EXPLICIT FORMULAS FOR AN EFFICIENT IMPLEMENTATION OF THE FREQUENCY–WEIGHTED MODEL REDUCTION APPROACH

Andras Varga
DLR – Institut für Robotik und Systemdynamik
Oberpfaffenhofen, D–8031 Wessling, F.R.G.

Keywords: model reduction, frequency–weighted approximation, numerical methods

1 Introduction

The *frequency–weighted model reduction* (FWMR) approach is primarily intended to enhance the approximation properties of existing powerful model reduction methods in specified frequency ranges. As pointed out in [1], controller reductions in feedback loops can be also viewed as a special class of FWMR problems. Several FWMR methodologies have been proposed in the literature. Enns [2] proposed a FWMR approach based on the *balanced truncation approximation* (BTA) method of Moore [3], but until now no L_∞–norm bound for the corresponding approximation error is known. An alternative FWMR methodology was proposed by Latham and Anderson [4] in conjunction with the *Hankel–norm approximation* (HNA) method of [5]. Upper bounds on the L_∞–norm of the approximation error for this method have been derived in [6] and [7].

This paper focuses on the computational aspects of the FWMR methodology of [4] (for more details see also [7]). Besides the solution of a standard model reduction problem, the underlying computations also consists of several "simple" manipulations of the transfer function matrices of the given system and of the given frequency–weighting functions, implying systems conjugations, inversions and cascading. A brute force implementation of this methodology (as that available in a recently developed MATLAB Toolbox [8]) is highly inefficient with respect to both storage requirements and computational effort.

In this paper we derive explicit formulas for implementing efficiently the FWMR approach for both continuous–time and discrete–time systems. New state–space formulas are derived for computing various stable projections. The use of these formulas circumvents the need to form explicitly conjugated or inverse systems, or to manipulate higher order systems resulting from systems cascading. A detailed implementable algorithm is presented for continuous–time systems and a similar procedure is discussed for discrete–time systems. The formulas derived for discrete–time systems are seemingly new, allowing the implementation of the FWMR methodology with the same numerical performances as in the continuous–time case. Robust implementations of the presented computational approaches are available in a recently developed software library for model reduction [9].

2 Frequency–weighted model reduction

Let $G(\lambda)$ be a p×m *transfer–function matrix* (TFM) of a stable system, where λ is either the complex variable s appearing in the Laplace–transform for a continuous–time system or the complex variable z in the Z–transform for a discrete–time system. Let (A,B,C,D) be an equivalent n–th order state–space representation of G. We denote the given system as G = (A,B,C,D) which expresses the identity $G(\lambda) = C(\lambda I - A)^{-1}B + D$. Let $W_1(\lambda)$ and $W_2(\lambda)$ be p×p and m×m stable, invertible and minimum–phase TFMs, representing respectively the output and the input frequency weights, and let (A_1,B_1,C_1,D_1) and (A_2,B_2,C_2,D_2) be corresponding state–space realizations of orders n_1 and n_2, respectively. W_i^* $(i = 1, 2)$ represents the conjugate of the system W_i having the TFM $W_i^T(-s)$ for a continuous–time system or $W_i^T(1/z)$ for a discrete–time system.

The following FWMR problem is considered in this paper: Given the stable system G with order n and the frequency–weights W_1 and W_2, determine an r–th order stable approximation G_r of G which minimizes $\|W_1^{*-1}(G-G_r)W_2^{*-1}\|_\infty$, the L_∞–norm of the frequency–weighted approximation error. A general procedure to compute an approximate solution of this problem is the following one [7]:

1. Compute G_1, the n–th order stable projection of $W_1^{*-1}GW_2^{*-1}$.
2. Determine G_{1r}, an r–th order approximation of G_1 by using a model reduction method suitable for stable systems (for example the HNA or the BTA method).
3. Compute G_r, the r–th order stable projection of $W_1^*G_{1r}W_2^*$.

In the next sections we derive explicit state-space formulas to compute the stable projections at steps 1 and 3 of this procedure for both continuous-time and discrete–time systems. The derived formulas allow an efficient implementation of the FWMR approach.

3 Projection formulas

Let us assume that $W_1 = (A_1,B_1,C_1,D_1)$ and $W_2 = (A_2,B_2,C_2,D_2)$ are such that A has no common eigenvalues with either A_1 or A_2. We can easily construct the system

$$W_1GW_2 := (A_w,B_w,C_w,D_w)$$

where

$$A_w = \begin{bmatrix} A_1 & B_1C & B_1DC_2 \\ 0 & A & BC_2 \\ 0 & 0 & A_2 \end{bmatrix}, \quad B_w = \begin{bmatrix} B_1DD_2 \\ BD_2 \\ B_2 \end{bmatrix}$$

$$C_w = [\, C_1 \;\; D_1C \;\; D_1DC_2 \,] \;, \quad D_w = D_1DD_2 \,.$$

Let T and T^{-1} be the transformation matrix and its inverse respectively, defined by

$$T = \begin{bmatrix} -X & I & XY \\ I & 0 & -Y \\ 0 & 0 & I \end{bmatrix}, \quad T^{-1} = \begin{bmatrix} 0 & I & Y \\ I & X & 0 \\ 0 & 0 & I \end{bmatrix}$$

where X and Y satisfy the Sylvester equations

$$-A_1X + XA + B_1C = 0$$

$$-AY + YA_2 + BC_2 = 0.$$

It is easy to verify that

$$T^{-1}A_wT = \begin{bmatrix} A & 0 & 0 \\ 0 & A_1 & * \\ 0 & 0 & A_2 \end{bmatrix}, \quad T^{-1}B_w = \begin{bmatrix} BD_2+YB_2 \\ * \\ B_2 \end{bmatrix}$$

$$C_wT = [\, D_1C - C_1X \;\; C_1 \;\; * \,] \,,$$

where the stars (*) denote matrices whose expressions are not important for what follows. The form of the matrix $T^{-1}A_wT$ allows to decompose W_1GW_2 additively as

$$W_1GW_2 = G_1 + G_2$$

where G_1 and G_2 are given by

$$G_1 = (A,\, BD_2+YB_2,\, D_1C - C_1X,\, D_1DD_2)$$

$$G_2 = \left(\begin{bmatrix} A_1 & * \\ 0 & A_2 \end{bmatrix}, \begin{bmatrix} * \\ B_2 \end{bmatrix}, [\, C_1 \; * \,], 0 \right) .$$

The expressions for the state–space matrices of G_1 are used in the next section to deduce the expressions for the stable projections of $W_1^*GW_2^*$ and $W_1^{*-1}GW_2^{*-1}$.

4 Continuous–time FWMR procedure

By taking into account that in the continuous–time case

$$W_i^* = (-A_i^T, C_i^T, -B_i^T, D_i^T) \quad (i = 1,2)$$

we obtain the following state–space representation of the stable projection of $W_1^*GW_2^*$

$$[W_1^*GW_2^*]\_ = (A, BD_2^T + YC_2^T, D_1^TC + B_1^TX, D_1^TDD_2^T)$$

$$A_1^TX + XA + C_1^TC = 0$$

$$AY + YA_2^T + BB_2^T = 0.$$

In order to compute the stable projection of $W_1^{*-1}GW_2^{*-1}$, the same equations are used with W_1 and W_2 replaced by their inverses W_1^{-1} and W_2^{-1}, respectively, where for i = 1, 2

$$W_i^{-1} = (A_i - B_iD_i^{-1}C_i,\, -B_iD_i^{-1},\, D_i^{-1}C_i,\, D_i^{-1})$$

Note that W_1 and W_2 are stable and minimum–phase TFMs and therefore W_1^*, W_2^*, W_1^{*-1}, W_2^{*-1} are completely unstable. Thus the conditions for the solvability of the corresponding Sylvester equations are fulfilled.

For the solution of the Sylvester equations either the Schur method [10] or the Hessenberg–Schur method [11] can be used. Tbe Schur method is based on a preliminary reduction of the state matrices A, A_1^T and A_2^T to an upper quasi–triangular form, the so–called *real Schur form* (RSF), with the help of orthogonal similarity transformations. By using the Hessenberg–Schur method, it is possible to solve the Sylvester equations by reducing only A_1^T and A_2^T to the RSF and the matrix A (usually of higher order) to the Hessenberg form. Thus, the Hessenberg–Schur method is usually more efficient than the Schur method. Both algorithms are numerically stable and can be safely used in solving Sylvester equations. For the purpose of the FWMR used in conjunction with a balancing related model reduction method (for instance the BTA or the HNA methods), the following procedure seems to be the most efficient with respect to both the storage requirements and the necessary computational effort.

FWMR Algorithm for continuous–time systems.

1. Compute an orthogonal transformation matrix Q to reduce A to the RSF and put
   $$A \longleftarrow Q^T AQ,\; B \longleftarrow Q^T B,\; C \longleftarrow CQ.$$
2. For $i = 1,2$ compute the orthogonal matrices Q_i to reduce $(A_i - B_i D_i^{-1} C_i)^T$ to the RSF and put
   $$\bar{A}_i = Q_i^T (A_i - B_i D_i^{-1} C_i)^T Q_i\,,\; \bar{B}_i = Q_i^T(-B_i D_i^{-1})$$
   $$\bar{C}_i = (D_i^{-1} C_i) Q_i\,,\; \bar{D}_i = D_i^{-1}\,.$$
3. By using the Schur method, solve the Sylvester eqnations
   $$\bar{A}_1 \bar{X} + \bar{X} A + \bar{C}_1^T C = 0$$
   $$A\bar{Y} + \bar{Y}\bar{A}_2 + B\bar{B}_2^T = 0.$$
4. Compute an r–th order approximation $G_{1r} = (A_r, B_{1r}, C_{1r}, D_{1r})$ of the system
   $$G_1 = (A,\, B\bar{D}_2^T + \bar{Y}\bar{C}_2^T,\, \bar{D}_1^T C + \bar{B}_1^T \bar{X},\, \bar{D}_1^T D \bar{D}_2^T).$$
5. Compute an orthogonal transformation matrix Z to reduce A_r to the Hessenberg form and put
   $$A_r \longleftarrow Z^T A_r Z,\; B_{1r} \longleftarrow Z^T B_{1r},\; C_{1r} \longleftarrow C_{1r} Z.$$
6. For $i = 1,2$ compute the orthogonal matrices Z_i to reduce A_i^T to the RSF and put
   $$\tilde{A}_i = Z_i^T A_i^T Z_i\,,\; \tilde{B}_i = Z_i^T B_i\,,\; \tilde{C}_i = C_i Q_i\,.$$
7. By using the Hessenberg–Schur method, solve the Sylvester equations
   $$\tilde{A}_1 \tilde{X} + \tilde{X} A_r + \tilde{C}_1^T C_{1r} = 0$$
   $$A_r \tilde{Y} + \tilde{Y}\tilde{A}_2 + B_{1r}\tilde{B}_2^T = 0.$$
8. Compute the r–th order reduced model
   $$G_r = (A_r, B_{1r}\tilde{D}_2^T + \tilde{Y}\tilde{C}_2^T, \tilde{D}_1^T C_{1r} + \tilde{B}_1^T \tilde{X}, \tilde{D}_1^T D_{1r} \tilde{D}_2^T).$$

Remarks. 1. At step 3 of the algorithm it is advantageous to use the Schur method to solve the Sylvester equations because the reduction of A to the RSF is usually necessary also at step 4 if a balancing related method is used for model reduction. In this way, the costly reduction of A to the RSF is no more necessary at step 4, A being already in this form from the previous step. At step 7, the Hessenberg–Schur method is the recommendable choice to be used to solve the Sylvester equations because of its increased computational efficiency.

2. The above algorithm circumvents completely the need to form explicitly either $W_1^{*-1} G W_2^{*-1}$ or $W_1^* G_{1r} W_2^*$ in order to compute their stable projections from their additive spectral decompositions. This computation would require the reduction of the corresponding state matrices to RSF and the reordering of the diagonal blocks of the RSF in order to separate the stable and unstable eigenvalues.

3. It is clear from the above remarks that the use of explicit formulas contributes decisively to enhancing the numerical performances of the overall FWMR procedure. Moreover, because the handling of output and input weights can be done separately at steps 2, 3, 6, 7 and 8 of the algorithm, a very efficient modular implementation of the algorithm is possible. Such implementation is desirable because frequently only one of the weights is present in the FWMR problem and in such a case a shorter algorithmic path can be performed. The above algorithm served as basis to implement the tools for solving FWMR problems in the recently developed model reduction package MODRED [9].

5 Discrete–time FWMR procedure

In the discrete–time case we assume additionally that W_1 and W_2 have no poles and zeros in the origin. This assumption allows us to form explicit standard state–space representations for the discrete–time conjugate systems. By using the following state–space representations of the conjugate systems ($i = 1,2$)

$$W_i^* = (A_i^{-T}, A_i^{-T} C_i^T, -B_i^T A_i^{-T}, D_i^T - B_i^T A_i^{-T} C_i^T)$$

we obtain after straightforward formula manipulations the following state–space representation of the stable projection of $W_1^* G W_2^*$

$$[W_1^* G W_2^*]_- = (A, \bar{B}, \bar{C}, \bar{D})$$

where

$$\bar{B} = BD_2^T - BB_2^T A_2^{-T} C_2^T + YA_2^{-T} C_2^T$$
$$\bar{C} = D_1^T C - B_1^T A_1^{-T} C_1^T C + B_1^T A_1^{-T} X$$
$$\bar{D} = (D_1^T - B_1^T A_1^{-T} C_1^T) D (D_2^T - B_2^T A_2^{-T} C_2^T)$$

and X and Y satisfy the Sylvester equations

$$-A_1^{-T} X + XA + A_1^{-T} C_1^T C = 0$$
$$-AY + YA_2^{-T} - BB_2^T A_2^{-T} = 0.$$

By replacing $A_1^{-T} X$ and YA_2^{-T} obtained from these equations in the expressions of $\bar{C}$ and $\bar{B}$ respectively, we obtain the equivalent simpler expressions

$$\bar{B} = BD_2^T + AYC_2^T$$
$$\bar{C} = D_1^T C + B_1^T XA.$$

and thus

$$[W_1^* G W_2^*]_- = (A,\ BD_2^T + AYC_2^T,\ D_1^T C + B_1^T XA,\ \bar{D})$$

It is easy to observe that X and Y satisfy the equivalent discrete–type Sylvester equations

$$A_1^T XA + C_1^T C = X$$
$$AYA_2^T + BB_2^T = Y.$$

In order to compute the stable projection of $W_1^{*-1} G W_2^{*-1}$, the same equations are used with W_1 and W_2 replaced by their inverses.

The procedure for the FWMR of discrete–time systems is analogous with the FWMR algorithm presented in the previous section for continuous–time systems, with obvious modifications in the expressions of matrices and in solving discrete–type Sylvester equations instead continuous–type ones. For solving discrete–type Sylvester equations, a Schur method was proposed in [12]. A Hessenberg-Schur variant of this algorithm can also be easily devised along the lines of techniques described in [11]. The overall numerical performances (storage requirements, number of operations, roundoff errors) of the discrete–time procedure are basically the same as for its continuous–time counterpart.

Remark. In deducing the expressions for the state space matrices of the stable projection of $W_1^* G W_2^*$ we used the assumptions that both A_1 and A_2 are non–singular. These assumptions are no more necessary if the matrix $\bar{D}$ is not included in the expression of the stable projection $[W_1^* G W_2^*]_-$ or if the original system G is strictly proper ($D = 0$). The expressions of $\bar{B}$ and $\bar{C}$ are still valid even if A_1 or A_2 are singular and the matrices X and Y satisfy the same discrete–type Sylvester equations. The deduction of the respective formulas can be done by working with equivalent descriptor systems representations of the conjugated systems W_1^* and W_2^*.

6 Conclusions

Explicit formulas for computing stable projections in the FWMR approach have been derived. The new formulas allow an efficient and modular implementation of the FWMR procedure for both continuous-time and discrete–time systems. Robust implementations of the proposed computational approach are already available in a recently developed model reduction Fortran library called MODRED [9]. This library is primarily intended to be used for solving large order model reduction problems on high performance computers. It is worth to mention that MODRED is one of the first available applications libraries based on the new, *de facto standard*, linear algebra package LAPACK [13].

References

[1] B.D.O.Anderson and Y.Liu, Controller reduction: Concepts and approaches, *IEEE Trans. Autom. Contr.*, **34**, pp.802–812, (1989).

[2] D.Enns, Model reduction for control systems design, Ph.D. dissertation, Dept. Aeronaut. Astronaut., Stanford Univ., Stanford, 1984.

[3] B.C.Moore, Principal components analysis in linear systems: controllability, observability and model reduction, *IEEE Trans. Autom. Contr.*, **26**, pp.17–32, (1981).

[4] G.A.Latham and B.D.O.Anderson, Frequency–weighted optimal Hankel–norm approximation of stable transfer functions, *Syst. Contr. Lett.*, **5**, pp.229–236, (1985).

[5] K.Glover, All optimal Hankel–norm approximations of linear multivariable systems and their L∞–error bounds, *Int. J. Contr.*, **39**, pp.1115–1193, (1984).

[6] B.D.O.Anderson, Weighted Hankel–norm approximation: Calculation of bounds, *Syst. Contr. Lett.*, **7**, pp.247–255, (1986).

[7] K.Glover, D.J.N.Limebeer and Y.S.Hung, A structured approximation problem with applications to frequency weighted model reduction, *IEEE Trans. Autom. Contr.*, **37**, pp.447–465, (1992).

[8] G.J.Balas, J.C.Doyle, K.Glover, A.Packard and R.Smith, μ–Analysis and Synthesis Toolbox, The MathWorks, (1991).

[9] A.Varga, MODRED — A LAPACK based Fortran library for model reduction, *User's Guide*, Dept. Mech. Eng., Univ. Bochum, F.R.G., (1992).

[10] R.H.Bartels and G.W.Stewart, Solution of the equation AX+XB=C, *Commun. ACM*, **15**, pp.820–826, (1972).

[11] G.H.Golub, S.Nash and C.F.Van Loan, A Hessenberg–Schur method for the problem AX+XB=C, *IEEE Trans. Autom. Contr.*, **24**, pp.909–913, (1979).

[12] V.Sima, Comparison of some algorithms for solving the Lyapunov type equations, *Rev. Roum. Scie. Techn. – Electrotechn. et Energ.*, **25**, pp.235–246, (1980).

[13] E.Anderson, Z.Bai, C.Bishof, J.Demmel, J.J.Dongarra, J.Du Croz, A.Greenbaum, S.Hammarling, A.McKenney and D.Sorensen, LAPACK: A portable linear algebra library for high–performance computers, *Proceedings of Supercomputer '90*, IEEE, pp.2–11, (1990).