SLICOT Working Note 2001-2

A Web Computing Environment for the SLICOT Library*

Erik Elmroth[†], Pedher Johansson[†], Bo Kågström[†], and Daniel Kreßner[‡]

December 2000

*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, K. Mercierlaan 94, 3001-Leuven-Heverlee, BELGIUM. This report is also available by anonymous ftp from `wgs.esat.kuleuven.ac.be/pub/WGS/REPORTS/SLWN2001-2.ps.Z`

†Department of Computing Science and HPC2N, Umeå University, SE-901 87 Umeå, Sweden
‡Department of Mathematics, TU Chemnitz, D-09107 Chemnitz, Germany

Abstract

A prototype web computing environment for computations related to the design and analysis of control systems using the SLICOT software library is presented. The web interface can be accessed from a standard world wide web browser with no need for additional software installations on the local machine. The environment provides user-friendly access to SLICOT routines where run-time options are specified by mouse clicks on appropriate buttons. Input data can be entered directly into the web interface by the user or uploaded from a local computer in a standard text format or in Matlab binary format. Output data is presented in the web browser window and possible to download in a number of different formats, including Matlab binary. The environment is ideal for testing the SLICOT software before performing a software installation or for performing a limited number of computations. It is also highly recommended for education as it is easy to use, and basically self-explanatory, with the users' guide integrated in the user interface.

Contents

1 Introduction

Today, many computing services are made available via the internet and the world wide web (www). Slowly, this is changing peoples way to think about computing and their expectations on computing services. For example, many people already find it quite natural to use their computers to pay bills, service bank accounts, or to receive professional brokerage service, instead of visiting their bank's local branch. Many banks provide these services through a standard www interface with no requirements for installing any specific software on the local computer. This greatly reduces the work needed the first time a person tries to use this service. It also makes it possible for one person to use the same service from virtually any computer the person may have access to. Ideally, all instructions needed are also included in the web interface.

In the world of scientific computing, the scene is quite different. A user that is familiar with the problem to be solved and knows where to find a software to solve it may still have to go through a whole series of tasks before she or he actually can try to solve the problem. Normally, the process starts by acquiring and installing the software. Then, there are manuals to read and programs to write. After compiling and linking, the actual computations can take place and an analysis of the results may be performed. In a worst case scenario, one would then find that the software was not appropriate for the problem, and one may start looking for another software and repeat the same procedure.

So, does it have to be like this? Of course not! In order to demonstrate the power of an easy-to-use interface we have developed a prototype for a web computing environment for the SLICOT library [2, 6]. SLICOT is a software library with numerical algorithms for computations in systems and control theory. Based on numerical linear algebra routines from high-performance BLAS [3, 4] and LAPACK [1] libraries, SLICOT provides methods for the design and analysis of control systems.

The use of the web computing service does not require any installation of software on the local computer, nor does it require any documentation besides the one that is integrated in the user interface. The user prerequisites are mainly to know what type of problems, i.e., what equations, that are to be solved. The software requirements are limited to a standard web browser.

This contribution provides a description of this environment that currently includes a small subset of SLICOT. The outline is as follows. Section 2 gives a brief description of the web computing environment, mainly from a user's point of view. This includes presentation of the major functionalities and a step-by-step illustration of a computing session. In Section 3, we give some details about the internal design and implementation. Conclusions and future work are presented in Section 4, mainly to illustrate that this environment is one piece of a larger concept with the common goal to give easy access to state-of-the-art software and different computing resources.

2 The SLICOT Web Computing Environment

The SLICOT web computing environment can be accessed from a standard www browser, with no need for additional software installations on the local computer. The main web page, presented in Figure 1, introduces the user to the computing services provided, which presently include a number of SLICOT routines and set of benchmark problems.

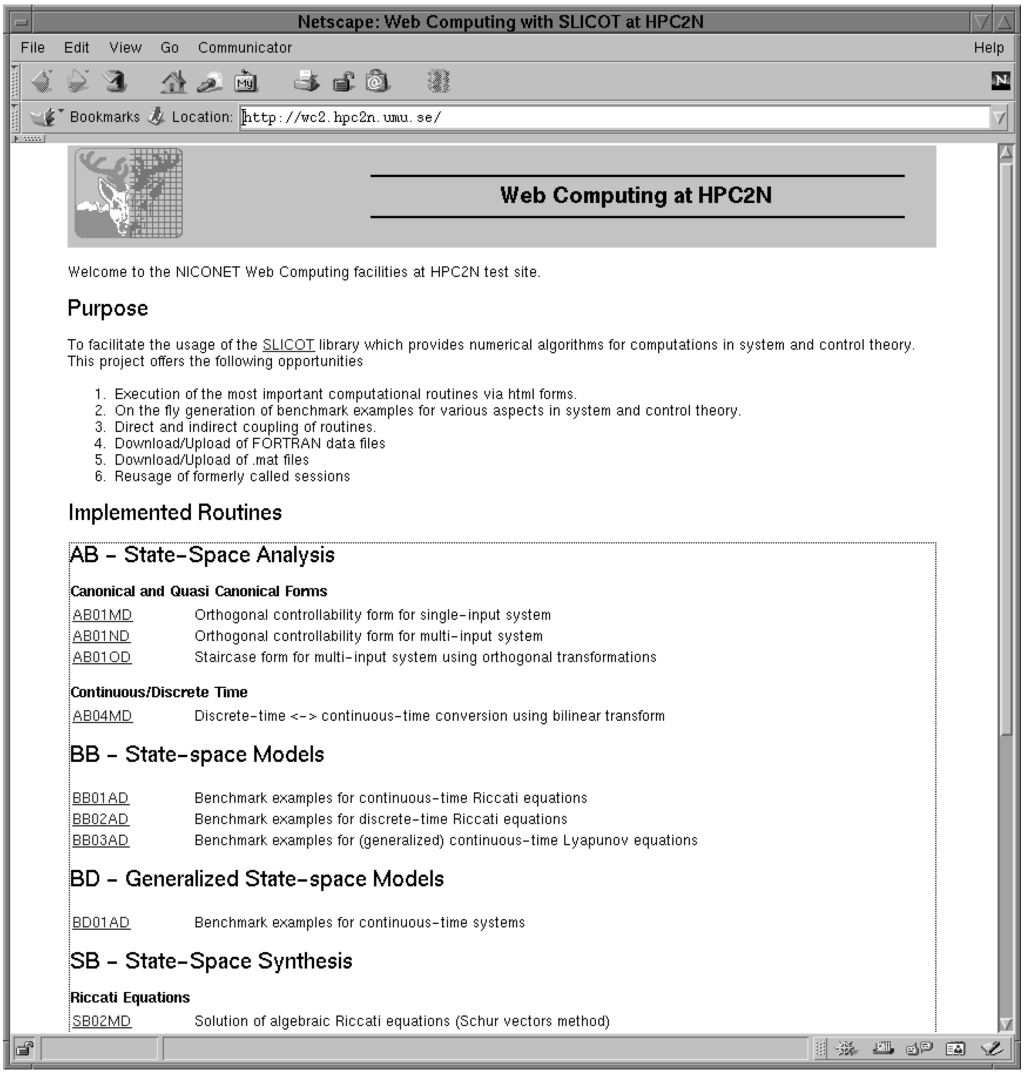


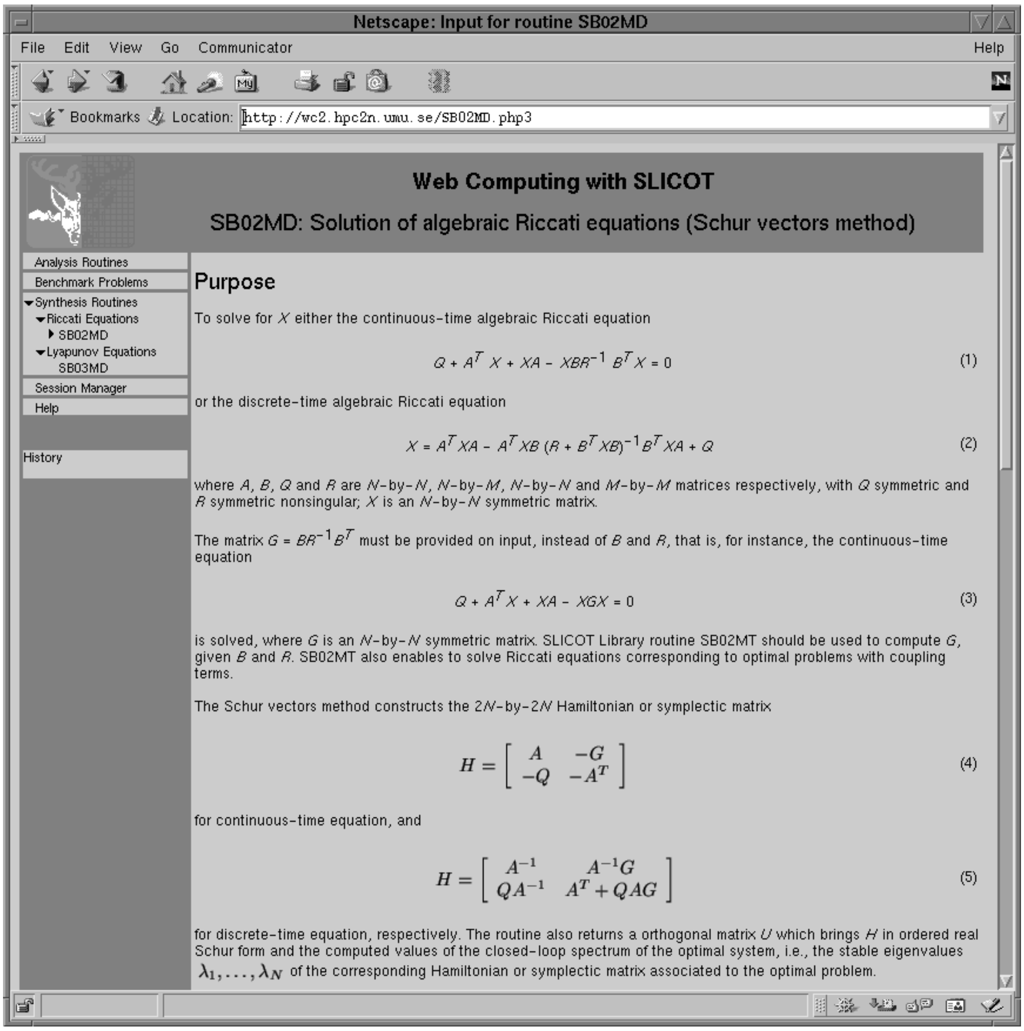
Figure 1: The main page of the web computing service.

In the following, we illustrate how the software can be used to solve a sample problem, namely to compute the solution of continuous-time algebraic Riccati equations

$$Q + A^T X + XA - XGX = 0. \tag{1}$$

The matrices defining the problem can be entered by the user in specific fields in the web interface. They can also be uploaded from Matlab binary or Fortran data files.

The underlying routine `SB02MD` is an implementation of a Schur vectors method for solving algebraic Riccati equations [5, 8, 7]. The web interface provides full flexibility in specifying data and computational options. Figures 2 and 3 show the user interface for solving algebraic Riccati



Netscape: Input for routine SB02MD

File Edit View Go Communicator Help

Bookmarks Location: http://wc2.hpc2n.umu.se/SB02MD.php3

Analysis Routines
Benchmark Problems
Synthesis Routines
Riccati Equations
SB02MD
Lyapunov Equations
SB03MD
Session Manager
Help

History

Web Computing with SLICOT

SB02MD: Solution of algebraic Riccati equations (Schur vectors method)

Purpose

To solve for X either the continuous-time algebraic Riccati equation

$$Q + A^T X + XA - XBR^{-1} B^T X = 0 \tag{1}$$

or the discrete-time algebraic Riccati equation

$$X = A^T XA - A^T XB (R + B^T XB)^{-1} B^T XA + Q \tag{2}$$

where A, B, Q and R are N-by-N, N-by-M, N-by-N and M-by-M matrices respectively, with Q symmetric and R symmetric nonsingular; X is an N-by-N symmetric matrix.

The matrix $G = BR^{-1}B^T$ must be provided on input, instead of B and R, that is, for instance, the continuous-time equation

$$Q + A^T X + XA - XGX = 0 \tag{3}$$

is solved, where G is an N-by-N symmetric matrix. SLICOT Library routine SB02MT should be used to compute G, given B and R. SB02MT also enables to solve Riccati equations corresponding to optimal problems with coupling terms.

The Schur vectors method constructs the $2N$-by-$2N$ Hamiltonian or symplectic matrix

$$H = \begin{bmatrix} A & -G \\ -Q & -A^T \end{bmatrix} \tag{4}$$

for continuous-time equation, and

$$H = \begin{bmatrix} A^{-1} & A^{-1}G \\ QA^{-1} & A^T + QAG \end{bmatrix} \tag{5}$$

for discrete-time equation, respectively. The routine also returns a orthogonal matrix U which brings H in ordered real Schur form and the computed values of the closed-loop spectrum of the optimal system, i.e., the stable eigenvalues $\lambda_1, \ldots, \lambda_N$ of the corresponding Hamiltonian or symplectic matrix associated to the optimal problem.

Figure 2: Introduction to solving algebraic Riccati equations.

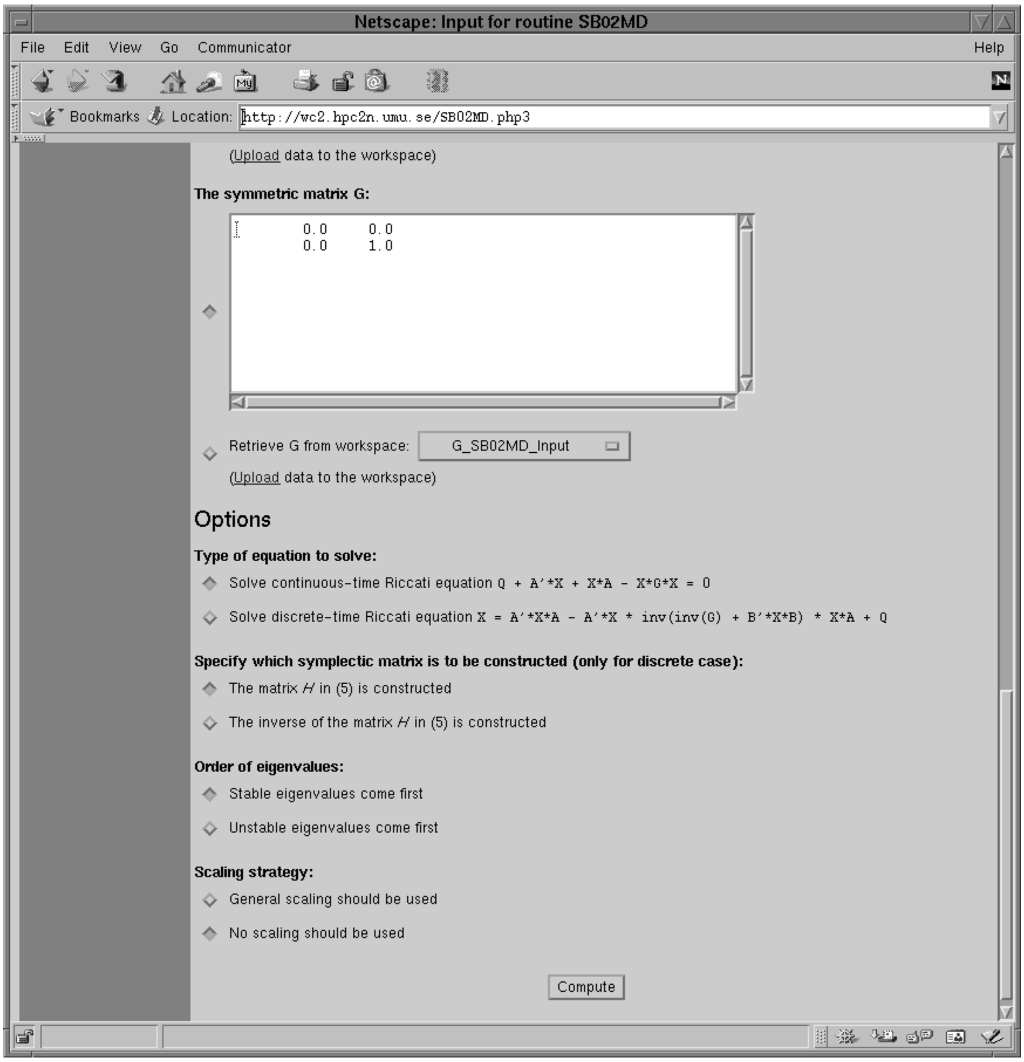


Figure 3: Solving algebraic Riccati equations.

equations. Here, the matrices have already been entered and the user may select a number of different options, e.g., the type of equation to solve, preferred eigenvalue ordering, and scaling strategy. Of course, the available options depend on the type of problem to solve.

After pressing the “Compute” button in the window of Figure 3, the output is presented as in Figure 4. The solution is presented on the screen but also possible to download, e.g., as plain text files or as Matlab binaries. By pressing the “Error Bound” button, an error bound of the computed solution is calculated.

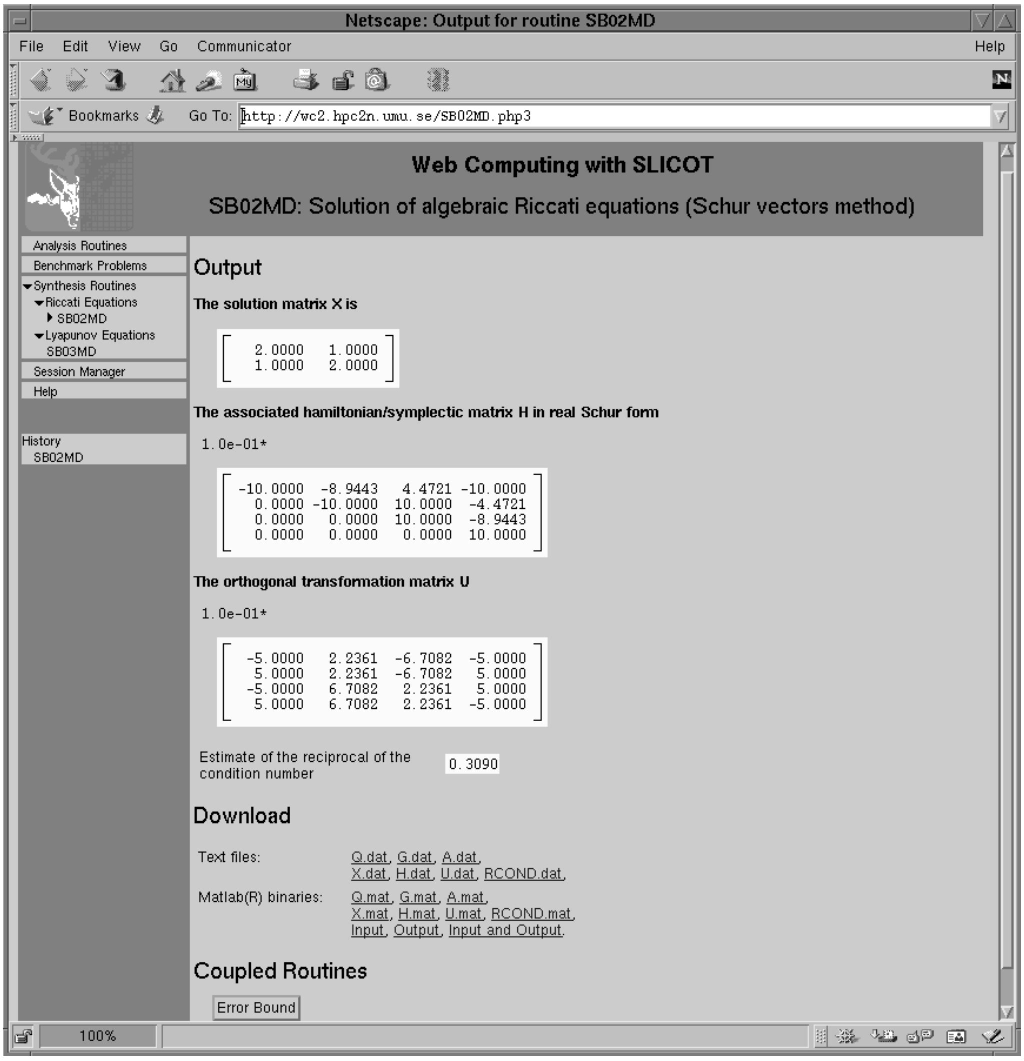


Figure 4: Solution and coupled routines.

During the web computing session, matrices are stored in a workspace, making the result from the algebraic Riccati equation solver available as input in subsequent computations. The workspace manager serves as a container for all input and output matrices. A convenient interface enables administrating this workspace, as illustrated in Figure 5.

Additional features provided by the workspace manager include possibilities to download

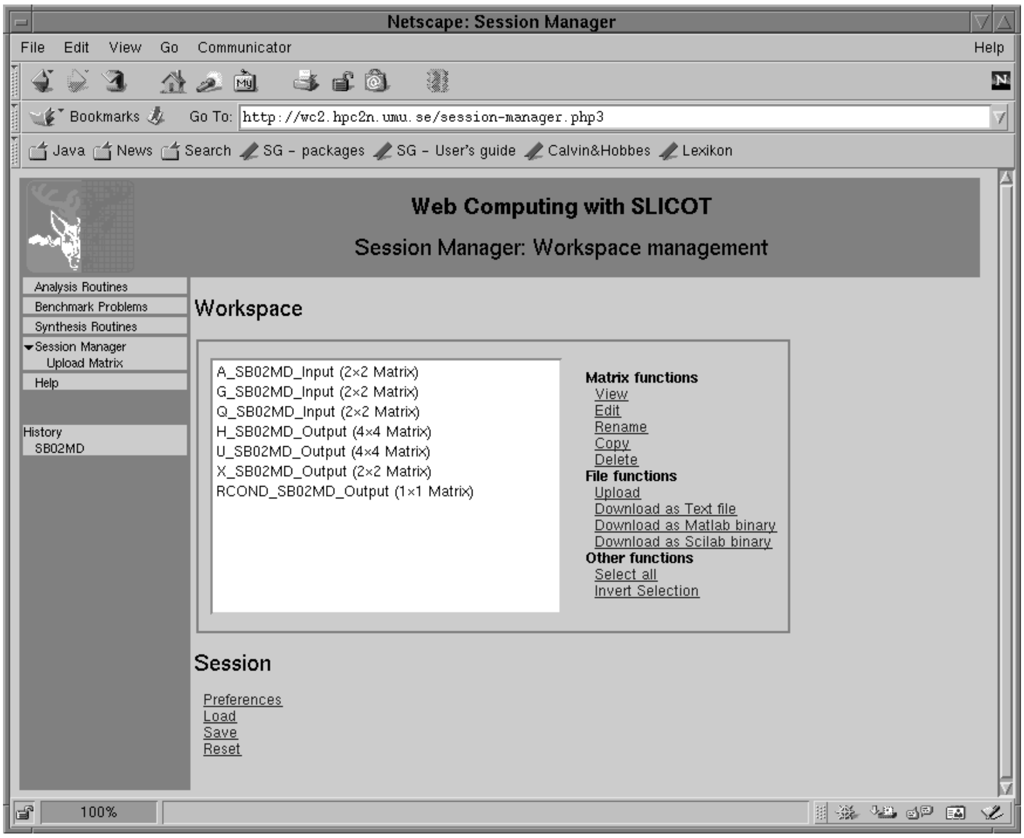


Figure 5: Workspace manager.

matrices in Fortran data format, LaTeX format, Matlab and Scilab binary format on the local computer and to save complete sessions. The session data is stored on the remote machine so that it may be recovered at a different client machine.

In order to facilitate the test and evaluation of the SLICOT library through the web interface, the SLICOT benchmark collection is also provided. This collection provides relevant test problems for the different routines.

3 Implementation Details

Apart from making the web interface intuitive and easy to use, an important aspect in the design of the web computing environment is to facilitate the expansion of the system. Future versions will include a larger number of SLICOT routines, and possibly also other software that may be useful together with SLICOT, e.g., BLAS and other basic computational routines.

Figure 6 describes in a schematic way the overall design of the system.

As the top level programming language we use PHP[1], which is a server-side HTML-embedded

[1]PHP: Hypertext Preprocessor (The first name was Personal Homepage).

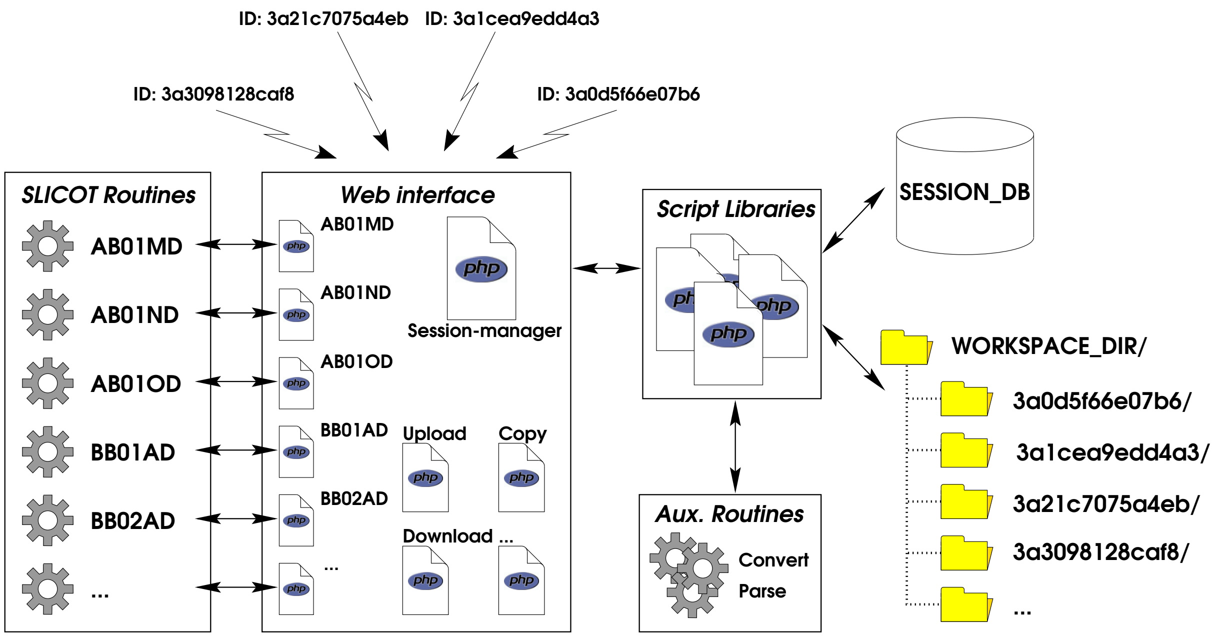


Figure 6: A schematic figure over the web computing system design.

scripting language. This means that based on the request of the user and the state of the system, a new web page is created and sent back to the user (PHP-scripts are visualized as a paper with a PHP-logo in Figure 6). In addition to the PHP-scripts, there are also the Fortran-based SLICOT-routines and some C++ based auxiliary routines for conversion and parsing (gear-wheel in Figure 6).

The process when a user first starts to use this interface with a browser is as follows. The input data and the computed results are stored for each individual user. These data are kept in a folder in the file system and maintained in a database. To identify different users, each user is handed a unique ID, stored by the user's browser in something called a cookie. Every time the user goes to one of the pages on the web computing server this ID is sent to the server. If there is no previous information stored about the ID, the server creates a new post in the database and a new folder where data can be stored.

When the user then uploads new data or enter data on the routine web page, the data is stored and accessible as long as the user's browser keeps the given ID. Notice that the identification with the cookie is handled by the interaction between the web server and the user's browser and not something the user needs to do.

When one of the SLICOT routines is called, the input data is supplied to the SLICOT routine that will be executed on the server. The routine then computes some output data that is presented to the user and stored on the server for later use or download. All data on the server can also be converted to different formats, e.g., Matlab binaries. Notice that all routines reside on the server and that the computation is performed there and therefore not dependent on the performance or the operating system of the user's machine.

The code in PHP-scripts for the routines and other pages are kept at a minimum. Instead, all code that handle, for example, the calls to routines, communication, parsing and layout, are gathered in libraries that can be used by the other scripts. This makes it easy to modify and add new routines to the system.

4 Conclusions and Future Work

The web computing environment presented provides test and use of SLICOT routines from a standard web browser. The routines can be executed on the remote web server with no need for any software installation on the local computer. Consequently, a user can test the SLICOT routines with very little effort, and he or she can connect to the same user interface from virtually any computer. Input matrices can either be entered directly by the user or uploaded from the local computer. Output matrices are presented on the screen and saved in a user workspace for future use, but they may also be downloaded to the local computer. A users' guide will be integrated with the user interface. Moreover, the prototype will be extended to include all SLICOT routines relevant to this type of environment.

Future directions for this project include the development of a computational grid environment. From a standard program, e.g., written in C or Fortran, the user will be able to execute SLICOT routines on more powerful remote computers. In order to do this, the user will only have to install a small software library for handling of the remote access. This library will be used instead of a local SLICOT library. From the point of view of a user, the compilation, execution and linking will be performed exactly as when running on the local computer. In practice, the SLICOT routines are run on some powerful remote computer instead of the local computer (where the other parts of the program are running). As an extension of this concept, the user will also be able to run the whole program on a computational grid of remote computers, where different parts of the execution may take place on different appropriate computers. Of course, the web interface described in this contribution will be able to direct the computations to this powerful grid.

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.* Society for Industrial and Applied Mathematics, Philadelphia, third edition, 1999.

[2] P. Benner, V. Mehrmann, V. Sima, S. Van Huffel, and A. Varga. SLICOT - A Subroutine Library in Systems and Control Theory. In Biswa N. Datta, editor, *Applied and Computational Control, Signal and Circuits*, volume 1, chapter 10, pages 499–539, 1999.

[3] B. Kågström, P. Ling, and C. Van Loan. GEMM-based level 3 BLAS: High-Performance Model Implementations and Performance Evaluation Benchmark. *ACM Trans. on Math. Software*, 24(3):268–302, 1998.

[4] B. Kågström, P. Ling, and C. Van Loan. GEMM-based level 3 BLAS: Portability and Optimization Issues. *ACM Trans. on Math. Software*, 24(3):303–316, 1998.

[5] A. Laub. A Schur method for solving algebraic Riccati equations. *IEEE Trans. Autom. Contr.*, AC-24:913–921, 1979.

[6] NICONET. Network for Development and Evaluation of Numerically Reliable Software in Control Engineering and its Implementation in Production Technologies. Brite/EuRam III RTD Thematic Network, Contract ERB BRRT-CT-97-5040, Coordinator Sabine Van Huffel, ESAT-SISTA, Katholieke Universiteit Leuven, Kardinaal Mercierlaan 94, B-3001 Leuven, Belgium, April, 1997. http://www.win.tue.nl/niconet/.

[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] W.M. Wonham. On a Matrix Riccati Equation of Stochastic Control. *SIAM J. Contr.*, 6:681–697, 1968.