Computational Techniques Based on the Block-Diagonal Form for Solving Large Systems Modeling Problems

A. Varga
German Aerospace Research Establishment (DLR)
Institute for Robotics and System Dynamics
Oberpfaffenhofen, 82230 Wessling, Germany.

Abstract

The reduction of the state-matrix of a linear time-invariant state-space model to a block-diagonal form by using a state coordinate transformation is equivalent with an additive decomposition of the corresponding transfer-function matrix. Computationally involved and large storage demanding algorithms for solving several systems modeling problems can be conveniently reformulated such that they perform exclusively on the low order subsystems corresponding to the individual terms of suitable additive decompositions. Important reductions of both the computational effort and required memory usually by using the reformulated algorithms and thus, their applicability can be extended to handle higher order systems. The paper presents several algorithms suitable to perform efficiently on additively decomposed systems. The effectiveness of these algorithms for solving large order systems modeling problems relies on a reliable numerical algorithm to compute the block-diagonal form of a matrix.

1. Introduction

The use of condensed forms of systems matrices in algorithms for analysis and design of control systems was also considered by other authors. An overview of computational techniqes based on the use of condensed forms obtainable by employing orthogonal system similarity transformations is presented in [10]. The use of the Hessenberg form to enhance the efficiency of several algorithms for systems analysis and modeling is discussed in [12]. The wide usage of the Schur form in control computations is illustrated by various algorithms presented in the book [6].

In this paper we address the usefulness of the *block-diagonal form* (BDF) of a square matrix in solving efficiently computational problems arising in systems modeling applications. Any square real matrix A of order n can be reduced by a similarity transformation of the form

$$\tilde{A} = T^{-1}AT, \tag{1}$$

where T is a real transformation matrix, to a BDF

$$\tilde{A} = \mathrm{diag}(A_1, \ldots, A_k) \tag{2}$$

in which each matrix A_i is square of order n_i. In applications it is generally desirable that the diagonal blocks have orders as small as possible. Theoretically the blocks cannot be smaller than the blocks in the Jordan canonical form of A. However, the computation of the Jordan form is generally a difficult and often, very ill-conditioned problem. From numerical point of view a sound approach is to determine blocks with the lowest achievable dimensions by using a well-conditioned transformation matrix. In this case, the application of transformation to other matrices will not cause significant accuracy losses. A very efficient and numerically reliable algorithm for computing the BDF was proposed by Bavely and Stewart [2]. Besides simplicity and economy, the main advantage of this algorithm is its ability to keep under control the condition number of the transformation matrix. Therefore, with this algorithm, the BDF can be computed with a prescribed accuracy loss. The computational effort to compute the BDF is approximately $11n^3 - 15n^3$ operations and all computations can be performed by using only $2n^2$ storage locations.

Consider now a state-space model of the form

$$\begin{array}{rcl} \dot{x}(t) & = & Ax(t) + Bu(t) \\ y(t) & = & Cx(t) \end{array} \tag{3}$$

(denoted also $\{A, B, C\}$), where x is the n-dimensional state vector, u is the m-dimensional input vector and y is the p-dimensional output vector. The reduction of the state matrix A to a BDF by a system similarity transformation of the form

$$\tilde{A} = T^{-1}AT,\ \tilde{B} = T^{-1}B,\ \tilde{C} = CT \tag{4}$$

where

$$\left.\begin{array}{rcl} \tilde{A} & = & \mathrm{diag}(A_1, \ldots, A_k) \\ \tilde{B} & = & \mathrm{row}(B_1, \ldots, B_k) \\ \tilde{C} & = & \mathrm{col}(C_1, \ldots, C_k), \end{array}\right\} \tag{5}$$

is equivalent with an additive decomposition of the corresponding *transfer-function matrix* (TFM)

$$G(s) = C(sI - A)^{-1}B \tag{6}$$

in the form

$$G(s) = \sum_{i=1}^{k} G_i(s), \tag{7}$$

where $G_i(s) = C_i(sI - A_i)^{-1}B_i$, for $i = 1, \ldots, k$.

The paper presents several computational algorithms for manipulating state-space models which can be reformulated in terms of the additively decomposed TFM. The algorithms considered in the paper are for: 1) discretization of continuous systems [11]; 2) computation of minimal realizations and model reduction by using balancing and balancing related techniques [13]; 3) evaluation of the TFM corresponding to a state-space description [15]; and 4) computation of frequency responses [4].

One of the advantages of using such an approach is that usually the main computational effort in the reformulated algorithms is the computation of the BDF, all subsequent efforts being negligible due to the low orders of the subsystems $\{A_i, B_i, C_i\}$. Another advantage is that, because of performing the computations on low order subsystems, usually no additional storage (excepting $2n^2$ storage locations) is necessary to perform the computations. In contrast, when the mentioned algorithms (excepting the last one) perform on full order models, the necessary storage is about $4n^2 - 5n^2$ storage locations. Therefore, by using the new approach, problems of higher orders can be solved in the same storage area. Moreover, experimental tests indicate that the accuracy of the new methods is better (sometimes much better) than the accuracy of the original methods. Finally, because the submatrices of the BDF resulting from the algorithm [2] are in an upper real Schur form and the eigenvalues of each block usually form a cluster of nearby eigenvalues, special algorithmic enhancements can be further devised which contribute additionally to an increased effectiveness of these methods.

2. Computation of block-diagonal form

We review very shortly the main features of the block-diagonalization algorithm of [2]. A preliminary step in this algorithm is to reduce the matrix A to a quasi-triangular, the so-called *real Schur form* (RSF), in which the diagonal blocks are of orders at most two. The 2×2 blocks have exclusively complex conjugate pairs of eigenvalues. The ordering of blocks is generally arbitrary and can be changed by using appropriate orthogonal reordering techniques [3]. We assume that A is already in a RSF.

The subsequent block-diagonalization is based on partitions of the matrix A in the form

$$A = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}. \tag{8}$$

An attempt to annihilate the off-diagonal block A_{12} is made by using a transformation T of the form

$$T = \begin{bmatrix} I & X \\ 0 & I \end{bmatrix}, \tag{9}$$

where the identity matrices are of same orders as A_{11} and A_{22}. If there exists X satisfying the Sylvester equation

$$A_{11}X - XA_{22} = -A_{12}, \tag{10}$$

then, by using T from (9), we obtain

$$T^{-1}AT = \mathrm{diag}(A_{11}, A_{22}). \tag{11}$$

A standard technique to solve (10) is the numerically stable algorithm proposed in [1].

Notice that

$$T^{-1} = \begin{bmatrix} I & -X \\ 0 & I \end{bmatrix} \tag{12}$$

and therefore

$$\mathrm{cond}(T)_F = \|T\|_F \|T\|^{-1}\|_F = n + \|X\|_F^2. \tag{13}$$

Clearly T is ill-conditioned when X has a large norm.

Initially, A_{11} in (8) is an 1×1 or 2×2 block depending on the dimension of the leading diagonal block. An attempt is made first to annihilate A_{12} with a well-conditioned T. If all elements of X are sufficiently small, then T is accepted and the algorithm proceeds further by deflating A_{11} from A and redefining a new A as A_{22}. If however, at least one element of X exceeds in magnitude a given bound, then the attempt to annihilate A_{12} is abandoned and a new, larger block A_{11} is formed by incorporating in it a new 1×1 or 2×2 of A_{22} whose eigenvalues are the nearest to the mean of eigenvalues of A_{11}. Standard block-interchange algorithms are available to reorder the RSF of A for this purpose [3].

The final result of the Bavely-Stewart algorithm is a BDF in which each diagonal block is in RSF and has nearly equal eigenvalues. The orders of the blocks are usually related to the bound specified by the user on the condition number of the reducing matrices of the form (9). This bound restricts only the elements of X and therefore the algorithm cannot guarantee that the final transformation is well-conditioned. Nonetheless, in most of cases there is a very little tendency toward excessive growth in the condition of the transformation matrix.

The number of *floating-point operations* (flops) for reducing a matrix in RSF to the BDF ranges from $n^3/2$, when each eigenvalue is real and can be deflated, to $n^4/12$, in the highly improbable extreme case, when none of them can be deflated. Several implementation tricks can improve the efficiency of the overall algorithms such that finally we can consider $15n^3$ flops as a good maximal figure for the reduction of a dense matrix to the BDF.

3. Discretization of continuous models

Let us consider the sampled-data system

$$\begin{array}{rcl} x(t+1) & = & A_d x(t) + B_d u(t) \\ y(t) & = & Cx(t) \end{array} \tag{14}$$

resulting from the discretization of the continuous system (3) with a sampling period τ. The standard method [11] to compute the matrices A_d and C_d consists in evaluating them from the matrix identity

$$\exp\left(\left[\begin{array}{cc} A & B \\ 0 & 0 \end{array}\right]\tau\right) = \left[\begin{array}{cc} A_d & B_d \\ 0 & I \end{array}\right]. \qquad (15)$$

One of the most reliable methods to evaluate the exponential of a matrix is by using diagonal Padé approximations combined with the scaling and squaring technique. A detailed algorithm was proposed by Ward in [16]. The method performs about $(q+j+\frac{1}{3})(n+m)^3$ flops, where q is the order of the Padé approximation and j is the number of performed squaring operations. The required memory is about $4(n+m)^2$ storage locations. In order to employ as small values of q and j as possible, Ward proposed two additional norm reducing techniques: a diagonal translation of eigenvalues combined with a row-columns balancing with diagonal similarity transformations matrices.

The usage of the BDF for evaluating matrix exponentials is discussed in [5]. The corresponding computational effort to compute (15) is about $17(n+m)^3$ flops. Here we discuss a somewhat cheaper approach which relies on the additive decomposition form of the system matrices in (5). Let the matrices of the system be already as in (5). Then, for each pair (A_i, B_i) we can compute the corresponding pair (A_{id}, B_{id}) from

$$\exp\left(\left[\begin{array}{cc} A_i & B_i \\ 0 & 0 \end{array}\right]\tau\right) = \left[\begin{array}{cc} A_{id} & B_{id} \\ 0 & I \end{array}\right] \qquad (16)$$

and then A_d, B_d and C_d can be assembled as

$$\left.\begin{array}{rcl} A_d & = & \mathrm{diag}(A_{1d}, \ldots, A_{kd}) \\ B_d & = & \mathrm{row}(B_{1d}, \ldots, B_{kd}) \\ C_d & = & \mathrm{col}(C_{1d}, \ldots, C_{kd}). \end{array}\right\} \qquad (17)$$

If A_i is non-singular and well-conditioned (a very cheap test because A_i is already in RSF), then a more efficient way to compute A_{id} and B_{id} is by using the formulas

$$A_{id} = \exp(A_i\tau),\ B_{id} = A_{id}^{-1}(A_{id} - I)B_i. \qquad (18)$$

The diagonal translation and balancing techniques proposed by Ward to be used in conjunction with the Padé approximation method are usually very effective when applied to the low order RSF matrices A_i. This is because often A_i has nearly equal eigenvalues and after a diagonal translation with γ_i, the mean of eigenvalues of A_i, the matrix $A_i - \gamma_i I$, is nearly nilpotent and its powers vanish rapidly. Additional diagonal balancing can be used to reduce the norm almost arbitrarily, avoiding thus completely the need for scaling and squaring operations. The costs to evaluate A_{id} and B_{id} by using either (16) or (18) is therefore negligible.

The described approach requires about $15n^3$ flops, which represent roughly the cost of computing the BDF. Therefore, the method compares favorably with most of other methods. The storage needs are $2n^2$ locations, which is about the half of the storage required by other approaches. The method is suitable for the discretization of systems with multiple sampling periods, because the evaluation of the matrices of the sampled-data systems corresponding to different values of τ is very cheap if the system is already in form (5).

4. Minimal realization and model reduction

Consider the following minimal realization problem: for a given non-minimal system $\{A, B, C\}$, determine a minimal realization $\{\hat{A}, \hat{B}, \hat{C}\}$ such that the two systems have the same TFMs. There exist two main classes of computational methods for solving this problem.

A first class of *minimal realization algorithms* (MRAs), based on the use of orthogonal staircase canonical forms, determines the matrices of a minimal realization by successively removing the uncontrollable and unobservable parts of the given non-minimal system. The complete MRA, using for example the orthogonal staircase algorithm proposed in [9], is numerically stable, performs no more than $\frac{10}{3}n^3$ flops and needs no additional storage. Although very efficient, this algorithm is very sensitive to the choice of zero-tolerances used in the rank determinations during the reduction to staircase forms. A single erroneous rank decision leads to the failure of the whole MRA.

The methods in the second class are projection methods based on balancing [8] or related to balancing techniques [7]. Both algorithms determine directly two projection matrices T and L (satisfying $LT = I$) such that the matrices of the minimal realization can be computed as

$$\hat{A} = LAT,\ \hat{B} = LB,\ \hat{C} = CT \qquad (19)$$

The algorithm of [8] determines the projection matrices by using the so-called square-root accuracy enhancing approach, by working exclusively with the Cholesky factors of gramians. The resulting minimal realization is balanced. Accuracy losses can be however induced in the resulting minimal realization when the projection matrices are ill-conditioned. The second method [7] uses a balancing-free approach to determine T and L and produces always well-conditioned projection matrices. However the method relies on forming the product of the gramians and usually is less accurate than the square-root method. An improved version of both methods, obtained by combining in a single algorithm both the square-root and the balancing-free approaches, was proposed recently in [13] and has very satisfactory accuracy properties.

The MRAs related to balancing have the very attractive property to remove simultaneously the uncontrollable and unobservable parts by using a single rank decision based on a singular value decomposition.

They are thus reliable in determining correctly the order of the minimal realization. The direct application of these methods to large systems raises however problems. First, the number of operations is considerable. A rough estimation is about $40n^3$ flops (with an order of magnitude more than for the staircase methods). Second, the memory requirement is also notable, about $5n^2$ storage locations are necessary. Therefore, the applicability of balancing related methods to high order systems is problematic.

The additive decomposition of the system in the form (5) can be used to compute minimal realizations provided any two diagonal blocks have no common eigenvalues. This condition can be easily fulfilled by incorporating in a single, larger block, all individual blocks produced by the block-diagonalization algorithm of [2] which have nearly equal eigenvalues (complex eigenvalues enter always in pairs in the newly formed blocks). This approach seems to be more economical than a preliminary segregation of the eigenvalues in clusters by reordering of the RSF of A. The following straightforward algorithm can be used to compute minimal realizations:

1. Reduce the system $\{A, B, C\}$ to the additively decomposed form (5), where $\lambda(A_i) \cap \lambda(A_j) \neq \phi$ for $i \neq j$.

2. For each subsystem $\{A_i, B_i, C_i\}$ determine a minimal realization $\{\hat{A}_i, \hat{B}_i, \hat{C}_i\}$ by using the algorithm from [13].

3. Construct the minimal system $\{\hat{A}, \hat{B}, \hat{C}\}$ as

$$\begin{aligned} \hat{A} &= \mathrm{diag}(\hat{A}_1, \ldots, \hat{A}_k), \\ \hat{B} &= \mathrm{row}(\hat{B}_1, \ldots, \hat{B}_k), \\ \hat{C} &= \mathrm{col}(\hat{C}_1, \ldots, \hat{C}_k). \end{aligned}$$

If the orders of diagonal blocks A_i are small, then the operations performed at step 2 are negligible in comparison with the cost to reduce A to the BDF. The additional storage is also substantially reduced.

This algorithm is particularly well suited for computing minimal state-space realizations of TFMs. In this case, the initial non-minimal realization can be directly constructed in a BDF. If necessary, the grouping of eigenvalues of A can be performed by simple row and column permutations performed on matrices A, B and C. The number of performed operation is comparable with that performed by a staircase algorithm and in many cases can be even smaller.

The above algorithm can also be used for model reduction by approximating the diagonal subsystems with lower order approximations or by removing completely those subsystems which have negligible contribution to the TFM of the system.

It is worth mentioning an interesting aspect revealed by tests performed with the block-diagonal approach used in conjunction with seven other MRAs related to balancing [14]. Although some of particular methods performed very inaccurate when used on the full order system, the results obtained in conjunction with the block-diagonalization technique were always very accurate. Due to this very satisfactory numerical behavior, the usage of this method is to be recommended even for low order systems.

5. Evaluation of transfer-function matrices

For the computation of the TFM corresponding to a state-space model $\{A, B, C\}$, there exist several algorithms. A numerically reliable method based exclusively on orthogonal transformations is the *poles-zeros method* proposed in [15]. This method determines the elements of the TFM by evaluating for each of them the corresponding poles, zeros and gain. The method is generally applicable regardless the original system is minimal or not. The resulting elements are in canceled, minimal forms.

The applicability of this algorithm to high order systems encounters the same difficulties as discussed previously. The need to compute for each element eigenvalues (representing poles or zeros) can lead in extreme (although highly improbable) cases to more than $18n^3$ flops per element to be performed (compare with $15n^3$ flops necessary to compute the BDF). The required memory is at most $4n^2 + 4nmp$ storage locations, which is also substantial.

The usage of the block-diagonal decomposition of system matrices in the form (5) could be advantageous also in this case. Each term $G_i(s)$ of the decomposition (7) can be evaluated separately by applying the poles-zeros method to the corresponding subsystem $\{A_i, B_i, C_i\}$. Then, the TFM $G(s)$ can be computed from its partial fraction decomposition by summing the individual terms. This summation reduces to simple polynomial operations (without any need to compute greatest common divisors) if we ensure that the diagonal blocks have no common eigenvalues. This can be done similarly as for the algorithm presented in the previous section.

The block-diagonalization approach to evaluate the TFM of a state-space model is thus a very effective alternative to existing methods. The algorithm performs roughly the usual $15n^3$ flops to compute the BDF of A. The storage requirements are also substantially reduced to less than $n^2 + max(n^2, 2nmp)$ storage locations.

6. Computation of frequency responses

An efficient method to evaluate the frequency response

$$G(j\omega) = C(j\omega I - A)^{-1}B \tag{20}$$

for many values of the frequency ω was proposed by Laub [4]. The method is based on a preliminary reduction of the state-matrix A to a Hessenberg form by means of an orthogonal similarity transformation and

to modify B and C accordingly. After that, the evaluation of $G(j\omega)$ for a given frequency value can be performed relatively cheaply with about $\frac{1}{2}(m+1)n^2 + pmn$ complex flops. If the number of frequency values is large, a more efficient approach (suggested in [10]) is to determine first the TFM $G(s)$ given by (6) (see the method in the previous section) and then to evaluate $G(j\omega)$ from $G(s)$ directly (only simple polynomial evaluations are necessary).

Alternatively, the decomposition (5) with A in a BDF can be used to evaluate (21) instead the representation with A in the Hessenberg form. For a given frequency value ω, $G(j\omega)$ can be computed as

$$G(j\omega) = \sum_{i=1}^{k} G_i(j\omega), \tag{21}$$

where $G_i(j\omega) = C_i(j\omega I - A_i)^{-1}B_i$. Recall that A_i is already in a RSF and thus the evaluation of $G_i(j\omega)$ is computationally very cheap because of usually very low order of A_i. The possibility to use more condensed forms (the diagonal form) was mentioned in [4], but rejected because of potential accuracy problems due to ill-conditioned eigenstructures. Our experience indicates that even when the computed eigenvalues appear to be inaccurate due to ill-conditioning, the accuracy of resulting $G_i(j\omega)$ is still comparable with the accuracy of the Hessenberg method.

7. Conclusions

The paper presented several useful applications of the BDF of a square matrix in solving efficiently some systems modeling problems. The block-diagonalization approach allows to extend the ranges of applicability of several existing methods to higher order systems by reducing simultaneously the associated computational burden and storage requirements. The effectiveness of the new approach is guaranteed by the availability of a numerically reliable algorithm to compute the BDF with a prescribed accuracy loss. For most practical problems, the reformulated algorithms have the additional benefits of providing more accurate results than those resulted by direct application of the original methods. The usage of the BDF for efficient on-line implementation of compensators and filters is also an area of potential applications of this condensed form.

References

[1] R. H. Bartels and G. W. Stewart. Algorithm 432: Solution of the matrix equation AX+XB=C. *Comm. ACM*, 15:820–826, 1972.

[2] C. Bavely and G. W. Stewart. An algorithm for computing reducing subspaces by block diagonalization. *SIAM J. Numer. Anal.*, 16:359–367, 1979.

[3] J. J. Dongarra, S. Hammarling, and J. Wilkinson. Numerical considerations in computing invariant subspaces. *SIAM J. Matrix Anal. Appl.*, 13:145–161, 1992.

[4] A. J. Laub. Efficient multivariable frequency response computations. *IEEE Trans. Autom. Control*, AC-26:407–408, 1981.

[5] C. B. Moler and C. F. Van Loan. Nineteen dubious ways to compute the exponential of a matrix. *SIAM Review*, 20:801–836, 1978.

[6] P. H. Petkov, N. D. Christov, and M. M. Konstantinov. *Computational Methods for Linear Control Systems*. Prentice Hall Int., 1991.

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

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

[9] P. Van Dooren. The generalized eigenstructure problem in linear systems theory. *IEEE Trans. Autom. Control*, AC-26:111–129, 1981.

[10] P. Van Dooren and M. Verhaegen. *On the use of unitary state-space transformations*, volume 47 of *Special Issue of Contemporary Mathematics in Linear Algebra and Its Role in Systems Theory*. Amer. Math. Soc., Providence, R.I., 1985.

[11] C. F. Van Loan. Computing integrals involving matrix exponentials. *IEEE Trans. Autom. Control*, AC-24:395–404, 1978.

[12] C. F. Van Loan. Using the Hessenberg decomposition in control theory. In D. C. Sorenson and R. J. Wets, editors, *Mathematical Programming Study*, volume 18, pages 102–111. North Holland, 1982.

[13] 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.

[14] A. Varga. Minimal realization procedures based on balancing and related techniques. In F. Pichler and R. Moreno Diaz, editors, *Computer Aided Systems Theory - EUROCAST'91*, volume 585 of *Lect. Notes Comp. Scie.*, pages 733–761. Springer Verlag, Berlin, 1992.

[15] A. Varga and V. Sima. A numerically stable algorithm for transfer-function matrix evaluation. *Int. J. Control*, 33:1123–1133, 1981.

[16] R. C. Ward. Numerical computation of the matrix exponential with accuracy estimate. *SIAM J. Numer. Anal.*, 14:600–610, 1977.