Reduced-Order Modeling*

Zhaojun Bai† Patrick M. Dewilde‡ and Roland W. Freund§

In recent years, reduced-order modeling techniques have proven to be powerful tools for various problems in circuit simulation. For example, today, reduction techniques are routinely used to replace the large RCL subcircuits that model the interconnect or the pin package of VLSI circuits by models of much smaller dimension. In this paper, we review the reduced-order modeling techniques that are most widely employed in VLSI circuit simulation.

*Numerical Analysis Manuscript No. 02–4–13, Bell Laboratories, Murray Hill, New Jersey, March 2002. Available on WWW at `http://cm.bell-labs.com/cs/doc/02`.

†Department of Computer Science, University of California at Davis, One Shields Avenue, Davis, California 95616 (bai@cs.ucdavis.edu).

‡TU Delft, Circuits and Systems Group, Faculty of Electrical Engineering, Mekelweg 4, 2628 CD Delft, The Netherlands (dewilde@dimes.tudelft.nl).

§Bell Laboratories, Lucent Technologies, 700 Mountain Avenue, Room 2C–525, Murray Hill, New Jersey 07974–0636 (freund@research.bell-labs.com).

1 Introduction to the problem of model reduction

Roughly speaking, the problem of model reduction is to replace a given mathematical model of a system or process by a model that is much "smaller" than the original model, but still describes—at least approximately—certain aspects of the system or process. Clearly, model reduction involves a number of interesting issues. First and foremost is the issue of selecting appropriate approximation schemes that allow the definition of suitable reduced-order models. In addition, it is often important that the reduced-order model preserves certain crucial properties of the original system, such as stability or passivity. Other issues include the characterization of the quality of the models, the extraction of the data from the original model that is needed to actually generate the reduced-order models, and the efficient and numerically stable computation of the models.

In this paper, we discuss reduced-order modeling techniques for large-scale linear dynamical systems, especially those that arise in the simulation of electronic circuits and of microelectromechanical systems.

We begin with a brief description of reduced-order modeling problems in circuit simulation. Electronic circuits are usually modeled as networks whose branches correspond to the circuit elements and whose nodes correspond to the interconnections of the circuit elements. Such networks are characterized by three types of equations. The *Kirchhoff's current law* (KCL) states that, for each node of the network, the currents flowing in and out of that node sum up to zero. The *Kirchhoff's voltage law* (KVL) states that, for each closed loop of the network, the voltage drops along that loop sum up to zero. The *branch constitutive relations* (BCRs) are equations that characterize the actual circuit elements. For example, the BCR of a linear resistor is Ohm's law. The BCRs are linear equations for simple devices, such as linear resistors, capacitors, and inductors, and they are nonlinear equations for more complex devices, such as diodes and transistors. Furthermore, in general, the BCRs involve time-derivatives of the unknowns, and thus they are ordinary differential equations. On the other hand, the KCLs and KVLs are linear algebraic equations that only depend on the topology of the circuit.

The KCLs, KVLs, and BCRs can be summarized as a system of first-order, in general nonlinear, *differential-algebraic equations* (DAEs) of the form

$$\frac{\mathrm{d}}{\mathrm{d}t} q(\hat{x}, t) + f(\hat{x}, t) = 0, \tag{1}$$

together with suitable initial conditions. Here, $\hat{x} = \hat{x}(t)$ is the unknown vector of circuit variables at time t, the vector-valued function $f(\hat{x}, t)$ represents the contributions of nonreactive elements such as resistors, sources, etc., and the vector-valued function $\frac{\mathrm{d}}{\mathrm{d}t} q(\hat{x}, t)$ represents the contributions of reactive elements such as capacitors and inductors. There are a number of established methods, such as sparse tableau, nodal formulation, modified nodal analysis, etc. [58], for generating a system of equations of the form (1) from a so-called *netlist* description of a given circuit. The vector functions $\hat{x}$, f, q, as well as their dimension, depend on the chosen formulation method. The most general method is sparse tableau, which consists of just listing all the KCLs, KVLs, and BCRs. The other formulation methods can be interpreted as starting from sparse tableau and eliminating some of the unknowns by using some of the KCL or KVL equations.

For all the standard formulation methods, the dimension of the system (1) is of the order of the number of elements in the circuit. Since today's VLSI circuits can have up to hundreds of millions of circuit elements, systems (1) describing such circuits can be of extremely large dimension. Reduced-order modeling allows to first replace large systems of the form (1) by systems of smaller dimension and then tackle these smaller systems by suitable DAE solvers. Ideally, one would like to apply nonlinear reduced-order modeling directly to the nonlinear system (1). However, since nonlinear reduction techniques are a lot less developed and less well-understood than linear ones, today, almost always linear reduced-order modeling is employed. To this end, one either linearizes the system (1) or decouples (1) into nonlinear and linear subsystems; see, e.g., [31] and the references given there.

For example, the first case arises in *small-signal analysis*; see, e.g., [35]. Given a *DC operating point*, say $\hat{x}_0$, of the circuit described by (1), one linearizes the system (1) around $\hat{x}_0$. The resulting linearized version of (1) is of the following form:

$$E\frac{\mathrm{d}x}{\mathrm{d}t} = Ax + Bu(t), \tag{2}$$

$$y(t) = C^T x(t). \tag{3}$$

Here, $A = \mathrm{D}_x f$ and $E = \mathrm{D}_x q$ are the Jacobian matrices of f and q, respectively, at the DC operating point $\hat{x}_0$, $x(t) = \hat{x}(t) - \hat{x}_0$, $u(t)$ is the vector of excitations applied to the sources of the circuit, and $y(t)$ is the vector of circuit variables of interest. Equations (2) and (3) represent a *time-invariant linear dynamical system*. Its *state-space dimension*, N, is the length of the vector x of circuit variables. For a circuit with many elements, the system (2) and (3) is thus of very high dimension. The idea of reduced-order modeling is then to replace the original system (2) and (3) by one the same form,

$$E_n\frac{\mathrm{d}z}{\mathrm{d}t} = A_n z + B_n u(t),$$

$$y(t) = C_n^T z(t),$$

but of much smaller state-space dimension $n \ll N$.

Time-invariant linear dynamical systems of the form (2) and (3) also arise when equations describing linear subcircuits of a given circuit are decoupled from the system (1) that characterizes the whole circuit; see, e.g., [31]. For example, the interconnect or the pin package of VLSI circuits are often modeled as large linear RCL networks. Such linear subcircuits are described by systems of the form (2) and (3), where $x(t)$ is the vector of circuit variables associated with the subcircuit, and the vectors $u(t)$ and $y(t)$ contain the variables of the connections of the subcircuit to the, in general nonlinear, remainder of the whole circuit. By replacing, in the nonlinear system (1), the linear subsystem (2) and (3) by a reduced-order model of much smaller state-space dimension, the dimension of (1) can be reduced significantly before a DAE solver is then applied to such a smaller version of (1).

The remainder of this paper is organized as follows. In Section 2, we review some basic facts about time-invariant linear dynamical systems. In Section 3, we discuss reduced-order modeling of linear dynamical systems via Krylov-subspace techniques. In Section 4, we describe the use of Schur interpolation for various reduced-order modeling problems. In Section 5, we discuss Hankel-norm model reduction. Section 6 and 7 are concerned with reduced-order modeling of second-order and semi-second-order dynamical systems. Finally, in Section 8, we make some concluding remarks.

2 Time-invariant linear dynamical systems

In this section, we review some basic facts about time-invariant linear dynamical systems and introduce reduced-order models defined by Padé or Padé-type approximants. We also discuss stability and passivity of linear dynamical systems.

2.1 State-space description

We consider m-input p-output time-invariant linear dynamical systems given by a *state-space description* of the form

$$E\frac{\mathrm{d}x}{\mathrm{d}t} = Ax + Bu(t), \tag{4}$$

$$y(t) = C^T x(t) + Du(t), \tag{5}$$

together with suitable initial conditions. Here, A, $E \in \mathbb{R}^{N\times N}$, $B \in \mathbb{R}^{N\times m}$, $C \in \mathbb{R}^{N\times p}$, and $D \in \mathbb{R}^{p\times m}$ are given matrices, $x(t) \in \mathbb{R}^N$ is the vector of state variables, $u(t) \in \mathbb{R}^m$ is the vector of inputs, $y(t) \in \mathbb{R}^p$ the vector of outputs, N is the state-space dimension, and m and p are the number of inputs and outputs, respectively. Note that systems of the form (2) and (3) are just a special case of (4) and (5) with $D = 0$.

The linear system (4) and (5) is called *regular* if the matrix E in (4) is nonsingular, and it is called *singular* or a *descriptor system* if E is singular. Note that, in the regular case, the linear system (4) and (5) can always be re-written as

$$\begin{aligned}\frac{\mathrm{d}x}{\mathrm{d}t} &= \left(E^{-1}A\right)x + \left(E^{-1}B\right)u(t),\\ y(t) &= C^T x(t) + Du(t),\end{aligned}$$

which is just a system (4) and (5) with $E = I$.

The linear dynamical systems arising in circuit simulation are descriptor systems in general. Therefore, in the following, we allow $E \in \mathbb{R}^{N\times N}$ to be a general, possibly singular, matrix. The only assumption on the matrices A, $E \in \mathbb{R}^{N\times N}$ in (4) is that the matrix pencil $A - sE$ is *regular*, i.e., the matrix $A - sE$ is singular for only finitely many values of $s \in \mathbb{C}$.

In the case of singular E, equation (4) represents a system of DAEs. Solving DAEs is significantly more complex than solving systems of ordinary differential equations (ODEs). Moreover, there are constraints on the possible initial conditions that can be imposed on the solutions of (4). For a detailed discussion of DAEs and the structure of their solutions, we refer the reader to [13, 14, 18, 57]. Here, we only present a brief glimpse of the issues arising in DAEs.

We start by bringing the matrices A and E in (4) to a certain normal form. For any regular pencil $A - sE$, there exist nonsingular matrices P and Q such that

$$P\,(A - sE)\,Q = \begin{bmatrix} A^{(1)} - sI & 0 \\ 0 & I - sJ \end{bmatrix}, \tag{6}$$

where the submatrix J is nilpotent. The matrix pencil on the right-hand side of (6) is called the *Weierstrass form* of $A - sE$. Assuming that the matrices A and E in (4) are already in Weierstrass form, the system (4) can be decoupled as follows:

$$\frac{\mathrm{d}x^{(1)}}{\mathrm{d}t} = A^{(1)}x^{(1)} + B^{(1)}u(t), \tag{7}$$

$$J\frac{\mathrm{d}x^{(2)}}{\mathrm{d}t} = x^{(2)} + B^{(2)}u(t). \tag{8}$$

The first subsystem, (7), is just a system of ODEs. Thus for any given initial condition $x^{(1)}(0) = \hat{x}^{(1)}$, there exits a unique solution of (7). Moreover, the so-called *free-response* of (7), i.e., the solutions $x(t)$ for $t \geq 0$ when $u \equiv 0$, consists of combinations of exponential modes at the eigenvalues of the matrix $A^{(1)}$. Note that, in view of (6), the eigenvalues of $A^{(1)}$ are just the finite eigenvalues of the pencil $A - sE$. The solutions of the second subsystem, (8), however, are of quite different nature. In particular, the free-response of (8) consists of $k_i - 1$ independent impulsive motions for each $k_i \times k_i$ Jordan block of the matrix J; see [57].

For example, consider the case that the nilpotent matrix J in (8) is a single $k \times k$ Jordan block, i.e.,

$$J = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 1 \\ 0 & \cdots & \cdots & 0 & 0 \end{bmatrix} \in \mathbb{R}^{k\times k}.$$

The k components of the free-response $x^{(2)}(t)$ of (8) are then given by

$$
\begin{aligned}
x_1^{(2)}(t) &= -x_2^{(2)}(0-)\delta(t) - x_3^{(2)}(0-)\delta^{(1)}(t) - \cdots - x_k^{(2)}(0-)\delta^{(k-2)}(t), \\
x_2^{(2)}(t) &= -x_3^{(2)}(0-)\delta(t) - x_4^{(2)}(0-)\delta^{(1)}(t) - \cdots - x_k^{(2)}(0-)\delta^{(k-3)}(t), \\
\vdots &= \vdots \\
x_{k-1}^{(2)}(t) &= -x_k^{(2)}(0-)\delta(t), \\
x_k^{(2)}(t) &= 0.
\end{aligned}
$$

Here, $\delta(t)$ is the delta function and $\delta^{(i)}(t)$ is its i-th derivative. Moreover, $x_i^{(2)}(0-)$, $i = 2, 3, \ldots, k$, are the components of the initial conditions that can be imposed on (7). Note that there are only $k-1$ degrees of freedom for the initial condition and that it is not possible to prescribe $x_1^{(2)}(0-)$. In particular, the free-response of (8) corresponding to an 1×1 Jordan blocks of J is just the zero solution, and there is no degree of freedom for the selection of an initial value corresponding to that block.

Finally, we remark that, in view of (6), the eigenvalues of the matrix pencil $A - sE$ corresponding to the subsystem (8) are just the infinite eigenvalues of $A - sE$.

2.2 Reduced-order models and transfer functions

The basic idea of reduced-order modeling is to replace a given system by a system of the same type, but with smaller state-space dimension. Thus, a *reduced-order model* of state-space dimension n of a given linear dynamical system (4) and (5) of dimension N is a system of the form

$$E_n \frac{\mathrm{d}z}{\mathrm{d}t} = A_n z + B_n u(t), \tag{9}$$

$$y(t) = C_n^T z(t) + D_n u(t), \tag{10}$$

where A_n, $E_n \in \mathbb{R}^{n \times n}$, $B_n \in \mathbb{R}^{n \times m}$, $C_n \in \mathbb{R}^{n \times p}$, $D_n \in \mathbb{R}^{p \times m}$, and $n < N$.

The challenge then is to choose the matrices A_n, E_n, B_n, C_n, and D_n in (9) and (10) such that the reduced-order model in some sense approximates the original system. One possible measure of the approximation quality of a reduced-order model is based on the concept of transfer function.

If we assume zero initial conditions, then by applying the Laplace transform to the original system (4) and (5), we obtain the following algebraic equations:

$$
\begin{aligned}
sEX(s) &= AX(s) + BU(s), \\
Y(s) &= C^T X(s) + DU(s).
\end{aligned}
$$

Here, the frequency-domain variables $X(s)$, $U(s)$, and $Y(s)$ are the Laplace transforms of the time-domain variables of $x(t)$, $u(t)$, and $y(t)$, respectively. Note that $s \in \mathbb{C}$. Then, formally eliminating $X(s)$ in the above equations, we arrive at the frequency-domain input-output relation $Y(s) = H(s)U(s)$. Here,

$$H(s) := D + C^T \left(sE - A\right)^{-1} B, \quad s \in \mathbb{C}, \tag{11}$$

is the so-called *transfer function* of the system (4) and (5). Note that

$$H \,:\, \mathbb{C} \mapsto \left(\mathbb{C} \cup \infty\right)^{p \times m}, \tag{12}$$

is an $p \times m$-matrix-valued rational function.

Similarly, the transfer function, H_n, of the reduced-order model (9) and (10) is given by

$$H_n(s) := D_n + C_n^T \left(sE_n - A_n\right)^{-1} B_n, \quad s \in \mathbb{C}. \tag{13}$$

Note that H_n is a also an $p \times m$-matrix-valued rational function.

2.3 Padé and Padé-type models

The concept of transfer functions allows to define reduced-order models by means of Padé or Padé-type approximation.

Let $s_0 \in \mathbb{C}$ be any point such that s_0 is not a pole of the transfer function H given by (11). In practice, the point s_0 is chosen such that it is in some sense close to the frequency range of interest. We remark that the frequency range of interest is usually a subset of the imaginary axis in the complex s-plane. Since s_0 is not a pole of H, the function H admits the Taylor expansion

$$H(s) = M_0 + M_1\,(s - s_0) + M_2\,(s - s_0)^2 + \cdots + M_j\,(s - s_0)^j + \cdots \tag{14}$$

about s_0. The coefficients M_j, $j = 0, 1, \ldots$, in (14) are called the *moments* of H about the expansion point s_0. Note that the M_j's are $p \times m$ matrices.

A reduced-order model (9) and (10) of state-space dimension n is called an *n-th Padé model* (at the expansion point s_0) of the original system (4) and (5) if the Taylor expansions about s_0 of the transfer functions H and H_n of the original system and the reduced-order system agree in as many leading terms as possible, i.e.,

$$H(s) = H_n(s) + \mathcal{O}\big((s - s_0)^{q(n)}\big), \tag{15}$$

where $q(n)$ is as large as possible. In [28, 29], it was shown that

$$q(n) \geq \left\lfloor \frac{n}{m} \right\rfloor + \left\lfloor \frac{n}{p} \right\rfloor,$$

with equality in the "generic" case. The meaning of "generic" will be described more precisely in Section 3.2.

Even though Padé models are defined via the local approximation property (15), in practice, they usually are excellent approximations over large frequency ranges. The following single-input single-output example illustrates this statement. The example is a circuit resulting from the so-called PEEC discretization [50] of an electromagnetic problem. The circuit is an RCL network consisting of 2100 capacitors, 172 inductors, 6990 inductive couplings, and a single resistive source that drives the circuit. Modified nodal analysis is used to set up the circuit equations, resulting in a linear dynamical system of dimension $N = 306$. In turns out that a Padé model of dimension $n = 60$ is sufficient to produces an almost exact transfer function in the relevant frequency range $s = 2\pi\,\mathrm{i}\,\omega$, $0 \leq \omega \leq 5 \times 10^9$. The corresponding curves for $|H(s)|$ and $|H_{60}(s)|$ are shown in Figure 1.

It is very tempting to compute Padé models directly via the definition (15). More precisely, one would first explicitly generate the $q(n)$ moments $M_0, M_1, \ldots, M_{q(n)-1}$, and then compute H_n and the system matrices in the reduced-order model (9) and (10) from these moments. However, computing Padé models directly from the moments is extremely ill-conditioned, and consequently, such a procedure is not viable; we refer the reader to [26, 27] for a detailed discussion and numerical examples.

The preferred way to compute Padé models is to use Krylov-subspace techniques, such as a suitable Lanczos-type process, as we will describe in Section 3. This becomes possible after the transfer function (11) is rewritten in terms of a single matrix M, instead of the two matrices A and E. To this end, let

$$A - s_0 E = F_1 F_2, \quad \text{where} \quad F_1,\, F_2 \in \mathbb{C}^{N \times N}, \tag{16}$$

be any factorization of $A - s_0 E$. For example, the matrices $A - s_0 E$ arising in circuit simulation are large, but sparse, and are such that a sparse LU factorization is feasible. In this case, the matrices F_1 and F_2 in (16) are the lower and upper triangular factors, possibly with rows and columns permuted due to pivoting, of such a sparse LU factorization of $A - s_0 E$. Using (16), the transfer function (11) can be rewritten as follows:

$$\begin{aligned} H(s) &= D + C^T\,(sE - A)^{-1}\,B \\ &= D - C^T\,(A - s_0 E - (s - s_0)E)^{-1}\,B \\ &= D - L^T\,(I - (s - s_0)M)^{-1}\,R, \end{aligned} \tag{17}$$

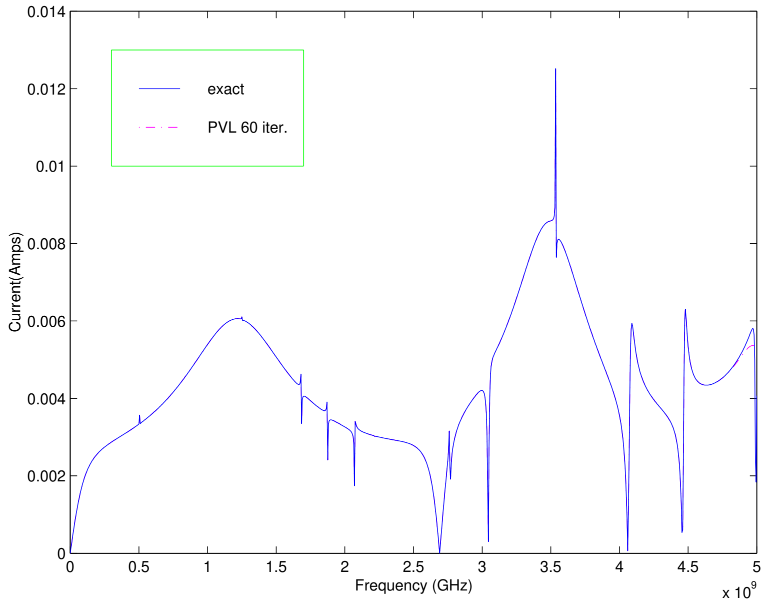

Figure 1: The PEEC transfer function, exact and Padé model of dimension $n = 60$

where

$$M := F_1^{-1} E F_2^{-1}, \quad R := F_1^{-1} B, \quad \text{and} \quad L := F_2^{-T} C. \tag{18}$$

Note that (17) only involves one $N \times N$ matrix, namely M, instead of the two $N \times N$ matrices A and E in (11). This allows to apply Krylov-subspace methods to the single matrix M, with the $N \times m$ matrix R and the $N \times p$ matrix L as blocks of right and left starting vectors.

While Padé models often provide very good approximations in frequency domain, they also have undesirable properties. In particular, in general, Padé models do not preserve stability or passivity of the original system. However, by relaxing the Padé-approximation property (15), it is often possibly to obtain stable or passive models. More precisely, we call a reduced-order model (9) and (10) of state-space dimension n an *n-th Padé-type model* (at the expansion point s_0) of the original system (4) and (5) if the Taylor expansions about s_0 of the transfer functions H and H_n of the original system and the reduced-order system agree in a number of leading terms, i.e.,

$$H(s) = H_n(s) + \mathcal{O}\big((s - s_0)^{q'}\big), \tag{19}$$

where $1 \le q' < q(n)$.

2.4 Stability

An important property of linear dynamical systems is stability. An actual physical system needs to be stable in order to function properly. If a linear dynamical system (4) and (5) is used as a description of such a physical system, then clearly, it should also be stable. Moreover, when (4) and (5) is replaced by a reduced-order model that is then used in a time-domain analysis, the reduced-order model also needs to be stable.

In this subsection, we present a brief discussion of stability of linear descriptor systems. For a more general survey of the various concepts of stability of dynamical systems, we refer the reader to [4, 59].

A descriptor system of the form (4) and (5) is said to be *stable* if its free-response, i.e., the solutions

$x(t)$, $t \geq 0$, of

$$E\frac{\mathrm{d}x}{\mathrm{d}t} = Ax,$$
$$x(0) = x_0,$$

remain bounded as $t \to \infty$ for any possible initial vector x_0. Recall from the discussion in Section 2.1 that for singular E, there are certain restrictions on the possible initial vectors x_0.

Stability can easily be characterized in terms of the finite eigenvalues of the matrix pencil $A - sE$; see, e.g., [43]. More precisely, we have the following theorem.

Theorem 1 *The descriptor system* (4) *and* (5) *is stable if, and only if, the following two conditions are satisfied* :

(i) *All finite eigenvalues* $\lambda \in \mathbb{C}$ *of the matrix pencil* $A - sE$ *satisfy* $\operatorname{Re}\lambda \leq 0$*;*

(ii) *All finite eigenvalues* λ *of the matrix pencil* $A - sE$ *with* $\operatorname{Re}\lambda = 0$ *are simple.*

We stress that, in view of Theorem 1, the infinite eigenvalues of the matrix pencil $A - sE$ have no effect on stability. The reason is that these infinite eigenvalues result only in impulsive motions, which go to zero as $t \to \infty$.

Recall that the transfer function H of the descriptor system (4) and (5) is of the form

$$H(s) = D + C^T \left(sE - A\right)^{-1} B, \tag{20}$$

$$\text{where} \quad A,\, E \in \mathbb{R}^{N \times N},\ B \in \mathbb{R}^{N \times m},\ C \in \mathbb{R}^{N \times m},\ \text{and } D \in \mathbb{R}^{p \times m}. \tag{21}$$

Note that any pole of H is necessarily an eigenvalue of the matrix pencil $A - sE$. Hence, it is tempting to determine stability via the poles of H. However, in general, not all eigenvalues of $A - sE$ are poles of H. For example, consider the following system

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t),$$
$$y(t) = \begin{bmatrix} 1 & 1 \end{bmatrix} x(t),$$

which is taken from [4, Page 128]. The pencil associated with this system is

$$A - sI = \begin{bmatrix} 1-s & 0 \\ 0 & -1-s \end{bmatrix}.$$

Its eigenvalues are ± 1, and hence this system is unstable. The transfer function $H(s) = 1/(s+1)$, however, only has the "stable" pole -1. Therefore, checking conditions (i) and (ii) of Theorem 1 only for the poles of H is, in general, not enough to guarantee stability. In order to infer stability of the system (4) and (5) from the poles of its transfer function, one needs an additional condition, which we formulate next.

Let H be a given $m \times p$-matrix-valued rational function. Any representation of H of the form (20) with matrices (21) is called a *realization* of H. Furthermore, a realization (20) of H is said to be *minimal* if the dimension N of the matrices (21) is as small as possible. We will also say that the state-space description (4) and (5) is a minimal realization if its transfer function (21) is a minimal realization.

The following theorem is the well-known characterization of minimal realizations in terms of conditions on the matrices (21); see, e.g., [57]. We also refer the reader to the related results on controllability, observability, and minimal realizations of descriptor systems given in [18, Chapter 2].

Theorem 2 *Let* H *be a* $m \times p$*-matrix-valued rational function given by a realization* (20)*. Then,* (20) *is a minimal realization of* H *if, and only if, the matrices* (21) *satisfy the following five conditions* :

(i) $\operatorname{rank}\begin{bmatrix} A - sE & B \end{bmatrix} = N$ *for all* $s \in \mathbb{C}$;
(Finite controllability)

(ii) $\operatorname{rank}\begin{bmatrix} E & B \end{bmatrix} = N$;
(Infinite controllability)

(iii) $\operatorname{rank}\begin{bmatrix} A^T - sE^T & C \end{bmatrix} = N$ *for all* $s \in \mathbb{C}$;
(Finite observability)

(iv) $\operatorname{rank}\begin{bmatrix} E^T & C \end{bmatrix} = N$;
(Infinite observability)

(v) $A \ker(E) \subseteq \operatorname{Im}(E)$.
(Absence of nondynamic modes)

For descriptor systems given by a minimal realization, stability can indeed be checked via the poles of its transfer function.

Theorem 3 *Let* (4) *and* (5) *be a minimal realization of a descriptor system, and let* H *be its transfer function* (20)*. Then, the descriptor system* (4) *and* (5) *is stable if, and only if, all finite poles* s_i *of* H *satisfy* $\operatorname{Re} s_i \leq 0$ *and any pole with* $\operatorname{Re} s_i = 0$ *is simple.*

2.5 Passivity

In circuit simulation, reduced-order modeling is often applied to large passive linear subcircuits, such as RCL networks consisting of only resistors, inductors, and capacitors. When reduced-order models of such subcircuits are used within a simulation of the whole circuit, stability of the overall simulation can only be guaranteed if the reduced-order models preserve the passivity of the original subcircuits; see, e.g., [15, 49]. Therefore, it is important to have techniques to check passivity of a given reduced-order model.

Roughly speaking, a system is *passive* if it does not generate energy. For descriptor systems of the form (4) and (5), passivity is equivalent to positive realness of the transfer function. Moreover, such systems can only be passive if they have identical numbers of inputs and outputs. Thus, for the remainder of this subsection, we assume that $m = p$. Then, a system described by (4) and (5) is passive, i.e., it does not generate energy, if, and only if, its transfer function (20) is *positive real*; see, e.g., [4]. A precise definition of positive realness is as follows.

Definition 1 *An* $m \times m$*-matrix-valued function* $H : \mathbb{C} \mapsto (\mathbb{C} \cup \infty)^{m \times m}$ *is called positive real if the following three conditions are satisfied*:

(i) H *is analytic in* $\mathbb{C}_+ := \{\, s \in \mathbb{C} \mid \operatorname{Re} s > 0 \,\}$;

(ii) $H(\overline{s}) = \overline{H(s)}$ *for all* $s \in \mathbb{C}$;

(iii) $H(s) + (H(s))^H \succeq 0$ *for all* $s \in \mathbb{C}_+$.

In Definition 1 and in the sequel, the notation $M \succeq 0$ means that the matrix M is Hermitian positive semi-definite. Similarly, $M \preceq 0$ means that M is Hermitian negative semi-definite.

For transfer functions H of the form (20), condition (ii) of Definition 1 is always satisfied since the matrices (21) are assumed to be real. Furthermore, condition (i) simply means that H cannot have poles in $\mathbb{C}_+$, and this can be checked easily. For the special case $m = 1$ of scalar-valued functions H, condition (iii) states that the real part of $H(s)$ is nonnegative for all s with nonnegative real part. In order to check this condition, it is sufficient to show that the real part of $H(s)$ is nonnegative for all purely imaginary s. This can be done by means of relatively elementary means. For example, in [9], a procedure based on eigenvalue computations is proposed. For the general matrix-valued case, $m \geq 1$, however, checking condition (iii) is much more involved. One possibility is to employ a suitable extension of the classical positive real lemma [3], [4, Chapter 5], [60, Section 13.5] that characterizes positive realness of regular linear systems via the solvability of certain linear matrix inequalities (LMIs). Such a version of the positive real lemma for general descriptor systems is stated in Theorem 4 below.

We remark that any matrix-valued rational function H has an expansion about $s = \infty$ of the form

$$H(s) = \sum_{j=-\infty}^{j_0} M_j s^j, \tag{22}$$

where $j_0 \geq 0$ is an integer. Moreover, the function H has a pole at $s = \infty$ if, and only if, $j_0 \geq 1$ and $M_{j_0} \neq 0$ in (22).

The positive real lemma for descriptor systems can now be stated as follows.

Theorem 4 (Positive real lemma for descriptor systems [38])
Let H be a real $m \times m$-matrix-valued rational function of the form (20) *with matrices* (21).

a) (Sufficient condition)
If the LMIs

$$\begin{bmatrix} A^T X + X^T A & X^T B - C \\ B^T X - C^T & -D - D^T \end{bmatrix} \preceq 0 \quad \textit{and} \quad E^T X = X^T E \succeq 0 \tag{23}$$

have a solution $X \in \mathbb{R}^{N \times N}$, then H is positive real.

b) (Necessary condition)
Suppose that (20) *is a minimal realization of H and that the matrix M_0 in the expansion* (22) *satisfies*

$$(D - M_0) + (D - M_0)^T \succeq 0. \tag{24}$$

If H is positive real, then there exists a solution $X \in \mathbb{R}^{N \times N}$ of the LMIs (23).

The result of Theorem 4 allows to check positive realness by solving the semi-definite programming problems of the form (23). Note that there are N^2 unknowns in (23), namely the entries of the $N \times N$ matrix X. Problems of the form (23) can be tackled with interior-point methods; see, e.g., [12, 39]. However, the computational complexity of these methods grows quickly with N, and thus, these methods are viable only for rather small values of N.

For the special case $E = I$, the result of Theorem 4 is just the classical positive real lemma [3], [4, Chapter 5], [60, Section 13.5]. In this case, (23) reduces to the problem of finding a symmetric positive semi-definite matrix $X \in \mathbb{R}^{N \times N}$ such that

$$\begin{bmatrix} A^T X + XA & XB - C \\ B^T X - C^T & -D - D^T \end{bmatrix} \preceq 0.$$

Moreover, if $E = I$, the condition (24) is always satisfied, since in this case $M_0 = 0$ and $D + D^T \succeq 0$.

2.6 Linear RCL subcircuits

In circuit simulation, an important special case of passive circuits is linear subcircuits that consist of only resistors, inductors, and capacitors. Such linear RCL subcircuits arise in the modeling of a circuit's interconnect and package; see, e.g., [36, 37, 41, 48].

The equations describing linear RLC subcircuits are of the form (4) and (5) with $D = 0$ and $m = p$. Furthermore, the equations can be formulated such that the matrices $A, E \in \mathbb{R}^{N \times N}$ in (4) are symmetric and exhibit a block structure; see [34, 37]. More precisely, we have

$$A = A^T = \begin{bmatrix} -A_{11} & A_{12} \\ A_{12}^T & 0 \end{bmatrix} \quad \text{and} \quad E = E^T = \begin{bmatrix} E_{11} & 0 \\ 0 & -E_{22} \end{bmatrix}, \tag{25}$$

where the submatrices A_{11}, $E_{11} \in \mathbb{R}^{N_1 \times N_1}$ and $E_{22} \in \mathbb{R}^{N_2 \times N_2}$ are symmetric positive semi-definite, and $N = N_1 + N_2$. Note that, except for the special case $N_2 = 0$, the matrices A and E are indefinite. The special case $N_2 = 0$ arises for RC subcircuits that contain only resistors and capacitors, but no inductors.

If the RCL subcircuit is viewed as an m-terminal component with $m = p$ inputs and outputs, then the matrices B and C in (4) and (5) are identical and of the form

$$B = C = \begin{bmatrix} B_1 \\ 0 \end{bmatrix} \quad \text{with} \quad B_1 \in \mathbb{R}^{N_1 \times m}. \tag{26}$$

In view of (25) and (26), the transfer function of such an m-terminal RLC subcircuit is given by

$$H(s) = B^T \left(sE - A\right)^{-1} B, \quad \text{where} \quad A = A^T, \quad E = E^T. \tag{27}$$

We call a transfer function H *symmetric* if it is of the form (27) with real matrices A, E, and B.

We will also use the following nonsymmetric formulation of (27). Let J be the block matrix

$$J = \begin{bmatrix} I_{N_1} & 0 \\ 0 & -I_{N_2} \end{bmatrix}, \tag{28}$$

where I_{N_1} and I_{N_2} is the $N_1 \times N_1$ and $N_2 \times N_2$ identity matrix, respectively.

Note that, by (26) and (28), we have $B = JB$. Using this relation, as well as (25), we can rewrite (27) as follows:

$$H(s) = B^T \left(s\tilde{E} - \tilde{A}\right)^{-1} B, \quad \text{where} \quad \tilde{A} = \begin{bmatrix} -A_{11} & A_{12} \\ -A_{12}^T & 0 \end{bmatrix}, \quad \tilde{E} = \begin{bmatrix} E_{11} & 0 \\ 0 & E_{22} \end{bmatrix}. \tag{29}$$

In this formulation, the matrix $\tilde{A}$ is no longer symmetric, but now

$$\tilde{A} + \tilde{A}^T \preceq 0 \quad \text{and} \quad \tilde{E} \succeq 0. \tag{30}$$

It turns out that the properties are the key to ensure positive realness. Indeed, in [33, Theorem 13], we established the following result.

Theorem 5 *Let $\tilde{A}$, $\tilde{E} \in \mathbb{R}^{N \times N}$, and $B \in \mathbb{R}^{N \times m}$. Assume that $\tilde{A}$ and $\tilde{E}$ satisfy* (30)*, and that the matrix pencil $\tilde{A} - s\tilde{E}$ is regular. Then, the $m \times m$-matrix-valued function*

$$H(s) = B^T \left(s\tilde{E} - \tilde{A}\right)^{-1} B$$

is positive real.

3 Krylov-subspace techniques

In this section, we discuss the use of Krylov-subspace methods for the construction of Padé and Padé-type reduced-order models of time-invariant linear dynamical systems.

3.1 Block Krylov subspaces

We consider general descriptor systems of the form (4) and (5). The key to using Krylov-subspace techniques for reduced-order modeling of such systems is to first replace the matrix pair A and E by a single matrix M. To this end, let $s_0 \in \mathbb{C}$ be any given point such that the matrix $A - s_0 E$ is nonsingular. Then, with M, R, and L denoting the matrices defined in (18), the linear system (4) and (5) can be rewritten in the following form:

$$M \frac{\mathrm{d}x}{\mathrm{d}t} = (I + s_0 M)\, x + Ru(t), \tag{31}$$

$$y(t) = L^T x(t) + Du(t). \tag{32}$$

Note that $M \in \mathbb{C}^{N \times N}$, $R \in \mathbb{C}^{N \times m}$, and $L \in \mathbb{C}^{N \times p}$, where N is the state-space dimension of the system, m is the number of inputs, and p is the number of outputs.

The transfer function H of the rewritten system (31) and (32) is given by (17). By expanding (17) about s_0, we obtain

$$H(s) = D - \sum_{j=0}^{\infty} L^T M^j R\,(s - s_0)^j. \tag{33}$$

Recall from Section 2.3 that Padé and Padé-type reduced-order models are defined via the leading coefficients of an expansion of H about s_0. In view of (33), the j-th coefficient of such an expansion can be expressed as follows:

$$-L^T M^j R = -\big((M^{j-i})^T L\big)^T \big(M^i R\big), \quad i = 0, 1, \dots, j. \tag{34}$$

Notice that the factors on the right-hand side of (34) are blocks of the *right* and *left block Krylov matrices*

$$\begin{bmatrix} R & MR & M^2R & \cdots & M^iR & \cdots \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} L & M^TL & \left(M^T\right)^2 L & \cdots & \left(M^T\right)^k L & \cdots \end{bmatrix}, \tag{35}$$

respectively. As a result, all the information needed to generate Padé and Padé-type reduced-order models is contained in the block Krylov matrices (35). However, simply computing the blocks M^iR and $\left(M^T\right)^i L$ in (35) and then generating the leading coefficients of the expansion (33) from these blocks is not a viable numerical procedure. The reason is that, in finite-precision arithmetic, as i increases, the blocks M^iR and $\left(M^T\right)^i L$ quickly contain only information about the eigenspaces of the dominant eigenvalue of M. Instead, one needs to employ suitable Krylov-subspace methods that generate numerically better basis vectors for the subspaces associated with the block Krylov matrices (35).

Next, we give a formal definition of the subspaces induced by (35). Note that each block M^iR consists of m column vectors of length N. By scanning these column vectors of the right block Krylov matrix in (35) from left to right and by deleting any column that is linearly dependent on columns to its left, we obtain the *deflated* right block Krylov matrix

$$\begin{bmatrix} R_1 & MR_2 & M^2R_3 & \cdots & M^{i_{\max}-1}R_{i_{\max}} \end{bmatrix}. \tag{36}$$

This process of detecting and deleting the linearly dependent columns is called *exact deflation*. We remark that the matrix (36) is finite, since at most N of the column vectors can be linearly independent. Furthermore, a column $M^i r$ being linearly dependent on columns to its left in (35) implies that any column $M^{i'} r$, $i' \geq i$, is linearly dependent on columns to its right. Therefore, in (36), for each $i = 1, 2, \dots, i_{\max}$, the matrix R_i is a submatrix of R_{i-1}, where, for $i = 1$, we set $R_0 = R$.

Let m_i denote the number of columns of R_i. The matrix (36) thus has

$$n_{\max}^{(\mathrm{r})} := \leq m_1 + m_2 + \cdots + m_{i_{\max}},$$

columns. For each integer n with $1 \leq n \leq n_{\max}^{(\mathrm{r})}$, we define the *$n$-th right block Krylov subspace* $\mathcal{K}_n(M, R)$ (induced by M and R) as the subspace spanned by the first n columns of the deflated right block Krylov matrix (36).

Analogously, by deleting the linearly independent columns of the left block Krylov matrix in (35), we obtain a deflated left block Krylov matrix of the form

$$\begin{bmatrix} L_1 & M^TL_2 & \left(M^T\right)^2 L_3 & \cdots & \left(M^T\right)^{i_{\max}-1} L_{k_{\max}} \end{bmatrix}. \tag{37}$$

Let $n_{\max}^{(\mathrm{l})}$ be the number of columns of the matrix (37). Then for each integer n with $1 \leq n \leq n_{\max}^{(\mathrm{l})}$, we define the *$n$-th left block Krylov subspace* $\mathcal{K}_n(M^T, L)$ (induced by M^T and L) as the subspace spanned by the first n columns of the deflated left block Krylov matrix (37).

For a more detailed discussion of block Krylov subspaces and deflation, we refer the reader to [1, 33].

3.2 Approaches based on Lanczos and Lanczos-type Methods

In this section, we discuss reduced-order modeling approaches that employ Lanczos and Lanczos-type methods for the construction of suitable basis vectors for the right and left block Krylov subspaces $\mathcal{K}_n(M, R)$ and $\mathcal{K}_n(M^T, L)$.

The MPVL algorithm

For the special case $m = p = 1$ of single-input single-output linear dynamical systems, each of the "blocks" R and L only consists of a single vector, say r and l, and $\mathcal{K}_n(M, r)$ and $\mathcal{K}_n(M^T, l)$ are just the standard n-th right and left Krylov subspaces induced by single vectors. The classical Lanczos process [42] is a well-known procedure for computing two sets of bi-orthogonal basis vectors for $\mathcal{K}_n(M, r)$ and $\mathcal{K}_n(M^T, l)$. Moreover, these vectors are generated by means of three-term recurrences the coefficients of which define a tridiagonal matrix T_n. It turns out that T_n contains all the information that is needed to set up an n-th Padé reduced-order model of a given single-input single-output time-invariant linear dynamical system. The associated computational procedure is called the *Padé via Lanczos* (PVL) algorithm [26, 27].

Here, we describe in some detail an extension of the PVL algorithm to the case of general m-input p-output time-invariant linear dynamical systems. The underlying block Krylov subspace method is the *nonsymmetric band Lanczos algorithm* [32] for constructing two sets of right and left Lanczos vectors

$$v_1, v_2, \ldots, v_n \quad \text{and} \quad w_1, w_2, \ldots, w_n, \tag{38}$$

respectively. These vectors span the n-th right and left block Krylov subspaces (induced by M and R, and M^T and L, respectively):

$$\operatorname{span}\{\, v_1, v_2, \ldots, v_n \,\} = \mathcal{K}_n(M, R) \quad \text{and} \quad \operatorname{span}\{\, w_1, w_2, \ldots, w_n \,\} = \mathcal{K}_n(M^T, L). \tag{39}$$

Moreover, the vectors (38) are constructed to be bi-orthogonal:

$$w_j^T v_k = \begin{cases} 0 & \text{if } j \neq k, \\ \delta_j & \text{if } j = k, \end{cases} \quad \text{for all} \quad j, k = 1, 2, \ldots, n. \tag{40}$$

It turns out that the Lanczos vectors (38) can be constructed by means of recurrence relations of length at most $m + p + 1$. The recurrence coefficients for the first n right Lanczos vectors define an $n \times n$ matrix $T_n^{(\mathrm{pr})}$ that is "essentially" a band matrix with total bandwidth $m + p + 1$. Similarly, the recurrence coefficients for the first n left Lanczos vectors define an $n \times n$ band matrix $\widetilde{T}_n^{(\mathrm{pr})}$ with total bandwidth $m + p + 1$. For a more detailed discussion of the structure of $T_n^{(\mathrm{pr})}$ and $\widetilde{T}_n^{(\mathrm{pr})}$, we refer the reader to [1, 32].

Algorithm 1 below gives a complete description of the numerical procedure that generates the Lanczos vectors (38) with properties (39) and (40). In order to obtain a Padé reduced-order model based on this algorithm, one does not need the Lanczos vectors themselves, but rather the matrix of right recurrence coefficients $T_n^{(\mathrm{pr})}$, the matrices $\rho_n^{(\mathrm{pr})}$ and $\eta_n^{(\mathrm{pr})}$ that contain the recurrence coefficients from processing the starting blocks R and L, respectively, and the diagonal matrix

$$\Delta_n = \operatorname{diag}(\delta_1, \delta_2, \ldots, \delta_n),$$

whose diagonal entries are the δ_j's from (40). The following algorithm produces the matrices $T_n^{(\mathrm{pr})}$, $\rho_n^{(\mathrm{pr})}$, $\eta_n^{(\mathrm{pr})}$, and Δ_n as output.

Algorithm 1 (Nonsymmetric band Lanczos algorithm)

INPUT: *A matrix* $M \in \mathbb{C}^{N \times N}$*;*

A block of m *right starting vectors* $R = [r_1 \; r_2 \; \cdots \; r_m] \in \mathbb{C}^{N \times m}$*;*

A block of p *left starting vectors* $L = [l_1 \; l_2 \; \cdots \; l_p] \in \mathbb{C}^{N \times p}$*.*

OUTPUT: *The* $n \times n$ *Lanczos matrix* $T_n^{(\mathrm{pr})}$*, and the matrices* $\rho_n^{(\mathrm{pr})}$*,* $\eta_n^{(\mathrm{pr})}$*, and* Δ_n*.*

0) *For* $k = 1, 2, \ldots, m$, *set* $\hat{v}_k = r_k$.
For $k = 1, 2, \ldots, p$, *set* $\hat{w}_k = l_k$.
Set $m_c = m$, $p_c = p$, *and* $\mathcal{I}_v = \mathcal{I}_w = \emptyset$.

For $n = 1, 2, \ldots,$ *until convergence or* $m_c = 0$ *or* $p_c = 0$ *or* $\delta_n = 0$ *do*:

1) (If necessary, deflate $\hat{v}_n$.)
Compute $\|\hat{v}_n\|_2$.
Decide if $\hat{v}_n$ *should be deflated. If yes, do the following*:

   a) *Set* $\hat{v}^{\text{defl}}_{n-m_c} = \hat{v}_n$ *and store this vector. Set* $\mathcal{I}_v = \mathcal{I}_v \cup \{ n - m_c \}$.
   b) *Set* $m_c = m_c - 1$. *If* $m_c = 0$, *set* $n = n - 1$ *and stop.*
   c) *For* $k = n, n+1, \ldots, n + m_c - 1$, *set* $\hat{v}_k = \hat{v}_{k+1}$.
   d) *Repeat all of Step* 1).

2) (If necessary, deflate $\hat{w}_n$.)
Compute $\|\hat{w}_n\|_2$.
Decide if $\hat{w}_n$ *should be deflated. If yes, do the following*:

   a) *Set* $\hat{w}^{\text{defl}}_{n-p_c} = \hat{w}_n$ *and store this vector. Set* $\mathcal{I}_w = \mathcal{I}_w \cup \{ n - p_c \}$.
   b) *Set* $p_c = p_c - 1$. *If* $p_c = 0$, *set* $n = n - 1$ *and stop.*
   c) *For* $k = n, n+1, \ldots, n + p_c - 1$, *set* $\hat{w}_k = \hat{w}_{k+1}$.
   d) *Repeat all of Step* 2).

3) (Normalize $\hat{v}_n$ and $\hat{w}_n$ to obtain v_n and w_n.)
Set

$$t_{n,n-m_c} = \|\hat{v}_n\|_2\,, \quad \tilde{t}_{n,n-p_c} = \|\hat{w}_n\|_2\,,$$
$$v_n = \frac{\hat{v}_n}{t_{n,n-m_c}}, \quad \textit{and} \quad w_n = \frac{\hat{w}_n}{\tilde{t}_{n,n-p_c}}.$$

4) (Compute δ_n and check for possible breakdown.)
Set $\delta_n = w_n^T v_n$. *If* $\delta_n = 0$, *set* $n = n - 1$ *and stop.*

5) (Orthogonalize the right candidate vectors against w_n.)
For $k = n+1, n+2, \ldots, n + m_c - 1$, *set*

$$t_{n,k-m_c} = \frac{w_n^T \hat{v}_k}{\delta_n} \quad \textit{and} \quad \hat{v}_k = \hat{v}_k - v_n\, t_{n,k-m_c}.$$

6) (Orthogonalize the left candidate vectors against v_n.)
For $k = n+1, n+2, \ldots, n + p_c - 1$, *set*

$$\tilde{t}_{n,k-p_c} = \frac{\hat{w}_k^T v_n}{\delta_n} \quad \textit{and} \quad \hat{w}_k = \hat{w}_k - w_n\, \tilde{t}_{n,k-p_c}.$$

7) (Advance the right block Krylov subspace to get $\hat{v}_{n+m_c}$.)

   a) *Set* $\hat{v}_{n+m_c} = M\, v_n$.
   b) *For* $k \in \mathcal{I}_w$ *(in ascending order), set*

$$\tilde{\sigma} = \left(\hat{w}_k^{\text{defl}}\right)^T v_n, \quad \tilde{t}_{n,k} = \frac{\tilde{\sigma}}{\delta_n},$$

   and, if $k > 0$, *set*

$$t_{k,n} = \frac{\tilde{\sigma}}{\delta_k} \quad \textit{and} \quad \hat{v}_{n+m_c} = \hat{v}_{n+m_c} - v_k\, t_{k,n}.$$

c) *Set* $k_v = \max\{\,1,\, n - p_c\,\}$.

d) *For* $k = k_v, k_v + 1, \ldots, n-1$, *set*

$$t_{k,n} = \tilde{t}_{n,k}\,\frac{\delta_n}{\delta_k} \quad \textit{and} \quad \hat{v}_{n+m_c} = \hat{v}_{n+m_c} - v_k\, t_{k,n}.$$

e) *Set*

$$t_{n,n} = \frac{w_n^T \hat{v}_{n+m_c}}{\delta_n} \quad \textit{and} \quad \hat{v}_{n+m_c} = \hat{v}_{n+m_c} - v_n\, t_{n,n}.$$

8) (Advance the left block Krylov subspace to get $\hat{w}_{n+p_c}$.)

a) *Set* $\hat{w}_{n+p_c} = M^T w_n$.

b) *For* $k \in \mathcal{I}_v$ *(in ascending order), set*

$$\sigma = w_n^T \hat{v}_k^{\mathrm{defl}}, \quad t_{n,k} = \frac{\sigma}{\delta_n},$$

and, if $k > 0$, *set*

$$\tilde{t}_{k,n} = \frac{\sigma}{\delta_k} \quad \textit{and} \quad \hat{w}_{n+p_c} = \hat{w}_{n+p_c} - w_k\, \tilde{t}_{k,n}.$$

c) *Set* $k_w = \max\{\,1,\, n - m_c\,\}$.

d) *For* $k = k_w, k_w + 1, \ldots, n-1$, *set*

$$\tilde{t}_{k,n} = t_{n,k}\,\frac{\delta_n}{\delta_k} \quad \textit{and} \quad \hat{w}_{n+p_c} = \hat{w}_{n+p_c} - w_k\, \tilde{t}_{k,n}.$$

e) *Set*

$$\tilde{t}_{n,n} = t_{n,n} \quad \textit{and} \quad \hat{w}_{n+p_c} = \hat{w}_{n+p_c} - w_n\, \tilde{t}_{n,n}.$$

9) *Set*

$$\begin{aligned}
T_n^{(\mathrm{pr})} &= \big[t_{i,k}\big]_{i,k=1,2,\ldots,n}, \\
\rho_n^{(\mathrm{pr})} &= \big[t_{i,k-m}\big]_{i=1,2,\ldots,n;\, k=1,2,\ldots,k_\rho} \quad \textit{where} \quad k_\rho = m + \min\{\,0,\, n - m_c\,\}, \\
\eta_n^{(\mathrm{pr})} &= \big[\tilde{t}_{i,k-p}\big]_{i=1,2,\ldots,n;\, k=1,2,\ldots,k_\eta} \quad \textit{where} \quad k_\eta = p + \min\{\,0,\, n - p_c\,\}, \\
\Delta_n &= \mathrm{diag}\,(\delta_1, \delta_2, \ldots, \delta_n)\,.
\end{aligned}$$

10) *Check if* n *is large enough. If yes, stop.*

Remark 1 When applied to single starting vectors, i.e., for the special case $m = p = 1$, Algorithm 1 reduces to the classical nonsymmetric Lanczos process [42].

Remark 2 It can be shown that, at step n of Algorithm 1, exact deflation of a vector in the right, respectively left, block Krylov matrix (35) occurs if, and only if, $\hat{v}_n = 0$, respectively $\hat{w}_n = 0$, in Step 1), respectively Step 2). Therefore, to run Algorithm 1 with exact deflation only, one deflates $\hat{v}_n$ if $\|\hat{v}_n\|_2 = 0$ in Step 1), and one deflates $\hat{w}_n$ if $\|\hat{w}_n\|_2 = 0$ in Step 2). In finite-precision arithmetic, however, so-called *inexact deflation* is employed. This means that in Step 1), $\hat{v}_n$ is deflated if $\|\hat{v}_n\|_2 \le \varepsilon$, and in Step 2), $\hat{w}_n$ is deflated if $\|\hat{w}_n\|_2 \le \varepsilon$, where $\varepsilon = \varepsilon(M) > 0$ is a suitably chosen small constant.

Remark 3 The occurrence of $\delta_n = 0$ in Step 4) of Algorithm 1 is called a *breakdown*. In finite-precision arithmetic, in Step 4) one should also check for *near-breakdowns*, i.e., if $\delta_n \approx 0$. In general, it cannot be excluded that breakdowns or near-breakdowns occur, although they are very unlikely. Furthermore, by using so-called *look-ahead* techniques, it is possible to remedy the problem of possible breakdowns or near-breakdowns. For the sake of simplicity, we have stated the band Lanczos algorithm without look-ahead only. A look-ahead version of Algorithm 1 is described in [1].

The *matrix-Padé via Lanczos* (MPVL) algorithm [28, 29] consists of applying Algorithm 1 to the matrices M, R, and L defined in (18), and running it for n steps. The matrices $T_n^{(\mathrm{pr})}$, $\rho_n^{(\mathrm{pr})}$, $\eta_n^{(\mathrm{pr})}$, and Δ_n produced by Algorithm 1 are then used to set up a reduced-order model of the original linear dynamical system (4) and (5) as follows:

$$T_n^{(\mathrm{pr})} \frac{\mathrm{d}z}{\mathrm{d}t} = \left(s_0 T_n^{(\mathrm{pr})} - I\right) z + \rho_n^{(\mathrm{pr})} u(t), \tag{41}$$

$$y(t) = \left(\eta_n^{(\mathrm{pr})}\right)^T \Delta_n z(t) + D u(t). \tag{42}$$

Note that the transfer function of this reduced-order model is given by

$$H_n(s) = D + \left(\eta_n^{(\mathrm{pr})}\right)^T \Delta_n \left(I - (s - s_0) T_n^{(\mathrm{pr})}\right)^{-1} \rho_n^{(\mathrm{pr})}. \tag{43}$$

The reduced-order model (41) and (42) is indeed a matrix-Padé model of the original system.

Theorem 6 (Matrix-Padé model [28, 29])
Suppose that Algorithm 1 is run with exact deflation only and that $n \geq \max\{m, p\}$. Then, the reduced-order model (41) *and* (42) *is a matrix-Padé model of the linear dynamical system* (4) *and* (5). *More precisely, the Taylor expansions about s_0 of the transfer functions, H,* (11) *and , H_n,* (43) *agree in as many leading coefficients as possible, i.e.,*

$$H(s) = H_n(s) + \mathcal{O}\left((s - s_0)^{q(n)}\right),$$

where $q(n)$ is as large as possible. In particular,

$$q(n) \geq \left\lfloor \frac{n}{m} \right\rfloor + \left\lfloor \frac{n}{p} \right\rfloor.$$

A disadvantage of Padé models is that, in general, they do not preserve the stability and possibly passivity of the original linear dynamical system. In part, these problems can be overcome by means of suitable post-processing techniques, such as the ones described in [8, 10]. However, the reduced-order models obtained by post-processing of Padé models are necessarily no longer optimal in the sense of Padé approximation. Furthermore, post-processing techniques are not guaranteed to always result in stable and possibly passive reduced-order models.

For special cases, however, Padé models can be shown to be stable and passive. In particular, this is the case for linear dynamical systems describing RC subcircuits, RL subcircuits, and LC subcircuits; see [11, 34, 36, 37].

Next, we describe the SyMPVL algorithm [34, 36, 37], which is a special version of MPVL tailored to linear RCL subcircuits.

The SyMPVL algorithm

Recall from Section 2.6 that linear RCL subcircuits can be described by linear dynamical systems (4) and (5) with $D = 0$, symmetric matrices A and E of the form (25), and matrices $B = C$ of the form (26). Furthermore, the transfer function, H, (27) is symmetric.

We now assume that the expansion point s_0 for the Padé approximation is chosen to be real and nonnegative, i.e., $s_0 \geq 0$. Together with (25) it follows that the matrix $A - s_0 E$ is symmetric indefinite, with N_1 nonpositive and N_2 nonnegative eigenvalues. Thus, $A - s_0E$ admits a factorization of the following form:

$$A - s_0E = -F_1 J F_1^T, \tag{44}$$

where J is the block matrix defined in (28). Instead of the general factorization (16), we now use (44). By (44) and (18), the matrices M, R, and L, are then of the following form:

$$M = F_1^{-1} E F_1^{-T} J, \quad R = F_1^{-1} B, \quad \text{and} \quad L = -J F_1^{-1} C.$$

Since $E = E^T$ and $B = C$, it follows that

$$JM = M^T J \quad \text{and} \quad L = -JR.$$

This means that M is J-symmetric and the left starting block L is (up to its sign) the J-multiple of the right starting block R. These two properties imply that all the right and left Lanczos vectors generated by the band Lanczos Algorithm 1 are J-multiples of each other:

$$w_j = Jv_j \quad \text{for all} \quad j = 1, 2, \ldots, n.$$

Consequently, Algorithm 1 simplifies in that only the right Lanczos vectors need to be computed. The resulting version of MPVL for computing matrix-Padé models of RCL subcircuits is just the SyMPVL algorithm. The computational costs of SyMPVL are half of that of the general MPVL algorithm.

Let $H_n^{(1)}$ denote the matrix-Padé model generated by SyMPVL after n Lanczos steps. For general RCL subcircuits, however, $H_n^{(1)}$ will not preserve the passivity of the original system.

An additional reduced-order model that is guaranteed to be passive can be obtained as follows, provided that all right Lanczos vectors are stored. Let

$$V_n = \begin{bmatrix} v_1 \ v_2 \ \cdots \ v_n \end{bmatrix}$$

denote the matrix that contains the first n right Lanczos vectors as columns. Then, by projecting the matrices in the representation (29) of the transfer function H of the original RCL subcircuit onto the columns of V_n, we obtain the following reduced-order transfer function:

$$H_n^{(2)}(s) = \left(V_n^T B\right)^T \left(s V_n^T \tilde{E} V_n - V_n^T \tilde{A} V_n\right)^{-1} V_n^T B. \tag{45}$$

The passivity of the original RCL subcircuit, together with Theorem 5 implies that the reduced-order model defined by $H_n^{(2)}$ is indeed passive. Furthermore, in [33], it is shown that $H_n^{(2)}$ is a matrix-Padé-type approximation of the original transfer function and that, at the expansion point s_0, $H_n^{(2)}$ matches half as many leading coefficients of H as the matrix-Padé approximant $H_n^{(1)}$.

Next, we illustrate the behavior of SyMPVL with two circuit examples.

A package model

The first example arises is the analysis of a 64-pin package model used for an RF integrated circuit. Only eight of the package pins carry signals, the rest being either unused or carrying supply voltages. The package is characterized as a passive linear dynamical system with $m = p = 16$ inputs and outputs, representing 8 exterior and 8 interior terminals. The package model is described by approximately 4000 circuit elements, resistors, capacitors, inductors, and inductive couplings, resulting in a linear dynamical system with a state-space dimension of about 2000.

In [36], SyMPVL was used to compute a Padé-based reduced-order model of the package, and it was found that a model $H_n^{(1)}$ of order $n = 80$ is sufficient to match the transfer-function components of interest. However, the model $H_n^{(1)}$ has a few poles in the right half of the complex plane, and therefore, it is not passive.

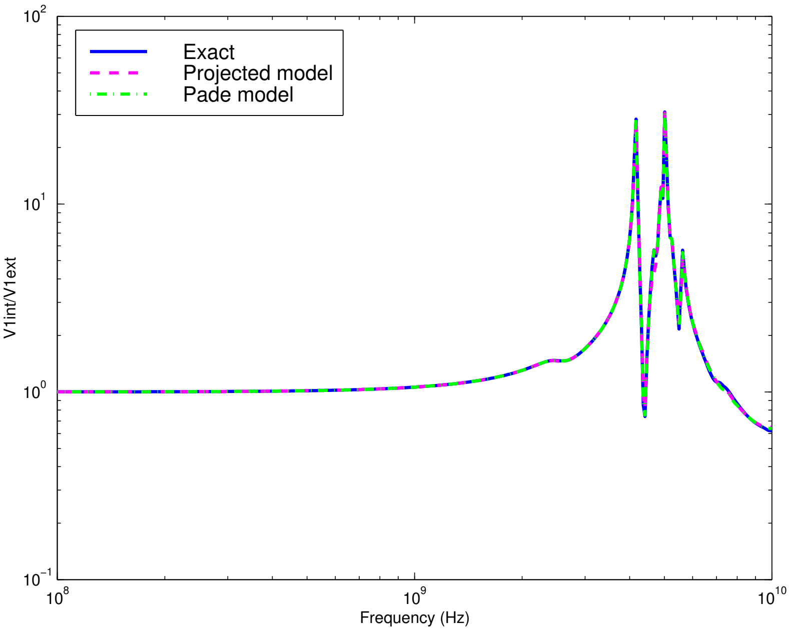

Figure 2: Package: Pin no.1 external to Pin no.1 internal, exact, projected model, and Padé model

In order to obtain a passive reduced-order model, we ran SyMPVL again on the package example, and this time, also generated the projected reduced-order model $H_n^{(2)}$ given by (45). The expansion point $s_0 = 5\pi \times 10^9$ was used. Recall that $H_n^{(2)}$ is only a Padé-type approximant and thus less accurate than the Padé approximant $H_n^{(2)}$. Therefore, one now has to go to order $n = 112$ to obtain a projected reduced-order model $H_n^{(2)}$ that matches the transfer-function components of interest. Figures 2 and 3 show the voltage-to-voltage transfer function between the external terminal of pin no. 1 and the internal terminals of the same pin and the neighboring pin no. 2, respectively. The plots show results with the projected model $H_n^{(2)}$ and the Padé model $H_n^{(2)}$, both of order $n = 112$, compared with an exact analysis.

In Figure 4, we compare the relative error of the projected model $H_{112}^{(2)}$ and the Padé model $H_{112}^{(1)}$ of the same size. Clearly, the Padé model is more accurate. However, out of the 112 poles of $H_{112}^{(1)}$, 22 have positive real parts, violating the passivity of the Padé model. On the other hand, the projected model is passive.

An extracted RC circuit

This is an extracted RC circuit with about 4000 elements and $m = 20$ ports. The expansion point $s_0 = 0$ was used. Since the projected model and the Padé model are identical for RC circuits, we only computed the Padé model via SyMPVL.

The point of this example is to illustrate the usefulness of the deflation procedure built into SyMPVL. It turned out that sweeps through the first two Krylov blocks, R and MR, of the block Krylov matrix (35) were sufficient to obtain a reduced-order model that matches the transfer function in the frequency range of interest. During the sweep through the second block, 6 almost linearly dependent vectors were discovered and deflated. As a result, the reduced-order model obtained with deflation is only of size $n = 2m - 6 = 34$. When SyMPVL was rerun on this example, with deflation turned off, a reduced-order model of size $n = 40$ was needed to match the transfer function. In Figure 5, we show the $H_{1,11}$ component of the reduced-order model obtained with deflation and without deflation, compared to the exact transfer function. Clearly, deflation leads to a significantly smaller reduced-order model that is as accurate as the bigger one generated without deflation.

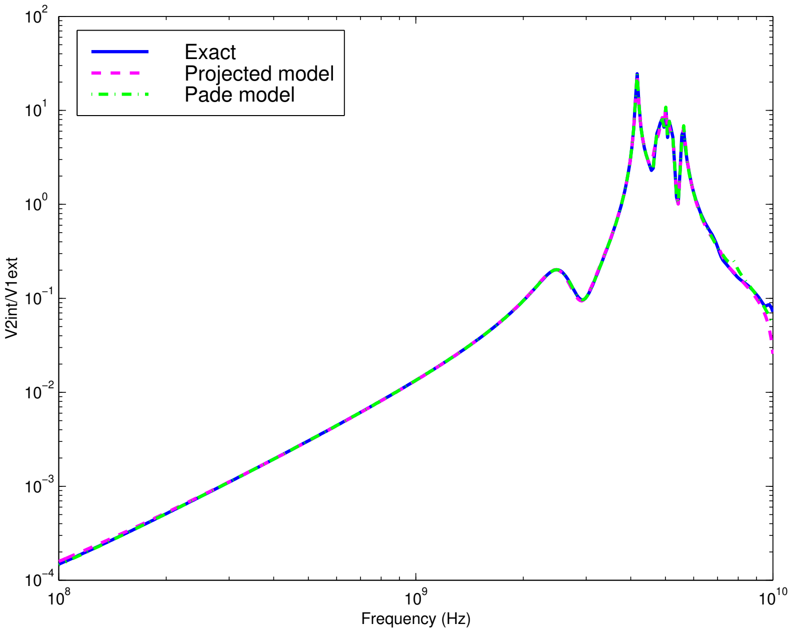

Figure 3: Package: Pin no.1 external to Pin no.2 internal, exact, projected model, and Padé model

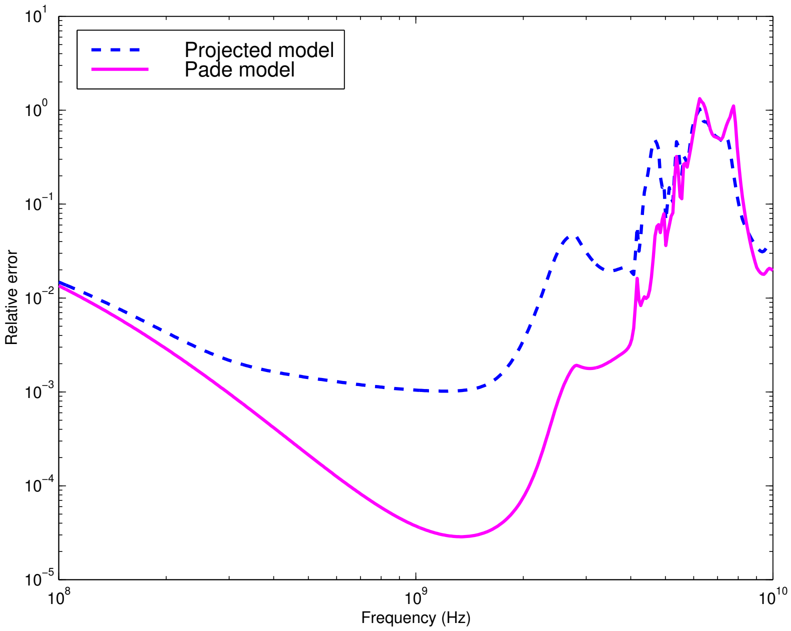

Figure 4: Relative error of projected model and Padé model

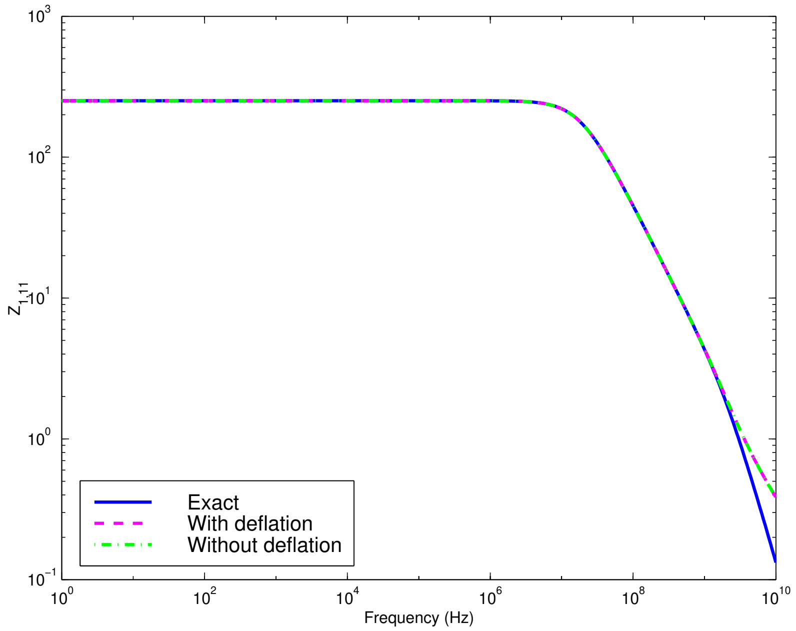

Figure 5: Impedance $H_{1,11}$

3.3 Approaches based on the Arnoldi Process

The Arnoldi process [5] is another widely-used Krylov-subspace method. A band version of the Arnoldi process that is suitable for multiple starting vectors can also be used for reduced-order modeling. However, the models generated from the band Arnoldi process are only Padé-type models.

In contrast to the band Lanczos algorithm, the band Arnoldi process only involves one of the starting blocks, namely R, and it only uses matrix-vector products with M. Moreover, the band Arnoldi process only generates one set of vectors, $v_1, v_2, \ldots, v_n$, instead of the two sequences of right and left vectors produced by the band Lanczos algorithm. The Arnoldi vectors span the n-th right block Krylov subspace (induced by M and R):

$$\operatorname{span}\{\, v_1, v_2, \ldots, v_n \,\} = \mathcal{K}_n(M, R).$$

The Arnoldi vectors are constructed to be orthonormal:

$$V_n^H V_n = I, \quad \text{where} \quad V_n := \begin{bmatrix} v_1 \ v_2 \ \cdots \ v_n \end{bmatrix}.$$

After n iterations, the Arnoldi process has generated the first n Arnoldi vectors, namely the n columns of the matrix V_n, as well as an $n \times n$ matrix $G_n^{(\mathrm{pr})}$ of recurrence coefficients, and, provided that $n \geq m$, an $n \times m$ matrix $\rho_n^{(\mathrm{pr})}$. The matrices $G_n^{(\mathrm{pr})}$ and $\rho_n^{(\mathrm{pr})}$ are projections of the matrices M and R onto the subspace spanned by the columns of V_n, which is just the block Krylov subspace $\mathcal{K}_n(M, R)$. More precisely, we have

$$G_n^{(\mathrm{pr})} = V_n^H M V_n \quad \text{and} \quad \rho_n^{(\mathrm{pr})} = V_n^H R. \tag{46}$$

The band Arnoldi process can be stated as follows.

Algorithm 2 (Band Arnoldi process)

INPUT: *A matrix* $M \in \mathbb{C}^{n \times n}$*;*

A block of m *right starting vectors* $R = \begin{bmatrix} r_1 \ r_2 \ \cdots \ r_m \end{bmatrix} \in \mathbb{C}^{n \times m}$.

OUTPUT: *The $n \times n$ Arnoldi matrix* $G_n^{(\mathrm{pr})}$.
The matrix $V_n = \begin{bmatrix} v_1 \; v_2 \; \cdots \; v_n \end{bmatrix}$ *containing the first n Arnoldi vectors,*
and the matrix $\rho_n^{(\mathrm{pr})}$.

0) *For* $k = 1, 2, \ldots, m$, *set* $\hat{v}_k = r_k$.
Set $m_{\mathrm{c}} = m$ *and* $\mathcal{I} = \emptyset$.

For $n = 1, 2, \ldots,$ *until convergence or* $m_{\mathrm{c}} = 0$ *do*:

1) (If necessary, deflate $\hat{v}_n$.)
Compute $\|\hat{v}_n\|_2$.
Decide if $\hat{v}_n$ should be deflated. If yes, do the following:

   a) *Set* $\hat{v}_{n-m_{\mathrm{c}}}^{\mathrm{defl}} = \hat{v}_n$ *and store this vector. Set* $\mathcal{I} = \mathcal{I} \cup \{\, n - m_{\mathrm{c}} \,\}$.
   b) *Set* $m_{\mathrm{c}} = m_{\mathrm{c}} - 1$. *If* $m_{\mathrm{c}} = 0$, *set* $n = n - 1$ *and stop.*
   c) *For* $k = n, n+1, \ldots, n + m_{\mathrm{c}} - 1$, *set* $\hat{v}_k = \hat{v}_{k+1}$.
   d) *Repeat all of Step* 1).

2) (Normalize $\hat{v}_n$ to obtain v_n.)
Set

$$g_{n,n-m_{\mathrm{c}}} = \|\hat{v}_n\|_2 \quad \textit{and} \quad v_n = \frac{\hat{v}_n}{g_{n,n-m_{\mathrm{c}}}}.$$

3) (Orthogonalize the candidate vectors against v_n.)
For $k = n+1, n+2, \ldots, n + m_{\mathrm{c}} - 1$, *set*

$$g_{n,k-m_{\mathrm{c}}} = v_n^H \hat{v}_k \quad \textit{and} \quad \hat{v}_k = \hat{v}_k - v_n \, g_{n,k-m_{\mathrm{c}}}.$$

4) (Advance the block Krylov subspace to get $\hat{v}_{n+m_{\mathrm{c}}}$.)

   a) *Set* $\hat{v}_{n+m_{\mathrm{c}}} = M \, v_n$.
   b) *For* $k = 1, 2, \ldots, n$, *set*

$$g_{k,n} = v_k^H \, \hat{v}_{n+m_{\mathrm{c}}} \quad \textit{and} \quad \hat{v}_{n+m_{\mathrm{c}}} = \hat{v}_{n+m_{\mathrm{c}}} - v_k \, g_{k,n}.$$

5) a) *For* $k \in \mathcal{I}$, *set* $g_{n,k} = v_n^H \, \hat{v}_k^{\mathrm{defl}}$.
   b) *Set*

$$\begin{aligned} G_n^{(\mathrm{pr})} &= \begin{bmatrix} g_{i,k} \end{bmatrix}_{i,k=1,2,\ldots,n}, \\ \rho_n^{(\mathrm{pr})} &= \begin{bmatrix} g_{i,k-m} \end{bmatrix}_{i=1,2,\ldots,n;\, k=1,2,\ldots,k_\rho} \quad \textit{where} \quad k_\rho = m + \min\{\, 0, n - m_{\mathrm{c}} \,\}. \end{aligned}$$

6) *Check if n is large enough. If yes, stop.*

Note that, in contrast to the band Lanczos algorithm, the band Arnoldi process requires the storage of all previously computed Arnoldi vectors.

Like the band Lanczos algorithm, the band Arnoldi process can also be employed to reduced-order modeling. Let M, R, and L be the matrices defined in (18). After running Algorithm 2 (applied to M and R) for n steps, we have obtained the matrices $G_n^{(\mathrm{pr})}$ and $\rho_n^{(\mathrm{pr})}$, as well as the matrix V_n of Arnoldi vectors. The transfer function H_n of a reduced-order model H_n can now be defined as follows:

$$H_n(s) = \left(V_n^H L\right)^H \left(I - (s - s_0) V_n^H M V_n\right)^{-1} \left(V_n^H R\right).$$

Using the relations (46) for $G_n^{(\mathrm{pr})}$ and $\rho_n^{(\mathrm{pr})}$, the formula for H_n reduces to

$$H_n(s) = \left(V_n^H L\right)^H \left(I - (s - s_0)G_n^{(\mathrm{pr})}\right)^{-1} \rho_n^{(\mathrm{pr})}. \tag{47}$$

The matrices $G_n^{(\mathrm{pr})}$ and $\rho_n^{(\mathrm{pr})}$ are directly available from Algorithm 2. In addition, one also needs to compute the matrix

$$\eta_n^{(\mathrm{pr})} = V_n^H L.$$

It turns out that the transfer function (47) defines a matrix-Padé-type reduced-order model.

Theorem 7 (Matrix-Padé-type model [33, 46])
Suppose that Algorithm 2 is run with exact deflation only and that $n \geq m$. Then, the reduced-order model associated with the reduced-order transfer function (47) *is a matrix-Padé-type model of the linear dynamical system* (4) *and* (5)*. More precisely, the Taylor expansions about s_0 of the transfer functions, H,* (11) *and , H_n,* (47) *agree in at least*

$$q'(n) \geq \left\lfloor \frac{n}{m} \right\rfloor$$

leading coefficients:

$$H(s) = H_n(s) + \mathcal{O}\big((s - s_0)^{q'(n)}\big). \tag{48}$$

Remark 4 The number $q'(n)$ is the exact number of terms matched in the expansion (48) provided that no exact deflations occur in Algorithm 2. In the case of exact deflations, the number of matching terms is somewhat higher, but so is the number of matching terms for the matrix-Padé model of Theorem 6; see [33]. In particular, the matrix-Padé model is always more accurate than the matrix-Padé-type model obtained from Algorithm 2. On the other hand, the band Arnoldi process is certainly simpler than the band Lanczos process., Furthermore, the true orthogonality of the Arnoldi vectors in general results in better numerical behavior than the bi-orthogonality of the Lanczos vectors.

Remark 5 For the special case of RCL subcircuits, the algorithm PRIMA proposed in [46, 47] can be interpreted as a special case of the Arnoldi reduced-order modeling procedure described here. Furthermore, in [30, 33] it is shown that the reduced-order model produced by PRIMA is mathematically equivalent to the additional passive model produced by SyMPVL. In contrast to PRIMA, however, SyMPVL also produces a true matrix-Padé model, and thus PRIMA does not appear to have any real advantage over or be even competitive with SyMPVL.

4 Schur interpolation

4.1 The setting

The modeling of physical effects often produces large, positive definite Hermitian matrices. For example, the modeling of interconnects in an integrated circuit produces in first instance a full elastance matrix G from which a sparse approximating capacitance matrix C has to be derived. Likewise, the behavior of the substrate of an integrated circuit is modeled by a conductivity matrix, and the inductive behavior of the interconnects by an inductance matrix. These matrices are positive definite, because they express either conservation of energy or dissipation. It is a non-trivial problem to find low-complexity approximations to a positive definite matrix, which are positive definite in their own right. For example, if $G = \left[G_{i,j}\right]$ is positive definite, then the matrix G_a obtained by putting elements outside a given band equal to zero, i.e., $(G_a)_{i,j} = G_{i,j}$ for $|i - j| < n$ some n, and zero otherwise, will not necessarily be positive definite. If a matrix is diagonally dominant, then putting some off-diagonal elements equal to zero while keeping the Hermitian property would preserve the dominancy and hence also the positive definiteness. We shall analyze some of the properties of such schemes soon. An important observation is that properties such as "banded" and "diagonally dominant" are not preserved under inversion: the inverse of a banded matrix is not banded (except when the matrix is block diagonal) and the inverse of a diagonally dominant matrix is not diagonally dominant. Consider for example the matrix (for real a)

$$M_a = \begin{bmatrix} 1 & a & a^2 \\ a & 1 & a \\ a^2 & a & 1 \end{bmatrix}.$$

It is positive definite for $|a| < 1$ with inverse

$$M_a^{-1} = \frac{1}{1-a^2} \begin{bmatrix} 1 & -a & 0 \\ -a & 1+a^2 & -a \\ 0 & -a & 1 \end{bmatrix}.$$

If we truncate M_a by putting $(M_a)_{1,3} = (M_a)_{3,1} = 0$, then the resulting matrix will be positive definite only when in addition $a \leq 1/\sqrt{2}$. We see that the inverse of M_a is diagonally dominant for $|a| < 1$ while that is only the case for M_a when $a < (\sqrt{5}-1)/2$. So, why would it be better to truncate a matrix rather than its inverse? A related issue is whether the inverse of a banded matrix has the same computational complexity as the original. Further in this section we shall develop a nice theory that is capable of answering such questions.

Another approach would be to perform the approximation on a Cholesky factor R where $G = R^H R$, R is upper triangular and R^H represents the Hermitian conjugate of R, rather than on the original matrix. Assuming that the off-diagonal elements of R become small the farther they are located from the main diagonal, it makes sense to approximate R by a banded matrix. Also, approximating R by some approximant R_a will produce automatically an approximant $G_a = R_a^H R_a$ that is positive definite. At first sight it would appear that it is not any better to approximate the square root than the original—an ϵ relative error on the square root of a scalar quantity would roughly produce a 2ϵ error on the square. The situation with matrices is, however, vastly different, since the condition number of the square root of a (positive definite) matrix, or of its Cholesky factor is just the square root of the original. Still the question arises whether a direct, element-wise approximation of the square root would be a "good" approximation technique, in the sense of either strong norms or complexity? What we need is a theory to gauge both complexity and approximation error. In addition, we would like the approximation procedure to be as simple as possible, for example, it should use a minimal amount of computations in its own right.

We start out this section with the celebrated theory of maximum-entropy interpolation of positive definite matrices. It gives a good stronghold on low-complexity approximation when "low-complexity" is understood as minimizing the number of independent algebraic parameters, e.g., by putting a sufficient number of elements in the matrix or its inverse zero. Immediately the question arises when the sparsity pattern of a positive definite matrix is preserved in its Cholesky factors. This question also has a very neat answer, namely when the matrix entries exhibit a "chordal pattern". In that case, the maximum-entropy interpolant can be found directly, in a minimal number of computations equal to the number of non-zero entries in the matrix, by a matrix interpolation algorithm that is a matrix version of the celebrated Schur interpolation algorithm of complex function analysis. The approximating properties of Schur's algorithm are known and we shall spend a few words explaining them. Finally, we shall show ways of generalizing Schur's algorithm to a more complex situation, namely the so-called "multiple band case".

4.2 Maximum-entropy interpolation of strictly positive definite matrices

Suppose that the following information on an otherwise unknown strictly positive definite (and of course Hermitian) matrix G of size $N \times N$ is given:

- The diagonal elements $G_{k,k}$ for all $k = 1, 2, \ldots N$;

- Some off-diagonal elements, characterized by a set $\mathcal{S}$: if $(i,j) \in \mathcal{S}$ then $G_{i,j}$ is known. Since G is Hermitian, we restrict elements of $\mathcal{S}$ to be in the strictly upper triangular zone where $i < j$.

This information is known as "interpolating conditions". The question we ask is: *is it possible to find a positive definite matrix G_a which has the assigned element values on the main diagonal and the set $\mathcal{S}$, and is otherwise in some sense "of minimal complexity"?*

It turns out that this question has a nice definite answer if "complexity" here is understood to mean: "the value of the off-diagonal elements $(G^{-1})_{i,j}$ is zero for (i,j) not in $\mathcal{S}$". A comfortable treatment of the theory leading to this result requires the introduction of the notion of "entropy of a strictly positive matrix H", originating from stochastic system theory and which is given by the (finite) quantity:

$$\mathcal{E}(H) = \log\det H.$$

The following theorem is valid.

Theorem 8 *Suppose that the diagonal elements $G_{k,k}$ and some off-diagonal elements belonging to an off-diagonal set of indices $\mathcal{S}$ of a strictly positive definite matrix G are given. Then, there exists a unique strictly positive definite matrix G_a such that G_a interpolates the given entries, i.e., $(G_a)_{i,j} = G_{i,j}$ for $i = j$ and $(i,j) \in \mathcal{S}$, and which is such that $(G_a^{-1})_{i,j} = 0$ for (i,j) not in $\mathcal{S}$. This G_a also maximizes the entropy $\mathcal{E}(H) = \log\det H$ over all H that meet the interpolation conditions.*

Sketch of proof Suppose that H is a strictly positive definite matrix depending on some parameter ξ. The differential of the entropy with respect to ξ is then given by

$$\frac{\partial}{\partial\xi}\log\det H = \frac{1}{\det H}\frac{\partial\det H}{\partial\xi}.$$

Let us observe that the dependency of $\det H$ on a given entry $H_{i,j}$ can be expressed using the Cramer minor expansion based on the row i:

$$\det H = \sum_{k=1}^{N} H_{i,k}M_{i,k}$$

where $M_{i,k}$ is the minor corresponding to the element at the position (i,k). The minor $M_{i,k}$ does not depend on any element in the i-th row of H, in particular it does not depend on $H_{i,j}$—the determinant is linear in that element. Let now $\xi = G_{i,j}$ for some (i,j) not in $\mathcal{S}$, corresponding to the position of an element that must be determined. Since the $\log\det H$ surface is smooth over the space of parameters to be determined, an extremum will only occur if each possible ξ is chosen so that the variation of the entropy with respect to ξ is zero (or else at the border of feasibility, but that situation cannot lead to a maximum since the border corresponds to matrices whose determinant is zero). The variation for $\xi = G_{i,j}$ on G is now given by:

$$\frac{\partial}{\partial\xi}\log\det G = \frac{1}{\det G}\frac{\partial\det G}{\partial\xi} = \frac{M_{i,j}}{\det G} = (G^{-1})_{j,i}.$$

Hence the top of the entropy surface in the parameter space of the unknown entries of the matrix G_a, i.e., the entries not in $\mathcal{S}$, must correspond to a strictly positive definite extension G_a of G for which $(G_a^{-1})_{i,j} = 0$. The proof now terminates by showing that this top exists and is unique. This must be reasonable in view of the fact that there is a uniform upper bound on the entropy, namely

$$\sum_{k=1}^{N}\log G_{k,k}.$$

This bound can be obtained through recursive evaluation via Cholesky decomposition, and the fact that the interpolating set is convex, if H_1 and H_2 are strictly positive definite and interpolating, so is $kH_1 + (1-k)H_2$ for $0 \le k \le 1$.

$\square$

Hence the maximum-entropy extension of entries of a strictly positive definite matrix does exist, and it produces a sparse inverse matrix! This is already a very useful result for model reduction of, for example, capacitive models of IC interconnects, as we shall soon see. However, it is a theoretical result in that the proof of existence does not produce a direct algorithm to compute the result. One may resort to dynamic optimization, and, indeed, that should lead to a solution, but maybe a problematic one, first because it leads to complex computations involving all the elements outside the interpolating set, and second because the entropy surface is most likely very flat, making the optimum hard to find even though there are very good algorithms for convex optimization. Hence it pays to find a way of computing the solution directly on the basis of the known data, if possible. This question is related to the question whether a sparsity pattern in an original, strictly positive definite matrix G is preserved in the Cholesky factor L, where $G = LL^H$, a question which we now address.

4.3 Chordal systems

Assume that we are given a strictly positive definite matrix G whose diagonal elements are known and which is otherwise sparse with upper triangular sparsity pattern $\mathcal{S}$, i.e., $G_{i,j} = 0$ for (i,j) with $i < j$ not belonging to $\mathcal{S}$ (G is of course Hermitian). Connected to $\mathcal{S}$ there is a *sparsity graph* defined as follows:

- Nodes: there are N nodes corresponding to the N rows of the matrix;
- Edges: there is an edge between node i and node j iff $(i,j) \in \mathcal{S}$, assuming $i < j$.

For example, a matrix with fillings

$$\begin{bmatrix} * & * & \cdot & * & \cdot \\ * & * & * & \cdot & * \\ \cdot & * & * & * & \cdot \\ * & \cdot & * & * & * \\ \cdot & * & \cdot & * & * \end{bmatrix} \tag{49}$$

has the sparsity graph shown in Figure 4.3.

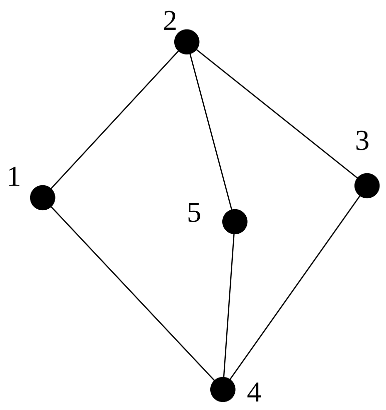

Figure 6: Sparsity graph of the matrix template (49)

We say that a sparsity graph is *chordal* when there is no loop of more than three nodes that has no chord in the graph, a chord being a direct connection between two nodes (with reference to a polygone). The graph shown in Figure 4.3 is non-chordal, the loop 1-2-3-4-1 has no chords (and there are more such loops). It turns out that the Cholesky factorization of a positive definite matrix with chordal sparsity graph will suffer no fill-ins provided it is executed in the right order. To find that order we need another property of chordal graphs.

We shall say that a node of a graph has an *adjacent clique* if the subgraph consisting of that node and the nodes directly connected to it together with the edges connecting these nodes form a clique, i.e., are fully connected. A chordal graph now has the two following properties:

- The graph obtained by deleting one node with the edges connected to it is chordal;
- It has at least one node which has an adjacent clique.

The first property is almost evident, while the second property can be proven recursively on the number of nodes. Hence, a reordering and peeling off of the nodes of a chordal graph is possible whereby each node in turn has an adjacent clique in the remaining graph: start with such a node in the original graph, remove it with its connecting edges and continue recursively. Finding a node with an adjacent clique can be done in less than N^2 steps, hence the complexity of the reordering is certainly polynomial in N.

With this reordering of nodes, performing the Cholesky factorization in the order of peeling will not produce any fill-ins, exactly because of the adjacent clique property at each step. The converse is "generically" true as well, if a Cholesky factorization does not result in fill-ins *generically* (an element might accidentally become zero), then the sparsity graph must be chordal as well. It turns out that the maximum-entropy interpolant of a matrix with chordal sparsity pattern can be computed directly on the given entries, the famous algorithm to do so is the generalized Schur algorithm described in the next subsection. Unfortunately, many problems in modeling or reduced modeling of integrated circuits involve strictly positive definite matrices that do not have chordal sparsity patterns. In particular, multiband patterns are almost essentially non chordal and hence will need additional, non-exact techniques for reduced modeling. This question is treated in the section on multiband generalization. A special case of a chordal graph is a graph representing a staircase filling, i.e., a filling corresponding to a non-regular band. One would obtain such a graph if in the order of nodes with adjacent cliques, each node in turn belongs to the adjacency set of its predecessor.

4.4 Schur's algorithm in the chordal case

We are now ready to introduce the generalized matrix Schur algorithm, originally presented as an estimation algorithm in [19], and whose matrix properties were analyzed in [22]. The application of the algorithm to reduced modeling of integrated circuits was given in [20]. We utilize the algebraic framework of the latter paper, slightly generalizing it to cover chordal sparsity in addition to staircases. Let the original, $N \times N$ strictly positive definite matrix be $G = \left[G_{i,j}\right]$ and let D be its main diagonal:

$$D = \operatorname{diag}\left(G_{1,1}, G_{2,2}, \ldots, G_{N,N}\right).$$

It is advantageous to work with a normalized version of G, for theoretical purposes if not for numerical ones. Hence, let

$$g = D^{-1/2} G D^{-1/2}.$$

The matrix g will have all its diagonal elements equal to one (the situation could be generalized to the case where all the entries in G are in fact matrices, the block case, but for simplicity of explanation we keep the procedures scalar and shall indicate later on how to handle the block-matrix case). Let us assume, moreover, that the nodes are put in a correct adjacent-clique order, the staircase order will do if available.

A side excursion: the classical Schur parametrization case

Before engaging in the description of the matrix Schur algorithm, let us make a brief side excursion to the original algorithm involved in Schur's parametrization of a contractive, analytic function on the unit disc $\mathbf{D} = \{z : |z| < 1\}$ of the complex plane. Suppose that

$$s(z) = s_0 + s_1 z + s_2 z^2 + \cdots$$

is such a function, represented by its MacLaurin series. The question answered by the Schur parametrization is whether the given MacLaurin series does indeed correspond to a contractive function. To start, either $|s_0| = 1$ and $s(z)$ reduces to a constant of modulus one (by the maximum modulus theorem of complex analysis), or $|s_0| < 1$ and then a new contractive function which is analytic in $\mathbf{D}$ may be derived from $s(z)$ via the recipe:

$$s^{(1)}(z) = \frac{s(z) - s_0}{z\,(1 - \overline{s_0}s(z))} = s_0^{(1)} + s_1^{(1)}z + \cdots .$$

Notice that the transformation

$$s \mapsto \frac{s - s_0}{1 - \overline{s_0}s}$$

maps the unit disc onto itself. The procedure may be repeated on $s^{(1)}(z)$, yielding a criterion on $s_0^{(0)}$ and a new $s^{(2)}(z)$, and then recursively continued further. Let $\rho_0 = s_0$, $\rho_1 = s_0^{(1)}$, ... be the so-called "Schur parameters" for $s(z)$. In an inverse scattering context where they often appear, the ρ_k's are also called reflection coefficients. The sequence of Schur parameters of a contractive function that is analytic in $\mathbf{D}$ is either finite, in which case the last coefficient is of unit modulus, or infinite, and then all Schur parameters are less than one in modulus. The Schur parameters determine $s(z)$ uniquely, just as the s_k's do, one series can be converted into the other and vice versa. In his famous paper [51], Schur demonstrates that $s(z)$ is contractive in the unit disc iff the Schur parametrization satisfies one of these two properties—this is the Schur criterion for contractivity (the proof is in fact pretty straightforward). The transformation that leads from $s^{(k)}(z)$ to $s^{(k+1)}(z)$ is obviously bilinear. It can be linearized if it is put in matrix form. Let us write for that purpose

$$s^{(n)}(z) = \frac{\delta^{(n)}(z)}{\gamma^{(n)}(z)}.$$

Then the following linear recursion produces the same effect as the original Schur parametrization

$$z\left[\gamma^{(n+1)}(z) \;\; \delta^{(n+1)}(z)\right] = \left[\gamma^{(n)}(z) \;\; \delta^{(n)}(z)\right] \frac{1}{\sqrt{1 - |\rho_n|^2}} \begin{bmatrix} z & -\rho_n \\ -\overline{\rho_n} & 1 \end{bmatrix}$$

when the Schur parameter chosen as

$$\rho_n = \frac{\delta^{(n)}(0)}{\gamma^{(n)}(0)}$$

is less than one in magnitude (the square roots are included for normalization purposes, they may be dispensed with in practical computations). The recursion is started with $\left[\gamma^{(0)}(z)\, \delta^{(0)}(z)\right] = \left[1 \; s(z)\right]$. Aside from a shift represented by z, the Schur recursion involves transformations with a hyperbolic matrix, sometimes called a Halmos transformation and defined as

$$H(\rho) = \frac{1}{\sqrt{1 - |\rho|^2}} \begin{bmatrix} 1 & -\rho \\ -\overline{\rho} & 1 \end{bmatrix}.$$

Let us define the signature matrix

$$J = \begin{bmatrix} 1 & \\ & -1 \end{bmatrix}.$$

Then, we compute easily that $H(\rho)J\left(H(\rho)\right)^H = \left(H(\rho)\right)^H JH(\rho) = J$, which represents the hyperbolic property.

The original Schur theory works on a contractive function $s(z)$. Alternatively, one could start from what is known as a *positive real function* $\phi(z)$, i.e., a function that is analytic in $\mathbf{D}$ and such that

$\mathrm{Re}(\phi(z)) = (\phi(z) + \overline{\phi(z)})/2 \geq 0$ in $\mathbf{D}$. The Cayley transformation relates a contractive function $s(z)$ to a *positive real function* $\phi(z)$ (i.e. a function with positive real part $\mathrm{Re}(\phi(z))$ for all z in the unit disc):

$$s(z) = \frac{\phi(z) - 1}{\phi(z) + 1}.$$

Schur's parametrization provides a test for positive reality on the sequence defined by the MacLaurin expansion of ϕ, the linearized recursion can now be started with

$$\left[\gamma^{(0)}(z) \;\; \delta^{(0)}(z)\right] = \frac{1}{2}\left[\phi(z) + 1 \;\; \phi(z) - 1\right].$$

After $n+1$ steps it will yield

$$\frac{1}{2}\left[\phi(z) + 1 \;\; \phi(z) - 1\right]\theta_0(z)\theta_2(z)\cdots\theta_n(z) = z^n\left[\gamma^{(n)}(z) \;\; \delta^{(n)}(z)\right]$$

with each $\theta_i(z)$ representing an elementary Schur step. Let us introduce the para-Hermitian conjugate of a function of z as $f^*(z) = \overline{f\left(1/\overline{z}\right)}$. In the Schur parametrization theory (see, e.g., [25]), one deduces that the overall Schur matrix $\Theta_n(z) = \theta_0(z)\theta_1(z)\cdots\theta_n(z)$ has the form

$$\Theta_n(z) = \frac{1}{2}\begin{bmatrix} (1 + \phi_n^*(z))\, T_{rn}^{-*}(z) & (1 - \phi_n(z))\, T_{fn}^{-1}(z) \\ (1 - \phi_n^*(z))\, T_{rn}^{-*}(z) & (1 - \phi_n(z))\, T_{fn}^{-1}(z) \end{bmatrix}$$

in which $\phi_n(z)$ is also PR in $\mathbf{D}$, $T_{rn}(z)$ and $T_{fn}(z)$ are analytic in $\mathbf{D}$ and

$$\frac{\phi_n(z) + \phi_n^*(z)}{2} = T_{rn}(z)T_{rn}^*(z) = T_{fn}^*(z)T_{fn}(z).$$

(Notice that the para-Hermitian conjugate is equal to the Hermitian conjugate only on the unit circle. Outside the unit circle it is its analytic continuation, when definable. Often in the engineering literature, the para-Hermitian conjugate is denoted by a sub-star, in contrast to the upper star, which is often interpreted as equal to complex conjugation. Here we use upper star, to indicate that the upper-stared quantity corresponds in fact to the analytic continuation of the adjoint in the Fourier domain on the unit circle). One of the central properties of $\phi_n(z)$, resulting from the Schur parametrization, is that it interpolates the original $\phi(z)$ to order n:

$$\phi(z) = \phi_n(z) + z^{n+1}r(z)$$

in which $r(z)$ is analytic in $\mathbf{D}$. Remark also that $\phi_n^{-1}(z)$ is polynomial hence $\phi_n(z)$ is of the "autoregressive type". The theory of maximum entropy interpolation is well developed in complex function theory, and it is satisfied by $\phi_n(z)$ as a maximum entropy interpolant of order n for $\phi(z)$, whereby the entropy measure now must be taken as

$$\int_{-\pi}^{\pi} \log \mathrm{Re}(\phi(\xi))\frac{d\xi}{2\pi}.$$

The matrix case

In the matrix case, the hyperbolic transformation will play a role similar to the complex case. We embed the Halmos transformation in an otherwise unitary matrix and index its position, much as is done in the classical QR algorithm based on Jacobi transformations. This leads to $2N \times 2N$ hyperbolic matrices $\theta_{i,j}(\rho)$ of the form

$$
\left[\begin{array}{ccccccccc|ccccccc}
1 & & & & & & & & & 0 & & & & & & \\
\ddots & & & & & & & & & \ddots & & & & & & \\
1 & & & & & & & & & 0 & & & & & & \\
 & \frac{1}{\sqrt{1-|\rho|^2}} & 0 \cdots & & & & & & & & 0 & \cdots 0 & -\frac{\rho}{\sqrt{1-|\rho|^2}} & & & \\
 & 0 & 1 & & & & & & & & 0 & & 0 & & & \\
 & \vdots & & \ddots & & & & & & & & \ddots & \vdots & & & \\
 & & & & 1 & & & & & & & 0 & & & & \\
 & & & & & 1 & & & & & & & 0 & & & \\
 & & & & & & 1 & & & & & & & 0 & & \\
 & & & & & & & \ddots & & & & & & & \ddots & \\
 & & & & & & & & 1 & & & & & & & 0 \\
\hline
0 & & & & & & & & & 1 & & & & & & \\
\ddots & & & & & & & & & \ddots & & & & & & \\
0 & & & & & & & & & 1 & & & & & & \\
 & 0 & & & & & & & & & 1 & & & & & \\
 & & 0 & & & & & & & & & 1 & & & & \\
 & \vdots & & \ddots & & & & & & & & \ddots & \vdots & & & \\
 & 0 & & & 0 & & & & & & & 1 & 0 & & & \\
 & -\frac{\rho^*}{\sqrt{1-|\rho|^2}} & 0 \cdots 0 & & 0 & & & & & & & \cdots 0 & \frac{1}{\sqrt{1-|\rho|^2}} & & & \\
 & & & & & 0 & & & & & & & & 1 & & \\
 & & & & & & & \ddots & & & & & & & \ddots & \\
 & & & & & & & & 0 & & & & & & & 1
\end{array}\right]
$$

in which the elements of the elementary hyperbolic transformation are on the intersections of rows i and $N+j$ with columns i and $N+j$. Similar as in the complex case we will use it to eliminate entries in an appropriate linearization. We give the analog for Schur's algorithm in the matrix case first before motivating it. This simplifies the discussion, but an alert reader will recognize the similarities with the complex case. First we define the equivalent of the function $\phi(z)$ (as announced earlier we work on a normalized version of G, although this is not strictly necessary):

$$
\Phi = \begin{bmatrix} 1 & 2g_{1,2} & 2g_{1,3} & \cdots & 2g_{1,N} \\ 0 & 1 & 2g_{2,3} & \cdots & 2g_{2,N} \\ 0 & 0 & 1 & \cdots & 2g_{3,N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}.
$$

Hence $g = \frac{1}{2}(\Phi + \Phi^H)$, where A^H indicates the Hermitian conjugate of the matrix A. We define as initial data

$$
\Gamma_0 = \begin{bmatrix} 1 & g_{1,1} & \cdots & g_{1,N} \\ 0 & 1 & \cdots & g_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}, \quad \Delta_0 = \begin{bmatrix} 0 & g_{1,1} & \cdots & g_{1,N} \\ 0 & 0 & \cdots & g_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}.
$$

Hence we have

$$
\begin{bmatrix} \Gamma_0 & \Delta_0 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \Phi + I & \Phi - I \end{bmatrix}.
$$

Let $S_0 = \Gamma_0^{-1}\Delta_0$. We see that

$$g = \frac{\Phi + \Phi^H}{2} = \frac{1}{4}(\Phi + I)(I - S_0 S_0^H)(\Phi^H + I),$$

and hence S_0 is a contractive matrix in the sense that $S_0 S_0^H \preceq I$. We shall say that a couple of $N \times N$ upper triangular matrices $\begin{bmatrix}\Gamma & \Delta\end{bmatrix}$ are (*strictly*) *admissible* if Γ is invertible and $\Gamma^{-1}\Delta$ is (strictly) contractive. Define the $2N \times 2N$ signature matrix

$$J = \begin{bmatrix} I_N & \\ & -I_N \end{bmatrix}$$

where I_N is the unit matrix of dimension N. If Θ is a $2N \times 2N$ is a J-unitary matrix, i.e.,

$$\Theta J \Theta^H = \Theta^H J \Theta = J,$$

then any transformation of an admissible $\begin{bmatrix}\Gamma & \Delta\end{bmatrix}$ on the right with Θ will yield a new matrix

$$\begin{bmatrix}\Gamma' & \Delta'\end{bmatrix} = \begin{bmatrix}\Gamma & \Delta\end{bmatrix}\Theta,$$

which is (strictly) admissible when the original is (strictly) admissible. A product of J-unitary matrices will itself be J-unitary as well.

The Schur elimination procedure based on the chordal set $\mathcal{S}$ will consist in applying a sequence of elementary Halmos transformations on recursively computed admissible matrices, starting with $\begin{bmatrix}\Gamma_0 & \Delta_0\end{bmatrix}$, in the adjacent-clique order on the interpolation data. Each Halmos transformation is intended to eliminate one off-diagonal entry corresponding to a position in the set $\mathcal{S}$. Let the matrices G, g, Γ_0, Δ_0 be ordered in the adjacent-clique order, and suppose that the elements of $\mathcal{S}$ in row i are given by $(i, n_{i,1}), (i, n_{i,2}), \ldots, (i, n_{i,m_i})$ where $i < n_{i,1} < \cdots < n_{i,m_i}$ (the set may even be empty of course). We shall perform the elimination procedure in the strict order $(1, n_{1,1}), (1, n_{1,2}), \ldots, (2, n_{2,1}), \ldots$. Let us number these steps by the integer K. At step K corresponding to, say, the predecessor of $(i, n_{i,k})$, we have available an admissible pair $\begin{bmatrix}\Gamma_K & \Delta_K\end{bmatrix}$, which is such that the elements $(\Delta_K)_{i,j}$ have been annihilated for all pairs (i,j)'s in the elimination list preceding $(i, n_{i,k})$. The new step will annihilate $(\Delta_K)_{i,n_{i,k}}$ and use for that purpose an elimination matrix of the Halmos type, namely $H_{i,n_{i,k}}(\rho_{i,n_{i,k}})$ with

$$\rho_{i,n_{i,k}} = (\Gamma_K)^{-1}_{i,n_{i,k}}(\Delta_K)_{i,n_{i,k}}.$$

At least three remarks are important here:

- The element $(\Delta_{K+1})_{i,n_{i,k}}$ is set equal to zero by the elimination procedure;
- The elements that were put to zero in previous steps remain zero in all the subsequent eliminations because of the adjacent-clique order;
- There are no fill-ins, also due to the adjacent-clique property at each step.

After completion of all the steps, an overall elimination matrix Θ_t results given by

$$\Theta_t = \theta_{1,n_{1,1}}\theta_{1,n_{1,2}}\cdots\theta_{N,n_{N,m_N}}$$

and finally

$$\begin{bmatrix}\Gamma_t & \Delta_t\end{bmatrix} = \begin{bmatrix}\Gamma_0 & \Delta_0\end{bmatrix}\Theta_t$$

are obtained, in which all the elements belonging to the set $\mathcal{S}$ in Δ_t have been annihilated (as well as all the diagonal elements due to the initial normalization). In parallel, the entries in Θ_t are essentially constrained to the diagonal, the set $\mathcal{S}$ and its reflection. To make this statement more precise, let

$$\Theta_t = \begin{bmatrix}\Theta_{1,1} & \Theta_{1,2} \\ \Theta_{2,1} & \Theta_{2,2}\end{bmatrix}.$$

Then, the non-zero entries of $\Theta_{1,1}$ are restricted to diagonals and $\mathcal{S}^*$, those of $\Theta_{2,2}$ to diagonals and $\mathcal{S}$ while the non-zero entries of $\Theta_{1,2}$ are restricted to $\mathcal{S}$ and those of $\Theta_{2,1}$ are restricted to $\mathcal{S}^*$. This follows also from the special structure of $\mathcal{S}$ and the order in which the eliminations have been done. We shall call such a J-unitary matrix "$\mathcal{S}$-based". The following theorem holds [22].

Theorem 9 *An $\mathcal{S}$-based J-unitary matrix Θ_t has the form*

$$\frac{1}{2}\begin{bmatrix} \left(I+\Phi_t^H\right) L_t^{-H} & \left(I-\Phi_t\right) M_t^{-1} \\ \left(I-\Phi_t^H)\right) L_t^{-H} & \left(I+\Phi_t\right) M_t^{-1} \end{bmatrix}$$

in which Φ_t, L_t and M_t are upper triangular matrices, Φ_t has unit main diagonal, L_t and M_t are invertible, and in addition

$$\frac{\Phi_t+\Phi_t^H}{2} = L_t L_t^H = M_t^H M_t.$$

The Schur procedure executed as detailed above yields the following interpolation result.

Theorem 10 *Let $g_t = \frac{1}{2}(\Phi_t + \Phi_t^H)$ be the result of the Schur elimination procedure based on the chordal set $\mathcal{S}$. Then [22]*

$$(g-g_t)_{i,j} = 0 \textit{ for } (i,j) \in \mathcal{S}$$

and in particular

$$\Phi - \Phi_t = 2\Delta_t M_t$$

where Δ_t is defined by $\begin{bmatrix}\Gamma_0 & \Delta_0\end{bmatrix}\Theta_t = \begin{bmatrix}\Gamma_t & \Delta_t\end{bmatrix}$.

Given the theory developed so far, the two theorems are not too hard to prove. The Schur recursion necessitates a number of elementary Halmos transformations precisely equal to the number of elements in the interpolation set $\mathcal{S}$, and it produces the desired maximum-entropy interpolant, due to the fact that the appropriate entries in the inverse matrix are zero. Notice also that L_t^{-1} and M_t^{-1} have supports on $\mathcal{S}$ and the diagonal, while L_t, M_t and Φ_t are full matrices, which in practical calculations will never be computed—a banded computational scheme exists for vector-matrix multiplication with both L_t and L_t^{-1}, see [22].

4.5 Generalizations

The preceding theory works only for matrices with a chordal sparsity pattern. Can the theory be extended to more general types of matrices, in particular to matrices with multiple bands, as often occur in 2D or 3D finite element or finite difference problems. We give an indication on how an approximate technique may yield satisfactory results. We refer the reader to the literature for further information [45]. A first remark is that in some, quite common cases, a double banded (or even multibanded) matrix can be chordal. For example, a $2n \times 2n$ matrix with four $n \times n$ blocks with filling pattern as in

$$\left[\begin{array}{cccc|cccc} * & * & & & * & * & & \\ * & * & * & & * & * & * & \\ & * & * & * & & * & * & * \\ & & * & * & & & * & * \\ \hline * & * & & & * & * & & \\ * & * & * & & * & * & * & \\ & * & * & * & & * & * & * \\ & & * & * & & & * & * \end{array}\right]$$

is actually of chordal type and can be solved exactly using Schur matrix interpolation (more general forms can easily be derived using the theory of adjacent cliques described above). This result can be used to factorize more general matrices approximatively. For example, a (positive definite) block matrix of the type

$$\begin{bmatrix} A_{11} & A_{12} & \\ A_{21} & A_{22} & A_{23} \\ & A_{32} & A_{33} \end{bmatrix}$$

in which all the non-zero blocks are only sparsely specified and which is such that the two submatrices

$$A_1 := \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad A_2 := \begin{bmatrix} A_{22} & A_{23} \\ A_{32} & A_{33} \end{bmatrix}$$

have chordal filling specifications has a sparse approximant for its inverse which can be constructed from sparse approximants of A_1, A_2, and A_{22} as follows. Let A_{ME} indicate the maximum-entropy approximant of a sparsely specified matrix A, then A_{ME}^{-1} has corresponding sparse fillings according to the theory developed above. In addition, let us introduce one more bit of notation: by "$\square A$" we mean the operation of fitting the matrix A in a larger matrix that extends its range of indices while padding it with zeros. A "good" approximant for the ME inverse of A is then given by

$$A_{ME}^{-1} \approx \square(A_1)_{ME}^{-1} + \square(A_2)_{ME}^{-1} - \square(A_{22})_{ME}^{-1}. \tag{50}$$

The significance of this formula is that the inverse of the maximum-entropy interpolant for the matrix A based on the given non-chordal definition pattern is expressed in terms of maximum interpolants of submatrices whose definition pattern is presumably chordal and which can hence be computed by a fast algorithm such as the Schur parametrization given in the previous section. We give a short motivation for this result, a complete theory with proofs is given in [44]. The main property used is the fact that for reasonably well-conditioned positive definite matrices with entries specified on a given pattern, the inverse of the ME approximant of a principal submatrix is actually a good approximation of the restriction of the inverse of the ME approximant to the same indices as the submatrix—in matrix notation, let $A(i,j)$ be the principal submatrix obtained by restraining A to the index range $i \cdots j$ then, utilizing the same pattern of specified entries,

$$(A(i,j))_{ME}^{-1} \approx (A_{ME}^{-1})(i,j). \tag{51}$$

Notice that the two matrices now have the same sparsity pattern corresponding to the pattern given, but they are not numerically the same. This opens the way for a "calculus of sparse inverse matrices" of the ME type. The formula (50) can now be interpreted as defining block-wise approximations on the ME inverse of the original matrix, whereby the middle matrix (corresponding to the "22" block) is repeated trice, each time with a different approximant. There is no guarantee that (50) actually defines a positive definite matrix, but since the approximants are assumed close, the approximation should be good when the original matrix is well conditioned, a detailed error analysis can be found in the already cited thesis [44]. The reason why (51) holds is the fact that ME approximants actually define strong norm approximants on the Cholesky factors. This seems to have been remarked first in [23]. Formula (50) generalizes to large matrices with intricate block sparsity patterns and has been used successfully in the modern finite-element modeling program for interconnects of integrated circuits SPACE [56].

5 Hankel-norm model reduction

5.1 The setting

In this section we are interested in linear operators—of the type T where T induces a linear map $y = Tu$—and where T is represented by a "model", more precisely a model that represents the linear computations the computer actually executes, based on its sequential intake of data, use of memory and sequential production of results. Such a model is called a "state-space model" because it is an instance of a classical model for a time-varying dynamical system adapted to the computational context. We start out with a simple but computationally intensive representation of the desired function and we shall proceed to reduce that representation to another one with much lower computational complexity. The model we start with will be a direct derivative of all the known data and will therefore be of much too high complexity, called a "model of high complexity". Our goal will be to reduce that model to one of smallest possible computational complexity, given a specified and acceptable tolerance on the accuracy. Here, T may be a matrix, but it may also be an infinite dimensional operator, the theory that we shall present is not restricted to finite operators. In our basic framework, T will be a lower triangular operator, it represents a "causal" transfer between the vectors u and y viewed as time series. If it so happens that T does not satisfies this property, e.g., when T is a full matrix, then we would decompose T first either additively or multiplicatively into lower/upper operators: $T = L + U$ or $T = LU$ and then approximate L and dually U separately, but it may also be useful to move the main diagonal up so that the whole matrix becomes lower triangular, see the special case treated later. Our theory does not really become more complicated if we assume that T is in fact a block matrix, i.e., that the entries in T are actually matrices themselves, provided dimensions in rows and columns match. Hence T will look as follows:

$$T = \begin{bmatrix} \ddots & & & & \\ \ddots & 0 & & 0 & \\ \ddots & T_{-1,-1} & 0 & & \\ \ddots & T_{0,-1} & \boxed{T_{0,0}} & 0 & \\ \ddots & T_{1,-1} & T_{1,0} & T_{1,1} & \ddots \\ \iddots & & \ddots & \ddots & \ddots \end{bmatrix}.$$

Here, the $T_{i,j}$ block has dimension $n_i \times m_j$ and represents a partial map of the vector entry u_j to an additive component of y_i in the output vector in the map:

$$\begin{bmatrix} \ddots & & & & \\ \ddots & 0 & & 0 & \\ \ddots & T_{-1,-1} & 0 & & \\ \ddots & T_{0,-1} & \boxed{T_{0,0}} & 0 & \\ \ddots & T_{1,-1} & T_{1,0} & T_{1,1} & \ddots \\ \iddots & & \ddots & \ddots & \ddots \end{bmatrix} \begin{bmatrix} \vdots \\ u_{-2} \\ u_{-1} \\ \boxed{u_0} \\ u_1 \\ u_2 \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ y_{-2} \\ y_{-1} \\ \boxed{y_0} \\ y_1 \\ y_2 \\ \vdots \end{bmatrix}.$$

We identify the 0-th (block-)element in a matrix by putting a square around it, and similarly for the (0,0)-th element of a matrix of operator for orientation purposes. The (linear) computation defined by $y = Tu$ as executed by a computer that takes the input data sequence u and produces the output sequence y can be represented by a "causal model" for T. The transfer from the input vector u to the output vector y can indeed be written in terms of an intermediate sequence $\{x_k\}$ of data which the computer stores in memory, and so called *realization matrices* representing the computations at the sequence point k, as:

$$\begin{aligned} x_{k+1} &= A_k x_k + B_k u_k, \\ y_k &= C_k x_k + D_k u_k. \end{aligned}$$

This is called a "time-varying state-space representation" of the computation. The dimension δ_k of the vector x_k is called the state dimension at point k, and the dimensions of the realization matrices A_k, B_k, C_k, D_k are respectively $\delta_{k+1} \times \delta_k$, $\delta_{k+1} \times m_k$, $n_k \times \delta_k$, $n_k \times m_k$. A graphical representation of the state representation is shown in Figure 5.1.

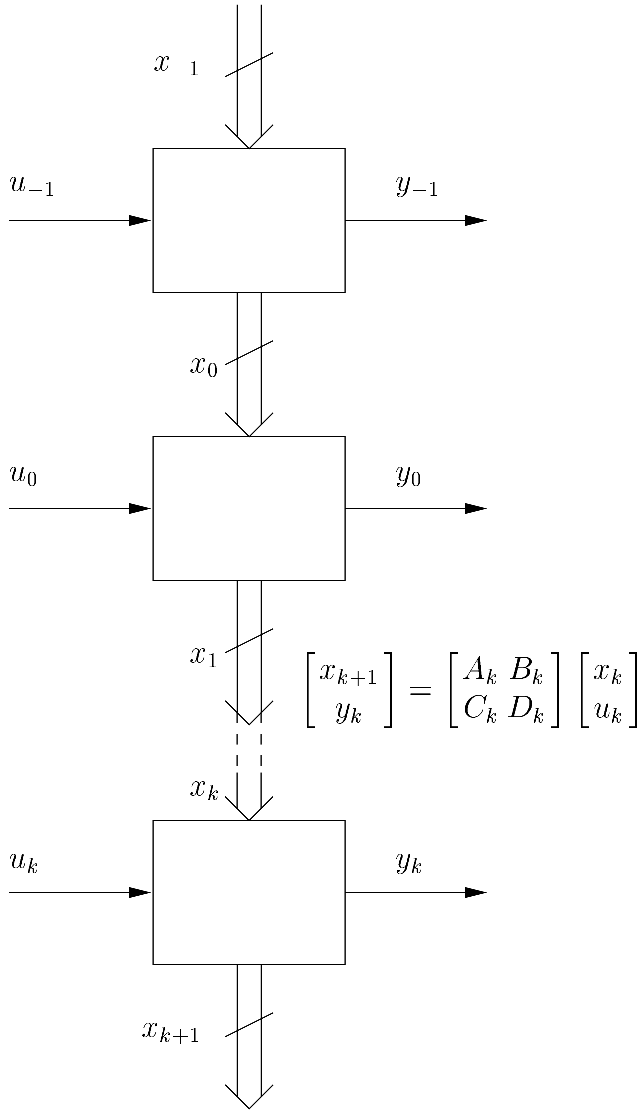

Figure 7: A causal state-space realization of an operator T: the state represents the data available for computation at a given stage.

We call A_k the *state transition matrix* at point k, while the other matrices B_k, C_k, and D_k stand for partial local maps input-state, state-output, and input-output, respectively, at point k. The system will be strictly causal when $D_k = 0$. It may happen that some of the vectors and matrices are not present. For example, if a matrix is represented by a state model, then the initial state in the representation (e.g. x_0) will not be present. In that case we say that the dimension of the respective vector is zero, it is represented by a place holder "·", but there is no numerical value present (not even zero). Similarly, the last state will also not be present when a finite matrix is represented, and disappears accordingly.

An anticausal system—represented by an upper block matrix—may similarly have an anticausal state realization as follows:

$$x'_k = A'_k x'_{k+1} + B'_k u_k,$$
$$y_k = C'_k X'_{k+1} + D'_k u_k.$$

Here, we have chosen to make the realization strictly anticausal by putting $D'_k = 0$. A graphical representation of an anticausal linear system is shown in Figure 5.1.

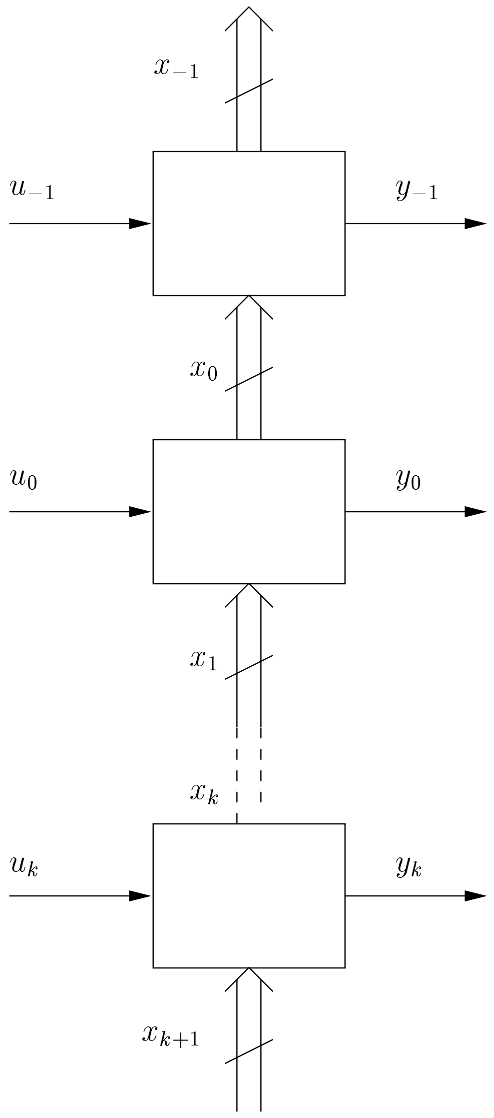

Figure 8: The signal flow in an anticausal system

It is convenient for notational purposes to assemble the realization matrices in diagonal matrices or operators with appropriate dimensions. So we define:

$$A = \begin{bmatrix} \ddots & & & & \\ & A_{-1} & & & \\ & & \boxed{A_0} & & \\ & & & A_1 & \\ & & & & \ddots \end{bmatrix}, \quad B = \begin{bmatrix} \ddots & & & & \\ & B_{-1} & & & \\ & & \boxed{B_0} & & \\ & & & B_1 & \\ \ddots & & & & \end{bmatrix},$$

and so forth. Introducing also the causal shift matrix:

$$Z = \begin{bmatrix} \ddots & & & & \\ \ddots & 0 & & & \\ & I & \boxed{0} & & \\ & & I & 0 & \\ & & & \ddots & \ddots \end{bmatrix},$$

we see that the original operator T can be expressed in terms of these diagonal matrices as follows:

$$T = D + C(I - ZA)^{-1}ZB.$$

This generalizes the classical representation matrix for stationary discrete time systems, where Z now replaces the classical causal shift z (notice, however, that the unit matrices in Z may have different dimensions and that Z does not commute with matrices in the sense that $ZA \neq AZ$, as the scalar shift "z" would). Some care must be exercised when one interprets these formulas in the case of finite matrices. The block diagonals, A, B, C, D, and Z are all block matrices or operators with appropriate dimensions. So, A will map a state sequence of dimensions $\cdots, \delta_{-1}, \boxed{\delta_0}, \delta_1, \cdots$ to $\cdots, \delta_0, \boxed{\delta_1}, \delta_2, \cdots$, and Z applied to the same state sequence will map to $\cdots, \delta_{-2}, \boxed{\delta_{-1}}, \delta_0, \cdots$. Numerically, Z will be a perfect unit matrix in the finite case, but its block decomposition will make it look like the shift matrix that it is, for a shift on a finite sequence will keep the numerical values of that sequence, but will shift their indices! The inverse in the formula for T can be interpreted in a purely formal sense as meaning $(I - AZ)^{-1} = I + AZ + (AZ)^2 + \cdots$, but the series will of course also converge in the operator sense, if AZ is idempotent (which would be the case with finite matrices) or if $(AZ)^k$ converges to zero quickly enough. To make the notion of convergence more precise, we introduce a norm on the input and output spaces, namely the ℓ_2 or quadratic norm:

$$\|u\| = \sqrt{\sum_j \|u_j\|^2},$$

where the $\|u_j\|$ is the usual Euclidean norm on a vector (square root of the sum of magnitudes square of the entries). In this paper we treat operators T that are bounded as maps between input and output spaces endowed with the quadratic norm (this corresponds to the L_∞ norm on the unit circle in the classical case). A sufficient condition for this is that the spectral radius of AZ is strictly less than one, in which case the Neumann series $I + AZ + (AZ)^2 + \cdots$ converges in norm. If that is the case we say that the realization for T is *uniformly exponentially stable* or *ues* (exponential stability of time-varying systems is extensively treated in the time-varying literature). To characterize this case further we define:

$$\ell_A = \sigma(ZA) = \lim_{n\to\infty} \|(ZA)^n\|^{1/n}$$

and a system realization will be ues if $\ell_A < 1$. The "Z" can be taken out of the formula for ℓ_A if we define the South-East diagonal shift (with "$*$" indicating the adjoint operator):

$$A^{(1)} = ZAZ^*$$

so that

$$\ell_A = \lim_{n\to\infty} \|AA^{(1)}A^{(2)}\cdots A^{(n-1)}\|^{1/n}.$$

The "continuous product" that appears in the formula is useful for other purposes. In particular, if we express the block entries in T in terms of a realization, we obtain, for $i > j$, $T_{i,j} = C_i A_{i-1}\cdots A_{j+1}B_j$, and we see that the entries become (uniformly) exponentially small for large $i-j$ when $\ell_A < 1$. In a later section, we shall see how we can recover a realization from the entries in T, but before doing so we turn to some more definitions and properties in the basic framework.

Lyapunov transformations

A state realization for an operator or matrix is not unique, even when it is minimal. In fact, we can permit ourselves a state transformation that introduces at each point k a transformed state x'_k related to the original via $x_k = R_k x'_k$ where the state transformation matrix R_k is non-singular for each k. In the case of infinite systems we usually require even more, namely R_k and R_k^{-1} should be uniformly bounded over k. Such transformations we call "Lyapunov transformations". They have the nice property that they are preserving the exponential stability of the realization. Under the state transformation, a causal realization transforms as follows:

$$\begin{bmatrix} A_k & B_k \\ C_k & D_k \end{bmatrix} \mapsto \begin{bmatrix} R_{k+1}^{-1}A_kR_k & R_{k+1}^{-1}B_k \\ C_kR_k & D_k \end{bmatrix}, \tag{52}$$

or, when expressed in the global diagonal notation:

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \mapsto \begin{bmatrix} (R^{(-1)})^{-1}AR & (R^{(-1)})^{-1}B \\ CR & D_k \end{bmatrix}.$$

State transformations are very important not only to achieve canonical representations discussed below, but also to obtain algebraically minimal calculations—see in this respect [24, Chap. 14].

Input/output normal forms

We say that a realization is in *output normal form* when

$$A^*A + C^*C = I$$

i.e., $A_k^*A_k + C_k^*C_k = I$ for each k. From (52) and putting $M_k = R_k^{-*}R_k^{-1}$, we see that a realization can be brought to output normal form if a bounded and invertible solution exists to the recursive set of *Lyapunov-Stein* equations

$$A_k^*M_{k+1}A_k + C_k^*C_k = M_k,$$

or, equivalently, if $A^*M^{(-1)}A + C^*C = M$ has a boundedly invertible diagonal operator M as a solution. The existence of the solution has been much studied in Lyapunov stability theory, we suffice here with some facts. If the original realization is ues (i.e., if $\ell_A < 1$), then the Lyapunov-Stein equation always has a bounded solution M. The solution M can be expressed as the so-called observability Gramian:

$$M = \sum_{k=0}^{\infty} (A^{\{k\}})^*(C^*C)^{(-k)}(A^{\{k\}}),$$

where we have put

$$A^{\{k\}} = A^{(-k+1)}\cdots A^{(-1)}A$$

and the sum converges in norm because of the ues assumption. The state transformation needed to bring the system in output normal is then obtained from $M^{-1} = RR^*$. The problem with its existence is whether M is boundedly invertible. We shall say that the system is *strictly observable* if that is the case. In the sequel we shall normally assume this property to be valid.

$$
\left[\begin{array}{ccc|cc}
\ddots & \ddots & & & \\
 & \ddots & T_{k,k} & & \\
\hline
 & T_{k+1,k-1} & T_{k+1,k} & T_{k+1,k+1} & \\
 & T_{k+2,k-1} & T_{k+2,k} & T_{k+2,k+1} & T_{k+2,k+2} \\
 & & & \ddots & \ddots
\end{array}\right]
$$

Figure 9: Generalized Hankel operators in a matrix or operator

Realization theory and canonical spaces

One may wonder when a causal transfer operator T has a finite-dimensional realization at each time point k. It turns out (see [24]) that this will be the case iff each k-th order operator

$$
H_k := \begin{bmatrix} T_{k+1,k} & T_{k+1,k-1} & T_{k+1,k-2} & \cdots \\ T_{k+2,k} & T_{k+2,k-1} & T_{k+2,k-2} & \cdots \\ T_{k+3,k} & T_{k+3,k-1} & T_{k+3,k-2} & \cdots \\ \vdots & \vdots & \vdots & \end{bmatrix}
$$

has finite rank δ_k. We call these operators local Hankel matrices, and their rank δ_k actually gives the minimal state dimension needed at point k. The here defined Hankel operators do not have the classical Hankel structure (elements equal along anti-diagonals), but they do fit the general functional definition of Hankel operators as exemplified in Figure 5.1, where the matrices are shown in a graphical way (notice that the columns in the picture are in reverse order, the definition of the H_k fits the classical matrix representation).

Realization theory shows that any collection of minimal factorizations of all H_k will produce a minimal realization. If we express the Hankel operators in terms of a state space representation we have:

$$
H_k = \mathcal{O}_k \mathcal{R}_k = \begin{bmatrix} C_k \\ C_{k+1} A_k \\ \vdots \end{bmatrix} \begin{bmatrix} B_{k-1} & A_{k-1} B_{k-2} & \cdots \end{bmatrix},
$$

and the "realization theory" is reduced to reading the A_k, B_k, C_k, D_k from the factorization. The columns of $\mathcal{O}_k$ form a basis for the columns of the Hankel matrix H_k while the rows of $\mathcal{R}_k$ form a basis for its rows. We shall obtain a realization in output normal form iff the columns of $\mathcal{O}_k$ have been chosen orthonormal for each k. The realization derived from the factorization is then given by:

$$
B_{k-1} = (\mathcal{R}_k)_1, \quad C_k = (\mathcal{O}_k)_0, \quad A_k = \mathcal{O}_{k+1}^{\dagger} \mathcal{O}_k^{\downarrow}
$$

where $(\mathcal{R}_k)_1$ is the first element of the "reachability" matrix $\mathcal{R}_k$, $(\mathcal{O}_k)_0$ the top element of the "observability" matrix $\mathcal{O}_k$, the "$\dagger$" indicates the Moore-Penrose inverse, and the "downarrow" on $\mathcal{O}_k$ indicates a matrix equal to $\mathcal{O}_k$ except for its first block-element, which has been deleted. The matrix A_k is uniquely defined because of the minimality of the factorization, even when any general inverse is used.

Balanced realization

It is also possible to define a balanced realization, by using a factorization based on a singular value decomposition of the Hankel operator:

$$H_k = U_k \begin{bmatrix} \sqrt{\sigma_1} & & \\ & \ddots & \\ & & \sqrt{\sigma_k} \end{bmatrix} \cdot \begin{bmatrix} \sqrt{\sigma_1} & & \\ & \ddots & \\ & & \sqrt{\sigma_k} \end{bmatrix} V_k.$$

However, balanced realizations and approximations are only of limited use in time-varying theory, they are unable to handle transfer operators of low rank with sparse entries far from the main diagonal adequately [24]. We give them here for the sake of completeness.

Reachability/observability bases in terms of realizations

It is easy to produce a direct relation between realizations and reachability or controllability bases, in particular we find:

$$\mathbf{F}_0 = C(I - ZA)^{-1} = \begin{bmatrix} \ddots & & & & \\ \ddots & C_{-1} & & & \\ \ddots & C_0 A_{-1} & \boxed{C_0} & & \cdots \\ \ddots & C_1 A_0 A_{-1} & C_1 A_0 & C_1 & \\ \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}$$

and dually

$$\mathbf{F} = B^* Z^* (I - A^* Z^*)^{-1}.$$

Each block column of $\mathbf{F}_o$ or $\mathbf{F}$ forms the basis for a local observability or controllability space.

5.2 Hankel-norm model reduction

We are given a lower (block-)operator T (we write: $T \in \mathcal{L}$) that we wish to approximate by a lower operator T_a of minimal complexity and that meets a certain pre-assigned complexity. First we make the notion "complexity" and "meeting a pre-assigned norm" more concrete.

Complexity

We identify "complexity" with "local state dimension". Suppose indeed that at stage k the state dimension (the total number of floating-point numbers the system has stored in memory from its past) is δ_k. Then it can be shown that that number, together with the dimensions of the local input and output space determines the local computational complexity. It turns out that the number of floating point operations needed at stage k is given by $\frac{1}{2}(m_k + n_k + \delta_k)(m_k + n_k + \delta_{k+1} + 1)$ [24], exactly equal to the number of "algebraically free parameters" at that stage.

Norm

What is an adequate approximating norm? In the classical model reduction context an L_∞-type norm is known to be too strong (because the polynomials or rationals are not dense in such a space), while an L_2 norm is usually too weak, because it gives rise to undesirable phenomena like the Gibbs phenomenon. A good compromise, one that also offers quite a bit of flexibility, is provided by the Hankel norm, i.e., the supremum of the norms of the local Hankel operators we defined before. This is the norm we shall be using, hence we define

$$\|T\|_H = \sup_k \|H_k\|.$$

We still need to characterize the approximation accuracy needed. We take as measure for precision a Hermitian, strictly positive diagonal operator Γ—in fact it could be taken as $\Gamma = \epsilon \cdot I$ for some small epsilon, but we may need the extra freedom of accommodating the precision at each time point.

High-order model

As described earlier, we start out our model reduction by selecting an appropriate representation of the desired computation as a high-complexity or high-order model that can be used computationally. An example of such a high-order model is given by a truncated Taylor-like series of high-enough order so that the truncation error has hardly any impact, but other, more convenient high-order representations may be adequate as well. If

$$T \approx T_0 + ZT_1 + Z^2T_2 + \cdots + Z^nT_n$$

(with n sufficiently large and where each T_k represents a shifted diagonal of T), then a simple but high-complexity realization for T is given by the generalized companion form (in formal output normal form)

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \left[\begin{array}{cccc|c} 0 & & & & T_n \\ I & 0 & & & T_{n-1} \\ & \ddots & \ddots & & \vdots \\ & & I & 0 & T_1 \\ \hline 0 & \cdots & 0 & I & T_0 \end{array}\right]. \tag{53}$$

Expression (53) should be interpreted as a matrix consisting of block diagonals. At time point k the local realization has the form

$$\begin{bmatrix} A_k & B_k \\ C_k & D_k \end{bmatrix} = \left[\begin{array}{cccc|c} 0 & & & & T_{n,k} \\ I & 0 & & & T_{n-1,k} \\ & \ddots & \ddots & & \vdots \\ & & I & 0 & T_{1,k} \\ \hline 0 & \cdots & 0 & I & T_{0,k} \end{array}\right].$$

Also the shift matrix Z must be interpreted in a block fashion and now has the form

$$\begin{bmatrix} Z & & & \\ & Z & & \\ & & \ddots & \\ & & & Z \end{bmatrix}$$

conformal with the block-diagonal decomposition of A. Given the higher model for T and the precision Γ, the model reduction problem becomes:
Find a causal operator T_a of minimal state complexity such that

$$\|(T - T_a)\Gamma^{-1}\|_H \leq 1,$$

i.e., T_a approximates T up to a precision given by Γ. It is customary to take the higher model T so that it is strictly causal, i.e., $T_0 = 0$ and to require the same of the low-order approximation. We follow that habit since it does not impair generality and simplifies some properties. Before embarking on the solution and its properties, we introduce the main ingredients needed.

Ingredient # 1: Nehari reduction

The Nehari theorem adapted to our context is as follows.

Theorem 11 *For any bounded, strictly causal operator T,*

$$\|T\|_H = \min_{T" \in \mathcal{U}} \|T + T"\|$$

where the norm in the second member is the operator norm and $T"$ is a bounded, anticausal operator.

A proof of the Nehari theorem in the general context of nest algebras (to which our setup conforms) goes back to the work of Arveson [6]. For a proof restricted to our specific context, see [24]. Application of the Nehari theorem reduces the problem to: *Find a (general) bounded operator T' so that its causal part $T_a = \mathbf{P}T'$ is of minimal complexity and*

$$\|(T - T')\Gamma^{-1}\| \leq 1.$$

Ingredient # 2: external factorization

We are given $T \in \mathcal{L}$. An "external factorization" consists of finding $\Delta \in \mathcal{L}$ and $U \in \mathcal{L}$ unitary such that $T = U\Delta^*$ (a more general type relaxes the requirement on U, see further). This type of factorization is reminiscent of the coprime factorization of classical system theory, where U is an all-pass function that collects the "poles" of T and Δ^* is obtained as U^*T - U^* pushes the poles of T to anticausality. It is easy to perform an external factorization on the state-space representation of T, especially when it is given in output normal form. So suppose that the realizations are given as (we use the $\approx$ sign to represent realizations).

$$\mathbf{T}_k \approx \begin{bmatrix} A_k & B_k \\ C_k & D_k \end{bmatrix}$$

in which, for all k,

$$A_k^* A_k + C_k^* C_k = I.$$

Then, the k-th realization matrix for U is found by completing the first block column to form unitary matrices:

$$\begin{bmatrix} A_k & B_{Uk} \\ C_k & D_{Uk} \end{bmatrix}$$

thereby producing B_{Uk} and D_{Uk} as completing matrices. The "remainder" Δ_k is then given by

$$\Delta_k \approx \begin{bmatrix} A_k & B_{Uk} \\ B_k^* A_k + D_k^* C_k & B_k^* B_{Uk} + D_k^* D_{Uk} \end{bmatrix}.$$

Algorithmically, a simplified "Householder-type" algorithm will provide the missing data. Numerical analysts would write, somewhat equivocally

$$\begin{bmatrix} B_{Uk} \\ D_{Uk} \end{bmatrix} = \begin{bmatrix} A_k \\ C_k \end{bmatrix}^{\perp}.$$

Ingredient # 3: J-unitary operators

In interpolation and approximation theory, J-unitary operators of various types play a central, if not crucial role. Causal J-unitary operators map input spaces of the type $\ell_2^{\mathcal{M}_1} \times \ell_2^{\mathcal{M}_2}$ to output spaces of the type $\ell_2^{\mathcal{N}_1} \times \ell_2^{\mathcal{N}_2}$, hence they are of the block type:

$$\Theta = \begin{bmatrix} \Theta_{11} & \Theta_{12} \\ \Theta_{21} & \Theta_{22} \end{bmatrix}.$$

These spaces are endowed with a non-definite metric. We denote

$$J_{\mathcal{M}} = \begin{bmatrix} I_{\mathcal{M}_1} & \\ & -I_{\mathcal{M}_2} \end{bmatrix}, \quad J_{\mathcal{N}} = \begin{bmatrix} I_{\mathcal{N}_1} & \\ & -I_{\mathcal{N}_2} \end{bmatrix}.$$

The J-unitary operators that we shall use will all be bounded and causal. The J-unitarity means

$$\Theta J_{\mathcal{N}} \Theta^* = J_{\mathcal{M}}, \quad \Theta^* J_{\mathcal{M}} \Theta = J_{\mathcal{N}}.$$

It has important consequences for the block entries of Θ:

- Θ_{22} is boundedly invertible and $\|\Theta_{22}^{-1}\| \ll 1$;
- $\|\Theta_{22}^{-1}\Theta_{21}\| \ll 1$.

The operator Θ_{22}^{-1} turns out to be of great importance in model-reduction theory. It is most likely of mixed type (causal/anti-causal). We return later to its state-space analysis.

Method of solution

With the ingredients previously detailed, the actual method to generate the solution appears very straightforward. It consists of two steps:

Step 1: Perform a coprime external factorization:

$$T = U\Delta^* \tag{54}$$

with $\Delta \in \mathcal{L}$ and $U \in \mathcal{L}$.

Step 2: Perform an external factorization of the type:

$$\Theta \begin{bmatrix} U^* \\ -\Gamma^{-1}T^* \end{bmatrix} = \begin{bmatrix} A' \\ -B' \end{bmatrix}. \tag{55}$$

Here, Θ is a block lower-triangular J-unitary operator of dimensions conforming to $\begin{bmatrix} U^* \\ -\Gamma^{-1}T^* \end{bmatrix}$, $A' \in \mathcal{L}$, and $B' \in \mathcal{L}$. The solution of the interpolation problem is now given by

$$\begin{aligned} T' &= \mathcal{B}'^* \Theta_{22}^{-*}\Gamma, \\ T_a &= \text{strictly lower part of } T'. \end{aligned} \tag{56}$$

Before embarking on computational issues, we show first that this recipe indeed produces a T' and a T_a that satisfies the norm and the minimality conditions. The norm condition is easy to treat directly. As to the study of complexity, it will be based on the state-space properties of the operator Θ appearing in the special J-unitary external factorization that have to be studied first.

The norm condition

From the second block row in (56), we obtain

$$\Theta_{21}U^* - \Gamma^{-1}\Theta_{22}T^* = -B',$$

and since Θ_{22} is invertible, it follows immediately by reordering of terms that

$$(T - T')\Gamma^{-1} = [\Theta_{22}^{-1}\Theta_{21}U^*]^*$$

where we have put $T' = B'^*\Theta_{22}^{-*}\Gamma$. Hence,

$$\|(T - T')\Gamma^{-1}\| < 1$$

since $\|U^*\| = 1$ and $\|\Theta_{22}^{-1}\Theta_{21}\| < 1$.

The construction of the special J-external factorization

We are looking for a minimal Θ that meets the factorization condition expressed in (55). As is the case of the regular external factorization, it will be based on the completion of appropriate reachability operators. A realization for $\begin{bmatrix} U & -T\Gamma^{-1} \end{bmatrix}$ is given by

$$\begin{bmatrix} A & B_U & -B\Gamma^{-1} \\ C & D_U & 0 \end{bmatrix}$$

whose reachability part is given by $\begin{bmatrix} A & B_U & -B\Gamma^{-1} \end{bmatrix}$, based on the realization

$$\begin{bmatrix} A & B_U \\ C & D_U \end{bmatrix}$$

for U (notice that in case one starts out with a companion form as detailed above, this part of the procedure is actually trivial, we simply have $U = Z^n$). From the realization theory we can deduce next that a bounded, causal ues J-unitary operator has the property that it possesses a realization which is J-unitary for some, still to be determined state signature

$$J_{\mathcal{B}} = \begin{bmatrix} I_{\mathcal{B}_+} & \\ & -I_{\mathcal{B}_-} \end{bmatrix}. \tag{57}$$

Hence, an appropriate state transformation should be able to produce the desired $J_{\mathcal{B}}$ and J-unitarity based on such a signature matrix on the state. As is the case for the regular external factorization, a somewhat special reachability Gramian will play a central role in finding this transformation. Indeed, let $\{R_k\}$ be the set of state-transformation matrices needed. Then the reachability matrices transform as

$$\begin{bmatrix} R_{k+1}^{-1}A_kR_k & R_{k+1}^{-1}(B_U)_k & -R_{k+1}^{-1}B\Gamma^{-1} \end{bmatrix},$$

and we wish each of these matrices to be part of a J-unitary matrix, i.e., they have each to be J-isometric for an adequate local signature matrix. Suppose that we already have the signature matrices $(J_{\mathcal{B}})_k$, and let $\Lambda_k = R_k(J_{\mathcal{B}})_kR_k^*$, then the J-unitarity of the Gramian can be expressed as follows:

$$A_k\Lambda_kA_k^* + (B_U)_k(B_U)_k^* - B_k\Gamma_k^{-2}B_k^* = \Lambda_{k+1}. \tag{58}$$

A solution for Λ will exist if this Lyapunov-Stein equation has a definite solution that is also boundedly invertible. Note that because of the ues condition on A, the equation has a unique bounded solution; the question is whether the solution is also boundedly invertible. The existence of the solution can be studied directly in terms of the original data by eliminating B_U, since

$$AA^* + B_UB_U^* = I.$$

Setting $M = I - \Lambda$ the equation turns into

$$M_{k+1} = A_k M_k A_k^* + B_k \Gamma_k^{-2} B_k^*.$$

Here, M is the reachability Gramian of $T\Gamma^{-1}$, and we find that a solution to the J-unitary embedding problem exists iff $(I - M)^{-1}$ exists and is bounded, i.e., iff the eigenvalues of M_k are bounded away from 1, uniformly over k. In the case the solution is not definite, a "borderline" solution may exist, and thus the case becomes singular. Although that singular case is beyond the present treatment, we shall devote some words to it in the discussion at the end of this section. Let us now assume that a strictly definite solution does exist and analyze it further. Let the inertia of Λ_k be given by

$$\Lambda_k = R_k \begin{bmatrix} (I_{\mathcal{B}_+})_k & \\ & -(I_{\mathcal{B}_-})_k \end{bmatrix} R_k^*.$$

After application of the state transformation $R_{k+1}^{-1} \cdots R_k$, the dataflow for Θ looks as in Figure 5.2.

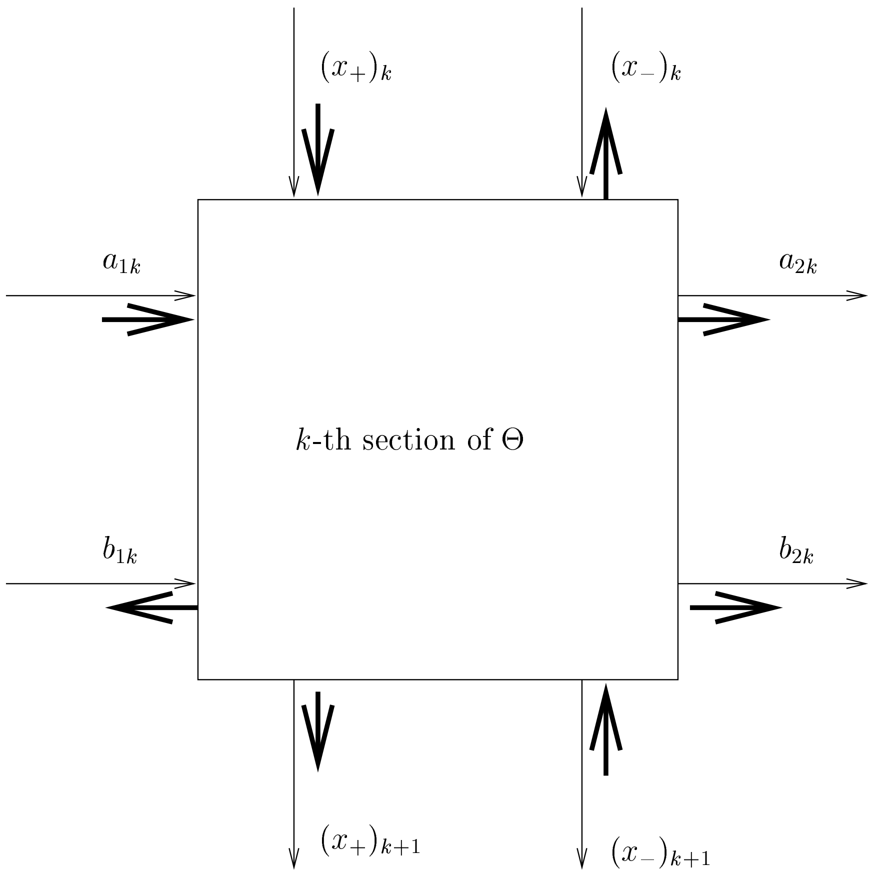

Figure 10: The dataflow in a Theta section is shown with normal arrows, the "energy flow" indicating the sign of the quadratic norms is indicated with fat arrows.

Associated with the various signature matrices, we can also imagine an "energy flow" representing the conservation of quadratic norm or energy which follows from the J-unitarity imposed on Θ. The energy flow corresponding to the signature is shown by fat arrows in Figure 5.2.

Complexity analysis

We have as proposed solution

$$T_a = \text{strictly causal part of } B'^* \Theta_{22}^{-*} \Gamma.$$

In this expression, B'^* is anticausal while Θ_{22}^{-*} is of mixed causality. We first establish that the complexity of T_a is essentially determined by the (strictly) causal part of Θ_{22}^{-*}. Next we shall analyze the complexity of the latter. Let

$$\begin{aligned} B' &= d + cZ(I - aZ)^{-1} b, \\ \text{causal part of } \Theta_{22}^{-*} &= D_2 + C_2 Z(I - A_2 Z)^{-1} B_2, \end{aligned}$$

be minimal realizations for B' and the causal part of Θ_{22}^{-*}, respectively (for the existence of the latter, see further). The computation of the causal part for the product is straightforward:

$$\begin{aligned} \text{causal part of } B'^* \Theta_{22}^{-*} \Gamma = {} & d^* D_2 \Gamma + d^* C_2 Z (I - A_2 Z)^{-1} B_2 \Gamma \\ & + \text{causal part of } b^* (I - Z^* a^*)^{-1} (c^* C_2)^{(-1)} (I - A_2 Z)^{-1} B_2 \Gamma. \end{aligned}$$

The computation reduces to the "generalized partial-fraction decomposition" of the last part. This is handled in the following generic lemma.

Lemma 1 *Let a and A_2 be transition operators with $\ell_a \leq 1$, $\ell_{A_2} \leq 1$ and at least one less than one, then*

$$(I - Z^* a^*)^{-1} (c^* C_2)^{(-1)} (A - A_2 Z)^{-1} = (I - Z^* a^*)^{-1} Z^* a^* m + m + m A_2 Z (I - A_2 Z)^{-1},$$

where m is the unique bounded solution of the Lyapunov-Stein equation

$$m^{(1)} = c^* C_2 + a^* m A_2.$$

Proof The proof of the lemma is by direct computation, after chasing the denominators and identifying the entries. □

Applying the lemma to the product that defines T_a, we obtain

$$T_a = (d^* D_2 \Gamma + b^* m B_2 \Gamma) + (d^* C_2 + b^* m)\, Z (I - A_2 Z)^{-1} B_2 \Gamma.$$

We see that T_a inherits the complexity of Θ_{22}^{-*}, at least essentially (further cancellations are theoretically possible but not very likely). In fact, they have the same reachability space based on $\{A_2, B_2\Gamma\}$. The complexity analysis hence proceeds with the analysis of the complexity of Θ_{22}^{-*}. This can be done in a particularly elegant way by studying the strict-past/future decomposition of the operator Θ. We decompose an arbitrary signal (say a belonging to some ℓ_2-space) in its strict-past part and its future part ($a_k = a_{pk} + a_{fk}$). Let the corresponding operators be denoted by $\mathbf{P}_k$ for the projection on the future and $\mathbf{P}'_k = I - \mathbf{P}_k$ for the projection on the strict past, then the splitting of the operator Θ happens as shown in Figure 5.2, where we also have indicated the sign decomposition of the state discussed earlier. The arrows in Figure 5.2 indicate flow of energy in the sense that each block satisfies the energy balance with respect to incoming and outgoing energetic contributions (isometric or J-isometric depending on whether a signal is considered an input or an output in the formulation at hand). The causal part of Θ_{22}^{-*} will of course correspond to the anticausal part of Θ_{22}^{-1}. Writing out

$$\Theta_{22}^{-1} = B_2^* Z^* (I - A_2^* Z^*)^{-1} C_2^* + D_2^* + \text{causal part},$$

we see that the relevant state dimension is given by the state dimension needed for the operator represented by the first term that produces the map b_{2f} to b_{1p} with $a_1 = 0$ and $b_{2p} = 0$ since Θ_{22}^{-1} maps b_2 to b_1 under the assumption $a_1 = 0$ and the portion b_{1f} in b_1 is to be neglected by the restriction to

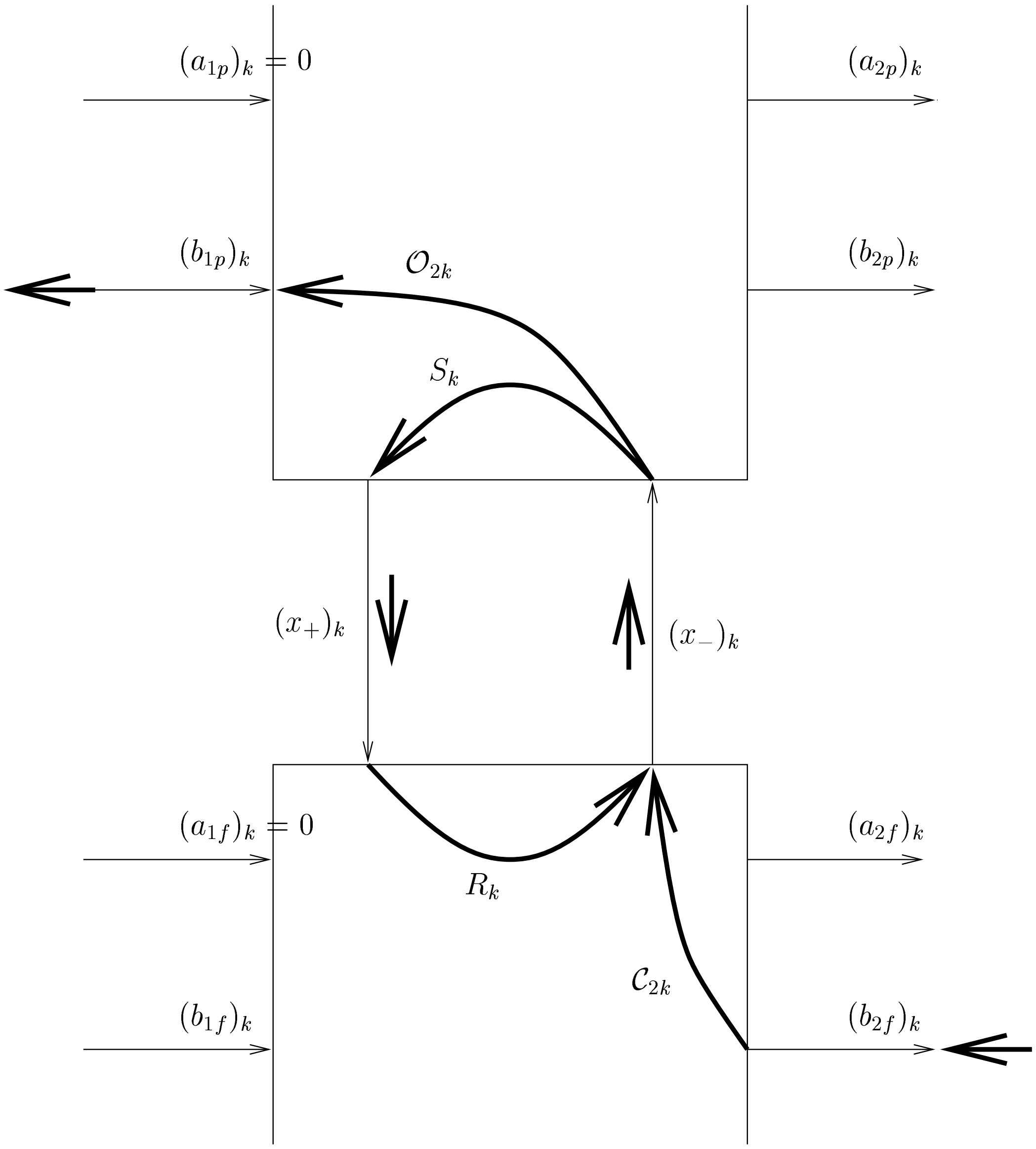

Figure 11: The figure shows the signal flow for $(\Theta_{22}^{-1})_k$. The energy flow of Figure 5.2 applies, here the relevant signal propagation is indicated with fat arrows.

the lower part of the result (with a slight abuse of notation we can handle all time points k in the same global formula—see [24] for details). With reference to the situation in Figure 5.2, let us define two new diagonal operators $S : x_- \mapsto x_+$ (in the past) and $R : x_+ \mapsto x_-$ (in the future). It is not hard to see (and a more detailed analysis would show) that both these operators are causal and strictly contractive. With b_{2f} as only non-zero input in this configuration, and with energy conservation in vigor, we see that both b_{1p} and x_+ are solely dependent on x_-. In fact, we have

$$\begin{aligned} x_- &= (I - RS)^{-1}\mathcal{C}_2 b_{2f}, \\ b_{1p} &= \mathcal{O}_2 x_-, \\ x_+ &= Sx_-, \end{aligned}$$

where $\mathcal{C}_2$ and $\mathcal{O}_2$ are appropriate reachability and observability maps derived from the anticausal part of Θ_{22}^{-1} (and which we do not detail any further here). The map from b_{2f} to b_{1p} then factors as

$$b_{2p} = \mathcal{O}_2 \cdot (I - RS)^{-1}\mathcal{R}_2 b_{2f},$$

and its state complexity is determined by the dimension of the "anticausal" state x_-. Hence, T_a has the same complexity as the strict lower part of Θ_{22}^{-*}, which is locally equal to the dimension δ_{k-} of x_-. This dimension is now easy to gauge from the original construction of Θ and is given by the following theorem.

Theorem 12 *Assuming that there exists an ϵ so that all singular values of all H_k, Hankel matrices of $T\Gamma^{-1}$, are at least ϵ distant from 1, the dimension δ_{k-} is given by the number of singular values of H_k larger than one. This is also the minimal dimension of any strictly causal approximant T_a satisfying $\|(T - T_a)\Gamma^{-1}\| < 1$.*

Proof Recall that the dimension of x_{k-} is given by the number of eigenvalues of M_k larger than one, where M_k satisfies

$$M_{k+1} = A_k M_k A_k^* + B_k \Gamma_k^{-2} B_k^*.$$

Since we started out with a system in output normal form, and $H_k = \mathcal{O}_k \mathcal{R}_k$, we have

$$H_k^* H_k = \mathcal{R}_k^* \mathcal{O}_k^* \mathcal{O}_k \mathcal{R}_k = \mathcal{R}_k^* \mathcal{R}_k = M_k$$

where $\mathcal{O}_k^* \mathcal{O}_k = I$ since we assumed the system in output normal form, and the singular values of H_k equal the eigenvalues of M_k. This proves the first statement. As for the second assertion, its proof is much more complex, and based on the fact that all approximants which meet the norm condition can be generated by loading Θ in a contractive and causal operator S_L, more precisely, all T' have the form

$$T' = T + US^*\Gamma.$$

Here,

$$S = (S_L\Theta_{21} + \Theta_{22})^{-1}(S_L\Theta_{11} + \Theta_{12}),$$

and U is as defined earlier. It turns out that the complexity of its lower part is then at least equal to the complexity of the lower part of Θ_{22}^{-*}. For a complete treatment, see [24, Chap. 10], in particular Theorem 10.18. □

These are the basic results on Hankel-norm approximation of a lower operator. Many more properties can be derived on this new and interesting method, in particular, state-space representations for the approximants are relatively easy to derive, for details we refer to the literature cited.

5.3 The recursive Schur algorithm for Hankel-norm approximation

A low-complexity Hankel-norm approximation to a strictly upper but otherwise general matrix can be derived from an elementary Schur-type elimination algorithm using both orthogonal and hyperbolic elementary matrices. It is a direct application of the previous theory to finite matrices and was first presented in [24, pp. 292ff]. We give the result without proof.

Suppose that the original matrix to be approximated is given by

$$T = \begin{bmatrix} \boxed{0} & & & \\ t_{21} & \underline{0} & & \\ \vdots & & \ddots & \\ t_{n1} & t_{n2} & \cdots & \underline{0} \end{bmatrix},$$

then a trivial external factorization for $T = U\Delta^*$ is given by

$$U = \begin{bmatrix} \boxed{1\,0\,0\,0} \\ 0\,1\,0\,0 \\ 0\,0\,1\,0 \\ 0\,0\,0\,1 \end{bmatrix}, \quad \Delta = \begin{bmatrix} \boxed{0 \; t_{21}^* \; \cdots \; t_{n1}^*} \\ 0 \;\; \cdots \;\; t_{n2}^* \\ \ddots \;\; \vdots \\ 0 \end{bmatrix}.$$

According to the theory in the previous sections, the Θ matrix necessary for the Hankel-norm approximation must now have the following three properties:
1. It must be J-unitary for appropriate signature matrices;
2. It must be block lower;
3. It must make the product

$$\Theta \begin{bmatrix} U^* \\ -T^* \end{bmatrix}$$

lower. (Point 1 may seem cryptic but will be partly justified in the sequel.)

The right-hand side signature of Θ is certainly given by $J_2 = I_n \oplus -I_n$, in accordance with the right factor, the left-hand side signature will follow from the construction and will differ case by case. It is possible at this point to determine the local arrow dimensions of Θ but not yet the signs of the state and output arrows. To illustrate the point, let us assume that the entries in T are scalar. Because of the structure of U^* and T^*, the first block in a realization for Θ will have n positive inputs (from U^*) and one negative input (from $-T^*$), and it will have $n-1$ states going to the next stage. This means that this first stage must have two outputs (the signs of the outgoing states and outputs are yet to be determined—see Figure 5.2 for extra information).

The matrix to be block lowered using elementary operations is given by:

$$\begin{bmatrix} U^* \\ -T^* \end{bmatrix} = \begin{array}{c} + \\ + \\ \vdots \\ + \\ - \\ - \\ \vdots \\ - \end{array} \left[\begin{array}{cccc} 1 & 0 & & \\ 0 & 1 & & \\ \vdots & \vdots & \ddots & \\ 0 & 0 & \cdots & 1 \\ \hline \boxed{-t_{11}^*} & -t_{21}^* & \cdots & -t_{n1}^* \\ 0 & -t_{22}^* & \cdots & -t_{n2}^* \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & -t_{nn}^* \end{array}\right].$$

The elimination procedure now starts with the elimination of $-t_{n1}^*$ in the first row of the second block, using the last row in the first block. We indicate this state of affairs with the pair of indices $< n, 1 >$. Since the sign of the last row of the first block is positive, and that of the first row of the second block negative, a hyperbolic rotation must be used, which can be of two forms, depending on the magnitude of $-t_{n1}^*$. One possibility is

$$\frac{1}{\sqrt{1-|\rho_{n1}|^2}}\begin{bmatrix} 1 & \overline{\rho_{n1}} \\ \rho_{n1} & 1 \end{bmatrix},$$

in which case $\rho_{n1} = t_{n1}$ has to be smaller than one in magnitude and the target signature is $<+,->$, while the other possibility, when $|t_{n1}| > 1$, is

$$\frac{1}{\sqrt{1-|\rho_{n1}|^2}}\begin{bmatrix} \rho_{n1} & 1 \\ 1 & \overline{\rho_{n1}} \end{bmatrix},$$

in which case $\rho_{n1} = 1/t_{n1}$, again of magnitude smaller than one. The case where $|t_{n1}| = 1$ is not allowable in the present state of the theory (for an extension, see [21]), the respective coefficient in Γ then has to be adapted (the condition on the singular values of the Hankel operator is not satisfied). It may happen that in the course of the elimination procedure, a signature of the type $<+,+>$ or $<-,->$ is encountered. In that case a regular (unitary) Jacobi rotation will do, and if $<-,+>$ as initial signature is found, then the mirror case of the case detailed above holds. The type of rotations used in the scheme will determine the actual flow of energy between the stages of the realization for Θ. The resulting complexity can also be deduced directly from the signature resulting at the output. For example, if the output sequence is $<+,->, <+,->, \ldots$, then all state transitions have positive signs and $\Theta_2 2$ is causally invertible. The low-complexity approximant then reduces to a diagonal matrix. At the opposite side, and taking for example the 4×4 case, the output sequence $<+,+>$, $<+,+>$, $<-,->$, $<-,->$ will result in a state sequence given by $<+,+,->$, $<-,->$, $<->$, resulting in an "approximant" of maximal complexity. The principle involved is that at each state there must be an equal number of incoming and outgoing arrows on the one hand, and an equal number of incoming and outgoing energy arrows as well. A connection will bear a "+" sign if the two arrows point in the same direction and a "-" sign in the opposite case. From the resulting diagram, a realization for Θ_{22}^{-*} can be derived, and from there a realization for the approximant T_a, we refer to the literature cited for details. Although the algorithm does provide for an optimal solution, the computational details are still somewhat extensive.

6 Second-order linear dynamical systems

Second-order models arise naturally in the study of many types of physical systems, such as electrical and mechanical systems. A *time-invariant multi-input multi-output second-order system* is described by equations of the form

$$M\frac{\mathrm{d}^2 q}{\mathrm{d}t^2} + D\frac{\mathrm{d}q}{\mathrm{d}t} + Kq = Pu(t), \tag{59}$$

$$y(t) = L^T q(t), \tag{60}$$

together with initial conditions $q(0) = q_0$ and $\frac{\mathrm{d}q}{\mathrm{d}t}(0) = \dot{q}_0$. Here, $q(t) \in \mathbb{R}^N$ is the vector of state variables, $u(t) \in \mathbb{R}^m$ is the input force vector, and $y(t) \in \mathbb{R}^p$ is the output measurement vector. Moreover, M, D, $K \in \mathbb{R}^{N\times N}$ are system matrices, such as mass, damping, and stiffness matrices in structural dynamics, $P \in \mathbb{R}^{N\times m}$ is the input distribution matrix, and $L \in \mathbb{R}^{N\times p}$ is the output measurement matrix. Finally, N is the state-space dimension, and m and p are the number of inputs and outputs, respectively. In most practical cases, m and p are much smaller than N.

The second-order system (59) and (60) can be reformulated as an equivalent linear first-order system in many different ways. We will use the following equivalent linear system:

$$E\frac{\mathrm{d}x}{\mathrm{d}t} = Ax + Bu(t), \tag{61}$$

$$y(t) = C^T x(t), \tag{62}$$

where

$$x = \begin{bmatrix} q \\ \frac{\mathrm{d}q}{\mathrm{d}t} \end{bmatrix}, \quad A = \begin{bmatrix} -K & 0 \\ 0 & W \end{bmatrix}, \quad E = \begin{bmatrix} D & M \\ W & 0 \end{bmatrix}, \quad B = \begin{bmatrix} P \\ 0 \end{bmatrix}, \quad C = \begin{bmatrix} L \\ 0 \end{bmatrix}.$$

Here, $W \in \mathbb{R}^{N \times N}$ can be any nonsingular matrix. A common choice is the identity matrix, $W = I$. If the matrices M, D, and K are all symmetric and M is nonsingular, as it is often the case in structural dynamics, we can choose $W = M$. The resulting matrices A and E in the linearized system (61) are then symmetric, and thus preserve the symmetry of the original second-order system.

Assume that, for simplicity, we have zero initial conditions, i.e., $q(0) = q_0$, $\frac{\mathrm{d}q}{\mathrm{d}t}(0) = 0$, and $u(0) = 0$ in (59) and (60). Then, by taking the Laplace transform of (59) and (60), we obtain the following system:

$$\begin{aligned} s^2 MQ(s) + DQ(s) + KQ(s) &= PU(s), \\ Y(s) &= L^T Q(s). \end{aligned}$$

Eliminating $Q(s)$ results in the frequency-domain input-output relation $Y(s) = H(s)U(s)$, where

$$H(s) := L^T \left(s^2 M + sD + K\right)^{-1} P$$

is the transfer function. In view of the equivalent linearized system (61) and (62), the transfer function can also be written as

$$H(s) = C^T (sE - A)^{-1} B.$$

If the matrix K in (59) is nonsingular, then $s_0 = 0$ is guaranteed not to be a pole of H. In this case, H can be expanded about $s_0 = 0$ as follows:

$$H(s) = M_0 + M_1 s + M_2 s^2 + \cdots,$$

where the matrices M_j are the so-called *low-frequency moments*. In terms of the matrices of the linearized system (61) and (62), the moments are given by

$$M_j = -C^T \left(A^{-1} E\right)^j A^{-1} B, \quad j = 0, 1, 2, \ldots.$$

6.1 Frequency-response analysis methods

In this subsection, we describe the use of eigensystem analysis to tackle the second-order system (59) and (60) directly.

We assume that the input force vector $u(t)$ of (59) is time-harmonic:

$$u(t) = \tilde{u}(\omega) e^{\mathrm{i}\omega t},$$

where ω is the frequency of the system. Correspondingly, we assume that the state variables of the second-order system can be represented as follows:

$$q(t) = \tilde{q}(\omega) e^{\mathrm{i}\omega t}.$$

The problem of solving the system of second-order differential equations (59) then reduces to solving the parameterized linear system of equations

$$\left(-\omega^2 M + \mathrm{i}\omega D + K\right) \tilde{q}(\omega) = P \tilde{u}(\omega) \tag{63}$$

for $\tilde{q}(\omega)$. This approach is called the *direct frequency-response analysis method.* For a given frequency ω_0, one can use a linear system solver, either direct or iterative, to obtain the desired vector $\tilde{q}(\omega_0)$.

Alternatively, we can try to reduce the cost of solving the large-scale parameterized linear system of equations (63) by first applying an eigensystem analysis. This approach is called the *modal frequency-response analysis* in structural dynamics. The basic idea is to first transfer the coordinates $\tilde{q}(\omega)$ of the state vector $q(t)$ to new coordinates $p(\omega)$ as follows:

$$q(t) \cong W_k p(\omega) e^{i\omega t}.$$

Here, W_k consists of k selected modal shapes to retain the modes whose resonant frequencies lie within the range of forcing frequencies. More precisely, W_k consists of k selected eigenvectors of the underlying quadratic eigenvalue problem $\left(\lambda^2 M + \lambda D + K\right) w = 0$. Equation (63) is then approximated by

$$\left(-\omega^2 M W_k + \mathtt{i}\omega D W_k + K W_k\right) p(\omega) = P \tilde{u}(\omega).$$

Multiplying this equation from the left by W_k^T, we obtain a $k \times k$ parameterized linear system of equations for $p(\omega)$:

$$\left(-\omega^2 \left(W_k^T M W_k\right) + \mathtt{i}\omega \left(W_k^T D W_k\right) + \left(W_k^T K W_k\right)\right) p(\omega) = W_k^T P(\omega).$$

Typically, $k \ll n$. The main question now is how to obtain the desired modal shapes W_k. One possibility is to simply extract W_k from the matrix pair (M, K) by ignoring the contribution of the damping term. This is called the *modal superposition method* in structural dynamics. This approach is applicable under the assumption that the damping term is of a certain form. For example, this is the case for so-called Rayleigh damping $D = \alpha M + \beta K$, where α and β are scalars [17]. In general, however, one may need to solve the full quadratic eigenvalue problem $\left(\lambda^2 M + \lambda D + K\right) w = 0$ in order to obtain the desired modal shapes W_k. Some of these techniques have been reviewed in the recent survey paper [55] on the quadratic eigenvalue problem.

6.2 Reduced-order modeling based on linearization

An obvious approach to constructing reduced-order models of the second-order system (59) and (60) is to apply any of the model-reduction techniques for linear systems to the linearized system (61) and (62). In particular, we can employ the Krylov-subspace techniques discussed in Section 3.

The resulting approach can be summarized as follows:

1. Linearize the second-order system (59) and (60) by properly defining the $2N \times 2N$ matrices A and E of the equivalent linear system (61) and (62). Select an expansion point s_0 "close" to the frequency range of interest and such that the matrix $A - s_0 E$ is nonsingular.

2. Apply a suitable Krylov process, such as the nonsymmetric band Lanczos algorithm described in Section 3.2, to the matrix $M := \left(A - s_0 E\right)^{-1} E$ and the blocks of right and left starting vectors $R := \left(A - s_0 E\right)^{-1} B$ and $L := C$ to obtain bi-orthogonal Lanczos basis matrices V_n and W_n for the n-th right and left block-Krylov subspaces $\mathcal{K}_n(M, R)$ and $\mathcal{K}_n(M^T, L)$.

3. Approximate the state vector $x(t)$ by $V_n z(t)$ where $z(t)$ is determined by the following linear reduced-order model of the linear system (61) and (62):

$$\begin{aligned} E_n \frac{\mathrm{d}z}{\mathrm{d}t} &= A_n z + B_n u(t), \\ y(t) &= C_n^T z(t). \end{aligned}$$

   Here, $E_n = T_n$, $A_n = \Delta_n + s_0 T_n$, $B_n = \rho_n^{(\mathrm{pr})}$, $C_n = \eta_n^{(\mathrm{pr})}$, and T_n, Δ_n, $\rho_n^{(\mathrm{pr})}$, $\eta_n^{(\mathrm{pr})}$ are the matrices generated by the nonsymmetric band Lanczos algorithm.

In Figure 12, we show the results of this approach applied to the linear-drive multi-mode resonator structure described in [16]. The solid lines are the Bode plots of the frequency response of the original second-order system, which is of dimension $N = 63$. The dashed line in the left, respectively right, plot

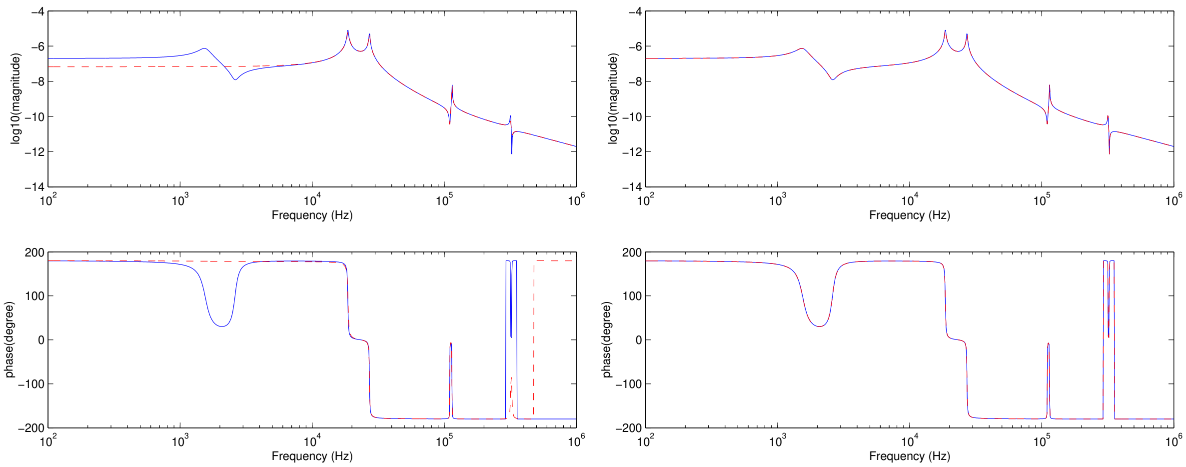

Figure 12: Bode plots for the original system and the reduced-order model of dimension $n = 8$ (left) and $n = 12$ (right)

is the Bode plot of the frequency response of the reduced-order model of dimension $n = 8$, respectively $n = 12$. The relative error between the transfer functions of the original system and the reduced-order model of dimension $n = 12$ is less than 10^{-4} over the frequency range shown in Figure 12.

There are a couple of advantages of the linearization approach. First, one can directly employ existing reduced-order modeling techniques developed for linear systems. Second, one can also exploit the structures of the linearized system matrices A and E in a Krylov process to reduce the computational cost. However, the linearization approach also has disadvantages. In particular, it ignores the physical meaning of the original system matrices, and more importantly, the reduced-order models are no longer in a second-order form. For engineering design and control of structural systems, it is often desirable to have reduced-order models that preserve the second-order form; see, e.g. [53].

6.3 Reduced-order modeling based on second-order systems

In this section, we discuss a Krylov-subspace technique that produces a reduced-order model of second-order form. This approach is based on the work [53].

The key observation is the following. In view of the linearization (61) and (62) of the second-order system (59) and (60), the desired Krylov subspace for reduced-order modeling is

$$\operatorname{span}\left\{ \widetilde{B}, \left(A^{-1}E\right)\widetilde{B}, \left(A^{-1}E\right)^2\widetilde{B}, \ldots, \left(A^{-1}E\right)^{n-1}\widetilde{B} \right\}.$$

Here, $\widetilde{B} := -A^{-1}\begin{bmatrix} B & C \end{bmatrix}$. Moreover, we have assumed that the matrix A in (61) is nonsingular. Let us set

$$R_j = \begin{bmatrix} R_j^d \\ R_j^v \end{bmatrix} := \left(-A^{-1}E\right)^j \widetilde{B},$$

where R_j^d is the vector of length N corresponding to the displacement portion of the vector R_j, and R_j^v is the vector of length N corresponding to the velocity portion of the vector R_j, see [53]. Then, in view of the structure of the matrices A and E, we have

$$\begin{bmatrix} R_j^d \\ R_j^v \end{bmatrix} = \left(-A^{-1}E\right)\begin{bmatrix} R_{j-1}^d \\ R_{j-1}^v \end{bmatrix} = \begin{bmatrix} K^{-1}DR_{j-1}^d + K^{-1}MR_{j-1}^d \\ -R_{j-1}^d \end{bmatrix}.$$

Note that the j-th velocity-portion vector R_j^v is the same (up to its sign) as the $(j-1)$-st displacement-portion vector R_{j-1}^d. In other words, the second portion R_j^v of R_j is the "one-step" delay of the first portion R^d_{j-1} of R_j. This suggests that one may simply choose

$$\operatorname{span}\{\, R^d_0, R^d_1, R^d_2, \ldots, R^d_{n-1} \,\} \tag{64}$$

as the projection subspace used for reduced-order modeling.

In practice, for numerical stability, one may opt to employ the Arnoldi process to generate an orthonormal basis Q_n of the subspace (64). The resulting procedure can be summarized as follows.

Algorithm 3 (Algorithm by Su and Craig Jr.)

0) (Initialization)
Set $R^d_0 = K^{-1}\begin{bmatrix} P & L \end{bmatrix}$, $R^v_0 = 0$, $U_0 S_0 V_0^T = (R^d_0)^T K R^d_0$ (by computing an SVD),
$Q^d_1 = R^d_0 U_0 S_0^{-1/2}$, *and* $Q^v_1 = 0$.

1) (Arnoldi loop)
For $j = 1, 2, \ldots, n-1$ *do*:

$$\textit{Set } R^d_j = K^{-1}\left(D Q^d_{j-1} + M Q^v_{j-1}\right) \textit{ and } R^v_j = -Q^d_{j-1}.$$

2) (Orthogonalization)
For $i = 1, 2, \ldots, j$ *do*:

$$\textit{Set } T_i = (Q^d_i)^T K R^d_j,\ R^d_j = R^d_j - Q^d_i T_i, \textit{ and } R^v_j = R^v_j - Q^v_i T_i.$$

2) (Normalization)
Set $U_0 S_0 V_0^T = (R^d_j)^T K R^d_j$ (by computing an SVD),
$Q^d_{j+1} = R^d_j U_0 S_0^{-1/2}$, *and* $Q^v_{j+1} = R^v_j U_0 S_0^{-1/2}$.

An approximation of the state vector $q(t)$ can then be obtained by constraining $q(t)$ to the subspace spanned by the columns of Q_n, i.e., $q(t) \approx Q_n z(t)$. Moreover, the reduced-order state vector $z(t)$ is defined as the solution of the following second-order system:

$$M_n \frac{\mathrm{d}^2 q}{\mathrm{d}t^2} + D_n \frac{\mathrm{d}q}{\mathrm{d}t} + K_n q = P_n u(t), \tag{65}$$

$$y(t) = L_n^T q(t), \tag{66}$$

where $M_n := Q_n^T M Q_n$, $D_n := Q_n^T D Q_n$, $K_n := Q_n^T K Q_n$, $P_n := Q_n^T P$, and $L_n := Q_n^T L$. Note that (65) and (66) is a reduced-order model in second-order form of the original second-order system (59) and (60).

In [53], a number of advantages of this approach are described. Here, we present some numerical results of a frequency-response analysis of a second-order system of order $N = 400$, which arises from a finite-element model of a shaft on bearing support with a damper. In the top of Figure 13, we plot the magnitudes of the transfer function H computed exactly, approximated by the model-superposition (MSP) method, and approximated by the Krylov-subspace technique (ROM). For the MSP method, we used the 80 modal shapes W_{80} from the matrix pencil (M, K). The reduced-order model (65) and (66) is also of dimension $n = 80$. The bottom plot of Figure 13 shows the relative errors between the exact transfer function and its approximations based on the MSP method (dash-dotted line) and the ROM method (dashed line). The plots indicate that no accuracy has been lost by the Krylov subspace-based method.

7 Semi-second-order dynamical systems

In some applications, in particular in the simulation of MEMS devices [52], the underlying mathematical models are second-order systems with nonlinear excitation forces of the following type:

$$M \frac{\mathrm{d}^2 q}{\mathrm{d}t^2} + D \frac{\mathrm{d}q}{\mathrm{d}t} + K q = P u\Big(q, \frac{\mathrm{d}q}{\mathrm{d}t}, t\Big), \tag{67}$$

$$y(t) = L^T q(t). \tag{68}$$

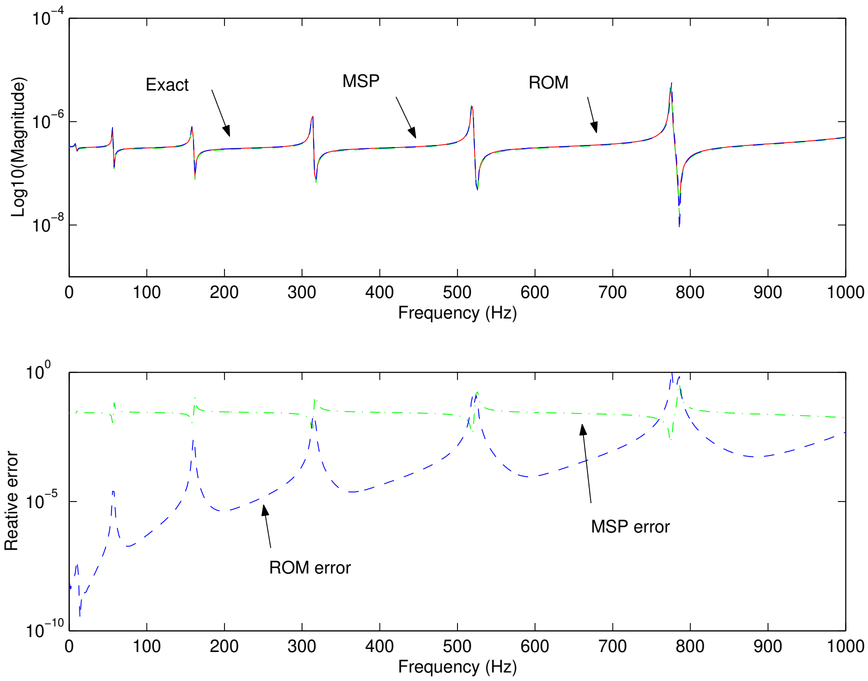

Figure 13: Frequency-response analysis (top plot) and relative errors (bottom plot) of a finite-element model of a shaft

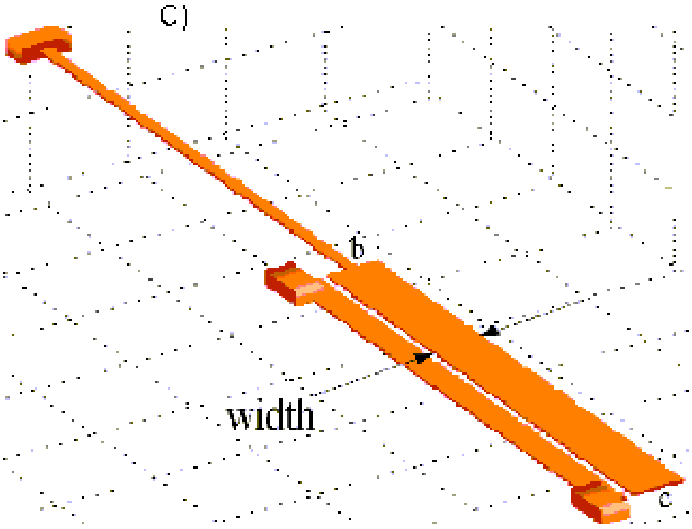

Figure 14: Electrostatic gap-closing actuator

Here, the system matrices M, D, K, P, and L have the same interpretation as in the standard second-order system (59) and (60). However, excitation force u is now a nonlinear function of q, and possibly $\frac{dq}{dt}$.

Systems of the form (67) and (68) are called *semi-second-order* time-invariant multi-input multi-output linear dynamical systems. Such systems are used as the underlying mathematical models in SUGAR [54], which is a system-level simulation package for MEMS devices. For example, Figure 14 shows a simple electrostatic gap-closing actuator, which is used as a demo in SUGAR. In this case,

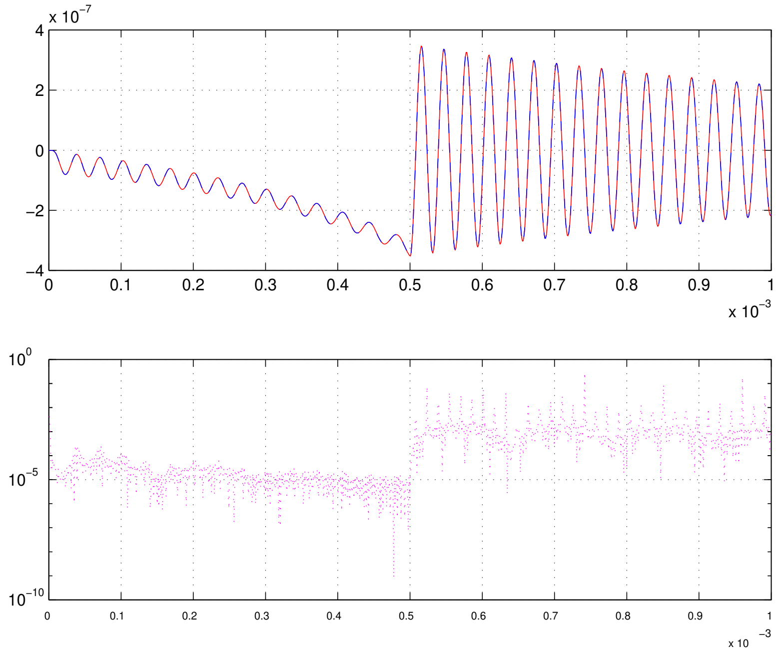

Figure 15: Transient responses of the gap-closing actuator

the excitation force u includes the electrostatic potential between the plates and is proportional to $(v(t)/\operatorname{gap}(q))^2$, where $v(t)$ is the voltage between electrodes and $\operatorname{gap}(q)$ is a scalar function of q for the distance between the two place electrodes. For mode details about the model used for the electrostatic gap-closing actuator, see [7].

Instead of treating the semi-second-order system (67) and (68) as a general nonlinear system, we can exploit the structure of the system and apply the idea of "nonlinear dynamics using linear modes". This approach is suggested in [2], where a non-damped system, i.e., $D = 0$ is considered and the eigenmodes of M and K are used to extract a reduced-order model. In [7], we described a Krylov-subspace based reduced-order modeling technique for systems (67) and (68). The idea is to first ignore the nonlinearity in the force term u, and treat the system as a second-order system. Using the approach discussed in Section 6.2, a projection space V_n is constructed, which may be regarded as the *linear Krylov modes.* The vector q is then expanded in terms of the constructed subspace, namely $q(t) \approx V_n z(t)$, and we obtain the following reduced-order model in terms of the vector $z(t)$:

$$
\begin{aligned}
E_n \frac{\mathrm{d}z}{\mathrm{d}t} &= A_n x + B_n u(V_n z(t), t), \\
y(t) &= C_n^T z(t).
\end{aligned}
$$

Here, the definitions of E_n, A_n, B_n, and C_n are the same as in Section 6.2. Note that the excitation force term $u(q,t)$ of the full-order system is replaced by $u(V_n z(t), t)$ in the reduced-order model. When the reduce-order model is solved by a numerical method, it is necessary that $u(V_n z_j, t)$ can be evaluated for the given z_j, which may be regarded as the approximation of $z(t)$ at time step $t = t_j$.

In Figure 15, we illustrate this approach for the transient analysis of the electrostatic gap-closing actuator shown in Figure 14. The first plot shows the output $y(t)$ of the original system and the output $y(\tilde{t})$ of the reduced-order system of dimension $n = 6$. The original systems has dimension $N = 30$. The second plot shows the accuracy of the reduced-order model of dimension $n = 6$ in terms of the relative error $\|y(t) - \tilde{y}(t)\|/\|y(t)\|$.

We remark that, as indicated in [40], the use of linear (eigen or Krylov) modes may not adequately capture all the features of nonlinear behavior. It is the subject of current research to further understand the approach sketched in this section and its limitations.

8 Concluding Remarks

We presented a survey of the most common techniques for reduced-order modeling of large-scale linear dynamical systems. By and large, the area of linear reduced-order modeling is fairly well explored, and we have a number of efficient techniques at our disposal. Still, some open problems remain. One such problem is the construction of reduced-order models that preserve stability or passivity and at the same time, have optimal approximation properties. In particular in circuit simulation, reduced-order modeling is used to substitute large linear subsystems within the simulation of even larger, in general nonlinear systems. It would be important to better understand the effects of these substitutions on the overall nonlinear simulation.

Finally, the systems arising in the simulation of electronic circuits are nonlinear in general, and it would be highly desirable to apply nonlinear reduced-order modeling techniques directly to these nonlinear systems. However, the area of nonlinear reduced-order modeling is in its infancy compared to the state-of-the-art of linear reduced-order modeling. We expect that further progress in model reduction will mainly occur in the area of nonlinear reduced-order modeling.

References

[1] J. I. Aliaga, D. L. Boley, R. W. Freund, and V. Hernández. A Lanczos-type method for multiple starting vectors. *Math. Comp.*, 69:1577–1601, 2000.

[2] G. K. Ananthasuresh, R. K. Gupta, and S. D. Senturia. An apporach to macromodeling of MEMS for nonlinear dynamic simulation. In *Microelectromechanical Systems (MEMS)*, volume 59 of *ASME Dynamics Systems & Control (DSC) ser.*, pages 401–407, 1996.

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

[4] B. D. O. Anderson and S. Vongpanitlerd. *Network Analysis and Synthesis.* Prentice-Hall, Englewood Cliffs, New Jersey, 1973.

[5] W. E. Arnoldi. The principle of minimized iterations in the solution of the matrix eigenvalue problem. *Quart. Appl. Math.*, 9:17–29, 1951.

[6] W. Arveson. Interpolation problems in nest algebras. *J. Functional Anal.*, 20:208–233, 1975.

[7] Z. Bai, D. Bindel, J. Clark, J. Demmel, K. S. J. Pister, and N. Zhou. New numerical techniques and tools in SUGAR for 3D MEMS simulation. In *Technical Proceedings of the Fourth International Conference on Modeling and Simulation of Microsystems*, pages 31–34, 2000.

[8] Z. Bai, P. Feldmann, and R. W. Freund. How to make theoretically passive reduced-order models passive in practice. In *Proc. IEEE 1998 Custom Integrated Circuits Conference*, pages 207–210, Piscataway, New Jersey, 1998. IEEE.

[9] Z. Bai and R. W. Freund. Eigenvalue-based characterization and test for positive realness of scalar transfer functions. *IEEE Trans. Automat. Control*, 45(12):2396–2402, December 2000.

[10] Z. Bai and R. W. Freund. A partial Padé-via-Lanczos method for reduced-order modeling. *Linear Algebra Appl.*, 332–334:139–164, 2001.

[11] Z. Bai and R. W. Freund. A symmetric band Lanczos process based on coupled recurrences and some applications. *SIAM J. Sci. Comput.*, 23(2):542–562, 2001.

[12] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory.* SIAM Publications, Philadelphia, Pennsylvania, 1994.

[13] S. L. Campbell. *Singular systems of differential equations.* Pitman, London, United Kingdom, 1980.

[14] S. L. Campbell. *Singular systems of differential equations II.* Pitman, London, United Kingdom, 1982.

[15] P. M. Chirlian. *Integrated and Active Network Analysis and Synthesis.* Prentice-Hall, Englewood Cliffs, New Jersey, 1967.

[16] J. V. Clark, N. Zhou, and K. S. J. Pister. MEMS simulation using SUGAR v0.5. In *Proc. Solid-State Sensors and Actuators Workshop*, pages 191–196, Hilton Head Island, SC, 1998.

[17] R. W. Clough and J. Penzien. *Dynamics of Structures.* McGraw-Hill, 1975.

[18] L. Dai. *Singular Control Systems*, volume 118 of *Lecture Notes in Control and Information Sciences.* Springer-Verlag, Berlin, Germany, 1989.

[19] E. Deprettere. Mixed-form time-variant lattice recursions. In *Outils et Modèles Mathématiques pour l'Automatique, l'Analyse de Systèmes et le Traitement du Signal*, pages 545–562. CNRS, Paris, 1981.

[20] P. Dewilde. New algebraic methods for modeling large-scale integrated circuits. Circuit Theory and Appl., 16:473–503, 1988.

[21] P. Dewilde. J-unitary matrices for algebraic approximation and interpolation — the singular case. In M. Moonen and B. D. Moor, editors, *SVD and Signal Processing, III, Algorithms, Architectures and Applications*, pages 209–223. Elsevier, 1995.

[22] P. Dewilde and E. F. Deprettere. Approximate inversion of positive matrices with applications to modelling. In R. F. Curtain, editor, *Modelling, Robustness and Sensitivity Reduction in Control Systems*, volume F34 of *NATO ASI Series*, pages 211–238, Berlin, 1987. Springer-Verlag.

[23] P. Dewilde and H. Dym. Schur recursions, error formulas, and convergence of rational estimators for stationary stochastic sequences. *IEEE Trans. Informat. Th.*, 27(4):446–461, 1981.

[24] P. Dewilde and A.-J. van der Veen. *Time-Varying Systems and Computations.* Kluwer, 1998.

[25] P. Dewilde, A. Vieira, and T.Kailath. On a generalized Szegö-Levinson realization algorithm for optimal linear predictors based on a network synthesis approach. *IEEE Trans. Circuits Syst.*, 25(9):663–675, 1978.

[26] P. Feldmann and R. W. Freund. Efficient linear circuit analysis by Padé approximation via the Lanczos process. In *Proceedings of EURO-DAC '94 with EURO-VHDL '94*, pages 170–175, Los Alamitos, California, 1994. IEEE Computer Society Press.

[27] P. Feldmann and R. W. Freund. Efficient linear circuit analysis by Padé approximation via the Lanczos process. *IEEE Trans. Computer-Aided Design*, 14:639–649, 1995.

[28] P. Feldmann and R. W. Freund. Reduced-order modeling of large linear subcircuits via a block Lanczos algorithm. In *Proc. 32nd ACM/IEEE Design Automation Conference*, pages 474–479, New York, New York, 1995. ACM.

[29] R. W. Freund. Computation of matrix Padé approximations of transfer functions via a Lanczos-type process. In C. Chui and L. Schumaker, editors, *Approximation Theory VIII, Vol. 1: Approximation and Interpolation*, pages 215–222. World Scientific Publishing Co., Inc., Singapore, 1995.

[30] R. W. Freund. Passive reduced-order models for interconnect simulation and their computation via Krylov-subspace algorithms. In *Proc. 36th ACM/IEEE Design Automation Conference*, pages 195–200, New York, New York, 1999. ACM.

[31] R. W. Freund. Reduced-order modeling techniques based on Krylov subspaces and their use in circuit simulation. In B. N. Datta, editor, *Applied and Computational Control, Signals, and Circuits*, volume 1, pages 435–498. Birkhäuser, Boston, 1999.

[32] R. W. Freund. Band Lanczos method (Section 7.10). In Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, editors, *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, pages 205–216. SIAM Publications, Philadelphia, Pennsylvania, 2000. Also available online from `http://cm.bell-labs.com/cs/doc/99`.

[33] R. W. Freund. Krylov-subspace methods for reduced-order modeling in circuit simulation. *J. Comput. Appl. Math.*, 123(1–2):395–421, 2000.

[34] R. W. Freund and P. Feldmann. Reduced-order modeling of large passive linear circuits by means of the SyPVL algorithm. In *Tech. Dig. 1996 IEEE/ACM International Conference on Computer-Aided Design*, pages 280–287, Los Alamitos, California, 1996. IEEE Computer Society Press.

[35] R. W. Freund and P. Feldmann. Small-signal circuit analysis and sensitivity computations with the PVL algorithm. *IEEE Trans. Circuits and Systems—II: Analog and Digital Signal Processing*, 43:577–585, 1996.

[36] R. W. Freund and P. Feldmann. The SyMPVL algorithm and its applications to interconnect simulation. In *Proc. 1997 International Conference on Simulation of Semiconductor Processes and Devices*, pages 113–116, Piscataway, New Jersey, 1997. IEEE.

[37] R. W. Freund and P. Feldmann. Reduced-order modeling of large linear passive multi-terminal circuits using matrix-Padé approximation. In *Proc. Design, Automation and Test in Europe Conference 1998*, pages 530–537, Los Alamitos, California, 1998. IEEE Computer Society Press.

[38] R. W. Freund and F. Jarre. An extension of the positive real lemma to descriptor systems. Numerical Analysis Manuscript No. 00–3–09, Bell Laboratories, Murray Hill, New Jersey, USA, December 2000. Also available online from `http://cm.bell-labs.com/cs/doc/00`.

[39] R. W. Freund and F. Jarre. Numerical computation of nearby positive real systems in the descriptor case. Numerical analysis manuscript, Bell Laboratories, Murray Hill, New Jersey, USA, 2002, in preparation.

[40] L. D. Gabbay, J. E. Mehner, and S. D. Senturia. Computer-aided generation of nonlinear reduced-order dynamic macromodels—I: non-stress-stiffened case. *J. of Microelectromechanical Systems*, 9(2):262–269, 2000.

[41] S.-Y. Kim, N. Gopal, and L. T. Pillage. Time-domain macromodels for VLSI interconnect analysis. *IEEE Trans. Computer-Aided Design*, 13:1257–1270, 1994.

[42] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. Res. Nat. Bur. Standards*, 45:255–282, 1950.

[43] I. Masubuchi, Y. Kamitane, A. Ohara, and N. Suda. h_∞ control for descriptor systems; a matrix inequalities approach. *Automatica J. IFAC*, 33(4):669–673, 1997.

[44] H. Nelis. *Sparse Approximations of Inverse Matrices.* PhD thesis, Delft Univ. Techn., The Netherlands, 1989.

[45] H. Nelis, P. Dewilde, and E. Deprettere. Inversion of partially specified positive definite matrices by inverse scattering. In *The Gohberg Anniversary Collection*, Operator Theory: Advances and Applications, Vol. 40, pages 325–357. Birkhäuser Verlag, Basel, 1989.

[46] A. Odabasioglu. Provably passive RLC circuit reduction. M.S. thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University, 1996.

[47] A. Odabasioglu, M. Celik, and L. T. Pileggi. PRIMA: passive reduced-order interconnect macromodeling algorithm. In *Tech. Dig. 1997 IEEE/ACM International Conference on Computer-Aided Design*, pages 58–65, Los Alamitos, California, 1997. IEEE Computer Society Press.

[48] L. T. Pileggi. Coping with RC(L) interconnect design headaches. In *Tech. Dig. 1995 IEEE/ACM International Conference on Computer-Aided Design*, pages 246–253, Los Alamitos, California, 1995. IEEE Computer Society Press.

[49] R. A. Rohrer and H. Nosrati. Passivity considerations in stability studies of numerical integration algorithms. *IEEE Trans. Circuits and Systems*, 28:857–866, 1981.

[50] A. E. Ruehli. Equivalent circuit models for three-dimensional multiconductor systems. *IEEE Trans. Microwave Theory Tech.*, 22:216–221, 1974.

[51] I. Schur. Über Potenzreihen, die im Innern des Einheitskreises beschränkt sind, I. *J. Reine Angew. Math.*, 147:205–232, 1917. English translation in Operator Theory: Advances and Applications, Vol. 18, pp. 31-59, Birkhäuser Verlag, Basel, 1986.

[52] S. D. Senturia, N. Aluru, and J. White. Simulating the behavior of MEMS devices: Computational methods and needs. *IEEE Comput. Sci. Eng. Mag.*, 4:30–43, 1997.

[53] T.-J. Su and J. R. R. Craig. Model reduction and control of flexible structures using Krylov vectors. *J. Guidance Control Dynamics*, 14:260–267, 1991.

[54] SUGAR. A MEMS simulation program, ver.2.0(beta), August 2001. Available at http://www-bsac.eecs.berkeley.edu/~cfm.

[55] F. Tisseur and K. Meerbergen. The quadratic eigenvalue problem. *SIAM Rev.*, 43(2):235–286, 2001.

[56] N. van der Meijs. *Accurate and Efficient Layout Extraction.* PhD thesis, Delft Univ. Techn., The Netherlands, 1992.

[57] G. C. Verghese, B. C. Lévy, and T. Kailath. A generalized state-space for singular systems. *IEEE Trans. Automat. Control*, 26(4):811–831, August 1981.

[58] J. Vlach and K. Singhal. *Computer Methods for Circuit Analysis and Design.* Van Nostrand Reinhold, New York, New York, second edition, 1994.

[59] J. L. Willems. *Stability Theory of Dynamical Systems.* John Wiley & Sons, Inc., New York, New York, 1970.

[60] K. Zhou, J. C. Doyle, and K. Glover. *Robust and Optimal Control.* Prentice-Hall, Upper Saddle River, New Jersey, 1996.