SLICOT Working Note 2002-5

New Numerical Software for Model and Controller Reduction[1]

Andras Varga[2]

June 2002

[1]This document presents research results of the European Community BRITE-EURAM III Thematic Networks Programme NICONET (contract number BRRT-CT97-5040) and is distributed by the Working Group on Software WGS. *WGS secretariat*: Mrs. Ida Tassens, ESAT - Katholieke Universiteit Leuven, K. Mercierlaan 94, 3001-Leuven-Heverlee, BELGIUM. This report is also available by anonymous ftp from `wgs.esat.kuleuven.ac.be/pub/WGS/REPORTS/SLWN2002-5.ps.Z`

[2]Deutsches Zentrum für Luft und Raumfahrt, Institut für Robotik und Mechatronik, DLR Oberpfaffenhofen, Postfach 1116, D-82230 Wessling, Germany

Abstract

In this report we describe the recently developed model and controller reduction software for SLICOT within Task II.B of the NICONET Project. A powerful collection of user callable Fortran 77 routines has been implemented based on the latest algorithmic developments which cover the relative error model reduction using the *balanced stochastic truncation* approach, model reduction using *frequency-weighted balancing* and *frequency-weighted Hankel-norm approximation* methods, as well as special controller reduction methods using *frequency-weighted balancing* and *coprime factorization* based techniques. All implemented routines can be employed to reduce both stable and unstable, continuous- or discrete-time models or controllers. The underlying numerical algorithms are based on extensions of the *square-root* and *balancing-free* accuracy enhancing technique developed by the author for balancing-related model reduction. The new model and controller reduction routines for SLICOT are among the most powerful and numerically most reliable software tools available for model and controller reduction. To facilitate their usage, easy-to-use and flexible interfaces have been developed to integrate them in MATLAB and `Scilab`.

Acknowledgments. The author thanks Vasile Sima and Diana Sima for their substantial contributions to software implementations.

Contents

1 Introduction

The design of low order controllers for high order plants is a challenging problem both theoretically as well as from a computational point of view. The advanced controller design methods like the LQG/LTR loop-shaping, H_∞ control design, μ and linear matrix inequalities (LMIs) based synthesis methods produce typically controllers with orders comparable with the order of the plant. Therefore, the orders of these controllers tend often to be too high for practical use, where simple controllers are preferred over complex ones. The main advantages of simpler controllers are their lower computational complexity allowing higher sampling rates in real-time implementation, and an easier maintenance (e.g., easier bug fix). Thus model reduction methods capable to address controller reduction problems are of primary importance to allow the practical applicability of advanced controller design methods for high order systems. Comprehensive presentations of controller reduction methods and the reasons behind different approaches can be found in the textbook [40] and in the recent monograph [19].

Software implementing advanced synthesis procedures is available in both commercial as well as in free CACSD software. For example, freely available high quality numerical software for the LQG/LTR loop-shaping and H_∞ control design approaches has been recently developed within the European Project NICONET[1] and is part of the Fortran 77 control and systems library SLICOT [3]. In contrast, software suitable for controller reduction is scarce and is only available as commercial software [16]. This is why, a systematic effort has been undertaken within the NICONET Project to complement the available design tools with high quality, numerically robust software for model and controller reduction suitable to be employed in obtaining low order controllers.

The result of the concentrated effort within the NICONET project was the development of a powerful collection of Fortran 77 subroutines for model and controller reduction which covers a range of possible approaches for obtaining low order controllers for high order plants. By using this software, there are three possible approaches to determine low order controllers:

1. perform the controller synthesis on low order approximate plant models;
2. perform controller reduction by employing standard model reduction approaches;
3. perform controller reduction by employing special stability/performance enforcing controller reduction approaches.

In the first and second approaches, we can employ standard model reduction methods which try to minimize either the absolute, or relative, or frequency-weighted approximation error for the plant or the controller, respectively. Software for absolute error model reduction methods has been developed previously and was reported in [31, 33]. A common disadvantage of direct controller reduction using the second approach is that, there are no straightforward ways for handling the problem of preserving closed-loop stability and performance when the reduced controller replaces the full order controller. Since the presence of the plant in the control loop is completely ignored, the resulting reduced controllers can lead to unsatisfactory closed-loop performance and even to the loss of closed-loop stability. This is why, in the third approach,

[1] `http://www.win.tue.nl/niconet/niconet.html`

special controller reduction methods can be employed, which are able to address explicitly closed-loop stability and performance preserving issues. These methods try either to minimize a frequency-weighted approximation error for special weights or perform controller reduction on special coprime factorization of the controller.

In this paper we describe the recently developed model and controller reduction software for SLICOT. A powerful collection of user callable Fortran 77 routines has been implemented based on the latest algorithmic developments for the following approaches:

- relative error model reduction using the *balanced stochastic truncation* (BST) approach [4, 21];
- model reduction using frequency-weighted balancing related [6, 35] and frequency-weighted Hankel-norm approximation methods [11, 34];
- controller reduction methods using frequency-weighted balancing related methods [36] and coprime factorization based techniques [14].

All implemented routines can be employed to reduce both stable and unstable, continuous- or discrete-time models or controllers. The underlying numerical algorithms are based on extensions of the *square-root* and *balancing-free* accuracy enhancing technique developed by the author for balancing-related model reduction [25].

The new model and controller reduction routines for SLICOT are among the most powerful and numerically most reliable software tools available for model and controller reduction. To facilitate their usage, easy-to-use and flexible interfaces have been developed to integrate them in two popular user friendly computing environments for engineering and scientific applications: the commercial package MATLAB[2] and the free software `Scilab` [5].

2 Overview of model and controller reduction methods

2.1 Basic model reduction approaches

In this section we discuss shortly the basic model reduction approaches which are potentially applicable to controller reduction as well, and present computational methods suitable to solve corresponding model reduction problems. Consider the n-th order original state-space model $\mathbf{G} := (A, B, C, D)$ with the *transfer-function matrix* (TFM)

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

and let $\mathbf{G}_r := (A_r, B_r, C_r, D_r)$ be an r-th order approximation of the original model ($r < n$), with the TFM

$$G_r(\lambda) = C_r(\lambda I - A_r)^{-1}B_r + D_r.$$

According to the system type, λ is either the complex variable s appearing in the Laplace transform in the case of a continuous-time system or the variable z appearing in the z-transform

[2] MATLAB is a registered trademark of The MathWorks, Inc.

in the case of a discrete-time system. Throughout the paper, we will use the following notational convention. The bold-notation $\mathbf{G}$ is used to denote a state-space system having the TFM $G(\lambda)$ or G. This notation is used consistently to denote systems corresponding to particular TFMs: $\mathbf{G}_1\mathbf{G}_2$ denotes the series coupling of two systems having the TFM $G_1(\lambda)G_2(\lambda)$, $\mathbf{G}_1+\mathbf{G}_2$ represents the (additive) parallel coupling of two systems with TFM $G_1(\lambda)+G_2(\lambda)$, $\mathbf{G}^{-1}$ denotes the inverse system corresponding to the inverse TFM $G^{-1}(\lambda)$, and $\mathbf{G}^{\sim}$ denotes the conjugate system corresponding to the conjugate TFM $G^{\sim}(\lambda)$, where $G^{\sim}(s) = G^T(-s)$ for a continuous-time system and $G^{\sim}(z) = G^T(1/z)$ for a discrete-time system.

The *absolute error* model reduction methods try to minimize the absolute approximation error

$$\|G - G_r\|_{\infty}. \tag{1}$$

For a stable original system $\mathbf{G}$, the *balanced truncation* (BT) [17], the *singular perturbation approximation* (SPA) [13] and the *Hankel-norm approximation* (HNA) [7] are the most frequently employed model reduction approaches. In conjunction with modal separation [29] and coprime factorization [27] techniques, these methods can be employed for the reduction of unstable systems as well. For all these methods, robust numerical software is available in the SLICOT library [33]. Applications of this software to solve model reduction problems with dense matrices up to an order of $n = 5000$ have been reported [15].

The *relative error* model reduction methods have several properties which recommend them for both model and controller reduction. While absolute error methods compute good approximations in terms of peak errors (e.g., H_∞-norm), relative error methods have good approximation properties over the whole frequency range. This is why it is expected that relative error methods in combination with modal separation techniques are better suited for controller approximation than absolute error methods. The *balanced stochastic truncation* (BST) method [4] is a relative error method which tries to minimize $\|\Delta_r\|_\infty$, where Δ_r is the relative error defined implicitly by $G_r = (I - \Delta_r)G$. If $G(\infty)$ is invertible, this is equivalent to minimize

$$\|G^{-1}(G - G_r)\|_{\infty}. \tag{2}$$

For a non-square G, the problem can still be solved if $G(\infty)$ has full row rank (i.e., no zeros at infinity). For a full column rank G, the same problem can be solved for the dual system with the TFM G^T. It is possible to combine the additive and relative approaches by performing the BST method on a modified system with the TFM $[\,G\ \beta I\,]$. A zero value of β leads to a pure relative error minimization, while large positive values of β produce approximations which minimize the absolute approximation error (1). When $\beta \to \infty$, the BST method produces identical results with the BT method. The BST method has been recently extended for arbitrary rank of $G(\infty)$ in [32]. The BST can be employed also in conjunction with the SPA approach [8].

The above mentioned additive and relative error methods have many convenient features which recommend them in solving model and controller reduction problems. All these methods have *a priori* guaranteed approximation error bounds, which can be employed to determine reduced order models satisfying a given approximation error. Moreover, all these methods, when applied to a stable system, produce stable reduced order approximations. In conjunction with modal separation, all these methods preserve the unstable eigenvalues, which is an useful feature when these methods are employed for controller reduction.

The methods for *frequency-weighted model reduction* (FWMR) try to minimize a *weighted approximation* error of the form

$$\|W_o(G - G_r)W_i\|_\infty, \tag{3}$$

where W_o and W_i are suitably chosen output and input weighting TFMs, respectively. The presence of weights reflects the desire that the approximation be more accurate at certain frequencies where W_o and/or W_i have larger singular values. The FWMR approach can be simultaneously interpreted as an extension and a generalization of the absolute and relative error methods. For example, for a square invertible G, by taking $W_o = G^{-1}$ and $W_i = I$, the BST relative error model reduction problem is a particular FWMR problem. Many controller reduction problems can be formulated as FWMR problems [1] (see also next section).

For the solution of the FWMR, a *frequency-weighted BT* (FWBT) approach, extending the BT method, has been proposed in [6], and further extended by various authors [12, 38, 35]. A *frequency-weighted SPA* (FWSPA) method has been discussed in [35] pointing out better approximation properties than for the FWBT, both in terms of smaller errors as well as of stability preserving. As reported in [35], the FWSPA produced stable approximations in some cases when the FWBT approach for two-sided weights failed. The *frequency-weighted HNA* (FWHNA) has been introduced in [11], and extended to the multivariable case in [10]. Recent developments [34] extend this method to arbitrary invertible weights by using descriptor systems based projection computations.

2.2 Basic controller reduction approaches

Controller reduction problems are frequently formulated as special model reduction problems [1]. Let $\mathbf{K} = (A_c, B_c, C_c, D_c)$ be a stabilizing controller of order n_c for a given system $\mathbf{G}$. We want to find $\mathbf{K}_r$, an r_c-th order approximation of $\mathbf{K}$ having the same number of unstable poles as $\mathbf{K}$, such that the reduced controller $\mathbf{K}_r$ is stabilizing and the closed-loop system using the reducing controller has satisfactory performances.

Virtually all model reduction methods in conjunction with modal separation approaches (to preserve the unstable poles) can be employed to perform controller reduction as well. The controller reduction problem is frequently formulated as a FWMR problem [1] to find $\mathbf{K}_r$, an r_c-th order approximation of $\mathbf{K}$ having the same number of unstable poles as $\mathbf{K}$, such that a weighted error of the form

$$\|W_o(K - K_r)W_i\|_\infty, \tag{4}$$

is minimized, where W_o and W_i are suitably chosen weighting TFMs. To enforce closed-loop stability, one-sided weights of the form

$$W_i = I, \quad W_o = (I + GK)^{-1}G \tag{5}$$

or

$$W_i = G(I + KG)^{-1}, \quad W_o = I \tag{6}$$

can be used, while performance-preserving considerations lead to two-sided weights

$$W_o = (I + GK)^{-1}G, \quad W_i = (I + GK)^{-1} \tag{7}$$

Efficient controller reduction methods to solve (4) for the three particular stability and performance enforcing weights defined in (5), (6) and (7) have been recently proposed in [36] by using frequency-weighted balancing related methods. The new method can be seen as an enhancement of the method of [14], where the FWBT approach has been specialized to the case of a *stable* state-feedback and full-order estimator based controller. In contrast to [14], the new method is applicable to an arbitrary stabilizing controller, without any restriction on the stability of the controller itself. The underlying computational procedure (see also the next section) retains the efficiency of the procedure of [14].

Feedback controllers resulting from LQG designs have a special structure which allows the direct use of model reduction techniques on appropriate "natural" coprime factorized representations. Provided the controller has a left coprime factorization

$$K = M^{-1}N,$$

or a right coprime factorization

$$K = NM^{-1},$$

the order reduction can be performed by approximating $[\, M \ N \,]$ or $\left[\begin{array}{c} N \\ M \end{array}\right]$ by some lower order approximations $[\, M_r \ \ N_r \,]$ or $\left[\begin{array}{c} N_r \\ M_r \end{array}\right]$, respectively. This leads to a reduced order controller defined in a left factorized form $K_r = M_r^{-1}N_r$ or right factorized form $K_r = N_r M_r^{-1}$. In the case of a state-feedback and observer-based controller, "natural" factorizations can be used to determine the factors.

Using the above unweighted approach, there is no guarantee for preserving closed-loop stability. This is why, the coprime factorization approach has been combined with frequency weighting in order to enforce the closed-loop stability. Let $K = M^{-1}N$ be a left coprime factorization of the controller. By using $\widetilde{K} = [\, M - I \ N \,]$ as the "controller" and $\widetilde{G} = [\, I \ G^T \,]^T$ as the "plant", a FWMR can be performed along the lines of the formulation based on (6). The controller reduction becomes one of minimizing the weighted error

$$\| [\, M - M_r \ N - N_r \,] \left[\begin{array}{c} Y \\ X \end{array}\right] \|_\infty \tag{8}$$

for M and N, where $G = XY^{-1}$ is a right comprime factorization of the plant (see [14]). Note that a particularly simple expression for the corresponding input weighting $W_i = \left[\begin{array}{c} Y \\ X \end{array}\right]$ is obtained in the case of using full-order observer based controllers (see [14] for further details).

3 Computational methods

3.1 Balancing related approaches

The computational algorithms for several *balancing related* model reduction methods as the BT, BST or FWBT, can be interpreted as performing a similarity transformation Z on a *stable* original system $\mathbf{G} = (A, B, C, D)$ yielding

$$\left[\begin{array}{c|c} Z^{-1}AZ & Z^{-1}B \\ \hline CZ & D \end{array}\right] := \left[\begin{array}{cc|c} A_{11} & A_{12} & B_1 \\ A_{21} & A_{22} & B_2 \\ \hline C_1 & C_2 & D \end{array}\right], \tag{9}$$

and then defining the reduced model $\mathbf{G}_r = (A_r, B_r, C_r, D_r)$ of order r as

$$\mathbf{G}_r = (A_{11},\, B_1,\, C_1,\, D).$$

When writing

$$Z := \left[\begin{array}{cc} T & U \end{array}\right], \quad Z^{-1} := \left[\begin{array}{c} L \\ V \end{array}\right], \tag{10}$$

then $\Pi = TL$ is an oblique projector on T along L, and $LT = I_r$. The reduced system is given by

$$\mathbf{G}_r = (LAT,\, LB,\, CT,\, D). \tag{11}$$

The matrices L and T are called *truncation matrices*. To compute the matrices of the reduced model $\mathbf{G}_r$ only these truncation matrices are necessary to be determined.

In the case of the SPA, BST-SPA and FWSPA methods, the truncation matrices L and T serve primarily to determine a minimal realization of the original system $\mathbf{G}$ in a special, so-called "balanced" coordinate form. The matrices of the minimal order "balanced" system can be then partitioned as in (9), and used to construct a SPA $\mathbf{G}_r$ given by

$$\begin{array}{rcl} A_r &=& A_{11} + A_{12}(\gamma I - A_{22})^{-1}A_{21}, \\ B_r &=& B_1 + A_{12}(\gamma I - A_{22})^{-1}B_2, \\ C_r &=& C_1 + C_2(\gamma I - A_{22})^{-1}A_{21}, \\ D_r &=& D + C_2(\gamma I - A_{22})^{-1}B_2. \end{array} \tag{12}$$

where $\gamma = 0$ for a continuous-time system and $\gamma = 1$ for a discrete-time system. Note that SPA formulas preserve the DC-gain of an original stable system.

The truncation matrices are computed from two positive semi-definite matrices P and Q, called generically the "controllability" and "observability" gramians, respectively. These gramians can always be determined in Cholesky factorized forms $P = SS^T$ and $Q = R^TR$, where S and R are upper-triangular matrices. The computation of the truncation matrices L and T can be done from the *singular value decomposition* (SVD)

$$RS = \left[\begin{array}{cc} U_1 & U_2 \end{array}\right] \mathrm{diag}(\Sigma_1, \Sigma_2) \left[\begin{array}{cc} V_1 & V_2 \end{array}\right]^T,$$

where

$$\Sigma_1 = \mathrm{diag}(\sigma_1, \ldots, \sigma_r), \quad \Sigma_2 = \mathrm{diag}(\sigma_{r+1}, \ldots, \sigma_n),$$

and $\sigma_1 \geq \ldots \geq \sigma_r > \sigma_{r+1} \geq \ldots \geq \sigma_n \geq 0$. The quantities $\sigma_1, \ldots, \sigma_n$ are called the *Hankel-singular values* and can be employed to determine a suitable order of the approximation $\mathbf{G_r}$. To compute L and T several formulas can be employed. For example, choosing

$$L = \Sigma_1^{-1/2} U_1^T R, \qquad T = SV_1\Sigma_1^{-1/2}. \tag{13}$$

as proposed in [24], leads to the so-called *square-root* (**SR**) BT, BST or FWBT methods. This naming suggests that all computations are done in terms of square-root quantities (i.e., the Cholesky factors of gramians).

If the truncation matrices are just used to compute a minimal realization (i.e., $\Sigma_2 = 0$) as for example in the case of using SPA formulas, then the reduced model is called a "balanced" minimal realization of the original system. The term "balanced" has the following precise meaning: the gramians corresponding to a "balanced" realization are both equal to the diagonal matrix Σ_1.

The **SR** approach is usually very accurate for well-equilibrated systems. However if the original system is highly unbalanced, potential accuracy losses can be induced in the reduced model if either L or T is ill-conditioned (i.e., nearly rank deficient). In order to avoid ill-conditioned truncation matrices, a *balancing-free* (**BF**) approach has been proposed in [22] in which always well-conditioned truncation matrices L and T can be determined. These matrices are computed from orthogonal matrices whose columns span orthogonal bases for the right and left eigenspaces of the product PQ corresponding to the first r largest eigenvalues $\sigma_1^2, \ldots, \sigma_r^2$. Because of the need to compute explicitly P and Q as well as their product, this approach is usually less accurate for moderately ill-balanced systems than the **SR** approach.

A *balancing-free square-root* (**BFSR**) algorithm which combines the advantages of the **BF** and **SR** approaches has been introduced in [26]. L and T are determined as

$$L = (Y^T X)^{-1} Y^T, \qquad T = X,$$

where X and Y are $n \times r$ matrices with orthogonal columns computed from two QR decompositions

$$S^T U_1 = XW, \quad R^T V_1 = YZ$$

with W and Z non-singular and upper-triangular. The accuracy of the **BFSR** algorithm is usually better than either of **SR** or **BF** approaches.

The BT, BST and FWBT methods as well as their SPA variants differ merely by the choice of the two gramians.

3.1.1 Unweighted BT and SPA methods

In the unweighted BT and SPA methods [17, 13], P and Q are the controllability and observability gramians of $\mathbf{G}$ satisfying a pair of continuous-time (c) or discrete-time (d) Lyapunov equations

$$\left\{ \begin{array}{rcl} AP + PA^T + BB^T & = & 0 \\ A^TQ + QA + C^TC & = & 0 \end{array} \right. \quad (c)\ , \quad \left\{ \begin{array}{rcl} APA^T + BB^T & = & P \\ A^TQA + C^TC & = & Q \end{array} \right. \quad (d)\ . \tag{14}$$

Since the gramians are positive semidefinite matrices for a stable system, the stability of the reduced model easily follows from the guaranteed positive semidefiniteness of the gramians of the reduced system. These Lyapunov equations with non-negative definite solutions can be solved directly for the Cholesky factors of the gramians, by using the methods proposed in [9].

3.1.2 BST and BST-SPA methods

The BST method for a stable original system $\mathbf{G}$ uses P as the controllability gramian of $\mathbf{G}$ and Q as the observability gramian of a minimum-phase spectral factor $\mathbf{W}$ satisfying

$$W^{\sim}(\lambda)W(\lambda) = [\, G(\lambda)\ \beta I \,] \left[\begin{array}{c} G^{\sim}(\lambda) \\ \beta I \end{array} \right] \tag{15}$$

In the continuous-time case, a state space realization $\mathbf{W} = (A, B_W, C_W, D_W)$ of $W(s)$ can be obtained with matrices defined as (see e.g., [40])

$$\begin{array}{rcl} B_W & = & PC^T + BD^T \\ C_W & = & \widetilde{D}^{-1}(C - B_W^T Q) \\ D_W & = & \widetilde{D}^T, \end{array}$$

where $\widetilde{D}$ satisfies $\widetilde{D}\widetilde{D}^T = DD^T + \beta^2 I$, and Q is the observability gramian of $\mathbf{W}$, being the solution of the Lyapunov equation

$$QA + A^T Q + C_W^T C_W = 0. \tag{16}$$

Since C_W depends linearly of Q, this is in fact a standard Riccati equation which need to be solved in general.

In the case of discrete-time systems, the bilinear transformation technique can serve as basis for a discrete-time BST algorithm. We apply the continuous-time BST algorithm to the continuous-time system $G_c(s) := G(\frac{s+1}{s-1})$ resulted by a bilinear transformation. If $G_{c,r}$ is the computed continuous-time BST approximation for G_c, then the discrete-time BST approximation is obtained by the inverse bilinear transformation $G_r(z) = G_{c,r}(\frac{z+1}{z-1})$. A possible disadvantage of this approach is that the resulting reduced system G_r has possibly a nonzero D_r matrix even if the original system is strictly proper.

The following general procedure combines the BST method with modal separation and served as basis for the implementation of the new SLCIOT routine **AB09HD**:

BST Procedure.

1. Compute an additive stable-unstable modal decomposition of $\mathbf{G}$ as

$$\mathbf{G} = \mathbf{G}_s + \mathbf{G}_u$$

   where $\mathbf{G}_s$, of order n_s, contains the stable poles of $\mathbf{G}$ and $\mathbf{G}_u$, of order $n - n_s$, contains the unstable poles of $\mathbf{G}$.

2. Compute the controllability gramian P of $\mathbf{G}_s$ and the observability gramian Q of the spectral factor $\mathbf{W}$ in (15) with $\mathbf{G}_s$ replacing $\mathbf{G}$.

3. Using P and Q in place of standard gramians of $\mathbf{G}_s$, determine a reduced order approximation $\mathbf{G}_{sr}$ by applying the BT or SPA method.

4. Compute

$$\mathbf{G}_r = \mathbf{G}_{sr} + \mathbf{G}_u$$

An enhancement of this procedure applicable with $\beta = 0$ even in the case of rank deficient D has been proposed in [32].

3.1.3 FWBT and FWSPA methods

The FWBT and FWSPA methods rely on special choices of gramians, by extending the unweighted BT and SPA methods for stable systems to the frequency-weighted case. A first choice for the so-called *frequency-weighted gramians* for the FWBT approach has been proposed by Enns in [6]. Let assume that G and the two weights W_o and W_i are all stable TFMs, and let $\mathbf{W_o} = (A_o, B_o, C_o, D_o)$ and $\mathbf{W_i} = (A_i, B_i, C_i, D_i)$ be minimal state-space realizations of the weighting matrices. Consider the following realizations of $\mathbf{GW_i}$ and $\mathbf{W_oG}$:

$$\mathbf{GW_i} = \left[\begin{array}{c|c} \overline{A}_i & \overline{B}_i \\ \hline \overline{C}_i & \overline{D}_i \end{array}\right] =: \left[\begin{array}{cc|c} A & BC_i & BD_i \\ 0 & A_i & B_i \\ \hline C & DC_i & DD_i \end{array}\right], \tag{17}$$

$$\mathbf{W_oG} = \left[\begin{array}{c|c} \overline{A}_o & \overline{B}_o \\ \hline \overline{C}_o & \overline{D}_o \end{array}\right] =: \left[\begin{array}{cc|c} A_o & B_oC & B_oD \\ 0 & A & B \\ \hline C_o & D_oC & D_oD \end{array}\right] \tag{18}$$

Let

$$\overline{P} = \left[\begin{array}{cc} P_{11} & P_{12} \\ P_{12}^T & P_{22} \end{array}\right], \quad \overline{Q} = \left[\begin{array}{cc} Q_{11} & Q_{12} \\ Q_{12}^T & Q_{22} \end{array}\right], \tag{19}$$

be the controllability gramian of $\mathbf{GW_i}$ and the observability gramian of $\mathbf{W_oG}$, respectively, partitioned such that P_{11} and Q_{22} are $n \times n$ matrices. The approach proposed by Enns defines

$$P_E = P_{11}, \quad Q_E = Q_{22} \tag{20}$$

as the frequency-weighted controllability and observability gramians, respectively. Although this method has been successfully employed in many applications, it lacks an easily computable *a priori* error bound and the stability of the reduced model is not guaranteed in the case of two-sided weighting, unless either $W_o = I$ or $W_i = I$. Occasionally, quite poor approximations result even for one-sided weighting (see [1, 38] for supplementary details on this approach).

The second approach was proposed in [12, 23] and, under certain circumstances, guarantees stability in the case of two-sided weighting. Provided P_{22} and Q_{11} are nonsingular (a condition ensured, for example, if the realizations (17) of $\mathbf{GW_i}$ and (18) of $\mathbf{W_oG}$ are minimal), the two gramians are chosen as

$$P_L = P_{11} - P_{12}P_{22}^{-1}P_{12}^T, \quad Q_L = Q_{22} - Q_{21}Q_{11}^{-1}Q_{21}^T. \tag{21}$$

Since P_L and Q_L satisfy this time Lyapunov equations of the form (14) (however with different B and C matrices), the stability of the reduced model is automatically guaranteed [20]. As with the Enns' method, no easily computable *a priori* error bound exists for this approach. The main weakness of this approach is the requirement that no pole-zero cancellations occur when forming $\mathbf{GW_i}$ or $\mathbf{W_oG}$. For example, this restriction prevents the applicability of this method to solve controller reduction problems involving the special stability/performance enforcing weights (5)–(7) (see next section), where pole-zero cancellations take always place.

Another modification of the method of Enns which guarantees stability in the case of two-sided weights has been proposed recently by Wang *et al.* [38] for continuous-time systems. With some preliminary assumptions, this approach allows the computation of an *a priori* error bound for the weighted approximation error, which in some cases appears to be helpful in guiding the choice of the order of the reduced order model. Two frequency-weighted gramians P_W and Q_W are determined in [38] as the solutions of the pair of Lyapunov equations

$$\begin{array}{rcl} AP_W + P_W A^T + \widetilde{B}\,\widetilde{B}^T & = & 0 \\ Q_W A + A^T Q_W + \widetilde{C}^T \widetilde{C} & = & 0 \end{array} \tag{22}$$

where $\widetilde{B}$ and $\widetilde{C}$ are fictitious input and output matrices. These matrices are computed from the orthogonal eigendecompositions of the symmetric matrices $X = -AP_E - P_E A^T$ and $Y = -A^T Q_E - Q_E A$:

$$X = U\Theta U^T, \quad Y = V\Gamma V^T, \tag{23}$$

where Θ and Γ are real diagonal matrices. Specifically, according to [38], $\widetilde{B}$ and $\widetilde{C}$ are determined as

$$\widetilde{B} = U|\Theta|^{\frac{1}{2}}, \quad \widetilde{C} = |\Gamma|^{\frac{1}{2}} V^T$$

where $|\cdot|$ denotes a matrix formed from the absolute values of its elements. Provided $\operatorname{rank} [\, \widetilde{B} \; B \,] = \operatorname{rank} \widetilde{B}$ and $\operatorname{rank} [\, \widetilde{C}^T \; C^T \,] = \operatorname{rank} \widetilde{C}^T$ it is also proven in [38] that

$$\|W_o(G - G_r)W_i\|_\infty \leq k \operatorname{tr} \Sigma_2, \tag{24}$$

where $k = 2\|W_o L\|_\infty \|KW_i\|_\infty$ with $L = C\widetilde{C}^{\#}$ and $K = \widetilde{B}^{\#} B$ ($X^{\#}$ denotes the pseudo-inverse of X). It is easy to see that with this choice of gramians we have $P_W - P_E =: \Delta P_W \geq 0$ and $Q_W - Q_E =: \Delta Q_W \geq 0$. Thus, the system $(A, \widetilde{B}, \widetilde{C})$ is minimal provided the original system is minimal.

Recently, new enhancements have been reported [35] to cope with the above deficiencies. For example, it is possible to construct frequency-weighted gramians which lead to a combination of the approach of [12, 23] with Enns' method [6]. In such a *combination method* we can use as frequency-weighted gramians

$$P_{EL} = P_{11} - \alpha_c^2 P_{12} P_{22}^{-1} P_{12}^T, \quad Q_{EL} = Q_{22} - \alpha_o^2 Q_{21} Q_{11}^{-1} Q_{21}^T, \tag{25}$$

where for $\alpha_c = \alpha_o = 0$ we have the choice for Enns' method, while for $\alpha_c = \alpha_o = 1$ we have the choice for the method of [12] with stability guarantee. Because stability is guaranteed for $\alpha_c = \alpha_o = 1$, it is to be expected this to be also true for nearby subunitary values of α_c and α_o (on the basis of a continuous variation of spectrum with α_c and α_o). Thus, stability will be guaranteed in a whole neighborhood of $\alpha_c = \alpha_o = 1$ regardless of whether pole-zero cancellations occur or not. This feature can be seen as a simultaneous enhancement of each of the methods in [6, 12].

Concerning enhancements of the approach of [38], it appears that in some cases the sizes of ΔP_W and ΔQ_W can be reduced by another choice of $\widetilde{B}$ and $\widetilde{C}$. Consider $\Theta = \operatorname{diag}(\Theta_1, \Theta_2)$ and $\Gamma = \operatorname{diag}(\Gamma_1, \Gamma_2)$ in the decompositions of X and Y in (23) partitioned such that $\Theta_1 > 0$ and $\Theta_2 \leq 0$, $\Gamma_1 > 0$ and $\Gamma_2 \leq 0$. Partition U and V in accordance with the partitioning of Θ and Γ, respectively, and define $\widetilde{B}$ and $\widetilde{C}$ as

$$\widetilde{B} = U_1 \Theta_1^{\frac{1}{2}}, \quad \widetilde{C} = \Gamma_1^{\frac{1}{2}} V_1^T. \tag{26}$$

With this choice, provided similar rank conditions are fulfilled as for [38], an error bound as in (24) holds with a smaller k (L and K have smaller norms) and a possibly smaller norm Σ_2. Thus, this modification of the method of [38] appears to be useful in some applications.

It is possible now to consider a *modified combination method*, where we include all above modifications in a single parameterized approach, which guarantees the stability of reduced models for two-sided weights. This can be done by defining new fictitious input and output matrices $\widehat{B}$ and $\widehat{C}$, respectively, such that the frequency-weighted gramians P_V and Q_V satisfy

$$\left\{\begin{array}{lcl} AP_V + P_V A^T + \widehat{B}\,\widehat{B}^T & = & 0 \\ Q_V A + A^T Q_V + \widehat{C}^T \widehat{C} & = & 0 \end{array}\right. \quad (c)\ , \quad \left\{\begin{array}{lcl} AP_V A^T + \widehat{B}\widehat{B}^T & = & P_V \\ A^T Q_V A + \widehat{C}^T \widehat{C} & = & Q_V \end{array}\right. \quad (d) \qquad (27)$$

$\widehat{B}$ and $\widehat{C}$ are defined just like $\widetilde{B}$ and $\widetilde{C}$ in (26), using the decompositions (23) but with

$$\left\{\begin{array}{lcl} X & = & -AP_{EL} - P_{EL}A^T \\ Y & = & -A^T Q_{EL} - Q_{EL} A \end{array}\right. \quad (c)\ , \quad \left\{\begin{array}{lcl} X & = & -AP_{EL}A^T + P_{EL} \\ Y & = & -A^T Q_{EL} A + Q_{EL} \end{array}\right. \quad (d)$$

Here, P_{EL} and Q_{EL} are the frequency-weighted gramians for the combination method defined in (25). Note that any combination of gramians (P_{EL}, Q_V), (P_V, Q_{EL}), or (P_V, Q_V), for all sub-unitary values of parameters α_c and α_o guarantees the stability of approximations for two-sided weighting. Besides guaranteeing stability of the reduced order model in the case of two-sided weighting, the new choices for frequency-weighted controllability and observability gramians enlarge substantially the applicability of frequency-weighted balancing related model reduction methods.

Another enhancement of existing methods is to use the SPA instead of BT. Although the use of the SPA method in conjunction with frequency-weighted balancing is apparently not even mentioned in the literature, surprisingly this approach provides, almost always, better approximations than the BT method, both in terms of preserving stability as well as leading to lower approximation errors.

The following general procedure combines the FWBT and FWSPA methods with modal separation and served as basis for implementation of the SLICOT routine **AB09ID**:

FWMR Procedure.

1. Compute an additive stable-unstable modal decomposition of $\mathbf{G}$ as
$$\mathbf{G} = \mathbf{G}_s + \mathbf{G}_u$$
where $\mathbf{G}_s$, of order n_s, contains the stable poles of $\mathbf{G}$ and $\mathbf{G}_u$, of order $n - n_s$, contains the unstable poles of $\mathbf{G}$.
2. Compute the controllability gramian of $\mathbf{G}_s\mathbf{W}_i$ and the observability gramian of $\mathbf{W}_o\mathbf{G}_s$ and define as the pair (P, Q) of controllability and observability gramians, any appropriate n_s order frequency-weighted controllability and observability gramians from the pairs (P_{EL}, Q_V), (P_V, Q_{EL}), or (P_V, Q_V).
3. Using P and Q in place of standard gramians of $\mathbf{G}_s$, determine a reduced order approximation $\mathbf{G}_{sr}$ by applying the BT or SPA method.
4. Compute
$$\mathbf{G}_r = \mathbf{G}_{sr} + \mathbf{G}_u$$

In the case of one-sided weights (i.e., either $W_o = I$ or $W_i = I$), one of gramians is the standard observability or controllability gramian, respectively. The second gramian can be chosen as proposed by Enns P_E or Q_E, respectively. In the unweighted case (i.e., $W_o = I$ and $W_i = I$), the **FWMR Procedure** is simply the BT or SPA applied to the stable projection of $\mathbf{G}$.

One important aspect for robust numerical implementation of the FWMR Procedure is the computation of the Cholesky factors of the "gramians". Efficient computational methods relying on updating Cholesky factorizations have been proposed in [35] to compute the Cholesky factors of the leading or trailing blocks, having the Cholesky factors of full gramians.

Some remarks are appropriate here on how to handle the frequency-weighted reduction in the case of unstable weights. In this case, we assume that $\mathbf{W_o}$ and $\mathbf{W_i}$ have no poles on the imaginary axis in continuous-time or on the unit circle in discrete-time. Then we can compute the left and right coprime factorizations

$$W_o = M_o^{-1}\widetilde{W}_o, \quad W_i = \widetilde{W}_i M_i^{-1} ,$$

where $\widetilde{W}_o$, M_o, $\widetilde{W}_i$, and M_i are stable TFMs with M_o and M_o inner (i.e., $M_o^{\sim} M_o = I$ and $M_i^{\sim} M_i = I$). It follows that

$$\|W_o(G - G_r)W_i\|_\infty = \|\widetilde{W}_o(G - G_r)\widetilde{W}_i\|_\infty ,$$

thus the problem has been reduced to the case with stable weights and can be solved by employing the FWMR Procedure. This technique is implemented in the routine **AB09ID**.

3.2 Hankel-norm approximation

The *Hankel-norm approximation* (HNA) method [7] belongs to the class of absolute (or additive) error model reduction methods and relies on a guaranteed error bound. Glover [7] has shown that for a stable $\mathbf{G}$, there exists an r-th order stable approximation $\mathbf{G}_r$ such that

$$\|G - G_r\|_H = \sigma_{r+1}(\mathbf{G}) \tag{28}$$

where $\sigma_{r+1}(\mathbf{G})$ is the $(r + 1)$-th largest Hankel singular value of $\mathbf{G}$. Note that, because the Hankel-norm is only a seminorm, the choice of D_r in (28) plays no role on the achieved optimal Hankel-norm of the approximation error.

The *frequency-weighted* HNA (FWHNA) problem has been originally formulated in [11] to minimize the weighted-error

$$\|W_o^{\sim}(G - G_r)W_i^{\sim}\|_H \tag{29}$$

where $\mathbf{W}_o$ and $\mathbf{W}_i$ are systems whose TFMs W_o and W_i represent suitable output and input weighting, respectively. The standard assumptions in [11] are: W_o and W_i are biproper, stable and minimum-phase TFMs. A solution of the FWHNA problem for scalar systems has been proposed by Latham and Anderson in [11] and extended to the multivariable case by Hung and Glover in [10].

An alternative formulation of the FWHNA problem is to minimize

$$\|W_o(G - G_r)W_i\|_H \tag{30}$$

for which the corresponding standard assumptions become: W_o and W_i are biproper TFMs, having only unstable poles and zeros. In this case, the optimal frequency-weighted approximation error satisfies [11, 10]

$$\|W_o(G - G_r)W_i\|_H = \sigma_{r+1}(\mathbf{G}_1), \tag{31}$$

where $\mathbf{G}_1 := [\mathbf{W}_o\mathbf{G}\mathbf{W}_i]_+$ is the stable projection of the system $\mathbf{W}_o\mathbf{G}\mathbf{W}_i$.

A numerically reliable computational approach to solve the FWHNA problem with antistable and invertible proper weights W_o and W_i is implemented in the SLICOT subroutine **AB09JD**. The following general procedure is applicable to both stable and unstable systems:

FWHNA Procedure

1. Compute an additive stable-unstable modal decomposition of $\mathbf{G}$ as

$$\mathbf{G} = \mathbf{G}_s + \mathbf{G}_u$$

   where $\mathbf{G}_s$, of order n_s, contains the stable poles of $\mathbf{G}$ and $\mathbf{G}_u$, of order $n - n_s$, contains the unstable poles of $\mathbf{G}$

2. Compute the n_s-th order stable projection

$$\mathbf{G}_{1s} = [\mathbf{W}_o\mathbf{G}_s\mathbf{W}_i]_+$$

3. Compute $\mathbf{G}_{1sr}$, the optimal reduced order HNA of $\mathbf{G}_{1s}$.
4. Compute $\mathbf{G}_{sr}$, the stable projection of $\mathbf{W}_o^{-1}\mathbf{G}_{1sr}\mathbf{W}_i^{-1}$ containing the poles of G_{1sr}.
5. Compute

$$\mathbf{G}_r = \mathbf{G}_{sr} + \mathbf{G}_u$$

This procedure is applicable for W_o and W_i having arbitrary zeros, provided the finite zeros of W_o and W_i are distinct from the poles of G_{1r}, the optimal HNA computed at step 3 of the above procedure. With obvious replacements, the same procedure can be employed to solve the FWHNA problem (29).

For an efficient implementation of the **FWHNA Procedure**, new projection formulas based on descriptor system descriptions have been derived in [36]. These formulas allow an inversion-free implementation of the **FWHNA Procedure**, for both formulations (29) and (30) of the FWHNA problem. The proposed computational solution of the FWHNA problem represents a general numerically reliable alternative to the procedure proposed in [39]. Interestingly, this approach can be employed even if the weights are improper, a possibility also mentioned in [39].

The main usage of the FWHNA method is in solving the frequency-weighted L_∞-approximation problems, where both the original system as well as the weights can be arbitrary, and even non-square (see [36] for details). The FWHNA can be used to produce good approximation errors in the L_∞-norm, which satisfy

$$\sigma_{r+1}(\mathbf{G}_1) \leq \|W_o(G - G_r)W_i\|_\infty$$

The best feedthrough matrix D_r of the reduced model can be determined by a convex optimization [39]. For the computation of an optimal D_r and possibly C_r or B_r, recently developed fast algorithms can be employed [37].

For the L_∞-norm approximation, we only need to assume that the weights have no poles on the imaginary axis in continuous-time or on the unit circle in discrete-time. In the case when W_o and/or W_i have stable poles, then left/right coprime factorizations with *antistable* all-pass denominators can be determined for W_o and W_i using the methods proposed in [30, 18]. If $W_o = M_o^{-1}N_o$ and $W_i = N_iM_i^{-1}$ are the respective factorizations with N_o and N_i antistable, then

$$\|W_o(G - G_r)W_i\|_\infty = \|N_o(G - G_r)N_i\|_\infty$$

Addressing non-square weights and placing unstable zeros via antistable inner-outer factorizations is discussed in [36].

3.3 Controller reduction methods

3.3.1 Frequency-weighted balancing approach

The balancing related **FWMR Procedure** can be immediately employed to solve the problem (4) to determine reduced order controllers. Note that this procedure applied to solve the controller reduction problem (4) ensures automatically that the resulting reduced order controller $\mathbf{K}_r$ has exactly the same number of unstable poles as the original one $\mathbf{K}$. The main computational burden in this procedure is the computation of the two gramians at Step 2. Typically, controller synthesis methods based on the LQG- or H_∞-design methodologies lead to a controller of order $n_c = n$. Thus apparently, for such controllers the computation of gramians involves the solutions of one or two $3n$ order Lyapunov equations. By exploiting the problem structure, it is shown in [14] that for a *stable* state-feedback and full-order estimator based controller, it is possible to solve Lyapunov equations of order only $2n$. The results of [14] have been recently extended in [36] to the general case of a possibly unstable controller. In the cited reference it is shown that the gramians can be determined by solving Lyapunov equations of order at most $n + n_c$. The resulting computational algorithm representing a specialization of the **FWMR Procedure** for controller reduction with the special stability and performance preserving gains (5), (6) and (7) has been implemented in the SLICOT subroutine **SB16AD**.

An important aspect is the direct computation of the Cholesky factors of the frequency-weighted gramians. This is a prerequisite for the applicability of the square-root and balancing-free accuracy-enhancing techniques to controller reduction. Details on how to compute these factors for both general as well as state feedback and observer-based controllers is discussed in [36].

3.3.2 Coprime factorization approach

The coprime factorization based approach for controller reduction involves the computation of gramians for the compound systems $[\,N\ M\,]$ or $\begin{bmatrix} N \\ M \end{bmatrix}$. Let F and L be the state-feedback gain and the Kalman-gain of a full order state estimator, respectively. The overall feedback controller is given by

$$K(\lambda) = F(\lambda I - A - BF - LC - LDF)^{-1}L.$$

This form of the controller allows to write immediately a left or right coprime factorization representation of the controller. A "natural" left coprime representation as

$$K = M^{-1}N,$$

results from

$$[\, N \;\; M \,] = \left[\begin{array}{c|cc} A+LC & L & B+LD \\ \hline F & O & I \end{array} \right] \tag{32}$$

and a "natural" right coprime factorization as

$$K = NM^{-1},$$

results from

$$\begin{bmatrix} N \\ M \end{bmatrix} = \left[\begin{array}{c|c} A+BF & L \\ \hline F & O \\ C+DF & I \end{array} \right] \tag{33}$$

Thus, the gramians of the compound systems can be computed using the corresponding realizations of the factors in (32) or (33). This technique has been implemented in the SLICOT subroutine **SB16BD** for the reduction of state-feedback and observer-based controller in conjunction with the BT and SPA methods.

The weighted coprime factorization approach for state-feedback and observer-based controllers is particularly simple. In the case when using the left coprime factorization (32) of the controller, the gramians used for computing the truncation matrices to solve the FWMR problem (8) can be computed as

$$\begin{array}{ll} \left\{ \begin{array}{rcl} (A+BF)P + P(A+BF)^T + BB^T & = & 0 \\ (A+LC)^TQ + Q(A+LC) + F^TF & = & 0 \end{array} \right. & (c) \\[2ex] \left\{ \begin{array}{rcl} (A+BF)P(A+BF)^T + BB^T & = & P \\ (A+LC)^TQ(A+LC) + F^TF & = & Q \end{array} \right. & (d) \end{array} \tag{34}$$

Similar formulas are used for a right coprime factorization (33) of the controller. These gramians can be used to determine directly the truncation matrices to obtain the matrices of the reduced controller similarly to the truncation formulas (11) used for model reduction. The frequency-weighted coprime factorization approach is implemented in the SLICOT subroutine **SB16CD**.

4 New model and controller reduction routines in SLICOT

4.1 User-callable routines

The basis for implementation of the new model and controller reduction routines in SLICOT was formed partly of the collection of model reduction routines for absolute error methods (BT, SPA, HNA) already available in SLICOT (see [33]) as well as some routines (BST, FWHNA) of the RASP-MODRED library [28]. All implementations rely on the standard linear algebra package LAPACK [2]. Some of the new model reduction routines can be seen as generalizations of the functionality of existing routines in SLICOT implemented within Task II.A. The new SLICOT routines which originate from the RASP-MODRED library have been completely rewritten and partly new algorithmic approaches have been employed. Several routines, including all controller reduction routines, represent completely new implementations and use recently developed algorithmic enhancements (see previous section).

The following table contains the complete list of the new user callable model and controller reduction routines available in SLICOT:

Name	Function
AB09HD	balanced stochastic truncation (BST and BST-SPA approaches)
AB09ID	frequency-weighted balancing related reductions (FWBT and FWSPA approaches)
AB09JD	frequency-weighted Hankel-norm approximation method with invertible proper weights
SB16AD	controller reduction using frequency-weighted balancing related reductions (FWBT and FWSPA approaches) for three closed-loop performance preserving problems defined by appropriate frequency weights
SB16BD	state-feedback/full-order estimator based controller reduction using coprime factorization in conjunction with BT and SPA techniques
SB16CD	state-feedback/full-order estimator based controller reduction using frequency-weighted coprime factorization in conjunction with BT technique

The routines for balancing related model and controller reduction methods rely on the well-established **SR** and **BFSR** accuracy enhancing techniques [25]. The implementation of the FWHNA method uses the **SR** BT method to compute a balanced minimal realization of the system to be reduced.

The user callable model reduction routines AB09HD, AB09ID, and AB09JD implement three basic model reduction algorithms for BST, FWBT/FWSPA and FWHNA approaches, respectively. All these routines perform optionally the scaling of the original system. The order of the reduced system can be selected by the user or can be determined automatically on the basis of the computed Hankel singular values. Each of routines can handle both continuous- and discrete-time systems. Unstable models are handled by separating the stable and unstable parts and applying the model reduction only to the stable parts. All routines for frequency-weighted model reduction approaches can address weighted problems with one-sided or two-sided weights as well as unweighted problems. The new routines provides powerful tools for model and controller reduction and they form the basis to implement the interface software to user-friendly environments (see next section). In what follows, we comment shortly on special functionality provided by various routines.

The model reduction routines AB09HD, AB09ID and AB09JD implement the additive modal separation approach in combination with the employed basic model reduction methods. They provide an additional flexibility by allowing to specify an arbitrary stability boundary inside the standard stability regions (continuous or discrete). The dominant part of the system having poles only in the "unstable" region is retained in the reduced model, and only the "stable" part is approximated. This leads to an effective combination of these methods with the modal reduction approach (see also [29]). It is important to emphasize that the model reduction routines for unstable systems can be applied with practically no efficiency loss to reduce stable systems too.

The routine AB09HD implements the relative-error BST approach in conjunction with BT and SPA techniques. Furthermore, the user can select via a parameter β, the absolute/relative weighting of the approximation error. A positive value of β is always necessary, when the direct feedthrough matrix D is rank deficient. A large positive value of β favours the minimization of the absolute approximation error, while a small value of β is appropriate for the minimization of the relative error. For implementing the discrete-time BST method, bilinear continuous-to-discrete transformation techniques have been employed.

The new frequency-weighted balancing related model reduction routine AB09ID, implementing both the FWBT and FWSPA approaches, practically covers the functionality of AB09AD, AB09BD, AB09MD and AB09ND routines already existing in SLICOT (see [33]). The AB09ID routine is completely general, allowing to handle unstable weights by solving a transformed approximation problem with the original weights replaced by the denominators of appropriate left and right coprime factorizations with inner denominators (see subsection 3.1.3). This routine has a large flexibility in combining different choices of the gramians, as for example Enn's choice, Lin's choice, combination of these choices as well as enhanced stability guaranteing choices. These choices can be independently selected for each gramian.

The new frequency-weighted HNA routine AB09JD covers the functionality of AB09CD and AB09ED routines. AB09JD allows for invertible proper weights, which must satisfy appropriate stability/antistability conditions. To be employed to solve a frequency-weighted L_∞-approximation problem, some preprocessing of weights must be performed before calling this routine. Note that, AB09JD supersedes AB09KD, a preliminary implementation based on explicit inversion formulas (applicable only to biproper weights). AB09JD is very flexible in allowing to handle arbitrary combinations of four types of invertible input and output weights: standard, inverse, conjugated, conjugated inverse. Furthermore, AB09JD allows the user to choose between using projection formulas involving explicit inverses (if they exist) or employing the computationally more expensive (but numerically more reliable) inversion-free setting. This latter option is always used when the feedtrough matrices of the weights are exactly or nearly singular. For implementing the discrete-time FWHNA method, bilinear continuous-to-discrete transformation techniques have been employed.

The controller reduction routine SB16AD is practically a specialization of AB09ID to the case of controller reduction with the special weights (5)-(7) used to enforce closed-loop stability and performance when using the reduced controller instead of the full order one. This routine works on a general controller. In contrast, the coprime factorization based controller reduction routines SB16BD and SB16CD are specially adapted to reduce state feedback and observer-based controllers. The routine AB09BD allows arbitrary combinations of BT and SPA methods with "natural" left and right coprime factorizations of the controller. The frequency-weighted coprime factorization based routine AB09CD can be employed only in conjunction with the BT technique, but also allows both left and right coprime factorization based approaches.

To illustrate the available rich functionality for frequency-weighted controller reduction, the complete specification of the user interface for the controller reduction routine SB16AD is presented in Appendix A.

4.2 Supporting routines

In implementing the new model and controller reduction software, a special emphasis has been put on an appropriate modularization of the routines. For this purpose, several new supporting computational routines have been implemented for some basic tasks. Some of these routines are shared among several user callable routines. A list of lower level model and controller reduction routines is given in the next table.

Name	Function
AB09HX	BST and BST-SPA approaches for a stable original system
AB09HY	computation of Cholesky factors of gramians for the BST approach
AB09IX	computation of truncation matrices for BT and SPA methods using the Cholesky factors of the controllability and observability gramians and computation of the corresponding reduced order model
AB09IY	computation of the Cholesky factors of the frequency-weighted controllability and observability gramians
AB09JV	projection formulas for left frequency weight W_oG or $W_o^{\sim}G$ with W_o a descriptor system
AB09JW	projection formulas for right frequency weight GW_i or $GW_i^{\sim}$ with W_i a descriptor system
AB09JX	testing eigenvalues or generalized eigenvalues for stability/antistability
SB16AY	computation of the Cholesky factors of the frequency-weighted controllability and observability gramians used for controller reduction with three special stability/performance enforcing frequency-weights
SB16CY	computation of the Cholesky factors of the gramians for the frequency-weighted coprime factorization based controller reduction

Since the computation of gramians is problem specific, separate routines have been implemented to compute the gramians for each problem. To determine the two gramians P and Q needed by different methods, the routines AB09HY, AB09IY, SB16AY and SB16AY compute directly the Cholesky factors of the gramians to be employed to determine the truncation matrices by using either the **SR** or **BFSR** accuracy enhancing techniques. For this purpose, a unique routine AB09IX has been implemented to compute the truncation matrices necessary to obtain the reduced order models from the Cholesky factors of the gramians and to perform model reduction with these matrices by using either the BT or the SPA method. Note that AB09IX is called by AB09HD, AB09ID, SB16AD and SB16CD. The routines AB09JV and AB09JW implement the recently proposed descriptor systems based projection formulas [34]. AB09JX is an auxiliary routine used to check the conditions to be fulfilled by the frequency weights when employing the FWHNA approach.

An important number of new, user callable supporting routines to compute inverse systems, to solve continuous- and discrete-time Sylvester equations, to compute the normal rank of a TFM in state-space form, to compute the L_∞-norm, or to perform orthogonal reductions to generalized Hessenberg form, have been implemented for the special needs of the new model and controller reduction routines. These routines are listed in the following table:

Name	Function
AB07ND	constructs for a system (A,B,C,D) with D invertible, the matrices of the inverse system (based on SYSINV routine from RASP)
SB04PD	solves the continuous- and discrete-time Sylvester equations $op(A)X \pm Xop(B) = C$ and $op(A)Xop(B) \pm X = C$, respectively, using the Schur method ($op(A) = A$ or A^T)
SB04QD	solves the discrete-time Sylvester equations using the Hessenberg-Schur method
SB04RD	solves the discrete-time Sylvester equation $X + AXB = C$, with at least one of the matrices A or B in Schur form and the other in Hessenberg or Schur form (both either upper or lower)
MB01WD	computes $R \leftarrow \alpha[op(A)^T op(S)^T op(S) + op(S)^T op(S) op(A)] + \beta R$ or $R \leftarrow \alpha[op(A)^T op(S)^T op(S) op(A) - op(S)^T op(S)] + \beta R$, where R is symmetric, A is a full or Hessenberg matrix, and S is upper or lower triangular
MB01XD	computes the matrix product $U^T U$ or LL^T, where U and L are upper or lower triangular matrices
MB01YD	performs the symmetric rank k update $C \leftarrow \alpha op(A) op(A)^T + \beta C$, where C is symmetric and $op(A)$ has k columns
AB13DD	L-infinity norm of a linear continuous- or discrete-time system in standard or descriptor form (supersedes AB13CD)
AB08MD	normal rank of the transfer-function matrix of a state-space model
TG01BD	orthogonal reduction of a descriptor system to generalized Hessenberg form

Additionally, many low level auxiliary routines have been implemented to perform basic control computations. These routines are listed in the following table:

Name	Function
SB04PX	solve discrete-time Sylvester equations with matrices of order at most 2 (slightly improved version of SB03MU)
SB04PY	solve discrete-time Sylvester equations with matrices in real Schur form
SB04QR	solve a linear algebraic system whose coefficient matrix has zeros below the third subdiagonal and zero elements on the third subdiagonal with even column indices. The matrix is stored compactly, row-wise.
SB04QU	construct and solve a linear algebraic system whose coefficient matrix has zeros below the third subdiagonal, and zero elements on the third subdiagonal with even column indices
SB04QY	construct and solve a linear algebraic system whose coefficient matrix is in upper Hessenberg form
SB04RV	construct the right-hand sides for a system of equations in Hessenberg form solved via SB04RX (case with 2 right-hand sides)
SB04RW	construct the right-hand side for a system of equations in Hessenberg form solved via SB04RY (case with 1 right-hand side)
SB04RX	solve a system of equations in Hessenberg form with two consecutive offdiagonals and two right-hand sides (discrete-time case)
SB04RY	solve a system of equations in Hessenberg form with one offdiagonal and one right-hand side (discrete-time case)
MB01XY	computes *in situ* the matrix product U^TU or LL^T, where U and L are upper or lower triangular matrices
AB13DX	maximum singular value of the frequency response gain

5 Integration in user-friendly environments

One of the main objectives of the NICONET project was to provide, additionally to standardized Fortran codes, high quality software embedded into user-friendly environments for *computer aided control system design*. Two target environments have been envisaged: the popular commercial numerical computational environment MATLAB and the public domain MATLAB-like environment `Scilab`. Both allows to easily add external functions implemented in general purpose programming languages like C or Fortran. In the case of MATLAB, the external functions are called *mex*-functions and have to be programmed according to precise programming standards. In `Scilab`, external functions can be similarly implemented and only several minor modifications are necessary to the MATLAB *mex*-functions to adapt them to `Scilab`.

Several *mex*-functions, similar to the additive error function `sysred` [33], have been implemented as main MATLAB interfaces to the new model and controller reduction routine available in SLICOT. All functions are able to reduce both continuos- and discrete-time, stable as well as unstable systems or controllers. If appropriate, the functions can be used for unweighted reduction as well, without any significant computational overhead.

The following new *mex*-function have been implemented:

Name	Function
`bstred`	balanced stochastic truncation based model reduction (based on AB09HD)
`fwered`	frequency-weighted balancing related model reduction (based on AB09ID)
`fwehna`	frequency-weighted Hankel-norm approximation (based on AB09JD)
`conred`	frequency-weighted balancing related controller reduction (based on SB16AD)
`sfored`	coprime factorization based reduction of state feedback controllers (based on SB16BD and SB16CD)
`linorm`	L_∞-norm of a linear time-invariant system (based on AB13DD)

To illustrate the functional richness provided by the implemented *mex*-interfaces, we present in Appendix B the *mex*-function interface `conred` to SB16AD.

To provide a convenient interface to work with control objects defined in the MATLAB Control Toolbox, several easy-to-use higher level model and controller reduction *m*-functions have been additionally implemented. The list of available *mex*- and *m*-functions is given below:

`bst`	balanced stochastic truncation based model reduction
`fwbred`	frequency-weighted balancing related model reduction
`fwhna`	frequency-weighted Hankel-norm approximation
`fwbconred`	frequency-weighted balancing related controller reduction
`sfconred`	coprime factorization based state feedback controller reduction
`sysredset`	creation of a SYSRED options structure
`sysredget`	extracts the value of a named parameter from a SYSRED options structure
`linfnorm`	L-infinity norm of a state-space system

A sample *m*-function, `fwbconred.m`, representing the high level interface to the *mex*-function `conred` is listed in Appendix C.

In the implemention of the *mex*- and *m*-functions, one main goal was to allow the access to all functional facilities provided by the underlying Fortran routines. To manage the multitude of user options, a so-called SYSRED structure has been defined. This structure is created and managed via the special functions `sysredset` and `sysredget`. The available of options which can be set via the SYSRED structure are shown below:

```
      BalredMethod: [ {bta} | spa ]
AccuracyEnhancing: [ {bfsr} | sr ]
           Tolred: [ positive scalar {0} ]
       TolMinreal: [ positive scalar {0} ]
            Order: [ integer {-1} ]
         CStabDeg: [ nonpositive scalar | {-sqrt(eps)} ]
         DStabDeg: [ subunitary scalar | {1-sqrt(eps)} ]
```

```
            BstBeta: [ scalar {0}  ]
      FWEContrGramian: [ {standard} | enhanced ]
     FWEObservGramian: [ {standard} | enhanced ]
        FWEAlphaContr: [ positive subunitary scalar {0} ]
       FWEAlphaObserv: [ positive subunitary scalar {0} ]
CoprimeFactorization: [ left | {right} ]
         OutputWeight: [ {stab} | perf | none]
          InputWeight: [ {stab} | none]
       CFConredMethod: [ {fwe}  | nofwe ]
      FWEConredMethod: [ none | outputstab | inputstab | {performance} ]
         FWEHNAMethod: [ {auto} | inverse | noinverse ]
            FWEHNAopV: [ {none} | inv | conj | cinv ]
            FWEHNAopW: [ {none} | inv | conj | cinv ]
          FWEOptimize: [ none | {d} | cd ]
```

6 Comparison of available model reduction tools

We present shortly the model and controller reductions tools available in other control packages and compare them with the model reduction tools provided in SLICOT. A summary of capabilities of the reviewed software is presented in the Table 1 (see Table 7.3 and Chapter 7 of [33] for details about packages). From this table it is apparent that at present the model and controller reduction tools available in SLICOT and `Scilab` (based on the SLICOT software) and the accompanying m/textitmex-file interfaces cover practically all aspects of the model and controller reduction area.

7 Summary of results

A powerful collection of new user callable Fortran 77 routines has been implemented for model and controller reduction. The new software is based on the latest algorithmic developments and covers the relative error model reduction using the *balanced stochastic truncation* approach, model reduction using *frequency-weighted balancing* and *frequency-weighted Hankel-norm approximation* methods, as well as special controller reduction methods using *frequency-weighted balancing* and *coprime factorization* based techniques. While the new model reduction routines can be seen as extensions of already existing routines with new functionality to perform model reduction using relative error and frequency-weighted approaches, the controller reduction software complements the H_2/H_∞ controller design software available in SLICOT[3]. All implemented routines can be employed to reduce both stable and unstable, continuous- or discrete-time models or controllers. The underlying numerical algorithms are based on recent extensions of the *square-root* and *balancing-free* accuracy enhancing techniques to frequency-weighted balancing-related model reduction. The new model and controller reduction routines for SLICOT are among the most powerful and numerically most reliable software tools available for model and controller

[3]see `ftp://wgs.esat.kuleuven.ac.be/pub/WGS/REPORTS/SLWN1999-12.ps.Z`

Table 1: Summary of comparison of model reduction tools.

Software	*SLICOT*	`Scilab`	*Control Toolbox*	*Robust Toolbox*	μ*-Toolbox*	*HTOOLS*	*MATRIX*$_X$	*WOR–Toolbox*
Provided features								
continuous-time	+	+	+	+	+	+	+	+
discrete-time	+	+	+	−	−	+	−	−
unstable	+	+	−	+	−	−	−	+
non-minimal	+	+	−	+	+	+	+	+
Methods								
balancing	+	+	+	+	+	+	+	+
balancing-free (BF)	+	+	−	+	−	+	+	−
square-root (SR)	+	+	−	−	+	+	−	+
BF-SR	+	+	−	−	−	+	−	−
Problem classes								
additive error	+	+	+	+	+	+	+	+
relative error	+	+	−	+	+	+	+	−
frequency weighted	+	+	−	−	+	−	+	+
controller reduction	+	+	−	−	−	−	+	+

reduction. To facilitate their usage, easy-to-use and flexible interfaces have been developed to integrate them in MATLAB and `Scilab`.

All newly implemented software is thoroughly tested and comprehensively documented. The documentation of user callable routines is automatically generated from the comments in the preamble of each routine (see for example Appendix A for `SB16AD`). The documentation is available in *html*-format and can be viewed with standard browsers like Windows Internet Explorer or Netscape. The documentation also includes for each user callable routine a test program example, test data and the corresponding test results. The documentation of all library routines can be accessed on-line via the ftp-site of NICONET.[4]

[4] `ftp://wgs.esat.kuleuven.ac.be/pub/WGS/SLICOT/libindex.html`

A User interface for SB16AD

```
      SUBROUTINE SB16AD( DICO, JOBC, JOBO, JOBMR, WEIGHT, EQUIL, ORDSEL,
     $                   N, M, P, NC, NCR, ALPHA, A, LDA, B, LDB,
     $                   C, LDC, D, LDD, AC, LDAC, BC, LDBC, CC, LDCC,
     $                   DC, LDDC, NCS, HSVC, TOL1, TOL2, IWORK, DWORK,
     $                   LDWORK, IWARN, INFO )
C
C     RELEASE 4.0, WGS COPYRIGHT 2000.
C
C     PURPOSE
C
C     To compute a reduced order controller (Acr,Bcr,Ccr,Dcr) for an
C     original state-space controller representation (Ac,Bc,Cc,Dc) by
C     using the frequency-weighted square-root or balancing-free
C     square-root Balance & Truncate (B&T) or Singular Perturbation
C     Approximation (SPA) model reduction methods. The algorithm tries
C     to minimize the norm of the frequency-weighted error
C
C           ||V*(K-Kr)*W||
C
C     where K and Kr are the transfer-function matrices of the original
C     and reduced order controllers, respectively. V and W are special
C     frequency-weighting transfer-function matrices constructed
C     to enforce closed-loop stability and/or closed-loop performance.
C     If G is the transfer-function matrix of the open-loop system, then
C     the following weightings V and W can be used:
C                       -1
C      (a)   V = (I-G*K) *G, W = I - to enforce closed-loop stability;
C                               -1
C      (b)   V = I,  W = (I-G*K) *G - to enforce closed-loop stability;
C                       -1              -1
C      (c)   V = (I-G*K) *G, W = (I-G*K)  - to enforce closed-loop
C            stability and performance.
C
C     G has the state space representation (A,B,C,D).
C     If K is unstable, only the ALPHA-stable part of K is reduced.
C
C     ARGUMENTS
C
C     Mode Parameters
C
C     DICO    CHARACTER*1
C             Specifies the type of the original controller as follows:
C             = 'C':  continuous-time controller;
C             = 'D':  discrete-time controller.
C
C     JOBC    CHARACTER*1
C             Specifies the choice of frequency-weighted controllability
C             Gramian as follows:
C             = 'S': choice corresponding to standard Enns' method [1];
```

```
C             = 'E': choice corresponding to the stability enhanced
C                    modified Enns' method of [2].
C
C     JOBO    CHARACTER*1
C             Specifies the choice of frequency-weighted observability
C             Gramian as follows:
C             = 'S': choice corresponding to standard Enns' method [1];
C             = 'E': choice corresponding to the stability enhanced
C                    modified combination method of [2].
C
C     JOBMR   CHARACTER*1
C             Specifies the model reduction approach to be used
C             as follows:
C             = 'B':  use the square-root B&T method;
C             = 'F':  use the balancing-free square-root B&T method;
C             = 'S':  use the square-root SPA method;
C             = 'P':  use the balancing-free square-root SPA method.
C
C     WEIGHT  CHARACTER*1
C             Specifies the type of frequency-weighting, as follows:
C             = 'N':  no weightings are used (V = I, W = I);
C             = 'O':  stability enforcing left (output) weighting
C                               -1
C                     V = (I-G*K) *G is used (W = I);
C             = 'I':  stability enforcing right (input) weighting
C                               -1
C                     W = (I-G*K) *G is used (V = I);
C             = 'P':  stability and performance enforcing weightings
C                               -1                -1
C                     V = (I-G*K) *G ,  W = (I-G*K)  are used.
C
C     EQUIL   CHARACTER*1
C             Specifies whether the user wishes to preliminarily
C             equilibrate the triplets (A,B,C) and (Ac,Bc,Cc) as
C             follows:
C             = 'S':  perform equilibration (scaling);
C             = 'N':  do not perform equilibration.
C
C     ORDSEL  CHARACTER*1
C             Specifies the order selection method as follows:
C             = 'F':  the resulting order NCR is fixed;
C             = 'A':  the resulting order NCR is automatically
C                     determined on basis of the given tolerance TOL1.
C
C     Input/Output Parameters
C
C     N       (input) INTEGER
C             The order of the open-loop system state-space
C             representation, i.e., the order of the matrix A.  N >= 0.
C
C     M       (input) INTEGER
```

```
C             The number of system inputs.  M >= 0.
C
C     P       (input) INTEGER
C             The number of system outputs.  P >= 0.
C
C     NC      (input) INTEGER
C             The order of the controller state-space representation,
C             i.e., the order of the matrix AC.  NC >= 0.
C
C     NCR     (input/output) INTEGER
C             On entry with ORDSEL = 'F', NCR is the desired order of
C             the resulting reduced order controller.  0 <= NCR <= NC.
C             On exit, if INFO = 0, NCR is the order of the resulting
C             reduced order controller. For a controller with NCU
C             ALPHA-unstable eigenvalues and NCS ALPHA-stable
C             eigenvalues (NCU+NCS = NC), NCR is set as follows:
C             if ORDSEL = 'F', NCR is equal to
C             NCU+MIN(MAX(0,NCR-NCU),NCMIN), where NCR is the desired
C             order on entry, NCMIN is the number of frequency-weighted
C             Hankel singular values greater than NCS*EPS*S1, EPS is the
C             machine precision (see LAPACK Library Routine DLAMCH) and
C             S1 is the largest Hankel singular value (computed in
C             HSVC(1)); NCR can be further reduced to ensure
C             HSVC(NCR-NCU) > HSVC(NCR+1-NCU);
C             if ORDSEL = 'A', NCR is the sum of NCU and the number of
C             Hankel singular values greater than MAX(TOL1,NCS*EPS*S1).
C
C     ALPHA   (input) DOUBLE PRECISION
C             Specifies the ALPHA-stability boundary for the eigenvalues
C             of the state dynamics matrix AC. For a continuous-time
C             controller (DICO = 'C'), ALPHA <= 0 is the boundary value
C             for the real parts of eigenvalues; for a discrete-time
C             controller (DICO = 'D'), 0 <= ALPHA <= 1 represents the
C             boundary value for the moduli of eigenvalues.
C             The ALPHA-stability domain does not include the boundary.
C
C     A       (input/output) DOUBLE PRECISION array, dimension (LDA,N)
C             On entry, the leading N-by-N part of this array must
C             contain the state dynamics matrix A of the open-loop
C             system.
C             On exit, if INFO = 0 and EQUIL = 'S', the leading N-by-N
C             part of this array contains the scaled state dynamics
C             matrix of the open-loop system.
C             If EQUIL = 'N', this array is unchanged on exit.
C
C     LDA     INTEGER
C             The leading dimension of array A.  LDA >= MAX(1,N).
C
C     B       (input/output) DOUBLE PRECISION array, dimension (LDB,M)
C             On entry, the leading N-by-M part of this array must
C             contain the input/state matrix B of the open-loop system.
```

```
C             On exit, if INFO = 0 and EQUIL = 'S', the leading N-by-M
C             part of this array contains the scaled input/state matrix
C             of the open-loop system.
C             If EQUIL = 'N', this array is unchanged on exit.
C
C     LDB     INTEGER
C             The leading dimension of array B.  LDB >= MAX(1,N).
C
C     C       (input/output) DOUBLE PRECISION array, dimension (LDC,N)
C             On entry, the leading P-by-N part of this array must
C             contain the state/output matrix C of the open-loop system.
C             On exit, if INFO = 0 and EQUIL = 'S', the leading P-by-N
C             part of this array contains the scaled state/output matrix
C             of the open-loop system.
C             If EQUIL = 'N', this array is unchanged on exit.
C
C     LDC     INTEGER
C             The leading dimension of array C.  LDC >= MAX(1,P).
C
C     D       (input) DOUBLE PRECISION array, dimension (LDD,M)
C             The leading P-by-M part of this array must contain the
C             input/output matrix D of the open-loop system.
C
C     LDD     INTEGER
C             The leading dimension of array D.  LDD >= MAX(1,P).
C
C     AC      (input/output) DOUBLE PRECISION array, dimension (LDAC,NC)
C             On entry, the leading NC-by-NC part of this array must
C             contain the state dynamics matrix Ac of the original
C             controller.
C             On exit, if INFO = 0, the leading NCR-by-NCR part of this
C             array contains the state dynamics matrix Acr of the
C             reduced controller. The resulting Ac has a
C             block-diagonal form with two blocks.
C             For a system with NCU ALPHA-unstable eigenvalues and
C             NCS ALPHA-stable eigenvalues (NCU+NCS = NC), the leading
C             NCU-by-NCU block contains the unreduced part of Ac
C             corresponding to the ALPHA-unstable eigenvalues.
C             The trailing (NCR+NCS-NC)-by-(NCR+NCS-NC) block contains
C             the reduced part of Ac corresponding to ALPHA-stable
C             eigenvalues.
C
C     LDAC    INTEGER
C             The leading dimension of array AC.  LDAC >= MAX(1,NC).
C
C     BC      (input/output) DOUBLE PRECISION array, dimension (LDBC,P)
C             On entry, the leading NC-by-P part of this array must
C             contain the input/state matrix Bc of the original
C             controller.
C             On exit, if INFO = 0, the leading NCR-by-P part of this
C             array contains the input/state matrix Bcr of the reduced
```

```
C            controller.
C
C     LDBC    INTEGER
C             The leading dimension of array BC.  LDBC >= MAX(1,NC).
C
C     CC      (input/output) DOUBLE PRECISION array, dimension (LDCC,NC)
C             On entry, the leading M-by-NC part of this array must
C             contain the state/output matrix Cc of the original
C             controller.
C             On exit, if INFO = 0, the leading M-by-NCR part of this
C             array contains the state/output matrix Ccr of the reduced
C             controller.
C
C     LDCC    INTEGER
C             The leading dimension of array CC.  LDCC >= MAX(1,M).
C
C     DC      (input/output) DOUBLE PRECISION array, dimension (LDDC,P)
C             On entry, the leading M-by-P part of this array must
C             contain the input/output matrix Dc of the original
C             controller.
C             On exit, if INFO = 0, the leading M-by-P part of this
C             array contains the input/output matrix Dcr of the reduced
C             controller.
C
C     LDDC    INTEGER
C             The leading dimension of array DC.  LDDC >= MAX(1,M).
C
C     NCS     (output) INTEGER
C             The dimension of the ALPHA-stable part of the controller.
C
C     HSVC    (output) DOUBLE PRECISION array, dimension (NC)
C             If INFO = 0, the leading NCS elements of this array
C             contain the frequency-weighted Hankel singular values,
C             ordered decreasingly, of the ALPHA-stable part of the
C             controller.
C
C     Tolerances
C
C     TOL1    DOUBLE PRECISION
C             If ORDSEL = 'A', TOL1 contains the tolerance for
C             determining the order of the reduced controller.
C             For model reduction, the recommended value is
C             TOL1 = c*S1, where c is a constant in the
C             interval [0.00001,0.001], and S1 is the largest
C             frequency-weighted Hankel singular value of the
C             ALPHA-stable part of the original controller
C             (computed in HSVC(1)).
C             If TOL1 <= 0 on entry, the used default value is
C             TOL1 = NCS*EPS*S1, where NCS is the number of
C             ALPHA-stable eigenvalues of Ac and EPS is the machine
C             precision (see LAPACK Library Routine DLAMCH).
```

```
C             If ORDSEL = 'F', the value of TOL1 is ignored.
C
C     TOL2    DOUBLE PRECISION
C             The tolerance for determining the order of a minimal
C             realization of the ALPHA-stable part of the given
C             controller. The recommended value is TOL2 = NCS*EPS*S1.
C             This value is used by default if TOL2 <= 0 on entry.
C             If TOL2 > 0 and ORDSEL = 'A', then TOL2 <= TOL1.
C
C     Workspace
C
C     IWORK   INTEGER array, dimension MAX(1,LIWRK1,LIWRK2)
C             LIWRK1 = 0,       if JOBMR  = 'B';
C             LIWRK1 = NC,      if JOBMR  = 'F';
C             LIWRK1 = 2*NC,    if JOBMR  = 'S' or 'P';
C             LIWRK2 = 0,       if WEIGHT = 'N';
C             LIWRK2 = 2*(M+P), if WEIGHT = 'O', 'I', or 'P'.
C             On exit, if INFO = 0, IWORK(1) contains NCMIN, the order
C             of the computed minimal realization of the stable part of
C             the controller.
C
C     DWORK   DOUBLE PRECISION array, dimension (LDWORK)
C             On exit, if INFO = 0, DWORK(1) returns the optimal value
C             of LDWORK.
C
C     LDWORK  INTEGER
C             The length of the array DWORK.
C             LDWORK >= 2*NC*NC + MAX( 1, LFREQ, LSQRED ),
C             where
C             LFREQ = (N+NC)*(N+NC+2*M+2*P)+
C                     MAX((N+NC)*(N+NC+MAX(N+NC,M,P)+7), (M+P)*(M+P+4))
C                                  if WEIGHT = 'I' or 'O' or 'P';
C             LFREQ  = NC*(MAX(M,P)+5) if WEIGHT = 'N' and EQUIL = 'N';
C             LFREQ  = MAX(N,NC*(MAX(M,P)+5)) if WEIGHT = 'N' and
C                                               EQUIL  = 'S';
C             LSQRED = MAX( 1, 2*NC*NC+5*NC );
C             For optimum performance LDWORK should be larger.
C
C     Warning Indicator
C
C     IWARN   INTEGER
C             = 0:  no warning;
C             = 1:  with ORDSEL = 'F', the selected order NCR is greater
C                   than NSMIN, the sum of the order of the
C                   ALPHA-unstable part and the order of a minimal
C                   realization of the ALPHA-stable part of the given
C                   controller; in this case, the resulting NCR is set
C                   equal to NSMIN;
C             = 2:  with ORDSEL = 'F', the selected order NCR
C                   corresponds to repeated singular values for the
C                   ALPHA-stable part of the controller, which are
```

```
C                 neither all included nor all excluded from the
C                 reduced model; in this case, the resulting NCR is
C                 automatically decreased to exclude all repeated
C                 singular values;
C           = 3:  with ORDSEL = 'F', the selected order NCR is less
C                 than the order of the ALPHA-unstable part of the
C                 given controller. In this case NCR is set equal to
C                 the order of the ALPHA-unstable part.
C
C     Error Indicator
C
C     INFO    INTEGER
C           = 0:  successful exit;
C           < 0:  if INFO = -i, the i-th argument had an illegal
C                 value;
C           = 1:  the closed-loop system is not well-posed;
C                 its feedthrough matrix is (numerically) singular;
C           = 2:  the computation of the real Schur form of the
C                 closed-loop state matrix failed;
C           = 3:  the closed-loop state matrix is not stable;
C           = 4:  the solution of a symmetric eigenproblem failed;
C           = 5:  the computation of the ordered real Schur form of Ac
C                 failed;
C           = 6:  the separation of the ALPHA-stable/unstable
C                 diagonal blocks failed because of very close
C                 eigenvalues;
C           = 7:  the computation of Hankel singular values failed.
C
C     METHOD
C
C     Let K be the transfer-function matrix of the original linear
C     controller
C
C          d[xc(t)] = Ac*xc(t) + Bc*y(t)
C          u(t)     = Cc*xc(t) + Dc*y(t),                      (1)
C
C     where d[xc(t)] is dxc(t)/dt for a continuous-time system and
C     xc(t+1) for a discrete-time system. The subroutine SB16AD
C     determines the matrices of a reduced order controller
C
C          d[z(t)] = Acr*z(t) + Bcr*y(t)
C          u(t)    = Ccr*z(t) + Dcr*y(t),                      (2)
C
C     such that the corresponding transfer-function matrix Kr minimizes
C     the norm of the frequency-weighted error
C
C             V*(K-Kr)*W,                                      (3)
C
C     where V and W are special stable transfer-function matrices
C     chosen to enforce stability and/or performance of the closed-loop
C     system [3] (see description of the parameter WEIGHT).
```

```
C
C     The following procedure is used to reduce K in conjunction
C     with the frequency-weighted balancing approach of [2] and [4]
C     (see also [3]):
C
C     1) Decompose additively K, of order NC, as
C
C          K = K1 + K2,
C
C        such that K1 has only ALPHA-stable poles and K2, of order NCU,
C        has only ALPHA-unstable poles.
C
C     2) Compute for K1 a B&T or SPA frequency-weighted approximation
C        K1r of order NCR-NCU using the frequency-weighted balancing
C        approach of [1] in conjunction with accuracy enhancing
C        techniques specified by the parameter JOBMR.
C
C     3) Assemble the reduced model Kr as
C
C           Kr = K1r + K2.
C
C     For the reduction of the ALPHA-stable part, several accuracy
C     enhancing techniques can be employed (see [2] for details).
C
C     If JOBMR = 'B', the square-root B&T method of [1] is used.
C
C     If JOBMR = 'F', the balancing-free square-root version of the
C     B&T method [1] is used.
C
C     If JOBMR = 'S', the square-root version of the SPA method [2,3]
C     is used.
C
C     If JOBMR = 'P', the balancing-free square-root version of the
C     SPA method [2,3] is used.
C
C     For each of these methods, two left and right truncation matrices
C     are determined using the Cholesky factors of an input
C     frequency-weighted controllability Gramian P and an output
C     frequency-weighted observability Gramian Q.
C     P and Q are determined as the leading NC-by-NC diagonal blocks
C     of the controllability Gramian of K*W and of the
C     observability Gramian of V*K. Special techniques developed in [2]
C     are used to compute the Cholesky factors of P and Q directly
C     (see also SLICOT Library routine SB16AY).
C     The frequency-weighted Hankel singular values HSVC(1), ....,
C     HSVC(NC) are computed as the square roots of the eigenvalues
C     of the product P*Q.
C
C     REFERENCES
C
C     [1] Enns, D.
```

```
C         Model reduction with balanced realizations: An error bound
C         and a frequency weighted generalization.
C         Proc. 23-th CDC, Las Vegas, pp. 127-132, 1984.
C
C     [2] Varga, A. and Anderson, B.D.O.
C         Square-root balancing-free methods for frequency-weighted
C         balancing related model reduction.
C         Proc. of CDC'2001, Orlando, FL, pp. 3659-3664, 2001.
C
C     [3] Anderson, B.D.O and Liu, Y.
C         Controller reduction: concepts and approaches.
C         IEEE Trans. Autom. Control, Vol. 34, pp. 802-812, 1989.
C
C     [4] Varga, A. and Anderson, B.D.O.
C         Frequency-weighted balancing related controller reduction.
C         Proc. of IFAC'2002 Congress, Barcelona, Spain, 2002.
C
C     NUMERICAL ASPECTS
C
C     The implemented methods rely on accuracy enhancing square-root
C     techniques.
C
C     KEYWORDS
C
C     Controller reduction, frequency weighting, multivariable system,
C     state-space model, state-space representation.
C
C     ******************************************************************
C
C     .. Parameters ..
      DOUBLE PRECISION  C100, ONE, ZERO
      PARAMETER         ( C100 = 100.0D0, ONE = 1.0D0, ZERO = 0.0D0 )
C     .. Scalar Arguments ..
      CHARACTER         DICO, EQUIL, JOBC, JOBO, JOBMR, ORDSEL, WEIGHT
      INTEGER           INFO, IWARN, LDA, LDAC, LDB, LDBC, LDC, LDCC,
     $                  LDD, LDDC, LDWORK, M, N, NC, NCR, NCS, P
      DOUBLE PRECISION  ALPHA, TOL1, TOL2
C     .. Array Arguments ..
      INTEGER           IWORK(*)
      DOUBLE PRECISION  A(LDA,*), AC(LDAC,*), B(LDB,*), BC(LDBC,*),
     $                  C(LDC,*), CC(LDCC,*), D(LDD,*), DC(LDDC,*),
     $                  DWORK(*), HSVC(*)
C
```

B Matlab *mex*-function interface CONRED to SB16AD

```
C CONRED.F - Gateway function for SLICOT controller reduction routine
C            SB16AD.F.
C
C RELEASE 4.0, WGS COPYRIGHT 2001.
C
C Matlab call:
C   [Acr,Bcr,Ccr,Dcr,HSVC,info] = CONRED(meth,Ac,Bc,Cc,Dc,A,B,C,D,...
C                                        tol,discr,ord,alpha)
C
C Purpose:
C   To compute a reduced order controller (Acr,Bcr,Ccr,Dcr) for an
C   original state-space controller representation (Ac,Bc,Cc,Dc) by
C   using the frequency-weighted square-root or balancing-free
C   square-root Balance & Truncate (B&T) or Singular Perturbation
C   Approximation (SPA) model reduction methods. The algorithm tries
C   to minimize the norm of the frequency-weighted error
C
C          ||V*(K-Kr)*W||
C
C   where K and Kr are the transfer-function matrices of the original
C   and reduced order controllers, respectively. V and W are special
C   frequency-weighting transfer-function matrices constructed
C   to enforce closed-loop stability and/or closed-loop performance.
C   If G is the transfer-function matrix of the open-loop system, then
C   the following weightings V and W can be used:
C                     -1
C      (a)   V = (I-G*K) *G, W = I - to enforce closed-loop stability;
C                             -1
C      (b)   V = I,  W = (I-G*K) *G - to enforce closed-loop stability;
C                     -1              -1
C      (c)   V = (I-G*K) *G, W = (I-G*K)  - to enforce closed-loop
C            stability and performance.
C
C   G has the state space representation (A,B,C,D).
C   If K is unstable, only the ALPHA-stable part of K is reduced.
C
C Input parameters:
C   meth  - method flag of decimal form ijkl, where
C            i = 1 : use standard choice for controllability Gramian;
C            i = 2 : use stability garanteeing choice for the
C                    controllability Gramian;
C            j = 1 : use standard choice for observability Gramian;
C            j = 2 : use stability garanteeing choice for the
C                    observability Gramian;
C            k = 1 : use the square-root BT method;
C            k = 2 : use the balancing-free square-root BT method;
C            k = 3 : use the square-root SPA method;
C            k = 4 : use the balancing-free square-root SPA method;
C            l = 1 : no weightings are used;
```

```
C            l = 2 : stability enforcing left (output) weighting;
C            l = 3 : stability enforcing right (input) weighting;
C            l = 4 : stability and performance enforcing weightings.
C          Note: For a complete explanation on Gramian choices
C                see subroutine SB16AD.
C   AC,BC,
C   CC,DC - state-space system matrices of size NC-by-NC, NC-by-P,
C          M-by-NC, and M-by-P, respectively.
C   A,B,
C   C,D   - (optional) state-space system matrices of size
C          N-by-N, N-by-M, P-by-N, and P-by-M, respectively.
C   tol   - (optional) tolerance vector for determining the order of
C          reduced system, of the form [tol1, tol2], where:
C            tol1 specifies the tolerance for model reduction.
C                 Default: tol1 = NCS*epsilon_machine*HSVC(1), where
C                 NCS is the order of the alpha-stable part of K.
C            tol2 specifies the tolerance for minimal realization.
C                 Default: tol2 = NCS*epsilon_machine*HSVC(1).
C   discr - (optional) type of system:
C             = 0 : continuous-time (default);
C             = 1 : discrete-time.
C   ord   - (optional) desired order of reduced system.
C             Default: ord = -1 (order determined automatically).
C   alpha - (optional) stability boundary for the eigenvalues of AC.
C             Default:    -sqrt(epsilon_machine)  for continuous-time;
C                      1.0-sqrt(epsilon_machine)  for discrete-time.
C
C Output parameters:
C   Acr, Bcr,
C   Ccr, Dcr - matrices of the reduced controller.
C   HSVC   - frequency-weighted Hankel singular values of the
C           alpha-stable part of K.
C   info   - warning message code:
C           info = 1 - selected order greater than the order
C                      of a minimal realization;
C           info = 2 - selected order corresponds to repeated singular
C                      values, which are neither all included nor all
C                      excluded from the reduced model;
C           info = 3 - selected order less than the order of
C                      the unstable part.
C
```

C Matlab *m*-function interface FWBCONRED to CONRED

```
function [syscr,hsvc] = fwbconred(sysc,varargin)
%FWBCONRED Frequency-weighted balancing related controller reduction.
%       [SYSCR,HSVC] = FWBCONRED(SYSC,SYS,TOL,ORD)  calculates for the
%       transfer function
%                                             -1
%               Gc(lambda) =  Cc(lambdaI-Ac) Bc + Dc
%
%       of an original controller SYSC = (Ac,Bc,Cc,Dc), an approximate
%       transfer function
%                                                -1
%               Gcr(lambda) =  Ccr(lambdaI-Acr) Bcr + Dcr
%
%       of a reduced order controller SYSCR = (Acr,Bcr,Ccr,Dcr) by
%       minimizing the frequency-weighted error norm
%
%              ||V*(Gc-Gcr)*W||
%
%       using frequency-weighted balancing related approximation
%       methods on the stable part of SYSC. V and W are special
%       frequency-weighting transfer-function matrices constructed
%       to enforce closed-loop stability and/or closed-loop performance.
%       If G is the transfer-function matrix of the open-loop system SYS,
%       then the following weightings V and W can be used:
%                         -1
%       (a)   V = (I-G*K) *G, W = I - to enforce closed-loop stability;
%                                 -1
%       (b)   V = I,  W = (I-G*K) *G - to enforce closed-loop stability;
%                         -1              -1
%       (c)   V = (I-G*K) *G, W = (I-G*K)  - to enforce closed-loop
%             stability and performance.
%       TOL is the tolerance for controller reduction.
%       ORD specifies the desired order of the reduced controller SYSCR.
%
%       HSVC contains the decreasingly ordered frequency-weighted Hankel
%       singular values of the stable part of SYSC.
%
%       [SYSCR,HSVC] = FWBCONRED(SYSC,SYS,OPTIONS) calculates the
%       reduced order controller using the option values in the structure
%       OPTIONS, created with the SYSREDSET function.  See SYSREDSET for details.
%       FWBCONRED uses these options: BalredMethod, AccuracyEnhancing,
%       FWEContrGramian, FWEObservGramian, TolRed, TolMinreal,
%       CStabDeg, DStabDeg, Order, FWEConredMethod.
%
%       The choice of frequency-weighting can be specified by the
%       OPTIONS.FWEConredMethod structure element as follows:
%       'none'        - no weighting;
%       'outputstab'  - for the stability enforcing choice (a);
%       'inputstab'   - for the stability enforcing choice (b);
%       'performance' - for the stability and performance enforcing choice (c).
```

```
%
%       An arbitrary stability degree parameter ALPHA can be specified
%       in the structure OPTIONS as OPTIONS.CStabDeg for a continuous-time
%       system or OPTIONS.DStabDeg for a discrete-time system.
%       ALPHA is the stability boundary for the eigenvalues of Ac.
%       For a continuous-time system ALPHA <= 0 is the boundary value
%       for the real parts of eigenvalues, while for a discrete-time
%       system, 1 >= ALPHA >= 0 represents the boundary value for the
%       moduli of eigenvalues.
%
%       The order NR of the reduced controller SYSCR is determined as follows:
%       let NU be the order of the ALPHA-unstable part of SYSC and let
%       NSMIN be the order of a minimal realization of the ALPHA-stable
%       part. Then
%       (1) if TOL > 0 and  ORD < 0, then NR = NU + min(NRS,NSMIN), where
%           NRS is the number of Hankel singular values greater than TOL;
%       (2) if ORD >= 0, then NR = NU + min(max(0,ORD-NU),NSMIN).

%       Method:
%       The following approach is used to reduce a given Gc:
%       1) Decompose additively Gc as
%
%            Gc = Gc1 + Gc2
%
%          such that Gc1 = (Acs,Bcs,Ccs,Dc) has only ALPHA-stable poles and
%          Gc2 = (Acu,Bcu,Ccu,0) has only ALPHA-unstable poles.
%       2) Determine Gc1r, a reduced order approximation of the
%          ALPHA-stable part Gc1 using the frequency-weighted
%          Balance & Truncate Approximation method.
%       3) Assemble the reduced model Gcr as
%
%             Gcr = Gc1r + Gc2.
%
%       Interface M-function to the SLICOT-based MEX-function CONRED.
%       A. Varga 05-11-2000; revised 22-05-2001.
%       Revised, V. Sima 23-06-2001.

defaultopt = struct( ...
    'BalredMethod', 'bta',...
    'AccuracyEnhancing', 'bfsr', ...
    'FWEContrGramian', 'standard', ...
    'FWEObservGramian', 'standard', ...
    'FWEConredMethod', 'performance', ...
    'TolRed', 0, ...
    'TolMinreal', 0, ...
    'CStabDeg', -sqrt(eps), ...
    'DStabDeg', 1-sqrt(eps), ...
    'FWEAlphaContr', 0, ...
    'FWEAlphaObserv', 0, ...
    'Order', -1);
```

```
% If just 'defaults' passed in, return the default options in SYSCR
if nargin == 1 & nargout <= 1 & isequal(sysc,'defaults')
   syscr = defaultopt;
   return
end

if ~isa(sysc,'lti')
   error('The input controller SYSC must be an LTI object')
end

% initialization
ni = nargin;
discr = sysc.ts > 0;
if nargin > 1
   if isstruct(varargin{nargin-1})
      options = varargin{nargin-1};
      ni = ni-1;
   else
      options = [];
   end
else
   options = [];
end
if ni < 4
   ord = sysredget(options,'Order',defaultopt,'fast');
else
   ord = varargin{3};
end
if ni < 3
   tol = sysredget(options,'TolRed',defaultopt,'fast');
else
   tol = varargin{2};
end
if ni < 2
   sys = [];
else
   sys = varargin{1};
end

if ~isempty(sys)
   if ~isa(sys,'lti')
      error('SYS must be an LTI object or a matrix')
   end
   if size(sys,2) ~= size(sysc,1) | size(sys,1) ~= size(sysc,2)
      error('SYSC and SYS have incompatible dimensions')
   end
   if sysc.ts ~= sys.ts
      error('SYSC and SYS must have the same sampling time')
   end
   [a,b,c,d] = ssdata(sys);
else
```

```
    a = []; b = []; c = []; d = [];
end

[ac,bc,cc,dc] = ssdata(sysc);

balmeth    = sysredget(options,'BalredMethod',defaultopt,'fast');
accenh     = sysredget(options,'AccuracyEnhancing',defaultopt,'fast');
tolmin     = sysredget(options,'TolMinreal',defaultopt,'fast');
alphac     = sysredget(options,'FWEAlphaContr',defaultopt,'fast');
alphao     = sysredget(options,'FWEAlphaObserv',defaultopt,'fast');
gramc      = sysredget(options,'FWEContrGramian',defaultopt,'fast');
gramo      = sysredget(options,'FWEObservGramian',defaultopt,'fast');
conredmeth = sysredget(options,'FWEConredMethod',defaultopt,'fast');
if discr
   alpha   = sysredget(options,'DStabDeg',defaultopt,'fast');
else
   alpha   = sysredget(options,'CStabDeg',defaultopt,'fast');
end

if strcmp(conredmeth,'none')
   meth = 1;
elseif strcmp(conredmeth,'outputstab')
   meth = 2;
elseif strcmp(conredmeth,'inputstab')
   meth = 3;
else
   meth = 4;
end
if strcmp(balmeth,'bta')
   meth = meth + 10;
else
   meth = meth + 30;
end
if strcmp(accenh,'bfsr')
   meth = meth + 10;
end
if strcmp(gramo,'standard')
   meth = meth + 100;
else
   meth = meth + 200;
end
if strcmp(gramc,'standard')
   meth = meth + 1000;
else
   meth = meth + 2000;
end

[acr,bcr,ccr,dcr,hsvc] = conred(meth,ac,bc,cc,dc,a,b,c,d,[tol tolmin],discr,...
                                ord,alpha);
syscr = ss(acr,bcr,ccr,dcr,sysc);
% end fwbconred
```

References

[1] B. D. O. Anderson and Y. Liu. Controller reduction: concepts and approaches. *IEEE Trans. Autom. Control*, 34:802–812, 1989.

[2] E. Anderson, Z. Bai, J. Bishop, J. Demmel, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK User's Guide, Second Edition.* SIAM, Philadelphia, 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, pages 499–539. Birkhäuser, 1999.

[4] U. B. Desai and D. Pal. A transformation approach to stochastic model reduction. *IEEE Trans. Autom. Control*, 29:1097–1100, 1984.

[5] C. Gomez (Ed.). *Engineering and Scientific Computing with Scilab.* Birkhauser, Boston, 1999.

[6] D. Enns. *Model Reduction for Control Systems Design.* PhD thesis, Dept. Aeronaut. Astronaut., Stanford Univ., Stanford, CA, 1984.

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

[8] M. Green and B.D.O. Anderson. Generalized balanced stochastic-truncation. In *Proc. 29th CDC*, pages 476–481, 1990.

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

[10] Y. S. Hung and K. Glover. Optimal Hankel-norm approximation of stable systems with first-order stable weighting functions. *Systems & Control Lett.*, 7:165–172, 1986.

[11] G. A. Latham and B. D. O. Anderson. Frequency-weighted optimal Hankel norm approximation of stable transfer functions. *Systems & Control Lett.*, 5:229–236, 1985.

[12] C.-A. Lin and T.-Y. Chiu. Model reduction via frequency weighted balanced realization. *CONTROL - Theory and Advanced Technology*, 8:341–351, 1992.

[13] Y. Liu and B. D. O. Anderson. Singular perturbation approximation of balanced systems. *Int. J. Control*, 50:1379–1405, 1989.

[14] Y. Liu, B. D. O. Anderson, and U. L. Ly. Coprime factorization controller reduction with Bezout identity induced frequency weighting. *Automatica*, 26:233–249, 1990.

[15] T. Lüttich and W. Marquardt. SLICOT: a useful tool in industry? Application of SLICOT model reduction routines. *NICONET Newsletter*, No.9, pp. 13–16, 2002. `ftp://wgs.esat.kuleuven.ac.be/pub/WGS/NEWSLETTER/issue-2-02.ps.Z`

[16] $\mathrm{MATRIX_X}$. *Xmath Model Reduction Module.* ISI, Santa Clara, CA, January 1998.

[17] B. C. Moore. Principal component analysis in linear system: controllability, observability and model reduction. *IEEE Trans. Autom. Control*, 26:17–32, 1981.

[18] C. Oară and A. Varga. Minimal degree coprime factorization of rational matrices. *SIAM J. Matrix Anal. Appl.*, 21:245–278, 1999.

[19] G. Obinata and B. D. O. Anderson. *Model Reduction for Control System Design.* Springer Verlag, Berlin, 2000.

[20] L. Pernebo and L. M. Silverman. Model reduction via balanced state space representations. *IEEE Trans. Autom. Control*, 27:382–387, 1982.

[21] M. G. Safonov and R. Y. Chiang. Model reduction for robust control: a Schur relative error method. *Int. J. Adapt. Contr.&Sign. Proc.*, 2:259–272, 1988.

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

[23] V. Sreeram, B.D.O. Anderson, and A.G. Madievski. New results on frequency weighted balanced reduction technique. In *Proc. 1995 ACC, Sreattle, WA*, pages 4004–4009, 1995.

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

[25] A. Varga. Efficient minimal realization procedure based on balancing. In A. El Moudni, P. Borne, and S. G. Tzafestas, editors, *Proc. of IMACS/IFAC Symp. on Modelling and Control of Technological Systems, Lille, France*, volume 2, pages 42–47, 1991.

[26] A. Varga. Efficient minimal realization procedure based on balancing. In A. El Moudni, P. Borne, and S. G. Tzafestas, editors, *Prepr. of IMACS Symp. on Modelling and Control of Technological Systems*, volume 2, pages 42–47, 1991.

[27] A. Varga. Coprime factors model reduction based on square-root balancing-free techniques. In A. Sydow, editor, *Computational System Analysis 1992, Proc. 4-th Int. Symp. Systems Analysis and Simulation, Berlin, Germany*, pages 91–96. Elsevier, Amsterdam, 1992.

[28] A. Varga. *RASP Model Order Reduction Programs.* University of Bochum and DLR-Oberpfaffenhofen, TR R88-92, August 1992.

[29] A. Varga. Enhanced modal approach for model reduction. *Mathematical Modelling of Systems*, 1:91–105, 1995.

[30] A. Varga. Computation of coprime factorizations of rational matrices. *Lin. Alg. & Appl.*, 271:83–115, 1998.

[31] A. Varga. Model reduction software in the SLICOT library. In *Proc. CACSD'2000 Symposium, Anchorage, Alaska*, 2000.

[32] A. Varga. On stochastic balancing related model reduction. In *Proc. CDC'2000, Sydney, Australia*, pages 2385–2390, 2000.

[33] A. Varga. Model reduction software in the SLICOT library. In B. N. Datta, editor, *Applied and Computational Control, Signals and Circuits*, volume 629 of *The Kluwer International Series in Engineering and Computer Science*, pages 239–282. Kluwer Academic Publishers, Boston, 2001.

[34] A. Varga. Numerical approach for the frequency-weighted Hankel-norm approximation. In *Proc. of ECC'2001, Porto, Portugal*, pages 640–645, 2001.

[35] A. Varga and B. D. O. Anderson. Square-root balancing-free methods for the frequency-weighted balancing related model reduction. *Proc. of CDC'2001, Orlando, FL*, pages 3659–3664, 2001.

[36] A. Varga and B. D. O. Anderson. Frequency-weighted balancing related controller reduction. In *Proc. of IFAC'2002 Congress, Barcelona, Spain*, 2002.

[37] A. Varga and P. Parrilo. Fast algorithms for solving H_∞-norm minimization problems. In *Proc. of CDC'2001, Orlando, FL*, pages 261–266, 2001.

[38] G. Wang, V. Sreeram, and W. Q. Liu. A new frequency-weighted balanced truncation method and error bound. *IEEE Trans. Autom. Control*, 44:1734–1737, 1999.

[39] K. Zhou. Frequency-weighted L_∞ norm and optimal Hankel norm model reduction. *IEEE Trans. Autom. Control*, 40:1687–1699, 1995.

[40] K. Zhou, J. C. Doyle, and K. Glover. *Robust and Optimal Control.* Prentice Hall, 1996.