12

Software for Numerical Linear Algebra

There is a variety of computer software available to perform the operations on vectors and matrices discussed in Chapter 11. We can distinguish the software based on the kinds of applications it emphasizes, the level of the objects it works with directly, and whether or not it is interactive. Some software is designed only to perform certain functions, such as eigenanalysis, while other software provides a wide range of computations for linear algebra. Some software supports only real matrices and real associated values, such as eigenvalues. In some software systems, the basic units must be scalars, and so operations on matrices or vectors must be performed on individual elements. In these systems, higher-level functions to work directly on the arrays are often built and stored in libraries. In other software systems, the array itself is a fundamental operand. Finally, some software for linear algebra is interactive and computations are performed immediately in response to the user's input.

There are many software systems that provide capabilities for numerical linear algebra. Some of these grew out of work at universities and government labs. Others are commercial products. These include the IMSL™ Libraries, MATLAB®, S-PLUS®, the GAUSS Mathematical and Statistical System™, IDL®, PV-Wave®, Maple®, Mathematica®, and SAS IML®. In this chapter, we briefly discuss some of these systems and give some of the salient features from the user's point of view. We also occasionally refer to two standard software packages for linear algebra, LINPACK (Dongarra et al., 1979) and LAPACK. (Anderson et al., 2000).

The Guide to Available Mathematical Software (GAMS) is a good source of information about software. This guide is organized by types of computations. Computations for linear algebra are in Class D. The web site is

`http://gams.nist.gov/serve.cgi/Class/D/`

Much of the software is available through `statlib` or `netlib` (see page 505 in the Bibliography).

For some types of software, it is important to be aware of the way the data are stored in the computer, as we discussed in Section 11.1 beginning on page 429. This may include such things as whether the storage is row-major or column-major, which will determine the stride and may determine the details of an algorithm so as to enhance the efficiency. Software written in a language such as Fortran or C often requires the specification of the number of rows (in Fortran) or columns (in C) that have been allocated for the storage of a matrix. As we have indicated before, the amount of space allocated for the storage of a matrix may not correspond exactly to the size of the matrix.

There are many issues to consider in evaluating software or to be aware of when developing software. The portability of the software is an important consideration because a user's programs are often moved from one computing environment to another.

Some situations require special software that is more efficient than general-purpose software would be. Software for sparse matrices, for example, is specialized to take advantage of the zero entries. For sparse matrices it is necessary to have a scheme for identifying the locations of the nonzeros and for specifying their values. The nature of storage schemes varies from one software package to another. The reader is referred to GAMS as a resource for information about software for sparse matrices.

Occasionally we need to operate on vectors or matrices whose elements are variables. Software for symbolic manipulation, such as Maple, can perform vector/matrix operations on variables. See Exercise 12.6 on page 476.

Operations on matrices are often viewed from the narrow perspective of the numerical analyst rather than from the broader perspective of a user with a task to perform. For example, the user may seek a solution to the linear system $Ax = b$. Most software to solve a linear system requires A to be square and of full rank. If this is not the case, then there are three possibilities: the system has no solution, the system has multiple solutions, or the system has a unique solution. A program to solve a linear system that requires A to be square and of full rank does not distinguish among these possibilities but rather always refuses to provide any solution. This can be quite annoying to a user who wants to solve a large number of systems using the same code.

Writing Mathematics and Writing Programs

In writing either mathematics or programs, it is generally best to think of objects at the highest level that is appropriate for the problem at hand. The details of some computational procedure may be of the form

$$\sum_i \sum_j \sum_k a_{ki} x_{kj}. \tag{12.1}$$

We sometimes think of the computations in this form because we have programmed them in some low-level language at some time. In some cases, it is important to look at the computations in this form, but usually it is better