational tasks involved are the computation of a minimal balanced realization, which requires the solution of two Lyapunov equations, and an additive decomposition of a transfer function into its stable and unstable parts. The additive decomposition can be performed via a block diagonalization of a matrix computed from a spectral division method followed by the solution of a Sylvester equation. All these computational tasks can be solved using special variants of Newton iteration for the sign function [11] which are specially appropriate for implementation on (parallel) high-performance computers. Numerical experiments on a cluster of Linux PCs will be reported in Section 3, showing the efficiency of our methods with respect to model reduction abilities and savings of computation time. Parallelization not only yields results faster than on serial computers, but also allows to solve larger problems on desktop computers because of the possibility of using the resources (memory) of several machines.

2 Computing the optimal Hankel norm approximation

2.1 The HNA reduced-order model

The computation of a realization $(\hat{A}, \hat{B}, \hat{C}, \hat{D})$ of the reduced-order model essentially consists of four steps.

In the first step, a balanced minimal realization of G is computed. This can be done using the square-root version of the BT method described in [12, 5] where we utilize the implementation described in [13]. That is, we compute full-rank factorizations of the system Gramians using a factored form of the sign function method for Lyapunov equations; for details of the algorithm and numerical tests reporting the efficiency, accuracy, and parallel performance of this approach see [13].

Next a transfer function

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

with the same McMillan degree as the original system (1) is computed as follows: first, the order r of the reduced-order model is chosen such that the Hankel singular values of G satisfy

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > \sigma_{r+1} = \ldots = \sigma_{r+k} > \sigma_{r+k+1} \geq \ldots\ldots \geq \sigma_{n_{\min}} > 0, \quad k \geq 1.$$

Then, by applying appropriate permutations, the minimal balanced realization of G is reordered such that the Gramians become

$$\begin{bmatrix} \check{\Sigma} & \\ & \sigma_{r+1} I_k \end{bmatrix}.$$

In a third step, the resulting balanced realization given by $\check{A}, \check{B}, \check{C}, \check{D}$ is partitioned according to the partitioning of the Gramians, i.e.,

$$\check{A} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad \check{B} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}, \quad \check{C} = [\, C_1 \;\; C_2],$$

where $A_{11} \in \mathbb{R}^{n-k \times n-k}$, $B_1 \in \mathbb{R}^{n-k \times m}$, $C_1 \in \mathbb{R}^{p \times n-k}$. Then the following formulae define a realization of $\tilde{G}$:

$$\begin{array}{rcl} \tilde{A} & = & \Gamma^{-1}(\sigma_{r+1}^2 A_{11}^T + \check{\Sigma} A_{11} \check{\Sigma} + \sigma_{r+1} C_1^T U B_1^T), \\ \tilde{B} & = & \Gamma^{-1}(\check{\Sigma} B_1 - \sigma_{r+1} C_1^T U), \\ \tilde{C} & = & C_1 \check{\Sigma} - \sigma_{r+1} U B_1^T, \\ \tilde{D} & = & D + \sigma_{r+1} U. \end{array} \tag{5}$$

Here, $U := (C_2^T)^\dagger B_2$, where $M^\dagger$ denotes the pseudoinverse of M, and $\Gamma := \check{\Sigma}^2 - \sigma_{r+1}^2 I_{n-k}$.

Finally, we compute an additive decomposition of $\tilde{G}$ such that $\tilde{G}(s) = \tilde{G}_-(s) + \tilde{G}_+(s)$ where $\tilde{G}_-$ is stable and $\tilde{G}_+$ is anti-stable. Then $\hat{G} := \tilde{G}_-$ is an optimal r-th order Hankel norm approximation of G.

We have implemented the additive decomposition of $\tilde{G}$ via a block diagonalization of $\tilde{A}$, where we first compute a block Schur form using the sign function of $\tilde{A}$ and then annihilate the off-diagonal block by solving a Sylvester equation using again a sign function-based iterative solution procedure. We will now describe this additive decomposition procedure in detail.

2.2 Additive decomposition of a transfer function

As the following procedure can be used for an additive decomposition of any transfer function having no poles on the imaginary axis, and therefore has many other applications apart from HNA, we present the method in terms of a generic realization (A, B, C, D) of a transfer function $T(s)$ without poles on the imaginary axis.

First, we introduce the sign function method [11] as it will serve as the major tool in the computations. Consider a matrix $Z \in \mathbb{R}^{n\times n}$ with $\Lambda(Z) \cap \imath\mathbb{R} = \emptyset$ and let

$$Z = S \begin{bmatrix} J^- & 0 \\ 0 & J^+ \end{bmatrix} S^{-1}$$

be its Jordan decomposition. Here, the Jordan blocks in $J^- \in \mathbb{R}^{k\times k}$ and $J^+ \in \mathbb{R}^{(n-k)\times(n-k)}$ contain, respectively, the eigenvalues of Z in the open left and right half of the complex plane, denoted as $\Lambda(J^-) \subset \mathbb{C}_-$ and $\Lambda(J^+) \subset \mathbb{C}_+$, respectively. The *matrix sign function* of Z is defined as

$$\text{sign}(Z) := S \begin{bmatrix} -I_k & 0 \\ 0 & I_{n-k} \end{bmatrix} S^{-1},$$

where I_k states for the identity matrix of order k. Note that sign (Z) is unique and independent of the order of the eigenvalues in the Jordan decomposition of Z. Many other definitions of the sign function can be given; see [14] for an overview.

The matrix sign function has proved useful in many problems involving spectral decomposition as $(I_n - \text{sign}(Z))/2$ defines the skew projector onto the stable Z–invariant subspace parallel to the unstable subspace. (By the *stable* invariant subspace of Z we denote the Z–invariant subspace corresponding to the eigenvalues of Z in $\mathbb{C}_-$.)

Applying Newton's root-finding iteration to $Z^2 = I_n$, where the starting point is chosen as Z, we obtain the Newton iteration for the matrix sign function:

$$\begin{aligned} Z_0 &\leftarrow Z, \\ Z_{k+1} &\leftarrow \frac{1}{2}(Z_k + Z_k^{-1}), \quad k = 0, 1, 2, \ldots \end{aligned} \tag{6}$$

Under the given assumptions, the sequence $\{Z_k\}_{k=0}^{\infty}$ converges to sign $(Z) = \lim_{k\to\infty} Z_k$ [11] with a locally quadratic convergence rate. As initial convergence may be slow, acceleration techniques are used, e.g., *determinantal scaling* [15]:

$$Z_k \leftarrow c_k Z_k, \quad c_k = |\det(Z_k)|^{-\frac{1}{n}},$$

where $\det(Z_k)$ denotes the determinant of Z_k. Other acceleration schemes can be employed; see [16] for a comparison of these schemes.

Once we have computed sign (Z), we can compute an orthogonal basis for the stable or anti-stable right invariant subspace of Z by computing a (rank-revealing) QR factorization [17] of $I_n - \text{sign}(Z)$ or $I_n + \text{sign}(Z)$, respectively. Specifically, let

$$I_n - \text{sign}(Z) = QRP,$$

where

$$R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \searrow & \square \\ & \end{bmatrix}, \quad R_{11} \in \mathbb{R}^{k\times k},$$

and P is a permutation matrix. Then

$$\tilde{Z} \;:=\; Q^T Z Q \;=\; \begin{bmatrix} Z_{11} & Z_{12} \\ 0 & Z_{22} \end{bmatrix} \tag{7}$$

where $\Lambda(Z_{11}) = \Lambda(Z) \cap \mathbb{C}^-$, $\Lambda(Z_{22}) = \Lambda(Z) \cap \mathbb{C}^-$.

In a second step, we compute a matrix V such that

$$\hat{Z} \;:=\; V^{-1} \tilde{Z} V \;=\; \begin{bmatrix} Z_{11} & 0 \\ 0 & Z_{22} \end{bmatrix}.$$

Note that, under the given assumption,

$$V \;=\; \begin{bmatrix} I_k & Y \\ 0 & I_{n-k} \end{bmatrix},$$

where $Y \in \mathbb{R}^{k \times n-k}$ satisfies the *Sylvester equation*

$$Z_{11} Y - Y Z_{22} + Z_{12} = 0. \tag{8}$$

As $\Lambda(Z_{11}) \cap \Lambda(Z_{22}) = \emptyset$, (8) has a unique solution [18]. Also, notice that $V^{-1} = \begin{bmatrix} I_k & -Y \\ 0 & I_{n-k} \end{bmatrix}$.

The Sylvester equation (8) can again be solved using a sign function-based procedure. For an equation

$$EY - YF + W = 0,$$

with E and $-F$ stable matrices, this iterative scheme, already derived in [11], can be written as follows

$$\begin{array}{l} E_0 \;:=\; E, \quad F_0 \;:=\; F, \quad W_0 \;:=\; W, \\ \text{for } j = 0, 1, 2, \ldots \\ \quad E_{j+1} \;:=\; \frac{1}{2}\left(E_j + E_j^{-1}\right), \\ \quad F_{j+1} \;:=\; \frac{1}{2}\left(F_j + F_j^{-1}\right), \\ \quad W_{j+1} \;:=\; \frac{1}{2}\left(W_j + E_j^{-1} W_j F_j^{-1}\right), \end{array} \tag{9}$$

It follows that $\lim_{j\to\infty} E_j = -I_k$, $\lim_{j\to\infty} F_j = I_{n-k}$, and

$$Y \;=\; \frac{1}{2} \lim_{j\to\infty} W_j.$$

For an efficient implementation of this iteration on modern computer architectures and numerical experiments reporting efficiency and accuracy, see [19].

Applying the above steps to the matrix A from the realization of $T(s)$ we now obtain the desired additive decomposition of $T(s) = C(sI - A)^{-1}B + D$ as follows: perform a state space transformation

$$(\hat{A}, \hat{B}, \hat{C}\hat{D}) \;:=\; (V^{-1}Q^T A Q V, V^{-1}Q^T B, CQV, D)$$

and partition

$$\hat{A} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \; \hat{B} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}, \; \hat{C} = [\, C_1 \;\; C_2],$$

according to the partitioning in (7). Then

$$\begin{aligned} T(s) &= C(sI-A)^{-1}B + D = \hat{C}(sI-\hat{A})^{-1}\hat{B} + \hat{D} \\ &= \left[\begin{array}{cc} C_1 & C_2 \end{array}\right] \left[\begin{array}{cc} (sI_k - A_{11})^{-1} & \\ & (sI_{n-k} - A_{22})^{-1} \end{array}\right] \left[\begin{array}{c} B_1 \\ B_2 \end{array}\right] + D \\ &= \{\hat{C}_1(sI_k - A_{11})^{-1}B_1\} + \{C_2(sI_{n-k} - A_{22})^{-1}B_2 + D\} \\ &=: T_-(s) + T_+(s), \end{aligned}$$

where $T_-(s)$ is a stable transfer function and $T_+(s)$ is an anti-stable transfer function.

The complete procedure for computing an optimal r–th order Hankel norm approximation of a transfer function is summarized in Table 1

1. Compute a balanced minimal realization of G with Gramians

$$P = Q = \mathrm{diag}(\sigma_1, \ldots, \sigma_{n_{\min}}),$$

where $n_{\min}$ is the McMillan degree of the system.

2. Choose r such that $\sigma_{r+1} > \sigma_r$. Permute the balanced realization such that the Gramians of the transformed system are

$$\begin{aligned} \tilde{P} = \tilde{Q} &= \mathrm{diag}(\sigma_1, \ldots, \sigma_r, \sigma_{r+1+m_{r+1}}, \ldots, \sigma_{n_{\min}}, \sigma_{r+1} I_{m_{r+1}}) \\ &=: \mathrm{diag}(\Sigma, \sigma_{r+1} I_{m_{r+1}}), \end{aligned}$$

where m_{r+1} is the multiplicity of σ_{r+1} as a Hankel singular value of G.

3. Partition the resulting realization

$$\tilde{A} =: \left[\begin{array}{cc} A_{11} & A_{12} \\ A_{21} & A_{22} \end{array}\right], \tilde{B} =: \left[\begin{array}{c} B_1 \\ B_2 \end{array}\right], \tilde{C} =: \left[\begin{array}{cc} C_1 & C_2 \end{array}\right],$$

such that $A_{22} \in \mathbb{R}^{m_{r+1} \times m_{r+1}}$. Then compute $\hat{A}, \hat{B}, \hat{C}, \hat{D}$ as in (5).

4. Compute an additive decomposition

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

with stable $\hat{G} := \hat{G}_-$ and anti-stable $\hat{G}_+$ using the method described in Section 2.2.

Table 1: Computing an optimal r–th order Hankel norm approximation

3 Numerical experiments

We implemented the algorithm described in Table 1 as subroutine `PAB09CX` of the Parallel Library in Control – Model Reduction (PLiCMR) [20] using the message-passing paradigm and the parallel linear algebra kernels in ScaLAPACK [21]. In order to compute a minimal realization we used the PLiCMR version of balanced truncation, described in [13].

Example	n	m	p	σ_1	η	r
Eady	598	1	1	$9.9 \times 10^{+2}$	1.0×10^{-3}	9
CDplayer	120	2	2	$1.2 \times 10^{+6}$	1.0×10^{-8}	42
FOM	1006	1	1	$5.0 \times 10^{+1}$	1.0×10^{-3}	10
PDE	84	1	1	$5.3 \times 10^{+0}$	1.0×10^{-3}	2
Heat	200	1	1	3.3×10^{-2}	1.0×10^{-3}	4
ISS	270	3	3	5.8×10^{-2}	1.0×10^{-3}	36
Build	48	1	1	2.5×10^{-3}	1.0×10^{-3}	30
Beam	348	1	1	$2.4 \times 10^{+3}$	1.0×10^{-3}	12

Table 2: Parameters of the examples employed in the numerical evaluation of the parallel model reduction routines.

All the experiments presented in this section were performed on a cluster of 32 nodes using IEEE double-precision floating-point arithmetic ($\varepsilon \approx 2.2204 \times 10^{-16}$). Each node consists of an Intel Pentium-II processor at 300 MHz, and 128 MBytes of RAM. We employ a BLAS library, specially tuned for the Pentium-II processor as part of the ATLAS and ITXGEMM projects [22, 23], that achieves around 180 Mflops (millions of flops per second) for the matrix product (routine `DGEMM`). The nodes are connected via a *Myrinet* crossbar network; the communication library BLACS is based on an implementation of MPI specially developed and tuned for this network. The performance of the interconnection network was measured by a simple loop-back message transfer resulting in a latency of 33 μsec and a bandwidth of 200 Mbit/sec. We made use of the LAPACK, PBLAS, and ScaLAPACK libraries wherever possible.

We first evaluate the accuracy of the Hankel norm approximations by comparing the frequency responses of the transfer functions from input k to output j of the reduced-order systems computed with the SLICOT subroutine `AB09CD` [24, 25] and our parallel PLiCMR routine `PAB09CX`. In other words, we compare $|G_{jk}(\jmath\omega)|$, $1 \leq j \leq p$, $1 \leq k \leq m$, with the $\hat{G}_{jk}$ of the computed reduced-order systems.

For comparison purposes we test the routines for eight different examples from very different applications. For a detailed description of the models see [26] and the references therein. Table 2 shows the parameters of the systems used here. In order to fix the order of the reduced-order system automatically, both the SLICOT and PLiCMR routines select r so that

$$\sigma_r > \max(\tau_1, n \cdot \varepsilon \cdot \sigma_1) > \sigma_{r+1},$$

where ε is the machine precision and τ_1 is a user-specified tolerance threshold. In our case, we set $\tau_1 = \eta \cdot \sigma_1$, where the value η is adjusted for each particular case as shown in the table. In order to determine a minimal realization of the system, a second tolerance $\max(\tau_2, n \cdot \varepsilon \cdot \sigma_1)$ is needed. In our experiments we used $\tau_2 = 0$.

The results summarized in Table 3 show that the approximation error, here measured in the H_∞ norm, is equal in all cases except for the CD player example. However, Figure 1 shows that the approximation quality for this particular example is equal for both implementations.

Due to space limitations, we do not display the Bode plots for all tested examples. The more interesting ones are the ISS (International Space Station), the building and beam

Example	AB09CD	PAB09CX
Eady	4.9×10^{-1}	4.9×10^{-1}
CDplayer	3.8×10^{-2}	5.9×10^{-2}
FOM	7.1×10^{-2}	7.1×10^{-2}
PDE	7.4×10^{-3}	7.4×10^{-3}
Heat	2.9×10^{-5}	2.9×10^{-5}
ISS	1.1×10^{-4}	1.1×10^{-4}
Build	4.6×10^{-6}	4.6×10^{-6}
Beam	$2.0 \times 10^{+0}$	$2.0 \times 10^{+0}$

Table 3: Absolute errors of the reduced-order models computed by the SLICOT and PLiCMR HNA subroutines.

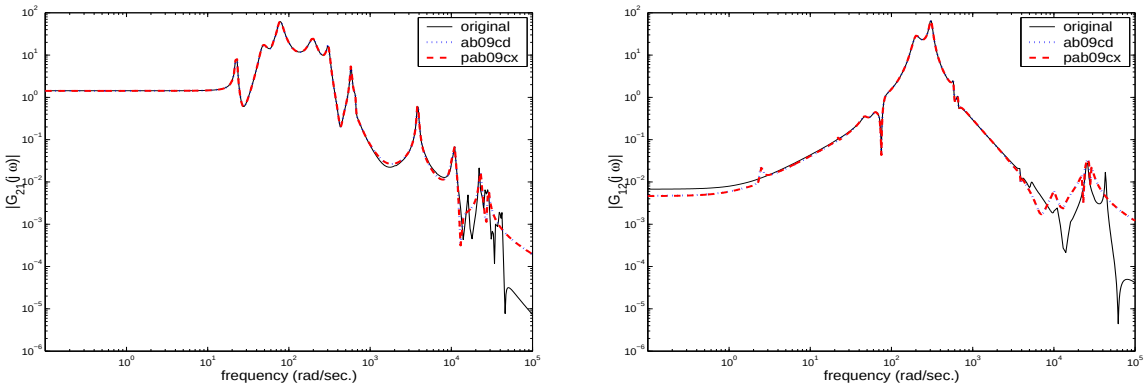

Figure 1: Accuracy of HNAs for CD Player example. G_{11}, G_{22} are easy to approximate and are therefore not displayed.

examples—for the other examples, the transfer function is smooth and easy to approximate. Figure 2 shows the magnitudes of two of the nine input-output maps for the ISS which exhibit the typical behavior in this example; Figure 3 shows the frequency responses for the building and beam examples. The plots also report equal approximation quality for the SLICOT and PLiCMR routines.

Next we analyze the performance of our parallel HNA subroutine using a random scalable LTI system constructed as follows. First, we generate a random positive semidefinite diagonal Gramian $W_c = \mathrm{diag}(\Sigma_{q_1}, \Sigma_{q_2}, 0_{q_3}, 0_{q_4})$, where $\Sigma_{q_1} \in \mathbb{R}^{q_1 \times q_1}$ contains the desired Hankel singular values for the system and $\Sigma_{q_2} \in \mathbb{R}^{q_2 \times q_2}$. Then, we construct a random positive semidefinite diagonal Gramian $W_o = \mathrm{diag}(\Sigma_{q_1}, 0_{q_2}, \Sigma_{q_3}, 0_{q_4})$, with $\Sigma_{q_3} \in \mathbb{R}^{q_3 \times q_3}$. Next, we set A to a random stable diagonal matrix and compute $F = -(AW_c + W_c A^T)$ and $G = -(A^T W_o + W_o A)$. Thus,

$$\begin{aligned} F &= \mathrm{diag}(f_1, f_2, \ldots, f_{q_1+q_2}, 0_{q_3+q_4}), \\ G &= \mathrm{diag}(g_1, g_2, \ldots, g_{q_1}, 0, \ldots, 0, g_{q_1+q_2+1}, \ldots, g_{q_1+q_2+q_3}, 0_{q_4}). \end{aligned}$$

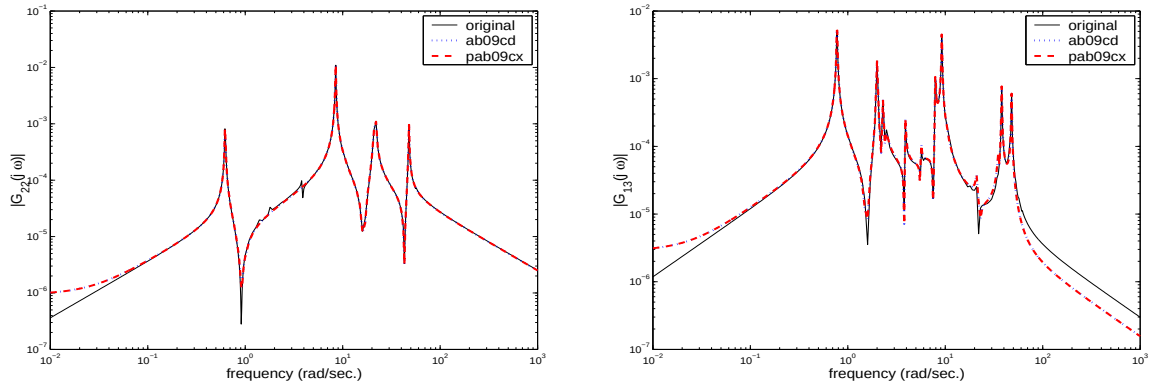

Figure 2: Accuracy of HNAs for ISS example. The approximations of all other G_{ij} behave similarly.

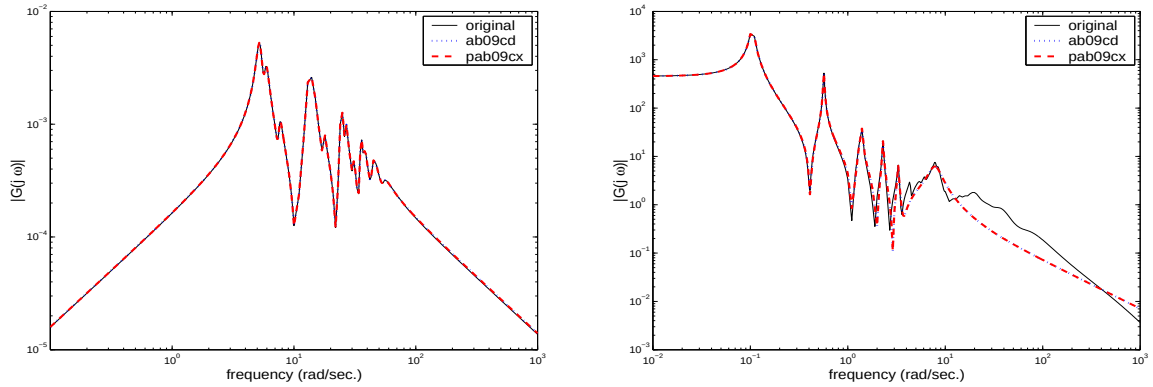

Figure 3: Accuracy of HNAs for the Building example (left plot) and the Beam example (right plot).

A matrix $B \in \mathbb{R}^{n\times(q_1+q_2)}$ such that $F = BB^T$ is then obtained as

$$B \;=\; \mathrm{diag}(\sqrt{f_1}, \sqrt{f_2}, \ldots, \sqrt{f_{q_1+q_2}}).$$

The procedure for obtaining C is analogous. The LTI system is finally transformed into $A := U^T AU$, $B := U^T B$, and $C := CU$ using a random orthogonal transformation $U \in \mathbb{R}^{n\times n}$. The system thus defined has a minimal realization of order $r = q_1$. The Cholesky factors satisfy rank $(S) = q_1 + q_2$ and rank $(R) = q_1 + q_3$.

We evaluate the efficiency of the parallel algorithm computing a reduced model of order $r = 40$ of a random LTI system with $n = 1000$, $m = p = 100$. Figure 4 reports the execution times of the serial routines for model reduction in SLICOT and the corresponding parallel algorithms as the number of nodes, n_p, is increased. The results show a considerable acceleration achieved by the parallel algorithm (with even super speed-ups). This is partially due to the efficiency of the Lyapunov solvers used in our algorithms which compute factors of the Gramians in compact (full-rank) form instead of square matrices, thus requiring less computations. Comparison of the results on 2 and 4 nodes roughly shows the efficiency of the parallel algorithm. The execution time is reduced by a factor of almost 2 (the number

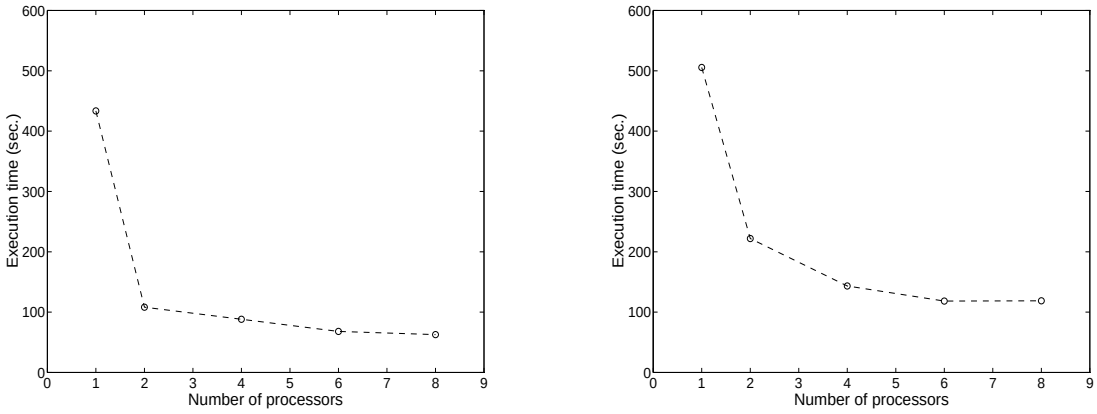

Figure 4: Execution times of HNA subroutines `AB09CD` (result on 1 node) and `PAB09CX` (results on more than 1 node) for the random system ($n = 1000$) with $q_1 = q_2 = q_3 = 25$ (left plot) and $q_1 = q_2 = q_3 = 100$ (right plot).

of resources, that is nodes, is doubled). Using a larger number of nodes does not achieve a significant reduction of the execution time due to the small ratio $n/\sqrt{n_p}$.

4 Conclusions

We have presented a numerical algorithm for computing optimal Hankel norm approximations to rational transfer functions. The method is suitable for parallel computing and allows the application of HNA to systems of order as big as $n = \mathcal{O}(10^4)$. Several examples demonstrate the accuracy and efficiency of our parallel implementation.

References

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

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

[3] B. Moore, "Principal component analysis in linear systems: Controllability, observability, and model reduction," *IEEE Trans. Automat. Control*, vol. AC-26, pp. 17–32, 1981.

[4] M. Safonov and R. Chiang, "A Schur method for balanced-truncation model reduction," *IEEE Trans. Automat. Control*, vol. AC–34, pp. 729–733, 1989.

[5] M. Tombs and I. Postlethwaite, "Truncated balanced realization of a stable non-minimal state-space system," *Internat. J. Control*, vol. 46, no. 4, pp. 1319–1330, 1987.

[6] A. Varga, "Efficient minimal realization procedure based on balancing," in *Prepr. of the IMACS Symp. on Modelling and Control of Technological Systems*, vol. 2, 1991, pp. 42–47.

[7] Y. Liu and B. Anderson, "Controller reduction via stable factorization and balancing," *Internat. J. Control*, vol. 44, pp. 507–531, 1986.

[8] K. Glover, "All optimal Hankel-norm approximations of linear multivariable systems and their L^∞ norms," *Internat. J. Control*, vol. 39, pp. 1115–1193, 1984.

[9] A. Antoulas and A. Astolfi, "H_∞-norm approximation," in *2002 MTNS Problem Book, Open Problems on the Mathematical Theory of Networks and Systems*, V. Blondel and A. Megretski, Eds., 2002, pp. 73–76, available online from `http://www.nd.edu/~mtns/OPMTNS.pdf`.

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

[11] J. Roberts, "Linear model reduction and solution of the algebraic Riccati equation by use of the sign function," *Internat. J. Control*, vol. 32, pp. 677–687, 1980, (Reprint of Technical Report No. TR-13, CUED/B-Control, Cambridge University, Engineering Department, 1971).

[12] A. Laub, M. Heath, C. Paige, and R. Ward, "Computation of system balancing transformations and other application of simultaneous diagonalization algorithms," *IEEE Trans. Automat. Control*, vol. 34, pp. 115–122, 1987.

[13] P. Benner, E. Quintana-Ortí, and G. Quintana-Ortí, "Balanced truncation model reduction of large-scale dense systems on parallel computers," *Math. Comput. Model. Dyn. Syst.*, vol. 6, no. 4, pp. 383–405, 2000.

[14] C. Kenney and A. Laub, "The matrix sign function," *IEEE Trans. Automat. Control*, vol. 40, no. 8, pp. 1330–1348, 1995.

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

[16] Z. Bai and J. Demmel, "Design of a parallel nonsymmetric eigenroutine toolbox, Part I," in *Proceedings of the Sixth SIAM Conference on Parallel Processing for Scientific Computing*, R. S. et al., Ed. SIAM, Philadelphia, PA, 1993, pp. 391–398,

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

[18] P. Lancaster and M. Tismenetsky, *The Theory of Matrices*, 2nd ed. Orlando: Academic Press, 1985.

[19] P. Benner, E. Quintana-Ortí, and G. Quintana-Ortí, "Solving linear matrix equations via rational iterative schemes," in preparation.

[20] ——, "State-space truncation methods for parallel model reduction of large-scale systems," *Parallel Comput.*, vol. 29, pp. 1701–1722, 2003.

[21] L. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, and R. Whaley, *ScaLAPACK Users' Guide*, SIAM, Philadelphia, PA, 1997.

[22] R. C. Whaley and J. Dongarrra, "Automatically tuned linear algebra software," in *Proc. of SC98*, 1998.

[23] J. Gunnels, G. Henry, and R. van de Geijn, "A family of high-performance matrix multiplication algorithms," in *Computational Science - ICCS 2001, Part I, Lecture Notes in Computer Science 2073*, V. Alexander, J. Dongarra, B. Julianno, R. Renner, and C. Kenneth Tan, Eds., 2001, pp. 51–60.

[24] P. Benner, V. Mehrmann, V. Sima, S. V. Huffel, and A. Varga, "SLICOT - a subroutine library in systems and control theory," in *Applied and Computational Control, Signals, and Circuits*, B. Datta, Ed. Boston, MA: Birkhäuser, 1999, vol. 1, ch. 10, pp. 499–539.

[25] A. Varga, "Model reduction software in the SLICOT library," in *Applied and Computational Control, Signals, and Circuits*, ser. The Kluwer International Series in Engineering and Computer Science, B. Datta, Ed. Boston, MA: Kluwer Academic Publishers, 2001, vol. 629, pp. 239–282.

[26] Y. Chahlaoui and P. Van Dooren, "A collection of benchmark examples for model reduction of linear time invariant dynamical systems," SLICOT Working Note 2002–2, Feb. 2002, available from `http://www.win.tue.nl/niconet/NIC2/reports.html`.