Standard Software for Astronomy – the Starlink SLALIB Example

Patrick Wallace
Starlink Project, Rutherford Appleton Laboratory,
Chilton, Didcot, Oxfordshire, UK
ptw@star.rl.ac.uk

Starlink's SLALIB library is a collection of algorithms used in astronomical position and time applications. After describing the library in the context of Starlink as a whole, we look at more general questions about the provision of such software.

1. STARLINK

The Starlink Project provides data reduction and analysis facilities for UK astronomers. It began in 1980, funded and operated by the SERC to support astronomy research in the universities. There are currently about 1800 users at 26 sites. The service is operated by 16 staff at RAL plus 35 at Universities. Starlink provides computer equipment, maintenance, and on-site system management. The present architecture is a mix of servers, X-terminals and Unix workstations at each site. A standard collection of software is maintained, which runs on several different types of computer.

The Starlink Software Collection (SSC) is large (about 4,000,000 lines of Fortran and C) and diverse. Most software items come from outside the Starlink-funded team. The SSC, which is divided into a *core* system and *options*, is available free of charge to non-profit research organizations.

2. SLALIB

SLALIB is a small part of the SSC. It is a library of about 150 subprograms, most of which perform standard mathematical or astronomical operations. Several key routines are adaptations of code from other sources. SLALIB routines are used mainly in applications related to positional astronomy and time, for example ASTROM (plate astrometry), COCO (conversion of star coordinates), RV (radial velocity corrections) and TPOINT (telescope pointing analysis). The 150 routines total about 15,000 lines of code; independent but equivalent Fortran and C versions are maintained. SLALIB can be used on VAX/VMS, PC (MS-DOS) and several different Unix systems.

The library does not claim to be canonical, orthogonal, comprehensive, homogeneous, foolproof and superfast. It is, however, self-contained, accessible, supported, documented, platform-independent, relatively stable, fairly rigorous, reasonably fast, comprehensive enough for most practical purposes, widely used and is available as source code.

3. STANDARD LIBRARIES IN GENERAL

Providing standard software poses many difficulties, and even for one purpose there may be multiple and often conflicting needs.

Such software must, of course, be free of charge, network-accessible, authoritative, reliable, easy to use, have no copyright strings attached, be properly maintained and the source code must be available and comprehensible. But who decides what is needed? Are libraries enough or should ready-made utility applications also be provided? What functions should be covered? Practical software or specimen code only? How many variants (*e.g.* for precession)? Who lays down style guidelines? Languages? Granularity and orthogonality? Approach to error handling? Who writes the software? Who provides support? Who guarantees it works? Does it have to be crashproof? How accurate? Up to date or canonical?

Feasible approaches include volunteer providers together with IAU referees, or special initiatives – by the IAU (*cf* SOFA), the space agencies, major observatories or by the national almanac/time offices. Whatever solution is chosen, it is important to ensure that the need to standardize does not stifle richness and evolution.

I. Appenzeller (ed.), Highlights of Astronomy, Vol. 10, 202.