Improving the efficiency and accuracy of the MATLAB Control Toolbox using SLICOT-based gateways

Vasile Sima
Research Institute for Informatics
Bd. Mareşal Al. Averescu, Nr. 8–10
71316 Bucharest 1, Romania
vsima@u3.ici.ro

Peter Benner
Zentrum für Technomathematik
Fachbereich 3–Mathematik und Informatik
Universität Bremen
D–28334 Bremen, Germany
benner@math.uni-bremen.de

Sabine Van Huffel
Department of Electrical Engineering
Katholieke Universiteit Leuven
Kardinaal Mercierlaan 94
B–3001 Leuven–Heverlee, Belgium
Sabine.VanHuffel@esat.kuleuven.ac.be

Andras Varga
German Aerospace Center
Institute of Robotics and System Dynamics
DLR Oberpfaffenhofen
D-82234 Wessling, Germany
Andreas.Varga@dlr.de

Keywords: computer-aided control system design, multivariable systems, numerical algorithms, numerical linear algebra, state-space methods

Abstract

The paper presents performance results for some components of the new, public-domain version of the SLICOT library (Subroutine Library in Control Theory), in comparison with equivalent computations performed by some MATLAB functions included in the Control Toolbox. SLICOT incorporates the new algorithmic developments in numerical linear algebra, implemented in the state-of-the-art software packages LAPACK and BLAS. The results show that, at comparable accuracy, SLICOT routines are several times faster than MATLAB computations.

1 Introduction

MATLAB[1] [5] is an excellent tool for developing and testing new algorithmic ideas or new control analysis and synthesis methods. However, the practical experience has shown a sometimes poor performance of MATLAB in a dedicated production-quality computer-aided control system design environment (CACSD). To achieve the robustness and efficiency needed for solving possibly ill-conditioned or large-scale real-life control problems, a new public-domain version of the SLICOT library (Subroutine Library in Control Theory) has been recently developed. SLICOT incorporates the new algorithmic developments in numerical linear algebra, implemented in the state-of-the-art software packages LAPACK [1] and BLAS [3], and can thus exploit the potential of modern high-performance computer architectures. The conversion of the SLICOT library to a freely available software package offered the opportunity to improve the modularity, functionality, reliability, as well as the performance of the codes, by using calls to Level 3 BLAS and LAPACK block algorithms whenever possible, exploiting any special problem structure, etc.

The paper presents performance results (efficiency and accuracy) for some components of the new SLICOT release[2], in comparison with equivalent computations performed by some MATLAB functions included in the Control Toolbox [4]. The calculations have been performed on a SUN Ultra 2 Creator 2200 workstation under SunOS 5.5, by calling from MATLAB the SLICOT gateways produced by the NAGWare Gateway Generator [6] of the Numerical Algorithms Group (NAG). The results show that, at comparable or better accuracy, SLICOT routines are several times faster than MATLAB computations; moreover, less memory is required by SLICOT routines, because the problem structure is fully exploited. Additional performance results and comparisons can be found in [2, 8]. Reference [2] also includes a brief history and perspective of system and control software.

2 Structure preserving algorithms

Among the various reasons for the development of SLICOT library [2], a special emphasis will be put here on *structure-preserving algorithms*. The main advantage

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

[2]see http://www.win.tue.nl/wgs/slicot.html and use the link to the FTP site.

of such algorithms is that the structural properties of a problem are preserved during finite precision computations. This allows the computed result to be interpreted as the exact solution of the original problem with perturbed input data, which may otherwise not be the case. This not only increases the reliability of the returned results, but often also improves their accuracy.

Example: The *controllability* and *observability Gramians* P_c, P_o, respectively, of a stable state-space realization (A, B, C) of a continuous-time linear time-invariant system are given by the solution of the stable *Lyapunov equations*

$$\begin{aligned} AP_c + P_cA^T &= -BB^T, & (1)\\ A^TP_o + P_oA &= -C^TC. & (2) \end{aligned}$$

By Lyapunov stability theory, P_c and P_o are positive semidefinite. The nonnegative square roots of the eigenvalues of the product P_cP_o, called the *Hankel singular values*, play a fundamental role in finding balanced realizations and in model reduction. To guarantee the symmetry and semidefiniteness of the computed Gramians, the special problem structure should be taken into account. Besides this, exploiting the structure usually results in a reduction of necessary computational operations and memory requirements. No MATLAB function specialized for equations like (1) or (2) is yet available. The new SLICOT release includes the routine `SB03OD`, which solves for $X = \mathrm{op}(U)^T\mathrm{op}(U)$ either continuous-time or discrete-time Lyapunov equations,

$$\mathrm{op}(A)^TX + X\mathrm{op}(A) = -\sigma^2\mathrm{op}(M)^T\mathrm{op}(M), \qquad (3)$$

$$\mathrm{op}(A)^TX\mathrm{op}(A) - X = -\sigma^2\mathrm{op}(M)^T\mathrm{op}(M), \qquad (4)$$

where $\mathrm{op}(K) = K$ or K^T, A is square (stable, or convergent, respectively), M is rectangular, U is upper triangular, and σ is a scale factor, set less than or equal to 1 to avoid overflow in X.[3] The routine uses the same real Schur form of A for all different computations, and optionally A can be given in such a form on input.

Results for two sets of experiments will be summarized. The data for the first set have been generated as

```
A = rand(n,n) - n*eye(n);
B = rand(n,m);  C = rand(1,n);
```

and the actual solutions were unknown.

Tables 1 and 2 give comparative results using the gateway for SLICOT routine `SB03OD` and MATLAB function `lyac`, for solving the two Lyapunov equations (1) and (2).[4] It is apparent that the speed-up compared with MATLAB is about 5. Besides improved efficiency, the accuracy of the SLICOT routine is sometimes better. More important, the Hankel singular values can be obtained as the singular values of the triangular matrix VU, where U and V are the Cholesky factors computed by `SB03OD` for the solutions of the equations (3), for $\mathrm{op}(K) = K^T$ and $\mathrm{op}(K) = K$, respectively. On the contrary, MATLAB calculations need to be based on `sqrt(eig(X*Y))`, where $X = P_c$ and $Y = P_o$ are the solutions of the Lyapunov equations (1) and (2), respectively. Similar results have been obtained for the relative residuals in Y.

For one sample of the largest example, we find that two eigenvalues of the matrix XY were negative (close to the machine accuracy), and four were complex conjugate. In other words, in extreme cases, MATLAB could give negative or complex Hankel "singular values". This, of course, does not happen for well-conditioned problems, considered in the next experiment.

In the second experiment, matrix A has been defined as above, but X and Y have been constructed as positive definite, with given eigenvalues (different for X and Y), generated by `rand(n,1) + 1`. Then, B and C have been computed by the MATLAB sequence

```
BBT = -(A*X + X*A');  BBT = (BBT + BBT')/2;
CTC = -(A'*Y + Y*A);  CTC = (CTC + CTC')/2;
B = chol(BBT)';  C = chol(CTC);
```

In this way, relative errors of the solutions could be obtained, and the results are given in Table 3. The timings and relative residuals have been similar with those presented above, and there has been a good agreement between the Hankel singular values. Similar results have been obtained for the relative errors in Y.

Table 1: Comparison between `SB03OD` and MATLAB results (timings).

Dimensions			Time	
n	m	p	SB03OD	MATLAB
16	8	8	0.01	0.06
32	16	16	0.04	0.24
64	32	32	0.24	1.30
128	64	64	1.68	9.44

Table 2: Comparison between `SB03OD` and MATLAB results (relative residuals in $X = P_c$).

Dimensions			Relative residuals in X	
n	m	p	SB03OD	MATLAB
16	8	8	3.17e-14	2.97e-14
32	16	16	8.41e-14	7.23e-14
64	32	32	1.48e-13	1.82e-13
128	64	64	4.13e-13	4.21e-13

[3]Note that the chosen functionality allows to easily implement LAPACK-style condition estimators for such equations, based on `SB03OD`.

[4]The relative residuals or errors reported have been computed using the Frobenius norm.

Table 3: Comparison between SB03OD and MATLAB results (relative errors in X).

Dimensions			Relative errors in X	
n	m	p	SB03OD	MATLAB
16	16	16	2.62e-15	2.71e-15
32	32	32	3.93e-15	3.61e-15
64	64	64	5.40e-15	5.63e-15
128	128	128	6.76e-15	7.67e-15

Table 4: Comparison between SB03MD and MATLAB results (relative residuals in X).

	Time		Relative residuals in X	
n	SB03MD	MATLAB	SB03MD	MATLAB
16	0.01	0.05	3.69e-15	3.59e-15
32	0.03	0.12	3.54e-15	8.96e-15
64	0.21	0.70	3.56e-15	1.96e-14
128	1.50	5.38	1.73e-14	2.86e-14

Table 5: Comparison between SB03MD and MATLAB results (relative errors in X).

	Time		Relative errors in X	
n	SB03MD	MATLAB	SB03MD	MATLAB
16	0.01	0.03	4.01e-13	2.22e-13
32	0.03	0.11	1.63e-13	1.14e-12
64	0.22	0.70	3.48e-12	9.04e-12
128	1.73	5.49	1.51e-11	5.82e-11

Table 6: Comparison between SB02OD and MATLAB results (relative residuals in X).

	Time		Relative residuals in X	
n	SB02OD	MATLAB	SB02OD	MATLAB
16	0.07	0.37	1.83e-14	3.81e-14
32	0.30	0.86	4.99e-14	1.13e-13
64	2.80	4.01	7.21e-12	2.10e-11
128	24.00	79.96	2.44e-11	6.72e-11

3 Performance Results

Table 4 gives performance results for SLICOT general Lyapunov solver SB03MD, which solves both continuous- and discrete-time equations, and MATLAB functions lyap and dlyap. The SLICOT routine can solve equations defined by (3) and (4), but with the right-hand side replaced by σC. Moreover, an estimate of the condition number is returned. The problems solved were generated using the MATLAB statements

```
A = rand(N,N);  C = rand(N,N);  C = C + C';
```

so the solutions were not known. Therefore, only relative residuals are given in table 4. Specifically, table 4 reports results for continuous-time problems with $\mathrm{op}(A) = A$. The relative errors between the SLICOT and MATLAB computed solutions have been of the order of 10^{-13} or less. Similar results have been obtained for the transposed case, or for discrete-time equations. SLICOT residuals were usually less then MATLAB residuals in our tests. Note also that the speed-up has been even larger when compared with the previous MATLAB versions of lyap and dlyap.

Table 5 reports results for discrete-time problems with $\mathrm{op}(A) = A^T$. The problems were generated using the MATLAB statements

```
A  = rand(N,N);
X0 = rand(N,N) + N*eye(N);  X0 = X0 + X0';
C  = A*X0*A' - X0;  C = (C + C')/2;
```

so the relative errors could be computed. The relative errors between the SLICOT and MATLAB computed solutions have been of the order of 10^{-11} or less.

Table 6 gives performance results for the SLICOT Riccati solver SB02OD, which solves both continuous- and discrete-time equations, and MATLAB functions care and dare, all based on the generalized Schur vectors approach. The SLICOT solver can also compute the anti-stabilizing solutions. The problems solved were generated using the MATLAB statements

```
A = rand(N,N);    B = rand(N,M);
G = B*inv(R)*B';  G = (G + G')/2;
```

(with $m = n$), and the weighting matrices R and Q have been computed to have given random (non-negative) eigenvalues, using a gateway for the test routine DLATMS from LAPACK. Table 6 reports the timings and relative residuals for computing stabilizing solutions for continuous-time problems. The reciprocal condition number of the linear system which is solved for obtaining the stabilizing Riccati solution X was of the order 10^{-2} or larger. The relative errors between the SLICOT and MATLAB computed solutions have been of the order of 10^{-12} or less. Better efficiency, but worse accuracy in the discrete-time case with A ill-conditioned, have been obtained by calling SB02MD, which uses the Schur vectors approach (see Table 7). For discrete-time problems, SB02MD includes an option to use the inverse of the symplectic $2n \times 2n$ matrix [7], which is always faster than the standard approach. The reciprocal condition number of the linear system which is solved for obtaining X was of the order 10^{-3} or larger. The relative errors between the SLICOT and MATLAB computed solutions have been of

Table 7: Comparison between SB02MD and MATLAB results (relative residuals in X).

	Time		Relative residuals in X	
n	SB02MD	MATLAB	SB02MD	MATLAB
16	0.02	0.15	2.12e-13	1.08e-13
32	0.10	0.91	2.10e-12	2.07e-12
64	0.82	7.63	3.46e-11	5.59e-12
128	5.83	84.02	1.41e-08	1.20e-10

Table 8: Comparison of some SLICOT gateways and MATLAB functions (timings).

			Time	
Routine/function	n	m	SLICOT	MATLAB
AB01ND/ctrbf	128	16	0.32	1.44
AB01ND/ctrbf	256	1	2.48	639.57
SB01MD/place	128	1	0.55	104.00
SB04MD/lyap2	128	64	0.88	3.85

the order of 10^{-9} or less. Note that the speed-up can be larger than 10. This is important for certain on-line well-conditioned applications. The efficiency has not been so impressive for continuous-time equations, where the speed-up factors have been about four. The larger relative errors for SB02MD for $n \geq 64$ are due to the increased ill-conditioning of the matrix A, which has to be inverted by this SLICOT solver.

Finally, some timing results and relative residuals for several other SLICOT gateways, in comparison with MATLAB calculations, are reported in Tables 8 and 9.

4 Conclusions

Performance results for some basic components of SLICOT library, in comparison with similar computations performed by some MATLAB functions included in the Control Toolbox have been presented. Using the NAGWare Gateway Generator, it is possible to embed SLICOT routines into MATLAB. The so produced MATLAB functions often outperform the available functions from the MATLAB toolboxes or even built-in functions. The library is in the public-domain and its development is continuing under the framework of the European "Numerics in Control" network NICONET.

Table 9: Comparison of some SLICOT gateways and MATLAB functions (accuracy).

			Relative residuals	
Routine/function	n	m	SLICOT	MATLAB
AB01ND/ctrbf	128	16	6.46e-16	1.57e-15
AB01ND/ctrbf	256	1	1.05e-15	4.18e-15
SB01MD/place	128	1	1.32e-14	4.10e-14
SB04MD/lyap2	128	64	4.86e-13	7.38e-13

Acknowledgments

This paper presents research results of the European Community BRITE-EURAM III Thematic Networks Programme NICONET (project BRRT–CT97-5040) and the Belgian Programme on Interuniversity Poles of Attraction (IUAP Phase IV/2 & 24), initiated by the Belgian State, Prime Minister's Office for Science, Technology and Culture. S. Van Huffel is a Research Associate with the Fund for Scientific Research—Flanders.

References

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK Users' Guide.* SIAM, Philadelphia, PA, second edition, 1994.

[2] P. Benner, V. Mehrmann, V. Sima, S. Van Huffel and A. Varga. SLICOT — A Subroutine Library in Systems and Control Theory. *Applied and Computational Control, Signals, and Circuits*, 1, 1988. To appear.

[3] J. J. Dongarra, J. Du Croz, I. S. Duff, and S. Hammarling. A set of Level 3 Basic Linear Algebra Subprograms. *ACM Trans. Math. Soft.*, 16:1–17, 1990.

[4] The MathWorks, Inc., Cochituate Place, 24 Prime Park Way, Natick, MA 01760. *Control System Toolbox User's Guide*, 1996.

[5] The MathWorks, Inc., Cochituate Place, 24 Prime Park Way, Natick, MA 01760. *Using* MATLAB, 1996.

[6] The Numerical Algorithms Group, Wilkinson House, Jordan Hill Road, Oxford OX2 8DR, UK. *NAGWare Gateway Generator, Release 2.0*, 1994.

[7] V. Sima. *Algorithms for Linear-Quadratic Optimization*, volume 200 of *Pure and Applied Mathematics: A Series of Monographs and Textbooks.* Marcel Dekker, Inc., New York, 1996.

[8] V. Sima. High-performance numerical software for control systems, and subspace-based system identification. Technical Report WGS-report 97-2, The Working Group on Software: WGS, 1997.