Introduction to Reducing HST Data
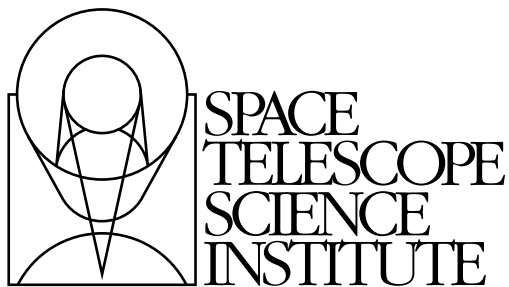
User Support

For prompt answers to any question, please contact the STScI Help Desk.

- **E-mail:** help@stsci.edu
- **Phone:** (410) 338-1082
  (800) 544-8125 (U.S., toll free)

World Wide Web

Information and other resources are available on the Web site:

- **URL:** http://www.stsci.edu.

Revision History

Version	Date	Editors
1.0	February 1994	Stefi Baum
2.0	December 1995	Claus Leitherer
3.0, Vol. I	October 1997	Mark Voit
3.0, Vol. II	October 1997	Tony Keyes
3.1	March 1998	Tony Keyes
4.0	January 2002	Bahram Mobasher, Chief Editor, HST Data Handbook Michael Corbin, Jin-chung Hsu, Editors, HST Introduction
5.0	July 2004	Diane Karakla and Jennifer Mack, Editors, HST Introduction. Susan Rose, Technical Editor, HST Introduction.

Contributors:

Faith Abney, Sarah Anderson, John Baum, Molly Brandt, Michael Corbin, Dorothy Fraquelli, Perry Greenfield, Al Holm, J. C. Hsu, Inge Kamp, Jessica Kim, Anton Koekemoer, Russ Makidon, Roeland van der Marel, Sid Parsons, Al Schultz, Mike Swam

Table of Contents

Preface

The HST Data Handbook is comprised of three separate sections which are merged together to form a single document:

- Part I is a general introduction which describes the process of retrieving and reducing Hubble Space Telescope (HST) data.

- Part II is an instrument-specific document which describes the reduction procedures, calibrations, and sources of error specific to each active HST instrument.

- Part III is a general set of appendices which includes an IRAF primer, a description of HST file names, and a summary of the observation log files.

Use of HST data necessarily involves using software to retrieve, analyze, and view it. With regard to analysis and visualization, there are many different tools and packages available. It would be impractical for a handbook to show how to do this for all available software. Since much of the software developed by STScI for calibrating, analyzing, and visualizing HST data has been based on the IRAF system, the focus of this handbook will be on use of IRAF, STSDAS, and TABLES for doing so.

Chapter 3 will briefly mention other software tools for accessing HST data and where to get more information about these tools. PyRAF is a new command language (CL) for IRAF and as such, allows use of almost all IRAF tools. It is briefly discussed in Chapter 3, but for the most part, the IRAF examples shown will work exactly the same for PyRAF.

The specifics of the data produced by each of these five active instruments (FGS, WFPC2, STIS, NICMOS, and ACS) are described in separate versions of Part II, while the general information from Parts I and III, referred to as 'the introductory chapters', are tacked onto the beginning and end of each. We recommend a careful reading of the introductory chapters before proceeding to the instrument-specific data handbooks and before starting to work on your HST data.

The present introductory chapters are based on information available as of June 2004. Many changes in the HST Data Archive and HST data reduction software have occurred since the last revision of the Hubble Data Handbook introductory chapters.

Future changes in this handbook are anticipated as the Multimission Archive at STScI (MAST) expands to cover additional missions, and as StarView, PyRAF, and STSDAS software continue to evolve. The reader is advised to consult the STScI web site at http://www.stsci.edu/hst for the latest information.

(Editors, HST Introduction)

Getting HST Data

In this chapter. . .

This chapter describes how to obtain Hubble Space Telescope (HST) data files. All HST data files are stored in the Hubble Data Archive (HDA), which forms part of the Multimission Archive at STScI (MAST)[1]. HST Guaranteed Time Observers (GTOs), Guest Observers (GOs) and Archival Researchers can retrieve data in either of two ways:

- Electronically over the Internet from the HDA, where data are stored immediately after they pass through HST pipeline processing.

- On data storage media written at STScI from the HDA. Data can be written to CDs or DVDs.

Non-proprietary data in the HDA can be retrieved electronically either by registered HDA users or via anonymous login. However most GO and GTO observations carry a proprietary period of up to one year after observation. Proprietary data may not be retrieved except by a registered HDA user who has the permission of the program's Principle Investigator (PI). Note that HST PIs are *not* automatically registered. PIs should register <u>before</u> their first observations have been taken. All calibration observations as well as observations made as part of the GO Parallel programs are

---

1. MAST currently includes data from HST, FUSE, GALEX, IUE, EUVE, ASTRO, HUT, UIT, WUPPE, ORFEUS, BEFS, IMAPS, TUES, Copernicus and ROSAT. Data from the FIRST radio survey, Digital Sky Survey (DSS) and Sloan Digital Sky Survey (SDSS) are also available.

immediately public. All observations made as part of the Treasury Programs begun in Cycle 11 will either be immediately public or have only a brief proprietary period. The High-Level Science Products section of MAST also contains several sets of fully reduced HST data, including the Hubble Deep Field, the Ultra Deep Field, and the GOODS Treasury program. These data are also public. As of September 2002, WFPC2 associations are also available through MAST. Read more about this in Section 1.3.1.

This chapter describes how to search the HDA, how to electronically retrieve files from it, and how to request and read disks containing HST data. As an aid to retrieving their data, PIs will automatically receive e-mail notification of the status of their observations two times: first, when the first datasets for their proposal are archived, and second, when all the datasets for their proposal and all necessary calibration files have been archived.

Note for Advanced Camera for Surveys (ACS) Users: Calibrated ACS images are approximately 168 MB in size, larger than those of any other HST instrument. Therefore, the preferred option for data retrieval is from the HDA staging disk via ftp/sftp. Users retrieving large numbers of ACS files should also consider requesting them on DVDs. In addition, the archive recommends to ask for compressed data, which distinctly shortens the retrieval times without any information loss.

1.1 Archive Overview

The HDA contains all HST observations ever made. It also contains a database that catalogs and describes these observations. The archive provides On-the-Fly Reprocessing (OTFR) of HST data for the following instruments: WFPC2, NICMOS, and STIS. The OTFR system reconstructs FITS data files from original telemetry and calibrates data at the time that a user's request for the data is processed. Through this system, users obtain data calibrated with up-to-date reference files, parameters, and software. There are currently two ways to search and retrieve data from the HDA. The first is a program called StarView, which acts as an interface to the HDA. StarView currently runs as Java-based, stand-alone application that can be downloaded from the web site http://starview.stsci.edu/html/. Previous versions of StarView, such as XStarView, are no longer available. The second search and retrieval method is through the HST section of the MAST web site, http://archive.stsci.edu. StarView is the more powerful of

the two methods, and in particular allows an examination of the calibration files applied to a given data file. StarView also provides an interface to the Visual Target Tuner (VTT) in the Astronomer's Proposal Tools (APT) suite of programs. The VTT interface can display archive observations on a Digital Sky Survey (DSS) image alongside planned observations. StarView is thus recommended for observation planning, duplication checking, calibration file review, investigation of On-The-Fly Reprocessing flags and proprietary status. It is also recommended for those needing to retrieve large numbers of datasets, and those needing to examine calibration files. The MAST web site interface to the HDA has the same basic capabilities as StarView, and may be preferable for those requiring simple retrievals of datasets. Both StarView and the MAST web site allow cross-qualified searches of the other MAST mission archives for all HDA searches. They also offer simple preview of HST datasets when available, as well as links to references citing a given dataset using the Astrophysics Data System (ADS). In later sections, StarView and the MAST web site are discussed in more detail.

1.1.1 Archive Registration

The simplest way to register and retrieve HST data is to complete the form on the Web page at: http://archive.stsci.edu/registration.html. If problems occur, registration requests may also be sent to the HDA hotseat, at: archive@stsci.edu.

The PI of each HST proposal must request access to their proprietary data for themselves, and for anyone else whom the PI wants to have access to it. PI retrieval permission is not granted automatically, for security reasons. PIs wishing to allow access to their proprietary data should make that request to archive@stsci.edu.

When registration is granted, your account will be activated automatically, and you will receive your username and password via e-mail.

1.1.2 Archive Documentation and Help

The MAST web site provides a wealth of useful information, including an online version of the HST Archive Manual available at http://archive.stsci.edu/hst/pub.html. Investigators expecting to work regularly with HST and other datasets supported by MAST should also subscribe to the MAST electronic newsletter by sending an e-mail to archive_news-request@stsci.edu and putting the single word *subscribe* in the body of the message. Questions about the HDA can be directed to archive@stsci.edu, or by phone to (410) 338-4547.

1.2 Getting Data with StarView

1.2.1 Downloading and Setting Up StarView

The latest version of StarView runs under versions 1.3 and later of Java and may be downloaded from http://starview.stsci.edu. Alternatively, a version of Starview is bundled with the Astronomer's Proposal Tools and that version enables more graphical interface between StarView and the VTT. This version may be downloaded with APT, see http://apt.stsci.edu for more information.

This StarView site also includes a FAQ page and news on releases and updates. StarView will automatically update itself to the latest version, so users do not have to worry about additional installations. Following its installation on computers running Unix and Linux, begin StarView by typing

```
> StarView
```

at the system prompt. Under Windows and Mac systems, StarView will appear as an icon. The StarView session then begins, first with an Information window explaining navigation within StarView, and a request for the user to specify an object name resolver (SIMBAD or NED) for use in HDA searches. First-time users are asked to supply their e-mail information in order to allow StarView to communicate the results of its attempts to retrieve the files requested from the HDA. This e-mail information includes the user's SMTP host, or the computer from which e-mail messages are routed. If unsure of your SMTP host, ask your system administrator. These queries can be turned off for future sessions once this information has been supplied.

1.2.2 Simple Use of StarView

We now proceed to an introduction to the use of StarView. A more detailed description of its capabilities is provided at the web site above, which should also be consulted for more advanced topics such as its Table Exportation and Cross-Qualification functions.

The basic function of StarView is to enable the user to first search the HDA (and the other mission archives in MAST) for data files matching criteria such as object name, position, or proposal number, then allow the user to navigate through the set of files matching those criteria, and finally to let the user select files for retrieval. Several options for the type of search that can be performed (e.g. by a particular instrument) will be discussed later.

The design of StarView is similar to that of a Web browser. At its top are pull-down menu bars including File, Searches, and Help. The Help menu offers links to documents including the StarView FAQ page. Beneath these menu bars is a row of buttons that run StarView's basic functions, such as searching, marking files for retrieval, and previewing images. A Help button allows users to display pop-up windows describing the function of the different StarView buttons and windows, by first clicking the Help button, then the item of interest. Beneath the row of buttons is the Qualifications panel, which is displayed when a search is begun. It consists of several cells corresponding to the search parameters the user wishes to use, e.g., object name, proposal ID, or instrument. Below this window will appear the Results panel, displaying the datasets found to match a given set of search parameters entered into the Qualifications panel. For the purpose of introduction, we will describe the use of the most basic search option, called "Quick Search," which can be started by clicking the "Quick" button at the top left of StarView.

As an example of the use of the Quick Search option, we will request all available WFPC2 data for the galaxy M87. This is done by typing "WFPC2" and "M87" in the Instrument and Target Name cells of the Qualifications section, then clicking the "Search" button at the top left of the StarView window. The results of the search will then be displayed in the bottom panel of StarView, as shown in Figure 1.1. These results include the dataset name, instrument name, R.A. and Dec. of the target, and the instrument aperture used. Note that these parameters could also have been specified in the Qualifications section, as can other parameters including proposal ID number, proposal PI name, and image central wavelength (corresponding to particular instrument filters or gratings).

An additional useful example would be to search by coordinates. Click "coords" button, select SIMBAD or NED, and click "resolve". This pushes coordinates into the corresponding qualifiers of the Quick Search panel.

Figure 1.1: Results of StarView Quick Search for WFPC2 files of M87

Clicking on a given dataset in the Results panel will display the information shown in the cells above it (Proposal ID, Release Date, PI, etc.). You may browse through the retrieved datasets either by using the mouse and scroll bar, or by using the navigation buttons (Scan, Previous, Next) in the top row of mouse buttons. The Scan option will automatically step through all of the files retrieved in the search, provided that the right most button at the bottom of the Results window is toggled to "Update." If this button is toggled to "No Update," the Scan option will go straight to the end of the list of files.

The ability to obtain a preview is available for many, but not all, of the datasets in the HDA (e.g., previews are not available for many FOC datasets). This is done with the "Preview" button, if it is enabled. For images, this will display a re-sampled version of the image using the Java Image Preview Application (JIPA) tool developed by ESO that is part of StarView. For spectra, simple GIF image of the calibrated spectrum will be displayed. JIPA and VTT can also display an image's FITS header. When viewing with JIPA, use the "Tools" menu. The JIPA preview of the WFPC2 image U2900103T retrieved in the previous search for WFPC2 images of M87 is shown in Figure 1.2, along with the window displaying part of the FITS header file of this image.

Other display options with StarView include "DSS," which will display a 20' x 20' Digital Sky Survey image at the target coordinates, while the "Overlay" button will display the same DSS image with outlines of the HST instrument apertures at the target coordinates superimposed on it, at the orientation of the selected observation. The "References" button provides a link to any known published papers citing the dataset, as listed in ADS. Note that the HST images displayed by Preview are of reduced quality compared to the actual data files, and cannot be downloaded. They are only meant to provide a quick check that the datasets found by the search met the search criteria, i.e., contained the object(s) of interest, and are of the desired quality.

Figure 1.2: JIPA preview of WFPC2 image U2900103T, along with image header file, using Preview option
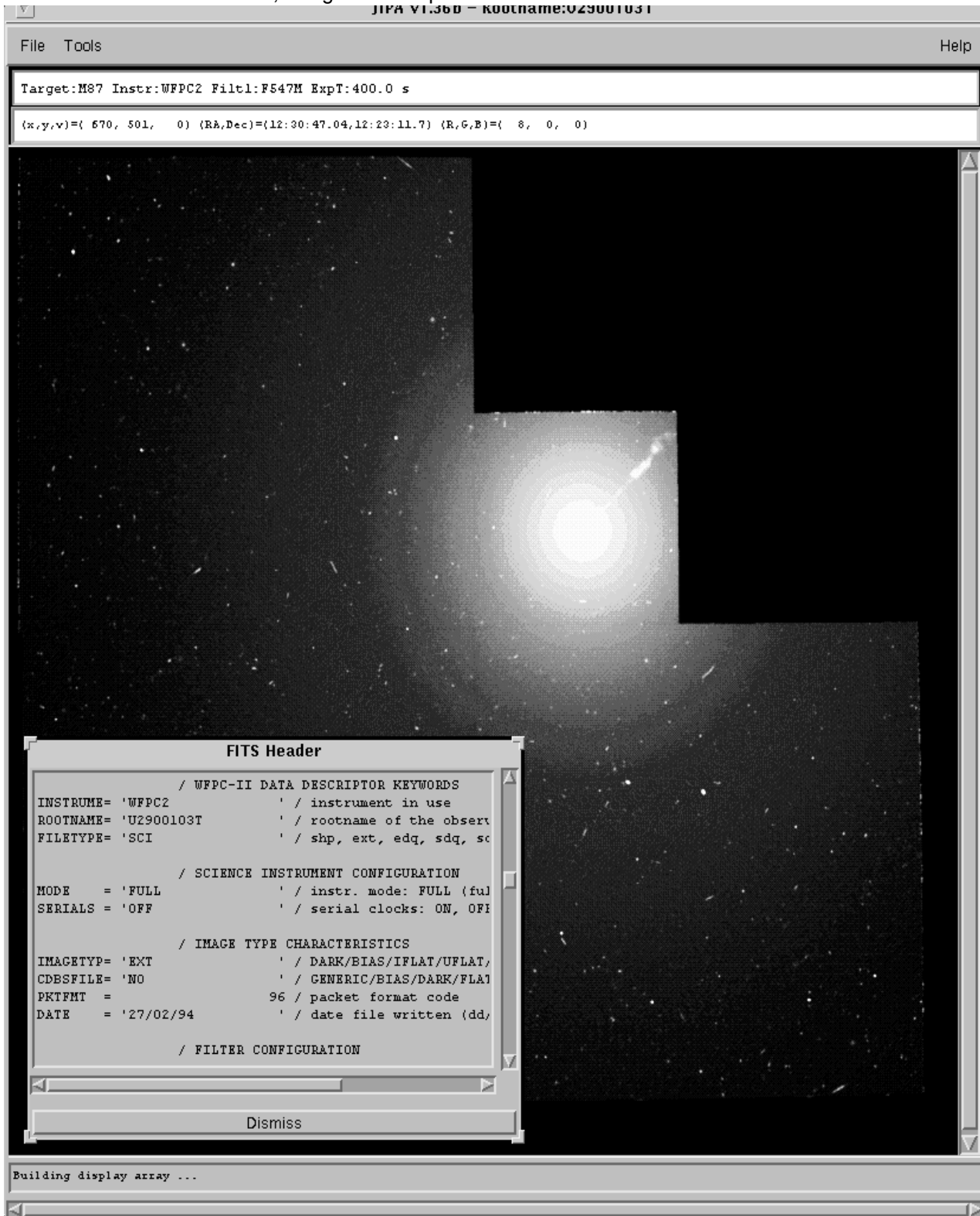
1.2.3 Marking and Retrieving Data with StarView

Datasets are marked for retrieval by first clicking on them, then using the "Mark" button at the top of StarView. An alternate method is to right-click on the datatset and choose "Mark Dataset(s)/fetch entry" from the menu (or simply Ctrl-M on the dataset). There is also the "All" button, which will mark all the datasets retrieved in the search. Marked datasets will be displayed in the Retrieval window. Datasets still within their proprietary period will be displayed in yellow. Only the proposal PI and those authorized by the PI will be able to retrieve them. The release date of files still within their proprietary period will also be indicated on the retrieval panel.

If satisfied with the marked datasets, choose "Submit" in the Retrieval window to retrieve them. You will then be queried for the type of data files associated with the dataset(s) to be retrieved, the method of delivery of these files, and your archive username and password.

Proprietary data retrieval requires an archive username and password. Public data retrieval can be requested using an archive username and password or by checking the "anonymous" box. For help with your username and password, contact archive@stsci.edu. The options for file type include files calibrated with the On-The-Fly-Recalibration (OTFR) pipeline for the WFPC2, NICMOS, STIS and ACS instruments. OTFR applies the best available calibration files (e.g., dark current and flat field images taken closest in time to the observations) to the uncalibrated data files. You may also request the uncalibrated (raw) files and calibration files separately, or even specify a three letter file extension (e.g. FLT). For some of the earlier instruments, e.g., WFPC and FOS, even though OTFR is not available you may request both the calibration files actually applied to the images, as well as those that should provide the best calibration of them, if recalibration is desired. You may also request Data Quality and Observation Log files.

Options for data delivery include ftp transfer by the user from the HDA staging disk, automatic transfer from the HDA via the ftp and sftp to a host and directory specified by user, and the mailing of CDs and DVDs. If ftp/sftp delivery is specified, you will be queried for the name of the computer and directory in which the files are to be placed, as well as your user name and password on that computer. These requests are encrypted, so there is no danger of your login information being stolen. Upon final submission of the request, you will receive an e-mail message acknowledging its receipt, and another message after all the requested files have been transferred. The status of the request, (i.e., how many files have been transferred and any errors that have occurred), can be checked on a Web page at the address given in the acknowledgment message.

1.2.4 Using StarView to Retrieve Calibration Files and Proposal Information

StarView allows several additional types of searches of the HDA besides the Quick Search option described above. These can be selected from the Searches menu bar at the top of the StarView screen. One such search option is by instrument. This is the option necessary for identifying calibration reference files. As an example, selecting the option "WFPC2 OTFR" under the Instrument and WFPC2 sub-menus of the Searches menu, and then entering "M87" under Target Name in the qualifications box, brings up the screen shown in Figure 1.3. This screen shows all the calibration images and files applied by OTFR to the first file in the set of WFPC2 images of M87, as well as whether the application of these files was performed or omitted in the calibration pipeline. This is the same set of images found by the Quick Search query described above, and the same information for the other datasets from this search can be found using the Previous, Next and Scan buttons. Once these calibration images have been identified, further information on them can be obtained. For example, taking the name of the flat field file found in the above search and entering it into the "WFPC2 Calibration Data" Searches option will retrieve information on and where this file was taken, and the date after which its use is recommended. This will help users decide if they would prefer to recalibrate their data using different files.

StarView can also be used to search for and view the abstracts of accepted HST proposals. Like the Preview capability of StarView, this provides additional information about a given dataset and whether it may be useful for your science goals. Viewing proposal abstracts is an option under the Searches menu, and an example is shown in Figure 1.4. The Qualifications window again offers several parameters by which this search can be constrained, including proposal ID number, HST cycle, PI name, and combinations thereof. In the example shown only the proposal ID number was used.

Finally, StarView can be used during the Phase I proposal process to see whether or not HST observations of a given object or object class have already been made, or else are scheduled for execution. Specifically, the Duplications option under the Searches menu allows users to check a database containing both HDA files and a list of queued observations in order to see if a given object has been or will be observed. Similarly, under Duplications the user may also query the database of proposal abstracts for a given object or object class, to check for archived or scheduled observations. It may be necessary to obtain the target name from NED or SIMBAD first, or to use wildcards in the target name to find all occurrences of a target.

1.2.5 Advanced Features of StarView

In addition to its basic search and retrieval functions, StarView allows users to cross-qualify ("push") results from separate searches of the HDA or external lists, and to export the results of searches to disk as ASCII files. These operations are performed with the "XQual" and "Export" functions, respectively.

As an example of cross-qualification, a user might want to take a list of targets observed in proposal 8721 and find out what WFPC2 observations were made of these targets. This could be accomplished with the Cross-Qualification search in several ways.

The most straightforward approach would be to first open 2 Quick Search panes. In one, do a Quick search, specifying the proposal ID, e.g. 8721. After searching, make sure that you scan through all the results using the Scan button at the top of StarView. You will find the number of results at the bottom right of the search panel. Next, click on the "XQual" button on the left side of StarView. This brings up a new Cross Qualification window, such as the one shown in Figure 1.5. On the left of this new window, select the search form just run and on the right the blank Quick search form. For each, select the RA and Dec fields by first marking them with the left mouse button and then importing them to "Selected Fields" with the arrow button. You can also simply double click on the respective fields. Make sure they are in the same order, then click the "X-Qualify" button in the middle. This will push the RA and Dec results from the first search into the qualifiers of the new search. On the new search form, add WFPC2 to the Instrument qualifier cell and execute the search. This will return all the datasets from observations using WFPC2 for the targets observed in proposal 8721. Be aware that when pushing multiple results from one search to another, very long lists may cause StarView to run out of memory. So, construct your queries to retrieve only the most relevant data.

Figure 1.3:   Results of StarView search for WFPC2 OTFR calibration files for M87

Figure 1.4: Results of the StarView search for the abstract of Proposal 8725

Figure 1.5: Example of Cross Correlation Feature, in which Target Name has been chosen as the common feature to search for in two Quick Search result lists.

StarView Cross Qualification

Query Results from

Results for: Quick Search ▼

Mapped to Query in

Results for: Quick Search [2] ▼

Available Fields | Selected Fields | View Fields as | Selected Fields | Available Fields

Available Fields

Dataset Name:
RA:
Dec :
Instrument:
Flag:
Apertures:
Central Wavelen
Proposal ID:
Release Date:
PI (last name):
Target Name:
Target Descriptic
Config:
Start Time:

Selected Fields

Target Name:

View Fields as

Labels ▼

X-Qualify

Selected Fields

Target Name:

Available Fields

Dataset Name:
RA:
Dec :
Instrument:
Flag:
Apertures:
Central Wavelen
Proposal ID:
Release Date:
PI (last name):
Target Name:
Target Descriptic
Config:
Start Time:

Cancel

1.2.6 StarView and the Visual Target Tuner

The Visual Target Tuner is part of the Astronomer's Proposal Tools (APT) package, which has been created to aid astronomers in planning their HST observations during the Phase I and Phase II proposal stages (see http://apt.stsci.edu). VTT is an image display tool which allows the user to display DSS images or local FITS images with proper World Coordinate System keywords in the headers. It offers more features than JIPA, which is the default StarView display tool. However, for a limited number of operating systems, the VTT can be used with StarView. VTT offers the particular advantage that it can overlay the instrument apertures of multiple observations on a single DSS image. Clicking on these apertures will also highlight the associated datasets in StarView.

Currently, to combine StarView and VTT requires downloading and installing APT from the above Web site. APT is only available for those operating systems with the Java Virtual Machine 1.4.1 (JVM 1.4.1). You can download the StarView/VTT package with JVM 1.4.1 included (a large download). Note, for Mac users, STScI provides a smaller APT/VTT package without JAVA. This option is only available for Mac users. To make StarView use VTT, you must change your Viewer options from JIPA to VTT. Go to the preferences sub-menu of Edit in StarView, and change JIPA to VTT in the Viewers section. If VTT is not listed here, you should reinstall the two programs. Following this change, the Preview, DSS and Overlay buttons of StarView should all bring up VTT.

Alternatively, once VTT has been installed, you can also bring up StarView from it. Enter into StarView mode by clicking on the StarView button in the lower left hand menu of VTT. Clicking on a DSS image will then spawn a Quick StarView screen with the R.A. and Dec of the position you clicked loaded into the search fields. You can enter other constraints into these fields as usual. Search results can be displayed on the VTT screen by selecting the results in StarView, and pressing the Overlay button.

1.2.7 Quick Proprietary Data Retrieval with StarView

The following steps summarize the basic process that users need to go through to retrieve their data with StarView. These steps follow registration as a MAST user, notification from STScI that the observations for a given proposal are complete, and providing StarView with your e-mail information. They are intended as a quick reference for this process.

1. Start StarView.

2. Click the "Quick" button.

3. Enter your PI name and/or proposal ID number in the appropriate cell.

4. Click the "Search" button.

5. Use the "Scan" button to step through the retrieved files, after toggling the right most button at the bottom of the Results window to "Update" to verify that all datasets have been retrieved.

6. Preview some or all of the datasets if desired, to verify data quality and target acquisition.

7. Click "All" to mark all datasets for retrieval, or "Mark" to mark individual datasets for retrieval.

8. Click "Submit" in the window that will be spawned by marking the files.

9. Enter your MAST username and password and specify the means of data delivery. StarView remembers your name and password from past searches so it does not have to be entered each time.

10. Click "Done", and your data are on their way. You will receive an e-mail message when your retrieval has been queued, and another when the transfer is complete.

1.3 Getting Data with the World Wide Web

HDA datasets can be searched for, previewed and retrieved via the World Wide Web in very much the same way as with StarView. As noted in Section 1.1, StarView offers more capabilities for this process, including cross-qualification, the use of VTT, and more information about instrument calibration files. However, Web retrievals may be preferable in some cases, particularly when information on calibration files is not needed, and the hypertext on the Results pages makes it easy to access all the information they contain. The starting point for Web-based searches of the HDA is the MAST web site at: http://archive.stsci.edu[2]

This web page is shown in Figure 1.6. A powerful feature of MAST is that all of its mission archives, including the HDA, can be searched simultaneously. This is done with the Quick Target Search option shown on the MAST home page. This search will return all datasets for all missions available for a given object or coordinates, according to the search constraints specified by the user (based on the wavelength region of interest), and will provide hypertext links to these datasets. If only HST datasets are desired, they can be accessed separately by clicking "HST" on

---

2. European archive users should generally use the ST-ECF Archive at http://archive.eso.org. Canadian users should request public archival data through the CADC web site at http://cadcwww.dao.nrc.ca. Proprietary data are only available through STScI.

the MAST home page. Searches of the HDA by object class can also be made with the VizieR Catalog Search tool at:

http://archive.stsci.edu/vizier.php

Figure 1.6: MAST Home Page

The HST section of MAST offers tutorials about the HDA as well as a FAQ page and HDA news. It also provides links to HST "Prepared" datasets such as the Ultra Deep Field and the Hubble Deep Field images. Clicking on the "Main Search Form" option of the HST section brings up the page shown in Figure 1.7. Here the user is queried for the same search parameters as requested by StarView, e.g., Object Name, Instrument and Proposal ID Once these are entered, clicking the Search button returns a page listing the datasets found, which can then be selectively marked for retrieval. The data type and retrieval options remain the same as those for StarView. Previews of GIF files of most datasets are also available by clicking on the image name. Datasets marked for retrieval can be retrieved via ftp, sftp, or can be placed on the Archive staging disk. The preferred format for ftp is compressed as gzip. Data placed on the staging disk (the preferred method) can be retrieved using ftp with the archive user name and password.

Figure 1.7: HST Archive Web search Form

1.3.1 WFPC2 Associations

Since November 2002, MAST contains WFPC2 associations, more than 15000 combined WFPC2 images stacked by CADC and ST-ECF (http://archive.stsci.edu/hst/wfpc2/about.html). These combined images are the products of the basic registration and averaging of related sets of WFPC2 images, referred to as *associations*, that is usually performed by archival researchers after the retrieval of individual images. The WFPC2 constituent images all have the same orientation on the sky and hence no rotation procedure is used to stack them. They form an important new set of high-quality astronomical data that should be useful for a wide range of investigations. In addition, they provide better previews of a given field than the individual WFPC2 images. The web page also provides a project description and the pipeline procedure applied.

Figure 1.8 shows the query form that can be used to search for WFPC2 associations in MAST. Entering a target like M87 and clicking on the search button will return you to a screen with the respective datasets in FITS format. The data can be downloaded directly from this page.

1.3.2 High Level Science Products

MAST also contains a number of High Level Science Products, which are accessible at http://archive.stsci.edu/prep_ds.html. High-Level Science Products (HLSP) are fully processed (reduced, co-added, cosmic-ray cleaned, etc.) images and spectra that are ready for scientific analysis. HLSP also include files such as object catalogs, spectral atlases, and README files describing a given set of data. The data originate from the Treasury, Archival Legacy and Large Programs (TALL) of cycle 11 and 12, but contain contributions from smaller HST programs and other MAST missions. Users who are interested in contributing to the HLSP, are referred to the Guidelines for Contributing High-Level Science Products to MAST (http://archive.stsci.edu/hlsp/hlsp_guidelines.html, please make sure you got the latest version). Furthermore, they are asked to contact the archive scientist involved as soon as they start working on the data.

Figure 1.8: HST Archive Web search FormWFPC2 Associations Screen
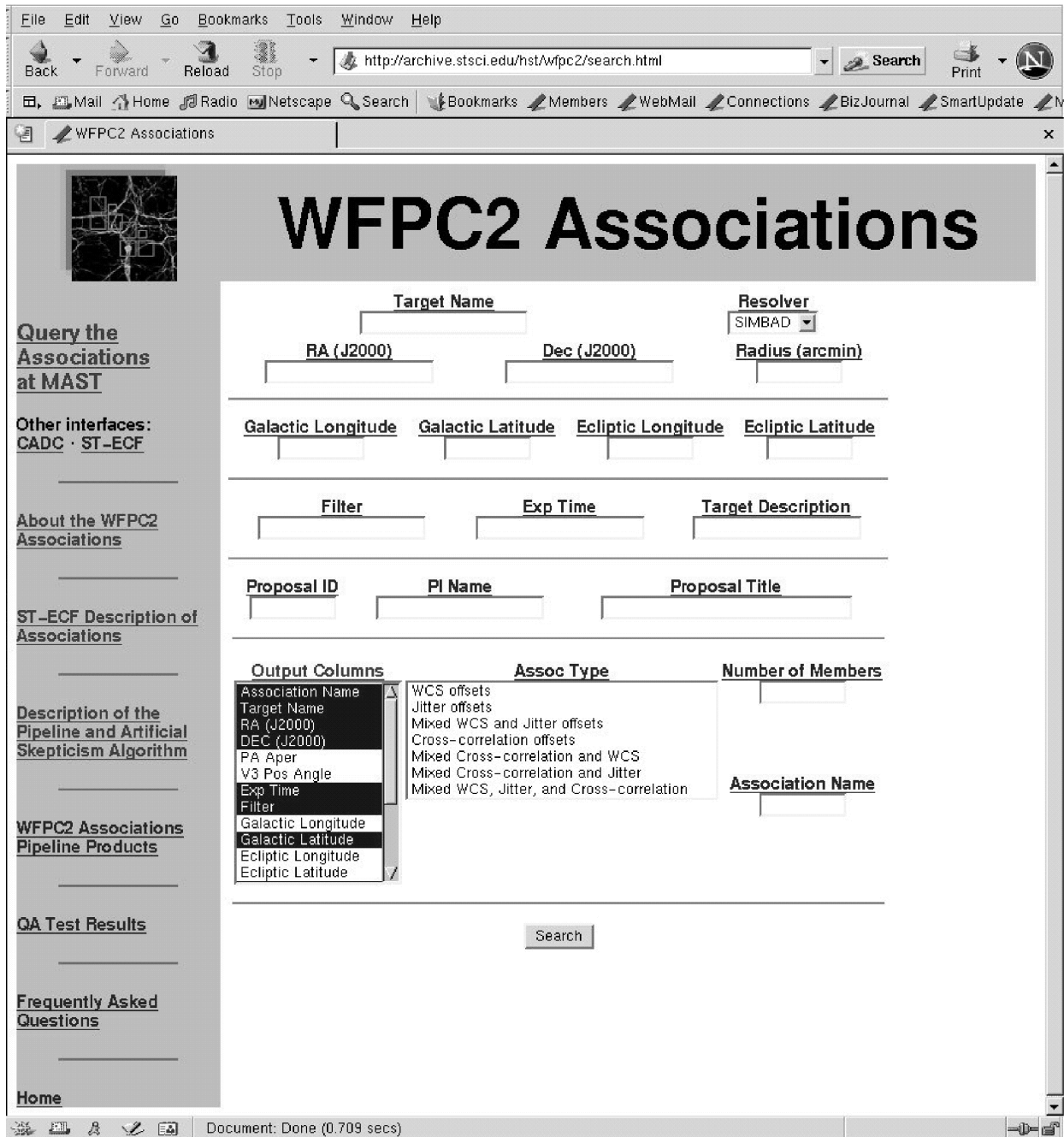
1.4 Reading HST Data Disks

If you request HDA files on CDs or DVDs, you will receive them within a few days of your request. The datasets will all be in FITS (Flexible Image Transport System) format[3]. On disk, the data is placed in subdirectories based on the rootname.

Currently, datasets obtained with HST's original instruments (FGS, FOC, FOS, GHRS, HSP and WFPC) as well as one of the current instruments WFPC2, must have their FITS files converted to GEIS (Generic Edited Information Set) format in order to work on them with IRAF/STSDAS. Further information on HST file formats is presented in Chapter 2. STSDAS is the package analysis software for HST data, and is discussed further in Chapter 3. Datasets obtained with the other current HST instruments (ACS, NICMOS and STIS) should be reduced in FITS format without conversion to GEIS. (STSDAS support for the analysis of WFPC2 data in FITS format is currently planned.)

The steps for reading and converting FITS files to GEIS files are as follows:

First bring up IRAF/STSDAS in your IRAF home directory by typing

```
> cl
```

If this does not start up an IRAF session, you will need to type "mkiraf" first and select an appropriate terminal window. See Appendix A.3

This will start an IRAF session. IRAF and STSDAS are organized into *packages*. To load a package, type its name. To begin with, you must load the **stsdas** and **fitsio** (FITS Input/Output) packages:

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

The IRAF prompt (such as st>) shows the first two letters of the most recently loaded package. The **fitsio** package contains the STSDAS programs (called *tasks* in the IRAF/STSDAS environment) required to read and write FITS files to and from tapes and disks. The two principle tasks are **strfits** for reading files, and **stwfits** for writing them.

Next, set the IRAF environment variable *imtype* to specify that your data files are to be written in GEIS format. This is done by typing

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

---

3. A description of FITS format and various supporting documents can be found at the Web site http://fits.gsfc.nasa.gov/fits_home.html

You should then move to the directory containing the FITS files.

The last step is to use **strfits** to read the data. Like most IRAF/STSDAS tasks, **strfits** has several parameters that control its function. You can either edit these tasks using the IRAF "epar" command, or specify them on the command line. For the purpose converting FITS files to GEIS files, the important parameter is *oldirafname*, which needs to be set to "yes" in order to keep the file rootname the same. To convert all the FITS files in a directory to GEIS files, type

```
fi> strfits *.fits "  " oldirafname=yes
```

This command will make GEIS format copies (having extension ".hhh") of all the FITS files in the directory with the same rootname. Following reduction and analysis of the GEIS files with the IRAF/STSDAS tasks, they may be written back into FITS format, on hard disk or to a tape or other storage media, with the **stwfits** task.

HST File Formats

In this chapter. . .

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. For the older instruments (FOC, FOS, FGS, GHRS, HSP, WF/PC-1, and WFPC2) pipeline processing assembles data received from HST into *datasets*, calibrates the data according to standard procedures described in the instrument sections (Part II) of this handbook, and stores both calibrated and uncalibrated datasets in the Archive. For newer instruments, calibrated data is no longer stored in the archive. The OTFR System generates calibrated data as it is requested from the archive.

Pipelines of the older instruments generate files in GEIS (which stands for Generic Edited Information Set) format. Since GEIS is a machine-dependent format, these files are converted to a specific kind of FITS file format, sometimes referred to as "waiver" FITS, before being archived. We'll explain the structure of this "waiver" FITS format later in this chapter. Since the "waiver" FITS format is only designed for archival purpose, it is necessary to convert it back to the GEIS format before further data processing and analysis using IRAF/STSDAS tasks.

Instruments installed after the 1997 servicing mission (STIS, NICMOS, and ACS) have pipelines which generate FITS files directly. They are ready to be used by relevant IRAF/STSDAS tasks and, unlike the "waiver" FITS files, do NOT need to (and indeed, should not) be converted to GEIS format. Sometimes FITS files for the newer instruments are referred to as

"FITS with extension" or "extended" FITS files. But this can be misleading, since a "waiver" FITS file also has one (ASCII table) extension.

Much confusion has occurred about the two kinds of FITS files archived at STScI. So we would like to repeat this warning one more time:

Older instruments (FOC, FOS, FGS, GHRS, HSP, WF/PC-1, and WFPC2) generate files in GEIS formats, but are stored and delivered as "waiver" FITS format in the archive, and need to be converted back to GEIS format before processing. Newer instruments (STIS, NIC-MOS, ACS) generate and store files in FITS format and should not be converted to GEIS.

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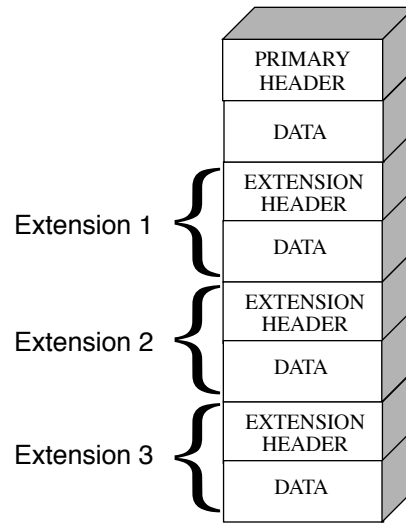
The three-letter identifier (e.g., d0h) that follows the rootname of a GEIS format HST data file (see Appendix B for more on HST file names) has often been called an "extension" in the past. However, because of the potential for confusion with FITS extensions, this handbook will refer to these three-letter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

Each FITS extension header has the required keyword XTENSION which specifies the extension type and has one of the following values: IMAGE, BINTABLE, and TABLE, corresponding to an image, binary table, and ASCII table, respectively.

A set of FITS extension images which are logically related to one another is called an *imset*. For example, the error image and the data quality image are in the same imset as the science image itself. The keyword EXTNAME is used to specify the extension names of different images in the same imset.

For the newer HST instruments (ACS, STIS, and NICMOS), EXTNAME can have the following values: SCI, ERR, DQ, corresponding to science, error, and data quality images. NICMOS files have two additional extension names: SAMP and TIME, for number of samples and exposure time, respectively. Thus, ACS and STIS imsets contain three images while NICMOS has five. Images of the same imset will have the same integer value in their headers' keyword EXTVER.

Here, the word "image" is used in a general sense. Often, an image has the same value at all pixels (e.g. data quality value). In this case, the extension has no data part and the constant pixel value is stored in the header keyword PIXVALUE. Except for SCI, any of the extensions may use this convention. Typically, SAMP and TIME extensions are single-valued and follow this rule.

2.1.1 Working with FITS Image Extensions

The FITS image kernel included in IRAF version 2.12.2 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 2.12.2 or higher. It covers how to:

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

Retaining the .fits suffix 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 NICMOS, STIS, or ACS data, you will need to upgrade to IRAF 2.12.2 or higher and STSDAS 3.2 or higher.

Generating a FITS File Listing

Once you have downloaded STIS, ACS, 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, in Table 2.1, illustrates the first 11 lines generated by **catfits** from a NICMOS multiaccum FITS file containing only images.

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 whose value is specified in the keyword XTENSION. The third column lists the extension name, given by the keyword EXTNAME.

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 Section 3.4.3), but most operate on individual images. See the Data Structure chapters of STIS, ACS, and NICMOS Data Handbooks for more information on the contents of a particular instruments' imsets.

Table 2.1: 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
```

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 or data block in the primary HDU (e.g. for WFPC2 or for the FGS).

Specifying the extension number is the most basic method of access, but it is not necessarily the most useful. 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

science image belonging to the imset with EXTVER = 2, as in the **catfits** listing above, it can be specified in two equivalent ways:

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

Designating an EXTNAME without an EXTVER refers to the first extension in the file with the specified value of EXTNAME. Thus, `fitsfile.fits[sci]` is the same as `fitsfile.fits[sci,1]`.

The syntax for designating image sections follows 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, ACS, and NICMOS data files use 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 include keyword inheritance, and while the idea itself is simple, its consequences are often complex and sometimes surprising to users.

In general, 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 both 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 in the second command, the `noinherit` specification is bracketed with the EXTNAME and EXTVER keywords and not in a separate bracket of its own, as in the first command where an absolute extension number is used. For a complete explanation of FITS file name specifications, see:

http://iraf.noao.edu/iraf/web/docs/fitsuserguide.html.

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, ACS, or NICMOS data:

If the specified 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]
```

In the previous example, and others that follow, note that the backslash is added to indicate that the remainder of the command follows on the next line, after the ">>>" prompt.

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]
```

2.1.2 Working with FITS Table Extensions

NICMOS, STIS, and ACS use FITS tables in two basic ways. Each instrument produces association tables (see Appendix B.3) 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. 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 3.2 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 Section 2.1.1), 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 specified. 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 output 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 out.fits[3][asn,2,overwrite]
```

This command would copy the table in the first extension of `n3tc01010_asn.fits` into the third extension of `out.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 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 that cell's physical attribute and spectral order.

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 (type `help system.match` for a description of 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. All STSDAS and TABLES tasks are now able to use row and column selection. For a complete explanation of the table selector syntax, type `help selectors`.

2.2 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

---

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

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 2.2 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 2.2: GEIS File Structure
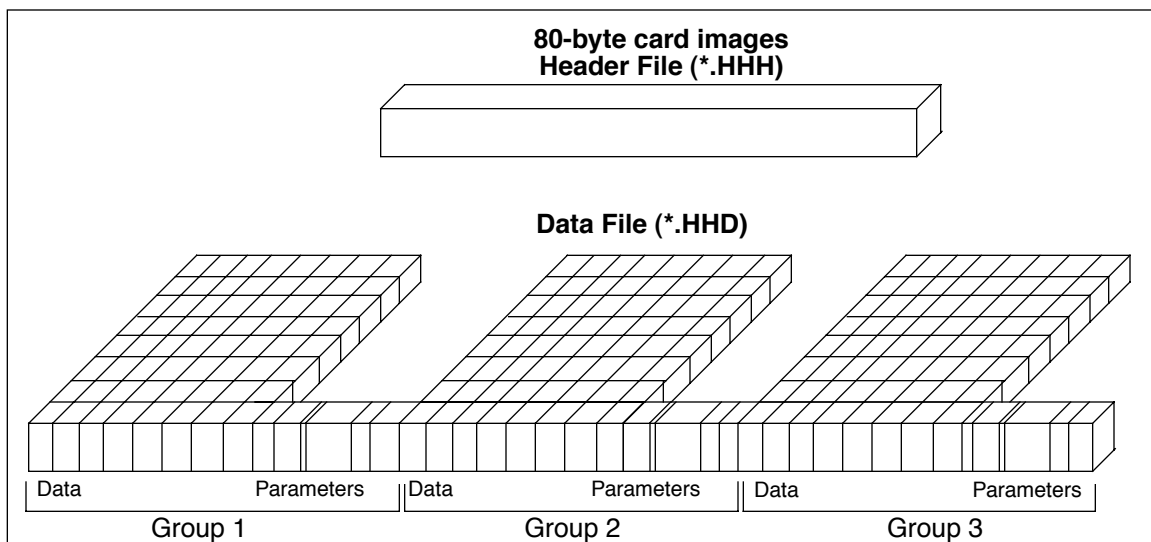
2.2.1 Converting FITS to GEIS

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 "" xdim=yes oldiraf=yes
```

In particular, for WFPC2 dataset `u6n20101m_clf.fits`, **strfits** produces `u6n20101m.clh` and `.cld`.

2.2.2 GEIS Data Groups

One of the original advantages of GEIS format 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 2.2 lists the *contents* and the number of groups in the final calibrated image for the most commonly-used modes of each instrument which uses the GEIS data format.

Table 2.2: Groups in Calibrated Images, by Instrument and Mode

Instrument	Mode	Number of Groups	Description
FGS	All	7	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 not all chips were used, see the DETECTOR keyword.

2.2.3 Working with GEIS Files

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

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 in "Printing Header Information" (this section, below).

You can use the IRAF **hedit** task to edit the keywords in GEIS headers. While it is 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's data handbook, 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 of the HST Introduction). 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 (always refer to a GEIS file by its header file name, i.e. *.??h, even though mathematically you are operating on the data portion):

```
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.

An image section of one group of a GEIS image may be overwritten or operated upon, leaving the rest of the image intact. For example, the following 2 lines will first create `outdata.hhh` and then overwrite a section of it:

```
cl> imarith indata.hhh * 0.0 outdata.hhh
```

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

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 an 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 group parameters for group 10 from the `*.hhd` file.

Other Group-Related Tasks

Currently, IRAF and 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 Chapter 3 and the *STSDAS User's Guide* for more details about working with GEIS images.

2.2.4 The "waiver" FITS Format

Although "waiver" is not an accurate word for the intended purpose, for historical reasons it has stuck and will be reluctantly adopted. However, in the past, a grammatically incorrect word "waivered" was used.

The "waiver" FITS format was developed when the HST archive needed a format to store and distribute the data products in a machine-independent medium for the community, at a time before FITS image extension was standardized. As a result, the "waiver" FITS format was adopted as a compromise.

Since, at the time, FITS could only have a single image while the HST data (in GEIS format) could have several images as multiple groups in one file, the idea of stacking the images of different groups together as a new dimension in the FITS image was conceived. As for group parameters, they are put in an ASCII table and the table becomes the first (and only) extension of the FITS file.

For example, the WFPC2 pipeline generates the science data as a GEIS file of 4 groups, each is an 800x800 image corresponding to one of the 4 detectors. When this GEIS file is converted to the "waiver" FITS file, the FITS file has an image of 800x800x4 (a three-dimensional image!) at its primary HDU. Similarly, an FOS GEIS file may have 40 groups, each group is a 1-D image (spectrum) of the size 2064. The waiver FITS file then will have one 2-D image of the size 2064x40, at its primary HDU. In the case of WFPC2, the first extension of the waiver FITS file will be an ASCII table containing 4 rows; each row corresponds to a group. The value of each group parameter is under a column named after the group parameter, e.g. the value of the group parameter CRVAL1 of the 2nd group will be at the 2nd row, under the column named "CRVAL1". In other words, the ASCII table has as many rows as there are groups in the original GEIS file, and as many columns as group parameters.

Although, *in theory, certain* IRAF/STSDAS tasks can directly access the data in the "waiver" FITS file, e.g. to display the 2nd "group" of a WFPC2 image:

```
st> display u67m0206r_c0f.fits[0][*,*,2]
```

will work, *most tasks, especially those specific to HST instruments, will not.*

▽ ***It is therefore STRONGLY recommended that all waiver FITS files be converted back to the GEIS format, by using the task strfits, before further processing and analysis with IRAF/STSDAS tasks.***

Analyzing HST Data

In this chapter. . .

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 the data handbooks.

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, 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 Alternative Means of Accessing HST Data

There is much software available to access HST data. This section will mention a few of the more popular alternatives to IRAF.

3.1.1 Interactive Data Language (IDL)

IDL is very popular in the astronomical community. In short it is an array-based, interactive programming language that is strong in numerical analysis and visualization tools. It is typically much easier to develop new analysis and visualization applications and utilities in IDL than in Fortran or C. As a result many astronomers use it for doing analysis of HST data. It is moderately expensive, however, and can be obtained from Research Systems Inc. (www.rsinc.com/idl/). Libraries for reading HST data are part of the freely available ASTRON library (idlastro.gsfc.nasa.gov) which also has links to other astronomy IDL libraries.

3.1.2 Python

Python is rapidly becoming a popular language for astronomical data reduction applications. It is a freely available, general-purpose, dynamically-typed interactive language that provides modules for scientific programming. These include:

- numarray: provides IDL-style array manipulation facilities
- PyFITS: ability to read and write FITS files to and from arrays
- matplotlib: plotting and image display package
- numdisplay: package to display arrays to saoimage, ds9 and ximtool
- PyRAF: package to run IRAF tasks from Python

Python is a very powerful language that is well suited to writing programs to solve many needs beside scientific analysis. It is generally much more productive than Fortran, C, or Java. Tools are available to read (but currently not write) GEIS files. PyRAF allows easy use of IRAF tasks with code written in Python.

STScI is developing most of its new calibration and data analysis software in Python. More information on the use of Python to analyze HST data can be obtained from:

http://www.stsci.edu/resources/software_hardware/pyraf

http://www.stsci.edu/resources/software_hardware/numarray

http://www.stsci.edu/resources/software_hardware/pyfits

3.1.3 Fortran and C

For those that wish to write their own Fortran or C applications, we recommend use of the FITSIO library for reading FITS files (http://heasarc.gsfc.nasa.gov/docs/software/fitsio/fitsio.html; note that the C library is called CFITSIO).

This library does not support the GEIS format directly so users will need to use the waiver FITS form obtained from the archive and manually extract the needed information.

3.1.4 Java

The most widely used FITS libraries for Java are the Java FITS Utilities (http://heasarc.gsfc.nasa.gov/docs/heasarc/fits/java/v0.9/) and the Java FITS Class Library (http://www.eso.org/~pgrosbol/fits_java/jfits.html). Like FITSIO, neither can read GEIS files, however; one must read the waiver FITS file of these data sets.

3.1.5 PyRAF

PyRAF is a new command language for IRAF that is based on Python. It has a number of advantages over the IRAF CL. Most importantly, with few exceptions, it allows use of exactly the same syntax that the IRAF CL accepts. Some of the advantages that it provides are:

- true command line recall (with arrow key editing)
- command and filename completion
- GUI-based graphics windows, previous plot recall, multiple graphics windows
- GUI epar editor with display of help in separate window
- IDL-like capabilities
- true error handling for scripts (shows which line the script fails at when errors occur)
- can script IRAF tasks in Python language
- exception handling capability

Since PyRAF is so highly compatible with the IRAF CL, virtually all of the examples shown in this handbook will work the same for PyRAF. Minor differences include the user prompt and the graphics windows appearance.

More information on PyRAF can be found at:

http://www.stsci.edu/resources/software_hardware/pyraf

3.2 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:

- `apropos` *word* - searches the online help database for tasks relating to the specified word (see Figure A.4).

- `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.

3.2.1 STSDAS Structure

STSDAS is structured so that related tasks are grouped together as packages. For example, tasks used in the calibration process can be found in the **hst_calib** package, and tasks used for image display and plotting can be found in the **graphics** package. Table 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 or higher.

3.2.2 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**), and **gcombine** for coadding GEIS images.

Table 3.1: STSDAS Version 3.2 Package Structure

analysis	**Data analysis package.**
dither	Dithered image combination.
fitting	Curve fitting tools.
fourier	Fourier analysis.
gasp	Access the HST Guide Star Catalog on CD-ROM.
isophote	Elliptical isophote image analysis.
nebular	Tasks for analyzing nebular emission lines
restore	Deconvolve or filter 1- or 2-dimensional images.
statistics	Statistical analysis software.
contrib	**User-contributed software.**
redshift	Tasks for determining redshifts and dispersions.
spfitpkg	Fitting spectra with non-linear chi-square minimization.
vla	Spectral image reduction for VLA data.
fitsio	**FITS input/output for Space Telescope data (images and tables).**
graphics	**Graphics and image display package.**
sdisplay	Image display package for SAOImage display device.
stplot	General plotting utilities.
hst_calib	**HST Science Instrument calibration package.**
acs	Tasks for calibrating ACS data.
ctools	General calibration tools.
foc	Tasks for calibrating FOC data.
focprism	FOC prism package.
fos	Tasks for calibrating FOS data.
spec_polar	Tasks for reducing and analyzing FOS polarimetry.
hrs	Tasks for calibrating HRS data.
nicmos	Tasks for calibrating NICMOS data.
paperprod	Tasks for generating paper products.
stis	Tasks for calibrating STIS data.
synphot	Synthetic photometry and modelling instrument response.
simulators	Synthetic photometry simulation package.
wfpc	Tasks for calibrating WFPC and WFPC-2 data.
w_calib	Tasks for deriving the WF/PC instrument calibration.
playpen	**Miscellaneous experimental tasks.**
sobsolete	**Package of tasks that have been retired.**
foccs	FOC calibration software package.
focgeom	FOC geometry package.
focphot	FOC photometry package.
focutility	Obsolete FOC utility package.
hsp	Tasks for calibrating HSP data.

olddither	Older version (1.2) of dither.
registration	Compute registration parameters and resample unaligned data files.
testdata	Tools for creating artificial images.
timeseries	Time series photometry data reduction and analysis.
y_calib	Tasks supporting the FOS calibration process.
z_calib	Tasks supporting the HRS calibration process.
toolbox	**General tools package.**
convfile	Reformat images between VAX and Sun.
headers	Tools for modifying image headers.
imgtools	Tools for manipulating & examining images and bad pixel lists.
mstools	Tasks to handle STIS/NICMOS IMSETs.
tools	Generic data handling and utility tools.
ttools	Table manipulation tools.

- **stsdas.toolbox.imgtools.mstools**: This package contains tools for working with FITS image extensions, in particular NICMOS, STIS, and ACS image sets (imsets). **Msstatistics**, for example, will print statistics on these images and **msarith** can do image arithmetic.

- **stsdas.analysis**: This package contains general tasks for image analysis, such as Fourier analysis (**fourier**) and dithering (**dither**).

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 supported, but for input only.) The *STSDAS User's Guide* describes the STSDAS table format in 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.

3.3 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. The FOC photon-counting hardware does not detect cosmic rays as easily as CCDs; the NICMOS pipeline automatically removes cosmic rays from MULTIACCUM observations; and the STIS and ACS pipelines automatically remove cosmic rays from CR-SPLIT association products.

3.3.1 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 a Unix window:

```
> saoimage &
```

2. Set the size or the window to 1024x1024 by typing:

```
cl> set stdimage = imt1024
```

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

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

Several different display servers, including SAOimage, ds9 (the next generation of SAOimage), and Ximtool, can be used with IRAF. ds9 may be retrieved from http://hea-www.harvard.edu/RD/ds9/. Ximtool may be retrieved from ftp://iraf.noao.edu/iraf/x11iraf/.

4. 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 3.1 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 3.1: Displaying an Image
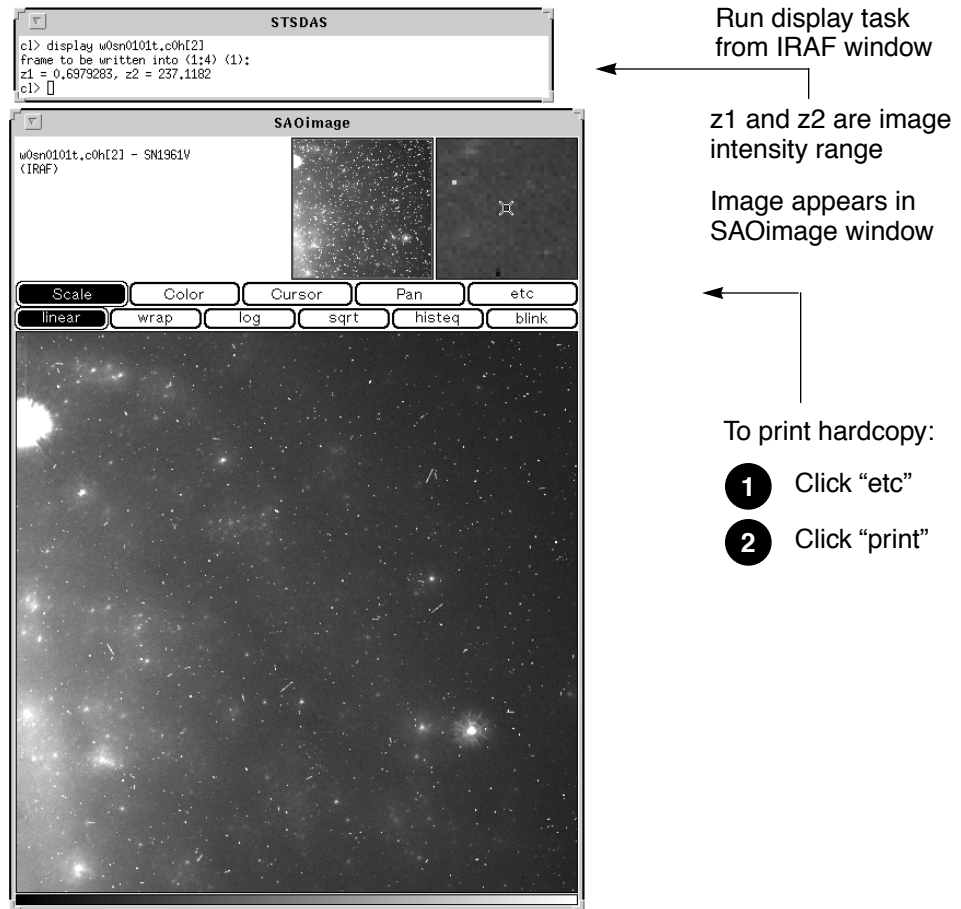
Modifying the Display

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 3.1 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 3.1 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 greater 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`.

Otherwise, the user can simply do:

```
im> display  x2ce0502t.c1h ztrans=log
```

The image display buffer can also be adjusted in IRAF by setting the `stdimage` parameter. For example,

```
im> set stdimage = imt 2048
```

will allow a larger image to be displayed without losing the borders.

---

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

3.3.2 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:

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

To specify the same pixel range in the second SCI extension of a NICMOS FITS image:

```
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 FITS image extensions, you must specify the extension, which also comes before the image section.

Figure 3.2 shows examples of displaying an image and an image section.

3.4 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 NICMOS, STIS, and ACS imsets.
- Converting counts to fluxes.

Figure 3.2: Displaying Sections and Groups of an Image
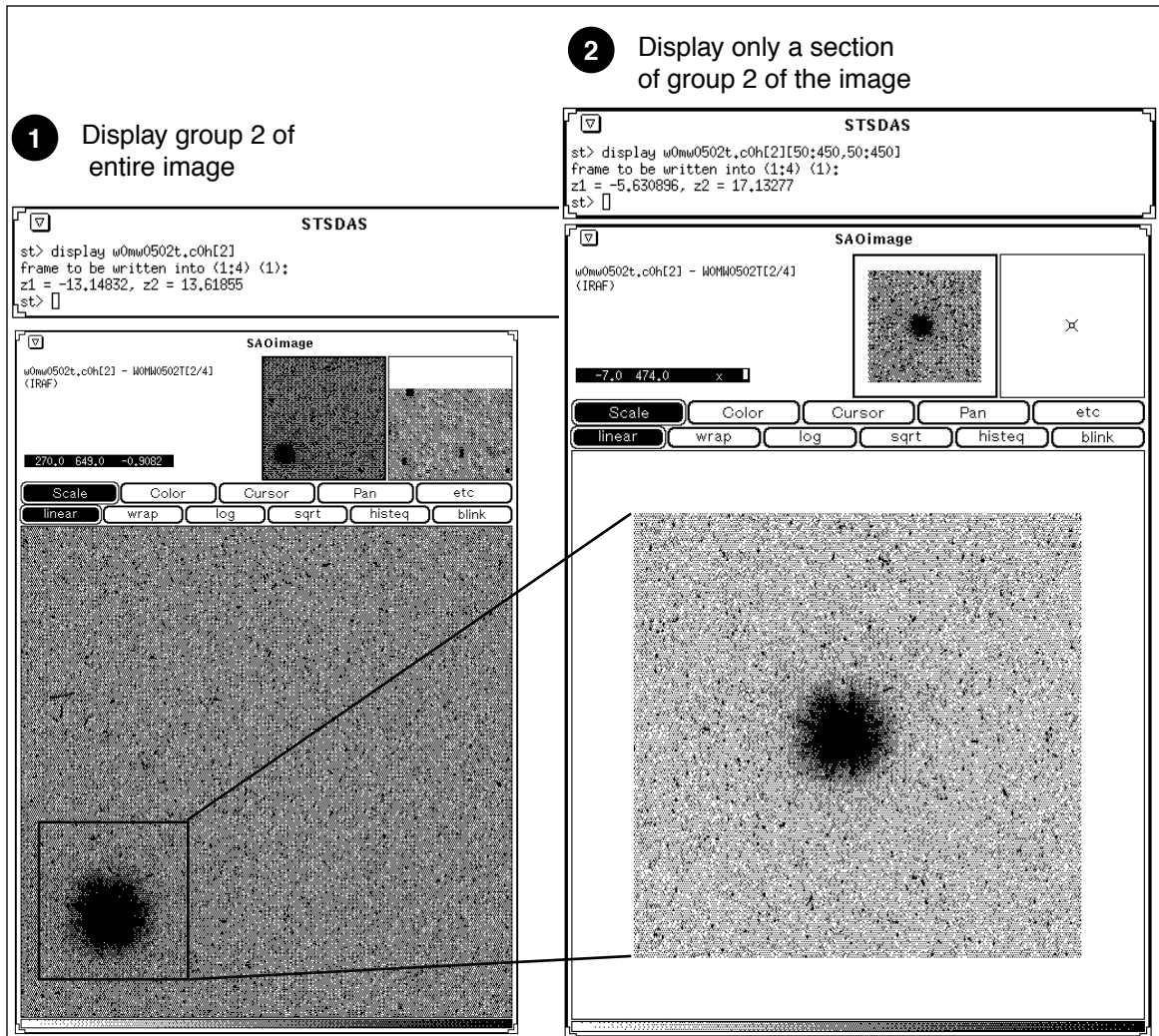
3.4.1 Basic Astrometry

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: reference pixel values (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] hms+ x y
```

Note, the `hms` option formats the results in hours, min, and seconds.

Observers should be aware that these tasks do not correct for geometric distortion. Only FOC, STIS, and ACS images currently undergo geometric correction during standard pipeline processing. (For WFPC2, a pixel area map correction image (f1k1552bu.r9h) may be obtained from the archive.) If precise relative astrometry is needed, an instrument-specific task that accounts for image distortion, such as the **metric** task for WF/PC-1 and WFPC2 images, should be used.

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

Table 3.2 lists some additional tasks that draw on the standard astrometry keywords.

Table 3.2: 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[1] to convert between IRAF coordinate systems.
wcslab	Produce sky projection grids for images.

1. World Coordinate System (WCS). Type `help specwcs` at the IRAF prompt for details.

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. Absolute astrometry, on the other hand, 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 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 provide an absolute positional accuracy of approximately 0.7 arcsec. Contact the Help Desk for assistance (send E-mail to `help@stsci.edu`).

3.4.2 Examining and Manipulating Image Data

This section describes **implot** and **imexamine**, two basic IRAF tools for studying the characteristics of an image, and Table 3.4 lists some useful IRAF/STSDAS tasks for manipulating images.

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 3.3. Figure 3.3 shows an example of how to use the **implot** task.

Table 3.3: Basic implot Commands

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

Figure 3.3: Plotting Image Data with implot

Plot line 200 of a WF/PC-1 image

To Print This Plot:

1 Press ⬜ =

2 Type **:.gflush** to flush the buffer
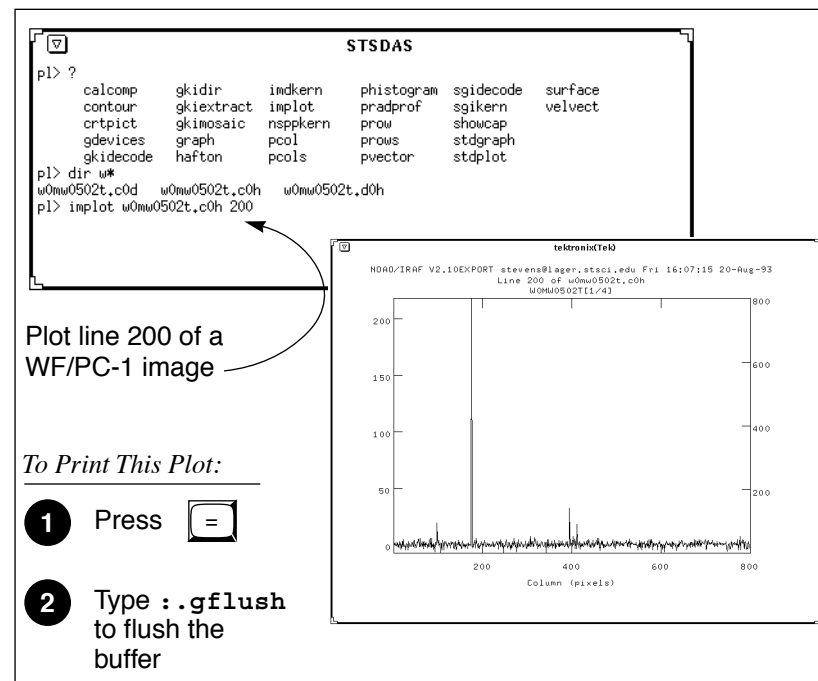
imexamine

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`.

3.4.3 Working with STIS, ACS, and NICMOS Imsets

NICMOS, STIS, and ACS data files contain groups of images, called *imsets*, associated with each individual exposure. A STIS or ACS imset comprises SCI, ERR, and DQ images, which hold science, error, and data quality information. A NICMOS imset, in addition to 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 NICMOS, STIS, and ACS Data Structures chapters for more details on imsets.

Table 3.4: Image Manipulation Tasks

Task	Package	Purpose
boxcar	images.imfilter	Boxcar smooth a list of images
blkavg	images.imgeom	Block average or sum an image
gcombine	stsdas.toolbox.imgtools	Combine GEIS images using various algorithms and rejection schemes[1]
gcopy	stsdas.toolbox.imgtools	Copy multigroup GEIS images[1]
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)[1]
gstatistics	stsdas.toolbox.imgtools	Compute image statistics[1]
imcalc	stsdas.toolbox.imgtools	Perform general arithmetic on images
imedit	images.tv	Fill in regions of an image by background estimation or copy and paste
imexamine	images.tv	Examine images using display, plots, and text (see page 3-15)
implot	plot	Plot lines and columns of images (see page 3-14)
magnify	images.imgeom	Magnify an image
msarith	stsdas.toolbox.mstools	Performs basic arithmetic on NICMOS and STIS imsets[2]
mscombine	stsdas.toolbox.mstools	Extension of **gcombine** for NICMOS, STIS, and ACS imsets[1]
msstatistics	stsdas.toolbox.mstools	Extension of **gstatistics** for NICMOS and STIS imsets[1, 2]
newcont	stsdas.graphics.stplot	Draw contours of two-dimensional data
pixcoord	stsdas.hst_calib.wfpc	Compute pixel coordinates of stars in a GEIS image[1]
plcreate	xray.ximages	Create a pixel list from a region file (e.g., from SAOimage regions)
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

1. Will process all groups of a multigroup GEIS file. An OIF (imh/pix) file is like a GEIS file with one group.
2. Task is not yet functional with ACS data.

Here we describe several STSDAS tasks, located in the **stsdas.toolbox.imgtools.mstools** package, that have been designed to work 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 have associated errors, which will propagate into the error array(s) of the output file.

mscombine

This task runs the STSDAS task **gcombine** on NICMOS, STIS, and ACS data files. It divides each imset into its basic components (SCI, ERR, and DQ, plus SAMP and TIME for NICMOS). The SCI extensions then become the inputs for 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 are then combined into 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** has finished, **mscombine** then reassembles the individual output images into imsets and outputs them as one NICMOS, STIS, or ACS 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 in **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 NICMOS and STIS images in computing statistical quantities. This task is not yet functional with ACS data.

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:

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

and:

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

The data quality information carried by the NICMOS or STIS 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 NICMOS, STIS, or ACS 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 NICMOS, STIS, or ACS file containing many imsets or for performing analysis on a specific imset. The **msjoin** task does the opposite 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.

3.4.4 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/docs/photom.html

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

Calibrated HST images record signal in various units: WFPC2 and STIS imaging data are in DN (Data Numbers or counts); NICMOS data are in DN per second; ACS calibrated data are in electrons; and ACS drizzled data are in electrons per second. The pipeline calibration tasks do not alter these units in the images when performing the photometric correction step. Instead they calculate and write the 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. For ACS observers, the PHOTFLAM values for the two WFC chips are defined to be the same.

PHOTFLAM is defined as 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. Note that the word 'count' may

refer to DNs or electrons, depending on the instrument used. For example, calibrated ACS images are already corrected for the instrumental gain, and the PHOTFLAM values are computed accordingly. The PHOTFLAM values for WFPC2, on the other hand, are dependent on the gain.

Calibrated images in units of counts may be converted to flux in units of erg cm^{-2} s^{-1} Å^{-1} by multiplying the image by the value of the PHOTFLAM header keyword and dividing by the value of the EXPTIME keyword (exposure time). Calibrated images in units of counts per second (i.e. NICMOS data in DN s^{-1} and drizzled ACS data in electrons s^{-1}), may simply be multiplied by the PHOTFLAM value to obtain the flux in units of erg cm^{-2} s^{-1} Å^{-1}. NICMOS headers also contain the keyword PHOTFNU in units of Jy s. Multiplying NICMOS image by the PHOTFNU value will therefore yield fluxes in Janskys.

The STSDAS task **imcalc** may be used to convert an image from counts to flux units. For example, to create a flux-calibrated output image `outimg.fits` from an input WFPC2 image `inimg.fits[1]` with header keywords PHOTFLAM = 2.5E-18 and EXPTIME = 1000.0, type:

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

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 PHOTFLAM using **synphot**, described below. WF/PC-1 and WFPC2 observers should note that the PHOTFLAM values calculated during pipeline processing do not include a correction for temporal variations in throughput owing to contamination buildup, or Charge Transfer Efficiency (CTE) effects. 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. Consult the appropriate instrument section of the Data Handbook for more information.

If your HST image contains a source whose flux you know from ground based measurements, you may choose to calibrate 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, the following equation may be used:

$$m = -2.5 \times \log 10\,(F) + \text{PHOTZPT}$$

where the value of the PHOTZPT keyword is the zero point of the ST magnitude or STMAG scale. The STMAG system is designed so that the

specrtum of Vega has constant flux per unit wavelength. The zero point of the STMAG system is 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*). Further zeropoint corrections are necessary for converting from STMAG to other systems like Johnson/Cousins, and depend on the color of your sources. See specific photometry examples in the instrument part of the Data Handbook (Part II).

Synphot

The STSDAS synthetic photometry package, called **synphot**, can simulate HST observations of astronomical targets with known spectra. It contains throughput curves of all HST optical components, 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 HST sensitivity tables. The **bandpar** task may be used to compute the photometric parameters of a passband using the combined throughputs of the individual HST components. For example, to recalculate PHOTFLAM for an ACS observation, type:

```
sy> bandpar acs,wfc1,f555w
```

where the observation mode string is a comma separated list consisting of the instrument and its configuration, in this case the ACS detector with the WFC chip 1 and the F555W filter. (See the **obsmode** task in **synphot** and the *Synphot User's Guide* for help with these observation-mode keywords.) To see a list of observation mode keywords for the ACS, type:

```
sy> obsmode acs
```

Using the default parameters, the **bandpar** command shown above will print to the screen a table of photometric parameters. The URESP parameter contains the flux (in F_λ) of a source that produces a response of one count per second in this passband and is therefore identical to PHOTFLAM.

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

3.4.5 Combining Dithered HST Datasets with MultiDrizzle

Many HST observations make use of the technique of dithering, or offsetting the telescope to different locations in order to move the target around the detector. This can be for several reasons, including sub-pixel offsets to improve PSF sampling, offsets to move bad pixels around to different locations on the sky, or large shifts comparable to the detector size, to create large mosaics of the target field.

The recommended software to combine dithered HST datasets is MultiDrizzle (Koekemoer et al. 2002, 2004), which is a Pyraf script designed to provide fully automated image registration, cosmic ray cleaning and final image combination using the drizzle software (Fruchter & Hook 2002) and PyDrizzle. MultiDrizzle is currently available within STSDAS 3.2 as a prototype release and has been tested on a representative set of commonly-used ACS, WFPC2 and STIS observing modes.

The basic required input for MultiDrizzle is simply a set of calibrated, flat-fielded files, with additional optional inputs including a user-defined bad pixel mask, or a user-defined shift file if the header shifts in the images are not sufficiently accurate. The script carries out the following steps:

- Calculate and subtract a background sky value for each exposure

- Search for additional bad pixels that are strongly negative, that may not have been flagged in the data quality arrays.

- Determine shifts from the coordinates in the image headers, which are then applied in drizzling all the input images onto a series of separate output images that are all registered with respect to one another.

- Use the drizzled exposures to create a median (optionally choosing the minimum instead of the median if there is a large enough difference)

- Transform the median back to the frame of the original input exposures and calculate the derivative of this clean median image

- Compare the median and its derivative against the original input exposure to create a cosmic ray mask for each exposure.

- Use the cosmic ray masks in a final step that drizzles all the original input exposures together onto a single output image.

The various steps can each be turned on or off by the user, since there may be cases where not all the steps need to be run or some of them may have been run already in an earlier iteration of the script. In addition, parameters controlling the behavior of each step can be adjusted by the user. The default parameter values are set such that the script should produce a scientifically useful combined, drizzled image in a single

one-touch operation. However, this may not be the optimal scientific image for a given set of exposures, therefore access is provided to parameters for drizzle and other steps for fine-tuning the results.

Please see the HST Dither Handbook (Koekemoer et al. 2004), as well as the on-line help documentation for MultiDrizzle within Pyraf, for further information about the various parameters to the script, as well as examples showing how to use the script to combine various types of HST datasets. Please note that since the software is still actively being improved, there may be some datasets for which problems might occur, in which case users are encouraged to send email to help@stsci.edu to obtain assistance. In general, however, the code has been well tested on a wide variety of the more commonly used observing modes and should produce useful results even with default parameters.

Finally, work is currently on-going to incorporate a more robust, fully redesigned version of MultiDrizzle into the operational HST Pipeline. The initial implementation of this version will be focussed on ACS associations, thus it will produce fully CR-cleaned, drizzle-combined images for all ACS associations. It will also be released within Pyraf as an off-line version capable of running WFPC2 and STIS data, in the same way as the current version. We expect the Pipeline release to occur in mid to late Summer 2004, followed shortly by the updated release within Pyraf.

3.5 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.

3.5.1 STIS Spectra

STIS data files retrieved from the MAST Archive can contain spectra in two different forms: as long-slit spectral images in FITS IMAGE extensions or as extracted spectra in FITS BINTABLE extensions.

Plotting STIS Imaging Spectra

You can use **sgraph** in the **graphics.stplot** package of STSDAS to plot STIS long-slit spectral images 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 Section 3.3.1) allows you to see the range of your spectrum in *x* and *y* pixel space, so you can choose a suitable image section for plotting.

Plotting STIS Tabular Spectra

To plot STIS spectra in BINTABLE extensions, you first need to understand how STIS spectra are stored as binary arrays in FITS table cells. Section 2.1.2 discusses this format and describes the *selectors* syntax used to specify these data arrays. Each row of a STIS tabular spectrum contains a separate spectral order (first-order spectra will have one row, while echelle spectra will have many rows), and each column contains data of a certain type, such as wavelength or flux. To specify a particular array, you must first type the file name, then the extension containing the BINTABLE followed by the column selector, and finally the row selector. For example, to select the WAVELENGTH array corresponding to spectral order 80 of the echelle spectrum in extension 4 (EXTNAME=SCI, EXTVER=2) of `stis.fits`, you would specify the file as either:

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

or

stis.fits[sci,2][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 flux vs. 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.

3.5.2 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 Section 2.2.1 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. 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.

Sgraph 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 separate parameters, rather than enclosed in the standard square brackets.

3.5.3 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 CL command window.

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> sgraph o43balbnm_x2d.fits[1][*,100:1000] device=psi_land
st> gflush
/tmp/pskxxx
```

The above commands would write a plot in landscape-mode into a temporary PostScript file, named /tmp/pskxxxx 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 3.4 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 3.4: Sample igi Plot.
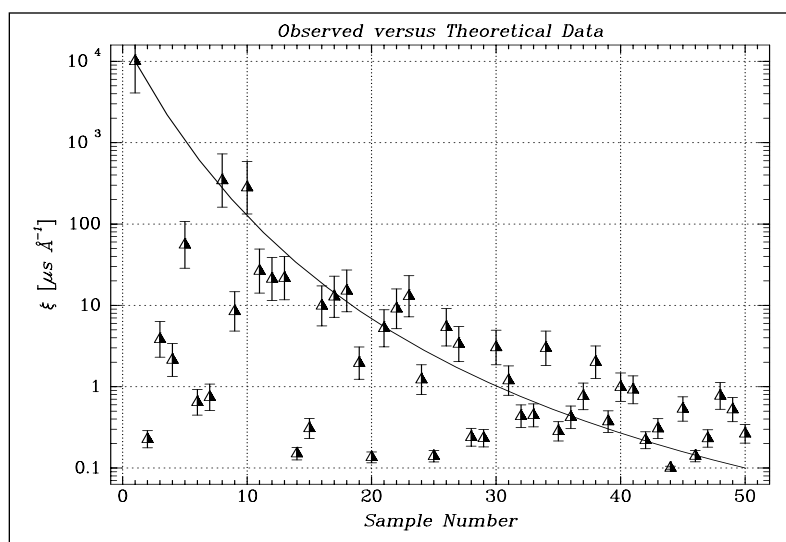
If you are working in the Python/PyRaf environment, the plotting library **matplotlib** is available. It uses most of the MATLAB syntax.

3.6 Analyzing HST Spectra

This section describes some IRAF/STSDAS tasks that can be used for analyzing and manipulating spectral data. Some 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 the data (except for STIS two-dimensional spectra). Before discussing these tasks we will first show how to recast your data into forms that are more generally accessible.

3.6.1 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 analyze 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 in Section 2.1.2. However, in order to use IRAF tasks that rely on the multispec format WCS, such as **splot**, or other STSDAS tasks that do not understand three-dimensional tables, you will have to prepare your data appropriately. This section describes some 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 spectral order (or several, for echelle data) 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 "mkmultispec" on page 3-31). 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, type

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

The output file format of **tomultispec** will be OIF regardless of the specified extension. This format is similar to GEIS format, (see Section A.2.6). OIF format files have a header component (extension `.imh`) and a binary data component (extension `.pix`).

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:

```
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
```

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

Tomultispec extracts the calibrated flux by default. However, other intensity data (e.g., NET counts) can be extracted by specifying the `flux_col` parameter appropriately.

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

STIS spectral information is stored as data arrays within individual cells of FITS binary tables (see Section 2.1.2). 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 each spectral order. The **txtable** task 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 of 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 tabular 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
```

3.6.2 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 arrays.

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 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`.

There is another method by which **mkmultispec** can incorporate wavelength information into a flux file and that 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 necessary 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 them (thereby degrading the individual spectra to a common linear dispersion scale) before joining them. Instead, you could create separate tables for 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.

3.6.3 Photometry

For STIS, FOS and GHRS spectra, photometric correction is done by the pipeline during spectral extraction, resulting in flux-calibrated spectra. For STIS, see the "Photometric Calibration" section 3.4.12, in the *STIS Data Handbook*.

3.6.4 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: Spectral Analysis Tasks

Task	Package	Input Format	Purpose
boxcar	images.imfilter	Image	Boxcar smooth a list of images
bplot	noao.onedspec	Multispec image[1]	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 Section 3.6.5)
ngaussfit	stsdas.analysis.fitting	Image, table	Interactive 1-D multiple Gaussian fitting (see Section 3.6.5)
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 Section 3.6.2)
sarith	noao.onedspec	Multispec image[1]	Spectrum arithmetic
scombine	noao.onedspec	Multispec image[1]	Combine spectra
sfit	noao.onedspec	Multispec image[1]	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 Section 3.5.2)
specalign	stsdas.hst_calib.ctools	GEIS image	Align and combine shifted spectra (see **poffsets**)
specplot	noao.onedspec	Multispec image[1]	Stack and plot multiple spectra
splot	noao.onedspec	Multispec image[1]	Plot and analyze spectra & image lines (see "splot" on page 3-35)

1. Multispec image is a spectrum created with **tomutispec** or **mkmultispec**.

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 "mkmultispec" on page 3-31).

The **splot** task has *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 "tomultispec" on page 3-28), 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 that the beam label which indicates 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
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

3.6.5 STSDAS Fitting Package

The STSDAS **fitting** package contains several tasks, as listed in Table 3.7, for fitting and analyzing spectra and images. 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 only apply to linear coordinate systems and therefore would require resampling of your data onto a linear wavelength scale first. 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 (see "imtab" on page 3-32).

Table 3.7: Tasks in the STSDAS fitting Package

Task	Purpose
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, restricting 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 $\boxed{\text{S}}$ 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 $\boxed{\text{B}}$ keystroke.

- Use the $\boxed{\text{R}}$ 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 $\boxed{\text{P}}$.

- Press $\boxed{\text{F}}$ 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[2] 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 have a good fit, and then turn the error estimation on for one last iteration.

Figure 3.5 and Figure 3.6 show 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 3.6.

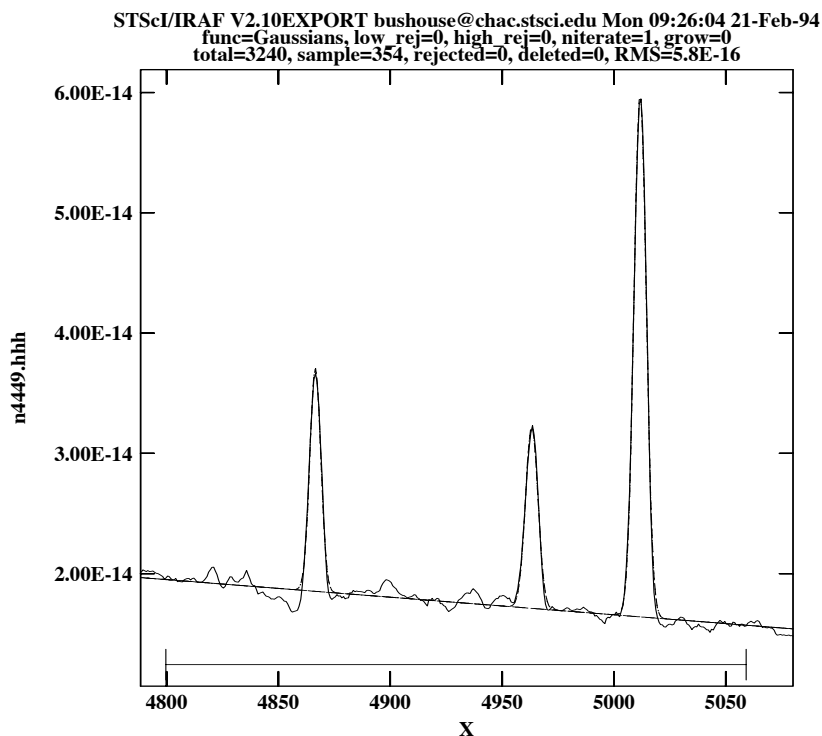2. To see the online help for details and a complete listing of cursor mode colon commands: type `help cursor`.

Figure 3.5: Fitting Hβ and [OIII] Emission Features in NGC 4449

STScI/IRAF V2.10EXPORT bushouse@chac.stsci.edu Mon 09:26:04 21-Feb-94
func=Gaussians, low_rej=0, high_rej=0, niterate=1, grow=0
total=3240, sample=354, rejected=0, deleted=0, RMS=5.8E-16

Figure 3.6: 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
```

3.6.6 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. Extensive online help is available to guide you through the task,[3] 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 flux 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.

3.7 References

3.7.1 Available from STScI

From the STSDAS web page,

http://www.stsci.edu/resources/software_hardware/stsdas/

the following documents are available:

- *STSDAS Users Guide,* version 1.3, September 1994.
- *STSDAS Site Manager's Installation Guide and Reference*, version 3.0, July 2002.
- *Synphot Users Guide*, December 1998.
- *IGI Reference Manual*, version 2.0, September 1998
- *Dither Handbook*, version 2.0, January 2002.

---

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

3.7.2 Available from NOAO

From the NOAO web page, http://iraf.noao.edu/docs/photom.html, the following documents are available:

- *A Beginners Guide to Using IRAF,* 1993, J. Barnes.

- *Photometry Using IRAF*, version 2.10, 1994, L. Wells.

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

3.7.3 Other References Cited in This Chapter

- Fruchter, A. S & Hook, R. N. 2002, PASP 114, 144

- 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.

- Koekemoer, A. M., Fruchter, A. S., Hook, R. N., & Hack, W. 2002, HST Calibration Workshop, ed. S. Arribas, A. M. Koekemoer, & B. Whitmore (STScI: Baltimore), 337

- Koekemoer, A. M. et al. 2004, HST Dither Handbook, V3.0

- 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.

- www.stsci.edu/resources/software_hardware/pyraf

- www.stsci.edu/resources/software_hardware/pyfits

- www.stsci.edu/resources/software_hardware/numarray

IRAF Primer

In this appendix . . .

The Image Reduction and Analysis Facility (IRAF), developed by the National Optical Astronomy Observatories (NOAO), forms the basis of the Space Telescope Science Data Analysis System (STSDAS). IRAF contains numerous packages of programs, called *tasks*, that perform a wide range of functions from reading data tapes to producing plots and images. Most astronomers will already be familiar with IRAF, but we provide this tutorial for HST observers who are beginners with IRAF. It includes information on:

- How to set up IRAF the first time you use the software.

- How to start and stop an IRAF session.

- Basic concepts, such as loading packages, setting parameters, etc.

- How to use the on-line help facility.

Additional information on IRAF, in particular *A Beginner's Guide to Using IRAF* is available through the NOAO IRAF Home Page at: http://iraf.noao.edu

A.1 Initiating IRAF

This section explains:

- How to set up your IRAF working environment.
- How to start and logout of the IRAF program.

▽ *We assume that your site has IRAF and STSDAS installed. If not, you must obtain and install the software. See Appendix Section A.3 for details.*

A.1.1 Setting Up IRAF in Unix/Linux

Before running IRAF for the first time you need to follow these three steps:

1. Create your IRAF root directory.

2. Move to that directory and set the necessary environment variables or system logicals and symbols.

3. Run **mkiraf** to create a `login.cl` file and a `uparm` subdirectory.

Users generally name their IRAF home directory `iraf` (also referred to as your IRAF *root* directory) and set it up in their account's root directory (i.e., the default directory that you are in when you log in to the system). The IRAF home directory doesn't need to be in your account's root directory, nor does it need to be called `iraf`, but you should *not* put it on a scratch disk that is periodically erased.

If you call your root IRAF directory "`iraf`", you can set up IRAF as follows:

Under Unix:

```
% mkdir iraf
% cd iraf
% setenv iraf /usr/stsci/iraf/
% source $iraf/unix/hlib/irafuser.csh
% mkiraf
```

Can be placed in .login file →

The directory name is site-dependent—check with your system staff

The **mkiraf** command initializes IRAF by creating a `login.cl` file and a subdirectory called `uparm`. After typing the **mkiraf** command, you will see the following:

```
% mkiraf
-- creating a new uparm directory
Terminal types: gterm=ttysw+graphics,vt640...
Enter terminal type:
```

Enter the type of terminal or workstation you will most often use with IRAF. [1] Generic terminal types that will work for most users are:

- `vt100` for most terminals.

- `xtermjhs` for most workstations running under X-Windows.

- `xgterm` for sites that have installed X11 IRAF and IRAF v2.10.3 BETA or later.

IRAF for other systems, like Mac OSX, can be obtained from the IRAF web page at: http://iraf.noao.edu

You can change your terminal type at any time by typing (`set term=new_type`) during an IRAF session. You can also change your default type by editing the appropriate line in your login.cl file.

After you enter your terminal type, you will see the following output before getting your regular prompt:

```
A new LOGIN.CL file has been created in the current ...
You may wish to review and edit this file to change ...
```

The `login.cl` file is the *startup file* used by the IRAF command language (CL). It is similar to the `.login` file used by Unix. Whenever IRAF starts, it looks at the `login.cl` file. You can edit this file to customize your IRAF environment. In fact, you should look at it to make sure that everything in it is correct. In particular, there is a line starting with `set home =` that tells IRAF where to find your IRAF home directory. You should verify that this statement does, in fact, point to your IRAF directory. If you will be working with standard IRAF format images you should also insert a line saying `set imdir = "HDR$"`. The `imdir` setting is ignored when working with GEIS format images.

---

1. Users at STScI should consult the *STScI Site Guide for IRAF and STSDAS.*

The uparm directory will contain your own copies of IRAF task parameters. This directory allows you to customize your IRAF environment by setting certain parameter values as defaults. Once you set up IRAF, you should rarely need to do it again, expect when updated version of IRAF are installed.

A.1.2 Starting and Stopping an IRAF Session

To start an IRAF session:

1. Move to your IRAF home directory.

2. Type cl.

IRAF starts by displaying several lines of introductory text and then puts a prompt at the bottom of the screen. Figure A.1 is a sample IRAF startup screen.

Figure A.1:IRAF Startup Screen

```
 NOAO Sun/IRAF Revision 2.11 Fri Aug 15 15:34:46 MST 1997
     This is the EXPORT version of Sun/IRAF V2.11 for SunOS 4 and Solaris 2.5

     Welcome to IRAF.  To list the available commands, type ? or ??.  To get
     detailed information about a command, type `help command'.   To  run  a
     command  or  load  a  package,  type  its  name.   Type  `bye' to exit a
     package, or `logout' to get out of the CL.  Type `news'  to  find  out
     what is new in the version of the system you are using.   The following
     commands or packages are currently defined:

      apropos          euv.              local.             spptools.
      ared.            fitsutil.         mem0.              stlocal.
      aspec.           focas.            newimred.          stsdas.
      c128.            ftools.           noao.              system.
      color.           hst_pipeline.     obsolete.          tables.
      ctio.            images.           plot.              utilities.
      dataio.          imcnv.            proto.             vol.
      dbms.            language.         rvsao.             xray.
      digiphotx.       lists.            softools.
 cl>
```

Startup Messages Change from Day to Day

Available Packages and Tasks

To quit an IRAF session:

1. Type logout.

A.2 IRAF Basics

This section describes basic IRAF techniques such as:

• Loading packages (below).

• Running tasks and commands.

- Getting online help.

- Viewing and setting parameters (see Appendix Section A.2.4).

- Setting and using environment variables (see Section A.2.5).

- File management

- Troubleshooting

A.2.1 Loading Packages

In IRAF jargon, an application is called a *task* and logically related tasks are grouped together in a *package*. Before you can use a task, you must load the package containing that task. To load a package, type the name of the package. The prompt will then change to the first two letters of the package name, and the screen will display the names of all the newly available tasks and subpackages. Even though the prompt has changed, previously loaded packages remain loaded, and all their tasks remain available.

Note that the standard way to specify a path through the IRAF package hierarchy to a task in a particular subpackage is to separate the package names with periods (e.g., **stsdas.hst_calib.foc.focgeom.newgeom**).

Most frequently used packages can be added to your `login.cl` file. Just list the packages one per line in that file and each will be loaded automatically when Iraf is started.

Figure A.2:Loading Packages
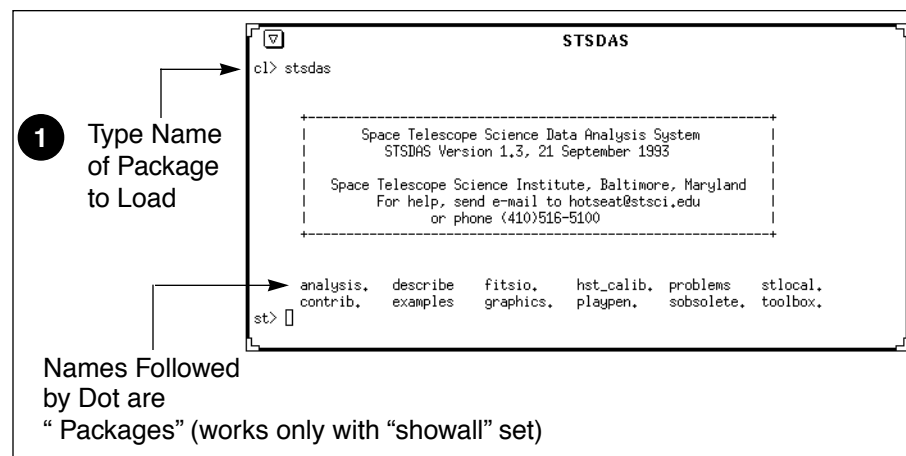
Some helpful commands for managing packages are:

- `?` - Lists tasks in the most recently-loaded package.

- `??` - Lists all tasks loaded, regardless of package.

- package - Lists names of all loaded packages.

- bye - Exits the current package.

A.2.2 Running Tasks

This section explains how to run tasks, background tasks, and system-level commands, and how to use piping and redirection.

Running a Task

The simplest way to run a task is to type its name or any unambiguous abbreviation of it. The task will then prompt you for the values of any required *parameters*, such as the names of input files. Alternatively, you can specify the values for the required *parameters* on the command line when you run the task. For example, if you want the task imheader to print header information on the file myfile.hhh, you can type

```
st> imhead myfile.hhh
```

IRAF does not require you to type the complete command name—only enough of it to make it unique. For example, "dir" is sufficient for directory.

Escaping System-Level Commands

To run an operating system-level command (i.e., Unix or VMS commands) from within the IRAF CL, precede the command with an exclamation point (!). This procedure is called *escaping* the command. For example:

```
st> !system_command
```

Piping and Redirection

You can run tasks in sequence if you desire, with the output of one task being used as the input for another. This procedure, called *piping*, and is done by separating commands with a vertical bar (|), using the following syntax:

```
st> task1 filename  |  task2
```

For example, if a particular task prints a large volume of textual output to the screen, you will often want to pipe it to `page`, which allows you to read the output one page at a time:

```
st> task1 filename | page
```

You can also redirect output from any task or command to a file by using the greater-than symbol (>) as follows:

```
st> command > outputfile
```

Background Tasks

To run a task as a background job, freeing your workstation window for other work, add an ampersand (&) to the end of the command line, like this:

```
st> taskname &
```

A.2.3 Getting Help

This section describes:

- How to use IRAF's on-line help facility.

- How to find a task that does what you want (see "Finding Tasks" on page A-8).

- IRAF tutorials and `cl` tips.

On-Line Help

You can get on-line help with any IRAF task or package by using the **help** command,[2] which takes as an argument the task or package name about which you want help. Wildcards are supported. For example, to display the on-line help for the STSDAS **mkmultispec** task, you would type:

```
fi> help mkmultispec
```

---

2. There is an optional *paging* front-end for help called **phelp**. For more information, type `help phelp` from within IRAF.

Figure A.3:Displaying On-line Help

Two STSDAS tasks that display only certain sections of the help file are also available:

- **examples** - Displays only the examples for a task.

- **describe** - Displays only the description of the task.

Typing help *package* will produce one-line descriptions of each task in the package.

Finding Tasks

There are several ways to find a task that does what you need:

- Use help *package* to search through the IRAF/STSDAS package structure.

- Use the **apropos** task as shown in Figure A.4 to search the online help database. This task looks through a list of IRAF and STSDAS package menus to find tasks that match a specified keyword. Note that the name of the package containing the task is shown in parentheses.

- Ask a more experienced user, who can usually point you in the right direction.

IRAF Tutorials and `cl` tips

A number of tricks and tutorials are available online to help the user to become more familiar with IRAF. See:
http://iraf.noao.edu/iraf/web/tutorials/tutorials.html and
http://iraf.noao.edu/tips/cl.html

Figure A.4:The **apropos** task Using apropos

```
                              STSDAS

ct> apropos WCS
     wcslab - Overlay a displayed image with a world coordinate grid  (cl.images.
tv)
     wcsedit - Edit the image coordinate system (cl.proto)
     wcsreset - Reset the image coordinate system (cl.proto)
     makewcs - Write the WCS on the image header based on the plate sol. (stsdas.a
nalysis.gasp)
     wcslab - Produce sky projection grids for images. (stsdas.graphics.stplot)
     wlpars - Pset to specify characteristics of WCS labelled graphs. (stsdas.gra
phics.stplot)
     wcspars - Pset to specify a WCS. (stsdas.graphics.stplot)
mkmultispec - Combine wavelength and data with the MULTISPEC MWCS. (stsdas.hst_c
alib.ctools)
ct> ▮
```

Look for Tasks Dealing with World Coordinates

Package

A.2.4 Setting Parameters

Parameters specify the input information for IRAF tasks. They can be the names of input or output files, particular pixel numbers, keyword settings, or many other types of information that control the behavior of the task.

The two most useful commands for handling parameters are:

- **lparam** to display the current parameter settings (often abbreviated **lpar**).

- **eparam** to edit parameters (often abbreviated **epar**).

Viewing Parameters with lparam

The **lpar** command lists the current parameter settings for a given task (Figure A.5).

Figure A.5:Displaying Parameter Settings with lpar

① Type *lpar* Followed by Name of Task

Parameters and Current Settings

```
                              STSDAS

fi> lpar strfits
     fits_file = "mtg"                FITS data source
     file_list = "1-999"              File list
     iraf_file = ""                   IRAF filename
     (template = "")                  template filename
     (long_header = no)               Print FITS header cards?
     (short_header = yes)             Print short header?
     (datatype = "default")          IRAF data type
     (blank = 0.)                     Blank value
     (scale = yes)                    Scale the data?
     (xdimtogf = yes)                 Transform xdim FITS to multigroup?
     (oldirafname = yes)              Use old IRAF name in place of iraf_file?
     (offset = 0)                     Tape file offset
     (mode = "ql")
fi> ▮
```

Setting parameters with eparam

The **epar** command is an interactive parameter set editor. It displays all of the parameters and their current settings on the screen. You can move around the screen using the arrow keys (also called *cursor* keys) and type

new settings for any parameters you wish to change. Figure A.6 shows a sample of the **epar** editor at work (invoked by typing epar strfits).

Figure A.6:Editing Parameters with epar in IRAF

Parameter Data Types—What to Specify

Parameters are either *required* or *hidden*, and each parameter expects information of a certain *type*. Usually, the first parameter is required, and very often it expects a file name. Parameters are described in the online help for each task [include reference to help]. Hidden parameters, shown in parentheses in the online help and the **lpar** and **epar** listings, need not be specified at each execution because their default values frequently suffice.

Wise IRAF users will check the values of hidden parameters, as they often govern important aspects of a task's behavior.

If you specify the wrong type of information for a parameter, **epar** will usually display an error message saying something like "Parameter Value is Out of Range." The message is displayed when you move to another parameter or if you press Return. Table A.1 lists the different parameter types.

Table A.1: Parameter Data Types

Type	Description
File Name	Full name of the file. Wild card characters (* and ?) are often allowed. Some tasks allow you to use special features when specifying file names, including "@" lists, IRAF networking syntax, and image section or group syntax. (See "File Management" below).
Integer	Whole number. Often the task will specify minimum or maximum values (see the help pages).
Real	Floating point numbers, can be expressed in exponential notation. Often will have minimum and maximum values.
Boolean	Logical "yes" or "no" values.
String	Any characters. Sometimes file names are specified as string.
Pset	Parameter set.

Restoring Parameter Default Values

Occasionally, IRAF (or you) will get confused by your parameter values. To alleviate this confusion, you can restore the default parameters with the **unlearn** command. You can use **unlearn** on either a task or on an entire package.

The unlearn *command generally will restore the parameters to reasonable values, a big help if you are no longer sure which parameter values you have changed in a complicated task.*

A.2.5 Setting Environment Variables

IRAF uses *environment variables* to define which devices are used for certain operations. For example, your terminal type, default printer, and the disk and directory used for storing images are all defined through environment variables. Environment variables are set using the **set** command and are displayed using the **show** command. Table A.2 lists some of the environment variables that you might want to customize.

Table A.2: Environment Variables

Variable	Description	Example of Setting
printer	Default printer for text	set printer = lp2
terminal	Terminal type	set term = xgterm
stdplot	Default printer for all graphics output	set stdplot = ps2
stdimage	Default terminal display setting for image output (most users will want this set to either imt512 or imt800)	set stdimage = imt800
stdgraph	Default graphics device	set stdgraph = xgterm
clobber	Allow or prevent overwriting of files	set clobber = yes
imtype	Default image type for output images. "imh" is original IRAF format, (OIF) "hhh" is STS-DAS GEIS format.	set imtype = "hhh"

▽ *If you are working with GEIS files, you should set imtype to "hhh". If you are working with ACS, STIS and NICMOS data in FITS files, you can set imtype to "fits".*

You can set your environment variables automatically each time you login to IRAF by adding the appropriate commands to your login.cl file. Use your favorite text editor to specify each variable on its own line. The **show** command with no arguments prints the names and current values of all environment variables.

A.2.6 File Management

This section describes:

- File formats commonly used with STSDAS and IRAF.

- Specification of file names.

- Navigation through directories.

File Formats

IRAF recognizes a number of different file structures. Among them are the standard HST file formats known as GEIS and FITS (see chapter 2 of the HST Introduction), both of which differ from the original IRAF format (OIF). GEIS is closer to OIF, in that two files are *always* used together as a pair:

- A *header file*, which consists of descriptive information. IRAF (OIF) header files are identified by the suffix `.imh`. GEIS header files are in ASCII text format and are identified by the suffix .hhh or another suffix ending in "h", such as .c0h or .q1h.

- A *binary data file*,[3] consisting of pixel information. IRAF data file names end with a `.pix` suffix. STSDAS data files end with a suffix of .hhd or another suffix that ends with "d", such as .c0d or .q0d.

STSDAS always expects both component files of a GEIS image to be kept together in the same directory. A single FITS file contains both the header information and the data.

When working with IRAF or STSDAS images, you need only specify the header file name—the tasks will automatically use the binary data file when necessary.

File Specification

Most tasks in IRAF and STSDAS operate on files and expect you to specify a file name for one or more parameters. Several types of special syntax can be used with certain tasks when specifying file names. These syntax features include:

- **Wild card characters**, often called *templates*, which are used to specify multiple files using pattern matching techniques. The wild cards are:

  - * Matches any number of characters, e.g. `z*.c0h`
  - ? Matches any single character, e.g. `z01x23x.c?h`

---

3. The binary data file format is host-dependent and may require translation before it can be moved to a computer using a different architecture.

When using wildcards with image-processing tasks, be sure to exclude the binary pixel files by ending your file name specification with an "h", for example: y*.??h

- **List files**, often called *@-files*, are ASCII files that contain lists of file names, one per line. If your task supports the list file feature, you would type the name of your list file, preceded by the "@" character. For example: `@infiles.txt`

- **Image section** specification. Tasks that work with image data will allow you to specify a part of the image rather than the entire image. To extract a particular image section, specify each axis range in square brackets, for example: `image.hhh[10:200,20:200]`

- **IRAF networking** specification. IRAF is capable of reading and writing files to and from remote systems on a network. This feature is often used with tasks in the **fitsio** and **convfile** packages, or with image display tasks. The *STSDAS Users Guide* and the online help (type `help networking`) describe how to enable this feature. To specify that you want to use the IRAF networking feature, type the remote host name followed by an exclamation point (!), followed by the file or device name. For example: `ra!mta`.

For example, when displaying from an IRAF session running on a remote machine back to your work station set the environment variable "node" by typing: `set node= my_workstation`!

Directory Navigation

To navigate through directories, you can use the following commands:

- **path** or **pwd** - Lists the current working directory.

- **cd** *directory* - Move to the named directory.

A.2.7 Troubleshooting

There are a couple of easy things you can do to make sure that you don't have a simple memory or parameter conflict—common causes of problems.

- Look at the parameter settings and make sure that you have specified reasonable values for every parameter.

- When you run an IRAF task for the first time in a session, IRAF stores the executable file in its *process cache*. If IRAF appears not to be running your tasks properly, you may need to use the **flprcache** command to clear the process cache. To do this type: `flpr` Sometimes you will need to execute this command twice in succession.

- Occasionally, you may need to logout of the CL, restart IRAF, and try your command again.

If you still have a problem, contact the STScI Help Desk at `help@stsci.edu`

A.3 Getting IRAF and STSDAS

Both IRAF and STSDAS are provided free of charge to the astronomical community. You must have IRAF and PyRAF to run STSDAS. Detailed information about installing and retrieving STSDAS is found in the *STSDAS Site Manager's Installation Guide and Reference* which can be dowloaded in postscript or PDF format from the World Wide Web page at:

`http://www.stsci.edu/resources/software_hardware/stsdas/download`

If you have any problems getting and installing STSDAS, TABLES, or any other packages or data described in this handbook, please contact the Help Desk by sending e-mail to: `help@stsci.edu.`

A complete description of how to install the **synphot** data files is provided in Section A.3.2.

A.3.1 Retrieving the IRAF and STSDAS Software

There are two ways to get the software:

- Use the World Wide Web page shown above

This page provides links and instructions for downloading the appropriate files to your local system or to display the software directory, from which you can select the series of smaller files.

- Use anonymous FTP

    Type `ftp ftp.stsci.edu`
    login: `anonymous`
    password: `(your email address)`

and go to the directory `software/stsdas` to retrieve source and binary files. Binaries are platform-specific and the filename contains the name of the platform, see the README file in that directory.

When you retrieve STSDAS, you must also retrieve the TABLES package, and TABLES must be installed first. You must have IRAF and Pyraf installed on your system in order to install TABLES and STSDAS.

The complete instructions for installing STSDAS, TABLES, and all of the supporting software and reference files (including instrument reference files and the **synphot** dataset) for the IRAF environment are found in the file called InstallGuide.book32.ps (which contains a copy of the *STSDAS Site Manager's Installation Guide and Reference*). Ancillary information and documentation for STSDAS and **synphot** are available in the `software/stsdas/docs` directory on the ftp site.

Registration

The software can also be registered and requested using on-line forms available through World Wide Web at the following URL:

`http://www.stsci.edu./resources/software_hardware/stsdas/STSDASRegistrationForm`

When you request the STSDAS software, you can also ask for the appropriate version of IRAF, which will be requested for you— simply check the appropriate box on the form under "Do You Already Have IRAF Installed?" If you prefer to request the IRAF software independent of STSDAS, you can do so by sending e-mail to: `iraf@noao.edu`

A.3.2 Getting the Synphot Database

This manual sometimes refers to the **synphot** dataset, which must be available in order to run tasks in the STSDAS **synphot** package. These data files are not included with the STSDAS software and must be retrieved independently. To do this, you may either retrieve the Synphot data online at `www.stsci.edu/resources/software_hardware/stsdas/synphot` or you can retrieve a series of compressed tar files from the STScI FTP site in the directory `software/stsdas/refdata/synphot`.
After uncompressing and extracting the tar files (see below), you need to unpack the FITS files as described below.

The synthetic photometry data are read in similar way as the instrument datasets, using the script `unpack.cl` provided in the top directory. This script is run within IRAF to convert data from FITS format into the format used by the **synphot** task. This script assumes you have the logical `crrefer` set up in your `extern.pkg` file (which is in the directory

`$iraf/unix/hlib` or have it set up in your session. You do this by placing the command below in `extern.pkg` or by typing it on the command line:

```
set crrefer = "/node/partition/stdata/synphot/"
```

Figure shows how to convert the files.

Unpacking Synthetic Photometry Files

```
% cl                                                     Just in case...
cl> cd /node/partition/stdata/synphot
cl> set crrefer = "/node/partition/stsdata/synphot/"
cl> task $unpack = unpack.cl
cl> tables
ta> fitsio                        The "$" is used because the task
fi> unpack                        has no parameter file
```

Alternatively you may retrieve the Synphot data online at:

http://www.stsci.edu/resources/software_hardware/stsdas/synphot

and follow the instructions for downloading and installing.

Note that all five synphot files must be unloaded for the script to complete successfully.

A.3.3 Extracting the Synphot Unix Tar Files

If you retrieved the **synphot** database as compressed tar files, you will need to copy them to an appropriate subdirectory and then expand and unpack the files. The `tar` and `compress` utilities that do this are commonly available on most Unix systems, but are not standard in the VMS environment. The examples shown below reflect Unix usage. If you are on a VMS system, you should consult with your systems support staff regarding the availability and usage of these commands. To process the files on a Unix system:

1. Get the compressed tar file that you want, as described in previous sections.

2. Make an appropriate subdirectory using the `mkdir` command.

3. Pipe the compressed tar file through the `uncompress` and `tar` files to expand and unpack the file.

The following example shows how to do this. The example assumes that you are putting the files in a subdirectory under /usr/iraf/stdata (note that the name of your file here is assumed to be XXX.tar.Z).

```
% pwd
/usr/iraf/stdata
% mkdir XXX
% mv XXX.tar.Z XXX/
% cd XXX
% cat XXX.tar.Z | uncompress | tar -xf -
```
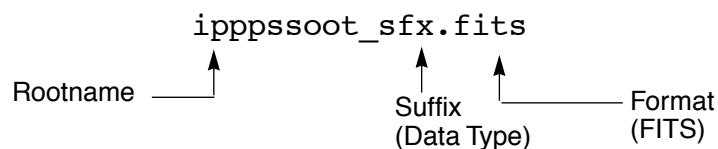
HST File Names

In this appendix. . .

This appendix describes the syntax of HST data file names, which encode a large amount of information about the files themselves. Datasets retrieved from the Archive as described in Chapter 1 of the HST Introduction (Part I) consist of multiple files in FITS format, each with a name that looks like this:

```
ipppssoot_sfx.fits
```

Rootname ──────┘   Suffix   └────── Format
                          (Data Type)   (FITS)

- *Rootname*: The first part of the file name (`ipppssoot`) is the *rootname* of the dataset to which the file belongs. All files belonging to a given dataset share the same rootname.

- *Suffix*: The three-character second part of the name (`sfx`) is called the *suffix*, and it indicates the type of data the file contains.

- *Format*: The identifier `.fits` indicates that this file is in FITS format.

For example, an FOC data file named `x3l80101t_d0f.fits` is a FITS file belong to the dataset with rootname `x3l80101t`, and its suffix `d0f` indicates that it contains raw science data.

In order to use IRAF/STSDAS tasks to work with data from instruments other than NICMOS, STIS, or ACS, you will want to convert these FITS files into GEIS format. See Section 2.1 in the HST Introduction for instructions on how to convert FITS files to GEIS files using **strfits**. Like FITS files, the names of GEIS files also derive from a file's rootname and suffix, and they look like this:

```
ipppssoot.sfx
```

Generally the suffixes of GEIS files end either in "d", indicating a binary data file, or "h", indicating an ASCII header file. The two GEIS files `x3l80101t_d0h` and `x3l80101t_d0d` together contain the same information as the single FITS file `x3l80101t_d0f.fits`.

The identifier referred to here as a "suffix" has often been called an "extension" in the past. However, the individual pieces of FITS files are also known as "extensions" (see Section 2.1.1 in the HST Introduction). For clarity, this handbook will use the term "extension" when referring to a component of a FITS file and the term "suffix" when referring to the three character identifier in a filename.

B.1 Rootnames

Rootnames of HST data files follow the naming convention defined in Table B.1.

Table B.1: IPPPSSOOT Root File Names

Character	Meaning
I	Instrument used, will be one of: *E* - Engineering data *F* - Fine Guidance Sensors *J* - Advanced Camera for Surveys *N* - Near Infrared Camera and Multi-Object Spectrograph *O* - Space Telescope Imaging Spectrograph *S* - Engineering subset data *T* - Guide star position data *U* - Wide Field/Planetary Camera-2 *V* - High Speed Photometer *W* - Wide Field/Planetary Camera *X* - Faint Object Camera *Y* - Faint Object Spectrograph *Z* - Goddard High Resolution Spectrograph
PPP	Program ID; can be any combination of letters or numbers (46,656 combinations possible). There is a unique association between program ID and proposal ID.
SS	Observation set ID; any combination of letters or numbers (1,296 possible combinations).
OO	Observation ID; any combination of letters or numbers (1,296 possible combinations).
T	Source of transmission or association product number *M* - Merged real time and tape recorded *N* - Retransmitted merged real time and tape recorded *O* - Retransmitted real time (letter 'O') *P* - Retransmitted tape recorded *Q* - Solid-state recorder *R* - Real time (not recorded) *S* - retransmitted solid-state recorder *T* - Tape recorded *0* - Primary association product (number zero) *1-8* - NICMOS or ACS background association product

B.2 Suffixes of Files Common to all Instruments

The three-character suffix of a data file (e.g., `d0h`) identifies the type of data that a file contains. Because the meanings of these suffixes change from instrument to instrument, please refer to the appropriate instrument-specific Data Structures chapter for their definitions. Several types of file suffixes are, however, common to all instruments.

OMS Files

Observatory Monitoring System (OMS) files were replaced by the Engineering Data Processing System (EDPS) after February 2003. Observations before February 2003 have OMS files. OMS files with suffixes `cm*` or `ji*`, contain Observation Logs describing how the HST spacecraft behaved during a given observation. OMS headers, which you can read with the IRAF task **imheader** (see Section 2.2.3 in the HST Introduction), are divided into groups of keywords that deal with particular topics such as spacecraft data, background light, pointing control data, and line-of-sight jitter summary. The headers themselves provide short descriptions of each keyword. OMS tables and images record spacecraft pointing information as a function of time. For more information on OMS files, you can consult Appendix C or the STScI Observation Logs Web pages at:

http://www.stsci.edu/hst/observatory/pointing/obslog/OL_1.html

PDQ Files

The suffix `pdq` denotes Post Observation Summary and Data Quality Comment files which contain predicted as well as actual observation parameters extracted from the standard header and science headers. These files may also contain comments on any obvious features in the spectrum or image, as noted in the OPUS data assessment, or automatically extracted information about problems or oddities encountered during the observation or data processing. These comments may include correction to the keywords automatically placed in the OMS files. PDQ were discontinued on May 9, 2002.

OCX Files

The suffix `ocx` denotes Observer Comment Files which are produced by STScI personnel to document the results of real-time commanding or monitoring of the observation, along with keywords and comments. Prior to April 17, 1992, OCX files were not always archived separately and, in some cases, were prepended to the trailer file.

After early February 1995, OCX files were produced only when an observation was used to locate the target for an Interactive Target Acquisition. At this time, mission and spacecraft information were moved

to the PDQ reports and the Observation Logs (OMS jitter image and jitter table). OCX files were also discontinued on May 9, 2002.

Trailer Files

Trailer files (suffix `trl`) are FITS ASCII tables that log the processing of your data by the OPUS pipeline.

Note that trailer files are formatted with 132 columns

EDPS Files (post February 19, 2003)

These files contain Observation Logs describing how the HST spacecraft behaved during a given observation. These files replaced the OMS files after February 19, 2003.

B.3 Associations

The NICMOS, STIS, and ACS calibration pipelines sometimes produce single calibrated images from *associations* of many exposures. These associations allow HST pipeline processing to proceed further than it has in the past. For example, a NICMOS or ACS observer might specify a dithering pattern in a Phase II proposal. NICMOS or ACS would then take several exposures at offset positions, and the pipeline would combine them into a single mosaic (suffix `mos`). In this case, the original set of exposures constitutes the association, and the mosaic is the *association product*. Similarly, a STIS observer might specify a CR-SPLIT sequence in a Phase II proposal. STIS would gather several exposures at the same pointing, and the STIS pipeline would process this association of exposures into single image, free of cosmic rays, that would be the association product (suffix `crj`).

When you search the Archive with StarView for observations involving associations of exposures, your search will identify the final association product. The rootnames of association products always end in zero (see Table B.1 above.) If you request both Calibrated and Uncalibrated data from the Archive, you will receive both the association product and the exposures that went into making it. The corresponding association table, located in the file with suffix `asn` and the same rootname as the association product, lists the exposures belonging to the association. You can read this file using the STSDAS **tprint** or **tread** tasks (see Section 2.1.2 in the HST Introduction). The exposure IDs in the association table share the same `ipppss` sequence as the association rootname, followed by a base 36

number nn (n = 0-9,A-Z) that uniquely identifies each exposure, and a character t that denotes the data transmission mode (see Table B.1).

In practice, STIS stores the exposures belonging to associations differently than NICMOS or ACS. The exposures belonging to a STIS association all reside in the same file, while the exposures belonging to a NICMOS or ACS association reside in separate datasets. See the relevant Data Structures chapters for more details.

Information on the exposures belonging to an association is also available through StarView (see chapter 1 of the HST Introduction). From the <Welcome> Screen, click on **[HST Instrument Searches]** to get the <HST Instruments> screen, and then click on the **[Associations]** button for the instrument of interest. You can then search for the various exposures belonging to an association by entering the rootname of the association in the Association ID field and clicking on **[Search]**. An Association Results Screen will display the results of the search, which you can step through using the **[Scan], [Prev], [Next]** buttons. Figure B.1 below gives an example of a NICMOS Association Results Screen. Note the differences between the association rootname and coordinates and those of the individual exposure.

Figure B.1:Association Results Screen from StarView
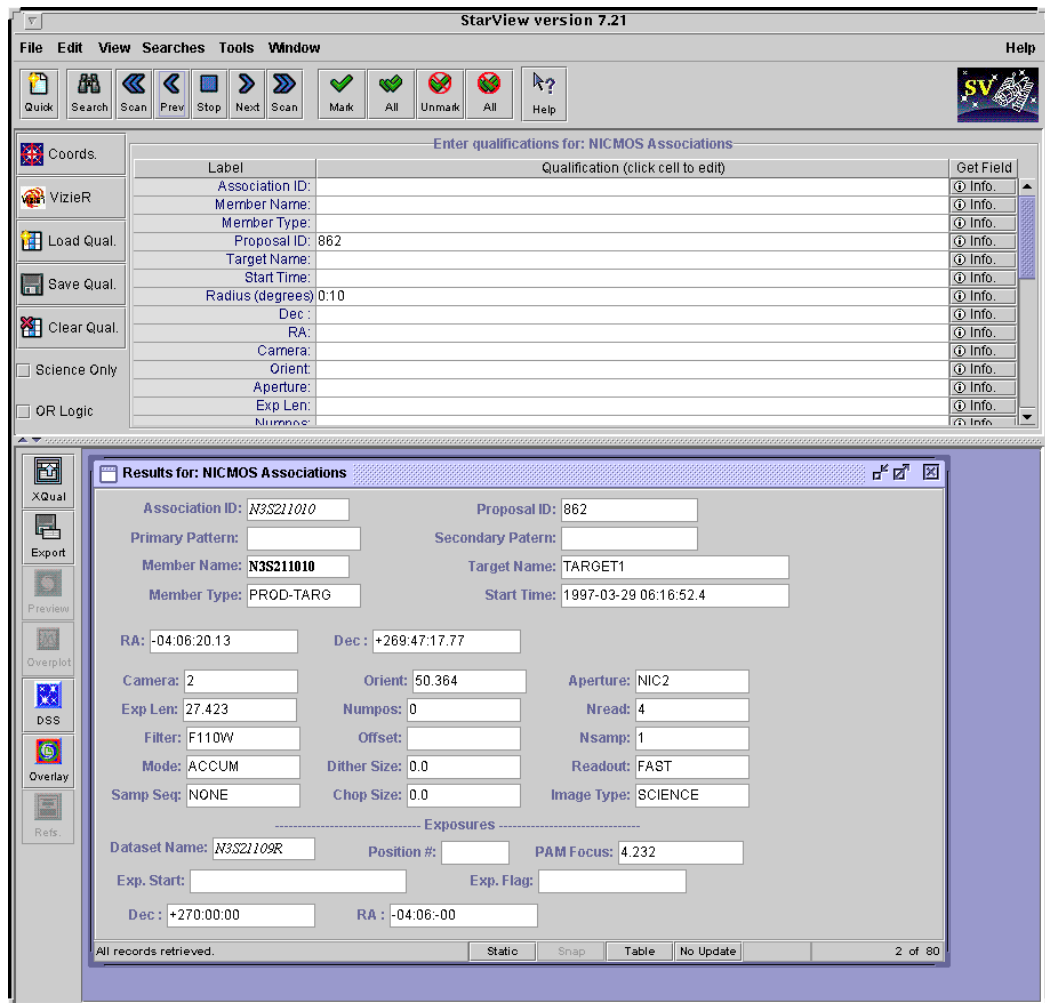
Observation Logs

In this appendix . . .

This Appendix describes the *Observation Log Files,* also known as OMS or *jitter* files. These files record pointing, jitter, and other Pointing Control System (PCS) data taken during an HST observation. You can use them to assess the behavior of the HST spacecraft during your observation, and in particular, to evaluate the jitter of the spacecraft while it was taking data. Here we describe the contents and structure of the observation log files, how to retrieve them from the Archive, and how to work with the data they contain.

C.1 Observation Log Files

Observation log files associated with each HST dataset contain pointing and specialized engineering data taken during the observation. These data files are produced by the Engineering Data Processing System (EDPS), an automated software system that interrogates the HST engineering telemetry and correlates the time-tagged engineering stream with HST's Science Mission Schedule (SMS), the seven-day command and event list that drives all spacecraft activities. The EDPS replaced the Observatory Monitoring System (OMS) in February 2003. EDPS provides observers with information about guide star acquisition, pointing, and tracking that is not normally provided in the science headers.

The observation log files share the same rootname as the observation they are associated with, except for the final character, which for observation log files is always a "j" (see appendix B for more on the names of HST data files). When OMS was installed in October 1994, it initially generated files with the suffixes cmh, cmj, cmi, which contained header information, high time resolution pointing data, and three-second average pointing data, respectively (see Table C.1). OMS observation logs changed to the jih/jid/jif image format after August 1995, at which time the cmi table was renamed jit to keep the naming convention consistent. In the OMS version of August 1995, cmj tables were replaced with a jitter image, which is a two-dimensional histogram of jitter excursions during the observation. The suffixes of the GEIS jitter image are jih for the header and jid for the image data. The jit table accompanies the jitter image. The header file of the image replaces the cmh file but includes the same information with the addition of some image-related keywords.

A detailed description of the old observation log files can be found on-line: http://www.stsci.edu/hst/observatory/pointing/obslog/OL_1.html, but for EDPS files, a description of the new Jitter File Format can be found on-line: http://www.ess.stsci.edu/projects/edps/jitter_format.html

Table C.1: OMS Observation Log Files

Suffix	Contents
October 1994 to August 1995	
cmh	OMS header
cmj	High time resolution (IRAF table)
cmi	Three-second averages (IRAF table)
_cmh.fits	Archived FITS file of cmh
_cmj.fits	Archived FITS file of cmj
_cmi.fits	Archived FITS file of cmi
August 1995 to February 1997	
jih/jid	Two-dimensional histogram and header (GEIS)
jit	Three-second averages (IRAF table)[1]
_jif.fits	Archived FITS file which bundles the jih/jid files.
_jit.fits	Archived FITS file of jit.
February 1997 onward	
_jif.fits	Two-dimensional histogram (FITS)
_jit.fits	Three-second averages table (FITS)
_jwf.fits	Two-dimensional histogram for STIS wavecals.
_jwt.fits	Three-second averages for STIS wavecals.

1. After May 11, 1995, the jit tables for exposures shorter than 6 seconds contain higher-resolution, one-second average pointing data.

▽ *Pointing and tracking information prior to October 1994 is not routinely available. Interested observers with data from this epoch, can send E-mail to help@stsci.edu.*

C.1.1 Observation Log File Contents (October 1994 version)

Observation logs created between October 1994 and August 1995 contain:

- *rootname*j.cmh: This ASCII header file contains the time interval, the rootname, averages of the pointing and spacecraft jitter, the guiding mode, guide star information, and alert or failure keywords. Figure C.1 shows a representative observation log header file.

- *rootname*j.cmj: This table presents the data at the highest time resolution for the telemetry mode in use. It contains the reconstructed pointing, guide star coordinates, derived jitter at the instrument aperture, and guiding-related flags. The intent is: (1) to provide high-time resolution jitter data for deconvolution or for assessing small aperture pointing stability, and (2) to display the slew and tracking anomaly flags with the highest resolution. Table C.2 lists the table column heading, units and a brief definition.

- *rootname*j.cmi: This table contains data that were averaged over three-second intervals. It includes the same information as the .cmj table and also includes orbital data (e.g., latitude, longitude, limb angle, magnetic field values, etc.) and instrument-specific items. It is best suited for a quick-look assessment of pointing stability and for studying trends in telescope or instrument performance with orbital environment. Table C.3 lists the table column heading, units and a brief definition.

- rootnamej_cmi/j/h.fits: The above three GEIS files are actually archived as FITS files. They may be worked with as such, or run through the STSDAS task **strfits**, to convert them.

- Observation Log File Contents (August 1995 version)

- The contents of observation log files created between August 1995 and February 1997 are as follows:

- *rootname*j.jih: This GEIS header file, the analog to the cmh file, contains the time interval, the rootname, averages of the pointing and spacecraft jitter, the guiding mode, guide star information, and alert or failure keywords. Figure C.1 shows a representative observation log header file.

- *rootname*j.jid: This GEIS image—a significant enhancement of the old cmj file—presents a two-dimensional histogram of the pointing fluctuations during the observation. You can display it to visualize the spacecraft stability during you observation, and is information for deconvolutions and PSF analyses.

- *rootname*j.jit: This table, the analog to the cmi table, contains data that were averaged over three-second intervals. Its content is identical (see Table C.3).

- *rootname*j_jif.fits: FITS file that is actually the de-archived product. This file can be converted to the jih/jid GEIS file via the **strftis** routine.

- *rootname*j_jit.fits: The de-archived FITS file corresponding to the jit IRAF table. It can be converted via **strfits**.

C.1.2 Observation Log File Contents (February 1997 version)

The contents of observation log files created since February 1997 are as follows:

- *rootname*j_jif.fits: The de-archived FITS file. Unlike the previous OMS epoch, this FITS file does not bundle a GEIS file and cannot be converted with **strfits**. This was done to more closely correlate the observation log files with the NICMOS, STIS, and ACS FITS files with extensions and associations. OMS will normally put all associated observation logs into a single file, to correspond to the associated science exposures. However, if even one science exposure is orphaned (not associated) then an individual observation log FITS file will be produced for every exposure in that association. For a description of NICMOS, STIS, and ACS association files, see appendix B. All of the information contained in the old cmh/jih ASCII header is now available as keywords in the FITS files.

- *rootname*j_jit.fits: The FITS file containing the table information. The comments for the _jif file apply here as well.

Table C.2: Contents of `.cmj` Table

Parameter	Units	Description
seconds	seconds	Time since window start
V2 dom	arcseconds	Dominant FGS V2 coordinate
V3 dom	arcseconds	Dominant FGS V3 coordinate
V2 roll	arcseconds	Roll FGS V2 coordinate
V3 roll	arcseconds	Roll FGS V3 coordinate
SI V2	arcseconds	Jitter at aperture reference
SI V3	arcseconds	Jitter at aperture reference
RA	degrees	Right ascension of aperture reference
DEC	degrees	Declination of aperture reference
Roll	degrees	Angle between North and +V3
DayNight	0,1 flag	Day (0) or night (1)
Recenter	0,1 flag	Recentering status
TakeData	0,1 flag	Vehicle guiding status
SlewFlag	0,1 flag	Vehicle slewing status

Figure C.1:A Representative `.jih` or `.cmh` Header

```
dSIMPLE  =                      F / data conforms to FITS standard          !
BITPIX   =                     32 / bits per data value                     !
DATATYPE= 'INTEGER*4       '     / datatype of the group array              !
NAXIS    =                      2 / number of data axes                     !
NAXIS1   =                     64 / length of the 1st data axis             !
NAXIS2   =                     64 / length of the 2nd data axis             !
GROUPS   =                      T / image is in group format                !
GCOUNT   =                      1 / number of groups                        !
PCOUNT   =                      0 / number of parameters                    !
PSIZE    =                      0 / bits in the parameter block             !
OMS_VER  = '16.2C           '     / OMS version used to process this observation
PROCTIME= '1994.133:06:24:18.35' / date-time OMS processed observation
                                  / date-times format (yyyy.ddd:hh:mm:ss.ss)

                                  / IMAGE PARAMETERS
CRVAL1   =                    0.0 / right ascension of zero-jitter pixel (deg)
CRVAL2   =                    0.0 / declination of zero-jitter pixel (deg)
CRPIX1   =                     32 / x-coordinate of zero-jitter pixel
CRPIX2   =                     32 / y-coordinate of zero-jitter pixel
CTYPE1   = 'RA---TAN        '     / first coordinate type
CTYPE2   = 'DEC--TAN        '     / second coordinate type
CD1_1    =                    0.0 / partial of ra w.r.t. x (deg/pixel)
CD1_2    =                    0.0 / partial of ra w.r.t. y (deg/pixel)
CD2_1    =                    0.0 / partial of dec w.r.t. x (deg/pixel)
CD2_2    =                    0.0 / partial of dec w.r.t. y (deg/pixel)
COORDSYS= 'WFPC2           '     / image coordinate system
XPIXINC =                    2.0 / plate scale along x (mas per pixel)
YPIXINC =                    2.0 / plate scale along y (mas per pixel)
PARITY  =                     -1 / parity between V2V3 frame and image frame
BETA1   =                 134.72 / angle from +V3 to image +x (toward +V2)
BETA2   =                 224.72 / angle from +V3 to image +y (toward +V2)

                                  / OBSERVATION DATA
PROPOSID=                  05233 / PEP proposal identifier
PROGRMID= '288             '     / program id (base 36)
OBSET_ID= '02              '     / observation set id
OBSERVTN= '03              '     / observation number (base 36)
TARGNAME= 'NGC3379-PO      '     / proposer's target name
STARTIME= '1994.133:06:24:18.35' / predicted observation window start time
ENDTIME = '1994.133:06:39:18.35' / predicted observation window end time
SOGSID  = 'U2880203        '     / SOGS observation name                    !

                                  / SCIENTIFIC INSTRUMENT DATA
CONFIG  = 'WFPC2           '     / proposed instrument configuration
PRIMARY = 'SINGLE          '     / single, parallel-primary, parallel-secondary
OPERATE = '1994.133:06:22:46.91' / predicted time instr. entered operate mode
TLMFORM = 'PN              '     / telemetry format
APERTURE= 'UWFALL          '     / aperture name
APER_V2 =                  1.565 / V2 aperture position in vehicle frame (arcsec)
APER_V3 =                  7.534 / V3 aperture position in vehicle frame (arcsec)

                                  / SPACECRAFT DATA
ALTITUDE=                 593.23 / average altitude during observation (km)
LOS_SUN =                 106.08 / minimum line of sight to Sun (deg)
LOS_MOON=                  77.11 / minimum line of sight to Moon (deg)
SHADOENT= '1994.133:05:11:29.00' / predicted Earth shadow last entry
SHADOEXT= '1994.133:05:42:45.00' / predicted Earth shadow last exit
LOS_SCV =                  12.46 / minimum line of sight to S/C veloc. (deg)
LOS_LIMB=                   58.0 / average line of sight to Earth limb (deg)

                                  / BACKGROUND LIGHT
ZODMOD  =                   22.3 / zodiacal light - model (V mag/arcsec2)
EARTHMOD=                   20.2 / peak Earth stray light - model (V mag/arcsec2)
MOONMOD =                   35.5 / moon stray light - model (V mag/arcsec2)
GALACTIC=                   -1.0 / diffuse galactic light - model (V mag/arcsec2)

                                  / POINTING CONTROL DATA
GUIDECMD= 'FINE LOCK       '     / commanded guiding mode
GUIDEACT= 'FINE LOCK       '     / actual guiding mode at end of GS acquisition
GSD_ID  = '0084900235      '     / dominant guide star id
GSD_RA  =              161.70720 / dominant guide star RA (deg)
GSD_DEC =               12.45407 / dominant guide star DEC (deg)
```

Figure C.2:Representative `.jih` or `.cmh` Header

```
GSD_MAG =                 12.867 / dominant guide star magnitude
GSR_ID  = '0085201189      ' / roll guide star id
GSR_RA  =             161.93314 / roll guide star RA (deg)
GSR_DEC =              12.78141 / roll guide star DEC (deg)
GSR_MAG =                12.977 / roll guide star magnitude
GSACQ   = '1994.133:06:31:02.92' / actual time of GS acquisition completion
PREDGSEP=              1420.775 / predicted guide star separation (arcsec)
ACTGSSEP=              1421.135 / actual guide star separation (arcsec)
GSSEPRMS=                   3.8 / RMS of guide star separation (milli-arcsec)
NLOSSES =                     0 / number of loss of lock events
LOCKLOSS=                   0.0 / total loss of lock time (sec)
NRECENT =                     0 / number of recentering events
RECENTR =                   0.0 / total recentering time (sec)

                                / LINE OF SIGHT JITTER SUMMARY
V2_RMS  =                   4.5 / V2 axis RMS (milli-arcsec)
V2_P2P  =                  51.6 / V2 axis peak to peak (milli-arcsec)
V3_RMS  =                  20.9 / V3 axis RMS (milli-arcsec)
V3_P2P  =                 267.3 / V3 axis peak to peak (milli-arcsec)
RA_AVG  =             161.85226 / average RA (deg)
DEC_AVG =              12.58265 / average dec (deg)
ROLL_AVG=             293.01558 / average roll (deg)

                                / PROBLEM FLAGS, WARNINGS and STATUS MESSAGES
                                / (present only if problem exists)
ACQ2FAIL= '              T' / target acquisition failure
GSFAIL  = 'DEGRADED        ' / guide star acquisition failure  (*1)
TAPEDROP= '              T' / possible loss of science data
TLM_PROB= '               ' / problem with the engineering telemetry
TM_GAP  =                404.60 / duration of missing telemetry (sec)
SLEWING = '              T' / slewing occurred during this observation
TAKEDATA= '              F' / take data flag NOT on throughout observation
SIPROBnn= '               ' / problem with specified science instrument (*2)

END
--------------------------------------------------------------------------------

notes
*1 - GSFAIL appears only once in a single header file.
     The following table lists all current possible values for the GSFAIL
     keyword:
     --------------------------------------------------------------
     GSFAIL   |DEGRADED        | / guide star acquisition failure
              |IN PROGR        | / guide star acquisition failure
              |SSLEXP          | / guide star acquisition failure
              |SSLEXS          | / guide star acquisition failure
              |NOLOCK          | / guide star acquisition failure
              |                |
              |SREXCS?         | / guide star acquisition failure
              |SREXCS1         | / guide star acquisition failure
              |SREXCS2         | / guide star acquisition failure
              |SREXCS3         | / guide star acquisition failure
              |                |
              |SREXCP?         | / guide star acquisition failure
              |SREXCP1         | / guide star acquisition failure
              |SREXCP2         | / guide star acquisition failure
              |SREXCP3         | / guide star acquisition failure
              |                |
              |UNKNOWN         | / guide star acquisition failure
              |VEHSAFE         | / guide star acquisition failure
     --------------------------------------------------------------

*2 - The SIPROBnn keywords appear in the header file with nn = 01 - 99.
     The following table lists all current possible values for the SIPROBnn
     keyword:
     ----------------------------------------------------------------------
     SIPROBnn |DCF_NUM unchanged| / This observation may not have been taken
              |FOS Safing!      | / This observation affected when FOS Safed!
              |HRS Safing!      | / This observation affected when HRS Safed!
              |WFII Safing!     | / This observation affected when WFII Safed!
              |FOC Safing!      | / This observation affected when FOC Safed!
              |Shut             | / FOS aperture door is not Open!
              |FAILED           | / FGS astrometry target acquisition failed
     ----------------------------------------------------------------------
```

Table C.3: Contents of `.jit or.cmi`Table, Three-Second Averaging

Parameter	Units	Description
seconds	seconds	Time since window start
V2 dom	arcseconds	Dominant FGS V2 coordinate
V3 dom	arcseconds	Dominant FGS V3 coordinate
V2 roll	arcseconds	Roll FGS V2 coordinate
V3 roll	arcseconds	Roll FGS V3 coordinate
SI V2 AVG	arcseconds	Mean jitter in 3 seconds
SI V2 RMS	arcseconds	rms jitter in 3 seconds
SI V2 P2P	arcseconds	Peak jitter in 3 seconds
SI V3 AVG	arcseconds	Mean jitter in 3 seconds
SI V3 RMS	arcseconds	rms jitter in 3 seconds
SI V3 P2P	arcseconds	Peak jitter in 3 seconds
RA	degrees	Right ascension of aperture reference
DEC	degrees	Declination of aperture reference
Roll	degrees	Angle between North and +V3
LimbAng	degrees	Angle between earth limb and target
TermAng	degrees	Angle between terminator and target
LOS_Zenith	degrees	Angle between HST zenith and target
Latitude	degrees	HST subpoint latitude
Longitude	degrees	HST subpoint longitude
Mag V1,V2,V3	degrees	Magnetic field along V1, V2, V3
EarthMod	V Mag/arcsec^2	Model earth background light
SI_Specific	–	Special science instrument data
DayNight	0,1 flag	Day (0) or night (1)
Recenter	0,1 flag	Recentering status
TakeData	0,1 flag	Vehicle guiding status
SlewFlag	0,1 flag	Vehicle slewing status

C.1.3 Jitter File Contents (February 2003 Version)

The current format of the jitter files produced by the EDPS are similar to the Observation Logs (Obslogs) produced by the Observation Monitoring System (OMS) that no longer exist. The EDPS jitter files are limited to the engineering data that describes the performance of the Pointing Control System (PCS) including the Fine Guidance Sensors that are used to control the vehicle pointing. The jitter files report on PCS engineering data for the duration of the observation. The old name for these jitter files, observation logs (Obslogs), is now inappropriate, because of the absence of instrument monitoring. However the FITS header and extension formats of the jitter files are not radically different from the original Obslogs. FITS-based tools that access only the PCS related data should be able to handle both the old Obslogs and the new jitter files. The jitter files retain the same file naming conventions as the old Obslogs.

One way to distinguish the new jitter files from the old Obslogs files is to check the existence of the OPUS_VER keyword (system version id) that replaced the OMS_VER keyword. The EDPS software is part of the larger OPUS system that also supports the Science Data Processing System (SDPS) and the On-the-Fly Reprocessing (OTFR) system. Most of the format changes involve deletions of keywords. Six keywords have been added. There are some improvements in the new files; the accuracy of HST orbit-related statistics has been improved by ignoring the values in the telemetry that must be interpolated. Instead, an orbit model is now used to recalculate the orbit position and velocity at the exact time needed. The new jitter files have been generated since mid-February, 2003, but there is no specific date since the OMS ran in parallel with EDPS for about a week.

The old format is well documented by the Observatory Support Web page maintained by the Telescopes Branch of the Instrument Division at

http://www.stsci.edu/instruments/observatory/

The differences between the new jitter files and the old observation log files are briefly described below. This jitter file information is supplemental to the Observation Log Documentation. For further details on the differences, along with sample file headers, see the following Web page that describes these files at length:

http://www.ess.stsci.edu/gsd/dst/edps/jitter_format.html

Changes in Image File

There are six new keywords for the jitter image file that has the file extension ".jif" or ".jwf". Keywords that are in the primary header are assigned the extension (Ext) value "0". Keywords in other extensions have the extension value ">0" since there can be more than one extension for associated products.

Keyword	Ext	Comment
ASN_PTYP	0	Association Product Type
GSD_FGS	0	1, 2 or 3: FGS ID of dominant FGS
GSR_FGS	0	1,2 or 3: FGS ID of subdominant (roll) FGS
GS_PRIM	0	DOMINANT or ROLL: GS that was acquired first
OPUS_VER	0	Build version: replaces OMS_VER
WCSAXES	>0	2: Number of axes for WCS parameters

There were 43 OMS Obslog keywords that were dropped for the new jitter image file.

Changes in Table File

The jitter table file that has the file extension ".jit" or ".jwt" has very few changes. There are still 28 columns of data in the table. Two of the columns have been replaced. Column 23 has changed from EarthMag, that was an estimate of stray light from the Earth based on a model that is no longer support by EDPS, to BrightLimb, which is a short integer value, where 1 indicated that the limb position closest to the target is bright, and 0 indicates it is dark. Column 24 has changed from SI_Specific, that is not supported by EDPS, to FGS_flags, that provides status data for all three FGS. In addition, the comment for the DayNight flag has been corrected. The day value is 1 and the night value is 0. These changes affected the values of the keywords: TTYPE23, TUNIT23, TFORM23, TUNIT24, TTYPE24, and TUNIT24. The value of keyword NAXIS1 changed from 116 to 114 because the size of column 23 changed from a four-byte floating value to a two-byte short integer value. The keyword ASN_PTYP was added to the association keywords. It has the same value as that keyword in the image file header.

Changes in Association File Structure

In the old Obslogs, the jitter files for associated exposures were always collected into files with the same filename extension, ".jit", ".jwt", ".jif", or ".jwf", as the member exposures. The file name has the association rootname that is found in keyword ASN_ID. Each member exposure provides a FITS extension for a table or image, having the keyword ASN_MTYPE that describes the type of member. The primary header of the association image file has the following keywords updated to reflect statistics for the set of FITS extensions: TSTRTIME, TENDTIME, MGSSPRMS, TNLOSSES, TLOCKLOS, TNREENT, TRECENTR, TV2_RMS, MV2_P2P, TV3_RMS, MV3_P2P, TRA_AVG, TROLLAVG,

T_GDACT, T_ACTGSP, T_GSFAIL, T_SGSTAR, T_TLMPRB, T_NOTLM, T_NTMGAP, T_GSGAP, T_SLEWING, and T_TFDDWN.

For STIS associations, the wavecal members are separated from the science members by different file extensions: ".jwt" and ".jwf". The structure for STIS associations has not changed. But for NICMOS and ACS, the file rootnames now match the rootnames of the association products. Only the members that are used to generate the association product are collected into FITS extensions for that product. The old Obslogs used only the association rootname. The new jitter-file associations may have multiple rootnames, ending in 0, 1, 2, etc. The statistical keywords in the primary header listed above are calculated for the subset of members collected for the specific association product. The ACS associations reserve the rootname of the ASN_ID for dithered data combined from multiple positions. At each position, there is an association product having the last character of its rootname being a sequence starting from 1. If there is no dithered product for ACS, the associated exposures at the single position have a product with a rootname ending in 1.

The Data Systems Branch maintains a Web page that allows access to detailed information on the most recent header formats and keyword definitions. This Web page can be accessed from the HST Keyword Dictionary entry found on the DSB home page at

http://www.ess.stsci.edu/gsd/dst/

C.2 Retrieving Observation Logs

You can retrieve observation log files for data taken after October 20, 1994 from the HST Archive using StarView as described in chapter 1 of the HST Introduction. Unlike science data, which generally has a one-year proprietary period, observation log files become public as soon as they are archived.

The easiest way to get OMS files through StarView is to identify the observation of interest and proceed with your request as described in chapter 1 of the HST Introduction, until you reach the "HST Retrieval Configurations Options" screen, reproduced in Figure C.3. You can then check the Observation Log Files box, along with any other desired boxes, and continue with your request. StarView will then deliver the associated observation log files.

For observations logged between October 1994 to August 1995, you will be delivered the `cmi`, `cmj`, and `cmh` files in FITS form (e.g., `_cmi.fits`). Observations archived from August 1995 to February 1997 will return `_jif.fits` and `_jit.fits` files. These, and the earlier FITS files can be worked with as such, or converted to their GEIS counterparts via the STSDAS **strfits** task. However, as of February 1997, the

`_jif.fits` and `_jit.fits` files are standard FITS files with extensions and cannot be converted to GEIS.

Figure C.3:Choosing Observation Log Files in StarView
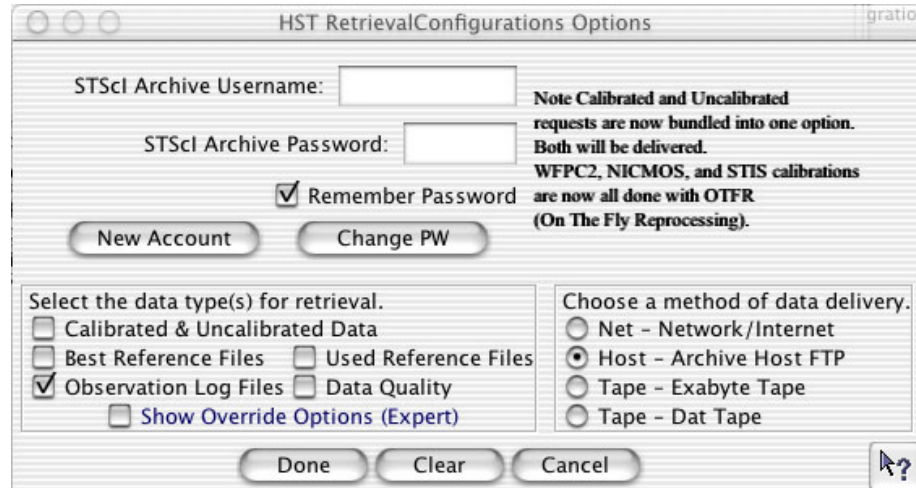
C.3 Using Observation Logs

Here are some simple examples of what can be learned from the observation log files. Note that for FITS format observation logs, current versions of STSDAS tools will handle the files with extensions properly. Keywords can be viewed with tools such as **imheader** or **hedit**, and data viewed, plotted, or displayed using the same tasks one might have for the GEIS files. For more information on FITS file structures, see chapter 2 of the HST Introduction.

C.3.1 Guiding Mode

Unless requested, all observations will be scheduled with FINE LOCK guiding, which may be one or two guide stars (dominant and roll). The spacecraft may roll slightly during an observation if only one guide star is acquired. The amount of roll depends upon the gyro drift at the time of the observation, the location during an orbit, and the lever arm from the guide star to the center of the aperture.

There are three commanded guiding modes: FINE LOCK, FINE LOCK/GYRO, and GYRO. OMS header keywords GUIDECMD (commanded guiding mode) and GUIDEACT (actual guiding mode) will usually agree. If there was a problem, they won't agree and the GUIDEACT value will be the guiding method actually used during the

exposure. If the acquisition of the second guide star fails, the spacecraft guidance, GUIDEACT, may drop from FINE LOCK to FINE LOCK/GYRO, or even to GYRO, which may result in a target rolling out of an aperture. Check the OMS header keywords to verify that there was no change in the requested guiding mode during the observation.

Until new flight software (version FSW 9.6) came online in September 1995, if the guide star acquisition failed, the guiding dropped to COARSE track. After September 1995, if the guide star acquisition failed, the tracking did not drop to COARSE track. Archival research- ers may find older datasets that were obtained with COARSE track guiding.

The dominant and roll guide star keywords (GSD and GSR) in the OMS header can be checked to verify that two guide stars were used for guiding, or in the case of an acquisition failure, to identify the suspect guide star. The dominant and roll guide star keywords identify the stars that were scheduled to be used, and in the event of an acquisition failure, may not be the stars that were actually used. The following list of cmh keywords is an example of two star guiding.

```
GSD_ID  = '0853601369        ' / Dominant Guide Star ID
GSD_RA  =            102.42595 / Dominant Guide Star RA (deg)
GSD_DEC =            -53.41362 / Dominant Guide Star DEC (deg)
GSD_MAG =              11.251 / Dominant Guide Star Magnitude
GSR_ID  = '0853602072        ' / Roll Guide Star ID
GSR_RA  =            102.10903 / Roll Guide Star RA (deg)
GSR_DEC =            -53.77683 / Roll Guide Star DEC (deg)
GSR_MAG =              12.426 / Roll Guide Star Magnitude
```

If you suspect that a target has rolled out of the aperture during an exposure, you can quickly check the counts in each group of the raw science data. As an example, the following IRAF commands can be used to determine the counts in each group.

```
cl> grlist z2o4040dt.d0h 1-24 > groups.lis
cl> imstat @groups.lis
```

Some observations can span several orbits. If during a multiple orbit observation the guide star reacquisition fails, the observation may be terminated with possible loss of observing time, or switch to other less

desirable guiding modes. The GSACQ keyword in the `cmh` header will state the time of the last successful guide star acquisition.

```
GSACQ   = '136:14:10:37.43   ' / Actual time of GS Acquisition Completion
```

C.3.2 Guide Star Acquisition Failure

The guide star acquisition at the start of the observation set could fail if the FGS fails to lock onto the guide star. The target may not be in the aperture, or maybe only a piece of an extended target is in the aperture. The jitter values will be increased because FINE LOCK was not used. The following list of `cmh` header keywords indicate that the guide star acquisition failed.

```
V3_RMS =                    19.3 / V3 Axis RMS (milli-arcsec)
V3_P2P =                   135.7 / V3 Axis peak to peak (milli-arcsec)
GSFAIL = '          DEGRADED' / Guide star acquisition failure!
```

The observation logs for all of the following observations in the observation set will have the "DEGRADED" guide star message. This is not a Loss of Lock situation but an actual failure to acquire the guide star in the desired guiding mode. For the example above, the guiding mode dropped from FINE LOCK to COARSE TRACK.

```
GUIDECMD= 'FINE LOCK          ' / Commanded Guiding mode
GUIDEACT= 'COARSE TRACK       ' / Actual Guiding mode at end of GS acquisition
```

If the observational dataset spans multiple orbits, the guide star will be re-acquired, but the guiding mode will not change from COARSE TRACK. In September 1995, the flight software was changed so that COARSE TRACK is no longer an option. The guiding mode drops from two guide star FINE LOCK to one guide star FINE LOCK, or to GYRO control.

C.3.3 Moving Targets and Spatial Scans

A type 51 slew is used to track moving targets (planets, satellites, asteroids, and comets). Observations are scheduled with FINE LOCK acquisition, i.e., with two or one guide stars. Usually, a guide star pair will stay within the pickle during the entire observation set, but if two guide stars are not available, a single guide star may be used, assuming the drift is small or the proposer says that the roll is not important for that particular observing program. An option during scheduling is to drop from FGS control to GYRO control when the guide stars move out of the FGS. Also, guide star handoffs (which are not a simple dropping of the guide stars to

GYRO control) will affect the guiding and may be noticeable when the jitter ball is plotted.

The jitter statistics are accumulated at the start of the observation window. Moving targets and spatial scan motion will be seen in the jitter data and image. Therefore, the OMS header keywords V2_RMS and V3_RMS values (the root mean square of the jitter about the V2 and V3 axis) can be quite large for moving targets. Also, a special anomaly keyword (SLEWING) will be appended to the OMS header stating movement of the telescope during the observation. This is expected for observing moving targets. The following list of .cmh header keywords is an example of expected values while tracking a moving target.

```
                        / LINE OF SIGHT JITTER SUMMARY
V2_RMS   =           3.2 / V2 Axis RMS (milli-arcsec)
V2_P2P   =          17.3 / V2 Axis peak to peak (milli-arcsec)
V3_RMS   =          14.3 / V3 Axis RMS (milli-arcsec)
V3_P2P   =          53.6 / V3 Axis peak to peak (milli-arcsec)
RA_AVG   =    244.01757 / Average RA (deg)
DEC_AVG  =    -20.63654 / Average DEC (deg)
ROLL_AVG=     280.52591 / Average Roll (deg)
SLEWING  = '          T' / Slewing occurred during this observation
```

C.3.4 High Jitter

The spacecraft may shake during an observation, even though the guiding mode is FINE LOCK. This movement may be due to a micro-meteorite hit, jitter at a day-night transition, or for some other unknown reasons. The FGS is quite stable and will track a guide star even during substantial spacecraft motion. The target may move about in an aperture, but the FGS will continue to track guide stars and reposition the target into the aperture. For most observations, the movement about the aperture during a spacecraft excursion will be quite small, but sometimes, especially for observations with the spectrographs, the aperture may move enough that the measured flux for the target will be less than a previous group. Check the OMS header keywords (V2_RMS, V3_RMS) for the root mean square of the jitter about the V2 and V3 axis. The following list of .cmh header keywords is an example of typical guiding rms values.

```
                        / LINE OF SIGHT JITTER SUMMARY
V2_RMS   =           2.6 / V2 Axis RMS (milli-arcsec)
V2_P2P   =          23.8 / V2 Axis peak to peak (milli-arcsec)
V3_RMS   =           2.5 / V3 Axis RMS (milli-arcsec)
V3_P2P   =          32.3 / V3 Axis peak to peak (milli-arcsec)
```

Recentering events occur when the spacecraft software decides that shaking is too severe to maintain lock. The FGS will release guide star control and within a few seconds reacquire the guide stars. It is assumed the guide stars are still within the FGS field of view. During the recentering

time, INDEF will be written to the OMS table. Recentering events are tracked in the OMS header file.

Be careful when interpreting "Loss of Lock" and "Recentering" events that occur at the very beginning or at the end of the OMS window. The OMS window is larger than the observation window. These events might not affect the observation since the observation start time will occur after the guide stars are acquired (or re-acquired), and the observation stop time may occur before the "Loss of Lock" or "Recentering" event that occurred at the end of an OMS window.

The **sgraph** commend in the **stsdas.graphics.stplot** package will plot time vs. jitter along the direction of HST's V2 axis (see Figure C.4):

```
cl> sgraph "x3y60102j_jit.fits seconds si_v2_avg"
```

Figure C.4:Plotting Jitter Along V3 Axis
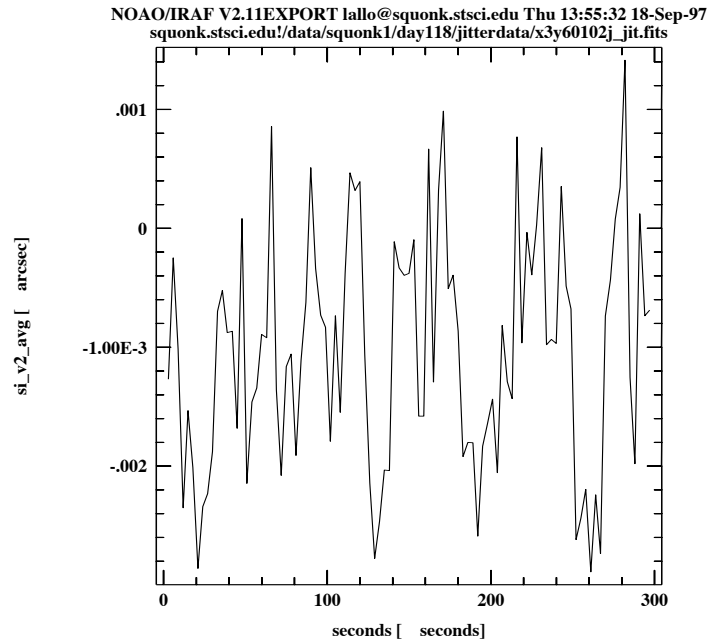
To get an idea of pointing stability, you can create a *jitter ball* by plotting jitter along the V2 axis vs. jitter along the V3 axis (see Figure C.5):

```
st> sgraph "x3660102j_jit.fits si_v2_avg si_v3_avg"
```

Figure C.5:Plotting V2 vs. V3 Jitter

NOAO/IRAF V2.11EXPORT lallo@squonk.stsci.edu Thu 13:52:50 18-Sep-97
squonk.stsci.edu!/data/squonk1/day118/jitterdata/x3y60102j_jit.fits
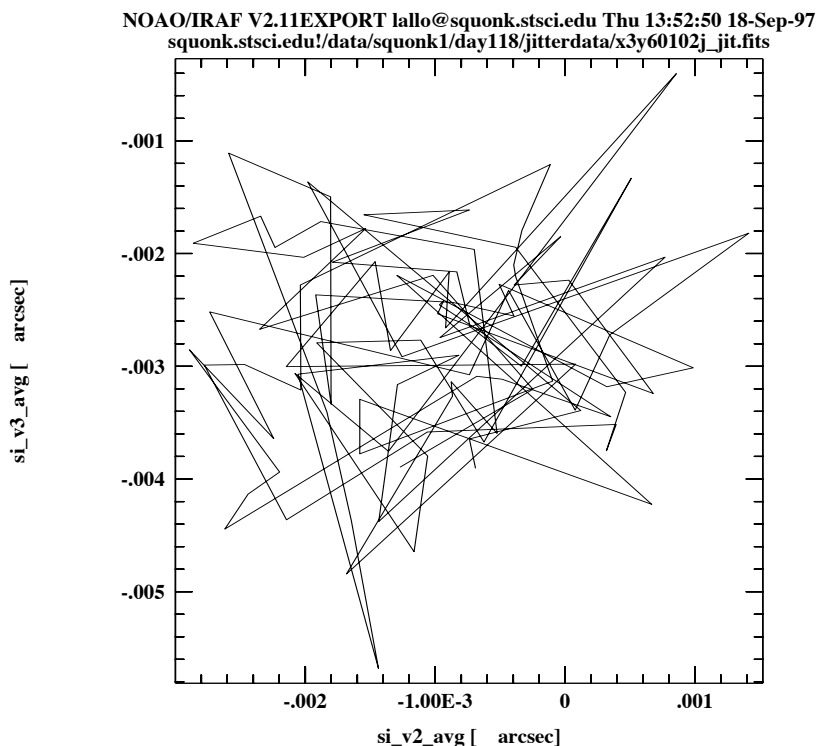
The **tstatistics** task can be used to find the mean value of the
`si_v3_avg` column—the amount of jitter (in arcseconds) in the direction
of the V3. This value can be used to model jitter in a PSF. In this example,
the mean jitter is ~3 mas, which is typical for post-servicing mission data:

Figure C.6:Averaging a Column with tstatistics

```
tt> tstat u26m0801j.cmi si_v3_avg
# u26m0801j.cmi  si_v3_avg
#
nrows           mean        stddev        median           min          max
   11   -0.003006443888   0.00362533   -7.17163E-4   -0.00929515   0.00470988
```

*Understanding and interpreting the meaning of the table columns and
header keywords is critical to understanding the observation logs.
Please read the available documentation and contact the STScI Help
Desk (help@stsci.edu.) if you have any questions about the files. Docu-
mentation is available via the Web at: http://www.stsci.edu/hst/observa-
tory/documents*

Index