Computation of Large-Scale Quadratic Form and Transfer Function Using the Theory of Moments, Quadrature and Padé Approximation

Zhaojun BAI

Department of Computer Science
University of California
Davis, CA 95616, USA

Gene GOLUB

Department of Computer Science
Stanford University
Stanford, CA 94305, USA

Abstract

Large-scale problems in scientific and engineering computing often require solutions involving large-scale matrices. In this paper, we survey numerical techniques for solving a variety of large-scale matrix computation problems, such as computing the entries and trace of the inverse of a matrix, computing the determinant of a matrix, and computing the transfer function of a linear dynamical system.

Most of these matrix computation problems can be cast as problems of computing quadratic forms $u^T f(A) u$ involving a matrix functional $f(A)$. It can then be transformed into a Riemann-Stieltjes integral, which brings the theory of moments, orthogonal polynomials and the Lanczos process into the picture. For computing the transfer function, we focus on numerical techniques based on Padé approximation via the Lanczos process and moment-matching property. We will also discuss issues related to the development of efficient numerical algorithms, including using Monte Carlo simulation.

1 Introduction

In the 1973 paper entitled "*Some Modified Eigenvalue Problems*" by Golub [34], numerical calculation of several matrix computation problems are considered. These include finding constrained eigenvalue problems, determining the eigenvalues of a matrix which is modified by a matrix of rank one, solving constrained and total least squares problems. All these problems require some manipulation before the standard algorithms may be used. Many of these problems have been further studied and widely applied, such as the total least squares problems [59].

In this paper, we survey a collection of "new" modified matrix computation problems. It includes computing the entries of the inverse of a matrix, $(A^{-1})_{ij}$, the determinant, $\det(A)$, the trace of the inverse of a matrix, $\mathrm{trace}(A^{-1})$, and the transfer function, $h(s) = l^T(I -$

$sA)^{-1}r$. More generally, we are concerned with the problem of computing the quadratic form $u^T f(A)u$ or bilinear form $u^T f(A)v$ involving a matrix functional $f(A)$. The matrix A in question is typically large and sparse. We will discuss those iterative methods in which the matrix A in question is only referenced via matrix-vector products.

The necessity of solving these problems appears in many applications. For example, let $\hat{x}$ be an approximation of the exact solution x of a linear system of equations $Ax = b$, where we assume that A is symmetric positive definite. In order to obtain error bounds of the approximation, we consider the A-norm of the forward error vector $e = x - \hat{x}$:

$$\|e\|_A^2 = e^T Ae = r^T A^{-1} AA^{-1} r = r^T A^{-1} r,$$

where r is the residual vector $r = b - A\hat{x}$. Therefore, the problem becomes to obtain computable bounds for the quadratic form $r^T A^{-1} r$. It is sometimes also of interest to compute the l_2-norm of the error e, $\|e\|_2$. Then it is easy to see that one needs to compute the quadratic form $\|e\|_2^2 = r^T A^{-2} r$. Bounds for the error of linear system of equations have been studied extensively in [21, 35, 36, 17].

There are a number of applications where it is required to compute the trace of the inverse, $\mathrm{tr}(A^{-1})$, and the determinant, $\det(A)$, of a very large and sparse matrix A, such as in the theory of fractals [56, 63] and lattice Quantum Chromodynamics (QCD) [57, 24, 31].

The solution of a linear least squares problem by the generalized cross-validation technique involves the computation of the trace of the matrix $I - K(K^T K + m\lambda I)^{-1} K$ for estimating a regularization parameter λ, where K is an $m \times n$ matrix, $m \geq n$ [38]. The quadratic forms $u^T \exp(\sqrt{-1}At)u$, $u^T \exp(-\beta A)u$ and $u^T(\omega I - A)^{-1}u$ appear in the computation of magnetic resonance spectra, quantum statistical mechanics and a variety of other problems in chemical physics [50]. In molecular dynamics, the total energy of an electronic structure requires the calculation of a partial eigenvalue sum of a generalized symmetric definite eigenvalue problem. In [6], it is shown that this sum can be obtained through the computation of the trace of the matrix $A(I + \exp((A - \tau)/\kappa)^{-1}$, where τ and κ are parameters. Many other origins and applications of the computation of the quadratic form are listed in [8, 25].

The problem of transfer function computing arises from the steady-state analysis of a linear dynamical system. A successful approximation of the transfer function also leads to a very desirable and potentially powerful reduced-order model of the original full-order system. We will discuss motivations and applications of transfer function computing in section 4.

2 Quadratic form

Given an $N \times N$ real symmetric matrix A and a smooth function f such that the matrix function $f(A)$ is defined, the problem of quadratic form computing is to evaluate the quadratic form

$$u^T f(A)u, \tag{1}$$

or computing tight lower and upper bounds γ_ℓ and γ_u of the quadratic form $u^T f(A)u$:

$$\gamma_\ell \leq u^T f(A)u \leq \gamma_u. \tag{2}$$

Here u is a given real column vector of length N. Without loss of generality, one may assume $u^T u = 1$.

The quadratic form computing problem was first proposed in [21] for bounding the error of linear systems of equations. It has been further studied in [35, 37, 5] and extended to other applications. In this section, we will first discuss the main idea of the approach, and show that the problem of the quadratic form computing can be transformed into a Riemann-Stieltjes integral, and then use Gauss-type quadrature rules to approximate the integral, which brings the theory of moments and orthogonal polynomials, and the underlying Lanczos process into the picture.

We note that the quadratic form (1) can be generalized to the block case, namely, to evaluate the matrix quadratic form

$$U^T f(A) U$$

where U is a block of N-vectors. The problem of computing a bilinear form $u^T f(A) v$, where $u \neq v$, can be reduced to the problem of quadratic form computing by the polarization expression:

$$u^T f(A) v = \frac{1}{4}\left(y^T f(A) y - z^T f(A) z\right),$$

where $y = u + v$ and $z = u - v$.

2.1 Main idea

Let us now go through the main idea. Since A is symmetric, the eigen-decomposition of A is given by $A = Q^T \Lambda Q$, where Q is an orthogonal eigenvector matrix and Λ is a diagonal matrix with increasingly ordered diagonal elements λ_i, the eigenvalues. Then the quadratic form $u^T f(A) u$ can be written as

$$u^T f(A) u = u^T Q^T f(\Lambda) Q u = \tilde{u}^T f(\Lambda) \tilde{u} = \sum_{i=1}^{N} f(\lambda_i) \tilde{u}_i^2,$$

where $\tilde{u} = (\tilde{u}_i) \equiv Qu$. The last sum can be interpreted as a Riemann-Stieltjes integral

$$u^T f(A) u = \int_a^b f(\lambda) d\mu(\lambda), \tag{3}$$

where the measure $\mu(\lambda)$ is a piecewise constant function and defined by

$$\mu(\lambda) = \begin{cases} 0, & \text{if } \lambda < a \leq \lambda_1, \\ \sum_{j=1}^{i} \tilde{u}_j^2, & \text{if } \lambda_i \leq \lambda < \lambda_{i+1} \\ \sum_{j=1}^{N} \tilde{u}_j^2, & \text{if } b \leq \lambda_N \leq \lambda \end{cases}$$

and a and b are the lower and upper bounds of the eigenvalues λ_i.

To obtain an estimate for the Riemann-Stieltjes integral (3), one can use Gauss-type quadrature rules [33, 22]:

$$I_n[f] = \sum_{j=1}^{n} \omega_j f(\theta_j) + \sum_{k=1}^{m} \rho_k f(\tau_k), \tag{4}$$

where the weights $\{\omega_j\}$ and $\{\rho_k\}$ and the nodes $\{\theta_j\}$ are unknown and to be determined. The nodes $\{\tau_k\}$ are prescribed. If $m = 0$, then it is the well-known Gauss rule. If $m = 1$ and

$\tau_1 = a$ or $\tau_1 = b$, it is the Gauss-Radau rule. The Gauss-Lobatto rule is for $m = 2$ and $\tau_1 = a$ and $\tau_2 = b$.

The accuracy of the Gauss-type quadrature rules may be obtained by an estimation of the remainder $R_n[f]$:

$$R_n[f] = \int_a^b f(\lambda) d\mu(\lambda) - I_n[f].$$

For the Gauss quadrature rule, the remainder $R_n[f]$ can be written as

$$R_n[f] = \frac{f^{(2n)}(\eta)}{(2n)!} \int_a^b \left[\prod_{i=1}^n (\lambda - \theta_i)\right]^2 d\mu(\lambda), \tag{5}$$

for some $\eta \in (a, b)$. If the sign of $R_n[f]$ can be determined, then the quadrature $I_n[f]$ is a lower bound (if $R_n[f] > 0$) or an upper lower bound (if $R_n[f] < 0$) of the quadratic form $u^T f(A) u$. Similarly, for the Gauss-Radau rule, the remainder $R_n[f]$ is given by

$$R_n[f] = \frac{f^{(2n+1)}(\eta)}{(2n+1)!} \int_a^b (\lambda - \tau_1) \left[\prod_{i=1}^n (\lambda - \theta_i)\right]^2 d\mu(\lambda), \tag{6}$$

for some $\eta \in (a, b)$, and for the Gauss-Lobatto rule, it is

$$R_n[f] = \frac{f^{(2n+2)}(\eta)}{(2n+2)!} \int_a^b (\lambda - \tau_1)(\lambda - \tau_2) \left[\prod_{i=1}^n (\lambda - \theta_i)\right]^2 d\mu(\lambda). \tag{7}$$

for some $\eta \in (a, b)$.

2.2 Moments

In defining Gaussian quadratures, we choose the weights ω_j and nodes θ_j so that $2n$ moments are calculated exactly, i.e.,

$$\mu_r = \int_a^b \lambda^r d\mu(\lambda) = \sum_{j=1}^n \omega_j \, \theta_j^r$$

for $r = 0, 1, 2, \ldots, 2n - 1$. A different perspective on this statement is that the summation on the right is of the form of a solution to a difference equation. In particular, let $n = 2$, the moments μ_r satisfy a second-order difference equation of the form

$$\alpha \mu_r + \beta \mu_{r-1} + \gamma \mu_{r-2} = 0$$

for certain coefficients α, β and γ. The nodes θ_j are the roots of the characteristic polynomial

$$\alpha \theta^2 + \beta \theta + \gamma = 0.$$

We can use this fact to bound the quadratic form. In particular, note that first three moments $\mu_0 = \mathrm{tr}(A^0) = N$, $\mu_1 = \mathrm{tr}(A)$ and $\mu_2 = \mathrm{tr}(A^2) = \|A\|_F^2$ can be easily calculated. By using these three moments, in [7], it is shown that for a symmetric positive definite matrix A, we have the following computable bounds for the trace of the inverse $\mathrm{tr}(A^{-1})$:

$$\begin{bmatrix} \mu_1 & \mu_0 \end{bmatrix} \begin{bmatrix} \mu_2 & \mu_1 \\ b^2 & b \end{bmatrix}^{-1} \begin{bmatrix} \mu_0 \\ 1 \end{bmatrix} \le \mathrm{tr}(A^{-1}) \le \begin{bmatrix} \mu_1 & \mu_0 \end{bmatrix} \begin{bmatrix} \mu_2 & \mu_1 \\ a^2 & a \end{bmatrix}^{-1} \begin{bmatrix} \mu_0 \\ 1 \end{bmatrix},$$

where a and b are the lower and upper bounds of the eigenvalues of A, $a > 0$. Furthermore, by the identity

$$\ln(\det(A)) = \mathrm{tr}(\ln A), \tag{8}$$

we can also use the first three moments of A to obtain lower and upper bounds of $\ln(\det(A))$:

$$\begin{bmatrix} \ln a & \ln \underline{t} \end{bmatrix} \begin{bmatrix} a & \underline{t} \\ a^2 & \underline{t}^2 \end{bmatrix}^{-1} \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \le \ln(\det(A)) \le \begin{bmatrix} \ln b & \ln \bar{t} \end{bmatrix} \begin{bmatrix} b & \bar{t} \\ b^2 & \bar{t}^2 \end{bmatrix}^{-1} \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$$

where $\underline{t} = (a\mu_1 - \mu_2)/(a\mu_0 - \mu_1)$ and $\bar{t} = (b\mu_1 - \mu_2)/(b\mu_0 - \mu_1)$.

2.3 Orthogonal polynomials and symmetric Lanczos process

Before we discuss how the weights and the nodes are calculated in the Gauss-type quadratures, let us briefly review the theory of orthogonal polynomials and symmetric Lanczos process. For a given nondecreasing measure function $\mu(\lambda)$, it is well-known [58] that a sequence of polynomials $\{p_j(\lambda)\}$ can be constructed via a three-term recurrence

$$\beta_j p_j(\lambda) = (\lambda - \alpha_j)p_{j-1}(\lambda) - \beta_{j-1}p_{j-2}(\lambda),$$

for $j = 1, 2, \ldots, n$ with $p_{-1}(\lambda) \equiv 0$ and $p_0(\lambda) \equiv 1$. Furthermore, they are orthonormal with respect to the measure $\mu(\lambda)$:

$$(p_i, p_j) = \int_a^b p_i(\lambda)p_j(\lambda)d\mu(\lambda) = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{if } i \ne j, \end{cases}$$

where it is assumed the normalization condition $\int d\mu = 1$ (i.e., $u^T u = 1$). Writing the three-term recurrence in matrix form, we have

$$\lambda p(\lambda) = T_n\, p(\lambda) + \beta_n p_n(\lambda) e_n$$

where

$$p(\lambda)^T = [p_0(\lambda), p_1(\lambda), \ldots, p_{n-1}(\lambda)],$$

and

$$T_n = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ \beta_1 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_{n-1} \\ & & \beta_{n-1} & \alpha_n \end{bmatrix}.$$

The following classical symmetric Lanczos process [46] is an elegant way to compute the recurrence coefficients α_j and β_j.

> **Symmetric Lanczos process:** Let A be a real symmetric matrix, u a real vector with $u^T u = 1$. Then the following procedure computes the tridiagonal matrix T_n and the orthonormal Lanczos vectors q_j.

```
Let $q_0 = 0$, $\beta_0 = 0$, and $q_1 = u$
$\alpha_1 = q_1^T A q_1$
For $j = 1, 2, \ldots, n$,
    $r_j = Aq_j - \alpha_j q_j - \beta_{j-1} q_j$
    $\beta_j = \|r_j\|_2$
    $q_{j+1} = r_j / \beta_j$
    $\alpha_{j+1} = q_{j+1}^T A q_{j+1}$
```

Let $Q_n = [q_1, q_2, \ldots, q_n]$. Then the symmetric Lanczos process can be written in compact matrix form

$$AQ_n = Q_n T_n + \beta_n q_{n+1} e_n^T,$$

and $Q_n^T Q_n = I$ and $Q_n^T q_{n+1} = 0$. The Lanczos vectors q_j produced by the symmetric Lanczos process and the orthonormal polynomials $\{p_j(\lambda)\}$ are connected as the following

$$q_j = p_{j-1}(A) u$$

for $j = 1, \ldots, n$. Note that the eigenvalues of T_n are the roots of the polynomial $p_n(\lambda)$.

2.4 Basic Algorithms

Gauss-Lanczos (GL) algorithm. Using the eigen-decomposition of T_n:

$$T_n = S_n D_n S_n^T$$

where $D_n = \mathrm{diag}(\theta_1, \theta_2, \ldots, \theta_n)$ and $S_n^T S_n = I_n$. Then for the Gauss quadrature rule, the eigenvalues θ_i of T_n (which are the zeros of the polynomial $p_n(\lambda)$) are the desired nodes. The weights ω_j are the squares of the first elements of the normalized eigenvectors of T_n, i.e., $\omega_j = (e_1^T S_n e_j)^2$.

By combining the Gauss quadrature and the symmetric Lanczos process, we have an algorithm for computing an estimate $I_n[f]$ of the quadrature form $u^T f(A) u$. We refer to it as the Gauss-Lanczos (GL) algorithm.

From the expression (5) of the remainder $R_n[f]$, we can determine whether the approximation $I_n[f]$ is a lower or upper bound of the quadratic form $u^T f(A) u$. For example, if $f^{(2n)}(\eta) > 0$, then $I_n[f]$ is a lower bound ℓ_l.

We note that it is not always necessary to explicitly compute the eigenvalues and the first components of eigenvectors of the tridiagonal matrix T_n for obtaining the estimation $I_n[f]$. In fact, by the fact that $\omega_j = (e_1^T S_n e_j)^2$, the Gauss rule can be written in the form

$$I_n[f] = \sum_{k=1}^{n} \omega_k f(\theta_k) = e_1^T S_n f(D_n) S_n^T e_1 = e_1^T f(T_n) e_1. \tag{9}$$

Therefore, if the (1,1) entry of $f(T_n)$ can be easily computed, for example, $f(\lambda) = 1/\lambda$, then the computation of the eigenvalues and eigenvectors of T_n can be avoided. This is the case when applying this technique to error estimation of an iterative method for solving linear system of equations [36, 17].

By exploiting the relation between the Lanczos vectors q_j and orthogonal polynomials p_j, it can be shown that the remainder $R_n[f]$ can be written as

$$R_n[f] = \frac{f^{(2n)}(\eta)}{(2n)!}\beta_1^2\beta_2^2\cdots\beta_n^2$$

for some $\eta \in (a,b)$. Therefore, if $f^{(2n)}(\eta)$ can be estimated or bounded, the error can be easily obtained with little additional cost [16, 25].

Gauss-Radau-Lanczos (GRL) algorithm. For the Gauss-Radau and Gauss-Lobatto rules, the nodes $\{\theta_j\}$, $\{\tau_k\}$ and weights $\{\omega_j\}, \{\rho_j\}$ come from eigenvalues and the squares of the first elements of the normalized eigenvectors of an adjusted tridiagonal matrix of T_n, which has the prescribed eigenvalues a and/or b.

To implement the Gauss-Radau rule with the prescribed node $\tau_1 = a$ or $\tau_1 = b$, the above GL algorithm needs to be slightly modified. For example, with $\tau_1 = a$, we need to extend the matrix T_n to

$$\widetilde{T}_{n+1} = \begin{bmatrix} T_n & \beta_n e_n \\ \beta_n e_n^T & \phi \end{bmatrix}. \tag{10}$$

Here the parameter ϕ is chosen such that $\tau_1 = a$ is an eigenvalue of $\widetilde{T}_{n+1}$. From [34], it is known that

$$\phi = a + \delta_n,$$

where δ_n is the last component of the solution δ of the tridiagonal system

$$(T_n - aI)\delta = \beta_n^2 e_n.$$

The eigenvalues and the squares of the first components of orthonormal eigenvectors of $\widetilde{T}_{n+1}$ are the nodes and weights of the Gauss-Radau rule.

By combining the Gauss-Radau quadrature and the symmetric Lanczos process, we have an algorithm for computing an estimate $I_n[f]$ of the quadrature form $u^T f(A)u$. We refer to it as the Gauss-Radau-Lanczos (GRL) algorithm.

If $f^{(2n+1)}(\eta) < 0$, then $\widetilde{I}_n[f]$ (with b as a prescribed eigenvalue of $\widetilde{T}_{n+1}$) is a lower bound ℓ_l of the quantity $u^T f(A)u$. $\widetilde{I}_n[f]$ (with a as a prescribed eigenvalue of $\widetilde{T}_{n+1}$) is an upper bound ℓ_u.

Similar to the GL algorithm, it is not always necessary to compute the eigenvalues and the first components of eigenvectors of the tridiagonal matrix $\widetilde{T}_{n+1}$ for obtaining $\widetilde{I}_n[f]$. In fact, one can show that

$$\widetilde{I}_n[f] = \sum_{k=1}^{n} \omega_k f(\theta_k) + \rho_1 f(\tau_1) = e_1^T f(\widetilde{T}_{n+1})e_1. \tag{11}$$

Therefore, if the (1,1) entry of $f(\widetilde{T}_{n+1})$ can be easily computed, for example, $f(\lambda) = 1/\lambda$, we can directly compute $e_1^T f(\widetilde{T}_{n+1})e_1$.

Gauss-Lobatto-Lanczos (GLL) algorithm. To implement the Gauss-Lobatto rule, T_n computed in the GL algorithm is updated to

$$\widehat{T}_{n+2} = \left[\begin{array}{cc} T_{n+1} & \psi e_{n+1} \\ \psi e_{n+1}^T & \phi \end{array} \right]. \tag{12}$$

Here the parameters ϕ and ψ are chosen so that a and b are eigenvalues of $\widehat{T}_{n+2}$. Again, from [34], it is known that

$$\phi = \frac{b\delta_{n+1} - a\mu_{n+1}}{\delta_{n+1} - \mu_{n+1}} \quad \text{and} \quad \psi^2 = \frac{b-a}{\delta_{n+1} - \mu_{n+1}},$$

where δ_n and μ_n are the last components of the solutions δ and μ of the tridiagonal systems

$$(T_{n+1} - aI)\delta = e_{n+1} \quad \text{and} \quad (T_{n+1} - bI)\mu = e_{n+1}.$$

The eigenvalues and the squares of the first components of eigenvectors of $\widehat{T}_{n+2}$ are the nodes and weights of the Gauss-Lobatto rule. Moreover, if $f^{(2n+2)}(\eta) > 0$, then $\widehat{I}_n[f]$ is an upper bound ℓ_u of $u^T f(A)u$.

By combining the Gauss-Lobatto quadrature and the symmetric Lanczos process, we have an algorithm for computing an estimation $\widehat{I}_n[f]$ of the quadrature form $u^T f(A)u$. We refer to it as the Gauss-Lobatto-Lanczos (GLL) algorithm.

Similarly to (11), we have

$$\widehat{I}_n[f] = \sum_{k=1}^{n} \omega_k f(\theta_k) + \rho_1 f(\tau_1) + \rho_2 f(\tau_2) = e_1^T f(\widehat{T}_{n+2}) e_1. \tag{13}$$

2.5 Pseudo-code, software and numerical examples

We have now established all basic algorithms we need to compute the quadratic form $u^T f(A)u$ by applying Gauss, Gauss-Radau and Gauss-Lobatto rules. These algorithms are summarized in the following pseudo-code.

> **GL, GRL, GLL algorithms:** Let A be a real symmetric matrix, u a real vector with $u^T u = 1$. f is a given smooth function. Then the following algorithms compute the estimates $I_n[f]$, $\widetilde{I}_n[f]$ and $\widehat{I}_n[f]$ of the quadratic form $u^T f(A)u$ by using the Gauss, Gauss-Radau, and Gauss-Lobatto rules via Lanczos process.

```
Let $q_0 = 0$, $\beta_0 = 0$, and $q_1 = u$
$\alpha_1 = q_1^T A q_1$
For $j = 1, 2, \ldots, n$,
    $r_j = Aq_j - \alpha_j q_j - \beta_{j-1} q_j$
    $\beta_j = \|r_j\|_2$
    $q_{j+1} = r_j / \beta_j$
    $\alpha_{j+1} = q_{j+1}^T A q_{j+1}$
For GRL algorithm, update $T_n$ according to (10)
For GLL algorithm, update $T_n$ according to (12)
Compute $I_n[f]$, $\widetilde{I}_n[f]$ and $\widehat{I}_n[f]$ according to (9), (11) and (13)
```

$(A^{-1})_{ii}$	GRL					CG	
i	iter1	lower bound γ_l	iter2	upper bound γ_u	$\gamma_u - \gamma_l$	iter	$(A^{-1})_{ii}$
1	12	$9.4801416e-01$	19	$9.4801776e-01$	$3.60e-06$	24	$9.4801e-1$
100	11	$1.1005253e+00$	20	$1.1005302e+00$	$4.90e-06$	24	$1.1005e+0$
2000	11	$1.1003685e+00$	19	$1.1003786e+00$	$1.01e-05$	23	$1.1004e+0$
3125	10	$6.4400234e-01$	16	$6.4401100e-01$	$8.66e-06$	21	$6.4400e-1$

Table 1: VFH matrix, $e_i^T A^{-1} e_i$ computed by GRL and CG methods.

We note that the "For" loop in the above algorithms is the standard symmetric Lanczos process as described above. The matrix A in question is only referenced in the form of the matrix-vector product. The symmetric Lanczos process can be implemented with only 3 N-vectors in the fast memory. This is the major storage requirement for the algorithm. These are attractive features for large-scale problems.

For the Gauss-Radau and Gauss-Lobatto rules, we need to have the estimates of a and b as the extreme eigenvalues λ_1 and λ_N of A. Numerical experiments show that more steps of the Lanczos process may be required with poor estimates of a and b. One needs to weigh the cost of using a sophisticated method to obtain good estimates of the extreme eigenvalues against the cost of additional Lanczos iterations. Gershgorin circles can be used to estimate a and b. It is usually sufficient for use in the Gauss-Radau and Gauss-Lobatto rules.

An *ad hoc* choice for determining the number of Lanczos iterations n is to use

$$|I_n[f] - I_{n-1}[f]| \leq \epsilon\, |I_n[f]|,$$

where ϵ is a prescribed tolerance value. This criterion removes the restriction to supply the number of iteration *a priori*. It implies that

$$|I[f] - I_n[f]| \leq |I[f] - I_{n-1}[f]| + \epsilon\, |I_n[f]|.$$

Therefore, the iteration stops if the error is no longer decreasing or is decreasing too slowly.

A Matlab toolset, called QUADFORM, is developed in [25] to implement all GL, GRL and GLL algorithms.

Example 1. This is a symmetric positive definite matrix from the transverse vibration of a vicsek fractal Hamiltonian (VFH). The fractal is recursively constructed. The details are described in [5]. Table 1 shows the numerical results for estimating a few diagonal elements of the inverse of the matrix of dimension $N = 3125$. The parameters a and b are computed by Gershgorin circles. We used $\epsilon = 10^{-4}$ for determining the number of Lanczos iterations in the GRL algorithm. The last two columns are the number of iterations and the approximate values of the conjugate gradient (CG) method.

Example 2. This is a 1711 × 1711 matrix from geophysical logs of oil wells data analysis [48]. It is a symmetric positive definite matrix of condition number on the order of 10^7. In this example, we first apply equilibration to improve the condition number. Specifically, the GRL algorithm is applied to the matrix DAD, where $D = \mathrm{diag}(a_{ii}^{-1/2})$. Note that $e_i^T A^{-1} e_i = (De_i)^T (DAD)^{-1} (De_i)$. Table 2 shows the numerical results of GRL and CG methods for estimating few diagonal elements of the inverse of A.

$(A^{-1})_{ii}$	GRL					CG	
i	iter1	lower bound γ_l	iter2	upper bound γ_l	$\gamma_u - \gamma_l$	iter	$(A^{-1})_{ii}$
1	62	$5.2589E-1$	115	$5.4968E-1$	$2.38E-2$	iter	$(A^{-1})_{ii}$
400	244	$7.3332E-2$	344	$7.3605E-2$	$2.73E-4$	335	$5.3138E-1$
900	89	$8.5561E+0$	218	$8.6671E+0$	$1.11E-1$	452	$7.3549E-2$
1400	190	$1.0137E+0$	343	$1.0236E+0$	$9.90E-3$	498	$1.0215E+0$
1711	36	$3.6961E-2$	68	$3.7213E-2$	$2.52E-4$	237	$3.7015E-2$

Table 2: Oil Wells matrix, $e_i^T A^{-1} e_i$ computed by GRL and CG methods.

3 Monte Carlo simulation

In this section, we discuss a Monte Carlo approach for estimating the trace of a matrix function, $\mathrm{tr}(f(A))$. For the task of computing the trace of the inverse of A, we simply take the function $f(\lambda) = 1/\lambda$. For computing the determinant, $\det(A)$, of a symmetric positive definite matrix A, by the identity (8), we take $f(\lambda) = \ln(\lambda)$. Instead of applying GL, GRL or GLL algorithm n times for each diagonal element $e_i^T f(A) e_i$ of $f(A)$, a Monte Carlo approach only applies it m times to obtain an unbiased estimator of the trace of $f(A)$, where in general $m \ll N$. The saving in computational costs could be very significant. Probabilistic confidence bounds for the unbiased estimator can also be obtained. An alternative Monte Carlo approach for computing the trace is presented [12].

Our Monte Carlo approach is based on the following basic property due to [42, 24].

Proposition 3.1 *Let H be an $n \times n$ symmetric matrix with $tr(H) \neq 0$. Let V be the discrete random variable which takes the values 1 and -1 each with probability 0.5 and let z be a vector of n independent samples from V. Then $z^T H z$ is an unbiased estimator of $tr(H)$, i.e.,*

$$E(z^T H z) = tr(H),$$

and

$$Var(z^T H z) = 2 \sum_{i \neq j} h_{ij}^2.$$

In practice, we take m random sample vectors z_i as described in Proposition 3.1, and then use GL algorithm to obtain an unbiased estimator of $\mathrm{tr}(f(A))$,

$$\mathrm{E}(z_i^T f(A) z_i) = \mathrm{tr}(f(A)),$$

for $i = 1, 2, \ldots, m$. By using GRL algorithm, we can obtain a lower bound L_i and an upper bound U_i of the quantity $z_i^T f(A) z_i$:

$$L_i \leq z_i^T f(A) z_i \leq U_i, \tag{14}$$

By taking the mean of the m computed lower and upper bounds L_i and U_i, we have

$$\frac{1}{m} \sum_{i=1}^{m} L_i \leq \frac{1}{m} \sum_{i=1}^{m} z_i^T f(A) z_i \leq \frac{1}{m} \sum_{i=1}^{m} U_i. \tag{15}$$

It is expected that with a suitable sample size m, Monte Carlo yields good bounds for the quantity $\mathrm{tr}(f(A))$.

To quantitatively assess the quality of such estimation, we can turn to confidence bounds of the estimation. In other words, we can find an interval so that the exact value of $\mathrm{tr}(f(A))$ is in the interval with probability p, where $0 < p < 1$. The Hoeffding's exponential inequality in probability theory can be immediately used to derive such confidence bounds [54]. Specifically, let $w_i = z_i^T f(A) z_i - \mathrm{tr}(f(A))$. Since z_i are taken as independent random vectors, w_i are independent random variables. From Proposition 3.1, w_i has zero means (i.e. $E(w_i) = 0$). Furthermore, from (14), we also know that w_i has bounded ranges

$$L_{\min} - \mathrm{tr}(f(A)) \leq w_i \leq U_{\max} - \mathrm{tr}(f(A))$$

for all i, $1 \leq i \leq m$, where $U_{\max} = \max\{U_i\}$ and $L_{\min} = \min\{L_i\}$. By Hoeffding's inequality, we have the following probabilistic bounds for the mean of m samples $z^T i f(A) z_i$,

$$P\left(\left| \frac{1}{m} \sum_{i=1}^{m} z_i^T f(A) z_i - \mathrm{tr}(f(A)) \right| \geq \frac{\eta}{m} \right) \leq 2 \exp\left(\frac{-2\eta^2}{d} \right), \tag{16}$$

where $d = m(U_{\max} - L_{\min})^2$ and $\eta > 0$ is a tolerance value, which is related to the probability in the right hand side of the inequality. In other words, inequality (16) tells us that

$$P\left(\frac{1}{m} \sum_{i=1}^{m} z_i^T f(A) z_i - \frac{\eta}{m} < \mathrm{tr}(f(A)) < \frac{1}{m} \sum_{i=1}^{m} z_i^T f(A) z_i + \frac{\eta}{m} \right) > 1 - 2 \exp\left(\frac{-2\eta^2}{d} \right).$$

Then from the bounds (15), we have

$$P\left(\frac{1}{m} \sum_{i=1}^{m} L_i - \frac{\eta}{m} < \mathrm{tr}(f(A)) < \frac{1}{m} \sum_{i=1}^{m} U_i + \frac{\eta}{m} \right) > 1 - 2 \exp\left(\frac{-2\eta^2}{d} \right). \tag{17}$$

Therefore, we conclude that the trace of $f(A)$ is in the interval

$$\left(\frac{1}{m} \sum_{i=1}^{m} L_i - \frac{\eta}{m}, \frac{1}{m} \sum_{i=1}^{m} U_i + \frac{\eta}{m} \right)$$

with probability $1 - 2\exp(-2\eta^2/d)$.

If we specify the probability p in (17), i.e. $p = 1 - 2\exp\left(\frac{-2\eta^2}{d}\right)$, then solving this equality for $\frac{\eta}{m}$, yields

$$\frac{\eta}{m} = \sqrt{-\frac{1}{2m}(U_{\max} - L_{\min})^2 \ln\left(\frac{1-p}{2} \right)}. \tag{18}$$

Since $(U_{\max} - L_{\min})^2$ is bounded by $2N^2 \|f(A)\|_2^2$, we see that with a fixed value of p, $\frac{\eta}{m} \to 0$ as $m \to \infty$, i.e., the confidence interval is essentially given by the means of the computed bounds.

We now have a Monte Carlo algorithm which computes an unbiased estimator of $\mathrm{tr}(f(A))$. The algorithm also returns a confidence interval with user specified probability. We note that L_i and U_i are generally very sharp bounds of $z_i^T f(A) z_i$. It would be ideal if we could have a sharp confidence interval, i.e., η/m is small. However, from equation (18), we may have to choose a quite large number of samples m. It would be too expensive. Instead, we generally choose a fixed number of samples m and the probability p to compute the corresponding confidence interval. Here is the algorithm based on the Monte Carlo approach.

> **Monte Carlo algorithm**. Suppose A is symmetric positive definite. Let m be a chosen number of samples. Then the following algorithm computes (a) an unbiased estimator I_p of the quantity $\mathrm{tr}(f(A))$, and (b) a confidence interval (L_p, U_p) such that $\mathrm{tr}(f(A)) \in (L_p, U_p)$ with a user-specified probability p, where $0 < p < 1$.

```
For $j = 1, 2, \cdots, m$
    Generate $n$-vector $z_j$ with uniformly distributed elements in (0,1).
    For $i = 1 : n$, if $z_j(i) < 0.5$, then $z_j(i) = -1$, otherwise, $z_j(i) = 1$.
    Apply GL algorithm to obtain an estimator $I_j$ of $z_j^T f(A) z_j$
    Apply GRL algorithm to obtain the bounds $(L_j, U_j)$ of $z_j^T f(A) z_j$
    $L_{\min} = \min\{L_{\min}, L_j\}$
    $U_{\max} = \max\{U_{\max}, U_j\}$
    $\eta^2 = -0.5j(U_{\max} - L_{\min})^2 \ln(\frac{1-p}{2})$
    $L_p(j) = \frac{1}{j}\sum_{i=1}^{j} L_i - \frac{\eta}{j}$
    $U_p(j) = \frac{1}{j}\sum_{i=1}^{j} U_i + \frac{\eta}{j}$
End
$I_p = \frac{1}{m}\sum_{j=1}^{m} I_j$
$L_p = L_p(m)$
$U_p = L_p(m)$
```

A couple of improvements of Monte Carlo simulation for computing the trace of a matrix function have been proposed recently. In [62], it is proposed to use a low discrepancy sampling method for a better choice of sample vectors z_j to improve convergence rate. One can also develop a variance reduction technique via control regression. The essential idea is to apply the first few easy-to-compute moments of the matrices A and T_n to minimize the variance of the estimates I_j via regression. The following example includes a preliminary result of variance reduction.

Example 3. This is a consistent mass matrix from a regular $n_x \times n_y$ grid of 8-node (serendipity) elements. It is from Higham's test matrix collection available in Matlab's `gallery`. The order of the matrix is $N = 3n_xn_y + 2n_x + 2n_y + 1$. We choose $n_x = n_y = 12$ and then $N = 481$. Numerical results of a Monte Carlo simulation of $\mathrm{tr}(A^{-1})$ and variance reduction are plotted in Figure 1. Solid horizontal lines in the first top two plots are the exact value $\mathrm{tr}(A^{-1})$. In the top plot, the solid plus and dash circle lines are the estimates by GL algorithm and improved ones with variance reduction for 30 different random samples z_j, respectively. In the middle plot, the solid plus and dash circle lines are the means of the GL estimates and improved ones. The bottom plot is the variances before (solid plus) and after (dash circle) applying the variance reduction technique via control regression.

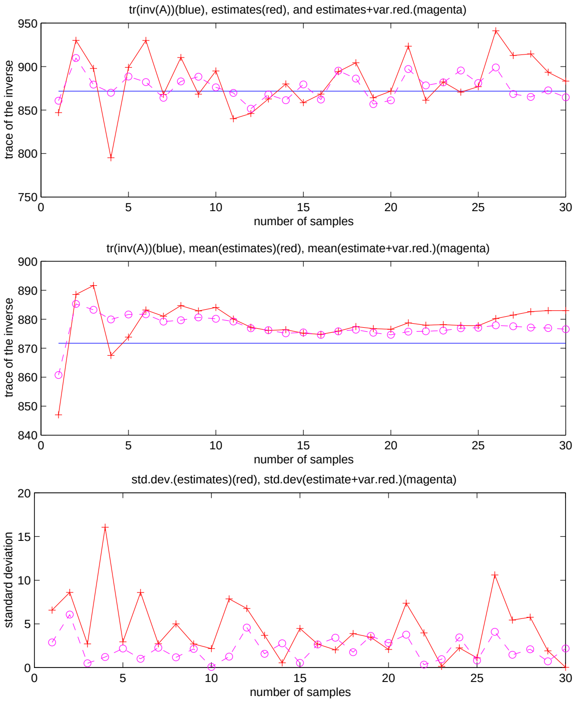

Figure 1: Monte Carlo simulation of $\mathrm{tr}(A^{-1})$.

4 Transfer function

4.1 Linear dynamical systems and transfer function

A continuous time-invariant lumped multi-input multi-output linear dynamical system is of the form

$$\left\{ \begin{array}{rcl} C\dot{x}(t) + Gx(t) & = & B\,u(t), \\ y(t) & = & L^T x(t), \end{array} \right. \tag{19}$$

with initial condition $x(0) = x_0$. Here t is the time variable, $x(t) \in \mathcal{R}^N$ is a state vector, $u(t) \in \mathcal{R}^m$ the input excitation vector, and $y(t) \in \mathcal{R}^p$ the output measurement vector. $C, G \in \mathcal{R}^{N\times N}$ are system matrices, $B \in \mathcal{R}^{N\times m}$ and $L \in \mathcal{R}^{N\times p}$ are input and output distribution arrays, respectively. N is the state space dimension and m and p are the number of inputs and outputs, respectively. In most practical cases, we can assume that m and p are much smaller than N and $m \geq p$.

Linear systems arise in many applications, such as the network circuit with linear elements [60], structural dynamics analysis with only lumped mass and stiffness elements [19, 20], linearization of a nonlinear system around an equilibrium point [27], and a semi-discretization with respect to spatial variables of a time-dependent differential-integral equations [55, 61].

The matrices C and G in (19) are allowed to be singular, and we only assume that the pencil $G + sC$ is *regular*, i.e., the matrix $G + sC$ is singular only for a finite number of values $s \in \mathcal{C}$. The assumption that $G + sC$ is regular is satisfied for all applications we are concerned with that lead to systems of the form (19). In addition, C and G in (19) are general nonsymmetric matrices. However, in some important applications, C and G are symmetric, and possibly positive definite or positive semidefinite. Note that when C is singular, the first equation in (19) is a first-order system of linear differential-algebraic equations. The corresponding linear system is called a descriptor system or a singular system.

The linear system of the form (19) is often referred to as the representation of the system in the time domain or in the state space. Equivalently, one can also represent the system in the frequency domain via a Laplace transform. Recall that for a vector-valued function $f(t)$, the Laplace transform of $f(t)$ is defined by

$$F(s) := \mathcal{L}\{f(t)\} = \int_0^{\infty} f(t)e^{-st}dt, \quad s \in \mathcal{C}. \tag{20}$$

The physically meaningful values of the complex variable s are $s = \mathtt{i}\omega$, where $\omega \geq 0$ is referred to as the frequency. Taking the Laplace transform of the system (19), we obtain the following frequency domain formulation of the system:

$$\left\{ \begin{array}{rcl} sCX(s) + GX(s) & = & B\,U(s), \\ Y(s) & = & L^T X(s), \end{array} \right. \tag{21}$$

where $X(s)$, $Y(s)$ and $U(s)$ represents the Laplace transform of $x(t)$, $y(t)$ and $u(t)$, respectively. For simplicity, we assume that we have zero initial conditions $x(0) = x_0 = 0$ and $u(0) = 0$.

Eliminating the variable $X(s)$ in (21), we see that the input $U(s)$ and the output $Y(s)$ in the frequency domain are related by the following $p \times m$ matrix-valued rational function

$$H(s) = L^T(G + sC)^{-1}B. \tag{22}$$

$H(s)$ is known as the *transfer function* or *Laplace-domain impulse response* of the linear system (19).

Steady-state analysis, also called frequency response analysis, is to determine the frequency responses $H(\mathtt{i}\omega)$ of the system to external steady-state oscillatory (i.e., sinusoidal) excitation.

Linear dynamical systems have been studied extensively, especially for the case $C = I$, for example, see [44]. Numerous techniques have been developed for performing various analyses of the system. One of the primary computational challenges we are confronted with today is the large state dimension N of the system (19). For example, in circuit simulation and structural dynamics applications, N could be as large as 10^6. In addition, the differential equations in the system (19) are often stiff from multi-energy and multi-scaling simulation. The system may be required to be analyzed repeatedly for different excitation inputs $u(t)$.

For the sake of simplicity, in the rest of this section we mostly confine our discussion to single-input single-output systems, i.e., $p = m = 1$. We will use lower case letters b and l to denote the input and output distribution vectors, instead of the capital letters B and L.

4.2 Eigensystem methods

Let us first review eigensystem methods as an introduction to compute the transfer function. To compute $H(s)$ about a selected expansion point s_0, let us set

$$A = -(G + s_0 C)^{-1} C \quad \text{and} \quad r = (G + s_0 C)^{-1} b,$$

where we assume that $G + s_0 C$ is nonsingular. Then $H(s)$ can be cast as

$$H(s) = l^T \left((G + s_0 C) + (s - s_0)C\right)^{-1} b = l^T (I - (s - s_0)A)^{-1} r. \tag{23}$$

In other words, we reduce the representation of the transfer function $H(s)$ using only one matrix A. Assume that the matrix A is diagonalizable,

$$A = S \, \Lambda \, S^{-1} = S \cdot \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_N) \cdot S^{-1}.$$

Let $f = S^T l = (f_j)$ and $g = S^{-1} r = (g_j)$, then the transfer function $H(s)$ can be expressed as a partial-fraction expansion,

$$H(s) = f^T (I - (s - s_0)\Lambda)^{-1} g = \sum_{j=1}^{N} \frac{f_j g_j}{1 - (s - s_0)\lambda_j} = \rho_\infty + \sum_{\lambda_j \neq 0} \frac{\kappa_j}{s - p_j}. \tag{24}$$

This is known as the *pole-residue representation.* $p_j = s_0 + \frac{1}{\lambda_j}$ are *poles* of the system[1], $\kappa_j = -\frac{f_j g_j}{\lambda_j}$ are *residues*, and $\rho_\infty = \sum_{\lambda_j = 0} f_j g_j$ is a constant, which corresponds to the poles at infinity (or zero eigenvalues). Note that it costs $\mathcal{O}(N^3)$ operations to diagonalize A, and only $\mathcal{O}(N)$ operations to evaluate the transfer function $H(s)$ for each given point s.

Unfortunately, in practice, diagonalization of A is prohibitive when it is ill-conditioned or is too large. As a remedy for the possible ill-conditioning of diagonalization, we may use the numerically stable Schur decomposition. Let $A = QTQ^T$ be the Schur decomposition of A. Then

$$H(s) = l^T (I - (s - s_0)A)^{-1} r = (Q^T l)^T (I - (s - s_0)T)^{-1} (Q^T r).$$

Now, it costs $\mathcal{O}(N^2)$ to evaluate the transfer function $H(s)$ at each given point s. Alternatively, we can also use the Hessenberg decomposition as suggested in [47].

To reduce the cost of diagonalizing A or computing its decomposition in Schur or Hessenberg form for large N, we may use partial eigen-decomposition. This is also referred to as the *modal superposition method*, for example, see [19]. By examining the pole-residue representation (24), it is easy to see that the motivation of this approach comes from the fact that only a few poles (and associated eigenvalues) around the region of frequencies of interest are necessary for the approximation of $H(s)$. Those poles are called the dominant poles. Therefore, to study the steady-state response to an input of the form $u(t) = \tilde{u} e^{\mathrm{i}\omega t}$, where $\tilde{u}$ is a constant vector, we express the solution as $x(t) = S_k v(\omega) e^{\mathrm{i}\omega t}$, where S_k contains k selected modal shapes (eigenvectors) of the matrix pair $\{C, G\}$ needed to retain all the modes whose resonant frequencies lie within the range of input excitation frequencies. Then one may solve the system

$$\left(\mathrm{i}\omega \, S_k^T C S_k + S_k^T G S_k\right) v(\omega) = S_k^T B \tilde{u} \tag{25}$$

for $v(\omega)$. Once the selected dominant poles and their corresponding modal shapes S_k are computed, the problem of computing the steady-state response is reduced to solving the $k \times k$ system (25). In practice, it is typical that only a relatively small number of the modal shapes is necessary, i.e., $k \ll N$. The problem of finding a few modal shapes S_k within a certain frequency range is one of the well-known algebraic eigenvalue problems in numerical linear algebra [3].

[1] By a simple exercise, it can be shown that the definition of poles and residues of the system is independent of the choice of the expansion point s_0.

4.3 Padé approximation and moment-matching

Note that the transfer function $H(s)$ of (22) is a rational function. More precisely, $H(s) \in \mathcal{R}_{N-1,N}$, where N is the state-space dimension of (19).[2] The Taylor series expansion of $H(s)$ of (23) about s_0 is given by

$$\begin{aligned} H(s) &= l^T (I - (s-s_0)A)^{-1} r \\ &= l^T r + (l^T A r)(s-s_0) + (l^T A^2 r)(s-s_0)^2 + \cdots \\ &= m_0 + m_1(s-s_0) + m_2(s-s_0)^2 + \cdots, \end{aligned} \tag{26}$$

where $m_j = l^T A^j r$ for $j = 0, 1, 2, \ldots$, are called *moments* about s_0. Since our primary concern is large state-space dimension N, we seek to approximate $H(s)$ by a rational function $H_n(s) \in \mathcal{R}_{n-1,n}$ over the range of frequencies of interest, where $n \le N$. A natural choice of such a rational function is a Padé approximation. A function $H_n(s) \in \mathcal{R}_{n-1,n}$ is said to be an nth Padé approximant of $H(s)$ about the expansion point s_0 if it matches with the moments of $H(s)$ as far as possible. Precisely, it is required that

$$H(s) = H_n(s) + \mathcal{O}\left((s-s_0)^{2n}\right). \tag{27}$$

For a thorough treatment of Padé approximants, we refer the reader to [11]. Note that equation (27) presents $2n$ conditions on the $2n$ degrees of freedom that describe any function $H_n(s) \in \mathcal{R}_{n-1,n}$. Specifically, let

$$H_n(s) = \frac{P_{n-1}(s)}{Q_n(s)} = \frac{a_{n-1}s^{n-1} + \cdots + a_1 s + a_0}{b_n s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + 1}, \tag{28}$$

where b_0 is chosen to be equal to 1, which eliminates an arbitrary multiplicative factor in the definition of $H_n(s)$. Then the coefficients $\{a_j\}$ and $\{b_j\}$ of polynomials $P_{n-1}(s)$ and $Q_n(s)$ can be computed as follows. Multiplying $Q_n(s)$ on both sides of (27) yields

$$H(s)Q_n(s) = P_{n-1}(s) + \mathcal{O}\left((s-s_0)^{2n}\right). \tag{29}$$

Comparing the first n $(s-s_0)^k$-terms of (29) for $k = 0, 1, \ldots, n-1$ shows that the coefficients $\{b_j\}$ of the denominator polynomial $Q_n(s)$ satisfy the following system of simultaneous equations:

$$\begin{bmatrix} m_0 & m_1 & \cdots & m_{n-1} \\ m_1 & m_2 & \cdots & m_n \\ \vdots & \vdots & \cdots & \vdots \\ m_{n-1} & m_n & \cdots & m_{2n-2} \end{bmatrix} \begin{bmatrix} b_n \\ b_{n-1} \\ \vdots \\ b_1 \end{bmatrix} = - \begin{bmatrix} m_n \\ m_{n+1} \\ \vdots \\ m_{2n-1} \end{bmatrix}. \tag{30}$$

The coefficient matrix of (30) is called the *Hankel matrix*, denoted as M_n. Once the coefficients $\{b_j\}$ are computed, then by comparing the second n $(s-s_0)^k$-terms of (29) for $k = n, n+1, \ldots, 2n-1$, we see that the coefficients $\{a_j\}$ of the numerator polynomial $P_{n-1}(s)$ can be computed according to

$$\begin{aligned} a_0 &= m_0 \\ a_1 &= m_0 b_1 + m_1 \\ &\vdots \\ a_{n-1} &= m_0 b_{n-1} + m_1 b_{n-2} + \cdots + m_{n-2} b_1 + m_{n-1}. \end{aligned}$$

[2] $\mathcal{R}_{m,n}$ denotes the set of rational functions with real numerator polynomial of degree at most m and real denominator polynomial of degree at most n.

It is clear that $H_n(s)$ defines a unique nth Padé approximant of $H(s)$ if, and only if, the Hankel matrix M_n is nonsingular. We will assume that this is the case for all n.

This formulates the framework of the asymptotic waveform evaluation (AWE) techniques as they are known in circuit simulation, first presented in [53] around 1990. The manuscript [18] has a complete treatment of the AWE technique and its variants. A survey of the Padé techniques for model reduction of linear systems is also presented in the earlier work [15]. It is well-known that in practice, the Hankel matrix M_n is generally extremely ill-conditioned. Therefore, the computation of Padé approximants using explicit moments is inherently numerically unstable. Indeed, this approach can be used only for very small values of n, such as $n \leq 20$, even with some sophisticated schemes to improve the conditioning of the underlying Hankel matrix M_n. As a result, the approximation range of a computed Padé approximant is limited to only a narrow frequency range around the selected expansion point s_0. A large number of expansion points is generally required for the approximation of the transfer function $H(s)$ over a broad frequency range of interest. Since for each expansion point s_0, one has to be concerned with the cost of applying the matrix $A = -(G + s_0 C)^{-1} C$, which is generally the most expensive part of the overall computational cost, one would like to use as few expansion points as possible by increasing the order n of Padé approximants with a selected expansion point s_0. Fortunately, numerical difficulties associated with explicit moments can be remedied by exploiting the well-known connection between the Padé approximants and the Lanczos process. We will discuss this connection in the next section.

4.4 Krylov subspaces and the Lanczos process

A Krylov subspace is a subspace spanned by a sequence of vectors generated by a given matrix and a vector as follows. Given a matrix A and a starting vector r, the nth Krylov subspace $\mathcal{K}_n(A, r)$ is spanned by a sequence of n column vectors:

$$\mathcal{K}_n(A, r) = \operatorname{span}\{\, r, Ar, A^2 r, \ldots, A^{n-1} r \,\}.$$

This is sometimes called the right Krylov subspace. When the matrix A is nonsymmetric, there is a left Krylov subspace generated by A^T and a starting vector l defined by

$$\mathcal{K}_n(A^T, l) = \operatorname{span}\{\, l, A^T l, (A^T)^2 l, \ldots, (A^T)^{n-1} l \,\}.$$

Note that the first $2n$ moments $\{m_j\}$ of $H(s)$ in (26), which define the Hankel matrix M_n in the Padé approximant (30), are connected with Krylov subspaces through computing the inner products between the left and right Krylov sequences:

$$m_{2j} = \left((A^T)^j l\right)^T \cdot \left(A^j b\right)^T, \quad m_{2j+1} = \left((A^T)^j l\right)^T \cdot \left(A^{j+1} b\right)^T,$$

for $j = 1, 2, \ldots, n-1$. Therefore, loosely speaking, the left and right Krylov subspaces contain the desired information of moments, but the vectors $\{A^j r\}$ and $\{(A^T)^j l\}$ are unsuitable as basis vectors. The remedy is to construct more suitable basis vectors:

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

such that they span the same desired Krylov subspaces, specifically,

$$\mathcal{K}_n(A, r) = \operatorname{span}\{\, v_1, v_2, \ldots, v_n \,\} \quad \text{and} \quad \mathcal{K}_n(A^T, l) = \operatorname{span}\{\, w_1, w_2, \ldots, w_n \,\}.$$

It is well-known that the nonsymmetric Lanczos process is an elegant way to generate the desired basis vectors of two Krylov subspaces [46]. Given a matrix A, a right starting vector r and a left starting vector l, the nonsymmetric Lanczos process generates the desired basis vectors $\{v_i\}$ and $\{w_i\}$, known as the *Lanczos vectors.* Moreover, these Lanczos vectors are constructed to be biorthogonal

$$w_j^T v_k = 0, \quad \text{for all } \ j \neq k. \tag{31}$$

The Lanczos vectors can be generated by two three-term recurrences. These recurrences can be stated compactly in matrix form as follows

$$\begin{array}{rcl} AV_n & = & V_n T_n + \rho_{n+1} v_{n+1} e_n^T, \\ A^T W_n & = & W_n \widetilde{T}_n + \eta_{n+1} w_{n+1} e_n^T, \end{array}$$

where T_n and $\widetilde{T}_n$ are the tridiagonal matrices

$$T_n = \begin{bmatrix} \alpha_1 & \beta_2 & & \\ \rho_2 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_n \\ & & \rho_n & \alpha_n \end{bmatrix} \quad \widetilde{T}_n^T = \begin{bmatrix} \alpha_1 & \gamma_2 & & \\ \eta_2 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \gamma_n \\ & & \eta_n & \alpha_n \end{bmatrix}$$

and they are related by a diagonal similarity transformation $\widetilde{T}_n^T = D_n T_n D_n^{-1}$, where $D_n = W_n^T V_n = \text{diag}(\delta_1, \delta_2, \ldots, \delta_k)$. The projection of the matrix A onto $\mathcal{K}_n(A, r)$ and orthogonally to $\mathcal{K}_n(A^T, l)$ is represented by

$$W_n^T A V_n = D_n T_n.$$

If the nonsymmetric Lanczos process is carried to the end with N being the last step, then it can be viewed as a means of tridiagonalizing A by a similarity transformation:

$$V_N^{-1} A V_N = T_N, \tag{32}$$

where T_N is a tridiagonal matrix, with T_n as its $n \times n$ leading principal submatrix, $n \leq N$. The Lanczos vectors are determined up to a scaling. We use the scaling $\|v_j\|_2 = \|w_j\|_2 = 1$ for all j.

An algorithm template for the basic nonsymmetric Lanczos process is presented as the following:

Nonsymmetric Lanczos process: Let A be a real nonsymmetric matrix, r and l are real vectors. Then the following procedure computes the tridiagonal matrices T_n and $\widetilde{T}_n$, and the biorthogonal Lanczos vectors v_k and w_k.

```
$\rho_1 = \|r\|_2$
$\eta_1 = \|l\|_2$
$v_1 = r/\rho_1$
$w_1 = l/\eta_1$
For $k = 1, 2, \ldots, n$
    $\delta_k = w_k^T v_k$
    $\alpha_k = w_k^T A v_k / \delta_k$
    $\beta_k = (\delta_k/\delta_{k-1})\eta_k$
    $\gamma_k = (\delta_k/\delta_{k-1})\rho_k$
    $v = Av_k - v_k\alpha_k - v_{k-1}\beta_k$
    $w = A^T w_k - w_k\alpha_k - w_{k-1}\gamma_k$
    $\rho_{k+1} = \|v\|_2$
    $\eta_{k+1} = \|w\|_2$
    $v_{k+1} = v/\rho_{k+1}$
    $w_{k+1} = w/\eta_{k+1}$
```

We note that the nonsymmetric Lanczos process could stop prematurely due to $\delta_k = 0$ (or $\delta_k \approx 0$ considering the finite precision arithmetic). This is called *breakdown.* Our assumption of the nonsingularity of the Hankel matrix M_n guarantees that no breakdown occurs, see [52]. In practice, the problem is curable by a variant of the nonsymmetric Lanczos process, for example, a look-ahead scheme is proposed in [29]. An implementation of the nonsymmetric Lanczos process with a look-ahead scheme to overcome the breakdown can be found in QMRPACK [30].

4.5 Padé approximation using the Lanczos process

Let us first consider the nonsymmetric Lanczos process as a process for tridiagonalizing the matrix A. Then by (32), the transfer function $H(s)$ of the original system (19) can be rewritten as

$$H(s) = (l^T r)\, e_1^T (I - (s-s_0)T_N)^{-1} e_1 = (l^T r)\, \frac{\det(I - (s-s_0)T_N')}{\det(I - (s-s_0)T_N)} \tag{33}$$

where T_N' is an $(N-1) \times (N-1)$ matrix obtained by deleting the first row and column of T_N. Note that for the second equality, we have used the following Cauchy-Binet theorem to the matrix $I - (s-s_0)T_N$:

$$(I - (s-s_0)T_N) \cdot \mathrm{adj}(I - (s-s_0)T_N) = \det(I - (s-s_0)T_N) \cdot I,$$

where $\mathrm{adj}(X)$ stands for the classical adjugate matrix made up of the $(N-1) \times (N-1)$ cofactors of X. Expression (33) is called the *zero-pole representation.* It is clear that the poles of $H(s)$ can be computed from the eigenvalues of the $N \times N$ tridiagonal matrix T_N and the zeros of $H(s)$ from the eigenvalues of the $(N-1) \times (N-1)$ tridiagonal matrix T_N'. More precisely, the poles are given by $p_j = s_0 + 1/\lambda_j$, $\lambda_j \in \lambda(T_N)$, and the zeros by $z_j = s_0 + 1/\lambda_j'$, $\lambda_j' \in \lambda(T_N')$.

Now, let us turn to large-scale linear systems where the order N of the matrix A is too large to fully tridiagonalize, and where the Lanczos process terminates at n $(\leq N)$ Then it is natural to define an n-th reduced-order approximation of the transfer function $H(s)$ as

$$H_n(s) = (l^T r)\, e_1^T (I - (s-s_0)T_n)^{-1} e_1, \tag{34}$$

where T_n is the $n \times n$ leading principal submatrix of T_N, as generated by the first n steps of the nonsymmetric Lanczos process. In analogy to (33), we have the zero-pole representation of $H_n(s)$:

$$H_n(s) = (l^T r)\, \frac{\det(I - (s-s_0)T_n')}{\det(I - (s-s_0)T_n)}, \tag{35}$$

where T_n' is an $(n-1)\times(n-1)$ matrix obtained by deleting the first row and column of T_n.

Now, the question is: what is $H_n(s)$? The answer, which seems surprising to many first-time readers, is that $H_n(s)$ is the Padé approximation of $H(s)$ as computed by using explicit moments in section 4.3. To show this, let us first recall the following proposition, which was originally developed in [64] for a convergence analysis of the Lanczos algorithm for eigenvalue problems.

Proposition 4.1 *If T_n is the $n\times n$ leading principal submatrix of T_N, where $n\le N$. Then for any $0\le j\le 2n-1$,*

$$e_1^T T_N^j e_1 = e_1^T T_n^j e_1$$

and for $j=2n$,

$$e_1^T T_N^{2n} e_1 = e_1^T T_n^{2n} e_1 + \beta_2\beta_3\cdots\beta_n\beta_{n+1}\cdot\rho_2\rho_3\cdots\rho_n\rho_{n+1}.$$

A verification of this proposition can be easily carried out by induction. By Proposition 4.1, we immediately see that the first $2n$ moments of $H(s)$ and $H_n(s)$ are matched:

$$m_j = l^T A^j r = (l^T r) e_1^T T_N^j e_1 = (l^T r) e_1^T T_n^j e_1 = \widehat{m}_j \tag{36}$$

for $j=0,1,\ldots,2n-1$. Furthermore, by Taylor expansions of $H(s)$ and $H_n(s)$ about s_0 and (36), we have

$$H(s) = H_n(s) + (l^T r)\left(\prod_{j=2}^{n+1}\beta_j\prod_{j=2}^{n+1}\rho_j\right)(s-s_0)^{2n} + \mathcal{O}\left((s-s_0)^{2n+1}\right).$$

Therefore, we conclude that $H_n(s)$ is a Padé approximant of $H(s)$.

This Lanczos-Padé connection at least goes back to [39] and [40]. The work of [26, 32] advocates the use of the Lanczos-Padé connection instead of the mathematical equivalent, but numerically unstable AWE method [53] in the circuit simulation community. The Lanczos-based Padé approximation method has become known as the PVL (Padé Via Lanczos) method, as coined in [26]. An overview of various Krylov methods and their applications in model reduction for state-space control models in control system theory is presented in [13]. The presentation style here partially follows the work of [10]. In the following, we present two examples, one from circuit simulation and one from structural dynamics, as empirical validation of the efficiency of the PVL method. We note that in both cases, we only use one expansion point s_0 over the entire range of frequencies of interest. However, the degree of the underlying Padé approximants constructed via the Lanczos process is as high as 60, which seems to be an impossible mission by using explicit moment-matching as discussed in section 4.3.

Example 4. This example demonstrates the efficiency of the PVL method for a popular circuit problem, which simulates a lumped element network generated by a 3-D electromagnetic problem modeled via the partial element equivalent circuit (PEEC) model [18, 26]. The PEEC model is obtained by appropriate discretizations of the boundary integral formulation of Maxwell's equations for the electric and magnetic fields at any point in a conductor [55]. The order of the system matrices C and G is 306. To capture the dynamic

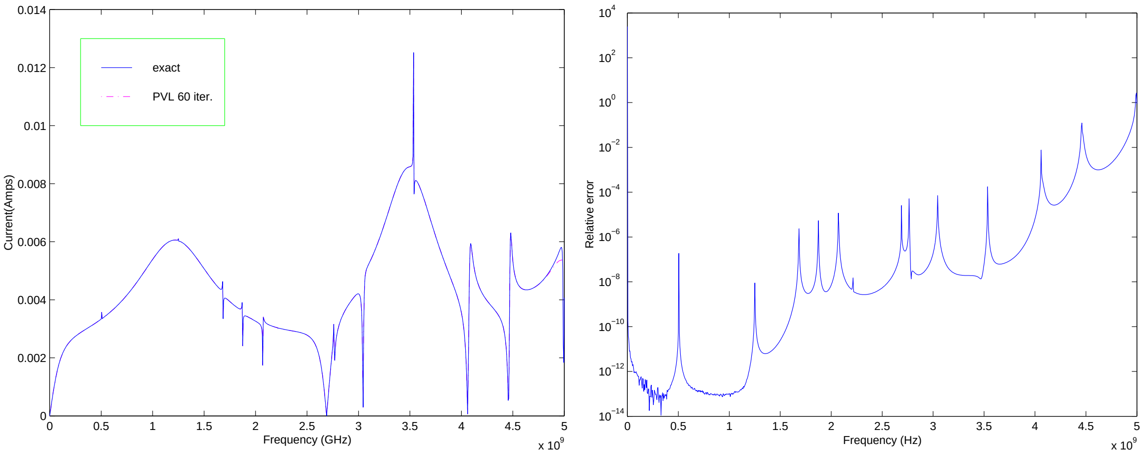

Figure 2: PEEC example, $|H(\mathtt{i}\omega)|$ and PVL $|H_{60}(\mathtt{i}\omega)|$ (left) and relative error $|H(\mathtt{i}\omega) - H_{60}(\mathtt{i}\omega)|/|H(\mathtt{i}\omega)|$ (right).

behavior of the transfer function $H(\mathtt{i}\omega)$ over the broad frequency range $[\omega_{\min}, \omega_{\max}] = [1, 5 \times 10^9]$, it is necessary to evaluate $H(s)$ at a large number of frequency points. We used a total of 1001 frequency points. On the left of Figure 2, we plot the absolute values of $H(\mathtt{i}\omega)$ and the Padé approximant $H_{60}(\mathtt{i}\omega)$ of order 60 generated by the PVL method with only a single expansion $s_0 = 2\pi \times 10^9$. Note that it is nearly indistinguishable from the curve of $|H(\mathtt{i}\omega)|$. The right plot of Figure 2 is the relative error between $H(\mathtt{i}\omega)$ and $H_{60}(\mathtt{i}\omega)$.

Example 5. This example is from dynamics analysis of automobile brakes, extracted from MSC/NASTRAN, a finite element analysis software for structural dynamics [45]. The order of the mass matrix M and stiffness matrix K is 834. The transfer function is of the form $H(\mathtt{i}\omega) = l^T(K - \omega^2 M)^{-1} b$. The expansion point is chosen as $s_0 = 0$. A total of 501 frequency points is evaluated between 0 and 10000Hz. The left plot of Figure 3 shows the magnitudes of the original transfer function $H(\mathtt{i}\omega)$ and the reduced-order transfer function $H_{45}(\mathtt{i}\omega)$ after 45 PVL iterations. The right plot of Figure 3 shows the relative error between $H(\mathtt{i}\omega)$ and $H_{45}(\mathtt{i}\omega)$.

4.6 Error estimation

An important question associated with the PVL method is how to determine the order n of a Padé approximant $H_n(s)$, or equivalently, the number of steps of the Lanczos process in order to achieve a desired accuracy of the approximation. In [9], through an algebraic derivation, it is shown that forward error between the full-order transfer function $H(s)$ and the reduced-order transfer function $H_n(s)$ is given by

$$H(s) - H_n(s) = (l^T r)\left(\frac{\rho_{n+1}\eta_{n+1}}{\delta_n}\right)\left[\sigma^2 \tau_{n1}(\sigma)\tau_{1n}(\sigma)\right]\gamma_{n+1}(\sigma), \qquad (37)$$

where $\sigma = s - s_0$, $\tau_{1n}(\sigma) = e_1^T(I - \sigma T_n)^{-1} e_n$, $\tau_{n1}(\sigma) = e_n^T(I - \sigma T_n)^{-1} e_1$, and $\gamma_{n+1}(\sigma) = w_{n+1}^T(I - \sigma A)^{-1} v_{n+1}$. From (37), we see that there are essentially two factors to determine the forward error of the PVL method, namely $\sigma^2 \tau_{n1}(\sigma)\tau_{1n}(\sigma)$ and $\gamma_{n+1}(\sigma)$. Numerous numerical

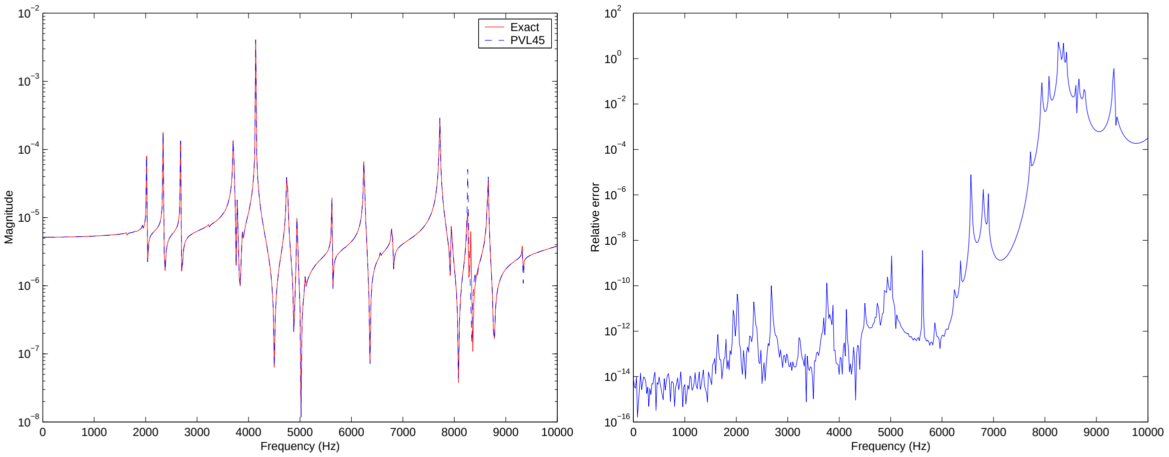

Figure 3: Automobile brake example, $|H(\mathtt{i}\omega)|$ and PVL $|H_{45}(\mathtt{i}\omega)|$ (left) and relative error $|H(\mathtt{i}\omega) - H_{45}(\mathtt{i}\omega)|/|H(\mathtt{i}\omega)|$ (right).

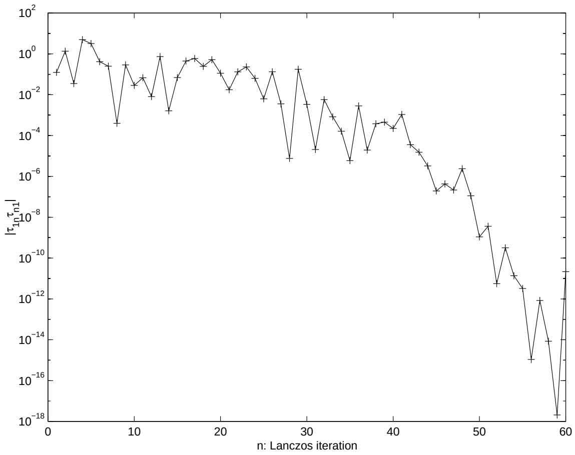

Figure 4: Convergence of $|\tau_{n1}(\sigma)\tau_{1n}(\sigma)|$ for a fixed $\sigma = s - s_0$.

experiments indicate that the first factor, which can be easily computed during the PVL approximation, is the primary contributor to the convergence of the PVL approximation, while the second factor tends to be steady when n increases. Note that $\tau_{n1}(\sigma)$ and $\tau_{1n}(\sigma)$ are the $(1, n)$ and $(n, 1)$ elements of the inverse of the tridiagonal matrix $I - \sigma T_n$. This is in agreement with the rapid decay phenomenon observed in the inverse of a band matrix [49]. Figure 4 shows typical convergence behavior of the factor $|\sigma^2\tau_{n1}(\sigma)\tau_{1n}(\sigma)|$ for a fixed σ. The direct computation of the second factor $\gamma_{n+1}(\sigma)$ would cost just as much as computing the original transfer function. It is advocated that $w_{n+1}^T A v_{n+1}$ be used as an estimation of the factor $\gamma_{n+1}(\sigma)$ near convergence. With this observation, it is possible to implement the PVL method with an adaptive stopping criteria to determine the required number of Lanczos iterations, see [9]. Related work for error estimation can be found in [43, 41] and recently in [51].

More efficient and accurate error estimations of the PVL approximation and its extension to the other moment-matching based Krylov techniques warrant further study. One alternative approach is to use the technique of backward error analysis. By some algebraic derivation, it can be shown that the reduced-order transfer function $H_n(s)$ of (34) can be interpreted as the exact transfer function of a perturbed full-order system. Specifically,

$$H_n(s) = (l^T r) \cdot e_1^T (I - (s - s_0)T_n)^{-1} e_1 = l^T \left[I - (s - s_0)(A + F_n)\right]^{-1} r,$$

where

$$F_n = -\frac{1}{\delta_n} \begin{bmatrix} v_n & v_{n+1} \end{bmatrix} \begin{bmatrix} 0 & \eta_{n+1} \\ \rho_{n+1} & 0 \end{bmatrix} \begin{bmatrix} w_n^T \\ w_{n+1}^T \end{bmatrix}$$

Therefore, one may use $\|F_n\|$ for monitoring convergence. However, it is observed that this is generally a conservative monitor and often does not indicate practical convergence. An open problem is to find an optimal normwise relative backward error

$$\eta(\epsilon) = \min \left\{ \epsilon \,:\, l^T \left[I - (s - s_0)(A + F_n)\right]^{-1} r = H_n(s),\ \|F_n\| \leq \epsilon \|A\| \right\}.$$

With this optimal backward error and perturbation analysis of the transfer function $H(s)$, one may be able to derive a more efficient error estimation scheme.

5 Reduced-order modeling

The Padé approximation of the transfer function $H(s)$ using the Lanczos process naturally leads to an efficient method for reduced-order modeling of large-scale linear dynamical systems. The desired attributes of a reduced-order model include replacing the full-order system by a system of the same type but with a much smaller state-space dimension such that it has an admissible error between the full-order and reduced-order models. Furthermore, the reduced-order model should also preserve essential properties of the full-order system. Such a reduced-order model would let designers efficiently analyze and synthesize the dynamical behavior of the original system within a tight design cycle. Specifically, given the linear dynamical system (19), we want to find a reduced-order linear system of the same form

$$\left\{ \begin{array}{rcl} C_n \dot{z}(t) + G_n z(t) & = & B_n u(t), \\ \tilde{y}(t) & = & L_n^T z(t), \end{array} \right. \tag{38}$$

where $z(t) \in \mathcal{R}^n$, $C_n, G_n \in \mathcal{R}^{n \times n}$, $B_n \in \mathcal{R}^{n \times m}$, $L_n \in \mathcal{R}^{n \times p}$, and $\tilde{y}(t) \in \mathcal{R}^p$. The state-space dimension n of (38) should generally be much smaller than the state-space dimension N of (19), i.e., $n \ll N$. Meanwhile, the output $\tilde{y}(t)$ of (38) approximates the output $y(t)$ of (19) in accordance with some criteria for all u in the class of admissible input functions. Furthermore, the reduced-order system (38) should preserve essential properties of the full-order system (19), such as stability and passivity. We refer to [1, 14] for the definitions of these properties.

Note that the $p \times m$ matrix-valued transfer function of the reduced-order model (19) is given by

$$H_n(s) = L_n^T (G_n + sC_n)^{-1} B_n.$$

Hence, for the steady-state analysis in the frequency domain, the objectives of constructing a reduced-order model (38) include that the reduced-order transfer function $H_n(s)$ should be an approximation of the transfer function $H(s)$ of the full-order model over the frequency range of interest with an admissible error, and that $H_n(s)$ preserves essential properties of $H(s)$.

We now show how to construct a reduced-order model of the linear system (19) in the time domain for transient analysis. With a selected expansion point s_0 as for the steady-state analysis, the linear system (19) under the so-called "shift-and-invert" transformation becomes

$$\left\{ \begin{array}{rcl} -A\dot{x}(t) + (I + s_0 A)x(t) & = & r\,u(t), \\ y(t) & = & l^T x(t), \end{array} \right.$$

where $A = -(G + s_0 C)^{-1} C$ and $r = (G + s_0 C)^{-1} b$. Let V_n be the Lanczos vectors generated by the nonsymmetric Lanczos process with matrix A and starting vectors r and l as discussed in section 4.4. Then considering the approximation of the state vector $x(t)$ by another state vector, constrained to stay in the subspace spanned by the columns of V_n, namely,

$$x(t) \approx V_n z(t) \quad \text{for some } z(t) \in \mathcal{R}^N,$$

yields an over-determined linear system with respect to the state variable $z(t)$:

$$\left\{ \begin{array}{rcl} -AV_n\dot{z}(t) + (I + s_0 A)V_n z(t) & = & r\,u(t), \\ \tilde{y}(t) & = & l^T V_n z(t). \end{array} \right.$$

After left-multiplying the first equation by W_n^T, we have

$$\left\{ \begin{array}{rcl} -W_n^T AV_n\dot{z}(t) + W_n^T (I + s_0 A)V_n z(t) & = & W_n^T r\,u(t), \\ \tilde{y}(t) & = & l^T V_n z(t). \end{array} \right.$$

Then an n-th reduced-order model of the linear system (19) in the time domain is naturally defined as

$$\left\{ \begin{array}{rcl} C_n\dot{z}(t) + G_n z(t) & = & r_n u(t), \\ \tilde{y}(t) & = & l_n^T z(t), \end{array} \right. \tag{39}$$

where $C_n = -W_n^T AV_n$, $G_n = W_n^T(I + s_0 A)V_n$, $r_n = W_n^T r$ and $l_n = V_n^T l$. By using the governing equations of the nonsymmetric Lanczos process presented in section 4.4, the quadruples (C_n, G_n, r_n, l_n) can be simply expressed as $C_n = -T_n$, $G_n = (I_n - s_0 T_n)$, $r_n = \rho_1 e_1$, and $l_n = \eta_1 \delta_1 e_1$.

Example 6. Figure 5 shows transient analysis of a small RLC network presented in [18, p.29]. The system matrices C and G have order 11. An input excitation $u(t)$ of 0.1 ns rise/fall and 0.3 ns duration was simulated. The convergence for orders 2 and 4 of the reduced-order models in the time domain is shown in Figure 5. The expansion point is chosen to be $s_0 = \pi \times 10^9$.

The continual and pressing need for accurately and efficiently simulating dynamical behavior of complex physical systems arising from computational science and engineering applications has led to increasingly large and complex models. Reduced-order modeling techniques play an indispensable role in providing an efficient computational prototyping tool to replace such a large-scale model by an approximate smaller model, which is capable of capturing dynamical behavior and preserving essential properties of the larger one.

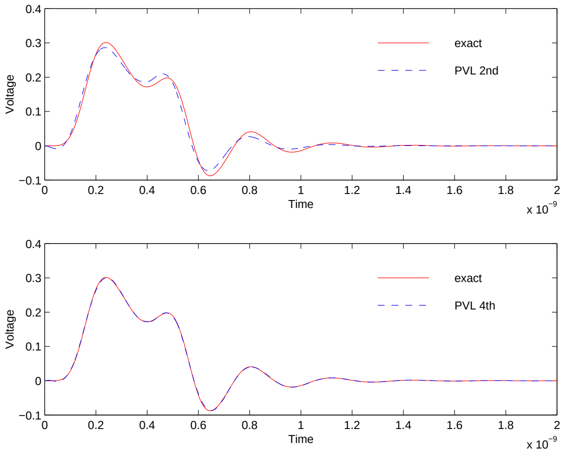

Figure 5: RLC network transient responses: 2nd and 4th order PVL approximation.

An accurate and effective reduced-order model can be applied for steady state analysis, transient analysis, or sensitivity analysis of such a system. As a result, it can significantly reduce design time and allow for aggressive design strategies. Such a computational prototyping tool would let designers try "what-if" experiments in hours instead of days.

A myriad of reduced-order modeling methods has been presented in various fields. We can categorize most of these methods into two classes. The first comprises techniques based on the optimization of a reduced-order model according to a suitably chosen criterion. The second class consists of methods that preserve exactly a limited number of parameters of the original model. The work of De Villemagne and Skelton [23] in 1987 provides a survey of early work on these methods. Over the past several years, Krylov-subspace-based techniques, such as the one presented in section 4, have emerged as one of the most powerful tools for reduced-order modeling of large-scale systems. We refer the reader to recent surveys [27, 28, 2, 4] for further study on the topic.

Acknowledgments

We are grateful to the organizers A. Bourlioux, M. Gander, K. Mikula and G. Sabidussi for their invitation and hospitality. Z. Bai is supported in part by NSF grant ACI-9813362 and DOE grant DE-FG03-94ER25219, and in part by a MICRO project (#00-005) from University of California and MSC.software Corporation. G. Golub is supported in part by NSF grant CCR9971010.

References

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

[2] A. C. Antoulas and D. C. Sorensen. Approximation of large-scale dynamical systems: An overview. Technical report, Electrical and Computer Engineering, Rice University, Houston, TX, Feb. 2001.

[3] 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*, Philadelphia, 2000. SIAM.

[4] Z. Bai. Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems. *Applied Numerical Mathematics*, 2002. (to appear).

[5] Z. Bai, M. Fahey, and G. Golub. Some large scale matrix computation problems. *J. of Comput. and Appl. Math.*, 74:71–89, 1996.

[6] Z. Bai, M. Fahey, G. Golub, and E. Menon, M.and Richter. Computing partial eigenvalue sum in electronic structure calculations. Scientific Computing and Computational Mathematics Program SCCM-98-03, Stanford University, 1998.

[7] Z. Bai and G. Golub. Bounds for the trace of the inverse and the determinant of symmetric positive definite matrices. *Annals of Numer. Math.*, 4:29–38, 1997.

[8] Z. Bai and G. Golub. Some unusual matrix eigenvalue problems. In J. Palma, J. Dongarra, and V. Hernandez, editors, *Proceedings of VECPAR'98 - Third International Conference for Vector and Parallel Processing*, volume 1573 of *Lecture Notes in Computer Science*, pages 4–19. Springer, 1999.

[9] Z. Bai, R. D. Slone, W. T. Smith, and Q. Ye. Error bound for reduced system model by Padé approximation via the Lanczos process. *IEEE Trans. Computer-Aided Design*, CAD-18:133–141, 1999.

[10] Z. Bai and Q. Ye. Error estimation of the Padé approximation of transfer functions via the Lanczos process. *Electronic Trans. Numer. Anal.*, 7:1–17, 1998.

[11] G. A. Baker, Jr. and P. Graves-Morris. *Padé Approximants.* Cambridge University Press, 1996.

[12] R. P. Barry and R. K. Pace. Monte Carlo estimates of the log determinant of large sparse matrices. *Linear Algebra and its Applications*, 289:41–54, 1999.

[13] D. L. Boley. Krylov subspace methods on state-space control models. *Circuit Systems Signal Process*, 13:733–758, 1994.

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

[15] A. Bultheel and M. van Barvel. Padé techniques for model reduction in linear system theory: a survey. *J. Comp. Appl. Math.*, 14:401–438, 1986.

[16] D. Calvetti, G. H. Golub, and L. Reichel. A computable error bound for matrix functionals. *J. Comput. Appl. Math.*, 103:301–306, 1999.

[17] D. Calvetti, S. Morigi, L. Reichel, and F. Sgallari. Computable error bounds and estimates for the conjugate gradient method. *Numer. Alg.*, 25:75–88, 2000.

[18] E. Chiprout and M. S. Nakhla. *Asymptotic Waveform Evaluation.* Kluwer Academic Publishers, 1994.

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

[20] R. R. Craig, Jr. *Structural Dynamics: An Introduction to Computer Methods.* John Wiley & Sons, 1981.

[21] G. Dahlquist, S. C. Eisenstat, and G. H. Golub. Bounds for the error of linear systems of equations using the theory of moments. *J. Math. Anal. and Appl.*, 37:151–166, 1972.

[22] P. Davis and P. Rabinowitz. *Methods of Numerical Integration.* Academic Press, N.Y., 1984.

[23] C. De Villemagne and R. E. Skelton. Model reductions using a projection formulation. *Int. J. Control*, 46:2141–2169, 1987.

[24] S. Dong and K. Liu. Stochastic estimation with z_2 noise. *Physics Letters B*, 328:130–136, 1994.

[25] M. Fahey. *Numerical Computation of Quadratic Forms Involving Large Scale Matrix Functions.* PhD thesis, University of Kentucky, 1998.

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

[27] 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.

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

[29] R. W. Freund, M. H. Gutknecht, and N. M. Nachtigal. An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices. *SIAM J. on Sci. Comput.*, 14:137–158, 1993.

[30] R. W. Freund and N. M. Nachtigal. QMRPACK: a package of QMR algorithms. *ACM Trans. Math. Softw.*, 22:46–77, 1996.

[31] A. Frommer, T. Lippert, B. Medeke, and K. Schilling, editors. *Numerical Challenges in Lattice Quantum Chromodynamics*, Berlin, 2000. Springer Verlag. Lecture Notes in Computational Science and Engineering. 15.

[32] K. Gallivan, E. Grimme, and P. Van Dooren. Asymptotic waveform evaluation via a Lanczos method. *Appl. Math. Lett.*, 7:75–80, 1994.

[33] W. Gautschi. A survey of Gauss-Christoffel quadrature formulae. In P. L Bultzer and F. Feher, editors, *E. B. Christoffel – the Influence of His Work on on Mathematics and the Physical Sciences*, pages 73–157. Birkhauser, Boston, 1981.

[34] G. Golub. Some modified matrix eigenvalue problems. *SIAM Review*, 15:318–334, 1973.

[35] G. Golub and G. Meurant. Matrics, moments and quadrature. In D. F. Griffiths and G. A. Watson, editors, *Proceedings of the 15th Dundee Conference, June 1993.* Longman Scientific & Technical, 1994.

[36] G. Golub and G. Meurant. Matrices, moments and quadrature II: how to compute the norm of the error iterative methods. *BIT*, 37:687–705, 1997.

[37] G. Golub and Z. Strakoš. Estimates in quadratic formulas. *Numerical Algorithms*, 8:241–268, 1994.

[38] G. Golub and U. Von Matt. Generalized cross-validation for large scale problems. *J. Comput. Graph. Stat.*, 6:1–34, 1997.

[39] W. B. Gragg. Matrix interpretations and applications of the continued fraction algorithm. *Rocky Mountain J. of Math.*, 5:213–225, 1974.

[40] W. B. Gragg and A. Lindquist. On the partial realization problem. *Lin. Alg. Appl.*, 50:227–319, 1983.

[41] E. Grimme. *Krylov projection methods for model reduction.* PhD thesis, Univ. of Illinois at Urbana-Champaign, 1997.

[42] M. Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Commun. Statist. -Simula.*, 18(3):1059–1076, 1989.

[43] I. M. Jaimoukha and E. M. Kasenally. Oblique projection methods for large scale model reduction. *SIAM J. Matrix Anal. Appl.*, 16:602–627, 1997.

[44] T. Kailath. *Linear Systems.* Prentice-Hall, New York, 1980.

[45] L. Komzsik. *MSC/NASTRAN, Numerical methods User's Guide, Version 70.5.* The MacNeal-Schwendler Corporation, Los Angeles, 1998.

[46] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. Res. Natl. Bur. Stand*, 45:225–280, 1950.

[47] A. J. Laub. Efficient calculation of frequency response matrices from state space models. *ACM Trans. Math. Software*, 12:26–33, 1986.

[48] H. Madrid. private communications, 1995.

[49] G. Meurant. A review of the inverse of symmetric tridiagonal and block tridiagonal matrices. *SIAM J. Mat. Anal. Appl.*, 13:707–728, 1992.

[50] H.-D. Meyer and S. Pal. A band-Lanczos method for computing matrix elements of a resolvent. *J. Chem. Phys.*, 91:6195–6204, 1989.

[51] A. Odabasioglu, M. Celik, and L. T. Pileggi. Practical considerations for passive reduction of RLC circuits. In *Proc. Inter. Conf. on Computer-Aided Design*, pages 214–219, 1999.

[52] B. Parlett. Reduction to tridiagonal form and minimal realizations. *SIAM J. Mat. Anal. Appl.*, 13(2):567–593, 1992.

[53] L. T. Pillage and R. A. Rohrer. Asymptotic waveform evaluation for timing analysis. *IEEE Trans. Computer-Aided Design*, 9:353–366, 1990.

[54] David Pollard. *Convergence of Stochastic Processes.* Springer-Verlag, 1984.

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

[56] B. Sapoval, Th. Gobron, and A. Margolina. Vibrations of fractal drums. *Phy. Rev. Lett.*, 67(21):2974–2977, 1991.

[57] J. C. Sexton and D. H. Weingarten. The numerical estimation of the error induced by the valence approximation. *Nuclear Physics B (Proc. Suppl.)*, pages xx–xx, 1994.

[58] G. Szegö. *Orthogonal polynomials.* American Mathematical Society, third edition, 1974.

[59] S. Van Huffel and J. Vandewalle. *The Total Least Squares Problem: Computational aspects and analysis.* SIAM, Philadelphia, 1991.

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

[61] K. Willcox, J. Peraire, and J. White. An Arnoldi approach for generalization of reduced-order models for turbomachinery. FDRL TR-99-1, Fluid Dynamic Research Lab., Massachusetts Institute of Technology, 1999. submitted to Computers and Fluids.

[62] M. N. Wong. Finding $\mathrm{tr}(A^{-1})$ for large A by low discrepancy sampling. Presentation at the Fourth International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing, Hong Kong, China, 2000.

[63] S. Y. Wu, J. A. Cocks, and C. S. Jayanthi. An accelerated inversion algorithm using the resolvent matrix method. *Comput. Phy. Comm.*, 71:125–133, 1992.

[64] Q. Ye. A convergence analysis for nonsymmetric Lanczos algorithms. *Math. Comp.*, 56:677–691, 1991.