# EFFICIENT NUMERICAL MODEL REDUCTION METHODS FOR DISCRETE-TIME SYSTEMS

**Peter Benner**[1], **Enrique S. Quintana-Ortí**[2], **and Gregorio Quintana-Ortí**[2]
[1]Zentrum für Technomathematik, Fachbereich 3
Universität Bremen, D–28334 Bremen (Germany)
[2]Departamento de Informática
Universidad Jaime I, 12080 Castellón (Spain)

**Abstract.** Model reduction is of fundamental importance in many modeling and control applications. Here we address algorithmic aspects of model reduction methods based on balanced truncation of linear discrete-time systems. The methods require the computation of the Gramians of the system. Using an accelerated fixed point iteration for computing the full-rank factors of the Gramians yields some favorable computational aspects, particularly for non-minimal systems. The computations only require efficient implementations of basic linear algebra operations readily available on modern computer architectures.

## Introduction

Consider the transfer function matrix (TFM) $G(\lambda) = C(\lambda I - A)^{-1}B + D$, and the associated stable, but not necessarily minimal, realization of a discrete, linear time-invariant (LTI) system,

$$x_{k+1} \;=\; Ax_k + Bu_k, \quad y_k \;=\; Cx_k + Du_k, \qquad k = 0, 1, 2, \ldots, \tag{1}$$

where $x_0 = \hat{x}$ is given and $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$. The number of state variables $n$ is said to be the order of the system. We assume that the spectrum of $A$ as denoted by $\Lambda(A)$ is contained in the open unit disk, i.e., $A$ is *(Schur) stable* or *convergent*. This implies that all the poles of the TFM $G(s)$ are contained in the open unit disk and hence the stability of the system (1).

We are interested in finding a reduced order LTI system,

$$\tilde{x}_{k+1} \;=\; \tilde{A}\tilde{x}_k + \tilde{B}\tilde{u}_k, \quad \tilde{y}_k \;=\; \tilde{C}\tilde{x}_k + \tilde{D}\tilde{u}_k, \qquad k = 0, 1, 2, \ldots, \tag{2}$$

of order $\ell$, $\ell \ll n$, with $\tilde{x}_0 = \tilde{\hat{x}}$, such that $\tilde{G}(\lambda) = \tilde{C}(\lambda I - \tilde{A})^{-1}\tilde{B} + \tilde{D}$ approximates $G(\lambda)$.

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 methods considered here are all based on balanced truncation (BT) methods [5, 8].

BT model reduction methods are based on information retrieved from the controllability and observability Gramians $W_c$ and $W_o$, respectively, of the system (1). These are given by the solutions of two "coupled" (as they share the same coefficient matrix $A$) *Stein equations* (or *discrete Lyapunov equations*)

$$AW_cA^T - W_c + BB^T = 0, \qquad A^TW_oA - W_o + C^TC = 0. \tag{3}$$

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 called the *Cholesky factors* of the Gramians. These factors are usually computed using Hammarling's method [3], yielding $S, R \in \mathbb{R}^{n\times n}$ upper triangular. Numerically reliable model reduction methods use these Cholesky factors rather than the Gramians themselves; see, e.g., [8, 9, 10]. However, if the system is not minimal, then $W_c$ and/or $W_o$ are singular and the Cholesky factors computed by this method are not full-rank matrices. In particular, for large systems, it can often be observed that the *numerical rank* of the Cholesky factors is much less than $n$.

We therefore describe in the next section a method based on an accelerated fixed point iteration that compute *full-rank factorizations* $W_c = \hat{S}^T\hat{S}$ and $\hat{W}_o = \hat{R}^T\hat{R}$, i.e., $\hat{S} \in \mathbb{R}^{\mathrm{rank}(W_c)\times n}$, $\hat{R} \in \mathbb{R}^{\mathrm{rank}(W_o)\times n}$. This can save a significant amount of computational cost and workspace solving the Stein equations and particularly in the subsequent computations for computing the reduced-order model. This approach also has some advantages regarding numerical robustness when determining the McMillan degree and a minimal realization of an LTI system.

We then describe the numerical implementation of model reduction methods based on balanced truncation using the full-rank factors of the Gramians. Numerical examples reporting the accuracy and performance of the resulting routines on serial and parallel computers will be provided in an extended version of this note[1]. Using parallel computers with distributed memory such as Linux-PC or workstation clusters allows the application of our methods to systems up to order $n = \mathcal{O}(10^5)$.

[1]Available from `http://www.math.uni-bremen.de/zetem/berichte.html`.

## Computing the Gramians

Consider the Stein equations in (3). Both can be formulated in a fixed point form $X = FXF^T + G$, from which it is straightforward to derive a fixed point iteration. This iteration converges to $X$ if $\rho(F) < 1$, where $\rho(F)$ denotes the spectral radius of $F$. That is, convergence is guaranteed under the given assumptions. The convergence rate of this iteration is linear. A quadratically convergent version of the fixed point iteration is suggested in [7]. Setting $X_0 := G$, $F_0 := F$, this iteration can be written as

$$X_{k+1} := F_k X_k F_k^T + X_k, \quad F_{k+1} := F_k^2, \qquad k = 0, 1, 2, \ldots. \tag{4}$$

The above iteration is referred to as the *squared Smith iteration*. As the two equations in (3) share the same coefficient matrix $A$, we can derive a coupled iteration to solve both equations simultaneously.

$$\begin{array}{rclrclrcl} X_0 & := & BB^T, & Y_0 & := & C^TC, & A_0 & := & A, \\ X_{k+1} & := & A_k X_k A_k^T + X_k, & Y_{k+1} & := & A_k^T Y_k A_k + Y_k, & A_{k+1} & := & A_k^2, \qquad k = 0, 1, 2, \ldots \end{array} \tag{5}$$

The most appealing feature of the squared Smith iteration regarding its implementation is that all the computational cost comes from matrix products. These can be implemented very efficiently on modern serial and parallel computers.

The convergence theory of the Smith iteration (4) derived in [7] yields that for $\rho(F) < 1$ there exist real constants $0 < \mu$ and $0 < \rho < 1$ such that $\|X - X_k\|_2 \leq \mu \|G\|_2 (1-\rho)^{-1} \rho^{2^k}$. This shows that the method converges for all equations with Schur stable coefficient matrices $F$.

In the case considered here, the "right-hand sides" of the Stein equations are positive semidefinite and are given in factored form $BB^T$ and $C^TC$. As $A$ is stable, Lyapunov stability theory (see, e.g., [4]) shows that the solution matrices are positive semidefinite and hence can be factored as $W_c = S^TS$, $W_o = R^TR$. The factors $S \in \mathbb{R}^{s\times n}$ and $R \in \mathbb{R}^{r\times n}$ are called the Cholesky factor of the solution. Usually, $s = r = n$ such that $S, R$ are square, possibly singular, matrices [3]. Here we will also use "Cholesky factor" to denote a *full-rank factor* of the solution, i.e., $\mathrm{rank}(S) = \mathrm{rank}(W_c) = s \leq n$, $\mathrm{rank}(R) = \mathrm{rank}(W_o) = r \leq n$. This has several advantages. The condition number of the solution matrix can be up to the square of that of its Cholesky factor. Hence, a significant increase in accuracy can often be observed working with the factor if the solution matrix is ill-conditioned. Moreover, for the case $r, s \ll n$ we will show in the next section that significant savings in computational work are obtained by using the full-rank factors rather than the square Cholesky factors for subsequent computations.

The coupled squared Smith iteration (5) can be modified to compute the full-rank factors of $W_c, W_o$ directly; see [1]. We focus here on the iteration for computing $W_c$; the iteration for $W_o$ can be treated analogously. Setting $G = BB^T$, the $X_k$ iteration in (5) can be re-written by setting $S_0 := B$ and

$$S_{k+1}S_{k+1}^T \quad \leftarrow \quad S_kS_k^T + A_k(S_kS_k^T)A_k^T = [\, S_k,\ A_kS_k \,] \begin{bmatrix} S_k^T \\ S_k^T A_k^T \end{bmatrix}, \quad \text{for } k = 0, 1, 2, \ldots. \tag{6}$$

In each step (6) the current iterate $S_k$ is augmented by $A_kS_k$ such that $S_{k+1} := [S_k,\ S_kA_k]$.

The above approach requires to double in each iteration step the workspace needed for the iterates $L_k$. Two approaches are possible to limit the required workspace to a fixed size [1]. We will focus here on one approach which is particularly appealing for the purpose of model reduction.

In each iteration step, we can compute a rank-revealing LQ factorization (see, e.g., [2, Chapter 5]) $\tilde{S}_{k+1} := \frac{1}{\sqrt{2}}[\, S_k,\ S_kA_k \,] = Q\hat{S}_{k+1}\Pi_{k+1}^T$. In that case, the next iterate $S_{k+1} \in \mathbb{R}^{n\times \mathrm{rank}(\tilde{S}_{k+1})}$ is obtained as the left $n \times \mathrm{rank}(\tilde{S}_{k+1})$ part of the product of the permutation matrix $\Pi_{k+1}$ and the lower triangular matrix $\hat{S}_{k+1}$, i.e.,

$$\begin{bmatrix} (\Pi_{k+1})_{11} & (\Pi_{k+1})_{12} \\ (\Pi_{k+1})_{21} & (\Pi_{k+1})_{22} \end{bmatrix} \begin{bmatrix} (\hat{S}_{k+1})_{11} & 0 \\ (\hat{S}_{k+1})_{21} & 0 \end{bmatrix} =: [\, S_{k+1},\ 0 \,],$$

starting from $S_0$ obtained by a rank-revealing LQ factorization of $B$. It follows that $\tilde{S}_{k+1}\tilde{S}_{k+1}^T = S_{k+1}S_{k+1}^T$ and $\sqrt{2}S = (\lim_{k\to\infty} S_k)^T$.

As $\mathrm{rank}(W_c)$ may be up to $n$, this requires a work space of size up to $2n \times n$. On the other hand, the (numerical) rank of the full-rank factors is often much less than $n$. Hence, the computational cost of performing LQ factorizations is well balanced by keeping the number of columns of $S_{k+1}$ small. Usually, the computational cost of this approach is less than the cost of Hammarling's method [3], see [1] for details. This approach can also be used to compute low-rank approximations to the full-rank factor by either increasing the tolerance threshold for determining the numerical rank or by fixing the allowed number of columns in $S_k$.

## Balanced Truncation Model Reduction using Full-Rank Factors

In [8] it is shown that BT model reduction can be achieved using $SR^T$ instead of the product of the Gramians themselves. Here, $S$ and $R$ denote the square, possibly singular Cholesky factors of $W_c$ and $W_o$, respectively. 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. The first step in the SR method is to compute the SVD

$$SR^T = [U_1\ U_2] \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix}. \tag{7}$$

Here, the matrices are partitioned at a given dimension $\ell$ with $\Sigma_1 = \operatorname{diag}(\sigma_1, \dots, \sigma_\ell)$ and $\Sigma_2 = \operatorname{diag}(\sigma_{\ell+1}, \dots, \sigma_n)$ such that $\sigma_1 \geq \sigma_2 \geq \dots \sigma_\ell > \sigma_{\ell+1} \geq \sigma_{r+2} \geq \dots \geq \sigma_n \geq 0$. If $\sigma_\ell > 0$ and $\sigma_{\ell+1} = 0$, i.e., $\Sigma_2 = 0$, then $r$ is the *McMillan* degree of the given discrete-time 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 look in the Hankel singular values $\sigma_k$, $k = 1, \dots, n$, for a large gap $\sigma_\ell \gg \sigma_{\ell+1}$ [8].

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 $\operatorname{rank}(S) < n$ and/or $\operatorname{rank}(R) < n$. Hence, rather than working with the Cholesky factors, we may use the full-rank factors $\hat{S}$, $\hat{R}$ of $W_c$, $W_o$ that are computed when using the method described in the last section. The SVD in (7) can then be obtained from that of $\hat{S}\hat{R}^T$ as follows. Here we assume $\operatorname{rank}(\hat{S}) =: s \geq r := \operatorname{rank}(\hat{R})$, the case $s < r$ can be treated analogously. Then we can compute the SVD

$$\hat{S}\hat{R}^T \ = \ \hat{U} \begin{bmatrix} \hat{\Sigma} \\ 0 \end{bmatrix} \hat{V}^T, \qquad \hat{\Sigma} = \operatorname{diag}(\sigma_1, \dots, \sigma_r), \tag{8}$$

where $\hat{U} \in \mathbb{R}^{s \times s}$, $\hat{V} \in \mathbb{R}^{r \times r}$. Now let $\ell \leq r$ be the order of the reduced (or minimal) system. Partitioning $\hat{U} = \begin{bmatrix} \hat{U}_1 & \hat{U}_2 \end{bmatrix}$ such that $\hat{U}_1 \in \mathbb{R}^{s \times \ell}$, $\hat{U}_2 \in \mathbb{R}^{s \times s-\ell}$, $\hat{V}_1 \in \mathbb{R}^{r \times \ell}$, $\hat{V}_2 \in \mathbb{R}^{r \times r-\ell}$, $\hat{\Sigma}_1 = \operatorname{diag}(\sigma_1, \dots, \sigma_\ell)$, and $\hat{\Sigma}_2 = \operatorname{diag}(\sigma_{\ell+1}, \dots, \sigma_r)$, the SVD of $SR^T$ is given by

$$SR^T \ = \ \left[\begin{array}{c|cc} \hat{U}_1 & \hat{U}_2 & 0 \\ \hline 0 & 0 & I_{n-s} \end{array}\right] \left[\begin{array}{c|cc} \hat{\Sigma}_1 & 0 & 0 \\ \hline 0 & \hat{\Sigma}_2 & 0 \\ 0 & 0 & 0 \end{array}\right] \left[\begin{array}{c|cc} \hat{V}_1 & \hat{V}_2 & 0 \\ \hline 0 & 0 & I_{n-s} \end{array}\right]^T. \tag{9}$$

We will see that all the subsequent computations can also be performed just working with $\hat{U}_1$, $\hat{\Sigma}_1$, and $\hat{V}_1$ rather than using the data from the full-size SVD in (9). This amounts in a significant savings of workspace and computational cost. For example, using the Golub–Reinsch SVD (see, e.g., [2]), (7) requires $22n^3$ flops and workspace for $2n^2$ real numbers if $U$, $V$ are to be formed explicitly while (8) only requires $14sr^2 + 8r^3$ flops and workspace for $s^2 + r^2$ 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 $n$. Suppose that (numerically) $s = r = n/10$, then the computation of (8) is 1000 times less expensive than that of (7) and only 1% of the workspace is required for (8) as compared to (7).

Defining

$$T_l = \Sigma_1^{-1/2} V_1^T R = \hat{\Sigma}_1^{-1/2} \hat{V}_1^T \hat{R} \qquad \text{and} \qquad T_r = S^T U_1 \Sigma_1^{-1/2} = \hat{S}^T \hat{U}_1 \hat{\Sigma}_1^{-1/2}, \tag{10}$$

the reduced system (2) is given by

$$\tilde{A} = T_l A T_r, \quad \tilde{B} = T_l B, \quad \tilde{C} = C T_r, \quad \text{and} \quad \tilde{D} = D. \tag{11}$$

In case that $\Sigma_1 > 0$ and $\Sigma_2 = 0$, (11) is a *minimal realization* of the TFM $G(\lambda)$ [8], i.e., $\ell$ is the minimum dimension of the state-space for which a realization of $G(\lambda)$ in the form of an LTI system (1) is possible. Hence, choosing $\ell$ in (7) maximal such that $\sigma_\ell > 0$ and $\sigma_{\ell+1} = 0$, this procedure can be used to compute minimal realizations if the decision "$\sigma_{\ell+1} = 0$" is based on a numerically reliable criterion.

It can further be proved that for a stable LTI system, choosing any partitioning in (7) such that $\sigma_\ell > \sigma_{\ell+1}$ yields a stable, minimal, and balanced reduced model. The Gramians corresponding to the resulting TFM $\tilde{G}(\lambda)$ are both equal to $\Sigma_1$. See [8] for a proof.

As the reduced model in (11) is balanced, the projection matrices in (10) tend to be ill-conditioned if the original system is highly unbalanced, resulting in inaccurate reduced-order models. An alternative here are *balancing-free (BF) methods* [6] for which the reduced-order model is not balanced. The *balancing-free square-root (BFSR) method* combines the best characteristics of the SR and BF approaches [9, 10]. It shares the first two steps (solving the equations in (3) for the Cholesky factors and computing the SVD in (7)) with the SR method described above. Then, two "skinny" QR factorizations are computed,

$$S^T U_1 = \hat{S}^T \hat{U}_1 = P T_S, \qquad R^T V_1 = \hat{R}^T \hat{V}_1 = Q T_R,$$

where $P$, $Q \in \mathbb{R}^{n\times\ell}$ have orthonormal columns and $T_S$, $T_R \in \mathbb{R}^{\ell\times\ell}$ are upper triangular. The reduced system is then given as in (11) with the projection matrices defined by $T_l = (Q^T P)^{-1} Q^T$ and $T_r = P$.

We have implemented only the SR and BFSR algorithms as the BF algorithm described in [6] usually shows no advantage over BFSR algorithms with respect to model reduction abilities. Moreover, the BF approach is potentially numerically unstable. For one, it uses the product $W_c W_o$ rather than $SR^T$, leading to a squaring of the condition number of the matrix product. Second, the projection matrices $T_l$ and $T_r$ computed by the BFSR approach are often significantly better conditioned than those computed by the BF approach [9, 10]. Furthermore, both SR and BFSR algorithms can be efficiently parallelized while the BF method needs a parallelized version of the QR algorithm with re-ordering of eigenvalues. This presents severe implementation difficulties; see, e.g., [1] for a discussion of this topic. Implementation details as well as accuracy and performance details are available in an extended version of this note[2].

### Concluding Remarks

We have described efficient and reliable numerical algorithms for the realization of model reduction methods based on the square-root version of balanced truncation. Using the full-rank factors of the Gramians often enhances the efficiency and accuracy of these methods significantly. Implementations of the discussed methods are based on highly optimized software packages for numerical linear algebra on serial and parallel computers. Parallel computing allows to use these methods for systems of state-space dimension up to order $\mathcal{O}(10^5)$.

## References


[1] P. Benner, E.S. Quintana-Ortí, and G. Quintana-Ortí. Numerical solution of Schur stable linear matrix equations on multicomputers. Berichte aus der Technomathematik, Report 99–13[2], FB3 – Mathematik und Informatik, Universität Bremen, D-28334 Bremen, November 1999.

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

[3] S.J. Hammarling. Numerical solution of the discrete-time, convergent, non-negative definite Lyapunov equation. *Sys. Control Lett.*, 17 (1991), 137–139.

[4] P. Lancaster and M. Tismenetsky. *The Theory of Matrices.* Academic Press, Orlando, 2nd edition, 1985.

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

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

[7] R.A. Smith. Matrix equation $XA + BX = C$. *SIAM J. Appl. Math.*, 16 (1968), 198–201.

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

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

[10] A. Varga. Model reduction routines for SLICOT. NICONET Report 1999–8[3], The Working Group on Software (WGS), June 1999.


[2] Available from `http://www.math.uni-bremen.de/zetem/berichte.html`.
[3] Available from `http://www.win.tue.nl/niconet/NIC2/reports.html`.