Model Reduction Software in the SLICOT Library

A. Varga

German Aerospace Center (DLR) - Oberpfaffenhofen
Institute of Robotics and Mechatronics
D-82234 Wessling, Germany
Andras.Varga@dlr.de

Abstract

We describe the model reduction software developed recently for the control and systems library SLICOT. Besides a powerful collection of Fortran 77 routines implementing the last algorithmic developments for several well-known balancing related methods, we also describe model reduction tools developed to facilitate the usage of SLICOT routines in user friendly environments like MATLAB or Scilab. Extensive testing of the implemented tools has been done using both special benchmark problems as well as models of several complex industrial plants. Testing results and performance comparisons show the superiority of SLICOT model reduction tools over existing model reduction software.

1 Introduction

Model reduction is of fundamental importance in many modeling and control applications. Still, reliable and high quality model reduction software tools are scarce. Even the model reduction tools available in commercial packages have strong limitations because using either inappropriate algorithms or poor software implementations. The lack of good general purpose model reduction software was the motivation to develop with the highest priority a model reduction chapter for the control and systems library SLICOT in the framework of the European project NICONET[1] [2].

In this paper we describe the recently developed model reduction software for SLICOT. A powerful collection of user callable Fortran 77 routines has been implemented based on the latest algorithmic developments for three balancing related methods: the *Balance and Truncate* (B&T) method [9], the *Singular Perturbation Approximation* (SPA) method [7] and the *Hankel-Norm Approximation* (HNA) method [5]. These methods belong to the class of additive error methods and rely on guaranteed error bounds. They are primarily intended for the reduction of linear, stable, continuous- or discrete-time systems. However, in combination with additive spectral decomposition or coprime factorization techniques the basic methods can be employed to reduce unstable systems too.

The new model reduction routines for SLICOT are among the most powerful and numerically most reliable software tools available for model reduction. To facilitate the usage of the new model reduction routines, 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 [3].

Several benchmark examples have been employed for testing and performance comparisons. The test results and performance comparisons show the superiority of SLICOT model reduction tools over existing model reduction software. The complete test results and comparisons with existing model reduction software is presented in an extended version of this paper [21].

2 Development of model reduction subroutines

In this section we present the standardization effort done within the NICONET project [2] to develop numerically reliable software for model reduction for the SLICOT library.

2.1 Balancing related model reduction

Three basic model reduction algorithms belonging to the class of methods based on or related to balancing techniques [9, 7, 5] form the basis of model reduction software in SLICOT. These methods are primarily intended for the reduction of linear, stable, continuous- or discrete-time systems. They rely on guaranteed error bounds and have particular features which recommend them for use in specific applications. In what follows we present succinctly the main features of balancing related model reduction.

Consider the n-th order original state-space model

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

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

$G := (A, B, C, D)$ with the *transfer-function matrix* (TFM) $G(\lambda) = C(\lambda I - A)^{-1}B + D$, and let $G_r := (A_r, B_r, C_r, D_r)$ be an r-th order approximation of the original model ($r < n$), with the TFM $G_r = C_r(\lambda I - A_r)^{-1}B_r + D_r$. According to the system type, λ is either the complex variable s appearing in the Laplace transform in case of a continuous-time system or the variable z appearing in the Z-transform in case of a discrete-time system. In our overview we focus on the so-called absolute (or additive) error model reduction method which essentially tries to minimize the absolute approximation error

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

A large class of model reduction methods can be interpreted as performing a similarity transformation Z yielding

$$\left[\begin{array}{c|c} Z^{-1}AZ & Z^{-1}B \\ \hline CZ & D \end{array}\right] := \left[\begin{array}{cc|c} A_{11} & A_{12} & B_1 \\ A_{21} & A_{22} & B_2 \\ \hline C_1 & C_2 & D \end{array}\right], \tag{2}$$

and then defining the reduced model G_r as the leading diagonal system

$$(A_r, B_r, C_r, D_r) = (A_{11},\, B_1,\, C_1,\, D). \tag{3}$$

When writing Z and Z^{-1} in partitioned form

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

then $LT = I_r$ and $\Pi = TL$ is a projection matrix. Thus the reduced system is given by

$$(A_r, B_r, C_r, D_r) = (LAT,\, LB,\, CT,\, D).$$

The matrices L and T are called *truncation* matrices.

The partitioned system matrices in (2) can be used to construct a so-called *singular perturbation approximation* (SPA):

$$\begin{array}{rcl} A_r &=& A_{11} + A_{12}(\gamma I - A_{22})^{-1}A_{21}\ , \\ B_r &=& B_1 + A_{12}(\gamma I - A_{22})^{-1}B_2\ , \\ C_r &=& C_1 + C_2(\gamma I - A_{22})^{-1}A_{21}\ , \\ D_r &=& D + C_2(\gamma I - A_{22})^{-1}B_2\ , \end{array} \tag{4}$$

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

Another class of so-called *modal* methods, determine Z such that

$$\left[\begin{array}{c|c} Z^{-1}AZ & Z^{-1}B \\ \hline CZ & D \end{array}\right] := \left[\begin{array}{cc|c} A_{11} & 0 & B_1 \\ 0 & A_{22} & B_2 \\ \hline C_1 & C_2 & D \end{array}\right], \tag{5}$$

and A_{11} and A_{22} contain the *dominant* and *non-dominant* modes of the system, respectively. Thus, the reduced model (3) contains essentially the dominant dynamics of the system. The modal approach can be easily used in combination with balancing related methods for reduction of unstable systems. In this case, the dominant part includes all unstable dynamics and the model reduction is performed only on the non-dominant stable part.

In selecting numerical algorithms for model reduction, specific requirements for a satisfactory algorithm have been formulated to assess its suitability to serve as basis for robust numerical software implementations: (1) general applicability regardless the original system is minimal or not; (2) emphasis on enhancing the numerical accuracy of computations; (3) relying on numerically reliable procedures; (4) independence of results of state space coordinate scaling. In what follows we only discuss aspects (1) and (2). For the complete discussion of all these features see [21].

The first requirement is very important because, in practice, due to the presence of roundoff errors, it is often impossible to distinguish between a *true* non-minimal and a *nearly* non-minimal system. At algorithmic level, this requirement can be fulfilled by using algorithms which compute L and T directly, without determining Z or Z^{-1}. In particular, if the original system is non-minimal, then L and T can be chosen to compute an *exact* minimal realization of the original system [14]. In this way, model reduction can also serve as a numerically sound alternative to solve minimal realization problems.

The emphasis on improving the accuracy of computations led to so-called algorithms with *enhanced accuracy*. In the balancing related model reduction methods, the truncation matrices L and T are usually determined from the controllability and observability gramians P and Q, satisfying a pair of continuous-time Lyapunov equations

$$\begin{array}{rcl} AP + PA^T + BB^T &=& 0 \\ A^TQ + QA + C^TC &=& 0 \end{array} \tag{6}$$

or a pair discrete-time Lyapunov equations

$$\begin{array}{rcl} APA^T + BB^T &=& P \\ A^TQA + C^TC &=& Q \end{array} \tag{7}$$

Since P and Q are positive semi-definite symmetric matrices, they can be expressed in Cholesky factorized forms $P = S^TS$ and $Q = R^TR$ with S and R upper-triangular matrices. The Cholesky factors can be computed directly by solving (6) or (7) using numerically reliable algorithms proposed by Hammarling [6] to solve non-negative definite Lyapunov equations. Then, the computation of L and T can be done entirely on basis of the Cholesky factors S and R, leading to the so-called *square-root* (**SR**) methods for model reduction.

Consider the *singular value decomposition* (SVD)

$$SR^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \mathrm{diag}(\Sigma_1, \Sigma_2) \begin{bmatrix} V_1 & V_2 \end{bmatrix}^T, \quad (8)$$

where

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

and $\sigma_1 \geq \ldots \geq \sigma_r > \sigma_{r+1} \geq \ldots \geq \sigma_n \geq 0$ are *the Hankel singular values* of the system. The (**SR**) method for the B&T approach of [9] determines L and T as [12]

$$L = \Sigma_1^{-1/2} V_1^T R, \qquad T = S^T U_1 \Sigma_1^{-1/2}.$$

If r is the order of a minimal realization of G then the gramians corresponding to the resulting realization are diagonal and equal. In this case the minimal realization is called *balanced*. Since the **SR** method can be used to compute balanced minimal representations resulting in a partitioned form like in (2), it can also be used for computing reduced order models by using the SPA formulas [7] or as the preliminary step in performing the HNA [5]. 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 losses maximal rank).

In order to avoid computations with ill-conditioned truncation matrices, a *balancing-free* (**BF**) approach has been proposed in [11], where well-conditioned truncation matrices L and T can be always determined. These matrices are computed from orthogonal matrices whose columns span the 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 for the B&T method, combining the main advantages of the **BF** and **SR** approaches has been introduced in [14]. The truncation matrices L and T can be 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

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

with W and Z non-singular (upper-triangular) matrices. The accuracy of the **BFSR** algorithm is usually better than either of **SR** or **BF** approaches. A **BFSR** method for the SPA approach has been proposed in [13]. The matrices L and T are computed such that the system (LAT, LB, CT, D) is minimal and the product of corresponding gramians has a block-diagonal structure which allows the application of the SPA formulas.

For all three methods B&T, SPA and HNA the norm of the absolute approximation error $G - G_r$ for an r-th order approximation satifies

$$\|G - G_r\|_\infty \leq 2 \sum_{k=r+1}^{n} \sigma_k. \quad (9)$$

Note that the actual error may be considerably less than the above error bound, so that this formula can be seen generally only as a guide to choose the appropriate order of the reduced system. In case of optimal HNA method, the optimum G_r achieves

$$\inf \|G - G_r\|_H = \sigma_{r+1}$$

and a feedtrough matrix D_r can be chosen (see [5] for details) such that the error bound in (9) is one half of the bound for B&T and SPA. This feature is not available in the implemented SLICOT routine for HNA.

2.2 Reduction of unstable systems

The reduction of unstable systems can be performed by using the methods for stable systems in conjunction with two embedding techniques. The first approach (see [19]) consists in reducing only the stable projection of G, computed via a spectral separation of the form (5), and then including the unstable projection unmodified in the resulting reduced model. The second approach (see [16]) is based on reducing the factors of a stable rational coprime factorization of G. For instance, having $G = M^{-1}N$, where M and N are stable rational TFMs of same order as G, we can compute an approximation $[\, N_r \; M_r \,]$ of order r of $[\, N \; M \,]$ and form the r-th order approximation $G_r = M_r^{-1} N_r$. Note that both approaches for the reduction of unstable system can be implemented with practically no computational overhead if the original system G is already stable.

2.3 Implemented model reduction software

The basis for implementation of the model reduction routines in SLICOT formed the collection of routines available in the RASP-MODRED library [15], implemented on basis of the linear algebra standard package LAPACK [1]. The underlying algorithms fulfill the requirements for generality, numerical reliability, enhanced accuracy, etc. formulated for model reduction algorithms and thus are completely satisfactory to serve as bases for robust software implementations. All new SLICOT routines originating from the RASP-MODRED library have been practically rewritten. Several routines represent completely new implementations. A special emphasis has been put on an appropriate modularization of the routines in the model reduction chapter of SLICOT. For this purpose, a computational kernel formed of three basic routines is shared by all higher level user callable routines.

Three user callable routines AB09AD, AB09BD and AB09CD implement the basic algorithms for B&T,

SPA and HNA methods, respectively. Both the **SR** and **BFSR** versions of the B&T and SPA algorithms are implemented in AB09AD and AB09BD. The implementation in AB09CD of the HNA method uses the **SR** B&T method to compute a balanced minimal realization of the original system. The singular perturbation formulas (4) are implemented in the user-callable routine AB09DD. All model reduction routines perform optionally the scaling of the original system and handle both continuous-time as well as discrete-time systems. For implementing the discrete-time HNA method, bilinear continuous-to-discrete transformation techniques have been employed.

Three supporting routines AB09AX, AB09BX and AB09CX perform basically the same reductions as the corresponding main user callable routines, but for systems with the state matrix already reduced to the real Schur form and possibly already scaled. These lower level routines form the computational kernel of the whole model reduction software in SLICOT, being called by the user-callable routines for reduction of both stable and unstable systems.

SLICOT also provides tools to perform the reduction of unstable systems. On the basis of newly implemented routines to compute left/right coprime factorizations with prescribed stability degree or with inner denominators, or to compute additive spectral decompositions (see next paragraph), several user callable routines have been implemented for reduction of unstable systems. A modular implementation allowed flexible combinations between various factorization/decomposition and model reduction methods for stable systems.

The routines AB09ED, AB09MD and AB09ND implement the spectral separation approach in combination with HNA, B&T, and SPA methods, respectively. 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 balancing methods with the dominant modal reduction (see also [19]). The coprime factorization based routines AB09FD and AB09GD allows arbitrary combinations of B&T and SPA methods, respectively, with four types of coprime factorizations.

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. Since these routines can be seen as completely general tools for order reduction of linear time-invariant systems, they form the basis to implement the interface software to user-friendly environments (see Section 3).

2.4 Additional software for model reduction

An important number of user-callable and auxiliary routines have been implemented for the special needs of the model reduction software, as for example for computing system norms, several factorizations and decomposition of TFMs, or frequently used similarity transformations on system matrices. A complete list of implemented model reduction and auxiliary routines is given in [21].

3 Integration in user-friendly environments

One of the main objectives of the NICONET project is to provide, additionally to standardized Fortran codes, high quality software embedded into user-friendly environments for *computer aided control system design* (CACSD). 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/C++ or Fortran. In 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 were necessary to the MATLAB *mex*-functions to adapt them to `Scilab`. With appropriate wrappers, the MATLAB *mex*-functions can also serve as basis to implement external function interfaces to other similar environments (e.g., MATRIX$_X$).

For the integration in MATLAB, one important aspect for implementing *mex*-functions was to keep their total size as small as possible. Since the standardized model reduction programs in SLICOT share many routines from BLAS, LAPACK and SLICOT, it was decided to implement a single function covering all model reduction functionality provided in SLICOT. The *mex*-function for model reduction is called `sysred` and provides a flexible interface to practically all functional features provided by the model reduction routines for reduction of stable/unstable linear systems using the B&T, SPA and HNA methods in conjunction with stable coprime factorization and stable/unstable spectral decomposition.

To provide a convenient interface to work with control objects defined in the Control Toolbox, several easy-to-use higher level model reduction functions have been additionally implemented explicitly addressing some of available features. A completely similar interface has been implemented for `Scilab` too[3].

[3] done by F. Delebeque

4 Testing and performance comparisons

Extensive testing of the implemented software has been performed using several benchmark problems [21]. Simple, well behaving models have been used to check the correct installation of the software. In what follows we only present test results where we compare the speed of our software with carefully implemented MATLAB *m*-functions from the HTOOLS Toolbox [20].

In the Table 1, we present timing results for randomly generated stable systems of orders up to 512 comparing, in case of square-root B&T method, the efficiency of the *mex*-function `sysred`, the *m*-functions `sqrmr` from HTOOLS and `balreal` from the Control Toolbox [8]. Note that for system orders above 32, `balreal` systematically failed issuing the message `"System must be reachable"`. The results in Table 1 have been obtained on a Pentium II 400 Hz Personal Computer running under Windows NT 4.0. The *mex*-function `sysred` has been produced using Digital/Compaq Visual Fortran V 5.1.

Order	Times [sec]		
	`sysred`	`sqrmr`	`balreal`
16	0.003	0.17	0.04
32	0.01	0.5	0.17
64	0.11	2.14	*
128	0.78	10.55	*
256	6.12	63.75	*
512	76.23	478.69	*

Table 1: Timing results for `sysred`, `sqrmr` and `balreal`.

This table illustrates not only the numerical robustness of structure exploiting algorithms, but also the significant speed-up obtained using *mex*-function based software (up to one order of magnitude) allowing to solve relatively large order dense problems on a desktop PC.

5 Testing on industrial examples

In [21] we report test results on three industrial models exhibiting some challenging features like poor scaling, lack of minimality, or presence of unstable modes. Here, we present some results for three linearized models of a gasifier at 0%, 50% and 100% loads. These models have been obtained by starting from a nonlinear model, developed by GEC ALSTHOM to serve as a benchmark problem for simulation and robust control. Some analysis results on the 100% load model are discussed in [10], where numerical difficulties have been encountered by using MATLAB model reduction tools, but also the numerical tools in Mathematica.

The gasifier models have order 25 and are non-minimal. The real cause of numerical difficulties appear to be the poor scaling of the models. The norms of state matrices for the three models are about 10^9, but after appropriate scaling, all norms can be reduced below 100. Such a preliminary scaling is an implicit feature of `sysred` and raised no special numerical challenges to the SLICOT model reduction codes. The computed 10 smallest Hankel singular values of the 100% load model are very accurate

$$\sigma_{16-25} = \{0.64046, 1.0852 \cdot 10^{-4}, 0, 0, 0, 0, 0, 0, 0, 0, \}$$

and indicate that the order of a minimal realization is 17. The computed 16 order reduced models can be practically not distinguished from the original models on basis of time or frequency responses. Approximations of orders 6, 8 and 12 have been also computed. The 12 order models represent very good approximations of the original models and can serve as basis for a robust controller design. A comparison of approximations for elements $g_{35}(s)$ of the corresponding TFMs is shown in Figure 1.

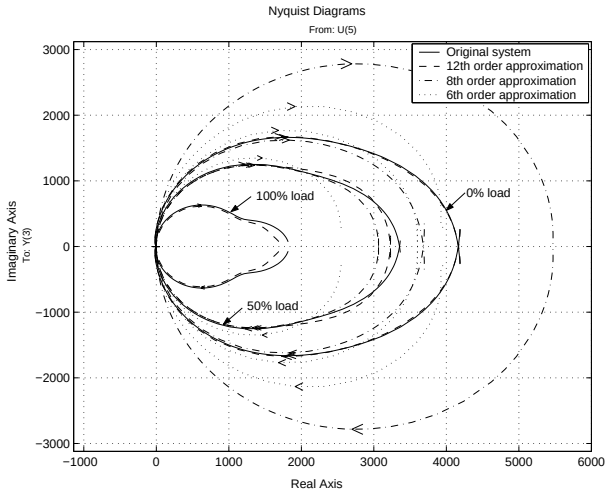

Figure 1: Frequency responses for $g_{35}(s)$ elements.

6 Summary of results and perspectives

The model reduction tools of SLICOT consists of a functionally reach collection of standardized, comprehensively documented and fully tested Fortran routines implementing rigorously selected methods for order reduction of continuous-/discrete-time, stable/unstable linear time-invariant systems. The final model reduction package consists of 9 user-callable routines and 3 supporting routines. All these routines are thoroughly documented. The documentation is automatically generated from the comments in the preamble of each routine. 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.

Besides standardized Fortran routines, the SLICOT model reduction tools include interface software to two popular user-friendly CACSD environments: MATLAB and `Scilab`. A special *mex*-function `sysred` has been implemented as Fortran interface to MATLAB to provide access to all facilities available in the SLICOT routines. This *mex*-function also served to prepare the interface software for `Scilab`. Additionally, 9 easy-to-use *m*-functions have been implemented. They fully exploit the advanced object oriented facilities available both in MATLAB Control Toolbox as well as in `Scilab` to manipulate control objects. Standard help facilities for the *mex*-function and *m*-functions are available both for MATLAB and `Scilab`.

Two main directions are envisaged to continue the efforts to develop reliable numerical model reduction software for SLICOT. The first direction focuses on the reduction of very high order systems using special implementations exploiting parallel architecture machines. The second direction continues the efforts to develop model reduction software for relative error methods and frequency weighted problems, with the main objective to have a powerful collection of tools for controller reduction. This software will complement the H_2/H_∞ software developed recently for SLICOT.

References

[1] 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, Third Edition*. SIAM, Philadelphia, 1999.

[2] 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 (Ed.), *Applied and Computational Control, Signals and Circuits*, vol. 1, pp. 499–539, Birkhäuser, 1999.

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

[4] H. Elmquist. Object-oriented modeling and automatic formula manipulation in Dymola. Scandinavian Simulation Society SIMS'93, Konsberg, Norway, 1993.

[5] K. Glover. All optimal Hankel-norm approximations of linear multivariable systems and their L^∞-error bounds. *Int. J. Control*, 39:1115–1193, 1974.

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

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

[8] MATLAB. *Control System Toolbox 4.2*. The MathWorks Inc., Natick, MA, 1998.

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

[10] N. Munro. Control system analysis and design using Mathematica. *Proc. CDC'98, Tampa, FL*, pp. 3681–3685, 1998.

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

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

[13] A. Varga. Balancing-free square-root algorithm for computing singular perturbation approximations. *Proc. CDC'91, Brighton, UK*, pp. 1062–1065, 1991.

[14] A. Varga. Efficient minimal realization procedure based on balancing. In A. El Moudni, P. Borne, and S. G. Tzafestas, (Eds.), *Prepr. of IMACS Symp. on Modelling and Control of Technological Systems*, vol. 2, pp. 42–47, 1991.

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

[16] A. Varga. Coprime factors model reduction based on accuracy enhancing techniques. *Systems Analysis Modelling and Simulation*, 11:303–311, 1993.

[17] A. Varga. A Schur method for computing coprime factorizations with inner denominators and applications in model reduction. *Proc. ACC'93, San Francisco, CA*, pp. 2130–2131, 1993.

[18] A. Varga. Numerical methods and software tools for model reduction. In I. Troch and F. Breitenecker, (Eds.), *Proc. of 1st MATHMOD Conf., Viena*, vol. 2, pp. 226–230, 1994.

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

[20] A. Varga and V. Ionescu. HTOOLS - A Toolbox for solving H_∞ and H_2 synthesis problems. *Proc. of IFAC/IMACS Symp. on Computer Aided Design of Control Systems, Swansea, UK*, pp. 508–511, July 1991.

[21] A. Varga. Model Reduction Software in the SLICOT Library. In B. N. Datta (Ed.), *Applied and Computational Control, Signals and Circuits*, vol. 2, 2000 (to appear); available also as NICONET Report NIC1999-8, `ftp://wgs.esat.kuleuven.ac.be/pub/WGS/REPORTS/NIC1999-8.ps.Z`