Solving Sylvester equations for Model Reduction : Slicot vs. Matlab [1]

Chris Denruyter[2]

September 2001

[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, Kasteelpark Arenberg 10, 3001-Leuven-Heverlee, BELGIUM. This report is available by anonymous ftp from `wgs.esat.kuleuven.ac.be/pub/WGS/REPORTS/SLWN2000-2.ps.Z`

[2]Université Catholique de Louvain, CESAME, B-1348 Louvain-la-Neuve, Belgium

Abstract In this report, we compare two Sylvester equation solvers: the Matlab function `lyap` and the Slicot function `slsylv`. An algorithm designed for model reduction and based on the resolution of a Sylvester equation is presented. In this context, timing results show the superiority of the Slicot based m.file `slsylv`.

Contents

1 Introduction

Model reduction consists in finding an approximate model for a high order system. The higher the order of the original system, the more complex the solution of optimisation or design problems for such a system. It is therefore useful to build a *low order* approximation of such a system, which can then be used for solving the optimisation or design problem in an approximate sense. An important aspect of model reduction is of course that the computational complexity of constructing the reduced order model should not be larger than that of solving the subsequent optimisation or design problem.

Below, we first state which models will be used to represent dynamical systems. Two model reduction techniques are then described in the next section. The hybrid technique we then focus on is based on the resolution of Sylvester equations. An iterative algorithm implementing this technique is presented, and we show how the Slicot routine `slsylv` can be used to improve speed and accuracy, in comparison to the Matlab routine `lyap`.

2 State-space representations

A dynamical system provides a response $y(t)$ from a given input function $u(t)$. The link between the input $u(t)$ and the output $y(t)$ depends on the parameters of the system. We restrict ourselves here to a single input/single output (SISO) system Σ that can be described by a state-space realization

$$\begin{aligned} \dot{\mathbf{x}}(t) &= A\mathbf{x}(t) + bu(t) \\ y(t) &= c\mathbf{x}(t) \end{aligned} \tag{1}$$

with $\mathbf{x}(t)$ the so-called *s*tate of the system and

$$A \in \mathbb{R}^{n\times n}, \quad b \in \mathbb{R}^{n\times 1}, \quad c \in \mathbb{R}^{1\times n}.$$

If this system is *m*inimal (i.e. controllable and observable), we say that the model is of *order* n. We will assume that the system representation is minimal and that the system is also stable, i.e. that all the eigenvalues of A lie in the left half plane. Any coordinate transformation on the state $\mathbf{x}$ ($\tilde{\mathbf{x}} = T^{-1}\mathbf{x}$) yields an equivalent representation of the system Σ. The new system matrices $(\tilde{A}, \tilde{b}, \tilde{c})$ are then given by $(T^{-1}AT, T^{-1}b, cT)$.

If we take the Laplace transform of each equation in the state-space representation, and eliminate the state variable $\mathbf{x}$ in these equations, we obtain the following input/output relation:

$$\mathcal{Y}(s) = H(s)\mathcal{U}(s), \qquad \text{where} \quad \left\{ \begin{array}{l} H(s) = c(sI - A)^{-1}b, \\ \mathcal{Y} \text{ is the Laplace transform of } y, \\ \mathcal{U} \text{ is the Laplace transform of } u. \end{array} \right.$$

$H(s)$ is called the *transfer function* of the system, and for a SISO system it is easy to see that $H(s)$ is a *rational function.* Furthermore, this function is *proper*, meaning that

$$\lim_{|s|\to\infty} H(s) = \lim_{|s|\to\infty} c(sI - A)^{-1}b = \lim_{|s|\to\infty} \frac{c\,\mathrm{adj}(sI - A)b}{\det(sI - A)} = 0.$$

3 Model Reduction

State-space models are very useful to *simulate* the behaviour of a dynamical system, but also to solve particular optimisation problems (such as finite horizon optimal control), design problems (such as robust controller design) or prediction problems (such as Kalman filter prediction). When the state dimension

n is very large, the solution of these problems become computationally very expensive : their complexity typically grows as $O(n^3)$. It is therefore desirable to have a smaller model at hand, say of degree k, whose behaviour is nevertheless very close to that of the original model. If $k << n$, the computational complexity of any design problem will be much lower and the reduced order design can then be used as an approximation for the higher dimensional problem.

Let us denote the reduced model of order $k << n$ corresponding to (1) as follows :

$$\begin{aligned} \dot{\hat{\mathbf{x}}}(t) &= \hat{A}\hat{\mathbf{x}}(t) + \hat{b}u(t) \\ \hat{y}(t) &= \hat{c}\hat{\mathbf{x}}(t) \end{aligned} \tag{2}$$

with

$$\hat{A} \in \mathbb{R}^{k\times k}, \quad \hat{b} \in \mathbb{R}^{k\times 1}, \quad \hat{c} \in \mathbb{R}^{1\times k}.$$

The transfer function of the reduced system $\hat{\Sigma}$ is given by

$$\hat{H} = \hat{c}(sI - \hat{A})^{-1}\hat{b}.$$

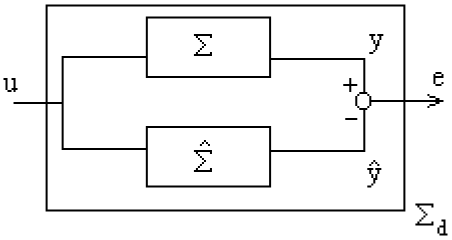

Figure 1: The Error System $H_d(s) = H(s) - \hat{H}(s)$.

Both models obviously give different output signals ($y(t)$ or $\hat{y}(t)$) for a same input signal $u(t)$ (figure 1). But we would like to impose that the relative approximation error $\|y - \hat{y}\|/\|u\|$ is small for any input function u. We will also expect the reduced model to be stable, because the former was. Finally, there is no need to seek a reduced model if this cannot be obtained at reasonable computational cost. Below we describe two approaches for obtaining such reduced order models : *interpolation* (also known as *moment matching*) and *balanced truncation*. For a survey of these methods, we refer to [2].

3.1 Interpolation

Single Interpolation Point ([7], [10], [5]) A first approach is to interpolate the transfer function about a given point s_1 up to a certain order. We thus need to find a reduced order model with transfer function $\hat{H}$, such that it has the same first $2k$ coefficients – called *moments* – as in the expansion of H about s_1:

$$H(s) = m_0 + m_1(s - s_1) + m_2(s - s_1)^2 + m_3(s - s_1)^3 + \dots$$

$$\hat{H}(s) = \hat{m}_0 + \hat{m}_1(s - s_1) + \hat{m}_2(s - s_1)^2 + \hat{m}_3(s - s_1)^3 + \dots$$

such that

$$m_i = \hat{m}_i \quad \text{for} \quad i = 0, \dots, 2k - 1.$$

One shows that there always exists a kth order model that matches $2k$ moments of such an expansion.

Different Interpolation Points ([9], [6]) A more elaborate approach is to match the first moments of $\hat{H}(s)$ with those of $H(s)$ in *several* expansion points. For two points s_1 and s_2 this would amount to :

$$\begin{aligned} H(s) &= m_{0,s_1} + m_{1,s_1}(s-s_1) + m_{2,s_1}(s-s_1)^2 + \dots \\ &= m_{0,s_2} + m_{1,s_2}(s-s_2) + m_{2,s_2}(s-s_2)^2 + \dots \end{aligned}$$

$$\begin{aligned} \hat{H}(s) &= \hat{m}_{0,s_1} + \hat{m}_{1,s_1}(s-s_1) + \hat{m}_{2,s_1}(s-s_1)^2 + \dots \\ &= \hat{m}_{0,s_2} + \hat{m}_{1,s_2}(s-s_2) + \hat{m}_{2,s_2}(s-s_2)^2 + \dots \end{aligned}$$

with $2k$ equalities

$$\begin{aligned} \hat{m}_{i,s_1} &= m_{i,s_1} \quad \text{for} \quad i = 0, \dots, k_1 - 1, \\ \hat{m}_{i,s_2} &= m_{i,s_2} \quad \text{for} \quad i = 0, \dots, 2k - 1 - k_1. \end{aligned}$$

This can be extended to several points $\{s_f, f = 1, \dots, F\}$ as well.

Matching the Coefficients ([9]) It is possible to match these coefficients by projecting the system data (A, b, c) in a particular subspace of $\mathbb{R}^n$. The projected system has lower order, but the quantities $\{m_{i,s_f} | i = 1, \dots, I_f\}$ are preserved for selected frequencies s_f. The projection subspace is a union of so-called *Krylov subspaces*. A Krylov subspace of order j associated with the matrices $A^{n\times n}$ and $b^{n\times 1}$ is defined as

$$\mathcal{K}_j(A, b) \doteq \text{span}\left\{b, Ab, \dots, A^{j-1}b\right\}.$$

If $V^{n\times k}$ is an isometry (i.e. an $n \times k$ matrix satisfying $V^T V = I_k$) and its columns span the subspace of projection (i.e. $\text{Im}(V) = \mathcal{K}_k(A, b)$), the projected system matrices are then

$$\begin{aligned} \hat{A} &= V^T A V, \\ \hat{b} &= V^T b, \\ \hat{c} &= cV. \end{aligned} \tag{3}$$

The above Krylov subspace is designed to obtain a reduced model with an interpolation "about infinity": $m_{0,\infty} = 0$ and $m_{i,\infty} = cA^{i-1}b$ for $i = 1, 2, \dots$

Depending on which points of interpolation we choose, the form of the Krylov subspace will vary. The general expression for the m_{i,s_f}'s is

$$m_{i,s_f} = -c(A - s_f I)^{-(i+1)} b,$$

and the corresponding Krylov subspaces of order k is

$$\mathcal{K}_k((A - s_f I)^{-1}, (A - s_f I)^{-1} b).$$

The following theorem gives the condition needed on the space of projection $\text{Im}(V)$ to achieve interpolation.

Theorem 1 ([9]) If

$$\bigcup_{f=1}^{F} \mathcal{K}_{I_f}((A - s_f I)^{-1}, (A - s_f I)^{-1} b) \subseteq \text{Im}(V)$$

then

$$c(A - s_f I)^{-i_f} b = \hat{c}(\hat{A} - s_f I)^{-i_f} \hat{b} \qquad \forall\, i_f = 1, \dots, I_f, \quad \forall\, f = 1, \dots, F.$$

Let $\hat{H}(s)$ denote the transfer function corresponding to projected model data as in (3). The theorem states that the expansion of $\hat{H}(s)$ about an s_f will have its first I_f coefficients matching those of $H(s)$.

An advantage of this method is that the computation of a matrix V can be achieved at low cost. Still, we do not know how to choose the interpolation points s_f, nor how many coefficients to match at each of these points. Finally, there is no guarantee for the reduced model to be stable.

3.2 Balanced Truncation

The quantities of interest in balanced truncation are the *Gramian matrices*: they measure the strength of the link between the state vector $\mathbf{x}$ and the input u, on one hand, and between the state vector $\mathbf{x}$ and the output y on the other hand.

The Gramian matrices can be defined as the solution of Lyapunov equations. The *controllability* Gramian P solves the controllability Lyapunov equation

$$AP + PA^T + bb^T = 0,$$

and the *observability* Gramian Q solves the observability Lyapunov equation

$$A^TQ + QA + c^Tc = 0.$$

In this method, one seeks a coordinate system where each component $\tilde{x}_i$ of the state is equally coupled to the input and to the output (*principal axis balancing*, see [12], [1], [4]). The weakest components, which have nearly no influence in the input/output relation, are discarded. This leads to a reduced system of order $k << n$ because there are usually many weak components, see [3]. The process amounts to three basic steps:

1. Compute the solutions P, Q to the Lyapunov equations.
2. From P and Q, compute the balancing coordinate transformation T_b. In the new coordinate system P and Q are equal and diagonal.
3. In the balanced state space representation $(T_b^{-1}AT_b, T_b^{-1}b, cT_b)$, discard all the weak components.

If we partition the matrices such that the weak components are in the (2,2) block of A, the reduced order model of

$$\left(\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, \begin{bmatrix} c_1 & c_2 \end{bmatrix} \right)$$

becomes

$$(A_{11}, b_1, c_1).$$

It can be shown that this method preserves the stability of the system and provides an upper bound on the error of approximation $\|y - \hat{y}\|/\|u\|$. Unfortunately, the first step consists in solving two Lyapunov equations of order n, which is computationally expensive ($O(n^3)$).

4 Sylvester Method

4.1 Projected Lyapunov Equation

The balanced truncation method requires the resolution of two Lyapunov equations of order n. However, only a small part of the computed information will be retained since typically $k << n$.

In order to explain the basic idea, we focus only on the controllability equation, involving the matrix $P^{n\times n}$. We should avoid computing the whole matrix P but rather directly compute the essential information in P. Multiplying the Lyapunov equation on the right with an orthogonal matrix $V^{n\times k}$ yields

$$\begin{gathered}(AP + PA^T + bb^T = 0) * V \\ \Downarrow \\ AZ + PA^TV + b\hat{b}^T = 0,\end{gathered}$$

with $Z = PV$ and $\hat{b} = V^Tb$. A Sylvester equation in $Z^{n\times k}$ is then obtained by projecting A^TV on V:

$$AZ + P\,VV^T\,A^TV + b\hat{b}^T = 0$$

or, equivalently,

$$AZ + Z\hat{A}^T + b\hat{b}^T = 0, \tag{4}$$

with $\hat{A}^{k\times k} = V^TAV$. The equation (4) is the core of the algorithm presented below and it is important to solve it quickly and accurately.

4.2 An Interpolation Interpretation

The resolution of a Sylvester equation can be seen as an interpolation method. We show below that the range of Z, solution of (4), is a union of different Krylov subspaces. From Theorem 1, it follows that projecting the system on $\mathrm{Im}(Z)$ corresponds to an interpolation about different points.

Under a change of coordinates on V, we can always rewrite the equation (4) with a projected $\hat{A}^T = V^TA^TV$ in its Jordan form. This does not change the range of V $(\mathrm{Im}(VT_k^{-1}) = \mathrm{Im}(V))$. One shows that this is a union of different Krylov subspaces (see [8]) :

$$AZ + Z\begin{bmatrix} \hat{A}_{j_1} & 0 & \cdots & 0 \\ 0 & \hat{A}_{j_2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \hat{A}_{j_l}\end{bmatrix} + b\begin{bmatrix}\hat{b}_1 & \hat{b}_2 & \cdots & \hat{b}_k\end{bmatrix} = 0 \qquad (\sum_{i=1}^{l} j_i = k)$$

each $\hat{A}_{j_i}$ being a Jordan block associated with $\lambda_i(\hat{A})$. Since all the Jordan blocks are decoupled, we may focus only on one of them, say the first one, which has order j_1. For the first column z_1 of Z, we have the equation

$$(A + \lambda_1 I)z_1 + b\hat{b}_1 = 0,$$

and for any other z_i with $2 \le i \le j_1$, we have

$$(A + \lambda_1 I)z_i + z_{i-1} + b\hat{b}_i = 0.$$

Hence the space spanned by the first j_1 columns of Z is given by

$$\mathcal{K}_{j_1}(\tilde{A}^{-1}, \tilde{A}^{-1}b) = \mathrm{span}\left\{\ -\tilde{A}^{-1}b\hat{b}_1,\ \tilde{A}^{-2}b\hat{b}_1 - \tilde{A}^{-1}b\hat{b}_2,\ \cdots\ \right\}, \qquad (\tilde{A} = A + \lambda_1 I)$$

as long as $\hat{b}_1$ is nonzero and $\tilde{A}$ is invertible.

If we build an orthogonal matrix $V^{n\times k}$ from Z, and use it to project the system data (A, b, c), the solution of a Sylvester equation can be seen as another technique of interpolation, where the set of interpolation points

$$\{-\lambda_i(\hat{A}_v)\,|\,i = 1, \ldots, l\}$$

is determined by $\hat{A}$, as long as it has been initialized. More details can be found in [8].

4.3 The Algorithm

The algorithm resulting from the above considerations on the controllability Lyapunov equation is the following.

% in: model data (A, b), reduced order k
% out: reduced model $(\hat{A}, \hat{b})$
Initialize $\hat{A}^{k\times k}, \hat{b}^{k\times 1}$
while (not converged)
  Solve $AZ + Z\hat{A}^T + b\hat{b}^T = 0$;
  $[V, R] = qr(Z)$;[1]
  Project A and b : $\hat{A} = V^T AV, \quad \hat{b} = V^T b$;
end

The algorithm is initialized by choosing a first set of interpolation points (the eigenvalues of $\hat{A}$). The vector $\hat{b}$ should be such that Z has full rank k. Another possible initialization is to build first a matrix $V^{n\times k}$, then to obtain $(\hat{A}, \hat{b})$ as in equation (3). The resolution of the Sylvester equation

$$AZ + Z\hat{A}^T + b\hat{b}^T = 0$$

will determine the next subspace of projection $\mathrm{Im}(Z)$. We can interpret each iteration as a new interpolation with the eigenvalues of $\hat{A}$ as interpolation points.

$$\ldots \to V_i \to (\hat{A}, \hat{b})_i \to V_{i+1} \to (\hat{A}, \hat{b})_{i+1} \to \ldots$$

It is observed that, after convergence, the spectrum of $\hat{A}$ is typically clustered near the imaginary axis, and opposite to a subset of the eigenvalues of A (see [4]). An idea to initialize $\hat{A}$ would be to choose its k eigenvalues near this axis, in the stable part of the complex plane.

4.4 Convergence

The Sylvester equation (4) was obtained by projecting A on the range of V (the projector is $\Pi = \Pi^2 = VV^T$) which is only an approximate equation. The correct equation is (using $PV = Z$) :

$$AZ + Z\hat{A}_v^T + b\hat{b}^T = -\underbrace{PV^\perp}_{\approx 0} V^{\perp T} A^T V \approx 0.$$

Because the eigenvalues of P are supposed to vanish quickly (see [3]), the residual in this equation is nearly zero. Hence, from one step to the other in the iterative algorithm, we should approximately follow a power method scheme since in fact we approximately perform the computation :

do
$Z := PV$;
$V := qr(Z)$
loop

As a consequence, V should approximately converge towards the dominant eigenspace of P.

[1] The qr method builds, from a full rank matrix $Z^{n\times k}$, an orthogonal matrix $V^{n\times k}$ and an upper triangular change of coordinates $R^{k\times k}$ such that $Z = VR$.

5 Numerical Experiments

Each step of the iterative algorithm consists essentially in the resolution of a Sylvester equation. We compare the performance of the Sylvester solvers `slsylv` (Slicot routine) and `lyap` (Matlab routine).

Heat equation The models which are used describe the evolution of the temperature for a 1D problem. The temperature is measured at n equidistant points on a bar, the temperature x_i in each point being a component of the state vector $\mathbf{x}$. We choose $n = 100$ and $n = 200$, and only the 21 middle states of the bar are under direct control of the function u. Both ends are at zero temperature. The dynamics of the problem are then described by the equation

$$\dot{\mathbf{x}} = A\mathbf{x} + Bu$$

with

$$A = -\tfrac{(n-1)^2}{100}\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix}, \quad B = \begin{array}{c} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \end{array} \begin{array}{l} \\ \\ \\ \leftarrow (\frac{n}{2}-10)\text{th. state} \\ \\ \leftarrow (\frac{n}{2}+10)\text{th. state} \\ \\ \\ \\ \end{array}$$

We compared the computing time and accuracy of the Slicot based m.file `slsylv` with those of the m.file `lyap` from the control toolbox of Matlab. As a measure of the relative error, we took the quantity $\|AZ + Z\hat{A}^T + b\hat{b}^T\|/\|Z\|$.

100 STATES	Elapsed time	$\frac{\|AZ+Z\hat{A}^T+b\hat{b}^T\|}{\|Z\|}$
Slicot	0.0500	$3.2478e-014$
Matlab	0.6100	$2.5716e-013$

Table 1: Results on Heat equation model with 100 states

200 STATES	Elapsed time	$\frac{\|AZ+Z\hat{A}^T+b\hat{b}^T\|}{\|Z\|}$
Slicot	0.6100	$8.9211e-014$
Matlab	5.1600	$1.4764e-012$

Table 2: Results on Heat equation model with 200 states

The computation time for the Slicot routine is about ten times smaller than for the Matlab routine. In an iterative algorithm, this is very important, since these computations are often performed and represent the main part of the work.

We also obtained a slightly higher accuracy with the Slicot routine. When orthogonal projectors are being constructed, this may be of a lesser importance since a small perturbation on the entries of the matrix V should not have a dramatic influence on the reduced system data $(\hat{A}, \hat{b}, \hat{c})$.

But the conceptual algorithm presented here was a simplified version of the general algorithm combining the two projected Lyapunov equations. In that more general case, the constructed projection will be *oblique* and a small perturbation on the data might have a stronger influence on the resulting reduced model (see [4]).

6 Concluding Remarks

Model reduction is necessary when one wants to design a controller for a high order model or to solve related optimisation problems.

We discussed an iterative algorithm based on the Sylvester equation, which achieves model reduction. Two possibilities to implement this algorithm were taken in consideration: one using the m.file `slsylv`, the other using the m.file `lyap`. The first was shown to be faster and more accurate than the latter.

The algorithm discussed in this paper was highly simplified : there are other reduction techniques based on the same idea, but which take into account the possible lack of symmetry between the input and the output vectors (b and c) of the system. However, these variants involve at least one Sylvester equation ([11], [4]). The observations made in this report therefore still apply to these cases : because the algorithms are iterative, and the most time consuming step is the solution of the Sylvester equation, it is important to have a fast Sylvester equation solver.

References

[1] A. Antoulas, *Lectures on the approximation of large-scale dynamical systems*, SIAM (2001).

[2] A. Antoulas, D. Sorensen and S. Gugercin, *A survey of model reduction methods for large-scale systems*, in "Structured Matrices in Operator Theory, Numerical Analysis, Control, Signal and Image Processing," Contemporary Mathematics, edited by V. Olshevsky, AMS publications (2001).

[3] A. Antoulas, D. Sorensen and Y. Zhou, *On the decay rate of Hankel singular values and related issues,* Rice University, Texas (2001). Available at http://www.caam.rice.edu/caam/tr01.html#TR01-09 .

[4] C. Denruyter, Model reduction for large-scale dynamical systems, final work, promotor: P. Van Dooren, Université Catholique de Louvain (2001). chrisden@hotmail.com

[5] K. Gallivan, E. Grimme and P. Van Dooren, *Asymptotic Waveform Evaluation via a Restarted Lanczos Method,* Appl. Math. Letters, **7**:75-80, 1994.

[6] K. Gallivan, E. Grimme and P. Van Dooren, *A Rational Lanczos Algorithm for Model Reduction,* Numerical Algorithms, **12**:33-63, 1996.

[7] K. Gallivan, E. Grimme, D. Sorensen and P. Van Dooren, *On some modifications of the Lanczos algorithm and the relation with Padé approximations,* Mathematical Research Series, **7**:87-116, 1996.

[8] K. Gallivan, A. Vandendorpe and P. Van Dooren, *Sylvester equations and projection based model reduction,* Internal Report CESAME, Université Catholique de Louvain (2001).

[9] E. Grimme, *Krylov Projection Methods for Model Reduction,* Ph.D. Thesis, ECE Dept., University of Illinois at Urbana-Champaign (1997).

[10] E. Grimme, D. Sorensen and P. Van Dooren, *Model reduction of state space systems via an implicitly restarted IR Lanczos method,* Numerical Algorithms, **12**:1-31, 1996.

[11] D. Sorensen and A. Antoulas, *Projection method for balanced model reduction*, (2001) available at http://www.caam.rice.edu/caam/tr01.html#TR01-03

[12] P. Van Dooren, *Gramian based model reduction of large-scale dynamical systems*, Numerical Analysis 1999, Griffiths and Watson editors.