Model Reduction Based on Spectral Projection Methods

Peter Benner* Enrique S. Quintana-Ortí†

April 8, 2005

Abstract

We discuss the efficient implementation of model reduction methods such as modal truncation, balanced truncation, and other balancing-related truncation techniques, employing the idea of spectral projection. Mostly, we will be concerned with the sign function method which serves as the major computational tool of most of the discussed algorithms for computing reduced-order models. Implementations for large-scale problems based on parallelization or formatted arithmetic will also be discussed. This chapter can also serve as a tutorial on Gramian-based model reduction using spectral projection methods.

1 Introduction

Consider the linear, time-invariant (LTI) system

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

where $A \in \mathbb{R}^{n\times n}$ is the state matrix, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$, and $x^0 \in \mathbb{R}^n$ is the initial state of the system. Here, n is the order (or state-space dimension) of the system. The associated transfer function matrix (TFM) obtained from taking Laplace transforms in (1) and assuming $x_0 = 0$ is

$$G(s) = C(sI - A)^{-1}B + D. \tag{2}$$

In model reduction we are faced with the problem of finding a reduced-order LTI system,

$$\begin{array}{rcll} \dot{\hat{x}}(t) & = & \hat{A}\hat{x}(t) + \hat{B}\hat{u}(t), & t > 0 \quad \hat{x}(0) = \hat{x}^0, \\ \hat{y}(t) & = & \hat{C}\hat{x}(t) + \hat{D}\hat{u}(t), & t \geq 0, \end{array} \tag{3}$$

of order r, $r \ll n$, and associated TFM $\hat{G}(s) = \hat{C}(sI - \hat{A})^{-1}\hat{B} + \hat{D}$ which approximates $G(s)$. Model reduction of discrete-time LTI systems can be formulated in an analogous manner; see, e.g., [OA01]. Most of the methods and approaches discussed here carry over to the discrete-time setting as well. Here, we will focus our attention on the continuous-time setting, the discrete-time case being discussed in detail in [BQQ03a].

Balancing-related model reduction methods are based on finding an appropriate coordinate system for the state-space in which the chosen Gramian matrices of the system are

*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.

diagonal and equal. In the simplest case of balanced truncation, the controllability Gramian W_c and the observability Gramian W_o are used. These Gramians are given by the solutions of the two dual *Lyapunov equations*

$$AW_c + W_cA^T + BB^T = 0, \quad A^TW_o + W_oA + C^TC = 0. \tag{4}$$

After changing to the coordinate system giving rise to diagonal Gramians with positive decreasing diagonal entries, which are called the Hankel singular values (HSVs) of the system, the reduced-order model is obtained by truncating the states corresponding to the $n - r$ smallest HSVs.

Balanced truncation and its relatives such as singular perturbation approximation, stochastic truncation, etc., are the most popular model reduction techniques used in control theory. The advantages of these methods, guaranteed preservation of several system properties like stability and passivity, as well as the existence of computable error bounds that permit an adaptive selection of the order of the reduced-order model, are unmatched by any other approach. However, thus far, in many other engineering disciplines the use of balanced truncation and other related methods has not been considered feasible due to its computational complexity. Quite often, these disciplines have a preferred model reduction technique as modal analysis and Guyan reduction in structural dynamics, proper orthogonal decomposition (POD) in computational fluid dynamics, Padé and Padé-like approximation techniques based on Krylov subspace methods in circuit simulation and microsystem technology, etc. *A goal of this tutorial is to convince the reader that balanced truncation and its relatives are viable alternatives in many of these areas if efficient algorithms from numerical linear algebra are employed and/or basic level parallel computing facilities are available.*

The ideas presented in this paper are part of an ongoing effort to facilitate the use of balancing-related model reduction methods in large-scale problems arising in the control of partial differential equations, the simulation of VLSI and ULSI circuits, the generation of compact models in microsystems, and other engineering disciplines. This effort mainly involves breaking the $\mathcal{O}(n^2)$ memory and $\mathcal{O}(n^3)$ flops (floating-point arithmetic operations) barriers. Several issues related to this challenge are addressed in this paper. By working with (approximations of) the full-rank factors of the system Gramians rather than using Cholesky factors as in previous balanced truncation algorithms, the complexity of all remaining calculations following the computation of the factors of the Gramians usually only grows linearly with the dimension of the state-space. This idea is pursued in several approaches that essentially only differ in the way the factors of the Gramians are computed. Approximation methods suitable for sparse systems based mainly on Smith- and ADI-type methods are discussed in Chapters [GL05] and [MS05]. These allow the computation of the factors at a computational cost and a memory requirement proportional to the number of nonzeros in A. Thus, implementations of balanced truncation based on these ideas are in the same complexity class as Padé-approximation and POD. In this chapter, we focus on the computation of full-rank factors of the Gramians by the sign function method which is based on spectral projection techniques. This does not lead immediately to a reduced overall complexity of the induced balanced truncation algorithm as we deal with general dense systems. However, for special classes of dense problems, a linear-polylogarithmic complexity can be achieved by employing hierarchical matrix structures and the related formatted arithmetic. For the general case, the $\mathcal{O}(n^2)$ memory and $\mathcal{O}(n^3)$ flops complexity remains, but the resulting algorithms are perfectly suited for parallel computations and are highly efficient on current desktops or clusters

of workstations. Provided efficient parallel computational kernels for the necessary linear algebra operations are available, balanced truncation can be applied to systems with state-space dimension $n = \mathcal{O}(10^4)$ and dense A-matrix on commodity clusters. By re-using these efficient parallel kernels for computing reduced-order models with a sign function-based implementation of balanced truncation, the application of many other related model reduction methods to large-scale, dense systems becomes feasible. We briefly describe some of the related techniques in this chapter, particularly we discuss sign function-based implementations of the following methods:

- balanced truncation,
- singular perturbation approximation,
- optimal Hankel norm approximation,
- balanced stochastic truncation, and
- truncation methods based on positive real, bounded real, and LQG balancing,

for stable systems. Using a specialized algorithm for the additive decomposition of transfer functions, again based on spectral projection techniques, all the above balancing-related model reduction techniques can also be applied to unstable systems. At this point, we would also like to mention that the same ideas can be applied to balanced truncation for descriptor systems, as described in Chapter [MS05]—for preliminary results see [BQQ04c]—but we will not elaborate on this as this is mostly work in progress.

This paper is organized as follows. In Section 2 we provide the necessary background from system and realization theory. Spectral projection, which is the basis for many of the methods described in this chapter, is presented in Section 3. Model reduction methods for stable systems of the form (1) based on these ideas are described in Section 4, where we also include modal truncation for historical reasons. The basic ideas needed to apply balanced truncation and its relatives to large-scale systems are summarized in Section 5. Conclusions and open problems are given in Section 6.

Throughout this paper, we will use I_n for the identity matrix in $\mathbb{R}^{n\times n}$ and I for the identity when the order is obvious from the context, $\Lambda(A)$ will denote the spectrum of the matrix A. Usually, capital letters will be used for matrices; lower case letters will stand for vectors with the exception of t denoting time, and i, j, k, m, n, p, r, s employed for integers such as indices and dimensions; Greek letters will be used for other scalars; and calligraphic letters will indicate vector and function spaces. Without further explanation, Π will always denote a permutation matrix of a suitable dimension, usually resulting from row or column pivoting in factorization algorithms. The left and right (open) complex half planes will be denoted by $\mathbb{C}^-$ and $\mathbb{C}^+$, respectively, and we will write $\jmath$ for $\sqrt{-1}$.

2 System-Theoretic Background

In this section, we introduce some basic notation and properties of LTI systems used throughout this paper. More detailed introductions to LTI systems can be found in many textbooks [GL95, Son98, ZDG96] or handbooks [Lev96, Mut99]. We essentially follow these references here without further citations, but many other sources can be used for a good overview on the subjects covered in this section.

2.1 Linear Systems, Frequency Domain, and Norms

An LTI system is (Lyapunov or exponentially) stable if all its poles are in the left half plane. Sufficient for this is that A is stable (or *Hurwitz*), i.e., the spectrum of A, denoted by $\Lambda(A)$, satisfies $\Lambda(A) \subset \mathbb{C}^-$. It should be noted that the relation between the controllability and observability Gramians of an LTI system and the solutions of the Lyapunov equations in (4) only holds if A is stable.

The particular model imposed by (1), given by a differential equation describing the behavior of the states x and an algebraic equation describing the outputs y is called a *state-space representation*. Alternatively, the relation between inputs and outputs can also be described in the *frequency domain* by an algebraic expression. Applying the *Laplace transform* to the two equations in (1), and denoting the transformed arguments as $x(s)$, $y(s)$, $u(s)$ where s is the Laplace variable, we obtain

$$\begin{aligned} sx(s) - x(0) &= Ax(s) + Bu(s), \\ y(s) &= Cx(s) + Du(s). \end{aligned}$$

By solving for $x(s)$ in the first equation and inserting this into the second equation, we obtain

$$y(s) = \left(C(sI_n - A)^{-1}B + D\right)u(s) + C(sI_n - A)^{-1}x^0.$$

For a zero initial state, the relation between inputs and outputs is therefore completely described by the *transfer function*

$$G(s) := C(sI_n - A)^{-1}B + D. \tag{5}$$

Many interesting characteristics of an LTI system are obtained by evaluating $G(s)$ on the positive imaginary axis, that is, setting $s = \jmath\omega$. In this context, ω can be interpreted as the operating frequency of the LTI system.

A stable transfer function defines a mapping

$$G : \mathcal{L}_2 \to \mathcal{L}_2 : u \to y = Gu \tag{6}$$

where the two function spaces denoted by $\mathcal{L}_2$ are actually different spaces and should more appropriately be denoted by $\mathcal{L}_2(\mathbb{C}^m)$ and $\mathcal{L}_2(\mathbb{C}^p)$, respectively. As the dimension of the underlying spaces will always be clear from the context, i.e., the dimension of the transfer function matrix $G(s)$ or the dimension of input and output spaces, we allow ourselves the more sloppy notation used in (6). The function space $\mathcal{L}_2$ contains the square integrable functions in the frequency domain, obtained via the Laplace transform of the square integrable functions in the time domain, usually denoted as $\mathcal{L}_2(-\infty, \infty)$. The $\mathcal{L}_2$-functions that are analytic in the open right half plane $\mathbb{C}^+$ form the *Hardy space* $\mathcal{H}_2$. Note that $\mathcal{H}_2$ is a closed subspace of $\mathcal{L}_2$. Under the Laplace transform $\mathcal{L}_2$ and $\mathcal{H}_2$ are isometric isomorphic to $\mathcal{L}_2(-\infty, \infty)$ and $\mathcal{L}_2[0, \infty)$, respectively. (This is essentially the *Paley-Wiener Theorem* which is the Laplace transform analog of Parseval's identity for the Fourier transform.) Therefore it is clear that the frequency domain spaces $\mathcal{H}_2$ and $\mathcal{L}_2$ can be endowed with the corresponding norms from their time domain counterparts. Due to this isometry, our notation will not distinguish between norms for the different spaces so that we will denote by $\|f\|_2$ the induced 2-norm on any of the spaces $\mathcal{L}_2(-\infty, \infty)$, $\mathcal{L}_2$, $\mathcal{L}_2[0, \infty)$, and $\mathcal{H}_2$. Using the definition (6), it is therefore possible to define an operator norm for G by

$$\|G\| := \sup_{\|u\|_2 \leq 1} \|Gu\|_2.$$

It turns out that this operator norm equals the $\mathcal{L}_\infty$-norm of the transfer function G, which for rational transfer functions can be defined as

$$\|G\|_\infty := \sup_{\omega\in\mathbb{R}} \sigma_{\max}(G(\jmath\omega)). \tag{7}$$

The $p \times m$-matrix-valued functions G for which $\|G\|_\infty$ is bounded, i.e., those essentially bounded on the imaginary axis, form the function space $\mathcal{L}_\infty$. The subset of $\mathcal{L}_\infty$ containing all $p \times m$-matrix-valued functions that are analytical and bounded in $\mathbb{C}^+$ form the Hardy space $\mathcal{H}_\infty$. As a consequence of the maximum modulus theorem, $\mathcal{H}_\infty$ functions must be bounded on the imaginary axis so that the essential supremum in (7) simplifies to a supremum for rational functions G. Thus, the $\mathcal{H}_\infty$-norm of the rational transfer function $G \in \mathcal{H}_\infty$ can be defined as

$$\|G\|_\infty := \sup_{\omega\in\mathbb{R}} \sigma_{\max}(G(\jmath\omega)). \tag{8}$$

A fact that will be of major importance throughout this paper is that the transfer function of a stable LTI system is rational with no poles in the closed right-half plane. Thus, $G \in \mathcal{H}_\infty$ for all stable LTI systems.

Although the notation is somewhat misleading, the $\mathcal{H}_\infty$-norm is the 2-induced operator norm. Hence the sub-multiplicativity condition

$$\|y\|_2 \leq \|G\|_\infty \|u\|_2 \tag{9}$$

holds. This inequality implies an important way to tackle the model reduction problem: suppose the original system and the reduced-order model (3) are driven by the same input function $u \in \mathcal{H}_2$, so that

$$y(s) = G(s)u(s), \qquad \hat{y}(s) = \hat{G}(s)u(s),$$

where $\hat{G}$ is the transfer function corresponding to (3); then we obtain the error bound

$$\|y - \hat{y}\|_2 \leq \|G - \hat{G}\|_\infty \|u\|_2. \tag{10}$$

Due to the aforementioned Paley-Wiener theorem, this bound holds in the frequency domain and the time domain. Therefore a goal of model reduction is to compute the reduced-order model so that $\|G - \hat{G}\|_\infty$ is smaller than a given tolerance threshold.

2.2 Balanced Realizations

A realization of an LTI system is the set of the four matrices

$$(A, B, C, D) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times m} \times \mathbb{R}^{p\times n} \times \mathbb{R}^{p\times m}$$

corresponding to (1). In general, an LTI system has infinitely many realizations as its transfer function is invariant under state-space transformations,

$$\mathcal{T} : \left\{ \begin{array}{ccc} x & \to & Tx, \\ (A, B, C, D) & \to & (TAT^{-1}, TB, CT^{-1}, D), \end{array} \right. \tag{11}$$

as the simple calculation

$$D + (CT^{-1})(sI - TAT^{-1})^{-1}(TB) = C(sI_n - A)^{-1}B + D = G(s)$$

demonstrates. But this is not the only non-uniqueness associated to LTI system representations. Any addition of states that does not influence the input-output relation, meaning that for the same input u the same output y is achieved, leads to a realization of the same LTI system. Two simple examples are

$$\frac{d}{dt}\begin{bmatrix} x \\ x_1 \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & A_1 \end{bmatrix}\begin{bmatrix} x \\ x_1 \end{bmatrix} + \begin{bmatrix} B \\ B_1 \end{bmatrix} u(t), \quad y(t) = \begin{bmatrix} C & 0 \end{bmatrix}\begin{bmatrix} x \\ x_1 \end{bmatrix} + Du(t),$$

$$\frac{d}{dt}\begin{bmatrix} x \\ x_2 \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & A_2 \end{bmatrix}\begin{bmatrix} x \\ x_2 \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u(t), \quad y(t) = \begin{bmatrix} C & C_2 \end{bmatrix}\begin{bmatrix} x \\ x_2 \end{bmatrix} + Du(t),$$

for arbitrary matrices $A_j \in \mathbb{R}^{n_j \times n_j}$, $j = 1, 2$, $B_1 \in \mathbb{R}^{n_1 \times m}$, $C_2 \in \mathbb{R}^{p \times n_2}$ and any $n_1, n_2 \in \mathbb{N}$. An easy calculation shows that both of these systems have the same transfer function $G(s)$ as (1) so that

$$(A, B, C, D),\ \left(\begin{bmatrix} A & 0 \\ 0 & A_1 \end{bmatrix}, \begin{bmatrix} B \\ B_1 \end{bmatrix}, \begin{bmatrix} C & 0 \end{bmatrix}, D\right),\ \left(\begin{bmatrix} A & 0 \\ 0 & A_2 \end{bmatrix}, \begin{bmatrix} B \\ 0 \end{bmatrix}, \begin{bmatrix} C & C_2 \end{bmatrix}, D\right)$$

are both realizations of the same LTI system described by the transfer function $G(s)$ in (5). Therefore, the order n of a system can be arbitrarily enlarged without changing the input-output mapping. On the other hand, for each system there exists a unique minimal number of states which is necessary to describe the input-output behavior completely. This number $\hat{n}$ is called the *McMillan degree* of the system. A *minimal realization* is a realization $(\hat{A}, \hat{B}, \hat{C}, \hat{D})$ of the system with order $\hat{n}$. Note that only the McMillan degree is unique; any state-space transformation (11) leads to another minimal realization of the same system. Finding a minimal realization for a given system can be considered as a first step of model reduction as redundant (non-minimal) states are removed from the system. Sometimes this is part of a model reduction procedure, e.g. optimal Hankel norm approximation, and can be achieved via balanced truncation.

Although realizations are highly non-unique, stable LTI systems have a set of invariants with respect to state-space transformations that provide a good motivation for finding reduced-order models. From Lyapunov stability theory (see, e.g., [LT85, Chapter 13]) it is clear that for stable A, the Lyapunov equations in (4) have unique positive semidefinite solutions W_c and W_o. These solutions define the *controllability Gramian* (W_c) and *observability Gramian* (W_o) of the system. If W_c is positive definite, then the system is controllable and if W_o is positive definite, the system is observable. Controllability plus observability is equivalent to minimality of the system so that for minimal systems, all eigenvalues of the product $W_c W_o$ are strictly positive real numbers. The square roots of these eigenvalues, denoted in decreasing order by

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n > 0,$$

are known as the *Hankel singular values (HSVs)* of the LTI system and are invariants of the system: let

$$(\hat{A}, \hat{B}, \hat{C}, D) = (TAT^{-1}, TB, CT^{-1}, D)$$

be the transformed realization with associated controllability Lyapunov equation

$$0 = \hat{A}\hat{W}_c + \hat{W}_c\hat{A}^T + \hat{B}\hat{B}^T = TAT^{-1}\hat{W}_c + \hat{W}_c T^{-T}A^T T^T + TBB^T T^T.$$

This is equivalent to

$$0 = A(T^{-1}\hat{W}_c T^{-T}) + (T^{-1}\hat{W}_c T^{-T})A^T + BB^T.$$

The uniqueness of the solution of the Lyapunov equation (see, e.g., [LT85]) implies that $\hat{W}_c = TW_cT^T$ and, analogously, $\hat{W}_o = T^{-T}W_oT^{-1}$. Therefore,

$$\hat{W}_c\hat{W}_o = TW_cW_oT^{-1},$$

showing that $\Lambda(\hat{W}_c\hat{W}_o) = \Lambda(W_cW_o) = \{\sigma_1^2, \ldots, \sigma_n^2\}$. Note that extending the state-space by non-minimal states only adds HSVs of magnitude equal to zero, while the non-zero HSVs remain unchanged.

An important (and name-inducing) type of realizations are *balanced realizations*. A realization (A, B, C, D) is called *balanced* iff

$$W_c = W_o = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{bmatrix};$$

that is, the controllability and observability Gramians are diagonal and equal with the decreasing HSVs on their respective diagonal entries. For a minimal realization there always exists a balancing state-space transformation of the form (11) with nonsingular matrix $T_b \in \mathbb{R}^{n\times n}$; for non-minimal systems the Gramians can also be transformed into diagonal matrices with the leading $\hat{n} \times \hat{n}$ submatrices equal to $\mathrm{diag}(\sigma_1, \ldots, \sigma_{\hat{n}})$, and

$$\hat{W}_c\hat{W}_o = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_{\hat{n}}^2, 0, \ldots, 0);$$

see, e.g., [TP87]. Using a balanced realization obtained via the transformation matrix T_b, the HSVs allow an energy interpretation of the states; see also [Van00] for a nice treatment of this subject. Specifically, the minimal energy needed to reach x^0 is

$$\inf_{\substack{u\in\mathcal{L}_2(-\infty,0]\\ x(0)=x^0}} \int_{-\infty}^{0} u(t)^T u(t)\,dt = (x^0)^T W_c^{-1} x^0 = (\hat{x}^0)^T \hat{W}_c^{-1}\hat{x}^0 = \sum_{k=1}^{n} \frac{1}{\sigma_k}\hat{x}_k^2,$$

where $\hat{x}^0 := \begin{bmatrix} \hat{x}_1 \\ \vdots \\ \hat{x}_n \end{bmatrix} = T_b x^0$; hence small HSVs correspond to states that are difficult to reach. The output energy resulting from an initial state x^0 and $u(t) \equiv 0$ for $t > 0$ is given by

$$\|y\|_2^2 = \int_0^\infty y(t)^T y(t)\,dt = x_0^T W_o x_0 = (\hat{x}^0)^T \hat{W}_o \hat{x}^0. = \sum_{k=1}^{n} \sigma_k \hat{x}_j^2;$$

hence large HSVs correspond to the states containing most of the energy in the system. The energy transfer from past inputs to future outputs can be computed via

$$E := \sup_{\substack{u\in\mathcal{L}_2(-\infty,0]\\ x(0)=x^0}} \frac{\|y\|_2^2}{\int_{-\infty}^{0} u(t)^T u(t)\,dt} = \frac{(x^0)^T W_o x^0}{(x^0)^T W_c^{-1} x^0} = \frac{(\bar{x}^0)^T W_c^{\frac{1}{2}} W_o W_c^{\frac{1}{2}} \bar{x}^0}{(\bar{x}^0)^T \bar{x}^0},$$

where $\bar{x}^0 := W_c^{-\frac{1}{2}} x^0$. Thus, the HSVs $\left(\Lambda\left(W_c W_o\right)\right)^{\frac{1}{2}} = \left(\Lambda\left(W_c^{\frac{1}{2}} W_o W_c^{\frac{1}{2}}\right)\right)^{\frac{1}{2}}$ measure how much the states are involved in the energy transfer from inputs to outputs.

In summary, it seems reasonable to obtain a reduced-order model by removing the least controllable states, keeping the states containing the major part of the system energy as these are the ones which are most involved in the energy transfer from inputs to outputs—that is, keeping the states corresponding to the largest HSVs. This is exactly the idea of balanced truncation, to be outlined in Section 4.2.

3 Spectral Projection Methods

In this section we will give the necessary background on spectral projection methods and the related computational tools leading to easy-to-implement and easy-to-parallelize iterative methods. These iterative methods will form the backbone of all the model reduction methods discussed in the next section.

3.1 Spectral Projectors

First, we give some fundamental definitions and properties of projection matrices.

Definition 3.1 *A matrix $P \in \mathbb{R}^{n\times n}$ is a* projector (onto a subspace $\mathcal{S} \subset \mathbb{R}^n$) *if* $\operatorname{range}(P) = \mathcal{S}$ *and* $P^2 = P$.

Definition 3.2 *Let $Z \in \mathbb{R}^{n\times n}$ with $\Lambda(Z) = \Lambda_1 \cup \Lambda_2$, $\Lambda_1 \cap \Lambda_2 = \emptyset$, and let $\mathcal{S}_1$ be the (right) Z-invariant subspace corresponding to Λ_1. Then a projector onto $\mathcal{S}_1$ is called a* spectral projector.

From this definition we obtain the following properties of spectral projectors.

Lemma 3.3 *Let $Z \in \mathbb{R}^{n\times n}$ be as in Definition 3.2, and let $P \in \mathbb{R}^{n\times n}$ be a spectral projector onto the right Z-invariant subspace corresponding to Λ_1. Then*

a) $\operatorname{rank}(P) = |\Lambda_1| =: k$,

b) $\operatorname{range}(P) = \operatorname{range}(ZP)$,

c) $\ker(P) = \operatorname{range}(I - P)$, $\operatorname{range}(P) = \ker(I - P)$,

d) *$I - P$ is a spectral projector onto the right Z-invariant subspace corresponding to Λ_2.*

Given a spectral projector P we can compute an orthogonal basis for the corresponding Z-invariant subspace $\mathcal{S}_1$ and a *spectral* or block decomposition of Z in the following way: let

$$P = QR\Pi, \quad R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \diagdown & \square \\ & \end{bmatrix}, \qquad R_{11} \in \mathbb{R}^{k\times k},$$

be a QR decomposition with column pivoting (or a rank-revealing QR decomposition (RRQR)) [GV96] where Π is a permutation matrix. Then the first k columns of Q form an orthonormal basis for $\mathcal{S}_1$ and we can transform Z to *block-triangular form*

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

where $\Lambda(Z_{11}) = \Lambda_1$, $\Lambda(Z_{22}) = \Lambda_2$.

The block decomposition given in (12) will prove very useful in what follows.

3.2 The Sign Function Method

Consider a matrix $Z \in \mathbb{R}^{n\times n}$ with no eigenvalues on the imaginary axis, that is, $\Lambda(Z) \cap \jmath\mathbb{R} = \emptyset$, 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 stable and unstable parts of $\Lambda(Z)$. The *matrix sign function* of Z is defined as $\operatorname{sign}(Z) := S \begin{bmatrix} -I_k & 0 \\ 0 & I_{n-k} \end{bmatrix} S^{-1}$. Note that $\operatorname{sign}(Z)$ is unique and independent of the order of the eigenvalues in the Jordan decomposition of Z, see, e.g., [LR95]. Many other definitions of the sign function can be given; see [KL95] for an overview. Some important properties of the matrix sign function are summarized in the following lemma.

Lemma 3.4 *Let $Z \in \mathbb{R}^{n\times n}$ with $\Lambda(Z) \cap \jmath\mathbb{R} = \emptyset$. Then:*

a) $(\operatorname{sign}(Z))^2 = I_n$, *i.e.,* $\operatorname{sign}(Z)$ *is a square root of the identity matrix;*

b) $\operatorname{sign}\left(T^{-1}ZT\right) = T^{-1}\operatorname{sign}(Z)\,T$ *for all nonsingular* $T \in \mathbb{R}^{n\times n}$*;*

c) $\operatorname{sign}\left(Z^T\right) = \operatorname{sign}(Z)^T$.

d) Let p_+ and p_- be the numbers of eigenvalues of Z with positive and negative real part, respectively. Then

$$p_+ = \frac{1}{2}(n + \operatorname{tr}(\operatorname{sign}(Z))), \quad p_- = \frac{1}{2}(n - \operatorname{tr}(\operatorname{sign}(Z))).$$

(Here, $\operatorname{tr}(M)$ denotes the trace of the matrix M.)

e) Let Z be stable, then

$$\operatorname{sign}(Z) = -I_n, \quad \operatorname{sign}(-Z) = I_n.$$

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:

$$Z_0 \leftarrow Z, \quad Z_{j+1} \leftarrow \frac{1}{2}(Z_j + Z_j^{-1}), \quad j = 0, 1, 2, \ldots. \tag{13}$$

Under the given assumptions, the sequence $\{Z_j\}_{j=0}^{\infty}$ converges with an ultimately quadratic convergence rate and

$$\operatorname{sign}(Z) = \lim_{j\to\infty} Z_j;$$

see [Rob80]. As the initial convergence may be slow, the use of acceleration techniques is recommended. There are several acceleration schemes proposed in the literature, a thorough discussion can be found in [KL92], and a survey and comparison of different schemes is given in [BD93]. For accelerating (13), in each step Z_j is replaced by $\frac{1}{\gamma_j}Z_j$, where the most prominent choices for γ_j are briefly discussed in the sequel.

***Determinantal scaling* [Bye87]:** here,

$$\gamma_j = |\det(Z_j)|^{\frac{1}{n}}.$$

This choice minimizes the distance of the geometric mean of the eigenvalues of Z_j from 1. Note that the determinant $\det(Z_j)$ is a by-product of the computations required to implement (13).

***Norm scaling* [Hig86]:** here

$$c_j = \sqrt{\frac{\|Z_j\|_2}{\|Z_j^{-1}\|_2}},$$

which has certain minimization properties in the context of computing polar decompositions. It is also beneficial regarding rounding errors as it equalizes the norms of the two addends in the finite-norm calculation $(\frac{1}{\gamma_j}Z_j) + (\frac{1}{\gamma_j}Z_j)^{-1}$.

Approximate norm scaling: as the spectral norm is expensive to calculate, it is suggested in [Hig86, KL92] to approximate this norm by the Frobenius norm or to use the bound (see, e.g., [GV96])

$$\|Z_j\|_2 \leq \sqrt{\|Z_j\|_1 \|Z_j\|_\infty}. \tag{14}$$

Numerical experiments and partial analytic considerations [BQQ04d] suggest that norm scaling is to be preferred in the situations most frequently encountered in the sign function-based calculations discussed in the following; see also Example 3.6 below. Moreover, the Frobenius norm approximation usually yields a better approximation than the one given by (14). As the computation of the Frobenius norm parallelizes very well, we will mostly use the Frobenius norm scaling in the algorithms based on (13).

There are also plenty of other iterative schemes for computing the sign function; many of those have good properties regarding convergence and parallelization (see [KL95] for an overview). Nevertheless, the basic Newton iteration (13) appears to yield the most robust implementation and the fastest execution times, both in serial and parallel implementations. Implementing (13) only requires computing matrix sums and inverses using LU factorization or Gauß-Jordan elimination. These operations are efficiently implemented in many software packages for serial and parallel computations; efficient parallelization of the matrix sign function has been reported, e.g., in [BDD+97, HQSW00].

Computations based on the matrix sign function can be considered as *spectral projection methods* as they usually involve

$$P_- := \frac{1}{2}(I_n - \operatorname{sign}(Z)), \tag{15}$$

which is a spectral projector onto the stable Z-invariant subspace. Also, $P_+ := (I_n + \operatorname{sign}(Z))/2$ is a spectral projector onto the Z-invariant subspace corresponding to the eigenvalues in the open right half plane. But note that P_- and P_+ are not orthogonal projectors, but skew projectors along the complementary Z-invariant subspace.

Remark 3.5 *The matrix sign function is criticized for several reasons, the most prominent one being the need to compute an explicit inverse in each step. Of course, it is undefined for matrices with purely imaginary eigenvalues and hence suffers from numerical problems*

in the presence of eigenvalues close to the imaginary axis. But numerical instabilities basically only show up if there exist eigenvalues with imaginary parts of magnitude less than the square root of the machine precision. Hence, significant problems can be expected in double precision arithmetic (as used in MATLAB*) for imaginary parts of magnitude less than* 10^{-8}*. (A thorough numerical analysis requires the condition of the stable subspace which is given by the reciprocal of the separation of stable and anti-stable invariant subspaces, though—the distance of eigenvalues to the imaginary axis is only an upper bound for the separation!) Fortunately, in the control applications considered here, poles are usually further apart from the imaginary axis. On the other hand, if we have no problems with the spectral dichotomy, then the sign function method solves a problem that is usually better conditioned than the Schur vector approach as it only separates the stable from the anti-stable subspace while the Schur vector method essentially requires to separate n subspaces from each other. For a thorough analysis of sign function-based computation of invariant subspaces, see [BD98, BHM97]. The difference in the conditioning of the Schur form and a block triangular form (as computed by the sign function) is discussed in [KMP01]. Moreover, in the applications considered here, mostly* $\mathrm{cond}\,(\mathrm{sign}\,(Z)) = 1$ *as* Z *is stable or anti-stable, hence the computation of* $\mathrm{sign}\,(Z)$ *itself is a well-conditioned problem!*

Therefore, counter to intuition, it should not be surprising that often, results computed by the sign function method are more accurate than those obtained by using Schur-type decompositions; see, e.g., [BQ99].

Example 3.6 *A typical convergence history (based on* $\|Z_j - \mathrm{sign}\,(Z)\,\|_F$*) is displayed in Figure 1, showing the fast quadratic convergence rate. Here, we computed the sign function of a dense matrix* A *coming from transforming a generalized state-space system (the* $n = 1357$ *case of the steel cooling problem described in Chapter [BS05] of this book) to standard state-space form. We compare the determinantal scaling and the Frobenius norm scaling. Here, the eigenvalue of* A *closest to* $\jmath\mathbb{R}$ *is* $\approx 6.7 \cdot 10^{-6}$ *and the eigenvalue of largest magnitude is* ≈ -5.8*. Therefore the condition of* A *is about* 10^6*. Obviously, norm scaling performs much better for this example. This is a typical behavior for problems with real spectrum. The computations were done using* MATLAB *7.0.1 on a Intel Pentium M processor at 1.4 GHz with 512 MBytes of RAM.*

3.3 Solving Linear Matrix Equations with the Sign Function Method

In 1971, Roberts [Rob80] introduced the matrix sign function and showed how to solve Sylvester and Lyapunov equations. This was re-discovered several times; see [BD75, DB76, HMW77]. We will briefly review the method for Sylvester equations and will then discuss some improvements useful for model reduction applications.

Consider the Sylvester equation

$$AX + XB + W = 0, \quad A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{m\times m}, W \in \mathbb{R}^{n\times m}, \tag{16}$$

with $\Lambda\,(A) \cap \Lambda\,(-B) = \emptyset$. The latter assumption is equivalent to (16) having a unique solution [LT85]. Let $X \in \mathbb{R}^{n\times m}$ be this unique solution. Then the straightforward calculation

$$\begin{bmatrix} I_n & 0 \\ X & I_m \end{bmatrix} \begin{bmatrix} A & 0 \\ W & -B \end{bmatrix} \begin{bmatrix} I_n & 0 \\ -X & I_m \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & -B \end{bmatrix} \tag{17}$$

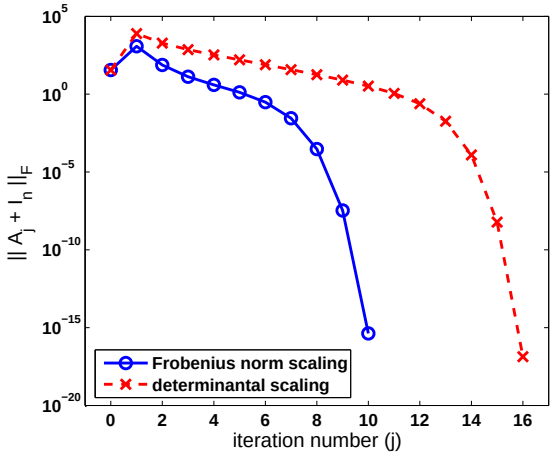

Figure 1: Example 3.6, convergence history for sign (Z) using (13).

reveals that the columns of $\begin{bmatrix} I_n \\ -X_* \end{bmatrix}$ span the invariant subspace of $Z := \begin{bmatrix} A & 0 \\ W & -B \end{bmatrix}$ corresponding to $\Lambda(A)$. In principle, this subspace, and after an appropriate change of basis, also the solution matrix X, can be computed from a spectral projector onto this Z-invariant subspace. The sign function is an appropriate tool for this whenever A, B are stable as in this case, P_- from (15) is the required spectral projector. A closer inspection of (13) applied to Z shows that we do not even have to form P_- in this case, as the solution can be directly read off the matrix sign (Z): using (17) and Lemma 3.4 reveals that

$$\operatorname{sign}(Z) = \operatorname{sign}\left(\begin{bmatrix} A & 0 \\ W & -B \end{bmatrix}\right) = \begin{bmatrix} -I_n & 0 \\ 2X & I_m \end{bmatrix}$$

so that the solution of (16) is given as the lower left block of the limit of (13), divided by 2. Moreover, the block-triangular structure of Z allows to decouple (13) as

$$\begin{aligned} &A_0 \leftarrow A, \quad B_0 \leftarrow B, \quad W_0 \leftarrow W, \\ &\texttt{for } j = 0, 1, 2, \ldots \\ &\qquad A_{j+1} \leftarrow \frac{1}{2\gamma_j}\left(A_j + \gamma_j^2 A_j^{-1}\right), \\ &\qquad B_{j+1} \leftarrow \frac{1}{2\gamma_j}\left(B_j + \gamma_j^2 B_j^{-1}\right), \\ &\qquad W_{j+1} \leftarrow \frac{1}{2\gamma_j}\left(W_j + \gamma_j^2 A_j^{-1} W_j B_j^{-1}\right). \end{aligned} \tag{18}$$

so that $X_* = \frac{1}{2}\lim_{j\to\infty} W_j$. As A, B are assumed to be stable, A_j tends to $-I_n$ and B_j tends to $-I_m$ so that we can base a stopping criterion on

$$\max\{\|A_j + I_n\|, \|B_j + I_m\|\} < \tau, \tag{19}$$

where τ is an error tolerance and $\|\,.\,\|$ is an appropriate matrix norm.

For Lyapunov equations

$$AX + XA^T + W = 0, \quad A \in \mathbb{R}^{n\times n}, W = W^T \in \mathbb{R}^{n\times n}, \tag{20}$$

we simply replace B by A^T in defining Z. Assuming again stability of A, and observing that the iteration for B_j in (18) is redundant (see also Lemma 3.4 c)), the sign function method for Lyapunov equation becomes

$$\begin{aligned} A_0 &\leftarrow A, \quad W_0 \leftarrow W, \\ &\texttt{for } j = 0, 1, 2, \ldots \\ &\quad A_{j+1} \leftarrow \frac{1}{2\gamma_j}\left(A_j + \gamma_j^2 A_j^{-1}\right), \\ &\quad W_{j+1} \leftarrow \frac{1}{2\gamma_j}\left(W_j + \gamma_j^2 A_j^{-1} W_j A_j^{-T}\right). \end{aligned} \tag{21}$$

with $X_* = \frac{1}{2}\lim_{j\to\infty} W_j$. Here, a reasonable stopping criterion is given by $\|A_j + I_n\| < \tau$, see (19).

If we consider the Lyapunov equations (4) defining the controllability and observability Gramians of stable LTI systems, we observe the following facts which will be of importance for an efficient implementation of (21) in the context of model reduction:

1. The right-hand side is given in factored form, that is, $W = BB^T$ or $W = C^TC$, and hence semidefinite. Thus, X is positive semidefinite [LT85], and can therefore also be factored as $X = SS^T$. A possibility here is a Cholesky factorization.

2. Usually, the number of states in (1) is much larger than the number of inputs and outputs, that is, $n \gg m, p$. In many cases, this yields a solution matrix with rapidly decaying eigenvalues so that its numerical rank is small; see [ASZ02, Gra04, Pen00] for partial explanations of this fact. Figure 2 demonstrates this behavior for the controllability Gramian of a random stable LTI system with $n = 500$, $m = 10$, and *stability margin* (minimum distance of $\Lambda(A)$ to $\jmath\mathbb{R}$) ≈ 0.055. Hence, if n_ε is the numerical rank of X, then there is a matrix $S_\varepsilon \in \mathbb{R}^{n\times n_\varepsilon}$ so that $X \approx S_\varepsilon S_\varepsilon^T$ at the level of machine accuracy.

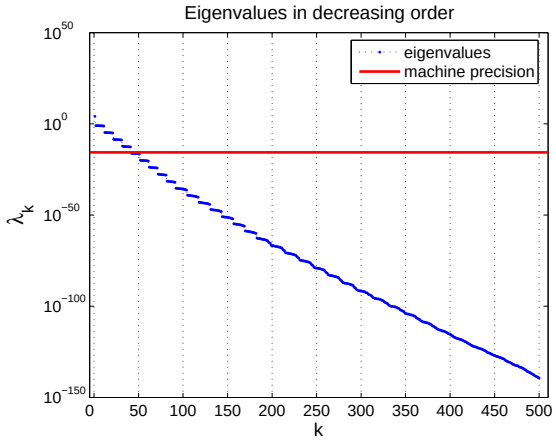

Figure 2: Eigenvalue decay rate for the controllability Gramian of a random LTI system with $n = 500$, $m = 10$, and stability margin ≈ 0.055.

The second observation also serves as the basic idea of most algorithms for large-scale Lyapunov equations; see [Pen00, AS01] as well as Chapters [GL05] and [MS05]. Storing S_ε is

much cheaper than storing X or S as instead of n^2 only $n \cdot n_\varepsilon$ real numbers need to be stored. In the example used above to illustrate the eigenvalue decay, this leads already to a reduction factor of about 10 for storing the solution of the controllability Gramian; in Example 3.6 this factor is close to 100 so that 99% of the storage is saved. We will make use of this fact in the method proposed for solving (4).

For the derivation of the proposed implementation of the sign function method for computing system Gramians, we will use the Lyapunov equation defining the observability Gramian,

$$A^T Y + Y A + C^T C = 0.$$

Re-writing the iteration for $W_j$in (21), we obtain with $W_0 = C_0^T C_0 := C^T C$:

$$W_{j+1} = \frac{1}{2\gamma_j}\left(W_j + \gamma_j^2 A_j^{-T} W_j A_j^{-1}\right) = \frac{1}{2\gamma_j}\begin{bmatrix} C_j \\ \gamma_j C_j A_j^{-1} \end{bmatrix}^T \begin{bmatrix} C_j \\ \gamma_j C_j A_j^{-1} \end{bmatrix}.$$

Thus, in order to compute a factor R of $Y = R^T R$ we can instead directly iterate on the factors:

$$C_0 \leftarrow C, \qquad C_{j+1} \leftarrow \frac{1}{\sqrt{2\gamma_j}}\begin{bmatrix} C_j \\ \gamma_j C_j A_j^{-1} \end{bmatrix}. \tag{22}$$

A problem with this iteration is that the number of columns in C_j doubles in each iteration step so that after $j \geq \log_2 \frac{n}{p}$ steps, the required workspace for C_j becomes even larger than n^2. There are several ways to limit this workspace. The first one, initially suggested in [LA93], works with an $n \times n$-matrix, sets C_0 to the Cholesky factor of $C^T C$, computes a QR factorization of $\begin{bmatrix} C_j \\ \gamma_j C_j A_j^{-1} \end{bmatrix}$ in each iteration, and uses its R-factor as next C_j-iterate. A slightly cheaper version of this is given in [BQ99], where (22) is used as long as $j \leq \log_2 \frac{n}{p}$ and only then starts computing QR factorizations in each step. In both cases, it can be shown that $\lim_{j\to\infty} C_j$ is a Cholesky factor of the solution Y of (20).

In order to exploit the second observation from above, in [BQ99] it is suggested to keep the number of rows in C_j less than or equal to the (numerical) rank of Y by computing in each iteration step a rank-revealing QR factorization

$$\frac{1}{\sqrt{2\gamma_j}}\begin{bmatrix} C_j \\ \gamma_j C_j A_j^{-1} \end{bmatrix} = U_{j+1}\begin{bmatrix} R_{j+1} & T_{j+1} \\ 0 & S_{j+1} \end{bmatrix}\Pi_{j+1}, \tag{23}$$

where $R_{j+1} \in \mathbb{R}^{p_{j+1}\times p_{j+1}}$ is nonsingular, $p_{j+1} = \operatorname{rank}\left(\begin{bmatrix} C_j \\ \gamma_j C_j A_j^{-1} \end{bmatrix}\right)$, and $\|S_{j+1}\|_2$ is "small enough" (with respect to a given tolerance threshold for determining the numerical rank) to safely set $S_{j+1} = 0$. Then, the next iterate becomes

$$C_{j+1} \leftarrow [\, R_{j+1}\; T_{j+1}\,]\Pi_{j+1}, \tag{24}$$

and $\frac{1}{\sqrt{2}}\lim_{j\to\infty} C_j$ is a (numerical) full-rank factor of the solution Y of (20).

The criterion that will be used to select the tolerance threshold for $\|S_{j+1}\|_2$ is based on the following considerations. Let

$$M = \begin{bmatrix} M_1 & M_2 \\ E_1 & E_2 \end{bmatrix}, \quad \tilde{M} = \begin{bmatrix} M_1 & M_2 \end{bmatrix}$$

so that M^TM and $\tilde{M}^T\tilde{M}$ are approximations to a positive semidefinite matrix $K \in \mathbb{R}^{n\times n}$. Assume

$$\|E_j\|_2 \leq \sqrt{\varepsilon}\|M\|_2, \quad j = 1, 2,$$

for some $0 < \varepsilon < 1$. Then

$$\begin{aligned} K - M^TM &= K - \begin{bmatrix} M_1^T & E_1^T \\ M_2^T & E_2^T \end{bmatrix} \begin{bmatrix} M_1 & M_2 \\ E_1 & E_2 \end{bmatrix} \\ &= K - \tilde{M}^T\tilde{M} - \begin{bmatrix} E_1^TE_1 & E_1^TE_2 \\ E_2^TE_1 & E_2^TE_2 \end{bmatrix} \end{aligned}$$

If M is a reasonable approximation with $\|M\|_2^2 \approx \|K\|_2$, then the relative error of the two approximations satisfies

$$\frac{\|K - \tilde{M}^T\tilde{M}\|_2}{\|K\|_2} \lessapprox \frac{\|K - M^TM\|_2}{\|K\|_2} + \mathcal{O}(\varepsilon). \tag{25}$$

If $\varepsilon \sim \mathbf{u}$, where $\mathbf{u}$ is the machine precision, this shows that neglecting the blocks E_1, E_2 in the factor of the approximation to K yields a relative error of size $\mathcal{O}(\mathbf{u})$ which is negligible in the presence of roundoff errors. Therefore, in our calculations we choose the numerical rank with respect to $\varepsilon = \sqrt{\mathbf{u}}$.

Example 3.7 *For the same random LTI system as used in the illustration of the eigenvalue decay in Figure 2, we computed a numerical full-rank factor of the controllability Gramian. The computed rank is 31, and* 10 *iterations are needed to achieve convergence in the sign function based iteration. Figure 3 shows the development of* $p_j = \operatorname{rank}(C_j)$ *during the iteration. Comparing (24) with the currently best available implementation of Hammarling's*

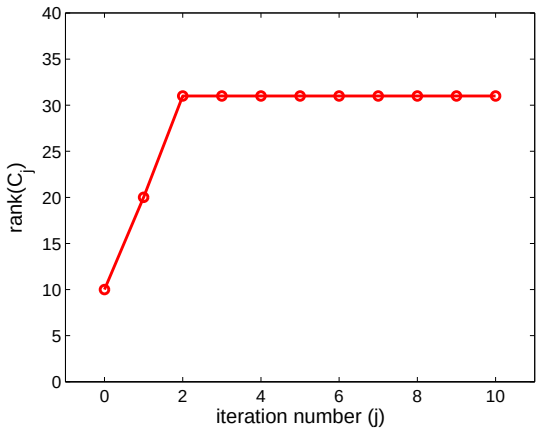

Figure 3: Example 3.7, number of columns in C_j in the full-rank iteration composed of (22), (23), and (24).

method [Ham82] for computing the Cholesky factor of the solution of a Lyapunov equation, contained in the SLICOT library [BMS+99], we note that the sign function-based method (pure MATLAB *code) required 4.69 sec. while the SLICOT function (compiled and optimized*

Algorithm 1 Coupled Newton Iteration for Dual Lyapunov Equations.

INPUT: Realization $(A, B, C) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times m} \times \mathbb{R}^{p\times n}$ of an LTI system, tolerances τ_1 for convergence of (21) and τ_2 for rank detection.

OUTPUT: Numerical full-rank factors of the controllability and observability Gramians of the LTI system such that $W_c = S^T S$, $W_o = R^T R$.

1: **while** $\|A + I_n\|_1 > \tau_1$ **do**

2: Use the LU decomposition or the Gauß-Jordan elimination to compute A^{-1}.

3: Set $\gamma := \sqrt{\frac{\|A\|_F}{\|A^{-1}\|_F}}$ and $Z := \gamma A^{-1}$.

4: Compute a rank-revealing LQ factorization

$$\frac{1}{\sqrt{2\gamma}} \begin{bmatrix} B & ZB \end{bmatrix} =: \Pi \begin{bmatrix} L & 0 \\ T & S \end{bmatrix} Q$$

with $\|S\|_2 \leq \tau_2 \|\frac{1}{\sqrt{2\gamma}} \begin{bmatrix} B & ZB \end{bmatrix}\|_2$.

5: Set $B := \Pi \begin{bmatrix} R \\ T \end{bmatrix}$.

6: Compute a rank-revealing QR factorization

$$\frac{1}{\sqrt{2\gamma}} \begin{bmatrix} C \\ CZ \end{bmatrix} =: Q \begin{bmatrix} R & T \\ 0 & S \end{bmatrix} \Pi$$

with $\|S\|_2 \leq \tau_2 \|\frac{1}{\sqrt{2\gamma}} \begin{bmatrix} C \\ CZ \end{bmatrix}\|_2$.

7: Set $C := \Pi \begin{bmatrix} R & T \end{bmatrix}$.

8: Set $A := \frac{1}{2}(\frac{1}{\gamma} A + Z)$.

9: **end while**

10: Set $S := B^T$, $R := C$.

Fortran 77 code, called via a mex file from MATLAB*) needed 7.75 sec., both computed using* MATLAB *7.0.1 on a Intel Pentium M processor at 1.4 GHz with 512 MBytes of RAM. The computed relative residuals*

$$\frac{\|AX + XA^T + BB^T\|_F}{2\|A\|_F\|X\|_F + \|BB^T\|_F}$$

are comparable, $4.6 \cdot 10^{-17}$ *for the sign function method and* $3.1 \cdot 10^{-17}$ *for Hammarling's method.*

It is already observed in [LL96] that the two sign function iterations needed to solve both equations in (4) can be coupled as they contain essentially the same iteration for the A_j-matrices (the iterates are transposes of each other), hence only one of them is needed. This was generalized and combined with the full-rank iteration (24) in [BCQ98, BQQ00a]. The resulting sign function-based "spectral projection method" for computing (numerical) full-rank factors of the controllability and observability Gramians of the LTI system (1) is summarized in Algorithm 1.

Algorithm 2 Sign Function-based Spectral Projection Method for Block-Diagonalization.

INPUT: $Z \in \mathbb{R}^{n\times n}$ with $\Lambda(Z) \cap \jmath\mathbb{R} = \emptyset$.
OUTPUT: $U \in \mathbb{R}^{n\times n}$ nonsingular such that

$$U^{-1}ZU = \begin{bmatrix} Z_{11} & \\ & Z_{22} \end{bmatrix}, \quad \Lambda(Z_{11}) = \Lambda(Z) \cap \mathbb{C}^-, \ \Lambda(Z_{22}) = \Lambda(Z) \cap \mathbb{C}^+.$$

1: Compute $\operatorname{sign}(Z)$ using (13).
2: Compute a rank-revealing QR factorization

$$I_n - \operatorname{sign}(Z) =: UR\Pi.$$

3: Block-triangularize A as in (12); that is, set

$$Z := U^T Z U =: \begin{bmatrix} Z_{11} & Z_{12} \\ 0 & Z_{22} \end{bmatrix}.$$

4: Solve the Sylvester equation $Z_{11}Y - YZ_{22} + Z_{12} = 0$ using (18).
{Note: $Z_{11}, -Z_{22}$ are stable!}
5: Set

$$Z := \begin{bmatrix} I_k & -Y \\ 0 & I_{n-k} \end{bmatrix} \begin{bmatrix} Z_{11} & Z_{12} \\ 0 & Z_{22} \end{bmatrix} \begin{bmatrix} I_k & Y \\ 0 & I_{n-k} \end{bmatrix} = \begin{bmatrix} Z_{11} & \\ & Z_{22} \end{bmatrix}, \quad U := U \begin{bmatrix} I_k & Y \\ 0 & I_{n-k} \end{bmatrix}.$$

3.4 Block-Diagonalization

In the last section we used the block-diagonalization properties of the sign function method to derive an algorithm for solving linear matrix equations. This feature will also turn out to be useful for other problems such as modal truncation and model reduction of unstable systems. The important equation in this context is (17), which allows us to eliminate the off-diagonal block of a block-triangular matrix by solving a Sylvester equation.

A spectral projection method for the block-diagonalization of a matrix Z having no eigenvalues on the imaginary axis is summarized in Algorithm 2. In case of purely imaginary eigenvalues, it can still be used if applied to $Z + \alpha I_n$, where $\alpha \in \mathbb{R}$ is an appropriate spectral shift which is not the real part of an eigenvalue of Z. Note that the computed transformation matrix is not orthogonal, but its first n columns are orthonormal.

4 Model Reduction Using Spectral Projection Methods

4.1 Modal Truncation

Modal truncation is probably one of the oldest model reduction techniques [Dav66, Mar66]. In some engineering disciplines, modified versions are still in use, mainly in structural dynamics. In particular, the model reduction method in [CB68] and its relatives, called nowadays *substructuring methods*, which combine the modal analysis with a static compensation following Guyan [Guy68], are frequently used. We will not elaborate on these type of methods, but

will only focus on the basic principles of modal truncation and how it can be implemented using spectral projection ideas.

The basic idea of modal truncation is to project the dynamics of the LTI system (1) onto an A-invariant subspace corresponding to the dominant modes of the system (poles of $G(s)$, eigenvalues of A that are not canceled by zeros). In structural dynamics software as ANSYS [ANS] or Nastran [MSC], usually an eigenvector basis of the chosen modal subspace is used. Employing the block-diagonalization abilities of the sign function method described in Subsection 3.4, it is easy to derive a spectral projection method for modal truncation. This was first observed by Roberts in his original paper on the matrix sign function [Rob80]. It has the advantage that we avoid a possible ill-conditioning in the eigenvector basis.

An obvious, though certainly not always optimal, choice of dominant modes is to select those eigenvalues of A having nonnegative or small negative real parts. Basically, these eigenvalues dominate the long-term dynamics of the solution of the linear ordinary differential equation describing the dynamics of (1)—solution components corresponding to large negative real parts decay rapidly and mostly play a less important (negligible) role in vibration analysis or control design. This viewpoint is rather naive as it neither takes into account the transient behavior of the dynamical system nor the oscillations caused by large imaginary parts or the sensitivity of the eigenvalues with respect to small perturbations. Nevertheless, this approach is often successful when A comes from an FEM analysis of an elliptic operator such as those arising in linear elasticity or heat transfer processes.

An advantage of modal truncation is that the poles of the reduced-order system are also poles of the original system. This is important in applications such as vibration analysis since the modes correspond to the resonance frequencies of the original system; the most important resonances are thus retained in the reduced-order model.

In the sequel we will use the naive mode selection criterion described above in order to derive a simple implementation of modal truncation employing a spectral projector. The approach, essentially already contained in the original work by Roberts [Rob80], is based on selecting a *stability margin* $\alpha > 0$, which determines the maximum modulus of the real parts of the modes to be preserved in the reduced-order model. Now, the eigenvalues of $A+\alpha I_n$ are the eigenvalues of A, shifted by α to the right. That is, all eigenvalues with stability margin less than α become unstable eigenvalues of $A + \alpha I_n$. Then, applying the sign function to $A + \alpha I_n$ yields the spectral projector $\frac{1}{2}(I_n + \operatorname{sign}(A + \alpha I_n))$ onto the unstable invariant subspace of $A+\alpha I_n$ which equals the A-invariant subspace corresponding to the modes that are dominant with respect to the given stability margin. Block-triangularization of A using (12), followed by block-diagonalization based on (17) give rise to the modal truncation implementation outlined in Algorithm 3. In principle, Algorithm 2 could also be used here, but the variant in Algorithm 3 is adapted to the needs of modal truncation and slightly cheaper.

The error of modal truncation can easily be quantified. It follows immediately that

$$G(s) - \hat{G}(s) = C_2(sI - A_{22})^{-1}B_2;$$

see also (42) below or [GL95, Lemma 9.2.1]. As A_{22}, B_2, C_2 are readily available, the $\mathcal{L}_2$-error for the outputs or $\mathcal{H}_\infty$-error for the transfer function (see (10)) is computable. For diagonalizable A_{22}, we obtain the upper bound

$$\|G - \hat{G}\|_\infty \leq \operatorname{cond}_2(T)\,\|C_2\|_2\|B_2\|_2 \frac{1}{\min_{\lambda\in\Lambda(A_{22})}|\operatorname{Re}(\lambda)|}, \tag{26}$$

Algorithm 3 Spectral Projection Method for Modal Truncation.

INPUT: Realization $(A, B, C, D) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times m} \times \mathbb{R}^{p\times n} \times \mathbb{R}^{p\times m}$ of an LTI system (1); a stability margin $\alpha > 0$, $\alpha \neq \mathrm{Re}(\lambda)$ for all $\lambda \in \Lambda(A)$.

OUTPUT: Realization $(\hat{A}, \hat{B}, \hat{C}, \hat{D})$ of a reduced-order model.

1: Compute $S := \mathrm{sign}(A + \alpha I_n)$.

2: Compute a rank-revealing QR factorization $S =: QR\Pi$.

3: Compute (see (12))

$$Q^T A Q =: \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \quad Q^T B =: \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}, \quad CQ =: \begin{bmatrix} C_1 \\ C_2 \end{bmatrix}.$$

4: Solve the Sylvester equation $(A_{11} - \beta I_k)Y - Y(A_{22} - \beta I_k) + A_{12} = 0$ using (18). {Note: If A is stable, $\beta = 0$ can be chosen; otherwise set

$$\beta \geq \max_{\lambda \in \Lambda(A_{11}) \cap \mathbb{C}^+} (\mathrm{Re}(\lambda)),$$

e.g., $\beta = 2\|A_{11}\|_F$.}

5: The reduced-order model is then

$$\hat{A} := A_{11}, \quad \hat{B} := B_1 - Y B_2, \quad \hat{C} := C_1, \quad \hat{D} := D.$$

---

where $T^{-1} A_{22} T = D$ is the spectral decomposition of A_{22} and $\mathrm{cond}_2(T)$ is the spectral norm condition number of its eigenvector matrix T.

As mentioned at the beginning of this section, several extensions and modifications of modal truncation are possible. In particular, static compensation can account for the steady-state error inherent in the reduced-order model; see, e.g., [Föl94] for an elaborate variant. This is related to singular perturbation approximation; see also subsection 4.3 below.

4.2 Balanced Truncation

The basic idea of balanced truncation is to compute a balanced realization

$$(TAT^{-1}, TB, CT^{-1}, D) = \left(\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}, \begin{bmatrix} C_1 & C_2 \end{bmatrix}, D \right), \tag{27}$$

where $A_{11} \in \mathbb{R}^{r\times r}$, $B_1 \in \mathbb{R}^{r\times m}$, $C_1 \in \mathbb{R}^{p\times r}$, with r less than the McMillan degree $\hat{n}$ of the system, and then to use as the reduced-order model the truncated realization

$$(\hat{A}, \hat{B}, \hat{C}, \hat{D}) = (A_{11}, B_1, C_1, D). \tag{28}$$

This idea dates essentially back to [Moo81, MR76]. Collecting results from [Moo81, Glo84, TP87], the following result summarizes the properties of balanced truncation.

Proposition 4.1 *Let (A, B, C, D) be a realization of a stable LTI system with McMillan degree $\hat{n}$ and transfer function $G(s)$ and let $(\hat{A}, \hat{B}, \hat{C}, \hat{D})$ with associated transfer function $\hat{G}$ be computed as in (27)–(28). Then the following holds:*

a) The reduced-order system $\hat{G}$ is balanced, minimal, and stable. Its Gramians are

$$\hat{P} = \hat{Q} = \hat{\Sigma} = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix}.$$

b) The absolute error bound

$$\|G - \hat{G}\|_\infty \leq 2 \sum_{k=r+1}^{\hat{n}} \sigma_k. \tag{29}$$

holds.

c) If $r = \hat{n}$, then (28) is a minimal realization of G and $G = \hat{G}$.

Of particular importance is the error bound (29) as it allows an adaptive choice of the order of the reduced-order model based on a prescribed tolerance threshold for the approximation quality. (The error bound (29) can be improved in the presence of Hankel singular values with multiplicity greater than one—they need to appear only once in the sum on the right-hand side.)

It is easy to check that for a controllable and observable (minimal) system, i.e., a system with nonsingular Gramians, the matrix

$$T = \Sigma^{\frac{1}{2}} U^T R^{-T} \tag{30}$$

provides a balancing state-space transformation. Here $W_c = R^T R$ and $R W_o R^T = U \Sigma^2 U^T$ is a singular value decomposition. A nice observation in [LHPW87, TP87] allows us to compute (28) also for non-minimal systems without the need to compute the full matrix T. The first part of this observation is that for $W_o = S^T S$,

$$S^{-T}(W_c W_o) S^T = (SR^T)(SR^T)^T = (U \Sigma V^T)(V \Sigma U^T) = U \Sigma^2 U^T$$

so that U, Σ can be computed from an SVD of SR^T,

$$SR^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix}, \qquad \Sigma_1 = \operatorname{diag}(\sigma_1, \ldots, \sigma_r). \tag{31}$$

The second part needed is the fact that computing

$$T_l = \Sigma_1^{-1/2} V_1^T R, \quad T_r = S^T U_1 \Sigma_1^{-1/2}, \tag{32}$$

and

$$\hat{A} := T_l A T_r, \quad \hat{B} := T_l B, \quad \hat{C} := C T_r \tag{33}$$

is equivalent to first computing a minimal realization of (1), then balancing the system as in (27) with T as in (30), and finally truncating the balanced realization as in (28). In particular, the realizations obtained in (28) and (33) are the same, T_l contains the first r rows of T and T_r the first r columns of T^{-1}—those parts of T needed to compute A_{11}, B_1, C_1 in (27). Also note that the product $T_r T_l$ is a projector onto an r-dimensional subspace of the state-space and model reduction via (33) can therefore be seen as projecting the dynamics of the system onto this subspace.

Algorithm 4 Spectral Projection Method for Balanced Truncation.

INPUT: Realization $(A, B, C, D) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times m} \times \mathbb{R}^{p\times n} \times \mathbb{R}^{p\times m}$ of an LTI system (1); a tolerance τ for the absolute approximation error or the order r of the reduced-order model.

OUTPUT: Stable reduced-order model, error bound δ.

1: Compute full-rank factors S,R of the system Gramians using Algorithm 1.

2: Compute the SVD

$$SR^T =: \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & \\ & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix},$$

such that $\Sigma_1 \in \mathbb{R}^{r\times r}$ is diagonal with the r largest Hankel singular values in decreasing order on its diagonal. Here r is either the fixed order provided on input or chosen as minimal integer such that $2\sum_{j=r+1}^{\hat{n}} \sigma_j \leq \tau$.

3: Set $T_l := \Sigma_1^{-1/2} V_1^T R$, $T_r = S^T U_1 \Sigma_1^{-1/2}$.

4: Compute the reduced-order model,

$$\hat{A} := T_l A T_r, \quad \hat{B} := T_l B, \quad \hat{C} := C T_r, \quad \hat{D} := D,$$

and the error bound $\delta := 2\sum_{j=r+1}^{\hat{n}} \sigma_j$.

The algorithm resulting from (33) is often referred to as the *SR method* for balanced truncation. In [LHPW87, TP87] and all textbooks treating balanced truncation, S and R are assumed to be the (square, triangular) Cholesky factors of the system Gramians. In [BQQ00a] it is shown that everything derived so far remains true if full-rank factors of the system Gramians are used instead of Cholesky factors. This yields a much more efficient implementation of balanced truncation whenever $\hat{n} \ll n$ (numerically). Low numerical rank of the Gramians usually signifies a rapid decay of their eigenvalues, as shown in Figure 3, and implies a rapid decay of the Hankel singular values. The resulting algorithm, derived in [BQQ00a], is summarized in Algorithm 4.

It is often stated that balanced truncation is not suitable for large-scale problems as it requires the solution of two Lyapunov equations, followed by an SVD, and that both steps require $\mathcal{O}(n^2)$ storage and $\mathcal{O}(n^3)$ flops. This is not true for Algorithm 4 although it does not completely break the $\mathcal{O}(n^2)$ storage and $\mathcal{O}(n^3)$ flops barriers. In Subsection 5.2 it will be shown that by reducing the complexity of the first stage of Algorithm 4 down to $\mathcal{O}(n \cdot q(\log n))$, where q is a quadratic or cubic polynomial, it is possible to break this curse of dimensionality for certain problem classes.

An analysis of Algorithm 4 reveals the following: assume that A is a full matrix with no further structure to be exploited, and define

$$n_{co} := \max\{\operatorname{rank}(S), \operatorname{rank}(R)\} \ll n,$$

where by abuse of notation "rank" denotes the numerical rank of the factors of the Gramians. Then the storage requirements and computational cost are as follows:

1. The solution of the dual Lyapunov equations splits into three separate iterations:

   (a) The iteration for A_j requires the inversion of a full matrix and thus needs $\mathcal{O}(n^2)$ storage and $\mathcal{O}(n^3)$ flops.

(b) The iterations for B_j and C_j need an additional $\mathcal{O}(n \cdot n_{co})$ storage, all computations can be performed in $\mathcal{O}(n^2 n_{co})$ flops. The n^2 part in the complexity comes from applying A_j^{-1} using either forward and backward substitution or matrix multiplication—if this can be achieved in a cheaper way as in Subsection 5.2, the complexity reduces to $\mathcal{O}(n \cdot n_{co}^2)$.

2. Computing the SVD of SR^T only needs $\mathcal{O}(n_{co}^2)$ workspace and $\mathcal{O}(n \cdot n_{co})$ flops and therefore does not contribute significantly to the cost of the algorithm.

3. The computation of the ROM via (32) and (33) requires $\mathcal{O}(r^2)$ additional workspace and $\mathcal{O}(nn_{co}r + n^2 r)$ flops where the n^2 part corresponds to the cost of matrix-vector multiplication with A and is not present if this is cheaper than the usual $2n^2$ flops.

An even more detailed analysis shows that the implementation of the SR method of balanced truncation outlined in Algorithm 4 can be significantly faster than the one using Hammarling's method for computing Cholesky factors of the Gramians as used in SLICOT [BMS+99, Var01] and MATLAB; see [BQQ00a]. It is important to remember that if A has a structure that allows to store A in less than $\mathcal{O}(n^2)$, to solve linear systems in less than $\mathcal{O}(n^3)$ and to do matrix-vector multiplication in less than $\mathcal{O}(n^2)$, the complexity of Algorithm 4 is *less than* $\mathcal{O}(n^2)$ *in storage and* $\mathcal{O}(n^3)$ *in computing time!*

If the original system is highly unbalanced (and hence, the state-space transformation matrix T in (27) is ill-conditioned), the *balancing-free square-root* (BFSR) balanced truncation algorithm suggested in [Var91] may provide a more accurate reduced-order model in the presence of rounding errors. It combines the SR implementation from [LHPW87, TP87] with the balancing-free model reduction approach in [SC89]. The BFSR algorithm only differs from the SR implementation in the procedure to obtain T_l and T_r from the SVD (31) of SR^T, and in that the reduced-order model is not balanced. The main idea is that in order to compute the reduced-order model it is sufficient to use orthogonal bases for range (T_l) and range (T_r). These can be obtained from the following two QR factorizations:

$$S^T U_1 = [P_1\ P_2] \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix}, \qquad R^T V_1 = [Q_1\ Q_2] \begin{bmatrix} \bar{R} \\ 0 \end{bmatrix}, \tag{34}$$

where P_1, $Q_1 \in \mathbb{R}^{n \times r}$ have orthonormal columns, and $\hat{R}$, $\bar{R} \in \mathbb{R}^{r \times r}$ are upper triangular. The reduced-order system is then given by (33) with

$$T_l = (Q_1^T P_1)^{-1} Q_1^T, \qquad T_r = P_1, \tag{35}$$

where the $(Q_1^T P_1)^{-1}$ factor is needed to preserve the projector property of $T_r T_l$.

The absolute error of a realization of order r computed by the BFSR implementation of balanced truncation satisfies the same upper bound (29) as the reduced-order model computed by the SR version.

4.2.1 Numerical Experiments

We compare modal truncation, implemented as MATLAB function `modaltrunc` following Algorithm 3 and balanced truncation, implemented as MATLAB function `btsr` following Algorithm 4 for some of the benchmark examples presented in Part II of this book. The MATLAB codes are available from

`http://www.tu-chemnitz.de/~benner/software.php`

In the comparison we included several MATLAB implementations of balanced truncation based on using the Bartels-Stewart or Hammarling's method for computing the system Gramians:

- the SLICOT [BMS+99] implementation of balanced truncation, called via a mex-function from the MATLAB function `bta` [Var01],
- the MATLAB Control Toolbox (Version 6.1 (R14SP1)) function `balreal` followed by `modred`,
- the MATLAB Robust Control Toolbox (Version 3.0 (R14SP1)) function `balmr`.

The examples that we chose to compare the methods are:

EX-RAND This is Example 3.7 from above.

RAIL1357 This is the steel cooling example described in Chapter [BS05]. Here, we chose the smallest of the provided test sets with $n = 1357$.

FILTER2D This is the optical tunable filter example described in Chapter [HBZ05]. For the comparison, we chose the 2D problem — the 3D problem is well beyond the scope of the discussed implementations of modal or balanced truncation.

ISS-II This is a model of the extended service module of the International Space Station, for details see Chapter [CV05].

(For a more complete comparison of balanced truncation based on Algorithm 4 and the SLICOT model reduction routines see [BQQ03b].)

The frequency response errors for the chosen examples are shown in Figure 4. For the implementations of balanced truncation, we only plotted the error curve for `btsr` as the graphs produced by the other implementations are not distinguishable with the exception of FILTER2D where the Robust Control Toolbox function yields a somewhat bigger error for high frequencies (still satisfying the error bound (29)). Note that the frequency response error here is measured as the pointwise absolute error

$$\|G(j\omega) - \hat{G}(j\omega)\|_2 = \sigma_{\max}\left(G(j\omega) - \hat{G}(j\omega)\right),$$

where $\|.\|_2$ is the spectral norm (matrix 2-norm).

From Figure 4 it is obvious that for equal order of the reduced-order model, modal truncation usually gives a much worse approximation than balanced truncation. Note that the order r of the reduced-order models was selected based on the reduced-order model computed via Algorithm 3 for a specific, problem-dependent stability margin α. We chose $\alpha = 244$ for EX-RAND, $\alpha = 0.01$ for RAIL1357, $\alpha = 5 \cdot 10^3$ for FILTER2D, and $\alpha = 0.005$ for ISS-II. That is, the reduced-order models computed by balanced truncation used a fixed order rather than an adaptive selection of the order based on (29).

The computation times obtained using MATLAB 7.0.1 on a Intel Pentium M processor at 1.4 GHz with 512 MBytes of RAM are given in Table 1.

Some peculiarities we found in the results:

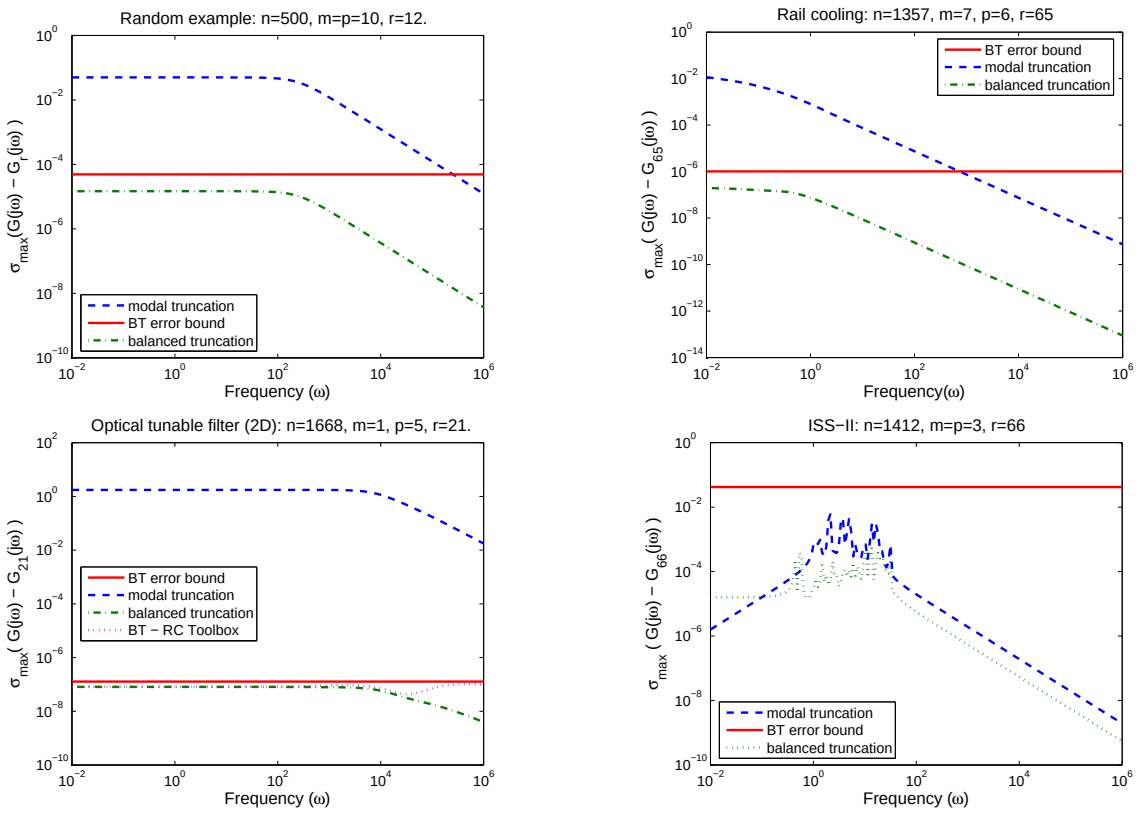

Figure 4: Frequency response (pointwise absolute) error for the Examples EX-RAND, RAIL1357, FILTER2D, ISS-II.

- the error bound (29) for EX-RAND as computed by the Robust Control Toolbox function is $2.2 \cdot 10^{-2}$; this compares unfavorably to the correct bound $4.9 \cdot 10^{-5}$, returned correctly by the other implementations of balanced truncation. Similarly, for FILTER2D, the Robust Control Toolbox function computes an error bound $10,000$ times larger than the other routines and the actual error. This suggests that the smaller Hankel singular values computed by `balmr` are very incorrect.

- The behavior for the first 3 examples regarding computing time is very much consistent while the ISS-II example differs significantly. The reason is that the sign function does converge very slowly for this particular example and the full-rank factorization computed reveals a very high numerical rank of the Gramians (roughly $n/2$). This results in fairly expensive QR factorizations at later stages of the iteration in Algorithm 1.

Altogether, spectral projection-based balanced truncation is a viable alternative to other balanced truncation implementations in MATLAB. If the Gramians have low numerical rank, the execution times are generally much smaller than for approaches based on solving the Lyapunov equations (4) employing Hammarling's method. On the other hand, Algorithm 4 suffers much from a high numerical rank of the Gramians due to high execution times of Algorithm 1 in that case. The accuracy of all implementations is basically the same for all investigated examples—an observation in accordance to the tests reported in [BQQ00a, BQQ03b]. Moreover, the efficiency of Algorithm 4 allows an easy and highly scalable parallel

Example	Modal Trunc.	Balanced Truncation			
	Alg. 3	Alg. 4	SLICOT	`balreal/modred`	`balmr`
EX-RAND	11.36	5.35	14.24	21.44	34.78
RAIL1357	203.80	101.78	241.44	370.56	633.25
FILTER2D	353.27	152.85	567.46	351.26	953.54
ISS-II	399.65	1402.13	247.21	683.72	421.69

Table 1: CPU times needed in the comparison of modal truncation and different balanced truncation implementations for the chosen examples.

implementation in contrast to versions based on Hammarling's method, see Subsection 5.1. Thus, much larger problems can be tackled using a spectral projection-based approach.

4.3 Balancing-Related Methods

4.3.1 Singular Perturbation Approximation

In some situations, a reduced-order model with perfect matching of the transfer function at $s = 0$ is desired. In technical terms, this means that the DC gain is perfectly reproduced. In state-space, this can be interpreted as zero steady-state error. In general this can not be achieved by balanced truncation which performs particularly well at high frequencies ($\omega \to \infty$), with a perfect match at $\omega = \infty$. However, DC gain preservation is achieved by *singular perturbation approximation* (SPA), which proceeds as follows: let $(\tilde{A}, \tilde{B}, \tilde{C}, D)$ denote a minimal realization of the LTI system (1), and partition

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

according to the desired size r of the reduced-order model, that is, $A_{11} \in \mathbb{R}^{r \times r}$, $B_1 \in \mathbb{R}^{r \times m}$, and $C_1 \in \mathbb{R}^{p \times r}$. Then the SPA reduced-order model is obtained by the following formulae [LA86]:

$$\begin{array}{rclcrcl} \hat{A} & := & A_{11} + A_{12}A_{22}^{-1}A_{21}, & & \hat{B} & := & B_1 + A_{12}A_{22}^{-1}B_2, \\ \hat{C} & := & C_1 + C_2A_{22}^{-1}A_{21}, & & \hat{D} & := & D + C_2A_{22}^{-1}B_2. \end{array} \tag{36}$$

The resulting reduced-order model satisfies the absolute error bound in (29).

When computing the minimal realization with Algorithm 4 or its balancing-free variant, followed by (36), we can consider the resulting model reduction algorithm as a spectral projection method for SPA. Further details regarding the parallelization of this implementation of SPA, together with several numerical examples demonstrating its performance, can be found in [BQQ00b].

4.3.2 Cross-Gramian Methods

In some situations, the product W_cW_o of the system Gramians is the square root of the solution of the Sylvester equation

$$AW_{co} + W_{co}A + BC = 0. \tag{37}$$

The solution W_{co} of (37) is called the *cross-Gramian* of the system (1). Of course, for (37) to be well-defined, the system must be square, i.e., $p = m$. Then we have $W_{co}^2 = W_cW_o$ if

- the system is symmetric, which is trivially the case if $A = A^T$ and $C = B^T$ (in that case, both equations in (4) equal (37)) [FN84a];
- the system is a single-input/single-output (SISO) system, i.e., $p = m = 1$ [FN84b].

In both cases, instead of solving (4) it is possible to use (37). Also note that the cross-Gramian carries information of the LTI system and its internally balanced realization if it is not the product of the controllability and observability Gramian and can still be used for model reduction; see [Ald91, FN84b]. The computation of a reduced-order model from the cross-Gramian is based on computing the dominant W_{co}-invariant subspace which can again be achieved using (13) and (12) applied to a shifted version of W_{co}.

For $p, m \ll n$, a factorized version of (18) can be used to solve (37). This again can reduce significantly both the work space needed for saving the cross-Gramian and the computation time in case W_{co} is of low numerical rank; for details see [Ben04]. Also note that the B_j-iterates in (18) need not be computed as they equal the A_j's. This further reduces the computational cost of this approach significantly.

4.3.3 Stochastic Truncation

We assume here that $0 < p \leq m$, $\operatorname{rank}(D) = p$, which implies that $G(s)$ must not be strictly proper. For strictly proper systems, the method can be applied introducing an ϵ-regularization by adding an artificial matrix $D = [\epsilon I_p \; 0]$ [Glo86].

Balanced stochastic truncation (BST) is a model reduction method based on truncating a balanced stochastic realization. Such a realization is obtained as follows; see [Gre88] for details. Define the *power spectrum* $\Phi(s) = G(s)G^T(-s)$, and let W be a *square minimum phase right spectral factor* of Φ, satisfying $\Phi(s) = W^T(-s)W(s)$. As D has full row rank, $E := DD^T$ is positive definite, and a minimal state-space realization (A_W, B_W, C_W, D_W) of W is given by (see [And67a, And67b])

$$\begin{aligned} A_W &:= A, & B_W &:= BD^T + W_c C^T, \\ C_W &:= E^{-\frac{1}{2}}(C - B_W^T X_W), & D_W &:= E^{\frac{1}{2}}, \end{aligned}$$

where $W_c = S^T S$ is the controllability Gramian defined in (4), while X_W is the observability Gramian of $W(s)$ obtained as the stabilizing solution of the *algebraic Riccati equation* (ARE)

$$F^T X + XF + XB_W E^{-1} B_W^T X + C^T E^{-1} C = 0, \tag{38}$$

with $F := A - B_W E^{-1} C$. Here, X_W is symmetric positive (semi-)definite and thus admits a decomposition $X_W = R^T R$. If a reduced-order model is computed from an SVD of SR^T as in balanced truncation, then the reduced-order model $(\hat{A}, \hat{B}, \hat{C}, \hat{D})$ is stochastically balanced. That is, the Gramians $\hat{W}_c$, $\hat{X}_W$ of the reduced-order model satisfy

$$\hat{W}_c = \operatorname{diag}(\sigma_1, \ldots, \sigma_r) = \hat{X}_W, \tag{39}$$

where $1 = \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$. The BST reduced-order model satisfies the following relative error bound:

$$\sigma_{r+1} \leq \|\Delta_r\|_\infty \leq \prod_{j=r+1}^{n} \frac{1+\sigma_j}{1-\sigma_j} - 1, \tag{40}$$

where $G\Delta_r = G - \hat{G}$. From that we obtain

$$\frac{\|G - \hat{G}\|_\infty}{\|G\|_\infty} \leq \prod_{j=r+1}^{n} \frac{1+\sigma_j}{1-\sigma_j} - 1. \tag{41}$$

Therefore, BST is also a member of the class of relative error methods which aim at minimizing $\|\Delta_r\|$ for some system norm.

Implementing BST based on spectral projection methods differs in several ways from the versions proposed in [SC88, VF93], though they are mathematically equivalent. Specifically, the Lyapunov equation for W_c is solved using the sign function iteration described in subsection 3.3, from which we obtain a full-rank factorization $W_c = S^T S$. The same approach is used to compute a full-rank factor R of X_W from a stabilizing approximation $\tilde{X}_W$ to X_W using the technique described in [Var99]: let $D = \begin{bmatrix} \hat{D}^T & 0 \end{bmatrix} U$ be an LQ decomposition of D. Note that $\hat{D} \in \mathbb{R}^{p \times p}$ is a square, nonsingular matrix as D has full row rank. Now set

$$H_W := \hat{D}^{-T} C, \quad \hat{B}_W := B_W \hat{D}^{-1}, \quad \hat{C} := (H_W - \hat{B}_W^T X).$$

Then the ARE (38) is equivalent to $A^T X + X A + \hat{C}^T \hat{C} = 0$. Using a computed approximation $\tilde{X}_W$ of X_W to form $\hat{C}$, the Cholesky or full-rank factor R of X_W can be computed directly from the Lyapunov equation

$$A(R^T R) + (R^T R)A + \hat{C}^T \hat{C} = 0.$$

The approximation $\tilde{X}_W$ is obtained by solving (38) using Newton's method with exact line search as described in [Ben97] with the sign function method used for solving the Lyapunov equations in each Newton step; see [BQQ01] for details. The Lyapunov equation for R is solved using the sign function iteration from subsection 3.3.

4.3.4 Further Riccati-Based Truncation Methods

There is a variety of other balanced truncation methods for different choices of Gramians to be balanced; see, e.g., [GA03, Obe91]. Important methods are

positive-real balancing: here, passivity is preserved in the reduced-order model which is an important task in circuit simulation;

bounded-real balancing: preserves the $\mathcal{H}_\infty$ gain of the system and is therefore useful for robust control design;

LQG balancing: a closed-loop model reduction technique that preserves closed-loop performance in an LQG design.

In all these methods, the Gramians are solutions of two dual Riccati equations of a similar structure as the stochastic truncation ARE (38). The computation of full-rank factors of the system Gramians can proceed in an analogous manner as in BST, and the subsequent computation of the reduced-order system is analogous to the SR or BFSR method for balanced truncation. Therefore, implementations of these model reduction approaches with the computational approaches described so far can also be considered as spectral projection methods. The parallelization of model reduction based on positive-real balancing is described in [BQQ04b]; numerical results demonstrating the accuracy of the reduced-order models and the parallel performance can also be found there.

4.4 Unstable Systems

Model reduction for unstable systems can be performed in several ways. One idea is based on the fact that unstable poles are usually important for the dynamics of the system, hence they should be preserved. This can be achieved via an *additive decomposition* of the transfer function as

$$G(s) \;=\; G_-(s) + G_+(s),$$

with $G_-(s)$ stable, $G_+(s)$ unstable, applying balanced truncation to G_- to obtain $\hat{G}_-$, and setting

$$\hat{G}(s) := \hat{G}_-(s) + G_+(s),$$

thereby preserving the unstable part of the system. Such a procedure can be implemented using the spectral projection methods for block-diagonalization and balanced truncation: first, apply Algorithm 2 to A and set

$$\begin{aligned}
\tilde{A} &:= U^{-1}AU = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix}, \\
\tilde{B} &:= U^{-1}B =: \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}, \quad \tilde{C} := CU =: [\, C_1 \; C_2 \,], \quad \tilde{D} := D.
\end{aligned}$$

This yields the desired additive decomposition as follows:

$$\begin{aligned}
G(s) &= C(sI - A)^{-1}B + D = \tilde{C}(sI - \tilde{A})^{-1}\tilde{B} + \tilde{D} \\
&= \begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} (sI_k - A_{11})^{-1} & \\ & (sI_{n-k} - A_{22})^{-1} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} + D \qquad (42) \\
&= \big\{C_1(sI_k - A_{11})^{-1}B_1 + D\big\} + \big\{C_2(sI_{n-k} - A_{22})^{-1}B_2\big\} \\
&=: G_-(s) + G_+(s).
\end{aligned}$$

Then apply Algorithm 4 to G_- and obtain the reduced order model by adding the transfer functions of the stable reduced and the unstable unreduced parts as summarized above. This approach is described in more detail in [BCQQ04] where also some numerical examples are given. An extension of this approach using balancing for appropriately defined Gramians of unstable systems is discussed in [ZSW99]. This approach can also be implemented using sign function-based spectral projection techniques similar to the ones used so far.

Alternative model reduction techniques for unstable systems based on coprime factorization of the transfer function and application of balanced truncation to the stable coprime factors are surveyed in [Var01]. Of course, the spectral projection-based balanced truncation algorithm described in Section 4.2 could be used for this purpose. The computation of spectral factorizations of transfer functions purely based on spectral projection methods requires further investigation, though.

4.5 Optimal Hankel Norm Approximation

BT and SPA model reduction methods aim at minimizing the $\mathcal{H}_\infty$-norm of the error system $G - \hat{G}$. However, they usually do not succeed in finding an optimal approximation; see [AA02]. If a best approximation is desired, a different option is to use the *Hankel norm* of a stable rational transfer function, defined by

$$\|G\|_H := \sigma_1(G), \qquad (43)$$

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 $\mathcal{H}_\infty$ 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 [Glo84] 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 $\hat{G}$. That is, we can compute a best approximation of the system for a given McMillan degree of the reduced-order model which is usually not possible for other system norms such as the $\mathcal{H}_2$- or $\mathcal{H}_\infty$-norms.

The derivation of a realization of $\hat{G}$ is quite involved, see, e.g., [Glo84, ZDG96]. Here, we only describe the essential computational tools required in an implementation of the HNA method.

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 SR version of the BT method as given in Algorithm 4. 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

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > \sigma_{r+1} = \ldots = \sigma_{r+k} > \sigma_{r+k+1} \geq \ldots \geq \sigma_{\hat{n}} > 0,\ k \geq 1.$$

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, that is,

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

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 $\tilde{G}$:

$$\begin{aligned} \tilde{A} &= \Gamma^{-1}(\sigma_{r+1}^2 A_{11}^T + \check{\Sigma} A_{11} \check{\Sigma} + \sigma_{r+1} C_1^T U B_1^T), \\ \tilde{B} &= \Gamma^{-1}(\check{\Sigma} B_1 - \sigma_{r+1} C_1^T U), \\ \tilde{C} &= C_1 \check{\Sigma} - \sigma_{r+1} U B_1^T, \\ \tilde{D} &= D + \sigma_{r+1} U. \end{aligned} \tag{44}$$

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. For this additive decomposition we use exactly the same algorithm described in the last subsection. Then $\hat{G} := \tilde{G}_-$ is an optimal r-th order Hankel norm approximation of G.

Thus, the main computational tasks of a *spectral projection implementation of optimal Hankel norm approximation* is a combination of Algorithm 4, the formulae (44), and Algorithm 2; see [BQQ04a] for further details.

5 Application to Large-Scale Systems

5.1 Parallelization

Model reduction algorithms based on spectral projection methods are composed of basic matrix computations such as solving linear systems, matrix products, and QR factorizations. Efficient parallel routines for all these matrix computations are provided in linear algebra libraries for distributed memory computers such as PLAPACK and ScaLAPACK [BCC+97, van97]. The use of these libraries enhances both the reliability and portability of the model reduction routines. The performance will depend on the efficiency of the underlying serial and parallel computational linear algebra libraries and the communication routines.

Here we will employ the ScaLAPACK parallel library [BCC+97]. This is a freely available library that implements parallel versions of many of the kernels in LAPACK [ABB+99], using the message-passing paradigm. ScaLAPACK is based on the PBLAS (a parallel version of the serial BLAS) for computation and BLACS for communication. The BLACS can be ported to any (serial and) parallel architecture with an implementation of the MPI or the PVM libraries [GBD+94, GLS94].

In ScaLAPACK the computations are performed by a logical grid of $n_p = p_r \times p_c$ processes. The processes are mapped onto the physical processors, depending on the available number of these. All data (matrices) have to be distributed among the process grid prior to the invocation of a ScaLAPACK routine. It is the user's responsibility to perform this data distribution. Specifically, in ScaLAPACK the matrices are partitioned into $mb \times nb$ blocks and these blocks are then distributed (and stored) among the processes in column-major order (see [BCC+97] for details).

Using the kernels in ScaLAPACK, we have implemented a library for model reduction of LTI systems, PLiCMR[1], in Fortran 77. The library contains a few driver routines for model reduction and several computational routines for the solution of related equations in control. The functionality and naming convention of the parallel routines closely follow analogous routines from SLICOT. As part of PLiCMR, three parallel driver routines are provided for absolute error model reduction, two parallel driver routines for relative error model reduction, and an expert driver routine capable of performing any of the previous functions on stable and unstable systems. Table 2 lists all the driver routines. The driver routines are based on several computational routines included in PLiCMR and listed in Table 3. Note that the missing routines in the discrete-time case are available in the Parallel Library in Control (PLiC) [BQQ99], but are not needed in the PLiCMR codes for model reduction of discrete-time systems.

[1] Available from `http://spine.act.uji.es/~plicmr.html`.

Purpose	Routine	
Expert driver	pab09mr	
SR/BFSR BT alg.	pab09ax	
SR/BFSR SPA alg.	pab09bx	
HNA alg.	pab09cx	
SR/BFSR BST alg.	pab09hx	
	Continuous-time	Discrete-time
SR/BFSR PRBT alg.	pab09px	–

Table 2: Driver routines in PLiCMR.

Purpose	Routine	
Solve dual Lyapunov equations and compute HSV	pab09ah	
Compute T_l, T_r from SR formulae	pab09as	
Compute T_l, T_r from BFSR formulae	pab09aw	
Obtain reduced-order model from T_l, T_r	pab09at	
Spectral division by sign function	pmb05rd	
Factorize TFM into stable/unstable parts	ptb01kd	
	Continuous-time	Discrete-time
ARE solver	pdgecrny	–
Sylvester solver	psb04md	–
Lyapunov solver	pdgeclnw	–
Lyapunov solver (for the full-rank factor)	pdgeclnc	–
Dual Lyapunov/Stein solver	psb03odc	psb03odd

Table 3: Computational routines in PLiCMR.

A more detailed introduction to PLiCMR and numerical results showing the model reduction abilities of the implemented methods and their parallel performance can be found in [BQQ03b].

5.2 Data-Sparse Implementation of the Sign Function Method

The key to a balanced truncation implementation based on Algorithm 4 with reduced complexity lies in reducing the complexity of storing A and of performing the required computations with A. Recall that the solution of the Lyapunov equation

$$A^T X + XA + C^T C = 0 \tag{45}$$

(or its dual in (4)) with the sign function method (21) involves the inversion, addition and multiplication of $n \times n$ matrices. Using an approximation of A in $\mathcal{H}$-matrix format [GH03, GHK03] and formatted $\mathcal{H}$-matrix arithmetic, the complexity of storing A and the aforementioned computations reduces to $\mathcal{O}(n \log^2 n)$.

We will briefly describe this approach in the following; for more details and numerical examples see [BB04].

Hierarchical ($\mathcal{H}$-)matrices are a data-sparse approximation of large, dense matrices arising from the discretization of non-local integral operators occurring in the boundary element

method or as inverses of FEM discretized elliptic differential operators, but can also be used to represent FEM matrices directly.

Important properties of $\mathcal{H}$-matrices are:

- only few data are needed for the representation of the matrix,
- matrix-vector multiplication can be performed in almost linear complexity ($\mathcal{O}(n \log n)$),
- sums, products, inverses of $\mathcal{H}$-matrices are of "almost" linear complexity.

The basic construction principle of $\mathcal{H}$-matrices can be described as follows: consider matrices over a product index set $\mathcal{I} \times \mathcal{I}$ and partition $\mathcal{I} \times \mathcal{I}$ by an *$\mathcal{H}$-tree* $T_{\mathcal{I}\times\mathcal{I}}$, where a problem dependent *admissibility condition* is used to decide whether a block $t \times s \subset \mathcal{I} \times \mathcal{I}$ allows for a low rank approximation of this block.

Definition 5.1 *[GH03] The set of* hierarchical matrices *is defined by*

$$\begin{aligned}\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k) \quad := \quad & \{M \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}} | \operatorname{rank}(M|_{t\times s}) \leq k \text{ for all} \\ & \text{admissible leaves } t \times s \text{ of } T_{\mathcal{I}\times\mathcal{I}}\}.\end{aligned}$$

Submatrices of $M \in \mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$ corresponding to inadmissible leaves are stored as dense blocks whereas those corresponding to admissible leaves are stored in factorized form as rank-k matrices, called *R_k-format.* Figure 5 shows the $\mathcal{H}$-matrix representation with $k = 4$ of the stiffness matrix of the FEM discretization for a 2D heat equation with distributed control and isolation boundary conditions using linear elements on a uniform mesh, resulting in $n = 1024$.

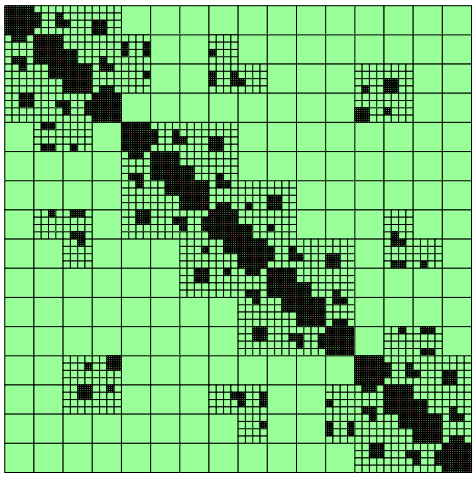

Figure 5: $\mathcal{H}$-matrix representation of stiffness matrix for 2D heat equation with distributed control and isolation boundary conditions. Here $n = 1024$ and $k = 4$.

The formatted arithmetic for $\mathcal{H}$-matrices is not a usual arithmetic as $\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$ is not a linear subspace of $\mathbb{R}^{\mathcal{I}\times\mathcal{I}}$, hence sums, products, and inverses of $\mathcal{H}$-matrices need to be projected into $\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$. In short, the operations needed here are

Formatted addition ($\oplus$) with complexity $\mathcal{N}_{\mathcal{H}\oplus\mathcal{H}} = \mathcal{O}(nk^2 \log n)$); the computed $\mathcal{H}$-matrix is the best approximation (with respect to the Frobenius-norm) in $\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$ of the sum of two $\mathcal{H}$-matrices.

Formatted multiplication ($\odot$) with complexity $\mathcal{N}_{\mathcal{H}\odot\mathcal{H}} = \mathcal{O}(nk^2 \log^2 n)$;

Formatted inversion ($\widetilde{\mathrm{Inv}}$) with complexity $\mathcal{N}_{\mathcal{H},\widetilde{\mathrm{Inv}}} = \mathcal{O}(nk^2 \log^2 n)$.

For the complexity results, some technical assumptions on the $\mathcal{H}$-tree $T_{\mathcal{I}\times\mathcal{I}}$ are needed.

The sign function iteration (21) for (45) using formatted $\mathcal{H}$-matrix arithmetic with $A_{\mathcal{H}}$ denoting the $\mathcal{H}$-matrix representation in $\mathcal{H}(T_{\mathcal{I}\times\mathcal{I}}, k)$ then becomes

$$\begin{aligned} &A_0 \;\leftarrow\; A_{\mathcal{H}}, \quad C_0 \;\leftarrow\; C, \\ &\texttt{for } j = 0, 1, 2, \ldots \\ &\qquad A_{j+1} \;\leftarrow\; \frac{1}{2\gamma_j}\left(A_j \oplus \gamma_j^2 \widetilde{\mathrm{Inv}}(A_j)\right), \\ &\qquad \tilde{C}_{j+1} \;\leftarrow\; \frac{1}{2\sqrt{\gamma_j}}\left[\; C_j, \quad \gamma_j C_j \odot \widetilde{\mathrm{Inv}}(A_j) \;\right], \\ &\qquad C_{j+1} \;\leftarrow\; R\text{-factor of RRQR as in (24).} \end{aligned} \tag{46}$$

Using this method for solving the Lyapunov equations in the first step of Algorithm 4, we obtain an implementation of balanced truncation requiring only $\mathcal{O}(n_{co} nk \log^2 n)$ storage and $\mathcal{O}(r n_{co} k^2 n \log^2 n)$ flops. Work on this topic is in progress, first numerical results reported in [BB04] are promising that this approach will lend itself to efficient model reduction methods for the control of parabolic partial differential equations.

6 Conclusions and Open Problems

Spectral projection methods, in particular those based on the matrix sign function, provide an easy-to-use and easy-to-implement framework for many model reduction techniques. Using the implementations suggested here, balanced truncation and related methods can easily be applied to systems of order $\mathcal{O}(10^3)$ on desktop computers, of order $\mathcal{O}(10^4)$ using parallel programming models, and to more or less unlimited orders if sparse implementations based on matrix compression techniques and formatted arithmetic can be used.

Further investigations could lead to a combination of spectral projection methods based on the sign function with wavelet techniques for the discretization of partial differential equations.

Open problems are the derivation of error bounds for several balancing-related techniques that allow an adaptive choice of the order of the reduced-order model for a given tolerance threshold. This would be particulary important for positive-real balancing as this technique could be very useful in circuit simulation and microsystem technology. The extension of the Riccati-based truncation techniques related to stochastic, positive-real, bounded-real, and LQG balancing to descriptor systems is another topic for further investigations, both theoretically and computationally.

Acknowledgements

We would like to thank our co-workers Ulrike Baur, Maribel Castillo, José M. Claver, Rafa Mayo, and Gregorio Quintana-Ortí— only through our collaboration all the results discussed

in this work could be achieved. We also gratefully acknowledge the helpful remarks and suggestions of an anonymous referee which significantly improved the presentation of this paper.

This work was partially supported by the CICYT project No. TIC2002-004400-C03-01 and project No. P1B-2004-6 of the *Fundación Caixa-Castellón/Bancaixa and UJI*.

References

[AA02] Antoulas, A.C., Astolfi, A.: H_∞-norm approximation. In: Blondel, V.D., Megretski, A. (editors), 2002 MTNS Problem Book, Open Problems on the Mathematical Theory of Networks and Systems, pages 73–76 (2002). Available online from `http://www.nd.edu/~mtns/OPMTNS.pdf`.

[ABB+99] Anderson, E., Bai, Z., Bischof, C., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Sorensen, D.: LAPACK Users' Guide, SIAM, Philadelphia, PA, third edition (1999).

[Ald91] Aldhaheri, R.W.: Model order reduction via real Schur-form decomposition. Internat. J. Control, **53**:3, 709–716 (1991).

[And67a] Anderson, B.D.O.: An algebraic solution to the spectral factorization problem. IEEE Trans. Automat. Control, **AC-12**, 410–414 (1967).

[And67b] Anderson, B.D.O.: A system theory criterion for positive real matrices. SIAM J. Cont., **5**, 171–182 (1967).

[ANS] ANSYS, Inc., http://www.ansys.com. ANSYS.

[AS01] Antoulas, A.C., Sorensen, D.C.: Approximation of large-scale dynamical systems, An overview. Int. J. Appl. Math. Comp. Sci., **11**:5, 1093–1121 (2001).

[ASZ02] Antoulas, A.C., Sorensen, D.C., Zhou, Y.: On the decay rate of Hankel singular values and related issues. Sys. Control Lett., **46**:5, 323–342 (2002).

[BB04] Baur, U., Benner, P.: Factorized solution of the Lyapunov equation by using the hierarchical matrix arithmetic. Proc. Appl. Math. Mech., **4**:1, 658–659 (2004).

[BCC+97] Blackford, L.S., Choi, J., Cleary, A., D'Azevedo, E., Demmel, J., Dhillon, I., Dongarra, J., Hammarling, S., Henry, G., Petitet, A., Stanley, K., Walker, D., Whaley, R.C.: ScaLAPACK Users' Guide. SIAM, Philadelphia, PA (1997).

[BCQ98] Benner, P., Claver, J.M., Quintana-Ortí, E.S.: Efficient solution of coupled Lyapunov equations via matrix sign function iteration. In: Dourado, A. et al. (editors), Proc. 3rd Portuguese Conf. on Automatic Control CONTROLO'98, Coimbra, pages 205–210 (1998).

[BCQQ04] Benner, P., Castillo, M., Quintana-Ortí, E.S., Quintana-Ortí, G.: Parallel model reduction of large-scale unstable systems. In: Joubert, G.R., Nagel, W.E., Peters, F.J., Walter, W.V. (editors), Parallel Computing: Software Technology,

Algorithms, Architectures & Applications. Proc. Intl. Conf. ParCo2003, Dresden, Germany, volume 13 of Advances in Parallel Computing, pages 251–258. Elsevier B.V. (North-Holland) (2004).

[BD75] Beavers, A.N., Denman, E.D.: A new solution method for the Lyapunov matrix equations. SIAM J. Appl. Math., **29**, 416–421 (1975).

[BD93] Bai, Z., Demmel, J.: Design of a parallel nonsymmetric eigenroutine toolbox, Part I. In: R.F. Sincovec et al., editor, Proceedings of the Sixth SIAM Conference on Parallel Processing for Scientific Computing, pages 391–398, SIAM, Philadelphia, PA (1993). *See also:* Tech. Report CSD-92-718, Computer Science Division, University of California, Berkeley, CA 94720.

[BD98] Bai, Z., Demmel, J.: Using the matrix sign function to compute invariant subspaces. SIAM J. Matrix Anal. Appl., **19**:1, 205–225 (1998).

[BDD+97] Bai, Z., Demmel, J., Dongarra, J, Petitet, A., Robinson, H., Stanley, K.: The spectral decomposition of nonsymmetric matrices on distributed memory parallel computers. SIAM J. Sci. Comput., **18**, 1446–1461 (1997).

[Ben97] Benner, P.: Numerical solution of special algebraic Riccati equations via an exact line search method. In: Proc. European Control Conf. ECC 97 (CD-ROM), Paper 786. BELWARE Information Technology, Waterloo, Belgium (1997).

[Ben04] Benner, P.: Factorized solution of Sylvester equations with applications in control. In: Proc. Intl. Symp. Math. Theory Networks and Syst. MTNS 2004, `http://www.mtns2004.be` (2004).

[BHM97] Byers, R., He, C., Mehrmann, V.: The matrix sign function method and the computation of invariant subspaces. SIAM J. Matrix Anal. Appl., **18**:3, 615–632 (1997).

[BMS+99] Benner, P., Mehrmann, V., Sima, V., Van Huffel, S., Varga, A.: SLICOT - a subroutine library in systems and control theory. In: Datta, B.N. (editor), Applied and Computational Control, Signals, and Circuits, volume 1, chapter 10, pages 499–539. Birkhäuser, Boston, MA (1999).

[BQ99] Benner, P., Quintana-Ortí, E.S.: Solving stable generalized Lyapunov equations with the matrix sign function. Numer. Algorithms, **20**:1, 75–100 (1999).

[BQQ99] Benner, P., Quintana-Ortí, E.S., Quintana-Ortí, G.: A portable subroutine library for solving linear control problems on distributed memory computers. In: Cooperman, G., Jessen, E., Michler, G.O. (editors), Workshop on Wide Area Networks and High Performance Computing, Essen (Germany), September 1998, Lecture Notes in Control and Information, pages 61–88. Springer-Verlag, Berlin/Heidelberg, Germany (1999).

[BQQ00a] Benner, P., Quintana-Ortí, E.S., Quintana-Ortí, G.: Balanced truncation model reduction of large-scale dense systems on parallel computers. Math. Comput. Model. Dyn. Syst., **6**:4, 383–405 (2000).

[BQQ00b] Benner, P., Quintana-Ortí, E.S., Quintana-Ortí, G.: Singular perturbation approximation of large, dense linear systems. In: Proc. 2000 IEEE Intl. Symp. CACSD, Anchorage, Alaska, USA, September 25–27, 2000, pages 255–260. IEEE Press, Piscataway, NJ (2000).

[BQQ01] Benner, P., Quintana-Ortí, E.S., Quintana-Ortí, G.: Efficient numerical algorithms for balanced stochastic truncation. Int. J. Appl. Math. Comp. Sci., **11**:5, 1123–1150 (2001).

[BQQ03a] Benner, P., Quintana-Ortí, E.S., Quintana-Ortí, G.: Parallel algorithms for model reduction of discrete-time systems. Int. J. Syst. Sci., **34**:5, 319–333 (2003).

[BQQ03b] Benner, P., Quintana-Ortí, E.S., Quintana-Ortí, G.: State-space truncation methods for parallel model reduction of large-scale systems. Parallel Comput., **29**, 1701–1722 (2003).

[BQQ04a] Benner, P., Quintana-Ortí, E.S., Quintana-Ortí, G.: Computing optimal Hankel norm approximations of large-scale systems. In: Proc. 43rd IEEE Conf. Decision Contr., pages 3078–3083. Omnipress, Madison, WI (2004).

[BQQ04b] Benner, P., Quintana-Ortí, E.S., Quintana-Ortí, G.: Computing passive reduced-order models for circuit simulation. In: Proc. Intl. Conf. Parallel Comp. in Elec. Engrg. PARELEC 2004, pages 146–151. IEEE Computer Society, Los Alamitos, CA (2004).

[BQQ04c] Benner, P., Quintana-Ortí, E.S., Quintana-Ortí, G.: Parallel model reduction of large-scale linear descriptor systems via Balanced Truncation. In: High Performance Computing for Computational Science. Proc. 6th Intl. Meeting VECPAR'04, June 28–30, 2004, Valencia, Spain, pages 65–78 (2004).

[BQQ04d] Benner, P., Quintana-Ortí, E.S., Quintana-Ortí, G.: Solving linear matrix equations via rational iterative schemes. Technical Report SFB393/04-08, Sonderforschungsbereich 393 *Numerische Simulation auf massiv parallelen Rechnern*, TU Chemnitz, 09107 Chemnitz, FRG (2004). Available from `http://www.tu-chemnitz.de/sfb393/preprints.html`.

[BS05] Benner, P., Saak, J.: A semi-discretized heat transfer model for optimal cooling of steel profiles. In: Benner, P., Mehrmann, V., Sorensen, D.C. (editors), Dimension Reduction of Large-Scale Systems, vol. 45 of Lecture Notes in Computational Science and Engineering, Springer-Verlag, Berlin/Heidelberg (2005).

[Bye87] Byers, R.: Solving the algebraic Riccati equation with the matrix sign function. Linear Algebra Appl., **85**, 267–279 (1987).

[CB68] Craig, R.R., Bampton, M.C.C.: Coupling of substructures for dynamic analysis. AIAA J., **6**, 1313–1319 (1968).

[CV05] Chahlaoui, Y., Van Dooren, P.: Benchmark examples for model reduction of linear time-invariant dynamical systems. In: Benner, P., Mehrmann, V., Sorensen, D.C. (editors), Dimension Reduction of Large-Scale Systems, vol. 45 of Lecture Notes in Computational Science and Engineering, Springer-Verlag, Berlin/Heidelberg (2005).

[Dav66] Davison, E.J.: A method for simplifying linear dynamic systems. IEEE Trans. Automat. Control, **AC-11**, 93–101 (1966).

[DB76] Denman, E.D., Beavers, A.N.: The matrix sign function and computations in systems. Appl. Math. Comput., **2**, 63–94 (1976).

[FN84a] Fernando, K.V., Nicholson, H.: On a fundamental property of the cross-Gramian matrix. IEEE Trans. Circuits Syst., **CAS-31**:5, 504–505 (1984).

[FN84b] Fernando, K.V., Nicholson, H.: On the structure of balanced and other principal representations of linear systems. IEEE Trans. Automat. Control, **AC-28**:2, 228–231 (1984).

[Föl94] Föllinger, O.: Regelungstechnik. Hüthig-Verlag, 8. edition (1994).

[GA03] Gugercin, S., Antoulas, A.C.: A survey of balancing methods for model reduction. In: Proc. European Control Conference ECC 2003, Cambridge, UK (2003). CD Rom.

[GBD+94] Geist, A., Beguelin, A., Dongarra, J., Jiang, W., Manchek, B., Sunderam, V.: PVM: Parallel Virtual Machine – A Users Guide and Tutorial for Network Parallel Computing. MIT Press, Cambridge, MA (1994).

[GH03] Grasedyck, L., W. Hackbusch, W.: Construction and arithmetics of $\mathcal{H}$-matrices. Computing, **70**, 295–334 (2003).

[GHK03] Grasedyck, L., Hackbusch, W., Khoromskij, B.N.: Solution of large scale algebraic matrix Riccati equations by use of hierarchical matrices. Computing, **70**, 121–165 (2003).

[GL95] Green, M., Limebeer, D.J.N.: Linear Robust Control. Prentice-Hall, Englewood Cliffs, NJ (1995).

[GL05] Gugercin, S., Li, J.-R.: Smith-type methods for balanced truncation of large sparse systems. In: Benner, P., Mehrmann, V., Sorensen, D.C. (editors), Dimension Reduction of Large-Scale Systems, vol. 45 of Lecture Notes in Computational Science and Engineering, Springer-Verlag, Berlin/Heidelberg (2005).

[Glo84] Glover, K.: All optimal Hankel-norm approximations of linear multivariable systems and their L^∞ norms. Internat. J. Control, **39**, 1115–1193 (1984).

[Glo86] Glover, K.: Multiplicative approximation of linear multivariable systems with L_∞ error bounds. In: Proc. American Control Conf., pages 1705–1709 (1986).

[GLS94] Gropp, W., Lusk, E., Skjellum, A.: Using MPI: Portable Parallel Programming with the Message-Passing Interface. MIT Press, Cambridge, MA (1994).

[Gra04] Grasedyck, L.: Existence of a low rank or H-matrix approximant to the solution of a Sylvester equation. Numer. Lin. Alg. Appl., **11**, 371–389 (2004).

[Gre88] Green, M.: Balanced stochastic realization. Linear Algebra Appl., **98**, 211–247 (1988).

[Guy68] Guyan, R.J.: Reduction of stiffness and mass matrices. AIAA J., **3**, 380 (1968).

[GV96] Golub, G.H., Van Loan, C.F.: Matrix Computations. Johns Hopkins University Press, Baltimore, third edition (1996).

[Ham82] Hammarling, S.J.: Numerical solution of the stable, non-negative definite Lyapunov equation. IMA J. Numer. Anal., **2**, 303–323 (1982).

[HBZ05] Hohlfeld, D., Bechtold, T., Zappe, H.: Tunable Optical Filter. In: Benner, P., Mehrmann, V., Sorensen, D.C. (editors), Dimension Reduction of Large-Scale Systems, vol. 45 of Lecture Notes in Computational Science and Engineering, Springer-Verlag, Berlin/Heidelberg (2005).

[Hig86] Higham, N.J.: Computing the polar decomposition—with applications. SIAM J. Sci. Statist. Comput., **7**, 1160–1174 (1986).

[HMW77] Hoskins, W.D., Meek, D.S., Walton, D.J.: The numerical solution of $A'Q+QA = -C$. IEEE Trans. Automat. Control, **AC-22**, 882–883 (1977).

[HQSW00] Huss, S., Quintana-Ortí, E.S., Sun, X., Wu, J.: Parallel spectral division using the matrix sign function for the generalized eigenproblem. Int. J. of High Speed Computing, **11**:1, 1–14 (2000).

[KL92] Kenney, C., Laub, A.J.:. On scaling Newton's method for polar decomposition and the matrix sign function. SIAM J. Matrix Anal. Appl., **13**, 688–706 (1992).

[KL95] Kenney, C., Laub, A.J.:. The matrix sign function. IEEE Trans. Automat. Control, **40**:8, 1330–1348 (1995).

[KMP01] Konstantinov, M.M., Mehrmann, V., Petkov, P.Hr.: Perturbation analysis for the Hamiltonian Schur form. SIAM J. Matrix Anal. Appl., **23**:2, 387–424 (2001).

[LA86] Liu, Y., Anderson, B.D.O. : Controller reduction via stable factorization and balancing. Internat. J. Control, **44**, 507–531 (1986).

[LA93] Larin, V.B., Aliev, F.A.: Construction of square root factor for solution of the Lyapunov matrix equation. Sys. Control Lett., **20**, 109–112 (1993).

[Lev96] Levine, W.S. (editor): The Control Handbook. CRC Press (1996).

[LHPW87] Laub, A.J., Heath, M.T., Paige, C.C., Ward, R.C.: Computation of system balancing transformations and other application of simultaneous diagonalization algorithms. IEEE Trans. Automat. Control, **34**, 115–122 (1987).

[LL96] Lang, W., Lezius, U.: Numerical realization of the balanced reduction of a control problem. In: H. Neunzert, editor, Progress in Industrial Mathematics at ECMI94, pages 504–512, John Wiley & Sons Ltd and B.G. Teubner, New York and Leipzig (1996).

[LR95] Lancaster, P., Rodman, L.: The Algebraic Riccati Equation. Oxford University Press, Oxford (1995).

[LT85] Lancaster, P., Tismenetsky, M.: The Theory of Matrices. Academic Press, Orlando, 2nd edition (1985).

[Mar66] Marschall, S.A.: An approximate method for reducing the order of a linear system. Contr. Eng., **10**, 642–648 (1966).

[Moo81] Moore, B.C.: Principal component analysis in linear systems: Controllability, observability, and model reduction. IEEE Trans. Automat. Control, **AC-26**, 17–32 (1981).

[MR76] Mullis, C., Roberts, R.A.: Synthesis of minimum roundoff noise fixed point digital filters. IEEE Trans. Circuits and Systems, **CAS-23**:9, 551–562 (1976).

[MS05] Mehrmann, V., Stykel, T.: Balanced truncation model reduction for large-scale systems in descriptor form. In: Benner, P., Mehrmann, V., Sorensen, D.C. (editors), Dimension Reduction of Large-Scale Systems, vol. 45 of Lecture Notes in Computational Science and Engineering, Springer-Verlag, Berlin/Heidelberg (2005).

[MSC] MSC.Software Corporation, `http://www.mscsoftware.com`. MSC.Nastran.

[Mut99] Mutambara, A.G.O.: Design and Analysis of Control Systems. CRC Press, Boca Raton, FL (1999).

[OA01] Obinata, G., Anderson, B.D.O.: Model Reduction for Control System Design. Communications and Control Engineering Series. Springer-Verlag, London, UK (2001).

[Obe91] Ober, R.: Balanced parametrizations of classes of linear systems. SIAM J. Cont. Optim., **29**, 1251–1287 (1991).

[Pen00] Penzl, T.: A cyclic low rank Smith method for large sparse Lyapunov equations. SIAM J. Sci. Comput., **21**:4, 1401–1418 (2000).

[Rob80] Roberts, J.D.: Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. Internat. J. Control, 32, 677–687 (1980). (Reprint of Technical Report No. TR-13, CUED/B-Control, Cambridge University, Engineering Department (1971).)

[SC88] Safonov, M.G., Chiang, R.Y.: Model reduction for robust control: A Schur relative error method. Int. J. Adapt. Cont. and Sign. Proc., **2**, 259–272 (1988).

[SC89] Safonov, M.G., Chiang, R.Y.: A Schur method for balanced-truncation model reduction. IEEE Trans. Automat. Control, **AC-34**, 729–733 (1989).

[Son98] Sontag, E.D.: Mathematical Control Theory. Springer-Verlag, New York, NY, 2nd edition (1998).

[TP87] Tombs, M.S., I. Postlethwaite, I.: Truncated balanced realization of a stable non-minimal state-space system. Internat. J. Control, **46**:4, 1319–1330 (1987).

[van97] van de Geijn, R.A.: Using PLAPACK: Parallel Linear Algebra Package. MIT Press, Cambridge, MA (1997).

[Van00] Van Dooren,P.: Gramian based model reduction of large-scale dynamical systems. In: D.F. Griffiths G.A. Watson, editors, Numerical Analysis 1999. Proc. 18th Dundee Biennial Conference on Numerical Analysis, pages 231–247, Chapman & Hall/CRC, London, UK (2000).

[Var91] Varga, A.: Efficient minimal realization procedure based on balancing. In: Prepr. of the IMACS Symp. on Modelling and Control of Technological Systems, volume 2, pages 42–47 (1991).

[Var99] Varga, A.: Task II.B.1 – selection of software for controller reduction. SLICOT Working Note 1999–18, The Working Group on Software (WGS) (1999). Available from `http://www.win.tue.nl/niconet/NIC2/reports.html`.

[Var01] Varga, A.: Model reduction software in the SLICOT library. In: B.N. Datta, editor, Applied and Computational Control, Signals, and Circuits, volume 629 of The Kluwer International Series in Engineering and Computer Science, pages 239–282, Kluwer Academic Publishers, Boston, MA (2001).

[VF93] Varga, A., Fasol, K.H.: A new square–root balancing–free stochastic truncation model reduction algorithm. In: Prepr. 12th IFAC World Congress, volume 7, pages 153–156, Sydney, Australia (1993).

[ZDG96] Zhou, K., Doyle, J.C., Glover, K.: Robust and Optimal Control. Prentice-Hall, Upper Saddle River, NJ (1996).

[ZSW99] Zhou, K., Salomon, G., Wu, E.: Balanced realization and model reduction for unstable systems. Int. J. Robust Nonlinear Control, **9**:3, 183–198, (1999).