MODEL REDUCTION OF LARGE-SCALE DENSE SYSTEMS

Peter Benner
Zentrum für Technomathematik
Fachbereich 3
Universität Bremen
28334 Bremen (Germany)
benner@math.uni-bremen.de

Enrique S. Quintana-Ortí and Gregorio Quintana-Ortí
Departamento de Informática
Universidad Jaime I
12080 Castellón (Spain)
quintana@inf.uji.es **and** gquintan@inf.uji.es

ABSTRACT

In this paper we present a new implementation of model reduction methods based on balanced truncation. These methods are particularly suitable for large-scale dense systems and can be implemented efficiently on parallel computers. The system Gramians are computed in factored form where instead of (square, possibly singular) Cholesky factors as in standard implementations we use (rectangular) full-rank factors. This yields some favorable computational aspects in the subsequent computation of the reduced-order model, particularly for non-minimal systems. Our experimental results on a PC cluster show the performance and scalability of the parallel implementation of the new method.
Keywords: balanced truncation model reduction, square-root methods, sign function method, parallel computing.

1 INTRODUCTION

Consider the transfer function matrix (TFM)

$$G(\lambda) = C(\lambda I - A)^{-1}B + D,$$

and the associated stable, but not necessarily minimal, linear time-invariant (LTI) system,

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

with $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, and $D \in \mathbb{R}^{p\times m}$. The number of state variables n is the order of the system.

The aim here is to compute an LTI system, represented by A_r, B_r, C_r, D_r of order $r \ll n$ with m inputs and p outputs such that the TFM $G_r(\lambda) = C_r(\lambda I - A_r)^{-1}B_r + D_r$ of the reduced-order system approximates (1).

A large class of model reduction methods rely on state-space transformations with a nonsingular matrix $T \in \mathbb{R}^{n\times n}$. Partitioning $T = \left[T_l^T, L_l^T\right]^T \in \mathbb{R}^{n\times n}$ and $T^{-1} = [T_r, L_r]$, with $T_l \in \mathbb{R}^{r\times n}$ and $T_r \in \mathbb{R}^{n\times r}$, the reduced-order model is determined by the projections $A_r = T_l A T_r$, $B_r = T_l B$, $C_r = C T_r$, and $D_r = D$. In this paper we will focus on such methods. Moreover, hereafter, we assume that A is a stable matrix, i.e., the spectrum of A as denoted by $\lambda(A)$ is contained in the open left half plane. This implies that the system (1) is stable, that is, all the poles of $G(s)$ have strictly negative real parts.

The methods considered here are all based on balanced truncation (BT) methods [14, 17, 18, 19]. They belong to the family of absolute error methods, which try to minimize $\|\Delta_a\|_\infty = \|G - G_r\|_\infty$. Here, $\|G\|_\infty$ denotes the $\mathcal{L}_\infty$- or $\mathcal{H}_\infty$-norm of a stable, rational matrix function defined as

$$\|G\|_\infty = \operatorname*{ess\,sup}_{\omega\in\mathbb{R}} \sigma_{\max}(G(\jmath\omega)), \tag{2}$$

where $\jmath := \sqrt{-1}$ and $\sigma_{\max}(M)$ is the largest singular value of the matrix M.

Model reduction of large-scale systems arises, e.g., in control of large flexible mechanical structures or large power systems as well as in circuit simulation and VLSI design; see, e.g., [7, 8, 9] and the references therein. LTI systems with state-space dimension n of order 10^2 to 10^4 (and higher) are common in these applications. All BT model reduction methods for LTI systems with dense coefficient matrices have a computational cost of $\mathcal{O}(n^3)$ floating-point operations (flops). Large-scale applications thus require the use of parallel computing techniques to obtain the reduced-order system.

BT model reduction methods are strongly related to the controllability Gramian W_c and the observability Gramian W_o of the system (1). The Gramians are given by the solutions of two *Lyapunov equations*

$$\begin{array}{rcl} AW_c + W_cA^T + BB^T & = & 0, \end{array} \tag{3}$$
$$\begin{array}{rcl} A^TW_o + W_oA + C^TC & = & 0. \end{array} \tag{4}$$

As A is assumed to be stable, W_c and W_o are positive semidefinite and therefore can be factored as $W_c = S^TS$ and $W_o = R^TR$. The factors S and R are often called the *Cholesky factors* of the Gramians. The BT model reduction methods described in Section 2 use the product SR^T for computing the reduced-order model. Hence, all the methods considered have in common that in the first step, they compute the factors S and R of the solutions of (3) and

(4). This can be achieved by Hammarling's method [12] which computes the Cholesky factors directly without computing the solution matrix explicitly.

In Section 3 we describe a technique based on the sign function method for solving Lyapunov equations. We will focus here only on BT model reduction methods using the product SR^T and hence describe algorithms for computing factors of the Gramians directly. The sign function-based algorithm has the advantage of returning full-rank factorizations of the Gramians. That is, if the system is not minimal, i.e., W_c and/or W_o are singular, then the factors computed by this method are full-rank matrices having less rows than columns in contrast to Hammarling's method which returns in this case square but singular matrices. This saves some computational cost and workspace in the subsequent computations for computing the reduced-order model.

The parallel implementations of the algorithms employ the kernels in ScaLAPACK [6]. This is a public-domain library for parallel computers which provides scalable parallel distributed subroutines for many matrix algebra kernels in LAPACK [1]. Our experimental results in Section 4 report the performance and scalability of the parallel implementations on a cluster of Intel Pentium-II processors. Finally, in Section 5 the conclusions of our work are outlined.

2 BALANCED TRUNCATION MODEL REDUCTION METHODS

In this section we review different implementations of BT model reduction methods. Serial implementations of these algorithms can be found in the *Subroutine Library in Control Theory* – SLICOT[1] [3, 20]. We also consider a modification of the algorithms by replacing the Cholesky factors of the Gramians by full-rank factors, resulting in a smaller arithmetic cost and workspace requirement in case of non-minimal LTI systems.

Equations (3) and (4) have a unique pair of solutions as A is assumed to be stable. Moreover, as BB^T and C^TC are semidefinite matrices, the Gramians are also semidefinite and can be decomposed as $W_c = S^TS$ and $W_o = R^TR$. The factors $S, R \in \mathbb{R}^{n\times n}$ can be chosen triangular and are called the Cholesky factors of the Gramians. In [18] it is shown that BT model reduction can be achieved using SR^T instead of the product of the Gramians themselves. The resulting *square-root (SR) method* avoids working with the Gramians as their condition number can be up to the square of the condition number of the Cholesky factors. In these algorithms equations (3) and (4) are initially solved for the Cholesky factors without ever forming the Gramians explicitly. This can be achieved, e.g., by the algorithm described in [12]. Then the SVD of the product

$$SR^T = [U_1\ U_2] \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} \qquad (5)$$

is computed. Here, the matrices are partitioned at a given dimension r with $\Sigma_1 = \mathrm{diag}\,(\sigma_1, \ldots, \sigma_r)$ and $\Sigma_2 = \mathrm{diag}\,(\sigma_{r+1}, \ldots, \sigma_n)$ such that

$$\sigma_1 \geq \ldots \geq \sigma_r > \sigma_{r+1} \geq \ldots \geq \sigma_n \geq 0. \qquad (6)$$

If $\sigma_r > 0$ and $\sigma_{r+1} = 0$, i.e., $\Sigma_2 = 0$, then r is the *McMillan* degree of the given LTI system. That is, r is the state-space dimension of a minimal realization of the system.

For model reduction, r should be chosen in order to give a natural separation of the states, i.e., one should search for a large gap $\sigma_r \gg \sigma_{r+1}$ in the Hankel singular values [18].

The SVD in (5) can be obtained without explicitly forming the product of the Cholesky factors; for a discussion and further references see, e.g., [5].

The standard SR method then computes the projection matrices

$$T_l = \Sigma_1^{-1/2} V_1^T R \quad \text{and} \quad T_r = S^T U_1 \Sigma_1^{-1/2}, \qquad (7)$$

while the *balancing-free square-root (BFSR) method* [19] computes T_l and T_r from orthogonal bases for V_1^TR and S^TU_1, respectively. The SR method yields a balanced reduced-order model in contrast to the BFSR method. the latter has the advantage that the computed projection matrices are usually better conditioned if the original system is highly unbalanced.

In both cases, the reduced-order system is given by $D = D_r$ and

$$A_r = T_l A T_r, \quad B_r = T_l B, \quad C_r = C T_r. \qquad (8)$$

In case $\Sigma_1 > 0$ and $\Sigma_2 = 0$, (8) is a minimal realization of the TFM $G(\lambda)$ [18]. Hence, choosing r in (5) maximal such that $\sigma_r > 0$ and $\sigma_{r+1} = 0$, this procedure can be used to compute minimal realizations. Of course, the decision if $\sigma_{r+1} = 0$ has to be based on a numerically reliable criterion. Further, for a stable LTI system, choosing any partitioning in (5) with $\sigma_r > \sigma_{r+1}$ yields a stable, minimal, and balanced reduced-order model. The Gramians corresponding to the resulting TFM $G_r(\lambda)$ are both equal to Σ_1; see [18] for details.

So far we have assumed that the Cholesky factors S and R of the Gramians are square $n \times n$ matrices. For non-minimal systems, we have rank $S < n$ and/or rank $R < n$. Hence, rather than working with the Cholesky factors, we may use *full-rank factors* of

[1] `ftp://wgs.esat.kuleuven.ac.be/pub/WGS/SLICOT`

W_c, W_o. As W_c and W_o are positive semidefinite, there exist matrices $\hat{S} \in \mathbb{R}^{n_c \times n}$, $\hat{R} \in \mathbb{R}^{n_o \times n}$, such that $W_c = \hat{S}^T \hat{S}$, $W_o = \hat{R}^T \hat{R}$, and

$$\begin{aligned} n_c &:= \operatorname{rank} \hat{S} = \operatorname{rank} S = \operatorname{rank} W_c, \\ n_o &:= \operatorname{rank} \hat{R} = \operatorname{rank} R = \operatorname{rank} W_o. \end{aligned}$$

The full-rank factors $\hat{S}$, $\hat{R}$ can be obtained from S and R by omitting the trailing rows of the factors that must be zero due to the triangular form of S and R. A more efficient way is to compute $\hat{S}$ and $\hat{R}$ directly, e.g., with the algorithm described in Section 3. In the latter case, S and R can be defined by $S := \left[\ \hat{S} \quad 0\ \right]$, $R := \left[\ \hat{R} \quad 0\ \right]$. The SVD in (5) can then be obtained from that of $\hat{S}\hat{R}^T$; for details see [5]. This amounts in a significant savings of workspace and computational cost. For example, using the Golub–Reinsch SVD (see, e.g., [11]), (5) requires $22n^3$ flops and workspace for $2n^2$ real numbers while the SVD of $\hat{S}\hat{R}^T$ only requires $14n_c n_o^2 + 8n_o^3$ flops and workspace for $n_c^2 + n_o^2$ real numbers. In particular, for large-scale dynamical systems, the *numerical rank* of W_c, W_o and $\hat{S}$, $\hat{R}$ is often much less than n; see [5, 15]. Suppose that (numerically) $n_c = n_o = n/10$, then the SVD of $\hat{S}\hat{R}^T$ is 1000 times less expensive than (5) and only 1% of the workspace is required as compared to (5). Some more savings are obtained from the cheaper computation of the projection matrices.

In the next section we show how the full-rank factors $\hat{S}$ and $\hat{R}$ can be computed directly without having to compute S, R or even W_c, W_o first.

3 COMPUTING FULL-RANK FACTORS OF THE GRAMIANS

All the model reduction methods in the previous section require, as a first stage, the solutions (or their Cholesky factors) of two Lyapunov equations. In this section we describe Lyapunov equation solvers based on the matrix sign function. Details of the algorithms can be found in [4]. The algorithms are specially appropriate for parallel distributed memory computers. Here we describe in particular those modifications needed for computing full-rank factors of the Gramians.

Roberts [16] was the first to use the matrix sign function for solving Lyapunov (and Riccati) equations. He shows that the sign function method for the solution of the stable Lyapunov equation

$$A^T X + XA + Q = 0$$

simplifies to

$$\begin{aligned} A_{k+1} &\leftarrow \tfrac{1}{2}\left(A_k + A_k^{-1}\right), \\ Q_{k+1} &\leftarrow \tfrac{1}{2}\left(Q_k + A_k^{-T} Q_k A_k^{-1}\right), \end{aligned} \qquad k = 0, 1, \ldots, \tag{9}$$

with $A_0 = A$ and $Q_0 = Q$. Moreover,

$$X = \left(\lim_{k\to\infty} Q_k\right)/2.$$

Iteration (9) can be applied to (4) directly by setting $Q_0 := C^T C$ in order to obtain the observability Gramian as $W_o = Q_\infty/2$. For (3), we just have to replace A by A^T while setting $Q_0 := BB^T$. Both iterations can be combined so that W_c and W_o are computed simultaneously as follows

$A_0 := A$, $P_0 := BB^T$, $Q_0 := C^T C$.
`FOR` $k = 0, 1, 2, \ldots$ *until convergence*

$$\begin{aligned} c_k &\leftarrow |\det(A_k)|^{-\frac{1}{n}}, \\ A_{k+1} &\leftarrow \left(A_k + c_k^2 A_k^{-1}\right)/(2c_k), \\ P_{k+1} &\leftarrow \left(P_k + c_k^2 A_k^{-1} P_k (A_k^{-1})^T\right)/(2c_k), \\ Q_{k+1} &\leftarrow \left(Q_k + c_k^2 (A_k^{-1})^T Q_k A_k^{-1}\right)/(2c_k), \end{aligned}$$

Here, c_k is a scaling parameter used to accelerate convergence in the early steps of the iteration. At convergence, $W_c = P_\infty/2$ and $W_o = Q_\infty/2$.

Efficient convergence criteria for these iterations have been proposed in [4]. As A is a stable matrix, $A_\infty = \lim_{k\to\infty} A_k = -I_n$, and a suitable convergence criterion for the iterations is to stop when the relative error in the A-iterates drops below a tolerance threshold, i.e., if

$$\|A_k + I_n\|_\infty \leq \tau \|A_k\|_\infty$$

for a user-defined tolerance τ.

In [4, 13] iteration (9) was modified to obtain the Cholesky factors rather than the solutions themselves. The basic idea is that if $Q = C^T C$ and $C_0 = C$, the iterations for the Q_k's can be written in factored form as $Q_{k+1} = C_{k+1}^T C_{k+1}$ with

$$C_{k+1} \leftarrow \frac{1}{\sqrt{2c_k}} \begin{bmatrix} C_k \\ c_k C_k A_k^{-1} \end{bmatrix}. \tag{10}$$

Similarly, for iteration (9) with $B_0 := B$ we obtain

$$B_{k+1} \leftarrow \frac{1}{\sqrt{2c_k}} [\,B_k,\ c_k A_k^{-1} B_k\,]. \tag{11}$$

Using (10) and (11), the workspace required to store the B_k's, C_k's is doubled in each iteration step. An approach to limit the workspace requirements is proposed in [5]. With this method, the Cholesky factors of the Gramians are computed. Here, we propose a slightly different strategy taking into account that for large-scale non-minimal systems, the Cholesky factors are often of low (numerical) rank. In each step we compute a rank-revealing QR decomposition (see [11] and references therein) of the matrix in (10) while for (11), a rank-revealing LQ decomposition is computed:

$$\frac{1}{\sqrt{2c_k}}\left[\begin{array}{c} C_k \\ c_k C_k A_k^{-1}\end{array}\right] =$$
$$= \quad V_{k+1}\left[\begin{array}{cc} (R_{k+1})_{11} & (R_{k+1})_{12} \\ 0 & (R_{k+1})_{22}\end{array}\right]\Pi^C_{k+1},$$
$$\frac{1}{\sqrt{2c_k}}[\,B_k,\ c_k A_k^{-1}B_k\,] =$$
$$= \quad \Pi^B_{k+1}\left[\begin{array}{cc} (S_{k+1})_{11} & 0 \\ (S_{k+1})_{21} & (S_{k+1})_{22}\end{array}\right]U_{k+1}.$$

Here, V_{k+1}, U_{k+1} are orthogonal, Π^C_{k+1}, Π^B_{k+1} are permutation matrices, $(R_{k+1})_{11} \in \mathbb{R}^{r_{k+1}\times r_{k+1}}$ and $(S_{k+1})^T_{11} \in \mathbb{R}^{s_{k+1}\times s_{k+1}}$ are upper triangular, r_{k+1}, s_{k+1} are the estimated ranks of the iterates, and $(R_{k+1})_{22} \approx 0$, $(S_{k+1})_{22} \approx 0$. Setting $(R_{k+1})_{22} = 0$, $(S_{k+1})_{22} = 0$, we can proceed with the new iterates

$$C_{k+1} \quad\leftarrow\quad [\,(R_{k+1})_{11}\ (R_{k+1})_{12}\,]\Pi^C_{k+1} \in \mathbb{R}^{r_{k+1}\times n},$$
$$B_{k+1} \quad\leftarrow\quad \Pi^B_{k+1}\left[\begin{array}{c} (S_{k+1})_{11} \\ (S_{k+1})_{21}\end{array}\right] \in \mathbb{R}^{n\times s_{k+1}}.$$

Using the notation from Section 2, we have

$$\lim_{k\to\infty} B_k = \hat{S}^T, \quad \lim_{k\to\infty} C_k = \hat{R},$$

i.e., the iterates converge to the full-rank factors of the Gramians. As in most applications, $m, p \ll n$ and the *numerical rank* (see, e.g., [11]) of the Cholesky factors S, R of the system Gramians is also usually much smaller than n, this technique quite often saves a large amount of workspace and computational cost compared to using the technique described above where the workspace for the iterates is increased up to size $n \times n$. As outlined in Section 2, the product $\hat{S}\hat{R}^T$ is a small size, in general rectangular matrix for which the subsequent computations needed for model reduction are much cheaper than for the full or triangular $n \times n$ Cholesky factors as obtained by Hammarling's method which is used in the implementation of the balanced truncation model reduction algorithms described in [20].

The Lyapunov solvers described above are iterative procedures composed of LU and QR factorizations and triangular linear system solves. These matrix operations are of wide appeal for computer architectures that can take advantage of block-partitioned algorithms, and specially for parallel distributed architectures [6].

4 EXPERIMENTAL RESULTS

We have implemented the described BT model reduction algorithms based on computing full-rank factors of the Gramians using the sign function method as double precision Fortran 77 subroutines. The routines `PDGECMSR` and `PDGECMBS` implementing the SR and BFSR methods, respectively, are designed for parallel computing using a distributed memory model. The core computations are based on available parallel BLAS and ScaLAPACK [6] subroutines.

In this section we evaluate the accuracy and performance of our model reduction algorithms. For this purpose, we compare the parallel routines with the analogous serial algorithms in SLICOT. This library includes the Fortran 77 routine `AB09AD` which implements the SR and BFSR methods based on computing the Cholesky factors of the Gramians using Hammarling's method. For an extended set of test results we refer to [5].

4.1 Numerical accuracy

We first illustrate the reliability of the parallel model reduction algorithms by means of an example of moderate order.

We employ the difference in the magnitude of the frequency response between the TFMs of the original system and the reduced-order system in order to measure the reliability of the model reduction algorithms. An upper bound for this difference is given by

$$\|G - G_r\|_\infty \le 2\sum_{i=r+1}^{n}\sigma_i =: \delta,$$

where σ_k, $k = 1, \ldots, n$, are the Hankel singular values of the system and r is the order of the reduced-order model [10].

Example 1 This is Example 18 from [2]. The system matrices come from a linear-quadratic regulator problem of one-dimensional heat flow. The system is single-input/single-output ($m = p = 1$) and the order n is scalable. Increasing n results in a finer grid for the underlying finite-element (FE) discretization of the space variable. Here, we report results for $n = 1000$. The system is parameterized by the tuple $(a, b, c, \beta_1, \beta_2, \gamma_1, \gamma_2) = (10^{-3}, 1, 1, 0.2, 0.3, 0.2, 0.3)$. Note that transforming the discretized system to an LTI system as in (1) results in a system with dense coefficient matrix A although the FE matrices are sparse.

We obtain a reduced-order model of fixed order $r = 6$. The computed full-rank factors $\hat{S}$ and $\hat{R}$ are of size $32\times n$ and $37\times n$, resulting in a significant savings of computational work in the subsequent SVD as compared to using the $n \times n$ Cholesky factors.

Figure 1 reports the magnitude of the frequency response of the absolute error system obtained by the serial and parallel algorithms. The upper limit δ is attained for small frequencies whereas the absolute error decreases for larger frequencies. As there is almost no difference in the reduced-order models obtained by either the SR or BFSR methods in this example, we only report the results for the SR method. There is no visible difference in the obtained accuracy for the serial and the parallel model reduction routines. It should be noted that reducing the meshsize of the FE discretization which usually increases the accuracy of the computed solution of the underlying partial differential equation does not change the system dynamics here — the $r = 6$ model is almost identical for all sizes of n and approximates the system behavior satisfactorily; see [5] for details.

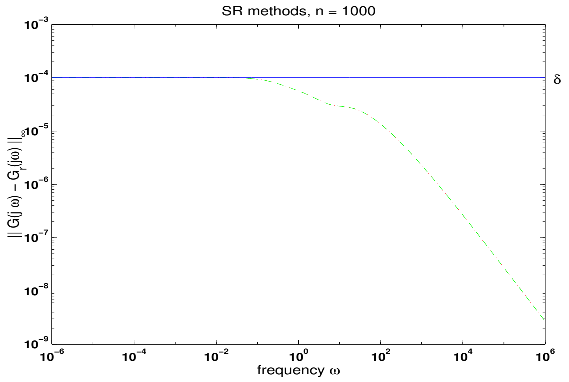

Figure 1: Frequency response (magnitude) of the error systems. Legend: "—" = upper bound given by δ, "– –" = PDGECMSR, "· · ·" = AB09AD.

4.2 Parallel performance

In this subsection we analyze the performance of the parallel algorithms on a parallel distributed cluster of 25 nodes. Each node consists of an Intel Pentium-II processor at 300 MHz, and 128 MBytes of RAM. We employ a BLAS library, specially tuned for the Pentium-II processor. The nodes are connected via a *Myrinet* crossbar network and the communication library BLACS is based on an implementation of the communication library MPI specially developed for this switch.

To evaluate the performance of our parallel model reduction algorithms we generate random LTI systems with McMillan degree $r = q_1$ and system Gramians of rank $n_c = q_1 + q_2$, $n_c = q_1 + q_3$; for details of the construction see [5].

We evaluate the reduction in the execution time achieved by the serial and parallel SR algorithms, `AB09AD` and `PDGECMSR`, respectively. We omit the results for the BFSR algorithms as they were closely similar. In this example, we set $n = 1000$, and choose $m = p = 400$, $q_1 = q_2 = q_3 = 200$. Figure 2 compares the execution times of the serial and the parallel algorithms as the number of nodes, n_p, is increased.

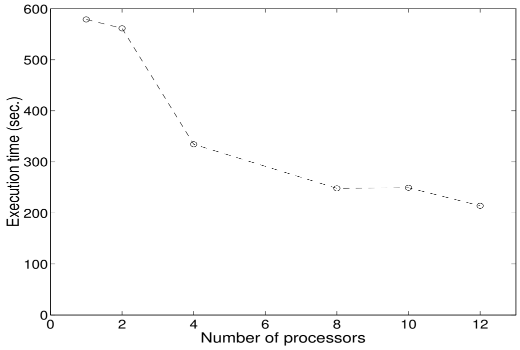

Figure 2: Execution time vs. number of nodes of the serial and parallel SR algorithms.

The figure shows that a considerable acceleration is achieved by the parallel algorithm `PDGECMSR` when compared to the serial algorithm `AB09AD` (= result on 1 node). This is partially due to the efficiency of the Lyapunov solvers used in our algorithms which yield full-rank factors rather than Cholesky factors. A comparison of the results on 2 and 4 nodes shows the parallelism of the `PDGECMSR` algorithm. The execution time is reduced by a factor of almost 2 (the number of resources, that is nodes, is doubled). Using 10–12 nodes does not achieve a significant reduction of the execution time in this case due to the small ratio $n/\sqrt{n_p}$.

Further experiments reported in [5] show a high scalability of the algorithms as there is only a minor decrease in the Mflop ratio per node as the number of nodes is increased up to 25 (a problem of size $n = 4000$). The scalability confirms that a larger problem can be solved by increasing proportionally the number of nodes employed.

5 CONCLUDING REMARKS

In this paper we have described the design and use of parallel algorithms for model reduction of large-scale systems. The methods are based on balanced truncation methods and benefit from low numerical rank of the Cholesky factors of the Gramian matrices. Using these algorithms it is possible to obtain a low-order approximation of large dense systems. Employing these reduced-order models, more advanced model reduction techniques can be applied to further reduce the system order.

Our experiments report similar numerical results for reliable serial model reduction algorithms from the SLICOT library and our model reduction approach, based on the matrix sign function. The results on a cluster of Intel Pentium-II nodes show the performance of our model reduction approach and the scalability of the parallel algorithms.

References

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK Users' Guide*. SIAM, Philadelphia, PA, second edition, 1995.

[2] P. Benner, A.J. Laub, and V. Mehrmann. A collection of benchmark examples for the numerical solution of algebraic Riccati equations I: Continuous-time case. Technical Report SPC 95_22, Fak. f. Mathematik, TU Chemnitz–Zwickau, FRG, 1995.

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

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

[5] P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. Balanced truncation model reduction of large-scale dense systems on parallel computers. Berichte aus der Technomathematik, Report 99–07, Fachbereich 3, Universität Bremen, FRG, September 1999.

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

[7] J. Cheng, G. Ianculescu, C.S. Kenney, A.J. Laub, and P. M. Papadopoulos. Control-structure interaction for space station solar dynamic power module. *IEEE Control Systems*, pages 4–13, 1992.

[8] P. Y. Chu, B. Wie, B. Gretz, and C. Plescia. Approach to large space structure control system design using traditional tools. *AIAA J. Guidance, Control, and Dynamics*, 13:874–880, 1990.

[9] R. 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, chapter 9, pages 435–498. Birkhäuser, Boston, MA, 1999.

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

[11] G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, third edition, 1996.

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

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

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

[15] T. Penzl. Eigenvalue decay bounds for solutions of Lyapunov equations: the symmetric case. *Sys. Control Lett.*, to appear.

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

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

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

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

[20] A. Varga. Model reduction routines for SLICOT. NICONET Report 1999–8, The Working Group on Software (WGS), June 1999. Available from `http://www.win.tue.nl/niconet`.