PART I:

Introduction to HST Data

Chapter 1: Getting Your Data

Chapter 2: HST File Formats

Chapter 3: STSDAS Basics

- Introduction to HST Data

Chapter 1
Getting HST Data

In This Chapter...

Archive Overview / 1-1
Getting Data with StarView / 1-4
Identifying Calibration Reference Files / 1-19
Reading HST Data Tapes / 1-22

This chapter describes how to obtain Hubble Space Telescope (HST) data files. Both Guest Observers (GO) and Archival Researchers can receive data in either of two ways:

- Electronically over the Internet from the Hubble Data Archive, where data are stored immediately after they pass through HST pipeline processing.
- On a data tape generated from the Archive.

To retrieve data electronically you must first register with the Archive. If you wish to retrieve proprietary data associated with a particular proposal you must also request authorization to retrieve data from that proposal. Tapes, if requested, are sent to Principal Investigators automatically. This chapter explains how to search the Archive for datasets of interest, how to retrieve data electronically from the Archive, and how to read a typical GO data tape.

1.1 Archive Overview

STScI maintains an Archive of all HST data and their associated calibration files. The Archive holds all HST observations ever made and provides a database that catalogs and describes these observations. A program called *StarView* acts as an interface to the Archive; it allows you to search the Archive and to retrieve data from it. StarView is the sole electronic source of proprietary data. A World Wide Web interface to the archive currently permits retrieval of public but not proprietary data.

1.1.1 Accessing the Archive

To retrieve data electronically you need to be able to access StarView (or the Archive web site if you are interested only in public data), and you need to be a registered user of the archive. Only *registered* users may retrieve data from the Archive. Proprietary data can be retrieved *only by authorized registered* users.

General Observers and Guaranteed Time Observers (GTO) normally have exclusive rights to their HST data for one year. However, all observations obtained under calibration proposals are immediately public.

Distributed Access

StarView operates most rapidly if you run it on your own machine, and it is available in executable form for several operating systems. To take advantage of the speed of distributed access, download the StarView software from our website:

`http://archive.stsci.edu/starview.html`

The web site contains directions and materials for installing StarView on your own computer. No compilation is necessary. Once StarView is installed, typing `xstarview` starts the program.

By running the `xstarview` client on your own computer, you avoid the overhead of running the software over the network, giving you a considerable speed improvement.

Remote Access

You can run StarView remotely at STScI, if it is not currently available on your local machine. Use `telnet` to connect to `archive.stsci.edu` and log in with the user name `guest` and password `archive`[1]. Once logged in, you can use StarView to peruse the database. Simply type `xstarview` for the X-Windows version or `starview` from the command line for the terminal version. The X-Windows version takes advantage of all of StarView's capabilities, including data previews; however, the terminal version will respond faster under remote access.

1. European archive users should generally use the ST-ECF Archive system via their web site at `http://archive.eso.org/`. Canadian users should request public archival data through the CADC web site; `http://cadc.dao.nrc.ca/`. Proprietary data is available only through STScI.

Web Access

STScI maintains a web site for current news, quick retrieval of special datasets, and retrieval of public data sets through online forms:

`http://archive.stsci.edu/`

Features of this web site include:

- Registration using online forms.
- Archive news.
- Form-based access to the Archive and the Digitized Sky Survey.
- A FITS Keyword Dictionary for interpreting header keywords.
- Archive documentation and policies.

1.1.2 Registering to Retrieve Hubble Data

The simplest way to register is through our World Wide Web form at the following location:

`http://archive.stsci.edu/registration.html`

You can also register by typing the `register` command while logged onto `archive.stsci.edu`.

Your retrieval account will be activated within two working days, and you will receive your password via E-mail.

1.1.3 Authorization to Retrieve Proprietary Data

To be authorized to retrieve proprietary data, you must (a) send E-mail to the archive hotseat (`archive@stsci.edu`) if you are the Principal Investigator (PI) of the proposal, specifying the proposal ID number and your registered username, or (b) request the PI to do so.

Note that PIs are *not* automatically authorized to retrieve their own data.

1.1.4 Archive Documentation and Help

The Archive's web page provides a wealth of useful information, including an online version of the *HST Archive Manual*. Postscript versions of the *HST Archive Primer* and the *HST Archive Manual* are available via anonymous FTP from `archive.stsci.edu`. in the `pub/manuals` directory. If you have any questions, direct them via E-mail to `archive@stsci.edu`, or phone (410) 338-4547.

1.1.5 Getting Your Data Quickly

The fastest way to retrieve your proprietary data is to:

- Start StarView.
- Go to either the <General Search> or <Quick Search> screen.
- Enter your PI name or proposal ID in the appropriate field.
- Click on **[Begin Search]**.
- Click on the **[Scan Forward]** button in the <Results> screen to complete your search.
- Select all of your observations for retrieval by clicking on **[Mark All]**, then click **[Retrieve Marked Data]**. This action will spawn another screen listing all of your marked datasets.
- Click on **[Submit Request]**, which spawns the <File Options> screen. To make sure you get all of your data, select both calibrated and uncalibrated data. (This step is particularly important for NICMOS observers who want to retrieve individual exposures as well as their pipeline data products.)
- Click on **[Submit Request]**, bringing up a screen in which you enter your archive username and password and specify the means and delivery location for your data. StarView users can receive data directly to their home computer (NET) or on tape. Users interested in non-proprietary data can transfer datasets to a public site on `archive.stsci.edu` for subsequent downloading to their own machines via FTP (HOST).
- Click on **[Submit Request]** once more, and your data are on their way.

Users will receive an E-mail message when the retrieval request has been queued, and another when it has been completed. If you need more guidance, see the StarView tutorial below or the Archive documentation (*HST Archive Primer* or *HST Archive Manual*.)

1.2 Getting Data with StarView

StarView is available in an X-Windows based version and a terminal version for basic terminals, such as a VT100. This section leads you through a simple example of an X-Windows StarView session, providing additional information about how to run the terminal version where the two versions differ. The terminal version is only available remotely and is mainly useful if your electronic connection to STScI is slow. If you need to access StarView remotely, consult “Running StarView via Remote Access” on page 1-18 before proceeding.

Start the X-Windows version of StarView by typing:

```
% xstarview
```

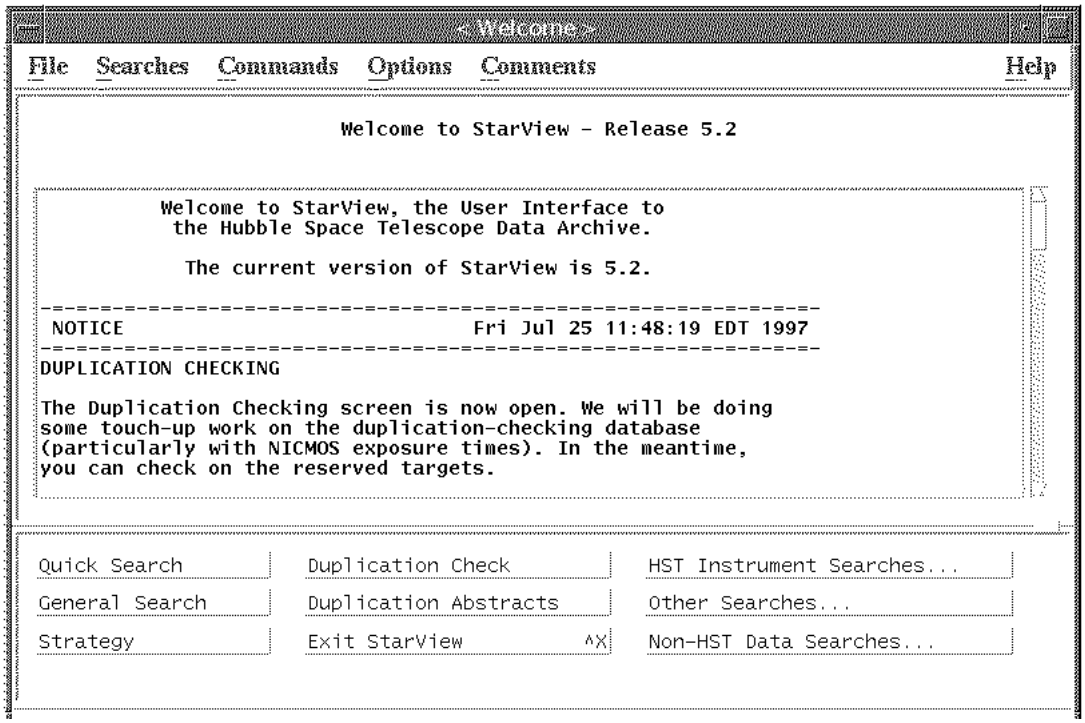
Some introductory messages will then appear on your screen. To see more of the text, press Space. To quit, press Q.

The StarView session then begins, displaying messages telling you what is happening (e.g., data dictionaries being loaded). The startup process may take a minute or two to complete.

1.2.1 Welcome Screen

The StarView <Welcome> screen (Figure 1.1) appears first. If there is any urgent news (e.g., a message about possible system downtime), it will appear at the top of the welcome text.

Figure 1.1: Welcome Screen



You can scroll through the text and read any additional information below the display area using the scroll bar on the X-Windows version of StarView. On the terminal version (for VT100 or other basic terminals), use the arrow keys or page up by pressing Control-V and page down by pressing Control-P.

1.2.2 Command Usage and Screen Interaction

In the *X-windows version* of StarView:

- Use the mouse to select all functions.
- Choose options by positioning the mouse pointer over the command button or menu and pressing the left mouse button.

In the *terminal version*:

- Press Control + T to cycle through the three screen areas (*menu*, *work area*, and *command box*).
- Use the arrow keys to move around within any one portion of the screen.
- Whenever an option is highlighted, press Return to invoke the highlighted function.

You can also use the *command accelerators* to invoke functions (i.e., run commands). Some command buttons show accelerators such as "^N", which means you can invoke the function or command by pressing down on the Control key while simultaneously pressing the N key. Other commands show accelerators such as "E+n", which means that you would press the Esc (escape) key followed by N.

1.2.3 Searching the Catalog

To search the catalog:

1. Choose a search screen.
2. Specify your search criteria, such as a range of sky coordinates (and a release date before today's date if you want public data.).
3. Click on the **[Begin Search]** button to start the search.
4. Click on the **[Scan Forward]** button to complete the search.

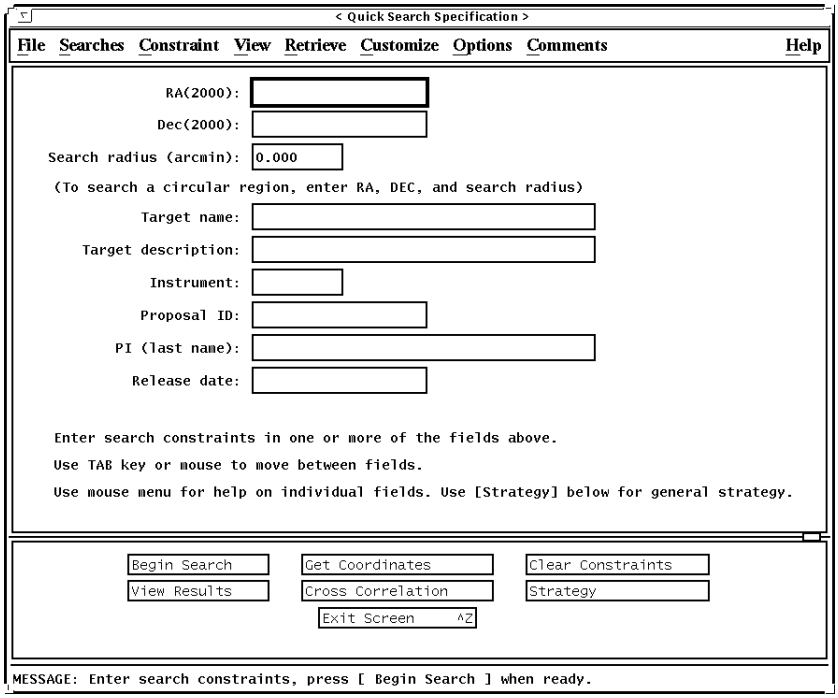
In this example we use the <Quick Search> screen to search the HST catalog.

The <Quick Search> screen is useful for most basic searches of the HST catalog. An extensive set of more detailed search screens is also available. To choose one of these, click on **[Other Searches]** or pull down the **| Searches |** menu. Consult the *HST Archive Manual* for more details.

The <Quick Search> Screen

Choose the <Quick Search> screen by clicking the **[Quick Search]** button. The <Quick Search> screen is shown in Figure 1.2. We will use this screen to request all of the public WFPC2 data for the galaxy M87.

Figure 1.2: Quick Search Screen



Specifying Search Criteria

There are various ways to search for observations of a particular target in the catalog. The easiest way is to enter the name (e.g., “M87”) in the target field. Because observers do not necessarily use the same convention to name sources, this strategy might not return all observations of a given source. The best way to ensure that you retrieve *all* observations of a given stationary target is to search for observations within a given (radial) distance of your source’s position by entering constraints in the “RA”, “Dec”, and “Search radius” fields on the <Quick Search> screen.

If you do not know the RA and Dec of your target, you can run either the SIMBAD or NED target name resolver from within StarView. Each resolver automatically determines the target’s position using a network connection to either the SIMBAD database in Europe or the NASA Extragalactic Database (NED) in California. It then populates the RA and Dec fields on the search screen with this information. Click on the **[Get Coordinates]** button to use the SIMBAD resolver; to use NED, pull down the **|Options|** menu, select User Defaults, and change the “Coordinates lookup server” field to NED.

Since we want all observations of M87 (even when the target name is something else), we will use “Get Coordinates” to fill in the RA and Dec. Click on the **[Get Coordinates]** button, enter “M87” as the name, and the RA and Dec fields will be automatically filled.

In this case, we want WFPC2 observations, so move to the "Instrument" field. The valid HST instruments are:

- Faint Object Camera (FOC).
- Faint Object Spectrograph (FOS).
- Fine Guidance Sensors (FGS).
- Goddard High Resolution Spectrograph (HRS).
- High Speed Photometer (HSP).
- Near Infrared Camera and Multiobject Spectrograph (NICMOS).
- Space Telescope Imaging Spectrograph (STIS).
- Wide Field Planetary Camera (WFPC).
- Wide Field Planetary Camera 2 (WFPC2).

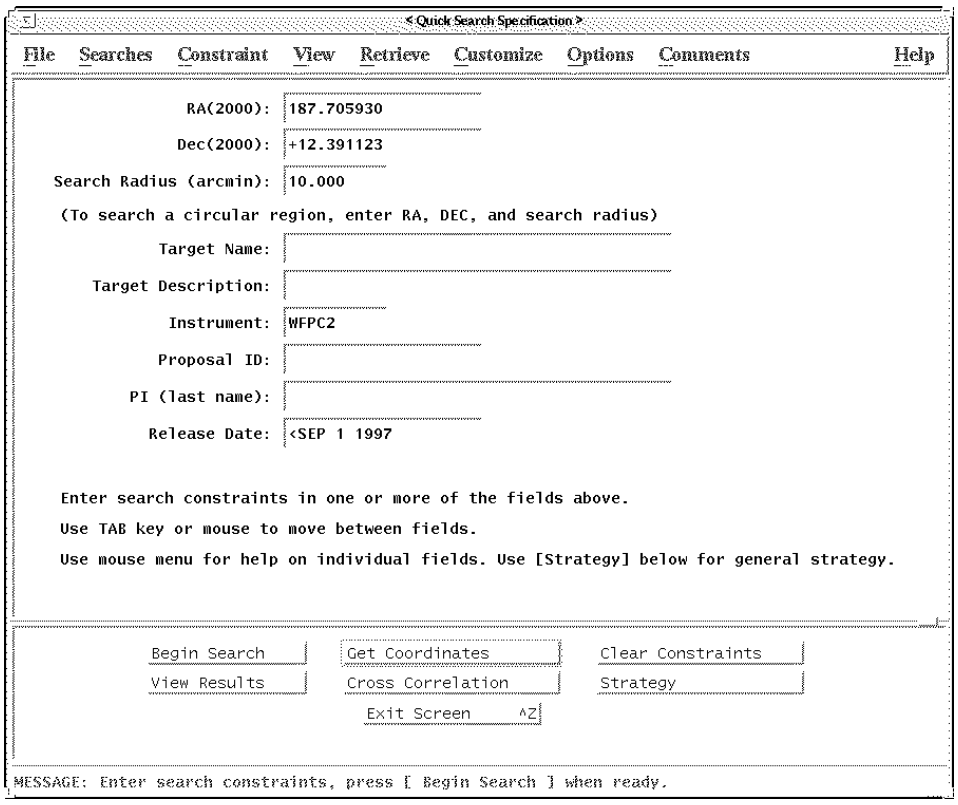
To get help on the valid ranges for any field, use the *field help*. In `xstarview`, move the cursor to the field and press the right mouse button (or press the Help button, often located in the bottom left corner of your keyboard).

Enter `WFPC2` in the instrument field. (To find observations from more than one instrument, use a comma-separated list; e.g., `WFPC2,WFPC,FOC`.)

We want public data, so now specify that we want data released prior to today's date. For example, move to the "Release date" field and enter `<Sep 1 1997` for any datasets that were released before September 1, 1997. Figure 1.3 shows how the <Quick Search> screen looks at this point.

Don't use commas in a date field. StarView will interpret the comma as a list operator.

Figure 1.3: Quick Search Screen With Constraints Entered



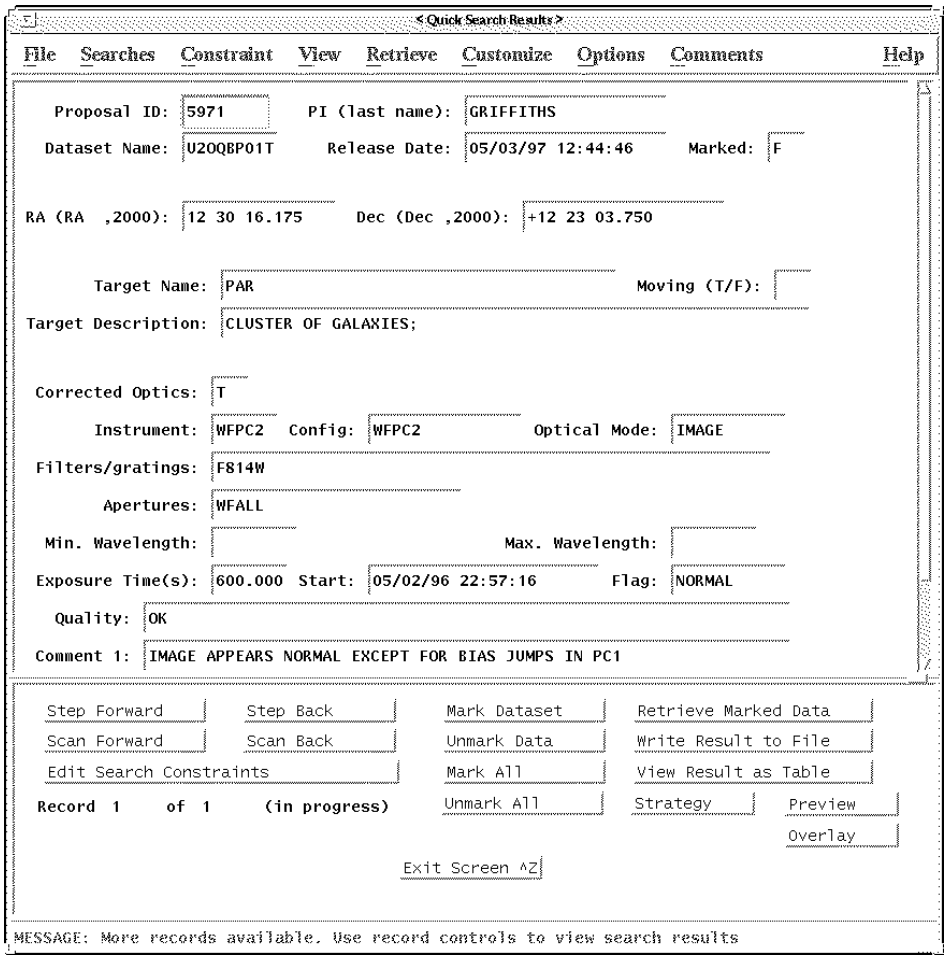
Use the **[Strategy]** button to get help using any StarView screen, or the pull down **|Help|** in the menu bar to see all the available StarView help.

Starting the Search

Click on the **[Begin Search]** button to search the catalog for the observations satisfying your search criteria. If none are found, a message will appear at the bottom of the screen, and you will need to enter different search constraints. If at least one observation is found, the screen will change to the <Quick Search Results> screen.

The <Quick Search Results> screen (Figure 1.4) shows the results of your catalog search. The first record that matches your search criteria will be displayed.

Figure 1.4: Quick Search Results Screen With Record Display



Viewing Subsequent Found Observations

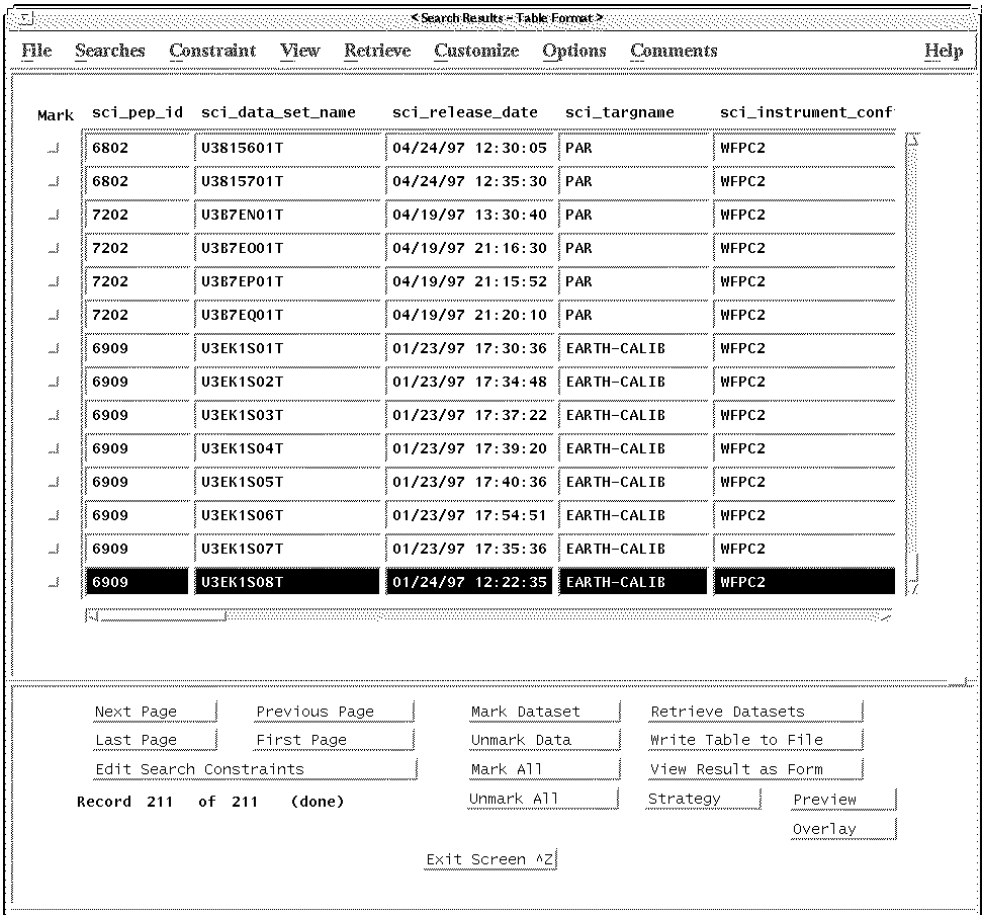
If you want to scan the full list of your search results:

- Click the **[Step Forward]** button to view one record at a time.
- Click **[Scan Forward]** to see all of the found records in rapid succession. Press any key to stop the scan. To complete the query, allow the Scan Forward process run to completion.
- To return to previous records, use **[Step Back]** or **[Scan Back]**.

Notice that many of the fields within 10’ of M87 do not have M87 as a target name. They are randomly-pointed parallel observations, high latitude fields, and so on. You should notice that some of the pointings are not astronomical exposures at all, but rather are random calibration fields such as biases and Earth flats. One way to exclude these datasets from your query is to bring up the General Search screen (which has more search parameters) and enter the constraint `!CAL` (meaning *not* CAL) in the Proposal Type field, as well as your coordinates, instrument, and release date constraints. This Proposal Type constraint excludes all standard (non-servicing mission) calibration proposals from your search. We will continue this example using the results of our <Quick Search> screen query.

Another way to view your search results is to use StarView's <TableFormat> screen. Click on **[View Result as Table]** to see several catalog records at the same time (see Figure 1.5). Click on **[View Result as Form]** to return to the single-record screen format.

Figure 1.5: Quick Search Results Displayed on the Table Format Screen



Use the **[Preview]** button to get a quick look at the data. Previewing can help you decide whether or not to retrieve a dataset. The Preview function displays compressed HST images (not suitable for science analysis), as well as FOS or GHRS spectra. Only public data are available for preview. Previews are not available in the terminal version.

1.2.4 Retrieving Datasets From the Archive

We now want to retrieve some of the data that we have identified in the catalog. The steps in this process are:

1. Mark the observations that you want to retrieve; you can mark them either individually or as a group.
2. Display and review the list of datasets to be retrieved.
3. Specify the file formats and media to be used in the retrieval process.
4. Submit the request.
5. Check the request status, if desired.

Marking Observations for Retrieval

To mark for retrieval the dataset displayed on the screen, click the **[Mark Dataset]** button. A message confirming this action will appear at the bottom of the screen. Also, the "Marked" field, in the upper right corner of the screen, will display "T" (true) indicating that the dataset has been marked for retrieval.

You can mark datasets for retrieval in either the table-row format display screen, in which case the highlighted record is marked, or on the <Quick Search Results> screen with the record displayed.

If you want to mark for retrieval *all* of the records matching your search criteria, click on the **[Mark All]** button. This volume of data could be very large, as it would be for the M87 search request described here. Alternatively, step through your search results records by clicking on the **[Step Forward]** button and click on the **[Mark Dataset]** button for the specific observations you desire. In this example, we mark only a few datasets for retrieval.

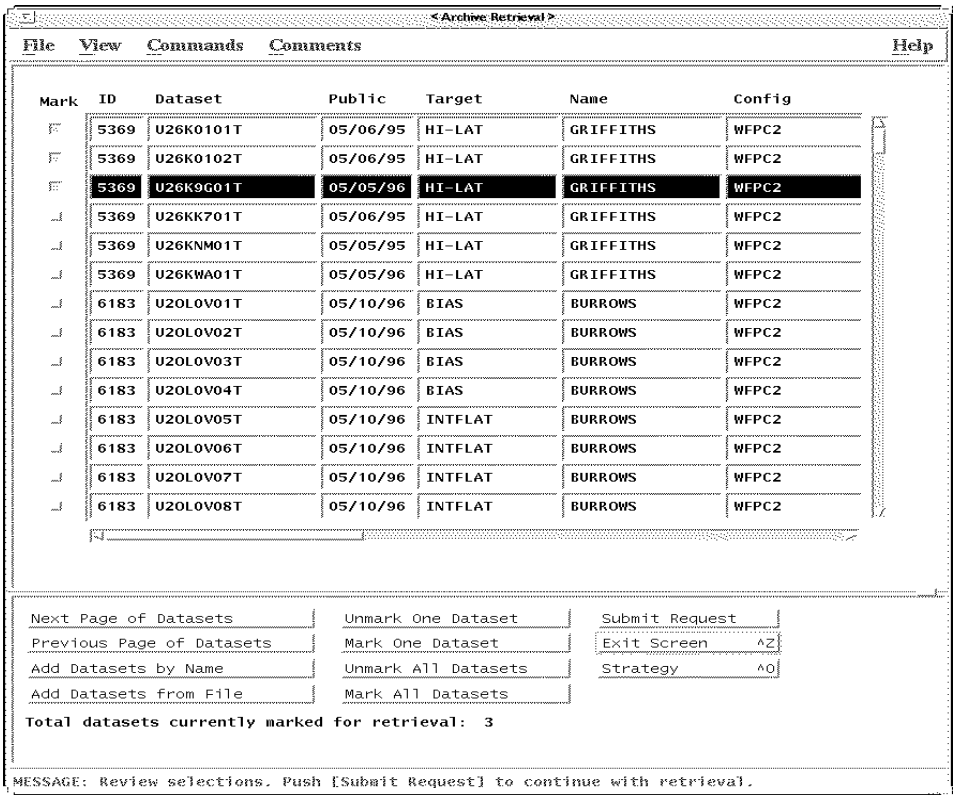
Reviewing the Retrieval Request

Once you have marked records for retrieval, you begin the retrieval process by displaying and reviewing the list of datasets to be retrieved:

1. Click on the **[Retrieve Marked Data]** button to exit the <Quick Search Results> screen and to begin the retrieval process by bringing up the <Archive Retrieval> screen.
2. Review the list of marked datasets.

The <Archive Retrieval> screen lists all of the datasets that you have marked for retrieval. In this case, you would see something like

Figure 1.6: The Archive Retrieval Screen



If you have marked numerous datasets, you may need to click on the **[Next Page of Datasets]** button to see additional screens of marked records. The total number of datasets that you have marked for retrieval is shown near the bottom of the screen.

If you wish to include additional datasets in your request, choose the **[Add Datasets by Name]** command from the <Archive Retrieval> screen (or use **[Add Datasets from File]** if you have a list of dataset names). Then enter the rootname (*no suffix*) of the calibration reference file(s) or science file(s) you wish to retrieve. (See Appendix B for more on rootnames and suffixes.)

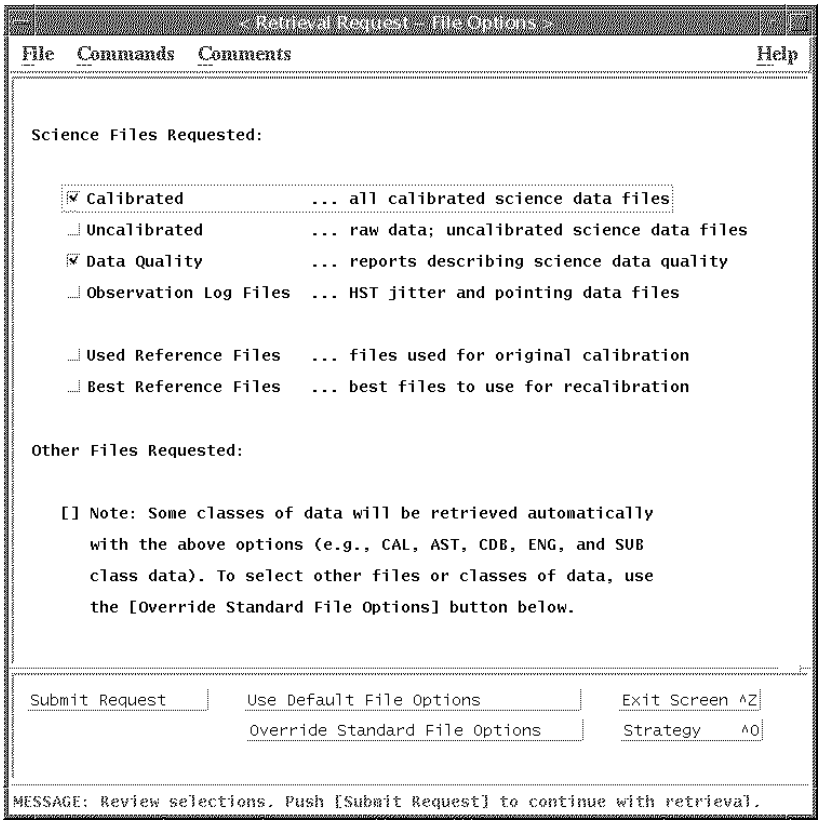
Specifying Formats and Media

To continue with the data retrieval process,:

1. Click the **[Submit Request]** button.
2. Specify the files that you want to retrieve.
3. Specify the type of medium (file transfer method) that you want.

When you click the **[Submit Request]** button, the <Retrieval Request - File Options> screen is displayed (Figure 1.7).

Figure 1.7: Retrieval Request - File Options Screen



The <Retrieval Request - File Options> screen indicates the kinds of files that will be retrieved. In this case the final calibrated science data files and the data quality report files will be retrieved. Click the **[Submit Request]** button to continue with the retrieval process.

If you plan to recalibrate your data, you can also request the uncalibrated data and the appropriate calibration reference files at this time. STIS users who need target acquisition data (ACQ or ACQ/PEAK images) should request uncalibrated data, because these images are not calibrated. NICMOS users who wish to obtain individual exposures that go into processed mosaics will also have to request the uncalibrated data. NICMOS users in particular may need to take care at this point not to exceed the ST-DADS dataset limit. This limit is currently set at 600 datasets, but be aware that multiple datasets can correspond to a single observation.

The <Retrieval Request - Media Options> screen is then displayed (Figure 1.8). You will need to enter your Archive user name and password, pressing Return after each entry. You can then choose how you would like to receive you data:

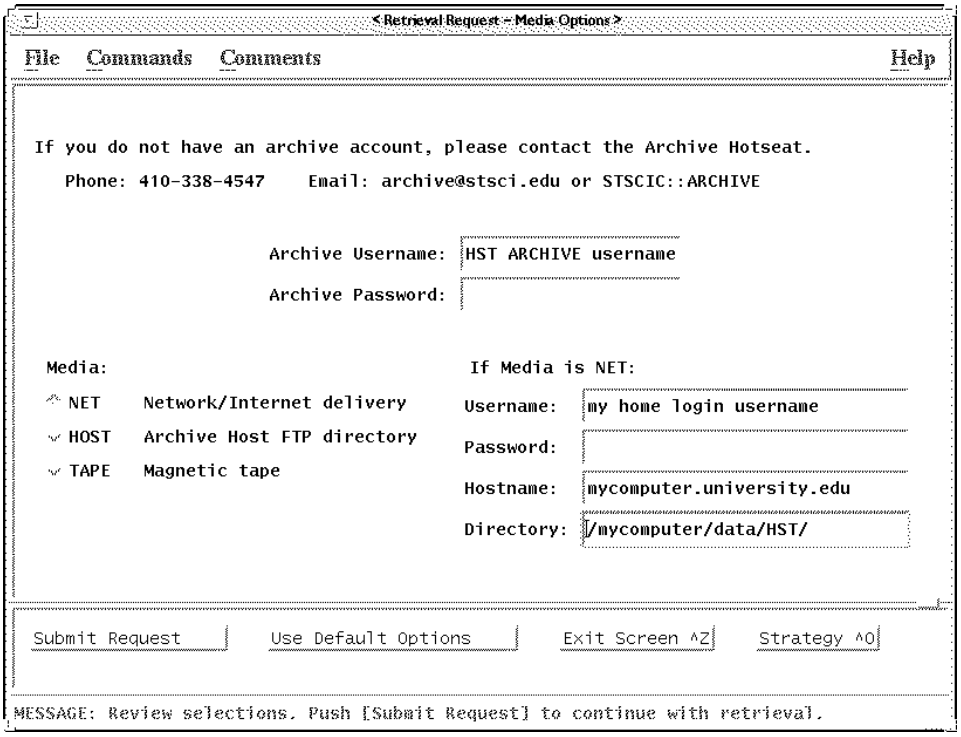
- Directly to your home computer via the Internet (NET).

- Via FTP from the Archive host computer (HOST).
- On magnetic tape shipped through the mail (TAPE).

To use the NET option, you must specify the Username and Password *of your own* computer account (or set Username to "anonymous" and Password to "archive@stsci.edu" if you opt to receive your data in an anonymous ftp site at your home institution), as well as the complete hostname of your workstation (e.g., `mycomputer.university.edu`) and full directory name (e.g., `/mycomputer/data/HST/`). All passwords are encrypted for secure transmission to STScI.

If you choose HOST delivery, your data will be sent to a subdirectory on `archive.stsci.edu` which will be named based on your archive username and a random number (e.g., SMITH1234) . To retrieve the data from this subdirectory, use anonymous FTP as described in "Transferring Your Data with FTP" on page 1-17.

Figure 1.8: Retrieval Request- Media Options Screen



The TAPE option is especially useful for large data requests.

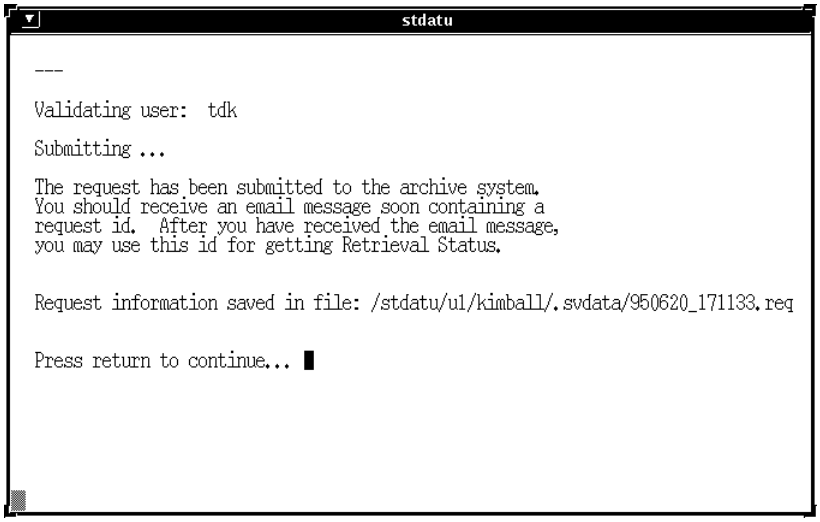
Submit the Request

Click on the **[Submit Request]** button to begin the submission process. StarView will validate your Archive account information and send your retrieval request to the Archive system.

At this point `xstarview` may want to interact with you using a special xterm window that it will initiate. Respond to any `xstarview` requests appearing in that window.

The list of datasets you have requested will be saved in a file named after the date and time of the request, with an extension of `.req`. StarView will display the name of this file in the xterm window. Figure 1.9 shows how a StarView screen might look at this stage.

Figure 1.9: Retrieval System Messages.



Press Return to exit from the retrieval process and to return to the StarView screen from which you initiated the retrieval request.

Shortly after your request is submitted, you will receive an E-mail message telling you that your request was accepted and queued by the Archive system and giving you the request ID.

You can use the request ID later to check the status of your request and also to locate your data on the Archive host's staging disk after it has been retrieved.

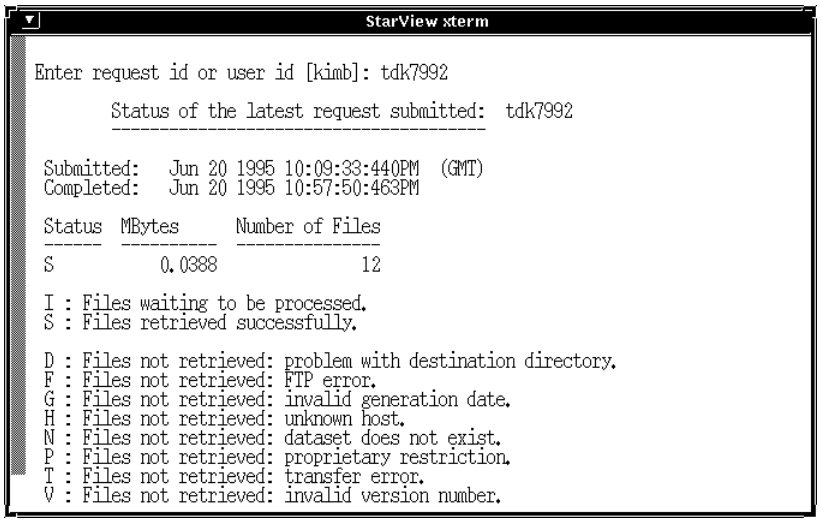
Checking Request Status

To check the status of your retrieval request:

1. Click on the **[Retrieval Status]** button from within the **| Retrieve |** menu on most StarView search screens, or click on the **| Commands |** menu from the <Welcome> screen.

2. You will be asked to enter your request ID, which will be E-mailed to you shortly after you submit your request. Type the request ID.
3. Press Return to continue with your StarView session. Figure 1.10 shows a sample retrieval status screen.

Figure 1.10: Sample Retrieval Status Screen



1.2.5 Exiting StarView

You can now either continue working in StarView, or you can exit. Press Control-X to exit StarView. A dialog box will appear asking you to confirm that you really want to exit. Click **[OK]** to exit.

1.2.6 Transferring Your Data with FTP

If you have chosen the HOST option, you will have to transfer your data yourself from `archive.stsci.edu` to your own disk via anonymous FTP. After your data have been retrieved from the Archive and placed in your `archive` subdirectory (e.g., SMITH1234 on `archive.stsci.edu`, see page 1-15), you will receive a second E-mail notification. You can then transfer the data to your home site as follows:

Figure 1.11: Retrieving Files Using FTP

Specify Binary

```
% ftp archive.stsci.edu
Connected to archive.stsci.edu.
220 archive.stsci.edu FTP server (Version 5.86) ready.
Name (archive.stsci.edu): anonymous
Password: Type your e-mail address
.
<message of the day displayed here>
.
ftp> cd tdk7992
250 CWD command successful.
ftp> binary
200 Type set to I.
ftp> prompt
Interactive mode off.
ftp> mget u*.fit
200 PORT command successful.
50 Opening BINARY mode data connection for u2900101t_c0f.fit
226 Transfer complete
.
.
ftp> bye
221 Goodbye.
```

Don't forget to set the FTP transfer type to "binary" before transferring the files.

1.2.7 Running StarView via Remote Access

If you must use telnet to access the Archive, here's how to do so.

```
% telnet archive.stsci.edu
Connected to archive.stsci.edu.
Escape character is '^]'.
Login: guest
Password: archive
```

The username and password for the guest account

The X-Windows version will ask for your X display host name. You should respond with the name of your home workstation. You will then be instructed to add `archive` to your computer's `xhost` file by typing the following line in another window and pressing Return to continue:

```
% xhost +archive.stsci.edu
```

You can now run the X-windows version of StarView remotely by typing

```
% xstarview
```

If you want to reduce networking overhead, you can use the terminal version instead by typing:

```
% starview
```

The terminal version of StarView will ask you to confirm your terminal setup. For example:

```
xterm 24 x 80 [Y]:
```

If this setup is correct, press Return to continue. If not, then answer "no" by pressing N followed by Return. If you answer "no", StarView will then ask some questions about your terminal type, number of lines, and number of columns. Type a question mark (?) to get help about your options.

Remember, if you have any problems or questions, contact the Archive hotseat at archive@stsci.edu.

1.3 Identifying Calibration Reference Files

StarView provides calibration reference file screens for each instrument. These screens let you see which calibration files and tables were used by pipeline processing to calibrate a given dataset at the time the data were taken, which calibration files and tables are currently recommended, and the degree to which the files used differ from the files recommended. You can mark either the used or the recommended reference files and tables for retrieval and retrieve them through StarView.

Note that you can retrieve the calibration reference files for your data very simply, either when you retrieve your data intially or later on, by searching and marking your data for retrieval and clicking on the "Best Reference Files" in the <Retrieval Request - File Options Screen>. (See Figure 1.7.)

In this example, we use a StarView reference file screen to display both the "used" and "recommended" calibration files for an M87 dataset.

From the <Welcome> Screen (Figure 1.3).

1. Click the **[HST Instrument Searches]** button.[2] (We want to find the search screen for calibration reference files.) The **[HST Instrument Searches]** screen will be displayed (Figure 1.12).
2. Click the **[Reference]** button on the **WFPC2** line to display <WFPC2 Reference Files - Search Specification> screen.
3. Specify search criteria. We want to specify a particular dataset, so move the cursor to the "Dataset Name" field, then type the dataset name for the observation whose calibration files you desire. For example, enter U2900101T (see Figure 1.13).

2. Users of the terminal version can use the accelerator E+2.

4. Click on the **[Begin Search]** button to submit your catalog search request. The <WFPC2 Reference Files - Search Results> screen will be displayed for the observation that you specified (Figure 1.14).
5. Click **[Mark USED Files for One Dataset]** to mark for retrieval those calibration files actually used to calibrate the dataset. If the files listed in the RECOMMENDED column differ from those in the USED column, then you can click on **[Mark RECOMMENDED for One Dataset]** to retrieve the calibration files that are now recommended for calibrating the data.
6. Click the **[Retrieve Marked Data]** button to begin the retrieval process for the marked reference files. Continue with the data retrieval procedures as outlined in “Retrieving Datasets From the Archive” on page 1-12.

The defaults on the <Retrieval Request-File Option> screen, i.e., “Calibrated”, will return the correct files for the specified calibration reference files and tables.

Figure 1.12: HST Instrument Searches Screen

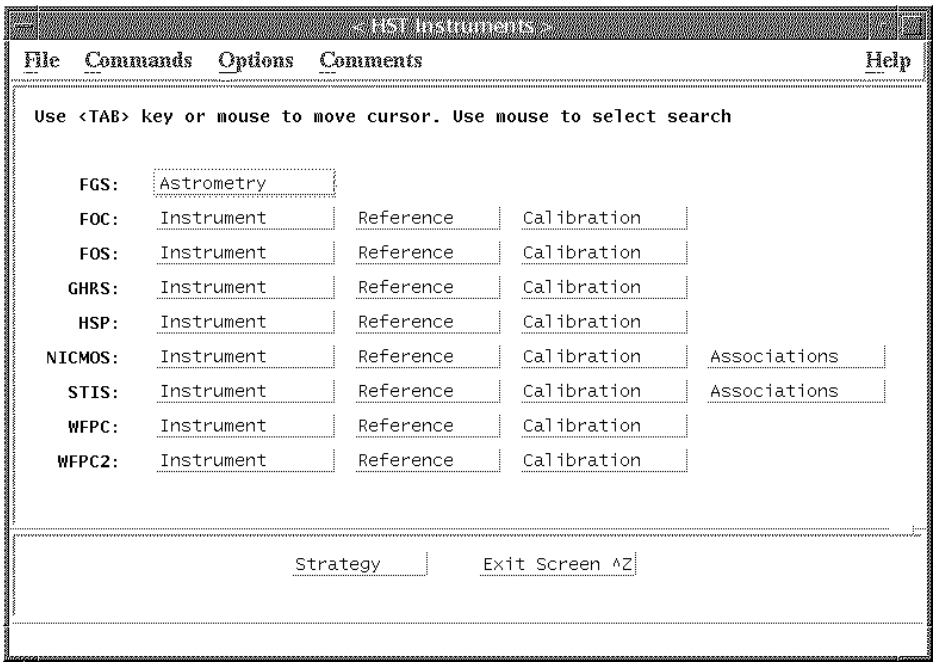


Figure 1.13: WFPC Reference Files - Search Specification Screen (Constrained)

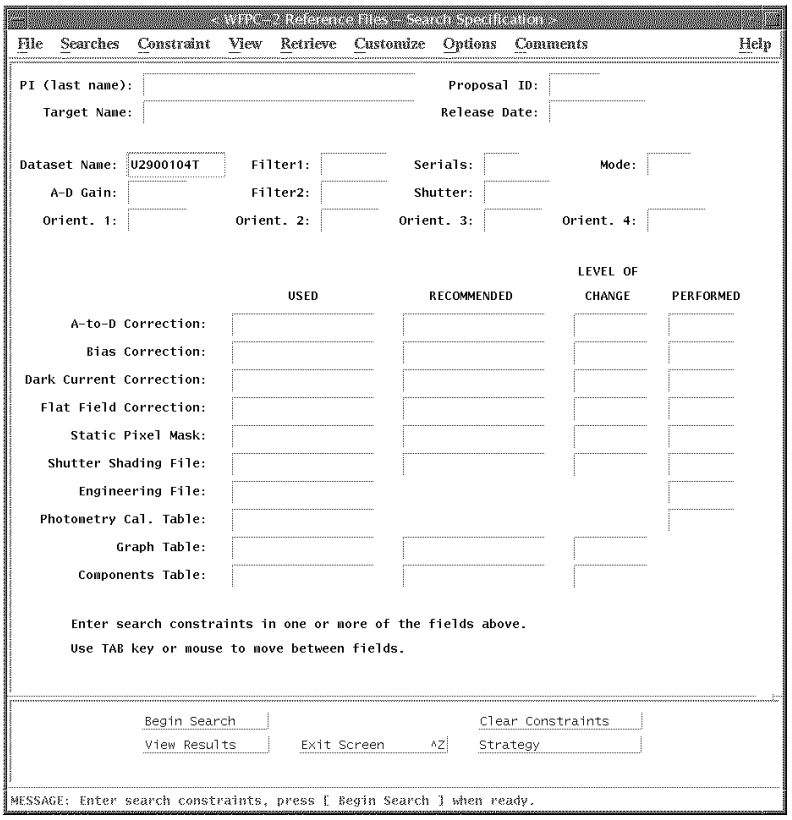
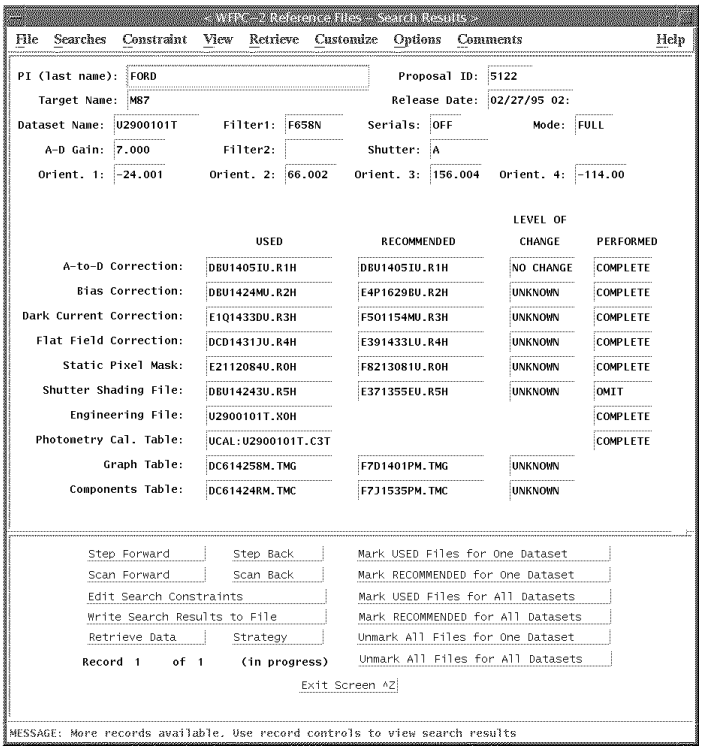


Figure 1.14: WFPC2 Reference Files - Search Results Screen



1.4 Reading HST Data Tapes

If you requested Exabyte or DAT tapes in your Phase II proposal, STScI will mail you one or more tapes containing your data within a few weeks of your HST observations. Your shipment of HST data ought to include:

- At least one data tape, together with a tape log listing its contents.
- Several printouts, called paper products, which are instrument-specific and are described in each instrument's Data Structures chapter.

Each of the files on your tape is in FITS (Flexible Image Transport System) format.[3] Any FITS reader should be able to read these files; however, GEIS (Generic Edited Information Set) is the standard format for analyzing all HST data other than STIS and NICMOS data. Chapter 2 discusses the HST implementations of the FITS and GEIS formats in more detail, explains the various types of HST data files, and interprets their names.

To read FITS files from your tape and write them to your disk, you can use the **strfits** task in IRAF/STSDAS. If you are not familiar with IRAF, consult the IRAF Primer in Appendix A before proceeding further. Chapter 3 contains additional information on STSDAS, the STScI data analysis software package.

The STSDAS **strfits** FITS reader preserves the multigroup format of an HST image. This format *must* be retained if you plan to recalibrate your data in STSDAS.

To read an HST data tape using **strfits**, you need to:

1. Start IRAF and load the **stsdas** and **fitsio** packages.
2. Mount the tape.
3. Set global parameters (e.g., `imtype`).
4. Set the **strfits** parameters and read the tape.

1.4.1 Loading Packages

Go to your IRAF home directory and start IRAF by typing:

```
cl
```

3. A description of the FITS format and the various options and parameters that can be used in the FITS standard can be found in the document "Implementation of the Flexible Image Transport System (FITS)," by the NASA/OSSA Office of Standards and Technology. The document is available via FTP to nssdca.gsfc.nasa.gov in the directory FITS. A listing of FITS standards and documentation is available from NRAO via the World Wide Web.

This action will start an IRAF session. Software in IRAF is organized into *packages*. To load a package, type its name. Once you are in IRAF, load the **stsdas** and **fitsio** packages by typing the following commands:

```
cl> stsdas
st> fitsio
```

The prompt (such as `fi>`) shows the first two letters of the most recently loaded package. The **fitsio** package contains tasks for handling the FITS format files used for HST images. You can use **catfits** (described on page 2-4) to produce a listing of the contents of your tape, and **strfits** to read the data onto disk. When you are done working with your data, you may choose to write it back to tape using **stwfits**.

1.4.2 Mounting the Tape

Mount the tape on your tape drive. Allocate the device within IRAF by typing:

`fi> allocate` *device*

where *device* is the IRAF name of the tape drive. If you are not sure how to mount tapes or don't know the IRAF names that match your tape drives, see your local system administrator for help.

1.4.3 Setting File Format

If you are reading FOC, FOS, FGS, GHRS, HSP, WF/PC-1, or WFPC2 data files, set the IRAF environment variable `imtype` to specify that your data files are to be written to your disk in GEIS format, for example:

```
fi> set imtype="hhh"
```

Currently **strfits** will also copy STIS and NICMOS files properly with `imtype` set to this value.

Then go to the directory in which you want your files to be stored. For example:

`fi> cd` */nem/data1/hstdata* ← **Replace this string with your directory path**

1.4.4 Using strfits

Like most IRAF and STSDAS tasks, **strfits** has several parameters that control the task's behavior. You can either specify the appropriate parameters on the command line or edit the parameter set using the **epar** task described on page A-8. In STSDAS version 2.0, most of the default parameters are set correctly for reading files from HST data tapes, so you need to specify only the name of the tape drive, the numbers of the data files to be read, and the oldirafname parameter.

For example, to read files 1 through 10 from the data tape on drive `mta`, type:

```
fi> strfits mta 1-10 oldirafname=yes
```

The tape log you received with your tape will tell you the number of each data file. You can also use the **catfits** task as follows to print a list of all the files on the tape to the screen:

```
fi> catfits mta 1-999 | page
```

Contact the Archive help desk at `archive@stsci.edu` if you have trouble reading your tape.

Be sure to set `oldirafname` and to "yes" or else the tape will not be read correctly and you will not be able to manipulate the data. This step is *vital* if you plan to recalibrate your data. (The parameter `xdimtogf` should be "yes" by default.)

Once your data are written on your disk, you can deallocate your tape drive by typing:

```
fi> deallocate device
```

You are now ready to work with your HST data files.

Deallocating the drive is important if you want to remain friends with your coworkers. No one else can use the drive until you deallocate it.

Chapter 2
HST File Formats

In This Chapter...

Historical Perspective / 2-2
FITS File Format / 2-3
GEIS File Format / 2-10

STScI automatically processes and calibrates all the data received from HST. The suite of software programs that performs this processing—part of a system known as OPUS—is frequently called the *pipeline*, and its purpose is to provide data to observers and to the HST Data Archive in a form suitable for most scientific analyses. Pipeline processing assembles data received from HST into *datasets*, calibrates the data according to standard procedures described in the instrument sections of this handbook, and stores both calibrated and uncalibrated datasets in the Archive.

Before the 1997 servicing mission, HST data files passed through pipeline processing in Generic Edited Information Set (GEIS) format and then were stored in the Archive in Flexible Image Transport System (FITS) format, a machine-independent format that simplifies the distribution of data to many different platforms. Users of FOC, FOS, FGS, GHRS, HSP, WF/PC-1, and WFPC2 data currently need to convert FITS data files received from the Archive back into GEIS format before they can reduce them further. Recent developments have made the FITS format much more useful for data processing, and it is now the standard format for pipeline processing and subsequent reduction of STIS and NICMOS data. Observers using these new instruments *should not* convert their files to GEIS format.

This chapter describes these two HST file formats, first giving some historical perspective on the reasons why they were selected, then explaining the FITS and GEIS formats in more detail. STIS and NICMOS observers should pay particular attention to the section on FITS files, which shows how to identify and access the contents of these files and covers some important conventions regarding header keywords. Veteran observers with the other instruments will find little new in the section on GEIS files, but newcomers to the older HST instruments should consult the material on data groups and conversion from FITS to GEIS before proceeding to Chapter 3.

2.1 Historical Perspective

In the early 1980's, when GEIS was selected as the standard format for HST data files, it held several advantages over both FITS and the original IRAF format (OIF):

- GEIS allows floating-point data. The early incarnations of FITS accommodated only integer data, and this restriction to integers would have made data reduction and storage of calibrated data rather cumbersome.
- GEIS files can hold multiple images, each with associated parameters. This feature allowed the packaging of images from the four WF/PC-1 chips into a single unit, as well as the packaging of multiple FOS or GHRS readouts into single files. OIF files and early FITS files could contain only single images.
- GEIS data are stored in two parts, an ASCII header and a binary data file. The separation of these two pieces and the restriction of the header to ASCII made these headers easier to read and print in the days when computers were less powerful and tasks for reading header information were less numerous. OIF headers combine ASCII and binary information, and FITS headers come packaged with the data in a single file.

GEIS was also the standard format for archiving and distribution of HST data until September 1994, when the Space Telescope Data Archive and Distribution Service (ST-DADS) came online. This new system stores and distributes HST data files in machine-independent FITS format, but observers with FOC, FOS, FGS, GHRS, HSP, WF/PC-1, and WFPC2 still must convert their files to machine-dependent GEIS format as described on page 2-11 before using IRAF/STSDAS software (see Chapter 3) to reduce their data.

Since the selection of GEIS as HST's standard data format, FITS has added features that have dramatically increased its flexibility. In particular, FITS files can now contain multiple *image extensions*, each with its own header, size, and datatype, that allow multiple exposures to be packaged into the same file, along with associated error and data quality information. The FITS image kernel in IRAF version 2.11, released in August 1997, enables users to access FITS image extensions in ways similar to how they would access GEIS data groups.

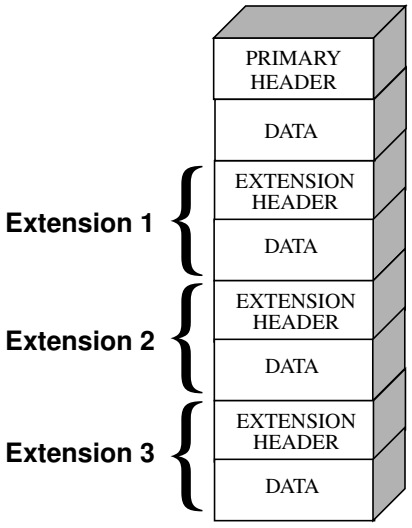
Because of these advantages, FITS was chosen as the standard reduction and analysis format for STIS and NICMOS. The STSDAS tasks written for these instruments expect FITS files as input and produce FITS files as output. *You cannot convert STIS and NICMOS files to GEIS format.* Observers using these instruments should therefore read the following section, which explains how to work with these new FITS files.

2.2 FITS File Format

Flexible Image Transport System (FITS) is a standard format for exchanging astronomical data between institutions, independent of the hardware platform and software environment. A data file in FITS format consists of a series of Header Data Units (HDUs), each containing two components: an ASCII text header and the binary data. The header contains a series of *header keywords* that describe the data in a particular HDU and the data component immediately follows the header.

The first header in a FITS file is known as the *primary header*, and any number of *extensions* can follow the primary HDU. The data unit following the primary header must contain either an image or no data at all, but each extension can contain one of several different data types, including images, binary tables, and ASCII text tables. The value of the XTENSION keyword in the extension's header identifies the type of data the extension contains. Figure 2.1 schematically illustrates the structure of a FITS file and its extensions.

Figure 2.1: FITS File Structure



The three-letter identifier (e.g., `d0h`) that follows the rootname of an HST data file (see Appendix B for more on HST file names) has often been called an "extension" in the past. However, because of the potential for confusion with FITS extensions, this handbook will refer to these three-letter identifiers as "suffixes."

2.2.1 Working with FITS Image Extensions

The FITS image kernel included in IRAF version 2.11 is designed to read and write the images in FITS extensions and their associated headers. Once IRAF has ingested a FITS image and its header, it treats the header-data pair like any other IRAF image. The following discussion describes how to specify the image extensions in FITS files that you would like to process with IRAF/STSDAS tasks and presumes that you are using IRAF 2.11 or higher. It covers how to:

- List a FITS file's extensions.
- Access data in particular FITS extensions.
- Inherit keywords from the primary header.
- Append new extensions to existing FITS files.

Retaining the `.fits` at the end of every FITS file name in your file specifications will ensure that IRAF both reads and writes these images in FITS format.

If you want to work with STIS and NICMOS data, you will need to upgrade to IRAF 2.11 or higher and STSDAS 2.0.

Generating a FITS File Listing

Once you have downloaded STIS or NICMOS FITS files from the Archive, you may want an inventory of their contents. To generate a listing of a FITS file's extensions, you can use the **catfits** task in the **tables** package. The following example illustrates the first thirteen lines generated by **catfits** from a NICMOS MULTIACCUM FITS file containing images only.

Figure 2.2: NICMOS MULTIACCUM Listing from catfits

```
tt> catfits n3t501c2r_raw.fits

EXT#  FITSNAME      FILENAME              EXTVE  DIMENS     BITPI OBJECT

0     n3t501c2r_raw n3t501c2r_raw.fits                      16    n3t501c2r_raw.f
1       IMAGE       SCI                   1      256x256    16    n3t501c2r_raw.f
2       IMAGE       ERR                   1                 -32
3       IMAGE       DQ                    1                 16
4       IMAGE       SAMP                  1                 16
5       IMAGE       TIME                  1                 -32
6       IMAGE       SCI                   2      256x256    16
7       IMAGE       ERR                   2                 -32
8       IMAGE       DQ                    2                 16
9       IMAGE       SAMP                  2                 16
10      IMAGE       TIME                  2                 -32
11      IMAGE       SCI                   3      256x256    16
12      IMAGE       ERR                   3                 -32
```

The first column of a **catfits** listing gives the extension numbers. Note that the primary HDU is labeled extension number zero. The second column lists the extension type, given by the keyword XTENSION (IMAGE = image, BINTABLE = binary table, TABLE = ASCII table). The third column lists the extension name, given by the keyword EXTNAME. In STIS and NICMOS image files, the EXTNAME values SCI, ERR, and DQ indicate science, error, and data quality images, respectively. NICMOS image files contain samples and exposure time images as well, with EXTNAME values SAMP and TIME.

Each STIS or NICMOS readout generates an image set or *imset*. STIS imsets comprise three images (SCI, ERR, DQ), while NICMOS imsets comprise five (SCI, ERR, DQ, SAMP, TIME). All images belonging to the same imset share the same integer value of the EXTVER keyword, given in the fourth column of a **catfits** listing. Several STSDAS tasks can work with entire imsets (see "Working with STIS and NICMOS Imsets" on page 3-12), but most operate on individual images. See the STIS and NICMOS Data Structures chapters for more information on the contents of imsets.

Accessing FITS Images

After you have identified which FITS image extension you wish to process, you can direct an IRAF/STSDAS task to access that extension using the following syntax:

`fitsfile.fits[`*extension number*`][`*keyword options*`][`*image section*`]`

Note that all the bracketed information is optional. However, the only time it is *valid* to provide only a file name without further specification is when the file is a simple FITS file that contains a single image in the primary HDU.

Designation of the extension number is the most basic method of access, but it is not necessary the most helpful. Referring to an extension's EXTNAME and EXTVER in the `[keyword options]` is often more convenient. If a number follows an EXTNAME, IRAF interprets the number as an EXTVER. For example, if extension number 6 holds the the science image belonging to the imset with EXTVER = 2, as in the **catfits** listing on the previous page, you can specify it in two equivalent ways:

```
fitsfile.fits[6]
fitsfile.fits[sci,2]
```

Designations giving an EXTNAME without an EXTVER refer to the first extension in the file with the specifed value of EXTNAME. For example, `fitsfile.fits[sci]` refers to the first science image extension in `fitsfile.fits`.

The syntax for designating image sections adheres to the IRAF standard, so in the current example the specifications

```
fitsfile.fits[6][100:199,100:299]
fitsfile.fits[sci,2][100:199,100:299]
```

both extract a 100 by 200 pixel subsection of the same science image in `fitsfile.fits`.

Header Keywords and Inheritance

STIS and NICMOS data files take advantage of an IRAF image kernel convention regarding the relationship of the primary header keywords to image extensions in the same file. In particular, IRAF allows image extensions to *inherit* keywords from the primary header under certain circumstances. When this inheritance takes place, the primary header keywords are practically indistinguishable from the extension header keywords. This feature circumvents the large scale duplication of keywords that share the same value for all extensions. The primary header keywords effectively become global keywords for all image extensions. The FITS standard does not cover or imply keyword inheritance, and while the idea itself is simple, its consequences are often complex and sometimes surprising to users.

Generally keyword inheritance is the default, and IRAF/STSDAS applications will join the primary and extension headers and treat them as one. For example, using **imheader** as follows on a FITS file will print the both the primary and extension header keywords to the screen:

```
cl> imheader fitsfile.fits[sci,2] long+ | page
```

Using **imcopy** on such an extension will combine the primary and extension headers in the output HDU, even if the output is going to an extension of another FITS file. Once IRAF has performed the act of inheriting the primary header keywords, it will normally turn the inheritance feature off in any output file it creates unless specifically told to do otherwise.

If you need to change the value of one of the global keywords inherited from the primary header, you must edit the primary header itself (i.e., “extension” [0]).

Keyword inheritance is not always desirable. For example, if you use **imcopy** to copy all the extensions of a FITS file to a separate output file, IRAF will write primary header keywords redundantly into each extension header. You can suppress keyword inheritance by using the NOINHERIT keyword in the file specification. For example:

```
im> imcopy fitsfile.fits[6][noinherit] outfile.fits
im> imcopy fitsfile.fits[sci,2,noinherit] outfile.fits
```

Both of the preceding commands will create an output file whose header contains only those keywords that were present in the original extension header. Note that the `noinherit` specification belongs bracketed with the EXTNAME and EXTVER keywords and not in a separate bracketed unit of its own.

Appending Image Extensions to FITS Files

IRAF/STSDAS tasks that produce FITS images as output can either create new FITS files or append new image extensions to existing FITS files. You may find the following examples useful if you plan to write scripts to reduce STIS or NICMOS data:

- If the specifed output file does not yet exist, a new output file is created containing only a primary HDU if no specification is appended to the output file name. For example, to copy the contents of the primary header of `fitsfile.fits` into the primary HDU of the FITS file `outfile.fits`, type the command:

```
cl> imcopy fitsfile.fits[0] outfile.fits
```

- If the specified output file already exists and you want to append a new extension to it, you need to include the APPEND option in the output file specification. The following command appends extension `[sci,2]` of `fitsfile.fits` onto the existing file `outfile.fits`, while retaining the original EXTNAME and EXTVER of the extension—the `noinherit` specification inhibits the copying of the primary header keywords from the input file into the output extension header:

```
cl> imcopy fitsfile.fits[sci,2,noinherit] \
>>> outfile.fits[append]
```

- If you want to change the EXTNAME or EXTVER of the appended extension, you can specify the new values of these keywords in the output extension, like this:

```
cl> imcopy fitsfile.fits[sci,2,noinherit] \
>>> outfile.fits[sci,3,append]
```

- For obvious reasons, it is not generally advisable for two file extensions in the same FITS file to share the same EXTNAME and EXTVER values. However, if you must append an extension to an output file already containing an extension with the same EXTNAME/EXTVER pair you can do so with the DUPNAME option:

```
cl> imcopy fitsfile.fits[7] \
>>> outfile.fits[append,dupname]
```

- If you need to replace an existing extension with a new output extension, you can use the OVERWRITE option as follows. Overwriting can cause a lengthy rewrite of the whole file to insert the new extension, if its size is not the same as the extension it replaces.

```
cl> imcopy fitsfile.fits[sci,2,noinherit] \
>>> outfile.fits[sci,2,overwrite]
```

2.2.2 Working with FITS Table Extensions

STIS and NICMOS use FITS tables in two basic ways. Both instruments produce association tables (see “Associations” on page B-4) listing the exposures that go into constructing a given association product. In addition, STIS provides certain spectra, calibration reference files, and time-tagged data in tabular form (see Chapter 20). Here we describe:

- How to access and read FITS table extensions.
- How to specify data arrays in FITS table cells.

This discussion assumes you are using STSDAS 2.0 or later. (The IRAF FITS kernel deals only with FITS images. The **tables** package installed with STSDAS handles FITS table extensions.)

Accessing FITS Tables

You can access data in FITS table extensions using the same tasks appropriate for any other STSDAS table, and the syntax for accessing a specific FITS table is similar to the syntax for accessing FITS images (see "Working with FITS Image Extensions" on page 2-4), with the following exceptions:

- The FITS table interface does not support header keyword inheritance.
- FITS tables cannot reside in the primary HDU of a FITS file. They must reside instead in a FITS table extension, in either ASCII form (XTENSION=TABLE) or binary form (XTENSION=BINTABLE).
- If the first extension in a FITS file is a TABLE or a BINTABLE, you can access it by typing the file name with no extension specifed. It is not sufficient for the table to be just the first BINTABLE or TABLE; *it must actually be the first extension*.

For example, running **catfits** on the NICMOS association table `n3tc01010_asn.fits` provides the following output:

```
fi> catfits n3tc01010_asn.fits

EXT#  FITSNAME      FILENAME              EXTVE ...
0     n3tc01010_asn N3TC01010_ASN.FITS ...
1       BINTABLE    ASN                   1 ...
```

Extension number 1 holds the association table, which has EXTNAME=ASN and EXTVER=1. You can use the **tprint** task in the STSDAS **tables** package to print the contents of this table, and the following commands are all equivalent:

```
tt> tprint n3tc01010_asn.fits
tt> tprint n3tc01010_asn.fits[1]
tt> tprint n3tc01010_asn.fits[asn,1]
```

STSDAS **tables** tasks can read both FITS TABLE and BINTABLE extensions, but they can write tabular results only as BINTABLE extensions. Tasks that write to a table in-place (i.e., **tedit**) can modify an existing FITS extension, and tasks that create a new table (i.e., **tcopy**) will create a new extension when writing to an existing FITS file. If the designated output file does not already exist, the task will create a new FITS file with the output table in the first extension. If the output file already exists, your task will append the new table to the end of the existing file; the APPEND option necessary for appending FITS image extensions is not required. As with FITS images, you can specify the EXTNAME and EXTVER of the ouput extension explicitly, if you want to assign them values different from those in the input HDU. You can also specify the OVERWRITE option if you want the output table to supplant an existing FITS extension. For example, you could type:

```
tt> tcopy n3tc01010_asn.fits outfile.fits[3][asn,2,overwrite]
```

This command would copy the table in the first extension of `n3tc01010_asn.fits` into the third extension of `outfile.fits`, while reassigning it the EXTNAME/EXTVER pair `[asn,2]` and overwriting the previous contents of the extension. Note that overwriting is the only time when it is valid to specify an extension, EXTNAME, and an EXTVER in the output specification.

Specifying Arrays in FITS Table Cells

A standard FITS table consists of columns and rows forming a two-dimensional grid of cells; however, each of these cells can contain a data array, effectively creating a table of higher dimensionality. Tables containing extracted STIS spectra (see "Tabular Storage of STIS Data" on page 20-5) take advantage of this feature. Each column of a STIS spectral table holds data values corresponding to a particular physical attribute, such as wavelength, net flux, or background flux. Each row contains data corresponding to one spectral order, and tables holding echelle spectra can contain many rows. Each cell of such a spectral table can contain a one-dimensional data array corresponding to the physical attribute and spectral order of the cell.

In order to analyze tabular spectral data with STSDAS tasks other than the **sgraph** task and the **igi** package, which have been appropriately modified, you will need to extract the desired arrays from the three-dimensional table. Two new IRAF tasks, named **tximage** and **txtable**, can be used to extract the table-cell arrays. Complementary tasks, named **tiimage** and **titable**, will insert arrays back into table cells. To specify the arrays which should be extracted from or inserted into the table cells, you will need to use the *selectors* syntax to specify the desired row and column. The general syntax for selecting a particular cell is:

`intable.fits[`*extension number*`][c:`*column_selector*`][r:`*row_selector*`]`

or

`intable.fits[`*keyword options*`][c:`*column_selector*`][r:`*row_selector*`]`

A *column selector* is a list of column patterns separated by commas. The column pattern is either a column name, a file name containing a list of column names, or a pattern using the IRAF pattern matching syntax. If you need a list of the column names, you can run the **tlcol** task (type `tlcol infile.fits`).

Rows are selected according to a *filter*. The filter is evaluated at each table row, and the row is selected if the filter is true. For example, if you specify:

```
infile.fits[3][c:WAVELENGTH,FLUX][r:SPORDER=(68:70)]
```

IRAF will extract data from the table stored in the third extension of the FITS file, `infile.fits`, specifically the data from the columns labelled WAVELENGTH and FLUX, and will restrict the extraction to the rows where the spectral order (SPORDER) is within the range 68–70, inclusive. Alternatively, if you specify:

```
infile.fits[sci,2][c:FLUX][r:row=(20:30)]
```

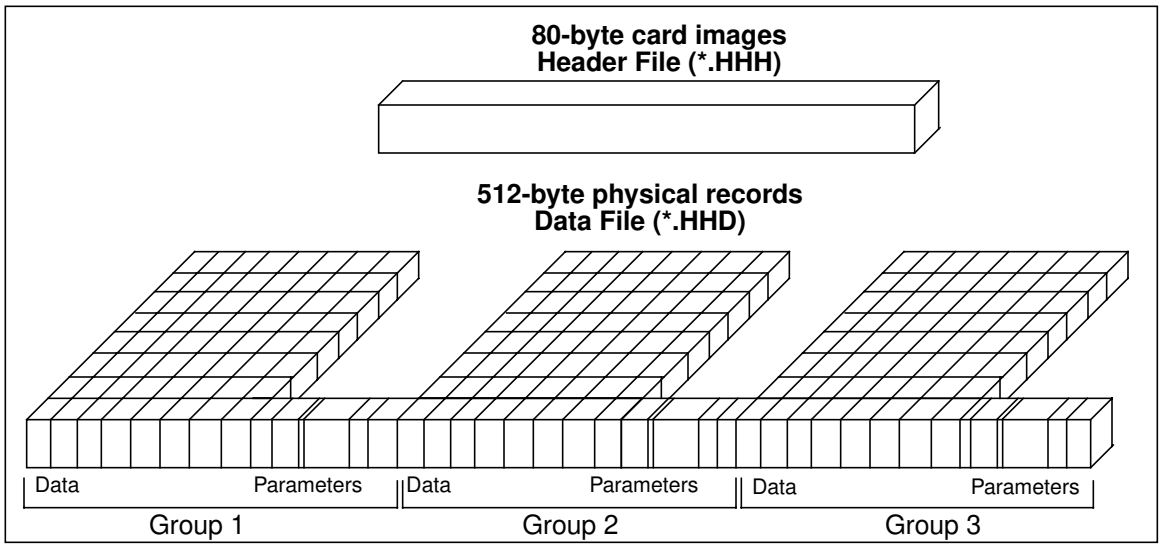
IRAF will obtain data from the table stored in the FITS file extension with an EXTNAME of SCI and EXTVER of 2. The data will come from the column FLUX and be restricted to the row numbers 20–30, inclusive. Eventually, all STSDAS and TABLES tasks will be able to use row and column selection.

2.3 GEIS File Format

The HST-specific Generic Edited Information Set (GEIS) format[1] is the standard format for reducing data from FOC, FOS, FGS, GHRS, HSP, WF/PC-1, and WFPC2. All HST images in GEIS format consist of two components: a *header file* and a separate *binary data file*, both of which should reside in the same directory. GEIS header files, whose suffixes end in "h" (e.g., `w0lo0105t.c1h`), consist entirely of ASCII text in fixed-length records of 80 bytes. These records contain header keywords that specify the properties of the image itself and the parameters used in executing the observation and processing the data. GEIS binary data files, whose suffixes end in "d" (e.g., `w0lo0105t.c1d`), contain one or more *groups* of binary data. Each group comprises a data array followed by an associated block of binary parameters called the Group Parameter Block (GPB). The sizes and datatypes of the data arrays and group parameters in each group of a GEIS file are identical. Figure 2.3 depicts the structure of a GEIS data file graphically.

The binary content of GEIS files is machine dependent. Copying GEIS files directly from one platform to another (e.g., from a VAX to a Sun) may result in unreadable data.

Figure 2.3: GEIS File Structure



1. GEIS files are also commonly referred to as STSDAS images.

2.3.1 Converting FITS to GEIS

The STScI archive stores and distributes datasets from FOC, FOS, FGS, GHRS, HSP, WF/PC-1, and WFPC2 in a special archival FITS format. *We highly recommend that users convert these datasets back into their native GEIS format before working with them.* Your data must be in GEIS format for you to use many of the STSDAS software tools developed specifically for analysis of these data. It is important to use the **strfits** task found in **stsdas.fitsio** or in **tables.fitsio** to perform the conversion from archival FITS format to the GEIS format because the data-processing pipeline employs a special convention for mapping GEIS files to FITS format. While other FITS readers may be able to read portions of the data correctly, they are unlikely to reconstruct the entire data file properly.

To recreate the original multigroup GEIS file using **strfits**, you must first type:

```
cl> set imtype=hhh
```

This command tells IRAF to write output files in GEIS format. You then need to set the **strfits** parameters `xdimtogf` and `oldirafname` both to "yes". For example, after you have set `imtype = hhh`, you can convert the FITS file `*_hhf.fits` into the GEIS format files `*.hhh` and `*.hhd` by typing:

```
cl> strfits *_hhf.fits iraf_fil="" xdim=yes oldiraf=yes
```

2.3.2 GEIS Data Groups

One of the original advantages of GEIS format noted in Section 2.1 was that it could accommodate multiple images within a single file. This feature is useful because a single HST observation often produces multiple images or spectra. For example, a single WF/PC-1 or WFPC2 exposure generates four simultaneous images, one for each CCD chip. Likewise, the FOS and GHRS obtain data in a time-resolved fashion so that a single FOS or GHRS dataset comprises many spectra—one corresponding to each readout. The data corresponding to each sub-image (for the WF/PC-1 or WFPC2) or each sub-integration (for the FOS or GHRS) are stored sequentially in the groups of a single GEIS binary data file. The header file corresponding to this data file contains the information that applies to the observation as a whole (i.e., to all the groups in the image), and the group-specific keyword information is stored in the group parameter block of each data group in the binary data file.

The *number* of groups produced by a given observation depends upon the instrument configuration, the observing mode, and the observing parameters. Table 2.1 lists the *contents* and the number of groups in the final calibrated image for the most commonly-used modes of each instrument.

2.3.3 Working with GEIS Files

This section briefly explains how to work with information in GEIS header and data files.

Table 2.1: Groups in Calibrated Images, by Instrument and Mode

Instrument	Mode	Number of Groups	Description
FGS	All	17	FGS data are not reduced with IRAF and STSDAS. Therefore, FGS groups have different meaning than for the other instruments.
FOC	All	1	All FOC images have only a single group.
FOS	ACCUM	*n*	Group *n* contains accumulated counts from groups (subintegrations) 1, 2, ... *n*. The last group is the full exposure.
	RAPID	*n*	Each group is an independent subintegration with exposure time given by group parameter EXPOSURE.
HSP	All	1	HSP datasets always have only a single group that represents either digital star (`.d0h`, `.c0h`), digital sky (`.d1h`, `.c1h`), analog star (`.d2h`, `.c2h`), or analog sky (`.d3h`, `.c3h`).
GHRS	ACCUM	*n*	Each group is an independent subintegration with exposure time given by group parameter EXPOSURE. If FP-SPLIT mode was used, the groups will be shifted in wavelength space. The independent subintegrations should be coadded prior to analysis.
	RAPID	*n*	Each group is a separate subintegration with exposure time given by group parameter EXPOSURE.
WF/PC-1	WF	4	Group *n* represents CCD chip *n*, e.g., group 1 is chip 1 (unless not all chips were used). Group parameter DETECTOR always gives chip used.
	PC	4	Group *n* is chip *n* + 4, e.g., group 1 is chip 5. If not all chips were used, see the DETECTOR parameter which always gives the chip used.
WFPC2	All	4	Planetary chip is group 1, detector 1. Wide Field chips are groups 2–4 for detectors 2–4. If all chips were not used, see the DETECTOR keyword.

GEIS Headers

Header keyword information relevant to each group of a GEIS file resides in two places, the header file itself and the parameter block associated with the group. Because GEIS header files are composed solely of ASCII text, they are easy to print using standard Unix or VMS text-handling facilities. However, the group parameters are stored in the binary data file. To access them you need to use a task such as **imheader**, as shown on page 2-13.

You can use the IRAF **hedit** task to edit the keywords in GEIS headers. While it it possible to edit GEIS header files using standard Unix and VMS text editors, you must maintain their standard 80-character line length. The **hedit** task automatically preserves this line length. If you need to add or delete group parameters, you can use the STSDAS **groupmod** task in the

stsdas.hst_calib.ctools package. The STSDAS **chcalpar** task, described in more detail in the Calibration chapters for each instrument, is useful for updating header keywords containing calibration switches and calibration reference files.

Always edit headers using tasks like **hedit**, **eheader**, and **chcalpar**. Editing headers with a standard text editor may corrupt the files by creating incorrect line lengths.

GEIS Data Files

Numerous IRAF/STSDAS tasks exist for working with GEIS images (see Chapter 3). Most of these tasks operate on only one image at a time, so you usually need to specify which group of a GEIS file is to be processed. If you do not specify a group, your task will choose the first group by default.

Specifying a Group

To specify a particular group in a GEIS file, append the desired group number in square brackets to the file name (e.g., `z2bd010ft.d0h[10]`). For example, to apply the **imarith** task to group 10 of a GEIS image, type the following:

```
cl> imarith indata.hhh[10] + 77.0 outdata.hhh
```

This command will add 77.0 to the data in group 10 of the file `indata.hhh`, and will write the output to a new single-group file called `outdata.hhh`. Any operation performed on a single group of a multigroup GEIS file results in an output file containing a single group.

Specifying an Image Section

If you wish to process only a portion of an image, you can specify the image section after the group specification in the following manner:

```
cl> imarith indata.hhh[2][100:199,200:399] * 32.0 outdata.hhh
```

This command extracts a 100 by 200 pixel subsection of the image in the second group of the file `indata.hhh`, multiplies this data by a factor of 32.0, and stores the result in a new output file, `outdata.hhh`, which is a 100 by 200 pixel single group GEIS file.

Printing Header Information

As discussed in the previous section, the task **imheader** extracts and prints information about the GEIS image. This task reports the image name, dimensions (including the number of groups), pixel type, and title of the image when it is run in default mode. For example:

```
cl> imhead indata.hhh
    indata.hhh[1/64][500][real]: INDATA[1/64]
```

The output line indicates that `indata.hhh` is a multigroup GEIS file which contains 64 groups of images, each consisting of a spectral array 500 pixels in length. The data type of the values is real (floating point). Note that since no group designation was provided, the task defaulted to the first group. To reveal more information regarding group 10, you can type:

```
cl> imhead indata.hhh[10] long+ | page
```

which will generate a long listing of both the ASCII header parameters in the `*.hhh` file and the specific GPB parameters for group 10 from the `*.hhd` file.

Other Group-Related Tasks

Currently, IRAF or STSDAS tasks cannot process all the groups in an input image and write the results to corresponding groups in an output image. However, there are several STSDAS tasks, particularly in the **toolbox.imgtools** and **hst_calib.ctools** packages, that simplify working with group format data. Please refer to the *STSDAS User's Guide* for more details about working with GEIS images.

Chapter 3
STSDAS Basics

In This Chapter...

Navigating STSDAS / 3-1
Displaying HST Images / 3-4
Analyzing HST Images / 3-8
Displaying HST Spectra / 3-17
Analyzing HST Spectra / 3-21
References / 3-32

The Space Telescope Science Data Analysis System (STSDAS) is the software system for calibrating and analyzing data from the Hubble Space Telescope. The package contains programs—called *tasks*—that perform a wide range of functions supporting the entire data analysis process, from reading tapes, through reduction and analysis, to producing final plots and images. This chapter introduces the basics of STSDAS, showing you how to display your data, leading you through some simple data manipulations, and pointing you towards more sophisticated tasks, some of which are described in the instrument sections of this handbook.

STSDAS is layered on top of the Image Reduction and Analysis Facility (IRAF) software developed at the National Optical Astronomy Observatory (NOAO). Any task in IRAF can be used in STSDAS, and the software is portable across a number of platforms and operating systems. To exploit the power of STSDAS effectively, you need to know the basics of IRAF. If you are not already familiar with IRAF, consult the IRAF Primer in Appendix A before reading further.

3.1 Navigating STSDAS

The tasks in STSDAS are far too numerous and complicated to describe comprehensively in this volume. Instead we will show you where to find the STSDAS tasks appropriate for handling certain jobs. You can refer to online help or the *STSDAS User's Guide* for details on how to use these tasks. Some useful online help commands are:

- `help` *task* - provides detailed descriptions and examples of each task.
- `help` *package* - lists the tasks in a given package and their functions.
- `describe` *task* - provides a detailed description of each task.
- `example` *task* - provides examples of each task.
- `apropos` *word* - searches the online help database for tasks relating to the specified word (see Figure A.4 in Appendix A).

3.1.1 STSDAS Structure

STSDAS version 2.0 is structured so that related tasks are logically grouped. In many cases, you can find a task that performs whatever function you need simply by looking in the appropriate package. For example, all of the tasks that are used in the calibration process can be found in the **hst_calib** package and all tasks used for image display and plotting can be found in the **graphics** package. Figure 3.1 shows the STSDAS package structure. Note that IRAF version 2.11 must be installed on your system in order for you to use STSDAS 2.0 and TABLES version 2.0

3.1.2 Packages of General Interest

Images

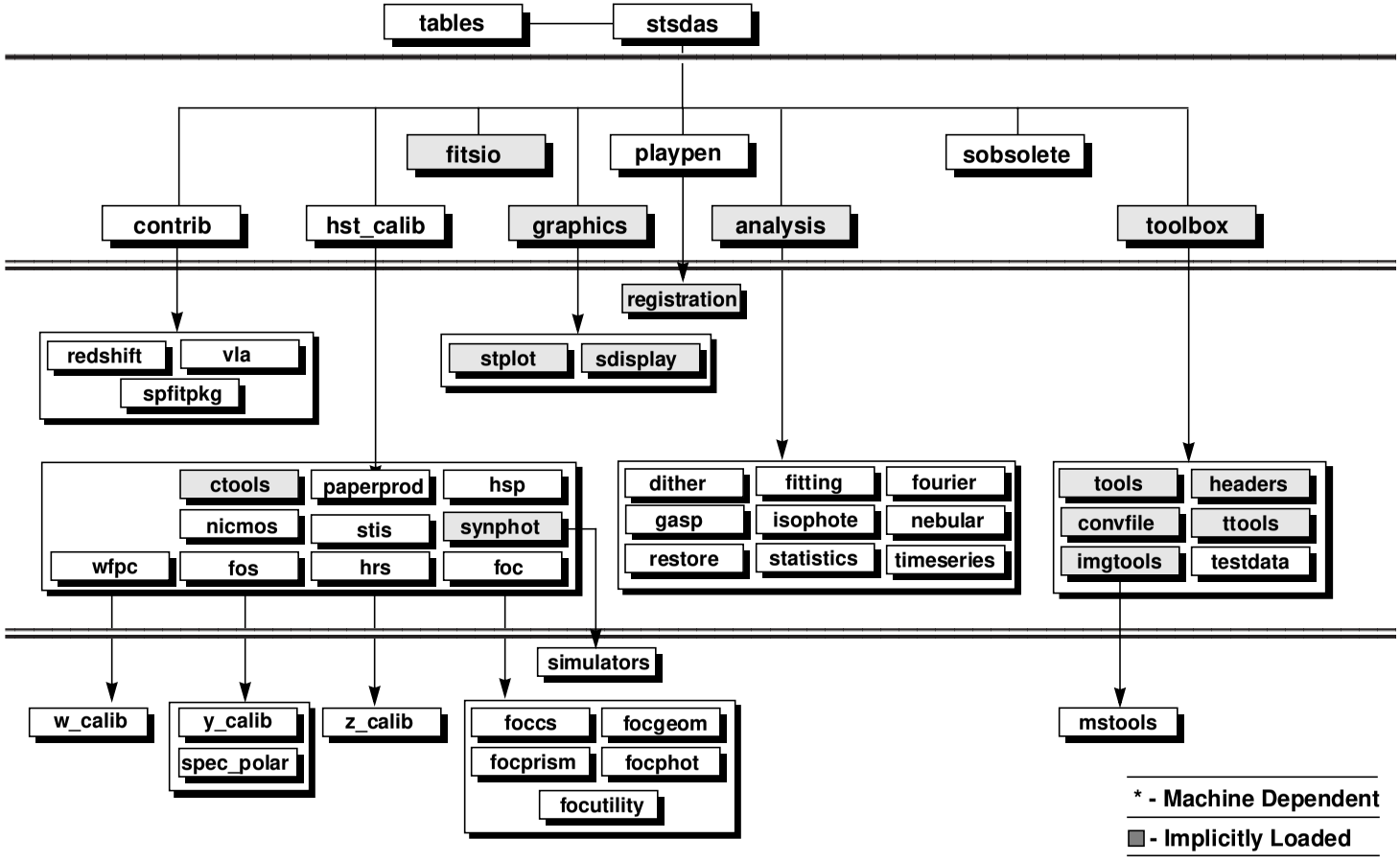
Both IRAF and STSDAS contain a large number of tasks that work with HST images. Some of the packages you should investigate are:

- **images**: This package includes general tasks for copying (**imcopy**), moving (**imrename**), and deleting (**imdelete**) image files. These tasks operate on both the header and data portions of the image. The package also contains a number of general purpose tasks for operations such as rotating and magnifying images.
- **stsdas.toolbox.imgtools**: This package contains general tools for working with multigroup GEIS images, including tasks for working with masks, and general purpose tasks for working with the pixel data, such as an interactive pixel editor (**pixedit**).
- **stsdas.toolbox.imgtools.mstools**: This package contains tools for working with FITS image extensions, in particular STIS and NICMOS image sets (imsets).
- **stsdas.analysis**: This package contains general tasks for image analysis.

Tables

Several of the analysis packages in STSDAS, including calibration pipeline tasks, create output files in STSDAS table format, which is a binary row-column format, or in FITS binary table format. (ASCII-format tables are also supported, for input only.) The *STSDAS User's Guide* describes the STSDAS table format in

Figure 3.1: STSDAS Version 2.0 Package Structure



detail. Tasks in the **ttools** package or in the external **tables** package can be used to read, edit, create, and manipulate tables. For example:

- **tread** displays a table, allowing you to move through it with the arrow keys.
- **tprint** displays a table.
- **tcopy** copies tables.
- **tedit** allows you to edit a table.

Many other tasks in **ttools** perform a variety of other functions. See the online help for details.

3.2 Displaying HST Images

This section will be of interest primarily to observers whose datasets contain two-dimensional images, as it explains:

- How to display images in IRAF using the **display** task.
- How to display subsections of images.

Observers viewing WF/PC-1 and WFPC2 data may wish to remove cosmic rays before displaying their data (see page 26-20). The FOC photon-counting hardware does not detect cosmic rays at easily as CCDs, the NICMOS pipeline automatically removes cosmic rays from MULTIACCUM observations, and the STIS pipeline automatically removes cosmic rays from CR-SPLIT association products.

3.2.1 The display Task

The most general IRAF task for displaying image data is the **display** task, the best choice for a first look at HST imaging data. To display an image, you need to:

1. Start an image display server, such as SAOimage, in a separate window from your IRAF session, either from a different xterm window or as a background job before starting IRAF. To start SAOimage, type the following in any xterm or other system window:

   ```
   saoimage &
   ```

2. Load the **images.tv** package from the window where you're running IRAF:

   ```
   cl> images
   im> tv
   ```

Several different display servers, including SAOimage, SAOtng (the next generation of SAOimage), and Ximtool, can be used with IRAF. SAOtng may be retrieved via anonymous FTP from `sao-ftp.harvard.edu` in the directory `~ftp/pub/rd`. Ximtool may be retrieved via anonymous FTP from iraf.noao.edu in the directory `~pub/v2103-beta`. Ximtool is particularly handy if you want to blink images.

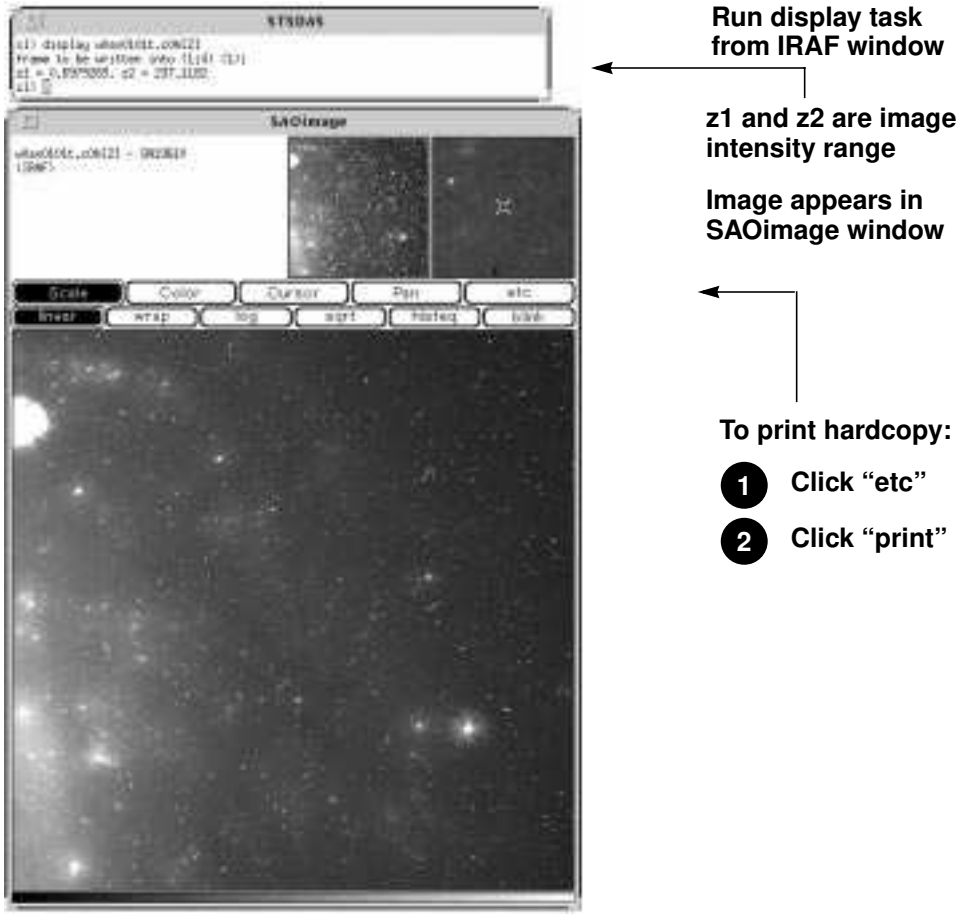
3. Display the image with the IRAF **display** task, using the syntax appropriate for the file format (Chapter 2 explains how to specify GEIS groups and FITS extensions):

```
tv> display fname.c0h[2] 1 (GEIS group 2)
tv> display fname.fits[11] 1 (FITS extension 11)
tv> display fname.fits[sci,3] 1 (FITS extension sci,3)
```

Note that when using **display** or any other task on GEIS images, you do not need to specify a group; the first group is the default. However, when working with FITS files you must specify an extension, unless the FITS file contains only a single image in the primary data unit and has no extensions. Figure 3.2 shows how to display group two of a WF/PC-1 image .

If you want to display all four chips of a WF/PC-1 or WFPC2 image simultaneously, you can create a mosaic with the STSDAS **wmosaic** task in the **hst_calib.wfpc** package. Type `help wmosaic` for details.

Figure 3.2: Displaying an Image



Modifying the Display

There are two ways to adjust how your image is displayed:

- Use the SAOimage command buttons that control zooming, panning, etc.
- Reset the **display** task parameters.

Once an image appears in your SAOimage window, you can use the SAOimage commands displayed near the top of the image window to manipulate or print your image. The *SAOimage Users Guide* describes these commands, although most are fairly intuitive. Just click on the buttons to scale, pan, or print the image, or to perform other commonly-used functions. On-line help is also available at the system level: type `man saoimage` in Unix or `help saoimage` in VMS.

The example in Figure 3.2 shows how you should display an image for a first look. By default, **display** automatically scales the image intensity using a sampling of pixels throughout the image. During your first look, you may want to experiment with the scaling using the `zscale`, `zrange`, `z1` and `z2` parameters. The `zscale` parameter toggles the autoscaling. Setting `zscale-` and `zrange+` tells the task to use minimum and maximum values from the image as the minimum and maximum intensity values. To customize your minimum and

maximum intensity display values, set `zscale-`, `zrange-`, `z1` to the minimum value and `z2` to the maximum value that you want displayed. For example:

```
im> disp w0mw0507v.c0h 1 zrange- zscale- z1=2.78 z2=15.27
```

Notice in Figure 3.2 that when you run **display**, the task shows you the `z1` and `z2` values that it calculates. You can use these starting points in estimating reasonable values for the minimum and maximum intensity display parameters.[1]

If you want to display an image with high dynamic range, you may prefer to use logarithmic scaling. However, the log scaling function in SAOimage divides the selected intensity range into 200 linearly spaced levels before taking the log. The resulting intensity levels are rendered in a linear rather than logarithmic sense. You can often obtain better results if you create a separate logarithmic image to display. One way to create a logarithmic image is with the **imcalc** task:

```
im> imcalc x2ce0502t.c1h x2ce0502t.hhh "log10(im1+1.0)"
```

If the peak pixel in your original image contained 2000 counts, for example, you would then display the logarithmic image with `z1=0` and `z2=3.3`.

3.2.2 Working with Image Sections

Sometimes you may want to display only a portion of an image, using the syntax for specifying image sections discussed in Chapter 2. Your specified pixel range should give the starting point and ending point, with a colon separating the two. List the horizontal (*x* axis) range first, followed by the vertical (*y* axis) range. For example, to specify a pixel range from 101 to 200 in the *x* direction and all pixels in the *y* direction from group three of a GEIS format image, you would use a command such as:

```
tv> display image.hhh[3][101:200,*] 1
```

To specify the same pixel range in the second SCI extension of a NICMOS FITS image, you would use a command such as:

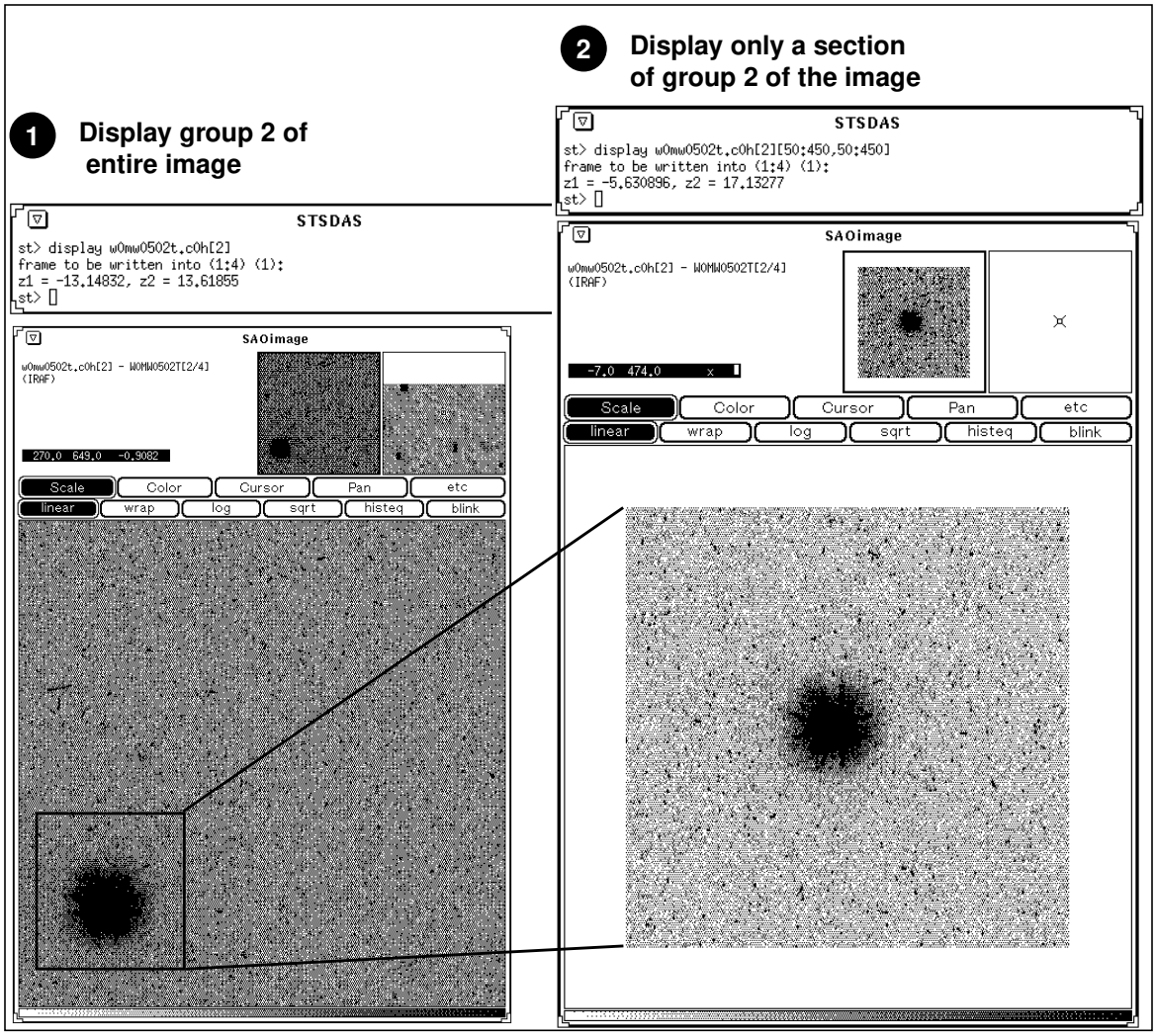
```
tv> display image.fits[sci,2][101:200,*] 1
```

If you specify both a group and an image section of a GEIS file, the group number must come first. When displaying sections of STIS and NICMOS FITS images, you must specify the extension, and the extension designation must come first.

Figure 3.3 shows examples of displaying an image and an image section.

1. Type `help display` within IRAF to obtain more information about these parameters.

Figure 3.3: Displaying Sections and Groups of an Image



3.3 Analyzing HST Images

This section describes methods for using STSDAS and IRAF to work with two-dimensional image data from HST. Subjects include:

- Relating your image to sky coordinates.
- Examining and manipulating your image.
- Working with STIS and NICMOS imsets.
- Converting counts to fluxes.

3.3.1 Basic Astrometry

This section describes how to determine the orientation of an HST image and the RA and Dec of any pixel or source within it, including:

- Tasks that supply positional information about HST images.
- Methods for improving your absolute astrometric accuracy.

Positional Information

The header of every calibrated HST two-dimensional image contains a linear astrometric plate solution, written in terms of the standard FITS astrometry header keywords: CRPIX1, CRPIX2, CRVAL1, CRVAL2, and the CD matrix—CD1_1, CD1_2, CD2_1, and CD2_2. IRAF/STSDAS tasks can use this information to convert between pixel coordinates and RA and Dec. Two simple tasks that draw on these keywords to relate your image to sky coordinates are:

- **disconlab**: Displays your image with a superimposed RA and Dec grid. Simply open an SAOimage window and type, for example:

  ```
  sd> disconlab n3tc01a5r_cal.fits[1]
  ```

- **xy2rd**: Translates *x* and *y* pixel coordinates to RA and Dec. (The task **rd2xy** inverts this operation.) SAOimage displays the current *x,y* pixel location of the cursor in the upper-left corner of the window. To find the RA and Dec of the current pixel, you supply these coordinates to **xy2rd** by typing

  ```
  sd> xy2rd n3tc01a5r_cal.fits[1] x y
  ```

Table 3.1 lists some additional tasks that draw on the standard astrometry keywords.

Observers should be aware that these tasks do not correct for geometric distortion. Only FOC images currently undergo geometric correction during standard pipeline processing (the `.c0h/.c0d` and `.c1h/.c1d` FOC images have been geometrically corrected); STIS images will be geometrically corrected in the pipeline once suitable calibration files are in hand. If you need precise relative astrometry, you should use an instrument-specific task that accounts for image distortion, such as the **metric** task for WF/PC-1 and WFPC2 images, described on page 28-18.

Table 3.1: Additional IRAF and STSDAS Astrometry Tasks

Task	Purpose
compass	Plot north and east arrows on an image.
north	Display the orientation of an image based on keywords.
rimcursor	Determine RA and Dec of a pixel in an image.
wcscoords	Use WCS[a] to convert between IRAF coordinate systems.
wcslab	Produce sky projection grids for images.

a. World Coordinate System (WCS). Type “help specwcs” at the IRAF prompt for details.

Do not use tasks like **rimcursor** or **xy2rd** directly on WF/PC-1 or WFPC2 images if you require accurate relative positions. Calibrated WF/PC-1 and WFPC2 images retain a residual distortion which will affect the accuracy of relative positions. Both **wmosaic** and **metric**, found in the **stsdas.hst_calib.wfpc** package, correct for this distortion.

Improving Astrometric Accuracy

Differential astrometry (measuring a position of one object relative to another in an image) is easy and relatively accurate for HST images, while absolute astrometry is more difficult, owing to uncertainties in the locations of the instrument apertures relative to the Optical Telescope Assembly (OTA or V1) axis and the inherent uncertainty in the Guide Star positions. However, if you can determine an accurate position for any single star in your HST image, then your absolute astrometric accuracy will be limited only by the accuracy with which you know that star's location and the image orientation.

If there is a star on your image suitable for astrometry, you may wish to extract an image of the sky around this star from the Digitized Sky Survey and measure the position of that star using, for example, the GASP software (described in the *STSDAS User's Guide*). These tools can provide an absolute positional accuracy of approximately 0″.7. Contact the Help Desk for assistance (send E-mail to `help@stsci.edu`).

3.3.2 Examining and Manipulating Image Data

This section describes **implot** and **imexamine**, two basic IRAF tools for studying the characteristics of an image, and Table 3.3 lists some useful IRAF/STSDAS tasks for manipulating image data.

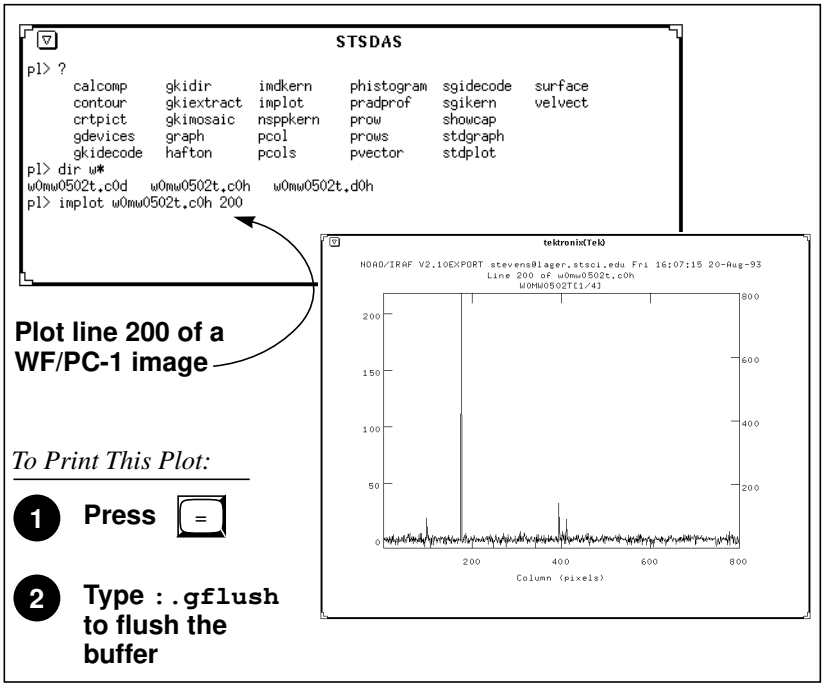
implot

The IRAF **implot** task (in the **plot** package) allows you to examine an image interactively by plotting data along a given *line* (*x* axis) or *column* (*y* axis). When you run the task, a large number of commands are available in addition to the usual cursor mode commands common to most IRAF plotting tasks. A complete listing of commands is found in the on-line help, but the most commonly used are listed in Table 3.2. Figure 3.4 shows an example of how to use the **implot** task.

Table 3.2: Basic implot Commands

Keystroke	Command
?	Display on-line help.
L	Plot a line.
C	Plot a column.
Q	Quit implot.
J	Move down.
K	Move up.
Space	Display coordinates and pixel values.

Figure 3.4: Plotting Image Data with implot



imexamine

The IRAF **imexamine** task (in the **images.tv** package) is a powerful task that integrates image display with various types of plotting capabilities. Commands can be passed to the task using the image display cursor and the graphics cursor. A complete description of the task and its usage are provided in the online help, available from within the IRAF environment by typing `help imexamine`.

Table 3.3: Image Manipulation Tasks

Task	Package	Purpose
boxcar	images.imfilter	Boxcar smooth a list of images
gcombine	stsdas.toolbox.imgtools	Combine images using various algorithms and rejection schemes
gcopy	stsdas.toolbox.imgtools	Copy GEIS multigroup images
geomap	images.immatch	Compute a coordinate transformation
geotran	images.immatch	Resample an image based on geomap output
grlist	stsdas.graphics.stplot	List of file names of all groups of a GEIS image (to make @lists)
gstatistics	stsdas.toolbox.imgtools	Compute image statistics[a]
imcalc	stsdas.toolbox.imgtools	Perform general arithmetic on GEIS images[a]
imedit	images.tv	Fill in regions of an image by interpolation
imexamine	images.tv	Examine images using display, plots, and text (see page 3-11)
implot	plot	Plot lines and columns of images (see page 3-10)
magnify	images.imgeom	Magnify an image
msarith	stsdas.toolbox.mstools	Performs basic arithmetic on STIS and NICMOS imsets
mscombine	stsdas.toolbox.mstools	Extension of **gcombine** for STIS and NICMOS imsets
msstatistics	stsdas.toolbox.mstools	Extension of **gstatistics** for STIS and NICMOS imsets
newcont	stsdas.graphics.stplot	Draw contours of two-dimensional data
pixcoord	stsdas.hst_calib.wfpc	Compute pixel coordinates of stars in a GEIS image
plcreate	xray.ximages	Create a pixel list from a region file (e.g., from SAOimage)
rotate	images.imgeom	Rotate an image
saodump	stsdas.graphics.sdisplay	Make image and colormap files from SAOimage display
siaper	stsdas.graphics.stplot	Plot science instrument apertures of HST

a. Will process all groups of a multigroup GEIS file.

3.3.3 Working with STIS and NICMOS Imsets

STIS and NICMOS data files contain groups of images, called imsets, associated with each individual exposure. A STIS imset comprises SCI, ERR, and DQ images, which hold science, error, and data quality information. A NICMOS imset, in addition to its SCI, ERR, and DQ images, also contains TIME and SAMP images recording the integration time and number of samples corresponding to each pixel of the SCI image. See the STIS and NICMOS Data Structures chapters for more details on imsets.

Here we describe several new STSDAS tasks, located in the **stsdas.toolbox.imgtools.mstools** package, that have been designed to help you work with with imsets as units and to deconstruct and rebuild them.

msarith

This tool is an extension of the IRAF task **imarith** to include error and data quality propagation. The **msarith** task supports the four basic arithmetic operations (+, -, *, /) and can operate on individual or multiple imsets. The input operands can be either files or numerical constants; the latter can appear with an associated error, which will propagate into the error array(s) of the output file. Table 3.4 below shows how this task operates on the SCI, ERR, and DQ images in a STIS or NICMOS imset, as well as the additional TIME and SAMP images belonging to NICMOS imsets:

Table 3.4: Task msarith Operations

Operation	Operand2	SCI	ERR	DQ	TIME	SAMP
ADD	file	op1+op2	$\sqrt{\sigma 1^2 + \sigma 2^2}$	OR	T1+T2	S1+S2
SUB	file	op1-op2	$\sqrt{\sigma 1^2 + \sigma 2^2}$	OR	T1	S1
MULT	file	op1*op2	$(op1 \times op2)\sqrt{(\sigma 1/op1)^2 + (\sigma 2/op2)^2}$	OR	T1	S1
DIV	file	op1/op2	$(op1/op2)\sqrt{(\sigma 1/op1)^2 + (\sigma 2/op2)^2}$	OR	T1	S1
ADD	constant	op1+op2	$\sqrt{\sigma 1^2 + \sigma 2^2}$	...	...	...
SUB	constant	op1-op2	$\sqrt{\sigma 1^2 + \sigma 2^2}$	...	...	...
MULT	constant	op1*op2	$(op1 \times op2)\sqrt{(\sigma 1/op1)^2 + (\sigma 2/op2)^2}$	...	T1*op2	...
DIV	constant	op1/op2	$(op1/op2)\sqrt{(\sigma 1/op1)^2 + (\sigma 2/op2)^2}$	...	T1*op2	...

In Table 3.4 the first operand (op1) is always a file, and the second operand (op2) can be either a constant or a file. The ERR arrays of the input files ($\sigma 1$ and $\sigma 2$) are added in quadrature. If the constant is given with an error ($\sigma 2$), the latter is added in quadrature to the input ERR array. Note that in Table 3.4 the pixels in the SCI images are in counts, but **msarith** can also operate on count rates.

mscombine

This task allows you to run the STSDAS task **gcombine** on STIS and NICMOS data files. It divides each imset into its basic components (SCI, ERR, and DQ, plus SAMP and TIME for NICMOS) to make them digestible for **gcombine**. The SCI extensions become the inputs proper to the underlying **gcombine** task, and the ERR extensions become the error maps. The DQ extensions are first combined with a user-specified Boolean mask allowing selective pixel masking and then fed ino the data quality maps. If scaling by exposure time is requested, the exposure times of each imset are read from the header keyword PIXVALUE in the TIME extensions.

Once **gcombine** finishes, **mscombine** reassembles the individual output images into imsets and outputs them as a STIS or NICMOS data file. The output images and error maps from **gcombine** form the SCI and ERR extensions of the

output imset. The DQ extension will be a combination of the masking operations and the rejection algorithms executed by **gcombine**. For NICMOS, the TIME extension will be the sum of the TIME values from the input files minus the rejected values, divided on a pixel-by-pixel basis by the number of valid pixels in the output image. The final TIME array will be consistent with the output SCI image (average or median of the science data). The SAMP extension for NICMOS is built from all the input SAMP values, minus the values discarded by masking or rejection.

msstatistics

This tool is an extension of **gstatistics** in the STSDAS package, which is in turn an extension of **imstatistics**. The main novelty is the inclusion of the error and data quality information included with STIS and NICMOS images in computing statistical quantities. In addition to the standard statistical quantities (min, max, sum, mean, standard deviation, median, mode, skewness, kurtosis), two additional quantities have been added to take advantage of the error information: the weighted mean and the weighted variance of the pixel distribution. If x_i is the value at the i-th pixel, with associated error σ_i, the weighted mean and variance used in the task are:

$$\langle x \rangle_w = \frac{\sum_i \frac{x_i}{\sigma_i \times \sigma_i}}{\sum_i \frac{1}{\sigma_i \times \sigma_i}}$$

and:

$$\langle \sigma \rangle_w^2 = \frac{1}{\sum_i \frac{1}{\sigma_i \times \sigma_i}}$$

The data quality information carried by the STIS or NICMOS file is used to reject pixels in the statistical computation. Users can supply additional masks to reject objects or regions from the science arrays.

mssplit and msjoin

The **mssplit** task extracts user-specified imsets from a STIS or NICMOS data file and copies them into separate files. Each output file contains a single imset along with the primary header of the original file. You might find this task useful for reducing the size of a STIS or NICMOS file containing many imsets or for performing analysis on a specific imset. The **msjoin** task inverts the operation of **mssplit**: it assembles separate imsets into a single data file.

There are additional tasks in this package for deleting and sorting imsets, as well as tasks for addressing a specific image class within an imset.

3.3.4 Photometry

Included in this section are:

- A list of IRAF/STSDAS tasks useful for determining source counts.
- Instructions on how to use header keyword information to convert HST counts to fluxes or magnitudes.
- A brief description of **synphot**, the STSDAS synthetic photometry package.

IRAF and STSDAS Photometry Tasks

The following are some useful IRAF/STSDAS packages and tasks for performing photometry on HST images:

- **apphot**: aperture photometry package.
- **daophot**: stellar photometry package useful for crowded fields.
- **isophote**: package for fitting elliptical isophotes.
- **imexamine**: performs simple photometry measurements.
- **imstat**: computes image pixel statistics.
- **imcnts**: sums counts over a specified region, subtracting background.
- **plcreate**: creates pixel masks.

Consult the online help for more details on these tasks and packages. The document "Photometry using IRAF" by Lisa A. Wells, provides a general guide to performing photometry with IRAF; it is available through the IRAF web page:

`http://iraf.noao.edu/`

The **apphot** package allows you to measure fluxes within a series of concentric apertures. This technique can be used to determine the flux in the wings of the PSF, which is useful if you wish to estimate the flux of a saturated star by scaling the flux in the wings of the PSF to an unsaturated PSF.

Converting Counts to Flux or Magnitude

All calibrated HST images record signal in units of counts or Data Numbers (DN)[2]—NICMOS data is DN s^{-1}. The pipeline calibration tasks do not alter the units of the pixels in the image. Instead they calculate and write the inverse sensitivity conversion factor (PHOTFLAM) and the ST magnitude scale zero point (PHOTZPT) into header keywords in the calibrated data. WF/PC-1 and WFPC2 observers should note that the four chips are calibrated individually, so these photometry keywords belong to the group parameters for each chip.

2. Except for 2-D rectified STIS images, which are in units of I_λ (see Chapter 23).

For all instruments other than NICMOS, PHOTFLAM is defined to be the *mean* flux density F_λ in units of erg cm^{-2} s^{-1} Å^{-1} that produces 1 count per second in the HST observing mode (PHOTMODE) used for the observation. If the F_λ spectrum of your source is significantly sloped across the bandpass or contains prominent features, such as strong emission lines, you may wish to recalculate the inverse sensitivity using **synphot**, described below. WF/PC-1 observers should note that the PHOTFLAM value calculated during pipeline processing does not include a correction for temporal variations in throughput owing to contamination buildup. Likewise, FOC observers should note that PHOTFLAM values determined by the pipeline before May 18, 1994 do not account for sensitivity differences in formats other than 512 x 512 (see "Format-Dependent Sensitivity" on page 7-10).

To convert from counts or DN to flux in units of erg cm^{-2} s^{-1} Å^{-1}, multiply the total number of counts by the value of the PHOTFLAM header keyword and divide by the value of the EXPTIME keyword (exposure time). You can use the STSDAS task **imcalc** to convert an entire image from counts to flux units. For example, to create a flux-calibrated output image `outimg.fits` from an input image `inimg.fits[1]` with header keywords PHOTFLAM = 2.5E-18 and EXPTIME = 1000.0, you could type:

```
st> imcalc inimg.fits[1] outimg.fits "im1*2.5E-18/1000.0"
```

Calibrated NICMOS data are in units of DN s^{-1}, so the PHOTFLAM values in their headers are in units of erg cm^{-2} Å^{-1}. You can simply multiply these images by the value of PHOTFLAM to obtain fluxes in units of erg cm^{-2} s^{-1} Å^{-1}. NICMOS headers also contain the keyword PHOTFNU in units of Jy s. Multiplying your image by the PHOTFNU value will therefore yield fluxes in Janskys.

If your HST image contains a source whose flux you know from ground based measurements, you may choose to determine the final photometry of your HST image from the counts observed for this source.

To convert a measured flux *F*, in units of erg cm^{-2} s^{-1} Å^{-1}, to an ST magnitude, plug it into the following equation:

$$m = -2.5 \times \log10\,(F) + \text{PHOTZPT}$$

where the value of the PHOTZPT keyword is the zero point of the ST magnitude scale. The zero point of the ST magnitude system has always been and probably always will be equal to –21.10, a value chosen so that Vega has an ST magnitude of zero for the Johnson *V* passband (see Koornneef et al., 1986; Horne, 1988; and the *Synphot Users Guide*).

synphot

The STSDAS synthetic photometry package, called **synphot**, can simulate HST observations of astronomical targets with known spectra. It contains throughput curves for the components of all HST instruments, such as mirrors, filters, gratings, apertures, and detectors, and can generate passband shapes for

any combination of these elements. It can also generate synthetic spectra of many different types, including stellar, blackbody, power-law and H II region spectra, and can convolve these spectra with the throughputs of HST's instruments. You can therefore use it to compare results in many different bands, to cross-calibrate one instrument with another, or to relate your observations to theoretical models.

One useful application of **synphot** is to recalculate the value of PHOTFLAM for a given observation using the latest calibration files. For example, to recalculate PHOTFLAM for an FOC observation, you could use the **calcphot** task in **synphot** as follows:

```
sy> calcphot foc,f/96,x96zlrg,f501n 'unit(1,flam)' counts
```

The first argument to **calcphot** gives the instrument and its configuration, in this case the FOC *f*/96 camera in full zoomed format with the F501 filter. (See the **obsmode** task in **synphot** and the *Synphot User's Guide* for help with these observation-mode keywords.) The second tells the task to model a flat F_λ spectrum having unit flux, and the third tells the task to produce output in units of counts per second. After you run **calcphot**, its `result` parameter will contain the count rate expected from the FOC, given this configuration and spectrum. The PHOTFLAM keyword, defined to be the flux required to produce one count per second, simply equals the reciprocal of this value, which you can print to the screen by typing `=1./calcphot.result` at the IRAF prompt.

Please see the *Synphot User's Guide* for more details on this multipurpose package, and see Appendix A for information on getting the **synphot** dataset, which is not included with STSDAS.

Information about retrieving the synphot dataset can be found in Appendix A.

3.4 Displaying HST Spectra

This section shows how to plot your HST spectra for a quick first look and how to generate hardcopies of your plots. Because the STIS data format differs from that of FOS and GHRS, we will discuss STIS data separately.

3.4.1 FOS and GHRS Spectra

Before you work with FOS and GHRS data within STSDAS, you will want to convert the FITS files you received from the Archive into GEIS format (see "Converting FITS to GEIS" on page 2-11 for instructions). After conversion, the `.c1h` file will hold the calibrated flux values for each pixel, the `.c0h` file will hold the corresponding wavelengths, and the `.c2h` file will hold the propagated statistical errors.

Each group of an FOS or GHRS GEIS file contains the results of a separate subintegration. FOS readouts taken in ACCUM mode are cumulative, so the last group contains the results of the entire integration. In contrast, GHRS readouts and FOS readouts in RAPID mode are independent. If you want to see the results

of an entire GHRS FP-SPLIT integration, you will need to align and coadd the spectra in the groups of the GHRS file as described in Volume 2 of this handbook. You can also combine all the groups in an FOS or GHRS data file, without wavelength alignment, using the **rcombine** task in the **hst_calib.ctools** package. See online help for details.

The STSDAS task **sgraph** (in the **graphics.stplot** package) can plot the contents of a single GEIS group. For example, if you want to see group 19 of the calibrated FOS spectrum with rootname `y3bl0104t` you can type

```
st> sgraph y3bl0104t.c1h[19]
```

Given an input flux image (`.c1h`), the task **fwplot** (in the **hst_calib.ctools** package) will look for the corresponding wavelength (`.c0h`) file and plot flux versus wavelength. If requested, it will also look for the error (`.c2h`) file and plot the error bars. To see a plot of the same spectrum as above, but with a wavelength scale and error bars, type

```
st> fwplot y3bl0104t.c1h[19] plterr+
```

If you ever need to plot the contents of multiple groups offset from one another on the same graph, you can use the **grspec** task in the **graphics.stplot** package. For example, to plot groups 1, 10, and 19 of a given flux file, you can type

```
st> grspec y3bl0104t.c1h 1,10,19
```

Note that **grspec** expects group numbers to be listed as a separate parameter, rather than enclosed in the standard square brackets.

3.4.2 STIS Spectra

STIS data files retrieved from the Archive can contain spectra in two different forms: as long-slit spectral images in FITS IMAGE extensions or as extracted echelle spectra in FITS BINTABLE extensions. Currently only echelle spectra emerge from the pipeline in tabular form, while long-slit spectra emerge as images.

You can use **sgraph** to plot STIS long-slit spectra by specifying the image section that contains the spectrum. For example, to plot the entire *x* range of the calibrated two-dimensional spectrum in the first extension of the file `o43ba1bnm_x2d.fits`, averaging rows 100 through 1000, you would type

```
st> sgraph o43ba1bnm_x2d.fits[1][*,100:1000]
```

Displaying the long-slit spectral image using the **display** task (see page 3-4) will allow you to see the range of your spectrum in *x* and *y* pixel space, so you can choose a suitable image section for plotting.

To plot STIS spectra in BINTABLE extensions, you first need to understand how STIS spectra are stored as binary arrays in FITS table cells. Chapter 2 (page 2-9) discusses this format and describes the *selectors* syntax used to specify these data arrays. Each row of a STIS echelle table contains a separate spectral order, and each column contains data of a certain type, such as WAVELENGTH data or FLUX data. To specify a particular array, you must first type the file name, then the extension containing the BINTABLE, then the column selector, then the

row selector. For example, to select the WAVELENGTH array corresponding to spectral order 80 of the echelle spectrum in extension 4 of `stis.fits`, you would specify the file as:

```
stis.fits[4][c:WAVELENGTH][r:sporder=80]
```

The **sgraph** task and the **igi** plotting package to be discussed below both understand the *selectors* syntax. In particular, if you wanted to plot the flux versus wavelength in STIS echelle order 80, you could type:

```
st> sgraph "stis.fits[4][r:sporder=80] WAVELENGTH FLUX"
```

Remember to include the quotation marks. Otherwise, **sgraph** will complain about too many positional arguments. Note also that **sgraph** understands only row selector syntax; columns are chosen by name.

The STIS-specific **echplot** task is particularly useful for browsing STIS echelle spectra. It can plot single spectral orders, overplot multiple orders on a single plot, or plot up to four orders in separate panels on the same page. For example, to overplot the orders contained in rows two through four and row six on a single page:

```
cl> echplot "stis_x1d.fits[1][r:row=(2:4,6)]" output.igi \
>>> plot_style=m
```

Note that the `plot_style` parameter governs how the spectral orders are plotted. The `plot_style` values `s`, `m`, and `p` plot one order per page, several orders on a single plot, and one order per panel, respectively. The default brightness unit is calibrated FLUX, although you can specify other quantities (e.g., NET counts) using the `flux_col` parameter. See the online help for details.

3.4.3 Producing Hardcopy

This section shows how to generate hardcopies of plots directly and describes **igi**, the Interactive Graphics Interpreter available in STSDAS.

Direct Hardcopies

To print a quick copy of the displayed plot:

1. Type `=gcur` in the command window (where your CL prompt is located).
2. Move the cursor to any location in the graphics window.
3. Press [=] to write the plot to the graphics buffer.
4. Type `q` to exit graphics mode.
5. At the cl prompt, type `gflush`.

Plots will be printed on the printer defined by the IRAF environment variable `stdplot`. Type `show stdplot` to see the current default printer; use `set stdplot =` *printer_name* to set the default printer.

The PostScript kernel **psikern** allows you to create PostScript files of your IRAF/STSDAS plots. For example, setting the `device` parameter in a plotting task equal to `psi_port` or `psi_land` invokes **psikern** and directs your plot to either a portrait-mode or a landscape mode PostScript file. For example:

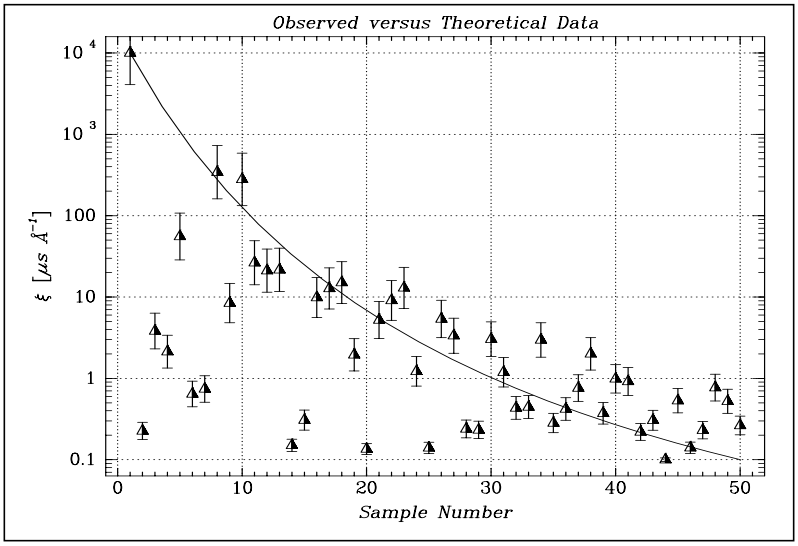
```
st> fwplot y3bl0104t.c1h[19] device=psi_land
st> gflush
/tmp/psk<i>xxxx</i>
```

The above commands would write a plot of flux vs. wavelength in landscape-mode into a temporary PostScript file, named `/tmp/psk`*xxxx* by a UNIX system. See the online help for more about **psikern**, including plotting in color and incorporating PostScript fonts into your plots.

igi

As your plotting needs grow more sophisticated—and especially as you try preparing presentations or publication-quality plots—you should investigate the Interactive Graphics Interpreter, or **igi**. This task, in the STSDAS **stplot** package, can be used with images as well as two- and three-dimensional tables and can draw axes, error bars, labels, and a variety of other features on plots. Different line weights, font styles, and feature shapes are available, enabling you to create complex plots. Figure 3.5 shows a sample plot created in **igi**, however, because **igi** is a complete graphics environment in itself, it is well beyond the scope of this document. You can learn more about **igi** in the *IGI Reference Manual*, available through the STSDAS Web pages.

Figure 3.5: Sample igi Plot.



3.5 Analyzing HST Spectra

This section describes some IRAF/STSDAS tasks that can be used for analyzing and manipulating spectral data. Certain of these tasks operate directly on HST data files created by the pipeline. However, a number of the most useful IRAF tasks, such as **splot**, require special preparations of data other than STIS two-dimensional spectra. Before discussing these tasks we will first show how to recast your data into forms that are more generally accessible.

3.5.1 Preparing FOS and GHRS Data

The FOS and GHRS data reduction pipelines store fluxes and wavelengths in separate files. In GEIS format, the `.c1h` file contains the flux information and the `.c0h` file contains the wavelength information. Because IRAF tasks generally require both the flux and wavelength information to reside in the same file, you will probably want to create a new file that combines these quantities.

Several options for combining flux and wavelength information are available:

- **resample**: This simple task resamples your flux data onto a linear wavelength scale, creating a new flux file containing the starting wavelength of the new grid in the CRVAL1 keyword and the wavelength increment per pixel in the CD1_1 keyword. Encoding the wavelength information into these standard FITS header keywords makes this format quite portable, but the resampling process loses some of the original flux information. In addition, the error (`.c2h`) and data quality (`.cqh`) files cannot be similarly resampled, limiting the usefulness of this technique.
- **mkmultispec**: This task writes wavelength information into the header of a flux file while preserving all the original information. It is therefore a better choice than **resample** for most applications, and we describe it in more detail below.
- **imtab**: An alternative to writing wavelength information into the header is to use the **imtab** task to create a table recording the wavelength, flux, and if desired, the error data corresponding to each pixel. Many STSDAS tasks, such as those in the STSDAS **fitting** package, can access data in tabular form, so we describe this approach in more detail as well.

mkmultispec

The most convenient method of combining wavelength and flux information, and one that has no effect on the flux data at all, is to use the **mkmultispec** task. This task places wavelength information into the headers of your flux files according to the IRAF multispec-format World Coordinate System (WCS). The multispec coordinate system is intended to be used with spectra having nonlinear dispersions or with images containing multiple spectra, and the format is recognized by many tasks in IRAF V2.10 or later. For a detailed discussion of the multispec WCS, type `help specwcs` at the IRAF prompt.

The **mkmultispec** task can put wavelength information into the flux header files in two different ways. The first involves reading the wavelength data from the `.c0h` file, fitting the wavelength array with a polynomial function, and then storing the derived function coefficients in the flux header file (`.c1h`) in multispec format. Legendre, Chebyshev, or cubic spline (spline3) fitting functions of fourth order or larger produce essentially identical results, all having rms residuals less than 10^{-4} Å, much smaller than the uncertainty of the original wavelength information. Because these fits are so accurate, it is usually unnecessary to run the task in interactive mode to examine them.

If there are discontinuities in the wavelengths, which could arise due to the splicing of different gratings, you should run **mkmultispec** in interactive mode to verify the fits.

Because **mkmultispec** can fit only simple types of polynomial functions to wavelength data, this method will *not* work well with FOS prism data, because of the different functional form of the prism-mode dispersion solution. For prism spectra, use the header table mode of **mkmultispec** (see below) or create an STSDAS table using **imtab**.

The other method by which **mkmultispec** can incorporate wavelength information into a flux file is simply to read the wavelength data from the `.c0h` file and place the entire data array directly into the header of the flux (`.c1h`) file. This method simply dumps the wavelength value associated with each pixel in the spectrum into the flux header and is selected by setting the parameter `function=table`. To minimize header size, set the parameter `format` to a suitable value. For example, using `format=%8.7g` will retain the original seven digits of precision of the wavelength values, while not consuming too much space in the flux header file.

Be aware that there is a physical limit to the number of header lines that can be used to store the wavelength array (approximately 1000 lines). This limit cannot be overridden. Under ordinary circumstances this limitation is not an issue. However, if many spectral orders have been spliced together, it may not be possible to store the actual wavelength array in the header, and a fit must be done instead.

imtab

Another way to combine wavelengths with fluxes is to create an STSDAS table from your spectrum. The **imtab** task in the STSDAS **ttools** package reads a GEIS format spectral image and writes the list of data values to a column of an STSDAS table, creating a new output table if necessary. The following example shows how

to create a flux, wavelength, and error table from group eight of a GEIS-format FOS dataset:

```
cl> imtab y0cy0108t.c0h[8] y0cy0108t.tab wavelength
cl> imtab y0cy0108t.c1h[8] y0cy0108t.tab flux
cl> imtab y0cy0108t.c2h[8] y0cy0108t.tab error
```

The last word on each command line labels the three columns “wavelength”, “flux”, and “error”.

Constructing tables is a necessary skill if you plan to use certain tasks—such as those in the STSDAS **fitting** package—that do not currently recognize the multispec format WCS header information. Tabulating your spectra is also the best option if you want to join two or more spectra taken with different gratings into a single spectrum covering the complete wavelength range. Because the data are stored as individual wavelength-flux pairs, you do not need to resample, and therefore degrade, the individual spectra to a common, linear dispersion scale before joining them. Instead, you could create separate tables for the spectra from different gratings, and then combine the two tables using, for example, the **tmerge** task:

```
cl> tmerge n5548_h13.tab,n5548_h19.tab n5548.tab append
```

Note that you will first have to edit out any regions of overlapping wavelength from one or the other of the input tables so that the output table will be monotonically increasing (or decreasing) in wavelength.

3.5.2 Preparing STIS Spectra for Analysis

Calibrated STIS spectra emerge from the pipeline either as two-dimensional images (_x2d files) or as one-dimensional spectra in tabular form (_x1d files.) You can analyze calibrated two-dimensional STIS spectra in IRAF as you would any other long-slit spectral image, because their headers already contain the necessary wavelength information. Tabulated STIS spectra can be analyzed directly using STSDAS tasks that understand the *selectors* syntax described on page 2-9. However, to use IRAF tasks, such as **splot**, that rely on the multispec WCS or to use STSDAS tasks that do not understand three-dimensional tables, you will have to prepare your data appropriately. This section describes two useful tasks for putting your data in the proper form:

- **tomultispec**: This task is the STIS analog to **mkmultispec**, described above; it extracts STIS spectra from tables and writes them as IRAF spectral images with wavelength information in the header.
- **txtable**: This task extracts specified data arrays from STIS table cells and places them in conventional two-dimensional tables for easier access.
- **tximage**: Extracts specified data arrays from STIS table cells and places them into 1-D images. This task can write single group GEIS files.

tomultispec

The **tomultispec** task in the **stsdas.hst_calib.ctools** package extracts one or more spectral orders from a STIS table, fits a polynomial dispersion solution to

each wavelength array, and stores the spectra in an output file in original IRAF format (OIF), using the multispec WCS. This task is layered upon the **mkmultispec** task, which performs a similar operation for FOS and GHRS calibrated spectra (see page 3-21). Most of the parameters for **tomultispec** echo those for **mkmultispec**. As a helpful navigational aid, the STIS spectral order numbers are written to the corresponding *beam* numbers in the multispec image; the aperture numbers are indexed sequentially starting from one. You can choose to fit the dispersion solution interactively, but the default fourth-order Chebyshev polynomial will likely suffice for all STIS spectral orders, except for prism-dispersed spectra. However, you cannot use the interactive option if you are selecting more than one order from the input file.

For example, if you want to write all spectral orders from the STIS file `myfile_x1d.fits` to a multispec file, you can type:

```
cl> tomultispec myfile_x1d.fits new_ms.imh
```

Note that the `.imh` suffix on the output file specifies that the output file is to be an OIF file. This format is similar to GEIS format, in that it consists of two files: a header file (`.imh`) and a binary data file (`.pix`). The output format for **tomultispec** will always be OIF.

If you want to select particular spectral orders, rather than writing all the orders to the multispec file, you will need to use the **selectors** syntax. To select only the spectrum stored in row nine of the input table, the previous example would change to:

```
cl> tomultispec "myfile_x1d.fits[r:row=9]" new_ms.imh
```

Note that the double quote marks around the file name and row selector are necessary to avoid syntax errors. To select a range of rows, say rows nine through eleven, you would type:

```
cl> tomultispec "myfile_x1d.fits[r:row=(9:11)]" new_ms.imh
```

You can also select rows based upon values in some other column. For example, to select all rows whose spectral order lies in the range 270 to 272, type:

```
cl> tomultispec "myfile_x1d.fits[r:sporder=(270:272)]" \
>>> new_ms.imh
```

The calibrated flux is extracted by default. However, other intensity data can be specified by setting the `flux_col` parameter.

Be careful not to restrict the search for matching rows too heavily.

Column selectors cannot be used with **tomultispec**.

Choose the type of fitting function for the **tomultispec** dispersion solution with care. Using the `table` option, which writes the entire wavelength array to the image header for each order, will fail if more than about three orders are selected. This restriction results from a limit to the number of keywords that can be used to store the dispersion relation.

txtable

Tabulated STIS spectra are stored as data arrays within individual cells of FITS binary tables (see Chapter 2, page 2-9). These tables are effectively three-dimensional, with each column holding a particular type of quantity (e.g., wavelengths, fluxes), each row holding a different spectral order, and each cell holding a one-dimensional array of values spanning the wavelength space of the order. The **txtable** in the **tables.ttools** package extracts these data arrays from the cells specified with the selectors syntax and stores them in the columns of conventional two-dimensional binary tables.

For example, suppose the first extension of the FITS file `data.fits` contains a STIS echelle spectrum and you want to extract only the wavelength and flux arrays corresponding to spectral order 68. You could then type:

```
tt> txtable "data.fits[1][c:WAVELENGTH,FLUX][r:sporder=68]" \
>>> out_table
```

This command would write the wavelength and flux arrays to the columns of the output table `out_table`. To specify multiple rows in a tabulated echelle spectrum, you would type:

```
tt> txtable "data.fits[1][c:WAVELENGTH,FLUX][r:row=(10:12)]" \
>>> echl
```

This command would generate three separate output files named `echl_r0010.tab`, `echl_r0011.tab`, and `echl_r0012.tab`.

See the online help for more details on **txtable** and the selectors syntax, and remember to include the double quotation marks.

The similar **tximage** task can be used to generate single-group GEIS files from STIS data, which can then be used as input to tasks such as **resample**.

```
tt> tximage "data.fits[1][c:WAVELENGTH][r:row=4]" wave.hhh
tt> tximage "data.fits[1][c:FLUX][r:row=4]" flux.hhh
```

3.5.3 General Tasks for Spectra

IRAF has many tasks for analyzing both one- and two-dimensional spectral data. Many observers will already be familiar with **noao.onedspec** and **noao.twodspec** packages, and those who are not should consult the online help. Table 3.5 lists some of the more commonly used IRAF/STSDAS spectral analysis tasks, and below we briefly describe **splot**, one of the most versatile and useful. Remember that many of these tasks expect to find WCS wavelength information

in the header, so you should first run **mkmultispec** or **tomultispec** on your data, if necessary.

Table 3.5: Tasks for Working with Spectra

Task	Package	Input Format	Purpose
boxcar	images.imfilter	Image	Boxcar smooth a list of images
bplot	noao.onedspec	Multispec image	Plot spectra non-interactively
continuum	noao.onedspec	Image	Continuum normalize spectra
fitprofs	noao.onedspec	Image	Non-interactive Gaussian profile fitting to features in spectra and image lines
gcopy	stsdas.toolbox.imgtools	GEIS image	Copy multigroup images
grlist	stsdas.graphics.stplot	GEIS image	List file names for all groups in a GEIS image; used to make lists for tasks that do not use group syntax
grplot	stsdas.graphics.stplot	GEIS image	Plot arbitrary lines from 1-D image; overplots multiple GEIS groups; no error or wavelength information is used
grspec	stsdas.graphics.stplot	Multispec GEIS image	Plot arbitrary lines from 1-D image; stack GEIS groups
magnify	images.imgeom	Image	Interpolate spectrum on finer (or coarser) pixel scale
nfit1d	stsdas.analysis.fitting	Image, table	Interactive 1-D non-linear curve fitting (see page 3-29)
ngaussfit	stsdas.analysis.fitting	Image, table	Interactive 1-D multiple Gaussian fitting (see page 3-29)
poffsets	stsdas.hst_calib.ctools	GEIS image	Determine pixel offsets between shifted spectra
rapidlook	stsdas.hst_calib.ctools	GEIS image	Create and display a 2-D image of stacked 1-D images
rcombine	stsdas.hst_calib.ctools	GEIS image	Combine (sum or average) GEIS groups in a 1-D image with option of propagating errors and data quality values
resample	stsdas.hst_calib.ctools	GEIS image	Resample FOS and GHRS data to a linear wavelength scale (see page 3-21)
sarith	noao.onedspec	Multispec image	Spectrum arithmetic
scombine	noao.onedspec	Multispec image	Combine spectra
sfit	noao.onedspec	Multispec image	Fit spectra with polynomial function
sgraph	stsdas.graphics.stplot	Image, table	Plot spectra and image lines; allows overplotting of error bars and access to wavelength array (see page 3-17)
specalign	stsdas.hst_calib.ctools	GEIS image	Align and combine shifted spectra (see **poffsets**)
specplot	noao.onedspec	Multispec image	Stack and plot multiple spectra
splot	noao.onedspec	Multispec image	Plot and analyze spectra & image lines (see page 3-27)

splot

The **splot** task in the IRAF **noao.onedspec** package is a good general analysis tool that can be used to examine, smooth, fit, and perform simple arithmetic operations on spectra. Because it looks in the header for WCS wavelength information, your file must be suitably prepared. Like all IRAF tasks, **splot** can work on only one group at a time from a multigroup GEIS file. You can specify which GEIS group you want to operate on by using the square bracket notation, for example:

```
cl> splot y0cy0108t.c1h[8]
```

If you don't specify a group in brackets, **splot** will assume you want the first group. In order to use **splot** to analyze your FOS or GHRS spectrum, you will first need to write the wavelength information from your `.c0h` file to the header of your `.c1h` files in WCS, using the **mkmultispec** task (see page 3-21).

The **splot** task is complex with *many* available options described in detail in the online help. Table 3.6 summarizes a few of the more useful cursor commands for quick reference. When you are using **splot**, a log file saves results produced by the equivalent width or de-blending functions. To specify a file name for this log file, you can set the `save_file` parameter by typing, for example:

```
cl> splot y0cy0108t.c1h[8] save_file=results.log
```

If you have used **tomultispec** to transform a STIS echelle spectrum into `.imh/.pix` OIF files with WCS wavelength information (see page 3-23), you can step through the spectral orders stored in image lines using the ")", "(", and "#" keys. To start with the first entry in your OIF file, type:

```
cl> splot new_ms.imh 1
```

You can then switch to any order for analysis using the ")" key to increment the line number, the "(" key to decrement, and the "#" key to switch to a specified image line. Note the beam label that gives the spectral order cannot be used for navigation. See the online help for details.

Table 3.6: Useful splot Cursor Commands

Command	Purpose
Manipulating spectra	
f	Arithmetic mode; add and subtract spectra
l	Convert spectrum from f_ν to f_λ (invert transformation with "n")
n	Convert spectrum from f_λ to f_ν
s	Smooth with a boxcar
u	Define linear wavelength scale using two cursor markings
Fitting spectra	
d	Mark two continuum points & de-blend multiple Gaussian line profiles
e	Measure equivalent width by marking points around target line
h	Measure equivalent width assuming Gaussian profile
k	Mark two continuum points and fit a single Gaussian line profile
m	Compute the mean, RMS, and S/N over marked region
t	Enter interactive curve fit function (usually used for continuum fitting)
Displaying and redrawing spectra	
a	Expand and autoscale data range between cursor positions
b	Set plot base level to zero
c	Clear all windowing and redraw full current spectrum
r	Redraw spectrum with current windowing
w	Window the graph
x	Etch-a-sketch mode; connects two cursor positions
y	Overplot standard star values from calibration file
z	Zoom graph by a factor of two in X direction
$	Switch between physical pixel coordinates and world coordinates
General file manipulation commands	
?	Display help
g	Get another spectrum
i	Write current spectrum to new or existing image
q	Quit and go on to next input spectrum

3.5.4 STSDAS fitting Package

The STSDAS **fitting** package contains several powerful and flexible tasks, listed in Table 3.7, for fitting and analyzing spectra. The **ngaussfit** and **nfit1d** tasks, in particular, are very good for interactively fitting multiple Gaussians and nonlinear functions, respectively, to spectral data. These tasks do not currently recognize the multispec WCS method of storing wavelength information. They recognize the simple sets of dispersion keywords such as W0, WPC and CRPIX, CRVAL, and CDELT, but these forms apply only to linear coordinate systems and therefore would require resampling of your data onto a linear wavelength scale before being used. However, these tasks do accept input from STSDAS tables, in which you can store the wavelength and flux data value pairs or wavelength, flux, error value triples (“imtab” on page 3-22).

Table 3.7: Tasks in the STSDAS fitting Package

Task	Purpose
convert	Convert ASCII data base format to STSDAS table format
function	Generate functions as images, tables, or lists
gfit1d	Interactive 1-d linear curve fit to images, tables, or lists
i2gaussfit	Iterative 2-d Gaussian fit to noisy images (script)
nfit1d	Interactive 1-d non-linear curve fit to images, tables, or lists
ngaussfit	Interactive 1-d multiple Gaussian fit to images, tables, or lists
n2gaussfit	2-d Gaussian fit to images
prfit	Print contents of fit tables created by fitting task

When using tasks such as **ngaussfit** and **nfit1d**, you must provide initial guesses for the function coefficients as input to the fitting algorithms. You can either specify these initial guesses via parameter settings in the task’s parameter sets (psets) or enter them interactively. For example, suppose you want to fit several features using the **ngaussfit** task. Using the default parameter settings you can start the task by typing:

```
fi> ngaussfit n4449.hhh linefits.tab
```

This command reads spectral data from the image `n4449.hhh` and stores the results of the line fits in the STSDAS table `linefits.tab`. After you start the task, your spectrum should appear in a plot window and the task will be left in cursor input mode. You can use the standard IRAF cursor mode commands to rewindow the plot, resticting your display to the region around a particular feature or features that you want to fit. You may then want to:

- Define a sample region (using the cursor mode S command) over which the fit will be computed so that the task will not try to fit the entire spectrum.

- Define an initial guess for the baseline coefficients by placing the cursor at two baseline locations (one on either side of the feature to be fitted) using the B keystroke.
- Use the R keystroke to redraw the screen and see the baseline that you've just defined.
- Set the initial guesses for the Gaussian centers and heights by placing the cursor at the peak of each feature and typing P.
- Press F to compute the fit once you've marked all the features you want to fit.

The results will automatically be displayed. You can use the `:show` command to see the coefficient values.

Note that when the **ngaussfit** task is used in this way (i.e., starting with all default values), the initial guess for the FWHM of the features will be set to a value of one. Furthermore, this coefficient and the coefficients defining the baseline are held fixed by default during the computation of the fit, unless you explicitly tell the task through cursor *colon* commands[3] to allow these coefficients to vary. It is sometimes best to leave these coefficients fixed during an initial fit, and then to allow them to vary during a second iteration. This rule of thumb also applies to the setting of the `errors` parameter which controls whether or not the task will estimate error values for the derived coefficients. Because the process of error estimation is very CPU-intensive, it is most efficient to leave the error estimation turned off until you've got a good fit, and then turn the error estimation on for one last iteration.

Figure 3.6 shows the results of fitting the Hβ (4861Å) and [OIII] (4959 and 5007 Å) emission features in the spectrum of NGC 4449. The resulting coefficients and error estimates (in parentheses) are shown in Figure 3.7.

3. See the online help for details and a complete listing of cursor mode colon commands: type `help cursor`.

Figure 3.6: Fitting Hβ and [OIII] Emission Features in NGC 4449

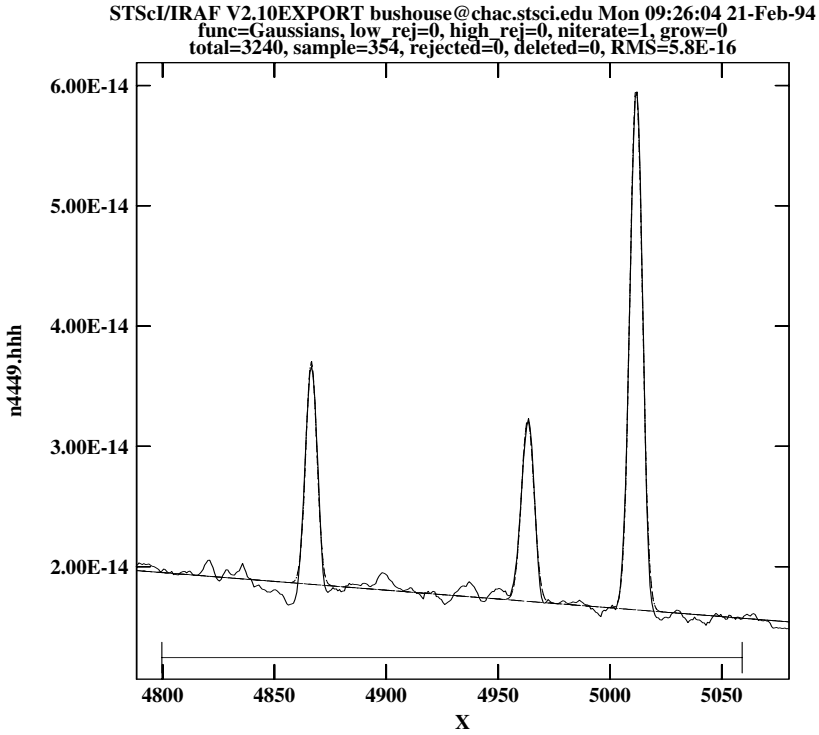


Figure 3.7: Coefficients and Error Estimates

```
function = Gaussians
coeff1 = 8.838438E-14    (0.)            - Baseline zeropoint (fix)
coeff2 = -1.435682E-17   (0.)            - Baseline slope (fix)
coeff3 = 1.854658E-14    (2.513048E-16)  - Feature 1: amplitude (var)
coeff4 = 4866.511        (0.03789007)    - Feature 1: center (var)
coeff5 = 5.725897        (0.0905327)     - Feature 1: FWHM (var)
coeff6 = 1.516265E-14    (2.740680E-16)  - Feature 2: amplitude (var)
coeff7 = 4963.262        (0.06048062)    - Feature 2: center (var)
coeff8 = 6.448922        (0.116878)      - Feature 2: FWHM (var)
coeff9 = 4.350271E-14    (2.903318E-16)  - Feature 3: amplitude (var)
coeff10 = 5011.731       (0.01856957)    - Feature 3: center (var)
coeff11 = 6.415922       (0.03769293)    - Feature 3: FWHM (var)
rms           = 5.837914E-16
grow          = 0.
naverage      = 1
low_reject    = 0.
high_reject   = 0.
niterate      = 1
sample        =  4800.132:5061.308
```

3.5.5 specfit

The **specfit** task, in the STSDAS **contrib** package, is another powerful interactive facility for fitting a wide variety of emission-line, absorption-line, and continuum models to a spectrum. This task was written by Gerard Kriss at Johns Hopkins University. Extensive online help is available to guide you through the task,[4] although because it is a contributed task, little to no support is provided by the STSDAS group.

The input spectrum to **specfit** can be either an IRAF image file or an ASCII file with a simple three-column (wavelength, flux, and error) format. If the input file is an IRAF image, the wavelength scale is set using values of W0 and WPC or CRVAL1 and CDELT1. Hence, for image input, the spectral data must be on a linear wavelength scale. In order to retain data on a non-linear wavelength scale, it is necessary to provide the input spectrum in an ASCII file, so that you can explicitly specify the wavelength values associated with each data value. The online help explains a few pieces of additional information that must be included as header lines in an input text file.

By selecting a combination of functional forms for various components, you can fit complex spectra with multiple continuum components, blended emission and absorption lines, absorption edges, and extinction. Available functional forms include linear, power-law, broken power-law, blackbody, and optically thin recombination continua, various forms of Gaussian emission and absorption lines, absorption-edge models, Lorentzian line profiles, damped absorption-line profiles, and mean galactic extinction.

3.6 References

3.6.1 Available from STScI

- *STSDAS Users Guide*, version 1.3.3, September 1994.
- *STSDAS Installation and Site Managers Guide*, version 2.0, August 1997.
- *Synphot Users Guide*, August 1997.
- *IGI Reference Manual*, version 3.0, November 1997.

3.6.2 Available from NOAO

- *A Beginners Guide to Using IRAF,* 1994, J. Barnes.
- *User Manual for SAOimage,* 1991, M. Van Hilst.

4. Additional information is available in the *Astronomical Data Analysis Software and Systems III*, ASP Conference Series, Vol. 61, page 437, 1994.

- *Photometry Using IRAF*, 1994, L. Wells.
- *A User's Guide to Stellar CCD Photometry with IRAF*, 1992, P. Massey and L. Davis.

3.6.3 Other References Cited in This Chapter

- Horne, K., 1988, in *New Directions in Spectrophotometry*, A.G.D. Philip, D.S. Hayes, and S.J. Adelman, eds., L. Davis Press, Schenectady NY, p. 145.
- Koorneef, J., R. Bohlin, R. Buser, K. Horne, and D. Turnshek, 1986, in *Highlights of Astronomy*, Vol. 7, J.-P. Swinds, ed., Reidel, Dordrecht, p. 833.
- Kriss, G., 1994, in *Astronomical Data Analysis Software and Systems III*, PASP Conference Series, Vol. 61, p. 437.