STSDAS Version 2.3
June 2001

STSDAS Site Manager's Installation Guide and Reference

SPACE
TELESCOPE
SCIENCE
INSTITUTE

Science Software Group
Science Support Division
3700 San Martin Drive
Baltimore, Maryland 21218

Operated by the Association of Universities for Research in Astronomy, Inc., for the National Aeronautics and Space Administration

The STSDAS Group

Perry Greenfield	Lead, Science Software Group, PyRAF, FOC, Fourier
Paul Barrett	FUSE, PyFITS
Howard Bushouse	NICMOS, Synthetic Photometry, WFC3
Ivo Busko	STIS, SpecView, Java, Isophote, Fitting
Michele de la Peña	PyRAF, CVOS, FOS, GHRS, Java
Nadezhda Dencheva	STSDAS/TABLES installation and releases, CVS
Warren Hack	Dither, ACS, Graphics, Paper Products, FOC
Phil Hodge	Table System, COS, STIS
J.C. Hsu	WF/PC, Convfile, Ctools, Timeseries, FITS, PyFITS
Robert Jedrzejewski	Deputy Lead, NICMOS, ACS, FOC, WF/PC
Todd Miller	PyRAF, numarray
Hemant Shukla	ETC, APT, Java
Eric Wyckoff	Software Testing, User Support

Send comments or corrections to:
STSDAS Group
Space Telescope Science Institute
3700 San Martin Drive
Baltimore, Maryland 21218
E-mail: help@stsci.edu

M0222

Table of Contents

Preface

NOTE: ***OpenVMS is no longer supported by STScI for TABLES and STSDAS.***

The Space Telescope Science Data Analysis System (STSDAS) is a set of application programs designed for the calibration and analysis of data from the Hubble Space Telescope. Applications include general image processing as well as tasks specific to the HST. STSDAS also has its own graphics package and a FITS I/O package specifically designed to read HST format image data.

STSDAS/TABLES version 2.3 is now available for the following architectures: Solaris 2.5.1 and 2.8, RedHat 6.1, Digital Unix 5.1, HP-UX B.10.20.

Before installing STSDAS/TABLES version 2.3 you must:

1 • Obtain and install IRAF version 2.11.1 (or higher) (2.11.3 suggested).

2 • Obtain and install the STSDAS and TABLES packages (see Chapter 1). All STSDAS/TABLES software is available via anonymous ftp to ftp.stsci.edu (130.167.1.2).

STSDAS is fully layered upon IRAF, the Image Reduction and Analysis Facility, which is developed and supported by NOAO. If you attempt to install STSDAS without IRAF you will get nowhere—the STSDAS installation procedures use IRAF utilities. STSDAS 2.3 requires version 2.11.1 of IRAF or later.

STSDAS uses the TABLES external package, also available from STScI, which is a table I/O system that supports the transfer of tabular data from one application to another. There is a table manipulation tool kit that allows one to use tables as small relational databases. Note that *you must already have installed* TABLES in order to compile STSDAS. STSDAS links against some of the TABLES system libraries, causing unresolved references if you try installing STSDAS without already having installed TABLES.

Changes to the TABLES libraries can also affect STSDAS compilation. We suggest that you match the version numbers of TABLES and STSDAS to maximize compatibility.

This release of STSDAS has been extensively tested at STScI. We encourage off-site users to try as much of the system as possible and send us comments and criticisms. Software problem reports (SPRs) can be submitted via e-mail (help@stsci.edu). The **problems** package in STSDAS is now obsolete. Alternatively, a software problem report form is included at the end of this guide; you may make photocopies of this form and send them in by regular mail. You should also feel free to contact us by telephone (410-338-1082). Problems will be addressed as quickly as possible.

If You Have Problems...

- Quick troubleshooting information for site managers is provided on the "Site Managers Quick Reference" sheet, available from the Help Desk staff.
- If you have any problems *installing or using* STSDAS or TABLES contact the Help Desk staff by sending e-mail to: help@stsci.edu, or by phoning (410) 338-1082.

CHAPTER 1: Installing TABLES

In This Chapter...

Installation of TABLES 2.3 is fairly easy and straightforward. It is done within the IRAF 2.11.1 (or higher) environment. This chapter is intended as a cookbook to help you do a TABLES installation.

It should be noted that there already exists a task in the **lists** package of IRAF named **table** which may cause a conflict with the package name **tables**. When loading this package, the whole name should be typed out to ensure that the correct task is being run.

The Installation Process

TABLES 2.3 is a new release of TABLES. You will need both the source code tar file and the binary tar file for a binary installation, or just the source code tar file if you will be compiling your own binaries.

To install TABLES, you must:

1 • Create the top-level directory for TABLES (below).

2 • Edit the file `hlib$extern.pkg` to define pointers.

3 • Install the TABLES source code from tape or from the tar files. (See “Installing the Source Code” on page 4.) The latest versions of the tar files are available via anonymous ftp: ftp.stsci.edu://pub/software/stsdas/tables_v2.3/source.

4 • Install the TABLES binaries for your architecture. **Solaris, RedHat, Digital Unix and HP-UX only**. (See “Installing the Binaries” on page 6.)The latest versions of the tar files are available via anonymous ftp from ftp.stsci.edu://pub/software/stsdas/tables_v2.3/binaries.

5 • Modify the help database. (See “The Help Database” on page 7.)

6 • Test the system. (See “Testing the TABLES Installation” on page 7.)

7 • Install additional binaries, if multi-architecture support is needed. (See “Multi-architecture Support for TABLES” on page 7.)

You may also choose to compile TABLES yourself. (See “Building TABLES from Scratch” on page 8.)

Selecting the Top Directory

TABLES is based on the structure of IRAF. We suggest installing TABLES as a separate directory structure and recommend naming the top directory tables. This will enable you to more easily make updates to the respective systems and allow you to easily add other packages. This is the method used in the examples in this guide.

If for some reason this procedure cannot be followed, it is still straightforward to install TABLES. All package directories are specified relative to the top level TABLES directory. There is one IRAF environment variable, **tables**, that is used as the basis for all package definitions.

Pre-Installation Site Modifications

Installation of TABLES is done within the IRAF **cl**. IRAF must know where you intend to put TABLES before the tape is read. The IRAF file `hlib$extern.pkg` contains the locations of all external packages.

TABLES 2.3 must be installed using IRAF 2.11.1 (2.11.3 preferred). All of the binaries in the TABLES 2.3 release were created under IRAF 2.11.3. If you do not have IRAF 2.11.3 (or 2.11.1 or 2.11.2) installed, you must install it before proceeding with the TABLES 2.3 installation. You can get IRAF at the anonymous ftp site, `iraf.noao.edu` in the sub-directory `/iraf/v211.`

To edit the `hlib$extern.pkg` file, change your current directory to `hlib$`:

```
cl> cd hlib$
cl> edit extern.pkg
```

Two modifications need to be made to the file `hlib$extern.pkg`:

- The TABLES package and its location must be defined.
- The path to the TABLES help database must be included.

To tell IRAF where the TABLES system will be located, add the package definition lines before the ‘`reset helpdb ...`’ line in `hlib$extern.pkg`:

```
reset tables = /path/tables/
task tables.pkg = tables$tables.cl
```

To include TABLES in the help search path, add the string ‘`tables$lib/helpdb.mip`’ to the list of help database locations.

An example of the modified `extern.pkg` file is shown in Figure 1.1.

```
# External (non core-system) packages. To install a new package, add the
# two statements to define the package root directory and package task,
# then add the package helpdb to the 'helpdb' list.
reset   noao          = iraf$noao/
task    noao.pkg      = noao$noao.cl

reset   local           = iraf$local/
task    local.pkg       = local$src/local.cl

reset   tables          = /node/dir/tables/
task    tables.pkg      = tables$tables.cl

reset   helpdb          = "lib$helpdb.mip\
                           ,noao$lib/helpdb.mip\
                           ,local$lib/helpdb.mip\
                           ,tables$lib/helpdb.mip\
                           "
keep
```

TABLES Package Location

Help Database Locations

Figure 1.1: Modified extern.pkg File

Installing the Source Code

The TABLES release is available in two forms:

- Tape
- Anonymous ftp (`ftp.stsci.edu`)

In each case, the release consists of source code only and is contained in a tar (tape archive) format.

The tape files were written using the IRAF task **wtar**. Files can be read using the IRAF task **rtar**. If you are on a Unix system, you can also use the Unix command **tar**. On System V Unix systems, you may experience some problems resolving links when using the IRAF **rtar** task. If you are running under System V, we recommend using the standard Unix **tar** command to read the files.

At this point you should `logout` from IRAF and restart the **cl** in order to pick up the new TABLES package definition that you set up in the previous step.

Installing the Export Tape

It will probably take about 30 minutes to read in the tape, which contains a total of 21 megabytes of source files. The full system, after all the executable images have been built, is about 37 megabytes .

Mount the TABLES installation tape on one of the tape drives available to your IRAF installation (this will be known as `mta, mtb,` etc.). Then load the **softools** package of IRAF and use the **rtar** task to read the tape:

```
cl> softools
cl> cd tables$
cl> allocate mta
cl> rtar -xtvf device
cl> deallocate mta
```

The string "mta" is the IRAF name of your tape drive...this may be mtb, mtc, etc.

The string "device" is the System name of your tape drive (e.g. /dev/rst0 or 1MUA0:)

Figure 1.2: Reading a TABLES Distribution Tape

Installing the FTP Tar File

The TABLES source tar files are available from the anonymous ftp site `ftp.stsci.edu` (130.167.1.2) in the directory `/pub/software/stsdas/tables_v2.3/source`

The TABLES source tar files are placed on the anonymous ftp site in a set of split and compressed tar files (`tables23.tar.Z.`*`nnn`*).

Due to the size of the file, we advise putting it in a temporary storage area, such as `/tmp`. The source tar is compressed and split, so you need to uncompress it. Use the Unix commands `cat` and `uncompress` to create an uncompressed tar file:

```
% cat tables23.tar.Z.* | uncompress > tables.tar
```

Once the uncompressed tar has been recreated, the tar file is called `tables.tar` and is located in the `tables` directory on the temporary storage disk. From within IRAF, load the **softools** package and use the **rtar** task to read from the tar file.

```
cl> softools
so> cd tables$
so> rtar -xtvf /tmp/tables.tar
```

Figure 1.3: Reading the tar File in IRAF on Unix

Installing the Binaries

If you installed the source from a tape the binaries will already be installed in the appropriate directory and you can proceed directly to the next section “The Help Database” on page 7. If you installed the source from the FTP tar file then you will need to get an additional tar file from the anonymous ftp site ftp.stsci.edu (130.167.1.2) from the directory `/pub/software/stsdas/tables_v2.3/binaries.` Here you will find a sub-directory for each supported architecture.

Architecture	***Command***
Sun SunOS 4.x	sparc
Sun Solaris 2.5.1 or 2.8	ssun
PC Platforms Linux Slackware 3.3 Linux Red Hat 6.1 FreeBSD 2.2.5	 linux redhat freebsd
DEC Alpha with Digital Unix 5.1(OSF/1)	alpha
HP-UX 10.20	hp700
SGI IRIX 6.5	irix

Table 0.1: Currently Supported Architectures

Each subdirectory will have the binaries in a series of split, compressed tar files (`tables23.bin.`*`arch`*`.tar.Z.`*`nnn`*)

Note: *We have compiled binaries only for Solaris 2.5.1 or 2.8, RedHat 6.1, Digital Unix 5.1 (OSF/1) and HP-UX 10.20. If there is a substantial interest in the other platforms we could provide binaries for those as well.*

Due to the size of the file, we advise putting it in a temporary storage area, such as `/tmp`. Use the Unix commands `cat` and `uncompress` to create an uncompressed tar file:

```
% cat tables23.bin.arch.tar.Z.* | uncompress > tables.tar
```

Once the uncompressed tar has been recreated, the tar file is called `tables.tar` and is located in the `tables` directory on the temporary storage disk.

From within IRAF, load the **softools** package and use the **rtar** task to read from the tar file.

```
cl> softools
so> cd tables$bin.arch
so> rtar -xtvf /tmp/tables/tables.tar
```

Figure 1.4: Reading the tar File in Unix

The Help Database

The help database is provided in a machine independent form and need not be rebuilt. If you wish to rebuild it, you may do so by running the **mkhelpdb** task within the **softools** package of IRAF.

```
cl> softools
so> mkhelpdb helpdir=tables$lib/root.hd \
>>>  helpdb=tables$lib/helpdb.mip
```

Figure 1.5: Rebuilding the Help Database

Testing the TABLES Installation

Once TABLES has been loaded from the export tape and rebuilt, some of the basic functions of TABLES as well as a few device-dependent tasks (e.g., plotting) should be exercised. When testing device-dependent things such as plotting and image display, be sure that the IRAF environment variables that point to the device(s) are correct.

Multi-architecture Support for TABLES

It is possible to support multiple architectures using a single source tree. If you wish to support, for example, both Solaris and SunOS architectures with a single source tree, you would follow these steps:

- Create a top-level directory for TABLES
- Edit the `hlib$extern.pkg` file. (See “Pre-Installation Site Modifications” on page 3.)
- Install the TABLES source code. (See “Installing the Source Code” on page 4.)
- Install the binaries for the Solaris architecture. (See “Installing the Binaries” on page 6.)
- Install the binaries for the SunOS architecture. (See “Installing the Binaries” on page 6.)
- Modify the help database. (See “The Help Database” on page 7.)

Building TABLES from Scratch

If binaries for your computer’s architecture are not available, then you will need to compile TABLES from scratch.

> Earlier versions of TABLES for some architectures can be obtained from `http://ra.stsci.edu/GetSoftware.html`

The system rebuild can be done with a batch procedure submitted while within the IRAF **cl**. Load the **tables** and **softools** packages, set the current directory to the top TABLES directory, and execute the **mkpkg** task.

The first step in a relink is to insure that some system variables are set. Type the following command at the system level before proceeding:

```
% setenv IRAFARCH arch
```

where *arch* is your specific architecture, e.g., `sparc` for a SparcStation.

```
% setenv iraf /path/iraf/
```

where *path* is the directory path to the top-level IRAF directory.

```
% source $iraf/unix/hlib/irafuser.csh
```

This sets up some other environment variables needed to compile under IRAF. You may set these up in your `.login` file so that they will be available.

Also, ensure that the directory that contains the local Unix commands (usually `/usr/local/bin`) is included in your PATH environment variable. If it is not, you can add it to your path by typing the following:

```
% setenv PATH :/usr/local/bin$PATH
```

Before attempting the total system rebuild, you should check the soft link for the `bin` directory. TABLES is shipped with a link for bin pointing to `bin.generic`. This should be changed so that it points to the appropriate `bin` for your architecture. To do this simply type the following command from the TABLES top-level directory:

```
% mkpkg arch
```

where *arch* is your specific architecture. A list of available architectures is provided in Table 0.2. For example, for a SparcStation, you would type:

```
% mkpkg sparc
```

You will get a warning message about a full “sysgen” needing to be done, but that is normal.

Architecture	*Command*
Sun SunOS 4.x	`mkpkg sparc`
Sun Solaris 2.5.1	`mkpkg ssun`
Sun Solaris 2.8	`mkpkg ssun`
PC Platforms with Linux	
Slackware 3.3	`mkpkg linux`
Red Hat 6.1	`mkpkg redhat`
FreeBSD 2.2.5	`mkpkg freebsd`
Alpha with Digital Unix 5.1(OSF/1)	`mkpkg alpha`
Hewlett-Packard with HP-UX 10.20	`mkpkg hp700`
SGI IRIX 6.5	`mkpkg irix`

Table 0.2: Supported Architectures

Warning for Unix users:
Several architectures supported by IRAF have some idiosyncrasies:

SystemV Unix (Solaris, IBM RISC,IRIX).: For some reason the link for `bin` in `tables$` is not correctly resolved on System V Unix machines using **rtar**. This causes the command `mkpkg arch` to fail. Create this link by hand using the **ln -s** command at the Unix shell level:

```
% rm bin
% ln -s bin.arch bin
```

This problem also causes the libraries in `tables$lib` to incorrectly resolve. This can cause the libraries being compiled by mkpkg to not be created and can yield unresolved references in links further on. Delete these links and things will work.

SUN OS, HP-UX: Some of the code in TABLES 2.3 is written in ANSI C. As such, this requires the SUN ANSI compiler, **acc**, or the HP-UX compiler, **c89**, be used rather than the usual **cc** compiler. In IRAF 2.11.1, this can be set by setting the following environment variable:

```
% setenv XC-CC acc -or-
% setenv XC-CC c89
```

Also, for SunOS, the ANSI library **libansi.a**, needs to be in your **LD_LIBRARY_PATH.** A suggested **LD_LIBRARY_PATH** is:

```
% setenv LD_LIBRARY_PATH /usr/lang/SC1.0/ansi_lib:/usr/lang/SC1.0:
```

The Unix system relink can be done with a batch procedure submitted while within the IRAF **cl.** Load the **tables** and **softools** packages, set the current directory to the top TABLES directory, and execute the **mkpkg** task:

```
cl> tables
ta> softools
so> cd tables
so> mkpkg -p tables update >& spool &
```

Figure 1.6: Rebuilding the Package as an IRAF Background Job

This will run the **mkpkg** task as a background process and put all output and errors into the `tables$spool` file.

The **mkpkg** program generates a long output file describing all steps taken. To reduce this log to the pertinent information about the success of your installation, re-run the **mkpkg** task with the summary option.

```
cl> softools
so> cd tables
so> mkpkg summary >& tables.summ
```

Figure 1.7: Running mkpkg with the Summary Option

FITS table support

For tables in FITS files, the tables library routines call subroutines in the HEASARC FITSIO package, so IRAF tasks that are linked with the tables library can transparently access FITS tables as well as ASCII or STSDAS format binary tables. Two versions of FITSIO are included in the tables distribution, one version written in Fortran and SPP, and one written in C.

The C version (CFITSIO) is currently supported by HEASARC, and it is faster than the Fortran version. The Fortran version has an SPP layer for the I/O routines, so that it is fully compatible with IRAF I/O; in particular, IRAF networking is supported. The C version, on the other hand, uses host system I/O to read and write the FITS files; IRAF virtual file names are supported by converting to host system names, but IRAF networking is not available.

CFITSIO is used by default. If the SPP and Fortran version is required instead, the following steps should be followed.

so> cd tables

```
so> delete tables$bin/libtbtables.a # if it already exists
so> mkpkg -p tables update sppfitsio=yes >& spool &
```

The only difference from a normal build is that sppfitsio=yes is specified when running mkpkg. (Note: mkpkg does not check the value assigned to sppfitsio, it just checks whether sppfitsio is defined, so it has the same effect regardless of the value.) Regardless of the sppfitsio switch, the CFITSIO source files will be compiled and included in the tables library, since they are used directly by some tasks in STSDAS.

If you have any problems installing or using TABLES, contact the STScI Help Desk via e-mail to: help@stsci.edu

---

CHAPTER 2:

Installing STSDAS

In This Chapter...

Installing STSDAS version 2.3 is fairly easy and straightforward and is done within the IRAF 2.11.1 (or higher) environment. This chapter briefly explains how to do an STSDAS installation. We recommend that you at least look at Figure 3.1 on page 36 and Figure 3.2 on page 39 before doing an installation so that you understand how the software is organized and where it expects its directories and files to be located.

Before You Begin...

Before installing STSDAS, you should be aware that:

- STSDAS is linked against libraries in the TABLES package. If you do not already have TABLES installed, you must install and compile TABLES first. If you have a tape distribution, TABLES is included on a separate distribution tape. If you retrieved STSDAS via anonymous FTP, TABLES is available at the same site—ftp.stsci.edu (130.167.1.2) in the directory /pub/software/tables.
- The version number of TABLES *must* be the same as the version number of STSDAS. If you are upgrading STSDAS, you should upgrade TABLES first. See Chapter 1 for details on installing TABLES.

STSDAS will *not* compile without TABLES being installed first!

- The calibration routines in the **hst_calib** packages **nicmos** and **stis** require a large amount of static memory to link and run. If you are using a workstation with less than 96 megabytes RAM, you will probably have problems with these three tasks. We suggest a *minimum* of 64 megabytes of static storage space and 200 megabytes of swap space to ensure a successful mkpkg.
- If you are interested in reading data such as the *Guide Star Catalog* from CD-ROM, you may use the **gasp** package and system-mounted CD-ROMS. Instructions on mounting the CD-ROM to your system are included in Appendix A.

Installation Process

STSDAS 2.3 is a new release of STSDAS. You will need to get the source code tar file and a binaries tar file if you would like to do a binary installation, or just the source code tar file if you are going to compile your own binaries.

To install STSDAS, you must:

1 • Create the top-level directory for STSDAS (below).

2 • Edit the file `hlib$extern.pkg` to define pointers.

3 • Install the STSDAS source code from tape or from the tar files. (See "Installing Source Code" on page 17). The latest versions of the tar files are available via anonymous ftp to ftp.stsci.edu:/pub/software/stsdas/stsdas_v2.3/source.

4 • Install the STSDAS binaries for your architecture. **Solaris, RedHat, Digital Unix and HP-UX only**. The latest versions of the tar files are available via anonymous ftp to ftp.stsci.edu/pub/software/stsdas/stsdas_v2.3/binaries. (See "Installing the Binaries" on page 19.)

5 • Modify the help and apropos databases. See ("The Help Database" on page 20. and See "The Apropos Task" on page 21.)

6 • Test the system. (See "Testing STSDAS" on page 23.)

7 • Install additional binaries, if multi-architecture support is needed. ("Multi-architecture Support for STSDAS" on page 24)

You may also choose to compile STSDAS yourself. ("Building STSDAS from Scratch" on page 24)

Selecting the Top Directory

STSDAS is based on the structure of IRAF. We suggest installing STSDAS as a separate directory structure and recommend naming the top directory `STSDAS`. This will enable you to more easily make updates to the respective systems and allow you to easily add other packages. This is the method used in the examples in this guide.

If for some reason this procedure cannot be followed, it is still straightforward to install STSDAS. All package directories are specified relative to the top level STSDAS directory. There is one IRAF environment variable, called **stsdas**, that is used as the basis for all package definitions.

Pre-Installation Site Modifications

Installation of STSDAS is done from within the IRAF **cl**. IRAF must know where you intend to put STSDAS before the tape is read. The IRAF file `hlib$extern.pkg` contains the locations of all external packages.

STSDAS 2.3 must be installed using IRAF 2.11.1 or higher (2.11.3 preferred). If you do not have IRAF 2.11.3 (or 2.11.1 or 2.11.2) installed, you must install it before proceeding with the STSDAS 2.3 installation. You can get IRAF at the anonymous ftp site, `iraf.noao.edu` in the sub-directory `/iraf/v211.`

To edit the `hlib$extern.pkg` file, change your default directory to `hlib` and edit the file `extern.pkg`:

```
cl> cd hlib
cl> edit extern.pkg
```

Two modifications need to be made to the file `hlib$extern.pkg`:

- The STSDAS package and its location must be defined
- The path to the STSDAS help database must be included

To tell IRAF where the STSDAS system will be located, add the package definition lines before the ‘`reset helpdb ...`’ line in `hlib$extern.pkg`:

```
reset stsdas = /path/stsdas/
task stsdas.pkg = stsdas$stsdas.cl
```

To include STSDAS in the help search path, add the string ‘`stsdas$lib/helpdb.mip`’ to the list of help database locations.

An example of the modified `extern.pkg` file is shown in Figure 2.1.

```
# External (non core-system) packages. To install a new package, add the
# two statements to define the package root directory and package task,
# then add the package helpdb to the 'helpdb' list.
reset   noao            = iraf$noao/
task    noao.pkg        = noao$noao.cl

reset   local           = iraf$local/
task    local.pkg       = local$src/local.cl

reset   stsdas          = /node/dir/stsdas/
task    stsdas.pkg      = stsdas$stsdas.cl

reset   tables          = /node/dir/tables/
task    tables.pkg      = tables$tables.cl

reset   helpdb          = "lib$helpdb.mip\
                           ,noao$lib/helpdb.mip\
                           ,local$lib/helpdb.mip\
                           ,tables$lib/helpdb.mip\
                           ,stsdas$lib/helpdb.mip\
                           "
keep
```

STSDAS Package Location

Help Database Locations

Figure 2.1: Modified extern.pkg File

Installing Source Code

The STSDAS release is available in two forms:

- Tape
- Anonymous ftp

In both cases, the release consists of source code only and is contained in a tar format.

The tape files were written using the IRAF task **wtar**. Files can be read using the IRAF task **rtar**., or if you are on a Unix system, you can use the Unix **tar** command. If you are on a System V Unix system, you may experience some trouble resolving links if you use the **rtar** task; if this is the case, use the Unix **tar** command.

At this point you should `logout` from IRAF and restart the **cl** in order to pick up the new STSDAS package definition that you set up in the previous step.

Installing the Export Tape

It should take approximately 30 minutes to read in the tape, which contains a total of 68 megabytes of source files. The full system, after all the executable images have been built, is 270 megabytes.

Mount the STSDAS installation tape on one of the tape drives available to your IRAF installation (this will be known as `mta, mtb,` etc.). Then load the **softools** package of IRAF and use the **rtar** task to read the tape:

```
cl> softools
so> cd stsdas$
so> allocate mta
so> rtar -xtvf device
so> deallocate mta
```

The string “mta” is the IRAF name of your tape drive...this may be mtb, mtc, etc.

The string “device” is the System name of your tape drive (e.g. /dev/rst0 or 1MUA0:)

Figure 2.2: Reading an STSDAS Distribution Tape

Installing the FTP Tar File

The STSDAS source tar files are available from the anonymous ftp site `ftp.stsci.edu` (130.167.1.2) in the directory `/pub/software/stsdas/stsdas_v2.3/source`

The STSDAS source tar files are placed on the anonymous ftp site in set of split and compressed tar files (`stsdas23.tar.Z.`*nnn*).

Due to the size of the file, we advise putting it in a temporary storage area, such as `/tmp`. If the source tar is compressed or compressed and split, you need to uncompress it. Use the Unix commands `cat` and `uncompress` to create an uncompressed tar file:

```
% cat stsdas23.tar.*.Z | uncompress > stsdas.tar
```

Once the uncompressed tar has been recreated, the tar file is called `stsdas.tar` and is located in the `stsdas` directory on the temporary storage disk. From within IRAF, load the **softools** package and use the **rtar** task to read from the tar file.

```
cl> softools
cl> cd stsdas$
cl> rtar -xtvf /tmp/stsdas/stsdas.tar
```

Figure 2.3: Reading the tar File in Unix

Installing the Binaries

Help Database Locations

If you installed the source from a tape the binaries will already be installed in the appropriate directory and you can proceed directly to the next section "The Help Database" on page 20. If you installed the source from the FTP tar then you will need to get an additional tar files from the anonymous ftp site ftp.stsci.edu (130.167.1.2) from the directory `/pub/software/stsdas/stsdas_v2.3/binaries.` Here you will find a sub-directory for each supported architecture.

Architecture	*Command*
Sun SunOS 4.x	sparc
Sun Solaris 2.5.1 Sun Solaris 2.8	ssun ssun
PC Platforms Linux Slackware 3.3 Linux Red Hat 6.1 FreeBSD 2.2.5	 linux redhat freebsd
Alpha with Digital Unix 5.1 (OSF/1)	alpha
HP-UX 10.20	hp700
SGI IRIX 6.5	irix

Table 2.1: Currently Supported Architectures

Each subdirectory will have the binaries in a series of split, compressed tar files (`stsdas23.bin.`*`arch`*`.tar.Z.`*`nnn`*).

Note: *We have compiled binaries only for Solaris 2.5.1 or 2.8, RedHat 6.1, Digital Unix 5.1 (OSF/1) and HP-UX 10.20. If there is a substantial interest in the other platforms we could provide binaries for those as well.*

Due to the size of the file, we advise putting it in a temporary storage area, such as `/tmp`. Use the Unix commands `cat` and `uncompress` to create an uncompressed tar file:

```
% cat stsdas23.bin.arch.tar.Z.* | uncompress > stsdas.tar
```

Once the uncompressed tar file has been recreated, the tar file is called `stsdas.tar` and is located in the `stsdas` directory on the temporary storage disk.

From within IRAF, load the **softools** package and use the **rtar** task to read from the tar file.

```
cl> softools
so> cd stsdas$bin.arch
so> rtar -xtvf /tmp/stsdas/stsdas.tar
```

Figure 2.4: Reading the tar File in Unix

The Help Database

The help database is provided in a machine independent form and need not be rebuilt. If you wish to rebuild it, you may do so by running the **mkhelpdb** task within the **softools** package of IRAF.

```
cl> softools
so> mkhelpdb helpdir=stsdas$lib/root.hd \
>>>  helpdb=stsdas$lib/helpdb.mip
```

Figure 2.5: Rebuilding the Help Database

The Apropos Task

The **apropos** task has been available at the Institute for several years. This task searches an apropos database for any input string and outputs descriptions of any tasks in its database that match the search string. This task is a powerful tool for those who are not familiar with the package structure of the IRAF, NOAO, STSDAS, or TABLES packages. This task has only been available at the Institute because the procedure used to build the apropos database was VMS-specific. We have rewritten this procedure in SPP, and thus it is now available on all architectures supported by IRAF.

For convenience, we placed this task at the top level of the IRAF **cl** so that it is available without loading any packages. The script for the **apropos** task is in the top level directory of STSDAS. To include this task in your system add the following line to your `hlib$extern.pkg` below the task statement for STSDAS:

```
task   apropos       = stsdas$apropos.cl
```

So, your `hlib$extern.pkg` might look like Figure 2.6.

Unix

```
reset   stsdas     = /path/stsdas/
task    stsdas.pkg = stsdas$stsdas.cl
task    apropos    = stsdas$apropos.cl
```

Figure 2.6: Sample hlib$extern.pkg File

The apropos database, `stsdas$lib/apropos.db`, is provided in a machine independent form and need not be rebuilt. If you wish to rebuild it, you may do so by running the **mkapropos** task within the **toolbox.tools** package of STSDAS.

```
cl> stsdas
st> toolbox
to> tools
to> mkapropos pkglist=iraf,noao,stsdas,tables \
>>> helpdir=lib/root.hd aproposdb=stsdas$lib/apropos.db
```

Figure 2.7: Rebuilding the apropos Database

Psikern Installation

STSDAS is distributed with an IRAF graphics kernel called *psikern* that produces output in Encapsulated PostScript. **psikern** provides a direct connection between the IRAF graphics system and PostScript. The kernel can use colors, fill areas, and has imaging capabilities. Many tasks in STSDAS, in particular those in the **stplot** package, take advantage of these capabilities. Installation of psikern is optional.

To use **psikern**, you must define graphics devices that invoke the kernel. To define new or different graphics devices, the file `dev$graphcap` must be modified. The file `stsdas$pkg/graphics/stplot/psikern.template` contains the basic entries necessary and some examples of how to use the basic entries to get output to a specific printer. To add **psikern** graphics devices to the `graphcap` file, follow these steps:

1 • Make a backup copy of `dev$graphcap`, so you can recover if mistakes are made. Prepend to `dev$graphcap` the two entries in the file `stsdas$pkg/graphics/stplot/psikern.template` marked "REQUIRED ENTRIES". These entries define two psikern graphics devices, `psi_land` and `psi_port`, for output on 8-1/2 x 11 inch paper. These entries create files with the names "tmp$pskxxxx" where "xxxx" are random numbers.

2 • Prepend to `dev$graphcap`, before the required entries specified above, graphics entries for specific printers. The entries marked "EXAMPLES" in `stsdas$graphics/stplot/psikern.template`, demonstrate some example devices used at the Institute.

To create an entry for a specific printer, basically all that needs to be changed is the DD parameter of an entry. For an example in Unix, to create an IRAF graphics device that will produce plots in landscape mode on the printer attached to queue `lw`, the entry you would add to the beginning of `dev$graphcap` would be:

```
lw|PostScript In Landscape mode on queue lw:\
    :DD=lw,tmp$psk,!{lpr -Plw $F; rm $F ;}:tc=psi_land:
```

3 • Once `dev$graphcap` has been modified, you can use the newly defined graphics devices as you would any other IRAF graphics device. For example, to make the device defined in the above example the default output device for plots, make the following definition in your `loginuser.cl` file:

```
set stdplot = lw
```

Also, for any graphics task that uses the `device` parameter to set the output printer, you can specify the newly defined devices. For example:

```
cl> plot
pl> prow dev$pix 256 device=lw
```

For more information on the IRAF graphics system, type the following command:

```
cl> help gio$doc/gio.hlp file+
```

For help about **psikern**, use the following command:

```
cl> help psikern
```

As with any STSDAS software, if you have questions or comments, please contact the STSDAS Helpdesk at the Space Telescope Science Institute. (help@stsci.edu).

Testing STSDAS

Once STSDAS has been loaded from the export tape and rebuilt, some of the basic functions of STSDAS as well as a few of the device-dependent tasks (e.g., plotting) should be exercised. When testing device-dependent functions such as plotting and image display, be sure that the IRAF environment variables that point to the device(s) are correct (e.g., **stdgraph** and **stdimage**).

Reading Exported Data Files

A few sample HST data files are provided in the directory `stsdas$data/fits`. These are in FITS disk format (with 512 byte records), and need to be expanded into IRAF or STSDAS disk format files before they can be accessed by STSDAS applications programs. You can use the STSDAS **fitsio** package to read these files. The expanded version of the files should be placed into the directory `stsdas$data/scidata` using the commands shown below:

```
cl> stsdas
st> fitsio
fi> cd stsdas$data/fits
fi> cl < read_fits.cl
```

Figure 2.8: Expanding Sample FITS Files

If your users want to use the **synphot** package, you will need to install the STDATA files described here

Additional sample data files and throughput tables for the HST components are needed by the **synphot** package; these files are provided as a separate release tape, or they can be retrieved using anonymous ftp. The installation instructions for these files—provided in FITS format—are provided in the *STDATA Installation Procedures* (Appendix C).

- If you have any problems using or installing STSDAS, contact the STScI Help Desk by sending e-mail to: help@stsci.edu

Multi-architecture Support for STSDAS

It is possible to support multiple architectures using a single source tree. If you wish to support, for example, both Solaris and SunOS architectures with a single source tree, you would follow these steps:

- Create a top-level directory for STSDAS
- Edit the `hlib$extern.pkg` file. (See “Pre-Installation Site Modifications” on page 16.)
- Install the STSDAS source code. (See “Installing Source Code” on page 17.)
- Install the binaries for the Solaris architecture. (See “Installing the Binaries” on page 19.)
- Install the binaries for the SunOS architecture. (See “Installing the Binaries” on page 19.)
- Modify the help and apropos databases. (See “The Help Database” on page 20. and See “The Apropos Task” on page 21.)

Building STSDAS from Scratch

If binaries for your computer's architecture are not available, then you will need to compile STSDAS from scratch.

Binaries for earlier versions of STSDAS for some architectures can be obtained from `http://ra.stsci.edu/GetSoftware.html.`

The system rebuild can be done with a batch procedure submitted while within the IRAF **cl**. Load the **tables** and **softools** packages, set the current directory to the top STSDAS directory, and execute the **mkpkg** task.

The first step in a total system rebuild is to insure that some system variables are set before proceeding. The following should be typed at the system level before proceeding:

```
% setenv IRAFARCH arch
```

where *arch* is your specific architecture, e.g., `sparc` for a SparcStation.

```
% setenv iraf /path/iraf/
```

where *path* is the directory path to the top-level IRAF directory.

```
% source $iraf/unix/hlib/irafuser.csh
```

This sets up some other environment variables needed to compile under IRAF. You may set these up in your `.login` file so that they will be available.

Also, ensure that the directory that contains the local Unix commands (usually `/usr/local/bin`) is included in your PATH environment variable. If it is not, you can add it to your path by typing the following:

```
% setenv PATH /usr/local/bin:$PATH
```

Before attempting the total system rebuild, you should check the soft link for the `bin` directory. STSDAS is shipped with a link for `bin` pointing to `bin.generic`. This should be changed so that it points to the appropriate `bin` for your architecture. To do this simply type the following command from the STSDAS top-level directory:

```
% mkpkg arch
```

where *arch* is your specific architecture. (A list of architectures is provided in Table 2.1). For example, for a SparcStation, you would type:

```
% mkpkg sparc
```

You will get a warning message about a full “sysgen” needing to be done, but that is normal.

Architecture	***Command***
Sun SunOS 4.x	`mkpkg sparc`
Sun Solaris 2.5.1	`mkpkg ssun`
Sun Solaris 2.8	`mkpkg ssun`
PC platforms	
Linux Slackware 3.3	`mkpkg linux`
Linux Red Hat 6.1	`mkpkg redhat`
FreeBSD 2.2.5	`mkpkg freebsd`
Alpha with Digital Unix 5.1(OSF/1)	`mkpkg alpha`
Hewlett-Packard with HP-UX 10.20	`mkpkg hp700`
SGI IRIX 6.5	`mkpkg irix`

Table 2.1: Supported Architectures

Warning for Unix users:
Several architectures supported by IRAF have some idiosyncrasies:

SystemV Unix (Solaris, IBM RISC,IRIX).: For some reason the link for `bin` in `tables$` is not correctly resolved on System V Unix machines using **rtar**, causing the command `mkpkg arch` to fail. Create this link by hand using the **ln -s** command at the Unix shell level:

```
% rm bin
% ln -s bin.arch bin
```

This problem also causes the libraries in `tables$lib` to incorrectly resolve. This can cause the libraries being compiled by mkpkg to not be created and can yield unresolved references in links further on. Delete these links and things will work.

SUN OS, HP-UX: Some of the code in STSDAS 2.3 is written in ANSI C. As such, this requires the SUN ANSI compiler, **acc**, or the HP-UC compiler, **c89**, be used rather than the usual **cc** compiler. In IRAF 2.11.3, this can be set by setting the following environment variable:

```
% setenv XC-CC acc -or-
% setenv XC-CC c89
```

Also, for SunOS, the ANSI library **libansi.a**, needs to be in your **LD_LIBRARY_PATH.** A suggested **LD_LIBRARY_PATH** is:

```
% setenv LD_LIBRARY_PATH /usr/lang/SC1.0/ansi_lib:/usr/lang/SC1.0:
```

The Unix system rebuild can be done with a batch procedure submitted while within the IRAF **cl.** Load the **stsdas** and **softools** packages, set the current directory to the top STSDAS directory, and execute the **mkpkg** task:

```
cl> stsdas
st> softools
so> cd stsdas
so> mkpkg -p tables -p stsdas >& spool &
```

Figure 2.9: Rebuilding the Package as IRAF Background Job

This will run the **mkpkg** task as a background process and put all output and errors into the `stsdas$spool` file.

The **mkpkg** program generates a long output file describing all steps taken. To reduce this log to the pertinent information about the success of you installation, re-run the **mkpkg** task with the summary option.

```
cl> softools
so> cd stsdas
so> mkpkg summary > stsdas.summ
```

Figure 2.10: Running mkpkg with Summary Option

If you have any problems installing or using STSDAS, contact the STScI Help Desk via e-mail to: help@stsci.edu

CHAPTER 3:

STSDAS Site Manager's Reference

In This Chapter...

This chapter explains information needed to maintain and troubleshoot the STSDAS package. This is the information that was previously provided in the *Site Manager's Guide* in STSDAS version 2.2.

User Account Privileges and Quotas

STSDAS does not require any special privileges or quotas when operated in a Unix environment. However, if you are using a workstation without additional space, you will probably have problems compiling and linking three calibration tasks; **fos**, **hrs** and **hsp**. We suggest a minimum of 16 megabytes of static storage space or 20 to 24 megabytes of swap space to ensure a successful mkpkg.

STSDAS Directory Structure

STSDAS is organized in a hierarchical directory structure (see Figure 3.1) that reflects the organization seen by users of the system. Since STSDAS is a part of IRAF, we have adopted the *package* structure of IRAF to organize the application functions available to users. When STSDAS is installed, the system manager controls the name of the directory in which the structure is rooted; typically, this is `STSDAS`, and we will assume this in the discussions that follow. All directory path names in STSDAS are relative to this top level directory. All STSDAS directories have names assigned as IRAF environment variables. To go to any STSDAS application package directory, just use the **cd** command in IRAF and specify the name of the package (e.g., `cd fourier`).

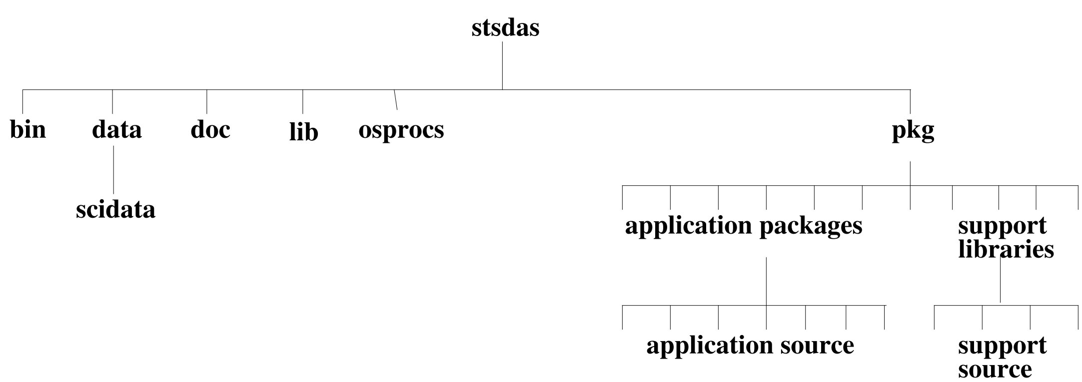

Figure 3.1: Overall STSDAS Directory Structure

Applications Software Directories

Each STSDAS package has a corresponding directory in the host file system that is a subdirectory of the `pkg` directory; the name of the subdirectory is the same as the name of the applications package. These package-level directories contain all the run-time files that may be needed by tasks within that package, including parameter files, help files, and a `mkpkg` file (used for recompiling and relinking the package). For example, suppose there were an STSDAS applications package called **applpkg**.

The files listed in Table 3.1 are stored in the package level directory (`applpkg` in this example):

File	*Description*
`*.cl`	CL scripts. There is one that defines the package, and one for each of the logical tasks within the package.
`*.par`	Parameter files for the logical programs in the package. Generally, there will be one for each source-level subdirectory.
`*.hd`	The help data base index for this package (one per package).
`*.men`	The help menu file for this package (one per package).
`*.hlp`	Help text file for this package as a whole.

Table 3.1: Files in Package Level Directory of an Application

File names given here are in the syntax of IRAF's virtual filename mapping. To avoid confusion, we recommend that you always view the contents of the STSDAS system while running the IRAF CL and using the `cd` command to change directories.

Beneath most applications package directories are subdirectories that contain the source code for the various tasks within that package. Each major task resides in its own directory. Once STSDAS has been installed (recompiled and relinked), the source code can be removed to conserve disk space (see "Saving Space" on page 43 for instructions).

File Name	*Contents*
*.c	C source code for program
*.f	Fortran source code for program
*.x	SPP source code for program
mkpkg	The IRAF mkpkg file used to update the object library for program and to relink package executable image.
program.mlb	**mkpkg** keeps track of changes to source code in this file. This file will not exist unless **mkpkg** was run and is VMS specific.
program.o	Object library for program. This file is not exported; it will not exist unless **mkpkg** has been run.

Table 3.2: Common Files in STSDAS Task Directories

Help text files for each of the logical tasks in the package are contained in the doc subdirectory of the package.

The executable images of the packages are located in stsdas$bin as x_pkg.e.

A complete directory tree for STSDAS is shown in Figure 3.2.

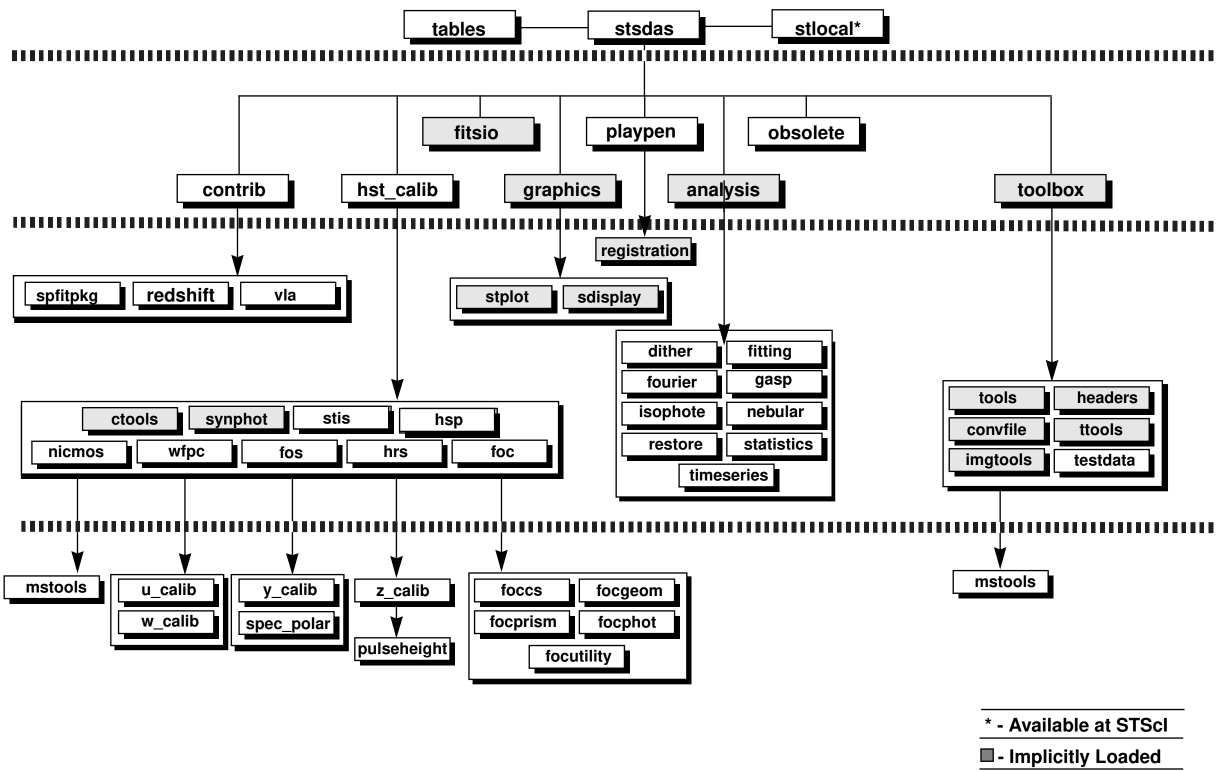

Figure 3.2: STSDAS Version 2.0 Organization

Support Software Directories

The applications directories described above contain the run-time connections to IRAF and the source code for the STSDAS applications. There are a number of other directories needed to support STSDAS. Most important is the subdirectory structure in which all STSDAS I/O and utility software is stored; this structure is rooted in the `lib` subdirectory of STSDAS (`stsdas$lib/`). This subdirectory contains several libraries, and for each of these there is a related subdirectory in which the corresponding source code resides.

Directory	*Contents*
`applib`	Applications subroutines
`cvos`	C programming interface
`f77util`	F77VOS utilities
`hstio`	HST data file I/O C interface
`iraf77`	F77VOS interface
`stalone`	F77 stand-alone interface
`synphot`	Synthetic photometry interface

Table 3.3: STSDAS I/O and Utility Libraries

In addition, several utility libraries for tables I/O are used by STSDAS that are in the TABLES package, these are listed in Table 3.4.

Library	*Contents*
`display`	Terminal display routines
`gflib`	Front end to gilib
`gilib`	STSDAS IEEE and GEIS format subroutines
`tbtables`	STSDAS tables I/O subroutines
`uttables`	STSDAS table utilities
`stxtools`	STSDAS special applications tools

Table 3.4: TABLES I/O and Utility Libraries

STSDAS will *not* compile without TABLES being installed first!

Exported Data Directories

A few data files have been sent along with the STSDAS installation, and these are found in the `stsdas$data/` directory tree. The directory stsdas$data/fits contains these files with the extension .fits. Chapter 2 describes how this data is to be read and installed into the scidata directory. After the STSDAS package has been loaded, this area can be referred to with IRAF environment variable scidata.

Sample calibration and image files for each of the major instruments on HST are provided as a separate release tape. The installation instructions for these files, provided in FITS format, are provided in the *STDATA Installation Procedures*.

STSDAS and IRAF System Directory

The STSDAS directory contains files that establish the entire STSDAS package structure in the IRAF environment (Table 3.5).

File	***Purpose***
`stsdas.cl`	Primary CL script that defines all STSDAS packages
`stsdas.hd`	Primary help data base index for all STSDAS help files
`stsdas.men`	The help menu file for the STSDAS package itself
`stsdas.hlp`	Highest-level help file for STSDAS (as a whole)

Table 3.5: Files Establishing STSDAS Package Structure

In addition, the directory stsdas$lib contains the file mkpkg.inc. This file contains the macro definitions for the IRAF **mkpkg** facility and is used when **mkpkg** compiles or links STSDAS programs.

Rebuilding STSDAS Applications

STSDAS applications are structured so that they can be rebuilt piecewise or as a whole using the **mkpkg** utility provided with IRAF. Users who wish to rebuild IRAF/STSDAS applications should familiarize themselves with specifics about the use of **mkpkg** as described in the IRAF help documentation for **mkpkg**. There are **mkpkg** files in various directories at three levels within the STSDAS applications hierarchy: Each applications program directory contains a **mkpkg** file that will rebuild that particular library and relink the package executable image. These directories are fourth-level nodes, i.e., `stsdas$pkg/*/*/`.

- Each STSDAS package directory has a **mkpkg** file that will rebuild all the libraries in the package and relink the package executable image. Package directories are third-level nodes, i.e., `stsdas$pkg/*/`.
- There is a **mkpkg** file in the directory `stsdas` that will rebuild the entire STSDAS system. This is a first-level node, i.e., `stsdas/`.

When rebuilding an STSDAS task or package, **mkpkg** must be told to use the appropriate environment variables and libraries. Therefore it is necessary to run the task **mkpkg** with the commands:

```
cl> mkpkg -p tables -p stsdas >& spool &
```

when making the entire STSDAS system from the `stsdas/` directory; or the command

```
cl> mkpkg update -p stsdas
```

when making a single package or application program. This will run the mkpkg task as a background process and put all output and errors into the stsdas$spool file.

Package executables are rebuilt on a time scale that is typically several minutes, although this can vary widely. STSDAS **mkpkg** files can either perform compilation *and* linking or just a relink (by typing `linkonly` at the end of the **mkpkg** command). Some programs contain a large number of subroutines; others contain few.

Recompilation and relink of the entire system is a lengthy process. It is *strongly* recommended that an entire system recompilation and relink be done as a batch job running over the weekend. It is possible that an overnight batch job will not be finished when the normal daily computer usage begins.

Saving Space

As with IRAF, a mechanism exists to remove the source files and other files not needed to actually run STSDAS. This procedure should only be used if disk space is a problem, you will not be doing any development using STSDAS, and you have an alternative method for getting revised object libraries and executables when system patches are made.

To remove all but the essential files, you need to run the **mkpkg strip** command from the top level of **stsdas**.

```
cl> cd stsdas
cl> mkpkg strip -p stsdas
```

STSDAS Disk Data Files

STSDAS disk data files are patterned after the extended FITS format described by Greisen and Harten (1981, *Astron. Astrophys. Suppl. Ser.* **44**, 371). In this scheme, every file has a *group* structure in which it is possible for a file to contain a set of related observations that share a number of common header parameters, but where each individual observation has its own specific header parameters. In the STSDAS implementation of group format data, the common header parameters are stored in an ASCII file consisting of fixed-length 80-byte records. The parameters are identified by keywords so the file contains records that look like FITS header records.

A sample STSDAS file header is shown in Figure 3.3.

```
SIMPLE = F /
BITPIX = 32 /
DATATYPE= 'REAL*4 ' / NOT a FITS standard
NAXIS = 2 /
NAXIS1 = 55 /
NAXIS2 = 55 /
GROUPS = T / All STSDAS data in group format
PSIZE = 512 / size of group parameter block in bits
GCOUNT = 1 / File has only 1 group
PCOUNT = 12 / There are 12 group parameters (2D standard)
PTYPE1 = 'DATAMIN ' / Group parameter descriptors
PDTYPE1 = 'REAL*4 ' /
PSIZE1 = 32 /
 .
ORIGIN = 'UH-IFA ' / Univ.Of.Hawaii Inst.For.Astronomy
INSTRUME= 'IRTF ' / NASA Infrared Telescope Facility
OBJECT = 'JUPITER '
DATE-OBS= '08/09/84 170708' / DD/MM/YY HHMMSS LOCAL
 .
 .
 .
HISTORY OBSERV: 1010 U.T.: 30708
HISTORY APERTURE: 0.0 STEP: 0
 .
 .
 .
HISTORY UPDATED at 19-SEP-1985 15:03:32.89 by FITSRD
HISTORY INTAPE = MTA0: /> TAPE DRIVE (CHOOSE ONE OF THE FOLLO
HISTORY SKFILE = 0 /> FILES TO SKIP BEFORE READING
HISTORY OUTFILE = TPF1 /> NAME OF OUTPUT DISK FILE
END
```

Figure 3.3: Example of STSDAS File Header

Each header file contains the keyword GROUPS and its value is always TRUE. The keyword GCOUNT is used to define the number of groups in the data file. GCOUNT can certainly be 1 (often this is the case), but the file is still considered to be in group format.

Please bear in mind that this is the *internal* disk format for STSDAS. This is *not* the format that will be exported to HST observers. For FITS export tapes, we will map the STSDAS internal group format onto an extra data dimension, and the values in the group parameter block will be appended in a FITS table extension. Alternatively, users may choose to have their multi-group data written as separate FITS files.

STSDAS Data File Names

By default, the name of an STSDAS data header file is a root name followed by the extension `.hhh`, although any three character extension ending in the letter `h` is a valid header name. The actual data array described by this type of header is stored in a binary file with the same root name as the corresponding header and a default extension of `.hhd`. Again, any three character extension is valid so long as the last letter is `d`. The names of each file in an STSDAS header and data file pair must be the same except for the last `h` and `d`. The following are valid STSDAS data file names:

- `mydata.hhh` and `mydata.hhd`
- `yourdata.abh` and `yourdata.abd`
- `otherdata` (implicitly refers to `otherdata.hhh` and `otherdata.hhd`)

When a user specifies a data file name to an STSDAS program, it is necessary to give only the root name; STSDAS will take care of attaching the default extensions `.hhh` and `.hhd`. If extensions other than `.hhh` and `.hhd` are used, they must be specified explicitly. Each data file pair is considered as a set, so that users can specify either the name of the header file or the data file and STSDAS will know where to find the other part.

Group Parameters

The header information that pertains to a specific group within the data file is stored in the binary file. The group parameters follow immediately after the data for each group (group parameters need not begin on a record boundary). The group parameters can be of different data types, and the names, data types, and sizes of group parameters are given in the ASCII header by triplets of keywords (`PTYPEn, PDTYPEn, PSIZEn`) that function as descriptors.

Coordinate System Parameter Matrix

Among the group parameters are values that describe the physical coordinate system for the data in that group. STSDAS has departed from the FITS standard somewhat in order to provide a more general means of describing physical coordinate systems (although an extension to the FITS standard is currently under consideration). As opposed to the purely linear `CDELT` and `CROTA` FITS parameters, the relation between the pixel coordinates and the physical coordinates is given in STSDAS by a *coordinate description* (`CD`) matrix. The elements of the `CD` matrix are the partial derivatives of the physical coordinates with respect to the pixel coordinates. For example, for a tangent-plane projection with coordinates (*l*,*m*) and a small field of view (so that a small angle approximation is valid) we can write

$$l = \frac{\partial l}{\partial x}(x - x_0) + \frac{\partial l}{\partial y}(y - y_0)$$

$$m = \frac{\partial m}{\partial x}(x - x_0) + \frac{\partial m}{\partial y}(y - y_0)$$

where x and y are the pixel coordinates in the image. The `CD` matrix gives the partial derivatives of l and m with respect to x and y, which, for a simple rotation, have the following representation:

$$\frac{\partial l}{\partial x} = D_1 \cos\theta \qquad \frac{\partial l}{\partial y} = -D_1 \sin\theta$$

$$\frac{\partial m}{\partial x} = -D_1 \sin\theta \qquad \frac{\partial m}{\partial y} = D_2 \cos\theta$$

where D_1 is the sampling interval along the x-axis (in degrees) and D_2 is the sampling interval along the y-axis (in degrees). θ is the rotation angle between the sampling grid and the physical coordinate system.

The resulting FITS-like keywords that describe the coordinate system are stored in the group parameter block as follows:

$$CD1\text{-}1 = \frac{\partial l}{\partial x} \qquad CRPIX1 = x_0$$

$$CD1\text{-}2 = \frac{\partial l}{\partial y} \qquad CRPIX2 = y_0$$

$$CD2\text{-}1 = \frac{\partial m}{\partial x} \qquad CRVAL1 = \alpha_0$$

$$CD2\text{-}2 = \frac{\partial m}{\partial y} \qquad CRVAL2 = \delta_0$$

$$CTYPE1 = 'RA'$$

$$CTYPE2 = 'DEC'$$

where α_0 is the right ascension at (x_0,y_0) and δ_0 is the declination at (x_0,y_0). In this representation the `CD` matrix has replaced the standard FITS keywords `CDELTn` and `CROTAn`. This `CD` matrix is clearly extensible to higher dimensions and can be used to account for non-linearities in the coordinate system, although these capabilities have not yet been incorporated into STSDAS.

Updates and Bug Fixes

As with any software system, there will be patches and enhancements made between official releases of STSDAS. These will be made available through anonymous ftp. From time to time we advise that you check the README file in the directory `software/stsdas/`.

For sites with ftp, the anonymous directory can be reached by:

```
ftp ftp.stsci.edu
```

or

```
ftp 130.167.1.2
```

For SPAN sites, the following are examples of different accesses to the anonymous ftp on stsci.edu:

```
$ type east"stsci.edu!anonymous name"::"STEIS.DIR"
$ directory east"stsci.edu!anonymous name"::"Stsci"
$ copy east"stsci.edu!anonymous name"::"STEIS.DIR" steis.dir
```

For Bitnet sites, send mail containing your ftp commands to "BITFTP@PUCC" where your mail message may be:

```
FTP stsci NETDATA
USER anonymous
cd Software/Stsdas
DIR
GET README
QUIT
```

More information about the Bitnet ftp server can be obtained by sending a mail message of "HELP" to "BITFTP@PUCC".

APPENDIX A:

Mounting CD-ROMS for the GASP Package

The Guide Star Catalog CD-ROMs are mounted on two Sun CD-ROM drives connected to a SparcStation running SunOS 4.1 or 4.1.1. (It should also work with SunOS 4.1.1b). The only really annoying feature is that eject doesn't work properly with an argument of "sr1". You have to use /dev/rsr1 instead.

You can probably also use a Sun CD attached to a SparcServer and only have one CD mounted at a time. The "sun4" version of the CD ROM driver (`sr.o`) that comes with versions 4.1 and 4.1.1 has some major problems with handling more than one CD-ROM drive on a single SCSI bus. This may also apply to more than one CD-ROM drive on different SCSI buses.

There is also a minor bug in the HSFS driver from Sun in the way it handles the Extended Record Attributes in the ISO 9660 format that are present on the GSC disks. You start getting messages about not being able to handle a certain record type when you start using `ls` on the mounted disk. To fix this problem you need to patch the file `/usr/sys/sun4c/OBJ/hsfs_node.o` by doing the following:

1 • Login as `root` or `su`

2 • cd to `/usr/sys/sun4c/OBJ`

3 • Make a copy of the file `hsfs_node.o` so that you can go back to it if necessary.

4 • Use the debugger as follows (type these lines as shown):

```
adb -w hsfs_node.o
hs_parsedir+0x9c?W 0ba0da0e6
hs_parsedir+0x14c?W 0ba0da0e6
$q
```

By doing this you will modify two instructions in the part of the driver that parses the directory structure. Basically it causes it to ignore extra information.

If you are running SunOS 4.1.1 on a SparcStation, you can pick up a patched copy of `hsfs_node.o` via anonymous ftp from hoth.stsci.edu in the directory `pub`. *Remember to use binary mode when copying the file.*

You will need to check your kernel configuration file and make sure it contains a line like:

```
options   HSFS           # Support for ISO 9660 CDs
```

and a line like:

```
 disk sr0 at scsibus0 target 6 lun 0 # CD-ROM device
```

If you are planning to connect two CD-ROM drives, you will ether need second SCSI interface for the machine (assuming it is not an SLC), to have a Sun supported configuration, or change the SCSI target on the second CD, and connect them to the same SCSI bus. I made `sr0` target 6 and `sr1` as target 5. The lines in to kernel configuration file look like:

```
 disk sr1 at scsibus0 target 5 lun 0 # CD-ROM device
```

After you do this you will need to rebuild the kernel, and make sure you save an old copy of the kernel before installing the new one in case you have to go back to the old version.

Once the new kernel is installed, you will need to reboot the machine.

To mount one of the Guide Star Catalog (GSC) disks, you will need to put it in one of the supplied CD caddies that comes with the CD ROM drive, and insert it into the CD-ROM drive. To mount the disk you will need to login as root or su and issue a `mount` command, for example:

```
mount -t hsfs -o ro,noquota /dev/sr0 /gsc/vol1
```

Where `/dev/sr0` is the device name and `/gsc/vol1` is the directory you are mounting it on. The directory you are mounting it on must exist before you issue the command. A similar command can be used to mount the second volume in another drive.

What we do is do here is mount the disks at system boot assuming the correct volumes are in the correct drives. To do this you will need to modify `/etc/fstab` and add the following lines:

```
/dev/sr0 /gsc/vol1 hsfs ro,noquota 0 0
/dev/sr1 /gsc/vol2 hsfs ro,noquota 0 0
```

and then add the following line to `/etc/rc.local` somewhere near where the NFS partitions are mounted:

```
intr -a mount -vat hsfs
```

If you have any problems you can probably find help in the System and Networking Administration Guide and in the mount(8), eject(1), fstab(5), and sr(4s) man pages.

The file `hsfs_node.o-4.1.2` is identical to the 4.1.3 version. The 4.1.1 and Solaris 2.x versions of the driver are different. To use the pre-patched version you should copy `hsfs_node.o-4.1.2` from the anonymous ftp host hoth.stsci.edu using binary mode and put it in `/usr/sys/sun4c/OBJ/hsfs_node.o`, rebuild the kernel, and then reboot. Once the kernel module has been patched, you should not have to make any additional changes using `adb`. This patch should not conflict with any other Sun patches that are presently available.

The ISO9660 patch that I generated for Solaris 2.0 based on the 4.1.3 patch also works in Solaris 2.1. With the problem supposedly fixed in Solaris 2.2 due out next month, I doubt that there will be much demand for Solaris 2.0 and 2.1 support. A patched version of the HSFS driver is available via anonymous ftp from hoth.stsci.edu in the directory `pub`.

The HSFS driver for Solaris 2.2 has changed, but the problem is still present. Under Solaris 2.2, you need to modify `/etc/system` and add the following lines at the bottom of the file and then reboot:

```
set hsfs:ide_prohibited=0xe6 set hsfs:hde_prohibited=0xe6
```

This changes two masks in the driver to allow it to access our CD-ROMs. These changes will probably also work in Solaris 2.3 when it is released.

The remainder of this appendix is the file `readme.txt` in the top level directory on the GSC CD-ROM. It describes the directory structure of the disk and provides some information about the catalog.

The Guide Star Catalog

The remainder of this appendix is from the file `readme.txt` in the top level directory on the GSC CD-ROM. It describes the directory structure of the disk and provides some information about the catalog.

Copyright, 1989, Association of Universities for Research in Astronomy, Inc.

This set of CD-ROMs contains the Guide Star Catalog—Version 1, with an issue date of 1 June 1989. The Guide Star Catalog (GSC) was prepared by the Space Telescope Science Institute (STScI), 3700 San Martin Drive, Baltimore, MD 21218, USA. STScI is operated by the Association of

Universities for Research in Astronomy, Inc. (AURA), under contract with the National Aeronautics and Space Administration (NASA).

The Guide Star Catalog is subdivided into regions, which are bounded by small circles of right ascension and great circles of declination, and which are numbered consecutively from 0001 to 9537. Data for each region are stored as separate files; these files are contained in subdirectories, each of which subtends a 7.5 degree zone of declination.

The Guide Star Catalog is distributed as a two CD-ROM set, divided at a declination of -7.5 degrees. This description file and supporting tables are duplicated on both discs.

All data files (i.e., with the exception of this file and the directory files) are in FITS (Flexible Image Transport System) table format.

The root directory contains the following files:

- README.TXT;1 - This file.
- GSC - Directory for GSC region files.
- TABLES - Directory for GSC supporting tables.

Directory GSC contains subdirectories for the 7.5 degree zones in declination; these subdirectories in turn contain the GSC region files in FITS format for the respective zone, with file identifiers of the form nnnn.GSC;1, where nnnn is the 4-digit decimal region number, with leading zeroes used as required to fill the field. The subdirectories are names named as follows:

Table A.1: Subdirectories

Subdirectory	*Declination*				*Regions*		*Disc*
	From		*To*		*From*	*To*	
N0000	+00d	00m	+07d	30m	0001	0593	1
N0730	+07d	30m	+15d	00m	0594	1177	1
N1500	+15d	00m	+22d	30m	1178	1728	1
N2230	+22d	30m	+30d	00m	1729	2258	1
N3000	+30d	00m	+37d	30m	2259	2780	1
N3730	+37d	30m	+45d	00m	2781	3245	1
N4500	+45d	00m	+52d	30m	3246	3651	1
N5230	+52d	30m	+60d	00m	3652	4013	1
N6000	+60d	00m	+67d	30m	4014	4293	1
N6730	+67d	30m	+75d	00m	4294	4491	1
N7500	+75d	00m	+82d	30m	4492	4614	1
N8230	+82d	30m	+90d	00m	4615	4662	1
S0000	-00d	00m	-07d	30m	4663	5259	1
S0730	-07d	30m	-15d	00m	5260	5837	2
S1500	-15d	00m	-22d	30m	5838	6411	2
S2230	-22d	30m	-30d	00m	6412	6988	2
S3000	-30d	00m	-37d	30m	6989	7522	2
S3730	-37d	30m	-45d	00m	7523	8021	2
S4500	-45d	00m	-52d	30m	8022	8463	2
S5230	-52d	30m	-60d	00m	8464	8839	2
S6000	-60d	00m	-67d	30m	8840	9133	2
S6730	-67d	30m	-75d	00m	9134	9345	2
S7500	-75d	00m	-82d	30m	9346	9489	2
S8230	-82d	30m	-90d	00m	9490	9537	2

Directory TABLES contains the following supporting files for the GSC, written in FITS table format:

- COMMENTS.TBL;1 - Introduction and general comments.
- PLATES.TBL;1 - Information on the plates used in the GSC.
- PROCESS.TBL;1 - Image processing parameters.
- ASTR_CAL.TBL;1 - Parameters of astrometric calibrations.
- PHOT_CAL.TBL;1 - Parameters of photometric calibrations.
- C_UP_POP.TBL;1 - Catalog update population statistics.
- ST_POP.TBL;1 - Population statistics for stars.
- NS_POP.TBL;1 - Population statistics for non-stars.
- REGIONS.TBL;1 - Boundaries of GSC regions.
- C_RE_POP.TBL;1 - GSC region population statistics.
- LG_REG_X.TBL;1 - Index to large regions.
- SM_REG_X.TBL;1 - Index to small regions.
- XREF_P2R.TBL;1 - Cross-reference table, plates to regions.
- XREF_R2P.TBL;1 - Cross-reference table, regions to plates.

Additional information on the Guide Star Catalog may be found in accompanying scientific publications and in comments in the FITS headers of the supporting tables.

APPENDIX B:

Software Problem Report

To submit a Software Problem Report, you can either use the **spr** task, send e-mail to help@stsci.edu, or you can complete the following form and mail it to us.

STSDAS SPR Form

Reported by:

Package/Task:

IRAF Version:

STSDAS Version:

Summary:

Description of Problem:

Send SPR Form to:
STSDAS Group
Space Telescope Science Institute
3700 San Martin Drive
Baltimore, MD 21218

APPENDIX C: STDATA Installation Procedures

In This Appendix...

Data for the STSDAS calibration tasks are provided on a separate tape from the STSDAS software system. Because we recognize that not all sites will want data for all instruments, each instrument is represented on the tape as a separate file, separated by tape marks.

This document describes the steps needed to load the data and convert it from FITS to your machine dependent file format for use in STSDAS.

The following data files are contained on the tape in tar format:

1 • Archive Utilities

2 • Table of Contents

3 • FOC

4 • FOS

5 • HRS

6 • HSP

7 • WFPC

8 • Synphot HST throughput data

9 • Synphot Non-HST throughput data

10 • Synphot Stellar model data

Selecting the Top Directory

Although the STDATA may be installed as part of the `stsdas/data` directory, we recommend installing it as a separate directory structure and recommend naming the top directory `STDATA`. This will enable you to more easily make updates to the STSDAS system or the data in STDATA. This is the method used in the examples.

Reading the Tape

The steps needed to read the STDATA tape are decribed below

The first file on the tape is a tar containing compress utilities for VMS users so that they can uncompress the compressed tar files on the remainder of the tape. This file may be skipped because Unix has an `uncompress` command.

1 • The second file on the tape is a tar file containing a single file, `contents.lst`. This file is an ASCII text file containing the names and sizes of the other files on the tape. Load this file into the STDATA directory and print it before continuing.

```
% mt -f /dev/rmt8 rew
% mt -f /dev/nrmt8 fsf 1
% tar -xvf /dev/rmt8
% more contents.lst
```

Figure B.11: Reading the contents.lst File

2 • For each instrument or other test data directory that you want to load, you need to:

- Create an appropriate subdirectory
- Skip to the desired file
- Read the file from tape using the `dd` command
- Pipe the output from `dd` through `uncompress`
- Pipe the output from `uncompress` through `tar`

In the example below, *XXX* is the instrument or test directory you want and # is the number of files to skip. Note that for the synphot data, all 3 files should be taken from the tape.

```
% mkdir XXX
% cd XXX
% mt -f /dev/rmt8 rew
% mt -f /dev/nrmt8 fsf #
% dd if=/dev/nrmt8 bs=20b | uncompress | tar -xvf -
```

Figure B.12: Reading Data From the Tape

For example, to read the High-Speed Photometer (HSP) data files, you need the sixth file from the tape (including the table of contents), the commands to do this are shown in Figure B.13.

```
% mkdir hsp
% cd hsp
% mt -f /dev/rmt8 rew
% mt -f /dev/nrmt8 fsf 5
% dd if=/dev/nrmt8 bs=20b | uncompress | tar -xvf -
```

Figure B.13: Reading the HSP Data Files

Once the data has been read from the tape, return to your IRAF directory and start the **cl**.

FTP tar Files

If you are getting the tar files from the anonymous ftp account, you can get the instrument compressed tar files, each named after the instrument, in directory `software/stsdas/testdata`. Synphot data is in directory `software/ stsdas/refdata/synphot` in 3 compressed tar files.

For each instrument or other test data directory that you want to install, you need to:

1 • Create the appropriate subdirectory.

2 • Use the Unix command `cat` to concatenate the file.

3 • Pipe the output of that through `uncompress`.

4 • Pipe the output of that through the `tar` command.

In the example below (Figure B.14), *XXX* is the instrument or test directory you want and # is the number of files to skip. Note that for the Synphot data, all 3 files should placed in the same directory. It is assumed in the example that the compressed tars reside on `/usr/iraf/stdata` and the resultant files will be placed in subdirectories of this diorectory.

```
% pwd
    /usr/iraf/stdata
% mkdir XXX
% !mv XXX.tar.Z ./XXX
% cd XXX
% cat XXX.tar.Z | uncompress | tar -xvf -
% rm XXX.tar.Z
```

Figure B.14: Syntax for Reading the Tar Files in Unix

For example, to read the files for the High-Speed photometer, the commands would be those used in Figure B.15.

```
% pwd
   /usr/iraf/stdata
% mkdir hsp
% !mv hsp.tar.Z ./hsp
% cd hsp
% cat hsp.tar.Z | uncompress | tar -xvf -
% rm hsp.tar.Z
```

Figure B.15: Reading the Tar Files for the HSP

Once the data has been read from the compressed tar files, return to your IRAF directory and start the **cl**.

Converting the Data Format

Instrument Data Sets

A script, `readXXX.cl`, is provided for each instrument data set, where *XXX* is the directory name. This script is run within IRAF to convert the data from FITS format into the format used by the calibration routines. Note that **stsdas** and **fitsio** packages must be loaded for the scripts to work correctly. Figure B.16 shows how the script is used.

```
% cl
cl> cd /node/partition/stdata/XXX
cl> task readXXX = readXXX.cl
cl> stsdas
st> fitsio
fi> readXXX
```

Figure B.16: General Syntax for Converting File Formats

Again, using HSP as the example, the actual commands for the directory located on `/usr/iraf/stdata/` are shown in Figure B.17:

```
% cl
cl> cd /usr/iraf/stdata/hsp
cl> task readhsp = readhsp.cl
cl> stsdas
st> fitsio
fi> readhsp
```

Figure B.17: Converting Files for HSP

Note: Once the FITS files have been converted, they remain on disk. You may want to delete these files to save disk space.

Synphot Data Set

Setting the Top Directory

The synphot tasks assume that all the synphot reference files are stored under a single top level directory. This directory is referred to inside STSDAS by the logical name `crrefer`. This directory may be anywhere you have sufficient space to install the reference files (approximately 400 megabytes is required for the full installation), but we recommend that it not be placed as subdirectory of the STSDAS or TABLES source code. This will make it easier to update STSDAS without needing to reinstall the Synphot data. Once the top directory is created, the environment variable `crrefer` should be set in your `hlib$extern.pkg` file. To set `crrefer$` add a command similar to the following to the file:

```
set crrefer = "/your/path/name/to/refer/"
```

The trailing slash is important, so do not omit it.

The Synphot data should be downloaded from our anonymous ftp at:

`http://www.stsci.edu/ftp/software/stsdas/refdata/synphot/`

If you do not have access to anonymous ftp, you can contact us at `help@stsci.edu}`, and ask for a tape containing the necessary files. There are four compressed tar files containing the data and this installation guide. The first tar file contains the Synphot component throughput tables, the second contains various observed and modelled spectral coatalogs, the third contains the 1993 Kurucz model stellar spectra, and the fourth contains the HST calibration standard spectra.

First, place the compressed tar files in the top level directory you created in the first section. Then, uncompress and untar the tar files. On a Unix system, the following commands will accomplish this.

```
% uncompress synphot1.tar.Z
% tar -xvf synphot1.tar

% uncompress synphot2.tar.Z
% tar -xvf synphot2.tar

% uncompress synphot3.tar.Z
% tar -xvf synphot3.tar

% uncompress synphot4.tar.Z
% tar -xvf synphot4.tar
```

The tar file `synphotpsf.tar.Z` contains the psf images used with the simulators package of synphot. If you are not planning to use this package, you do not need to install it. The tar file should be copied to the `stsdas$data/scidata` directory of stsdas, uncompressed, and untarred.

Type the following commands when in stsdas:

```
cl> copy /your/path/to/synphotpsf.tar.Z scidata$
cl> cd scidata$
cl> !uncompress synphotpsf.tar.Z
cl> rtar -xvf synphotpsf.tar
```

Updating the Synphot Parameter File

We have converted all the Synphot files to FITS format. For the most part, this change is transparent to you, but in order to have Synphot use the new files, you must edit the `refdata` parameter set. The new parameter set should have the values:

```
area = 45238.93416 (unchanged)
grtbl = mtab$*_tmg.fits
cmptbl = mtab$*_tmc.fits
```

Each user can edit the parameter set with the command

```
cl> epar refdata
```

in IRAF. Or, to make the change for all users on the system, modify the file `synphot\$refdata.par` with a text editor. (The name is an iraf logical. You can see the name on your system by loading the `hstcalib` and `synphot` packages in iraf and typing

```
cl> path synphot\$refdata.par
```

When Disaster Strikes

If you encounter problems installing the Synphot data files, we encourage you to contact us via the STSDAS help desk

help@stsci.edu

Index