Remote Parallel Model Reduction of Linear Time-Invariant Systems Made Easy[⋆]

Peter Benner[1], Rafael Mayo[2], Enrique S. Quintana-Ortí[2], and Gregorio Quintana-Ortí[2]

[1] Institut für Mathematik, Technische Universität Berlin, D-10623 Berlin, Germany; `benner@math.tu-berlin.de`. Tel.: +30-314-28035. Fax: +30-314-79706.

[2] Depto. de Ingeniería y Ciencia de Computadores, Universidad Jaume I, 12071–Castellón, Spain; {`mayo,quintana,gquintan`}`@icc.uji.es`. Tel.: +34-964-728257. Fax: +34-964-728486.

Abstract. This paper describes a library for model reduction of large-scale, dense linear time-invariant systems on parallel distributed-memory computers. Our library is enhanced with a mail service which serves as a demonstrator of the capabilities of the library. Remote requests submitted via e-mail are executed on a cluster composed of Intel Pentium-II nodes connected via a Myrinet switch. Experimental results show the numerical and parallel performances of our model reduction routines.

Key words: Model reduction, absolute error methods, matrix sign function, mail service.

1 Introduction

Model reduction is of fundamental importance in many modeling and control applications involving linear time-invariant (LTI) systems. In state-space form, such systems are described by the following models:

Continuous LTI system:

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

Discrete LTI system:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k, \quad x_0 = x^0, \\ y_k &= Cx_k + Du_k, \quad k = 0, 1, 2, \ldots. \end{aligned} \tag{2}$$

In both models, $A \in \mathbb{R}^{n\times n}$ is the state matrix, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$, n is said to be the order of the system, and $x^0 \in \mathbb{R}^n$ is the initial state of the system. The associated transfer function matrix (TFM) is $G(s) =$

[⋆] Supported by the Fundació Caixa-Castelló/Bancaixa PI-1B2001-14 and CICYT TIC2002-04400-C03-01.

$C(sI - A)^{-1}B + D$. Here, we assume that the state matrix A is stable. In the continuous-time case, this implies that the spectrum of A is contained in the open left half plane while in the discrete-time case, the spectrum of A must lie inside the unit circle. This condition ensures that the system is stable, but we do not assume minimality of the system.

In the model reduction problem, we are interested in finding a reduced-order LTI system,

Continuous LTI system:

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

Discrete LTI system:

$$\begin{aligned} \hat{x}_{k+1} &= \hat{A}\hat{x}_k + \hat{B}\hat{u}_k, \quad \hat{x}_0 = \hat{x}^0, \\ \hat{y}_k &= \hat{C}\hat{x}_k + \hat{D}\hat{u}_k, \quad k = 0, 1, 2, \ldots, \end{aligned} \tag{4}$$

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

There is no general technique for model reduction that can be considered as optimal in an overall sense since the reliability, performance, and adequacy of the reduced system strongly depends on the system characteristics. The model reduction methods we study here rely on truncated state-space transformations where, given $Y = \left[T_l^T, L_l^T\right]^T \in \mathbb{R}^{n\times n}$ and $Y^{-1} = [T_r, L_r]$, with $T_l \in \mathbb{R}^{r\times n}$ and $T_r \in \mathbb{R}^{n\times r}$,

$$\hat{A} = T_l A T_r, \quad \hat{B} = T_l B, \quad \hat{C} = C T_r, \quad \text{and} \quad \hat{D} = D. \tag{5}$$

Model reduction methods based on truncated state-space transformations usually differ in the measure they attempt to minimize. Balanced truncation (BT) methods [16, 20, 22, 24], singular perturbation approximation (SPA) methods [15], and Hankel-norm approximation (HNA) methods [12] all belong to the family of absolute error methods, which try to minimize $\|\Delta_a\|_\infty = \|G - \hat{G}\|_\infty$. Here, $\|G\|_\infty$ denotes the $\mathcal{L}_\infty$- or $\mathcal{H}_\infty$-norm of a stable, rational matrix function defined as

$$\|G\|_\infty = \operatorname*{ess\,sup}_{\omega\in\mathbb{R}} \sigma_{\max}(G(\jmath\omega)), \tag{6}$$

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

The family of relative error model reduction methods attempt to minimize, on the other hand, the relative error $\|\hat{\Delta}\|_\infty$, defined implicitly by $G - \hat{G} = \hat{\Delta}G$. Among these, the balanced stochastic truncation (BST) methods [9, 13, 25] are specially interesting. Currently, BST methods are not included in the model reduction library and the associated mail service described here. Therefore, we do not include BST methods in our presentation.

Model reduction of large-scale systems arises, among others, 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, 11, 17]. LTI systems with state-space dimension n of order 10^2 to 10^4 are common in these applications.

All absolute error model reduction methods for LTI systems with dense state matrices have a computational cost of $\mathcal{O}(n^3)$ floating-point operations (flops). Large-scale applications thus clearly benefit from using parallel computing techniques to obtain the reduced system. The usual approach a few decades ago was to solve these problems on very expensive parallel supercomputers, but the situation is changing in the last years. Clusters constructed from commodity systems (personal computers and local and system area switches) and "open" software have started to be widely used for parallel scientific computing. The enormous improvements in the performance of the commodity hardware has provided an affordable tool for dealing with large-scale scientific problems. However, many end-users (control engineers, VLSI circuit designers, etc.) are not familiar with the basics of parallel algorithms and/or parallel architectures. Thus, if a parallel library of control algorithms is to be useful to a broader spectrum of the scientific community, it should be part of our work to provide, along with the library, the necessary tools to facilitate its use.

The tool we propose to use, e-mail, has been employed from the first days of the Internet not only for personal communication but also for submission of jobs to remote systems, access to databases, document retrieval, etc. In this work we explore the use of e-mail to provide a parallel model reduction service to the scientific community. The advantage of providing this service via e-mail is that the end-user remains completely isolated from the complexities of the software and the hardware system, and the maintenance of both. A secondary goal of this service is to serve as a demonstrator of the capabilities of the parallel library, which can be then downloaded and installed on any (parallel) architecture with MPI and ScaLAPACK.

Our library provides parallel routines for model reduction using the three mentioned absolute error methods. These approaches are briefly reviewed in Section 2. The contents and structure of the library are described in Section 3. The library is completed with a mail service, presented in Section 4, which allows remote access to our parallel cluster and enables the user to apply any of the model reduction algorithms on a specific system. Finally, the numerical and parallel performances of the routines in the parallel library is reported in Section 5, and some concluding remarks are given in Section 6.

2 Model Reduction via Absolute Error Methods

In this section we briefly review the methods for absolute error model reduction. Serial implementations of these algorithms can be found in the *Subroutine Library in Control Theory* – SLICOT[1]. Note that we use implementations that are different from the serial versions in SLICOT but result in the same reduced-order model. Our implementation is particularly efficient on parallel computing platforms.

[1] Available from `http://www.win.tue.nl/niconet/NIC2/slicot.html`.

All these methods are strongly related to the controllability Gramian W_c and the observability Gramian W_o of the LTI system. In the continuous-time case, the Gramians are given by the solutions of two coupled *Lyapunov equations*

$$\begin{array}{l} AW_c + W_cA^T + BB^T = 0, \\ A^TW_o + W_oA + C^TC = 0, \end{array} \tag{7}$$

while, in the discrete-time case, the Gramians are the solutions of two coupled *Stein equations*

$$\begin{array}{l} AW_cA^T - W_c + BB^T = 0, \\ A^TW_oA - W_o + C^TC = 0. \end{array} \tag{8}$$

As A is assumed to be stable, the Gramians W_c and W_o are positive semidefinite, and therefore there exist factorizations $W_c = S^TS$ and $W_o = R^TR$. Matrices S and R are called the *Cholesky factors* of the Gramians.

The Lyapunov equations are solved in our model reduction algorithms using the Newton iteration for the matrix sign function introduced by Roberts [19]. The Stein equations are solved using the Smith iteration proposed in [21]. Both iterations are specially adapted taking into account that, for large-scale non-minimal systems, the Cholesky factors are often of low (numerical) rank. By not allowing these factors to become rank-deficient, a large amount of workspace and computational cost can be saved. For a detailed description of this technique, see, e.g., [1].

Consider now the singular value decomposition (SVD)

$$SR^T = [U_1\ U_2] \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix}, \tag{9}$$

where the matrices are partitioned at a given dimension r such that $\Sigma_1 = \operatorname{diag}(\sigma_1, \ldots, \sigma_r)$, $\Sigma_2 = \operatorname{diag}(\sigma_{r+1}, \ldots, \sigma_n)$, $\sigma_j \geq 0$ for all j, and $\sigma_r > \sigma_{r+1}$. Here, $\sigma_1, \ldots, \sigma_n$ are known as the *Hankel singular values* of the system. If $\sigma_r > \sigma_{r+1} = 0$, then r is the *McMillan* degree of the system. That is, r is the state-space dimension of a minimal realization of the system.

The so-called *square-root* (SR) BT algorithms determine the reduced-order model in (5) using the projection matrices

$$T_l = \Sigma_1^{-1/2} V_1^T R \quad \text{and} \quad T_r = S^T U_1 \Sigma_1^{-1/2}. \tag{10}$$

In case $\Sigma_1 > 0$ and $\Sigma_2 = 0$, the reduced-order model computed by using T_l, T_r from (10) in (5) is a minimal realization of the TFM $G(s)$.

The *balancing-free square-root* (BFSR) BT algorithms often provide more accurate reduced-order models in the presence of rounding errors [24]. These algorithms share the first two stages (solving the coupled equations and computing the SVD of SR^T) with the SR methods, but differ in the procedure to obtain T_l and T_r. Specifically, the following two QR factorizations are computed,

$$S^TU_1 = [P_1\ P_2] \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix}, \qquad R^TV_1 = [Q_1\ Q_2] \begin{bmatrix} \bar{R} \\ 0 \end{bmatrix}, \tag{11}$$

where P_1, $Q_1 \in \mathbb{R}^{n\times r}$ have orthonormal columns, and $\hat{R}$, $\bar{R} \in \mathbb{R}^{r\times r}$ are upper triangular. The reduced system is then given by the projection matrices

$$T_l = (Q_1^T P_1)^{-1} Q_1^T, \qquad T_r = P_1, \tag{12}$$

and (5).

SR SPA or BFSR SPA reduced-order models can be obtained by first using the SR BT or BFSR BT algorithms, respectively, to obtain a minimal realization of the original system. Let the tuple $(\tilde{A}, \tilde{B}, \tilde{C}, \tilde{D})$ be a realization of this minimal system and partition

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

according to the desired size r of the reduced-order model, i.e., $A_{11} \in \mathbb{R}^{r\times r}$, $B_1 \in \mathbb{R}^{r\times m}$, $C_1 \in \mathbb{R}^{p\times r}$, etc. Then the SPA reduced-order model is obtained by applying the following formulae

$$\begin{aligned} \hat{A} &:= A_{11} - A_{12}(\gamma I - A_{22})^{-1} A_{21}, \\ \hat{B} &:= B_1 - A_{12}(\gamma I - A_{22})^{-1} B_2, \\ \hat{C} &:= C_1 - C_2(\gamma I - A_{22})^{-1} A_{21}, \\ \hat{D} &:= \tilde{D} - C_2(\gamma I - A_{22})^{-1} B_2, \end{aligned} \tag{13}$$

where $\gamma = 0$ for continuous LTI systems and $\gamma = 1$ for discrete LTI systems [15, 23, 24].

BT and SPA model reduction methods aim at minimizing the $\mathcal{H}_\infty$-norm norm of the error system $G - \hat{G}$. However, they usually do not succeed in finding an optimal approximation. Using the *Hankel norm* of a stable rational transfer function, defined by

$$\|G\|_H = \sigma_1(G), \tag{14}$$

where $\sigma_1(G)$ is the maximum Hankel singular value of G, it is possible to compute an approximation minimizing $\|G - \hat{G}\|_H$ for a given order r of the reduced-order system [12]. The derivation of a realization of $\hat{G}$ is quite involved. Due to space limitations, we refer the reader to [12, 27] and only describe the essential computational tools required in an implementation of the HNA method. In the first step, a balanced minimal realization of G is computed. This can be done using the SR version of the BT method described above. Then a transfer function $\tilde{G}(s) = \tilde{C}(sI - \tilde{A})^{-1}\tilde{B} + \tilde{D}$ is computed as follows: first, the order r of the reduced-order model is chosen such that the Hankel singular values of G satisfy

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

Then, by applying appropriate permutations, the balanced transformation of G is re-ordered such that the Gramians become

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

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

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

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

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

Here, $U := (C_2^T)^\dagger B_2$, where $M^\dagger$ denotes the pseudoinverse of M, and $\Gamma := \check{\Sigma}^2 - \sigma_{r+1}^2 I_{n-k}$. This step only requires matrix scaling, matrix multiplication, and a QR decomposition. Now we can compute an additive decomposition of $\tilde{G}$ such that $\tilde{G}(s) = \hat{G}(s) + F(s)$ where $\hat{G}$ is stable and F is anti-stable. Then $\hat{G}$ is an optimal r-th order Hankel norm approximation of G. We have implemented the additive decomposition of $\tilde{G}$ via a block diagonalization of $\tilde{A}$, where we first compute a block Schur form using the sign function of $\tilde{A}$ and then annihilate the off-diagonal block by solving a Sylvester equation using again a sign function-based iterative solution procedure. For further details of the implementation of the HNA method and computational aspects see [3].

3 Parallelization, structure and contents of the Library

The absolute error methods basically perform matrix operations such as the solution of linear systems and linear matrix equations, and the computation of matrix products and matrix decompositions (LU, QR, SVD, etc.). All these operations are basic matrix algebra kernels parallelized in ScaLAPACK and PBLAS. Thus, our parallel model reduction library PLICMR[2] heavily relies on the use of the available parallel infrastructure in ScaLAPACK, the serial computational libraries LAPACK and BLAS, and the communication library BLACS (see Figure 1).

Details of the contents and parallelization aspects of PLICMR are given, e.g., in [1, 2].

The functionality and naming convention of the parallel routines closely follow analogous routines from SLICOT. The contents of PLICMR are structured in four main groups:

- Model reduction computational routines.
  - `pab09ax`: BT method.
  - `pab09bx`: SPA method.

[2] `http://spine.act.uji.es/~plicmr`

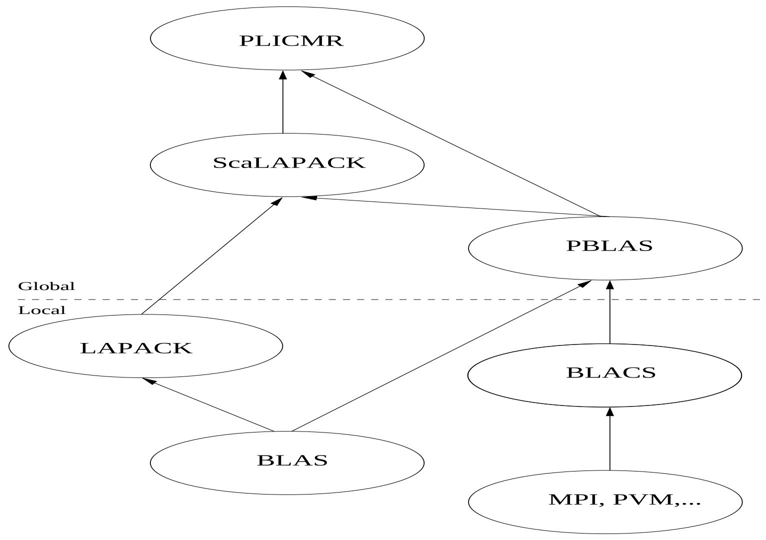


Fig. 1. PLICMR and the underlying libraries.

  - `pab09cx`: HNA method.
- Matrix functions.
  - `pmb05rd`: Computation of the sign function of a matrix.
- Linear matrix equation solvers.
  - `psb03odc`: Solution of coupled Lyapunov equations (7) using the Newton iteration for the matrix sign function.
  - `psb03odd`: Solution of coupled Stein equations (8) using the Smith iteration.
  - `psb04md`: Solution of a Sylvester equation using the Newton iteration for the matrix sign function.
- Linear algebra.
  - `pmb03td`: Computation of the SVD of a product of two rectangular matrices.
  - `pmb03ox`: Estimation of the numerical rank of a triangular matrix.

A standardized version of the library is integrated into the subroutine library PSLICOT [5], with parallel implementations of a subset of SLICOT. It can be downloaded from `ftp://ftp.esat.kuleuven.ac.be/pub/WGS/SLICOT`. The version maintained at `http://spine.act.uji.es/~plicmr` might at some stages contain more recent updates than the version integrated into PSLICOT. The library can be installed on any parallel architecture where the above-mentioned computational and communication libraries are available. The efficiency of the parallel routines will depend on the performance of the underlying libraries for matrix computation (BLAS) and communication (usually, MPI or PVM).

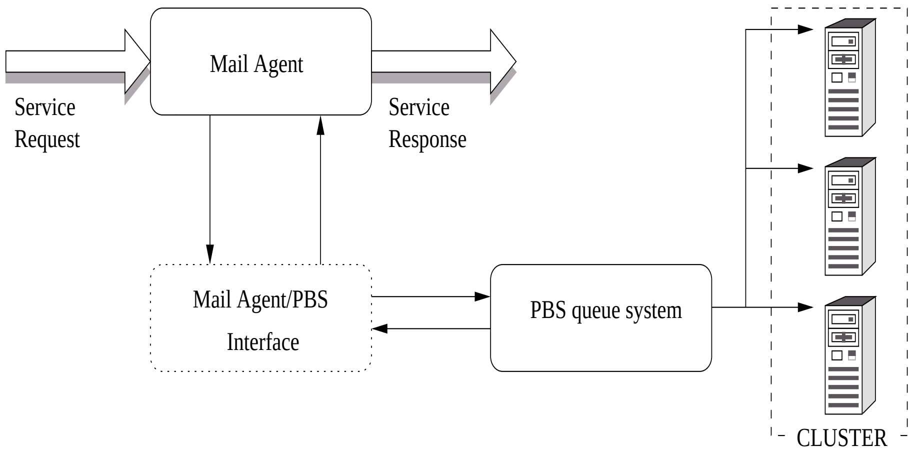


Fig. 2. Components of the mail service system.

4 Mail Service

In this section we describe the access procedure and the structure of the service for parallel model reduction via e-mail. This service is provided on a parallel cluster[3], composed of 32 Intel Pentium-II personal computers, running a Linux operating system, and connected with a 1 Gbps Myrinet network.

The user can submit a job using any standard application for sending an e-mail message to `plicmr@spine.act.uji.es`. A service request specifies the parameters of the job (like the model reduction algorithm to employ, the dimensions of the system, the desired order for the reduced system, etc.) in the body of the message. Validation of access is performed via a login and password included in the e-mail. Different compression tools are supported to reduce the size of the files specifying the system matrices, provided as attachments. The results of the execution are returned to the user via e-mail, with the matrices of the reduced system as attached (compressed) files.

The service and the specific formats of the e-mail body and attachments are further described in `http://spine.act.uji.es/~plicmr`.

The model reduction mail server is structured as a Mail Agent (MA) and a set of scripts which define the interface between this agent and PBS (portable batch system), a batch queuing and workload management system (see Figure 2).

The MA is the operating system mail daemon that processes e-mails. On reception of an e-mail addressed to `plicmr@spine.act.uji.es`, the MA creates a process, PLICMR-MA, which then employs the preprocessing scripts in the MA/PBS Interface to extract the contents of the e-mail, save the system matrices in the local file system, generate the submission scripts for PBS, and submit

[3] The cluster is owned by the Parallel Scientific Research Group, at the University Jaume I; `http://spine.act.uji.es/psc.html`.

the job. PBS is in charge of launching the execution of the job in the cluster and return the result to the PLICMR-MA using the postprocessing (epilogue) scripts in the MA/PBS Interface. The PLICMR-MA then generates an appropriate e-mail with the matrices of the reduced-order system and uses the MA to send it back to the user. Figure 3 illustrates the overall process, beginning with the reception of an e-mail with a service request from a remote user, and ending when an e-mail, containing the results of the execution of the request, is returned to the user.

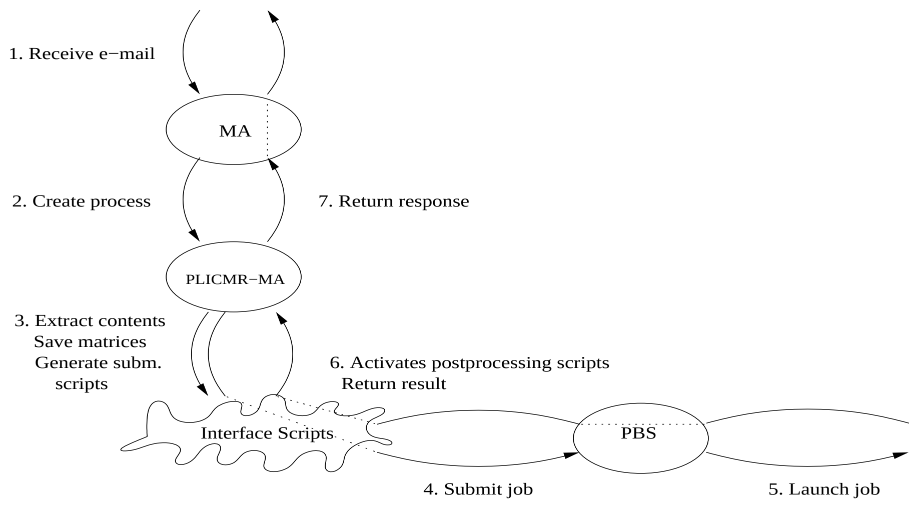


Fig. 3. Actors and actions involved in processing a service request for parallel model reduction.

5 Experimental Results

All the experiments presented in this section were performed on Intel Pentium-II processors using IEEE double-precision floating-point arithmetic ($\varepsilon \approx 2.2204 \times 10^{-16}$). The parallel algorithms were evaluated on a parallel distributed cluster of 32 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 as part of the ATLAS and ITXGEMM projects [26, 14], that achieves around 180 Mflops (millions of flops per second) for the matrix product (routine `DGEMM`). The nodes are connected via a *Myrinet* crossbar network; the communication library BLACS is based on an implementation of the communication library MPI specially developed and tuned for this network. The performance of the interconnection network was measured by a simple loop-back message transfer resulting in a latency of 33 μsec and a bandwidth of 200

Mbit/sec. We made use of the LAPACK, PBLAS, and ScaLAPACK libraries wherever possible.

We only report results using the BT BFSR algorithms for a continuous LTI system. Similar results were obtained for the SPA and HNA algorithms and for discrete LTI systems.

5.1 Numerical performance

In this subsection we compare the reliability and adequacy of the reduced systems computed using the SLICOT routines with those of the PLiCMR routines. For this purpose, we evaluate the response of the reduced systems, $|G(\jmath\omega)|$, at frequencies ω, by means of two large-scale examples from the collection in [6].

The first example models the atmosphere in the midlatitude of the Pacific. The area is discretized using an horizontal division of 1000×1000 km^2 and a vertical division 10 km long. The time is discretized in periods of 9 hours. The model is composed of 598 states, a single input, and a single output.

The second example is full-order model (FOM) of a large dynamical system, and has six badly damped poles, i.e., three pairs of complex conjugate poles with large imaginary part which are responsible for the three significant peeks in the frequency response. Except for these peeks, the dynamics of this system are smooth as all the remaining poles are real and well separated from the imaginary axis. There is no difficulty to be expected in reducing the order of this system significantly.

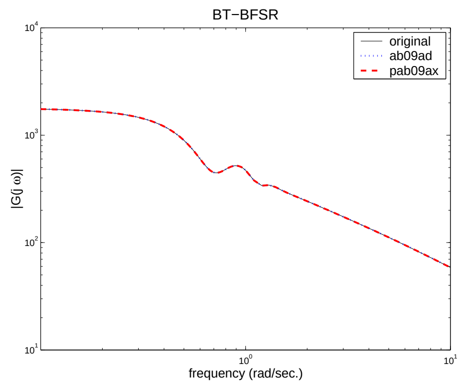
Figure 4 reports the frequency responses of the original system and the reduced models of orders r=9 and 10 for the atmosphere and the FOM examples, respectively. In both cases, no noticiable difference is encountered between the behaviour of the reduced systems computed by means of the SLICOT (`ab09ad`) and the PLiCMR (`pab09ax`) routines. As a matter of fact, in both examples the frequency response of the reduced systems perfectly match that of the original system for all frequencies.

5.2 Parallel performance

We employ two different large-scale examples to evaluate the parallel performance of the routines in PLiCMR.

The first example comes from a finite element discretization of a steel cooling process described by a boundary control problem for a linearized 2-dimensional heat equation; see [18] and references therein. The system has 6 inputs and outputs, and $n = 821$ or 3113 states, depending on the meshsize. As there is no significant gap in the Hankel singular values of the system, in this experiment we computed a reduced system of fixed order $r = 40$.

The system with $n = 821$ was reduced using the SLICOT routines in about 4 minutes. The parallel routines computed the reduced system in half of this time, using 4 processors. However, the system with $n = 3113$ could not be solved using the serial routines as the system matrices were too large to be stored in a single



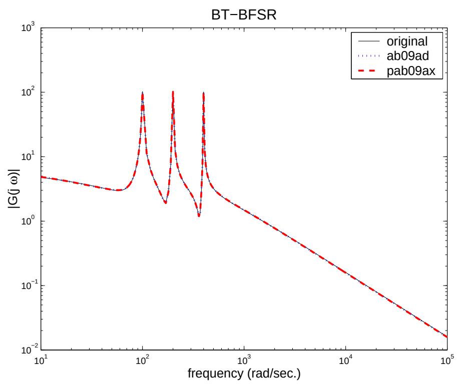


Fig. 4. Frequency response of the atmosphere (left) and the FOM (right) examples.

node. Our parallel routines provided the solution in around 15 minutes (using 16 processors).

The second example is a random continuous LTI system constructed as follows. First, we generate a random positive semidefinite diagonal Gramian $W_c = \operatorname{diag}(\Sigma_{q_1}, \Sigma_{q_2}, 0_{q_3}, 0_{q_4})$, where $\Sigma_{q_1} \in \mathbb{R}^{q_1 \times q_1}$ contains the desired Hankel singular values for the system and $\Sigma_{q_2} \in \mathbb{R}^{q_2 \times q_2}$. Then, we construct a random positive semidefinite diagonal Gramian $W_o = \operatorname{diag}(\Sigma_{q_1}, 0_{q_2}, \Sigma_{q_3}, 0_{q_4})$, with $\Sigma_{q_3} \in \mathbb{R}^{q_3 \times q_3}$. Next, we set A to a random stable diagonal matrix and compute $F = -(AW_c + W_cA^T)$ and $G = -(A^TW_o + W_oA)$. Thus,

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

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

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

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

We first evaluate the execution time of the parallel model reduction algorithms. In the example, we set $n = 1000$, $m = p = 100$, and $q_1 = q_2 = q_3 = 50$. As there is no noticeable gap in the Hankel singular value distribution of the system, we obtain a reduced-order model of order $r = 40$.

The left-hand plot in Figure 5 reports the execution time of the BFSR BT serial routine for model reduction in SLICOT and the corresponding parallel algorithm as the number of nodes, n_p, is increased. The results show a considerable acceleration achieved by the parallel algorithm (with even super speed-ups). This

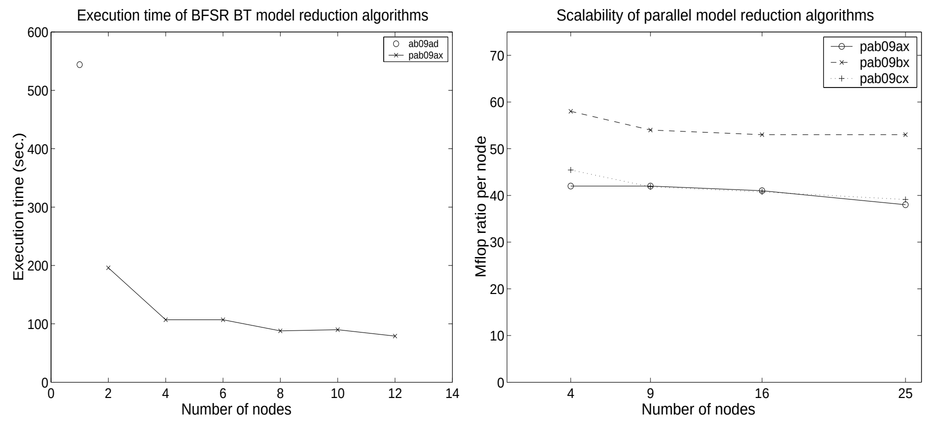


Fig. 5. Performance of the parallel model reduction algorithms.

is partially due to the efficiency of the Lyapunov solvers used in our algorithms which compute factors of the solutions in compact (full-rank) form instead of square matrices, thus requiring less computations. Comparison of the results on 2 and 4 nodes roughly shows the efficiency of the parallel algorithm. The execution time is reduced by a factor of almost 2 (the number of resources, that is nodes, is doubled). Using a larger number of nodes does not achieve a significant reduction of the execution time due to the small ratio $n/\sqrt{n_p}$.

We next evaluate the scalability of our parallel algorithms. As the memory of the system does not allow to test the serial algorithms on larger problems, in the experiment we fix the problem size per node using $n/\sqrt{n_p} = 800$, $m/\sqrt{n_p} = 400$, $p/\sqrt{n_p} = 400$, and $q/\sqrt{n_p} = 200$, with $q_1 = q_2 = q_3 = q$. In the right-hand plot in Figure 5 we report the Mflop ratio per node for the parallel model reduction algorithms. This is often referred as the *scaled speed-up*. The figure shows a high scalability of all three 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.

A more detailed analysis of the numerical accuracy and parallelism of the kernels in the parallel library for model reduction can be found in [4].

6 Concluding Remarks

In this paper we have described a parallel library for model reduction of large-scale, dense LTI systems using absolute error methods. Our experiments report a notable reduction in the execution time when the parallel algorithms are used. An important part of this reduction is due to the use of compact factors of the Gramian matrices.

Future extensions of the library will include parallel routines for relative error model reduction. A further extension will provide a service which allows an interaction between remote users and the parallel system through a web environment.

References

1. P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. Balanced truncation model reduction of large-scale dense systems on parallel computers. *Mathematical and Computer Modeling of Dynamical Systems*, 6(4):383–405, 2000.
2. P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. PSLICOT routines for model reduction of stable large-scale systems. In *Proc. 3rd NICONET Workshop on Numerical Software in Control Engineering, Louvain-la-Neuve, Belgium, January 19, 2001*, pages 39–44. 2001.
3. P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. Computing optimal Hankel norm approximations of large-scale systems. In preparation.
4. P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. Experimental evaluation of the parallel model reduction routines in PSLICOT. SLICOT Working Note 2002–7, `http://www.win.tue.nl/niconet/`,
5. I. Blanquer, D. Guerrero, V. Hernández, E.S. Quintana-Ortí, and P. Ruíz. Parallel-SLICOT implementation and documentation standards. SLICOT Working Note 1998–1, `http://www.win.tue.nl/niconet/`, September 1998.
6. Y. Chahlaoui and P. Van Dooren. A collection of benchmark examples for model reduction of linear time invariant dynamical systems. SLICOT Working Note 2002–2, February 2002. Available from `http://www.win.tue.nl/niconet/NIC2/reports.html`.
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. U.B. Desai and D. Pal. A transformation approach to stochastic model reduction. *IEEE Trans. Automat. Control*, AC–29:1097–1100, 1984.
10. E. Elmroth, P. Johansson, B. Kågström, and D. Kressner. A Web computing environment for the SLICOT library. SLICOT Working Note 2001–2, `http://www.win.tue.nl/niconet/`, June 2001.
11. L. Fortuna, G. Nummari, and A. Gallo. *Model Order Reduction Techniques with Applications in Electrical Engineering.* Springer-Verlag, 1992.
12. K. Glover. All optimal Hankel-norm approximations of linear multivariable systems and their L^∞ norms. *Internat. J. Control*, 39:1115–1193, 1984.
13. M. Green. Balanced stochastic realization. *Linear Algebra Appl.*, 98:211–247, 1988.
14. J. Gunnels, G. Henry, and R.A. van de Geijn. A family of high-performance matrix multiplication algorithms. In V.N. Alexander, J. Dongarra, B.A. Julianno, R.S. Renner, and C.J. Kenneth Tan, editors, *Computational Science - ICCS 2001, Part I, Lecture Notes in Computer Science 2073*, pages 51–60, 2001.
15. Y. Liu and B.D.O. Anderson. Controller reduction via stable factorization and balancing. *Internat. J. Control*, 44:507–531, 1986.

16. B. C. Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE Trans. Automat. Control*, AC-26:17–32, 1981.
17. C.R. Paul. *Analysis of Multiconductor Transmission Lines.* Wiley–Interscience, Singapur, 1994.
18. T. Penzl. Algorithms for model reduction of large dynamical systems. Technical Report SFB393/99-40, Sonderforschungsbereich 393 *Numerische Simulation auf massiv parallelen Rechnern*, TU Chemnitz, Germany, 1999.
19. 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).
20. M.G. Safonov and R.Y. Chiang. A Schur method for balanced-truncation model reduction. *IEEE Trans. Automat. Control*, AC–34:729–733, 1989.
21. R.A. Smith. Matrix equation $XA+BX = C$. *SIAM J. Appl. Math.*, 16(1):198–201, 1968.
22. 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.
23. A. Varga. Balancing-free square-root algorithm for computing singular perturbation approximations. In *Proc. of the 30th IEEE Conf. Dec. Control*, pages 1062–1065, 1991.
24. A. Varga. Efficient minimal realization procedure based on balancing. In *Prepr. of the IMACS Symp. on Modeling and Control of Technological Systems*, volume 2, pages 42–47, 1991.
25. A. Varga and K. H. Fasol. A new square–root balancing–free stochastic truncation model reduction algorithm. In *Prepr. 12th IFAC World Congress*, volume 7, pages 153–156, Sydney, Australia, 1993.
26. R. C. Whaley and A. Petitet and J. Dongarrra. Automated empirical optimizations of software and the ATLAS project. *Parallel Computing*, 27(1–2):3–15, 2001.
27. K. Zhou, J.C. Doyle, and K. Glover. *Robust and Optimal Control.* Prentice-Hall, Upper Saddle River, NJ, 1996.