Version 1.0
March 2009

HST Data Handbook for COS

SPACE TELESCOPE SCIENCE INSTITUTE

Space Telescope Science Institute
3700 San Martin Drive
Baltimore, Maryland 21218
help@stsci.edu

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

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 COS World Wide Web site:

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

COS Revision History

Version	Date	Editor
1.0	March 2009	Brittany Shaw, Derck Massa, and Mary Elizabeth Kaiser

Authorship

This document is written and maintained by the COS/STIS Team in the Instruments Division of STScI with the assistance of associates in the Operations and Engineering Division. Contributions to the current edition were made by T. Ake, A. Aloisi, P. Ghavamian, P. Hodge, M.E. Kaiser, C. Keyes, D. Massa, C. Oliveira, R. Osten, D. Sahnow, B. Shaw, D. Soderblom, and B. York.

In publications, refer to this document as: Shaw, B. et al. 2009, "COS Data Handbook", Version 1.0, (Baltimore: STScI).

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

Table of Contents

Preface

The HST Data Handbook is composed 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 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, and now includes PyRAF, the focus of this handbook will be on use of IRAF/PyRAF, Space Telescope Science Data Analysis System (STSDAS), and TABLES.

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 the use of almost all IRAF tools. It is briefly discussed in Chapter 3, but for the most part, the IRAF examples given will work exactly the same in PyRAF.

The specifics of the data produced by the ACS, COS, FGS, NICMOS, STIS, WFC3, and WFPC2 (to be decommissioned during Servicing Mission 4) instruments are described in separate versions of Part II. The general information in Parts I and III, referred to as "the introductory chapters," are appended to the beginning and end of each instrument-specific part. We recommend a careful reading of the introductory chapters before proceeding to the instrument-specific section and before starting to work on your HST data.

The present introductory chapters are based on information available as of December 2008. Changes in the HST Data Archive and data reduction software since then are not included in this revision of the HST 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 the Hubble Legacy Archive (HLA), 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.

PART I:

Introduction to Reducing HST Data

In this part. . .

The chapters in this part provide an introduction to the process of retrieving and reducing Hubble Space Telescope (HST) data.

CHAPTER 1:

Obtaining HST Data

In this chapter...

1.1 Archive Overview

All Hubble Space Telescope (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 one of the following ways:

- Via ftp/sftp, either directly from the HDA, or by accessing a staging disk set up upon request.
- Via data storage media (CDs or DVDs) which are shipped to the user.
- Via the use of external hard drives loaned by STScI: Users requesting large amounts of data (>100GB) over a short period of time may request the data be written to hard disk. The disk will be delivered to the user for reading, and then the disk must be returned. Use of this retrieval method must be coordinated through the MAST Help Desk (archive@stsci.edu). The number of external hard drives is limited, so users are strongly encouraged to make arrangements in advance. For reference, a DVD can hold 4.7 GB.

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

All datasets retrieved from the HDA, regardless of the method used, will be in FITS (Flexible Image Transport System) format. Further information on HST file formats is presented in Chapter 2.

Non-proprietary data in the HDA can be retrieved electronically either by registered HDA users or via anonymous login. Proprietary data may not be retrieved except by a registered HDA user who has the permission of the program's Principal Investigator (PI). Note that HST PIs are ***not*** automatically registered. PIs should register before their first observations have been taken. All calibration observations as well as observations made as part of the GO Parallel programs are immediately public. All observations made as part a Treasury Program will either be immediately public or have only a brief proprietary period. The High-Level Science Products (HLSP) section of MAST also contains several sets of publicly available and fully reduced HST data such as the Ultra Deep Field and the GEMS survey data. See http://archive.stsci.edu/hlsp/index.html for a complete listing.

The archive recommends to ask for compressed data, which distinctly shortens the retrieval times without any information loss.

Advanced Camera for Surveys (ACS), Cosmic Origins Spectrograph (COS), and Wide Field Camera 3 (WFC3) users should note that the preferred option for data retrieval is from the HDA staging disk, via ftp/sftp. Users retrieving large numbers of ACS, COS, or WFC3 files should also consider requesting them on DVDs.

1.1.1 Archive Processing

The HDA contains all observations ever made by HST and a catalog that describes the data. Each time a user makes a data request, the HDA delivers data which are either processed with the On-The-Fly-Reprocessing (OTFR) system or simply retrieved from a set of final calibrated and statically stored data, as listed in Table 1.1.

Table 1.1: HST Instrument Data Processing.

Instrument	HDA Processing
ACS	OTFR
COS	OTFR
FGS	Static
FOC	Static
FOS	Static
GHRS	Static
HSP	Static
NICMOS	OTFR
STIS	Static & OTFR[1]
WFC3	OTFR
WF/PC-1	Static
WFPC2	Static[1]

1. STIS and WFPC2 data taken before SM4 will not be processed with OTFR but will instead be fully calibrated and statically stored in the HDA.

The OTFR system reconstructs and calibrates the data at the time of retrieval. This allows users to obtain data calibrated with up-to-date reference files, parameters, and software. OTFR makes use of the OSS/PODPS Unified System (OPUS) system to orchestrate the processing of the data. OPUS retrieves the HST telemetry data (POD files) from the Data Archiving and Distribution System (DADS) and creates raw data files in a step called Generic Conversion. Calibration reference files specific to the different modes of HST operations are prepared and archived in the Calibration Data Base System (CDBS). During Generic Conversion, CDBS is queried to determine which reference files apply to the specific observation being processed. OPUS then calibrates the data using the specific instrument's calibration software and reference files. After calibration, the files are returned to DADS for distribution to the user, as calibrated and/or raw files.

All STIS data collected before its failure in August 2004 have been fully reprocessed and calibrated, and are now stored in a static archive. (STIS data taken after SM4 will be processed via OTFR.) A similar effort is currently underway to reprocess and calibrate WFPC2 data. All data taken with WFPC2 until its decommissioning will be reprocessed and permanently stored in a static archive.

Data for FOC, FOS, GHRS, HSP, and WF/PC-1 do not pass through OTFR or any pipeline. For FOC, FOS, and GHRS, final calibrated archives have been produced, since no further improvements in the calibration for these instruments are expected. The user is provided with a copy of the raw and final calibrated data from the static (final) archive once a request is made. For HSP and WF/PC-1, no reprocessing or recalibration has been done nor is any planned. Once raw data from these instruments are retrieved from the HDA, they need to be calibrated locally by the users.

Searches and retrievals are available through the MAST Web site, http://archive.stsci.edu. The MAST Web site allows cross correlation of HDA searches with other MAST mission archives. It offers simple previews of HST datasets when available, as well as links to references citing a given dataset using the Astrophysics Data System (ADS). The MAST Web site is discussed in more detail in Section 1.2.

1.1.2 Archive Registration

The simplest way to register for HST data is to complete the form on the Web page at: http://archive.stsci.edu/registration. If problems occur, registration requests may also be sent to the HDA Help Desk, at archive@stsci.edu. The PI of each HST proposal must request access (i.e., authorization) to their proprietary data for themselves, and for anyone else whom the PI wants to have access to them. 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.

1.1.3 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/manual. 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 message to archive_news-request@stsci.edu with the 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 Obtaining Data via the MAST Web

HDA datasets can be searched for, previewed, and retrieved via the World Wide Web. The starting point for Web-based searches of the HDA is the Multimission Archive at STScI (MAST) Web site at:

`http://archive.stsci.edu`[2]

1.2.1 MAST Overview

The MAST home page is shown in Figure 1.1. 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 on the MAST home page from the "Missions" pull-down menu. Searches of the HDA by object class can also be made with the VizieR Catalog Search tool at:

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

The HST section of MAST offers tutorials about the HDA, as well as news and a Frequently Asked Questions page. 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 "Search and Retrieval" menu in the HST section brings up the page shown in Figure 1.2. Here the user may query on several search parameters, such as Object Name, Instrument, and Proposal ID. Once these are entered, clicking the "Search" button returns a page listing the datasets found. An example search results page is shown in Figure 1.3. More information about an individual dataset, including a preview (for most datasets), are also available by clicking on the dataset name (Figure 1.4).

Datasets can be selectively marked for retrieval in the search results page. After marking datasets for retrieval, press the button labeled "Submit marked data for retrieval from STDADS". This brings you to the Retrieval Options page, shown in Figure 1.5. Here you may select which files (calibrated, uncalibrated, etc.) you would like retrieved, and where you would like them delivered.

---

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.

If you are retrieving proprietary data, you will need an archive account. Please refer to Section 1.1.1 for information on how to request one.

Non-proprietary data may be retrieved with or without an archive account. To retrieve non-proprietary data without an archive account, type “anonymous” in the “Archive Username” field, and your email address in the password field. (The email address is needed so that we can notify you when the retrieval request is finished.)

Options for data delivery include direct ftp and sftp, staging, and CD or DVD. If ftp/sftp delivery is specified, you will be need to provide the name of the computer and directory to which the files are to be delivered, as well as your username and password on that computer. (Retrieval requests are encrypted, so there is no danger of your login information being stolen.)

Shortly after submitting the request, you will receive an e-mail message acknowledging its receipt. Another message will be sent after all the requested files have been transferred. The status of the request, including 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.

Datasets retrieved to the staging disk using an Archive username and password may be accessed through ftp from archive.stsci.edu using this username and password. (Data that you stage this way will only be visible to you. Therefore, proprietary data as well as non-proprietary data may be safely staged.) Datasets that were retrieved as “anonymous” can be accessed using either an account username and password or through regular anonymous ftp.

Figure 1.1: MAST home page.

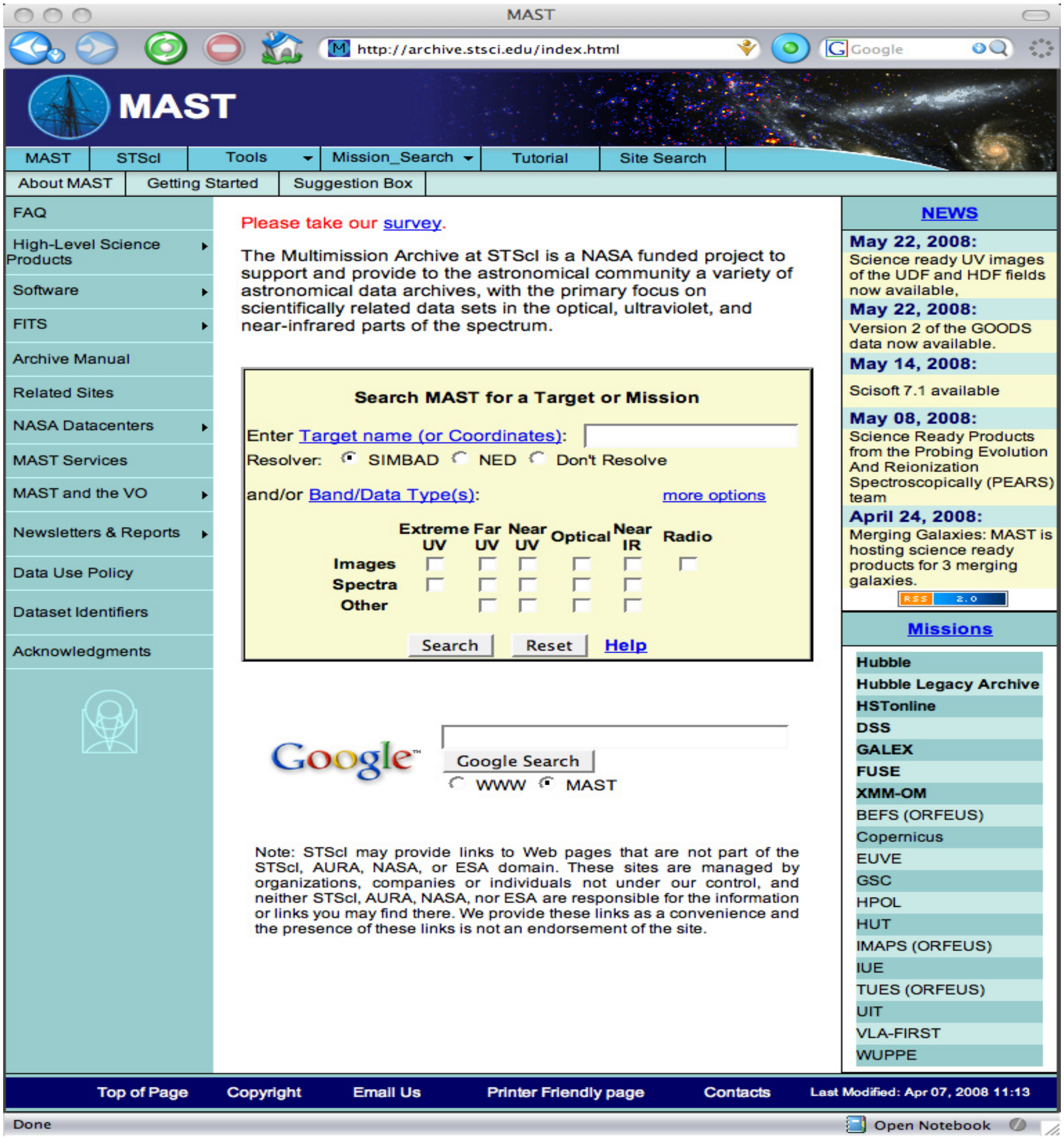


Figure 1.2: HST Archive Web search form.

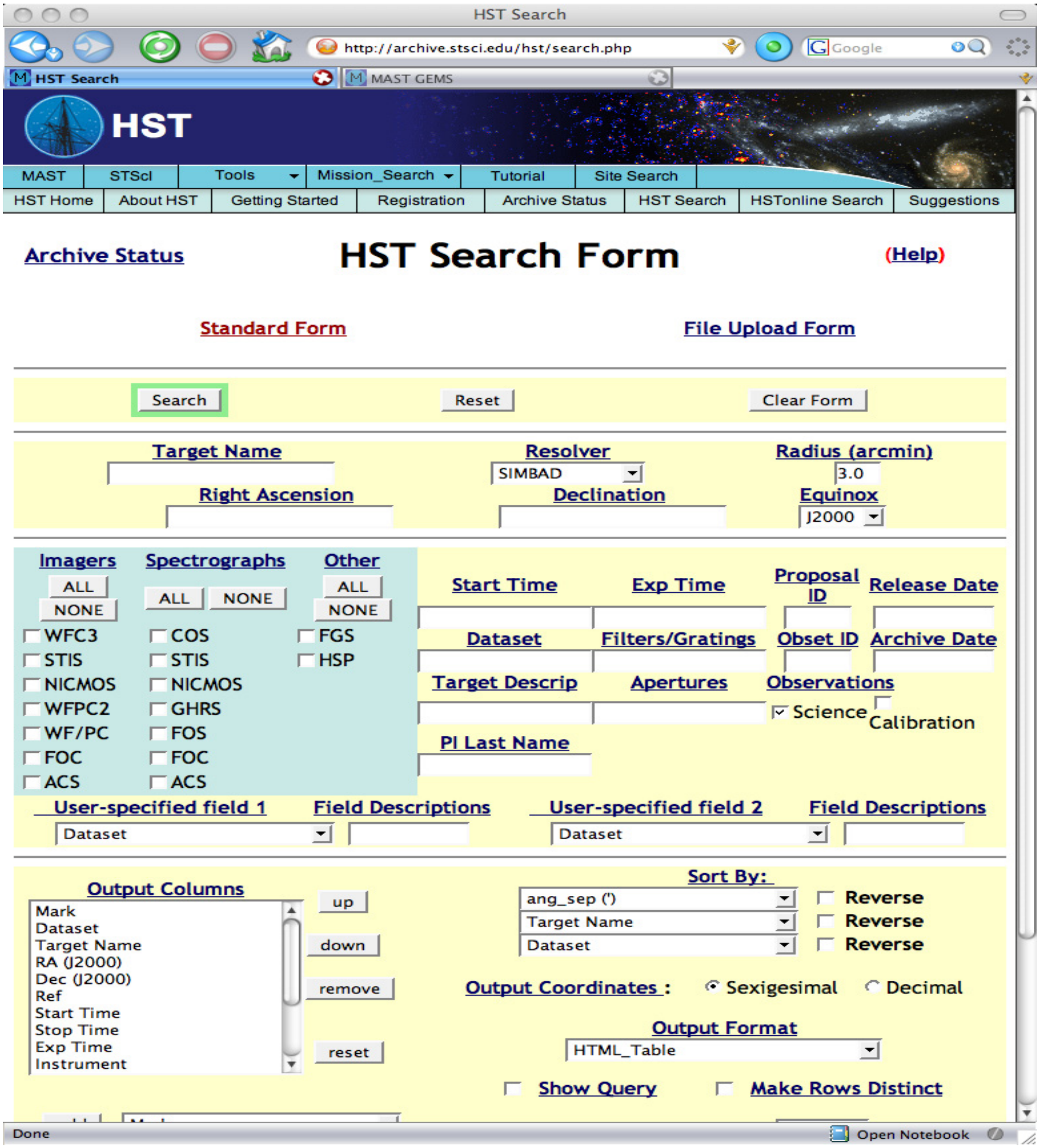
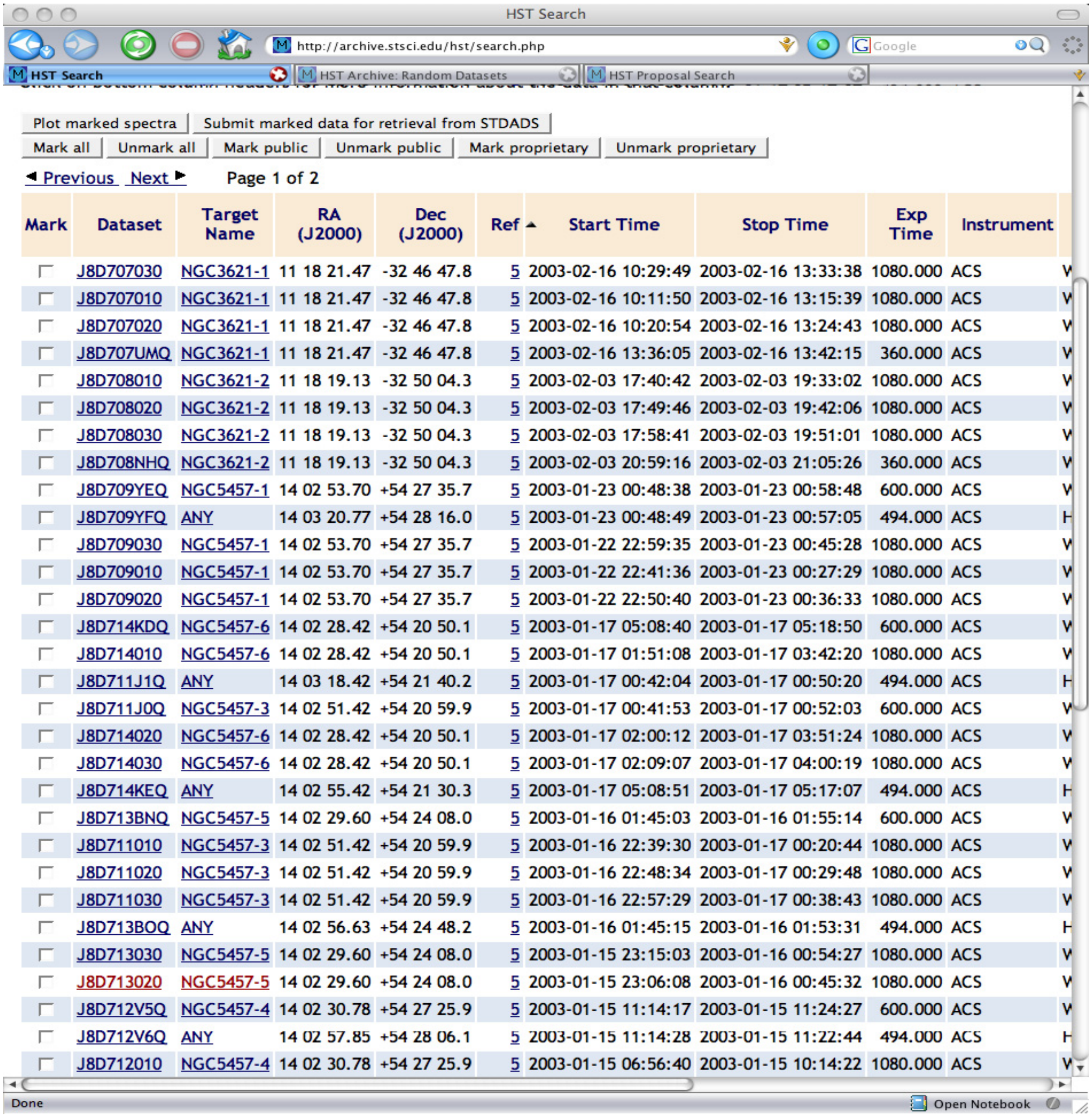


Figure 1.3: HST Archive search results page..



Mark	Dataset	Target Name	RA (J2000)	Dec (J2000)	Ref ▲	Start Time	Stop Time	Exp Time	Instrument	
☐	J8D707030	NGC3621-1	11 18 21.47	-32 46 47.8	5	2003-02-16 10:29:49	2003-02-16 13:33:38	1080.000	ACS	W
☐	J8D707010	NGC3621-1	11 18 21.47	-32 46 47.8	5	2003-02-16 10:11:50	2003-02-16 13:15:39	1080.000	ACS	W
☐	J8D707020	NGC3621-1	11 18 21.47	-32 46 47.8	5	2003-02-16 10:20:54	2003-02-16 13:24:43	1080.000	ACS	W
☐	J8D707UMQ	NGC3621-1	11 18 21.47	-32 46 47.8	5	2003-02-16 13:36:05	2003-02-16 13:42:15	360.000	ACS	W
☐	J8D708010	NGC3621-2	11 18 19.13	-32 50 04.3	5	2003-02-03 17:40:42	2003-02-03 19:33:02	1080.000	ACS	W
☐	J8D708020	NGC3621-2	11 18 19.13	-32 50 04.3	5	2003-02-03 17:49:46	2003-02-03 19:42:06	1080.000	ACS	W
☐	J8D708030	NGC3621-2	11 18 19.13	-32 50 04.3	5	2003-02-03 17:58:41	2003-02-03 19:51:01	1080.000	ACS	W
☐	J8D708NHQ	NGC3621-2	11 18 19.13	-32 50 04.3	5	2003-02-03 20:59:16	2003-02-03 21:05:26	360.000	ACS	W
☐	J8D709YEQ	NGC5457-1	14 02 53.70	+54 27 35.7	5	2003-01-23 00:48:38	2003-01-23 00:58:48	600.000	ACS	W
☐	J8D709YFQ	ANY	14 03 20.77	+54 28 16.0	5	2003-01-23 00:48:49	2003-01-23 00:57:05	494.000	ACS	H
☐	J8D709030	NGC5457-1	14 02 53.70	+54 27 35.7	5	2003-01-22 22:59:35	2003-01-23 00:45:28	1080.000	ACS	W
☐	J8D709010	NGC5457-1	14 02 53.70	+54 27 35.7	5	2003-01-22 22:41:36	2003-01-23 00:27:29	1080.000	ACS	W
☐	J8D709020	NGC5457-1	14 02 53.70	+54 27 35.7	5	2003-01-22 22:50:40	2003-01-23 00:36:33	1080.000	ACS	W
☐	J8D714KDQ	NGC5457-6	14 02 28.42	+54 20 50.1	5	2003-01-17 05:08:40	2003-01-17 05:18:50	600.000	ACS	W
☐	J8D714010	NGC5457-6	14 02 28.42	+54 20 50.1	5	2003-01-17 01:51:08	2003-01-17 03:42:20	1080.000	ACS	W
☐	J8D711J1Q	ANY	14 03 18.42	+54 21 40.2	5	2003-01-17 00:42:04	2003-01-17 00:50:20	494.000	ACS	H
☐	J8D711J0Q	NGC5457-3	14 02 51.42	+54 20 59.9	5	2003-01-17 00:41:53	2003-01-17 00:52:03	600.000	ACS	W
☐	J8D714020	NGC5457-6	14 02 28.42	+54 20 50.1	5	2003-01-17 02:00:12	2003-01-17 03:51:24	1080.000	ACS	W
☐	J8D714030	NGC5457-6	14 02 28.42	+54 20 50.1	5	2003-01-17 02:09:07	2003-01-17 04:00:19	1080.000	ACS	W
☐	J8D714KEQ	ANY	14 02 55.42	+54 21 30.3	5	2003-01-17 05:08:51	2003-01-17 05:17:07	494.000	ACS	H
☐	J8D713BNQ	NGC5457-5	14 02 29.60	+54 24 08.0	5	2003-01-16 01:45:03	2003-01-16 01:55:14	600.000	ACS	W
☐	J8D711010	NGC5457-3	14 02 51.42	+54 20 59.9	5	2003-01-16 22:39:30	2003-01-17 00:20:44	1080.000	ACS	W
☐	J8D711020	NGC5457-3	14 02 51.42	+54 20 59.9	5	2003-01-16 22:48:34	2003-01-17 00:29:48	1080.000	ACS	W
☐	J8D711030	NGC5457-3	14 02 51.42	+54 20 59.9	5	2003-01-16 22:57:29	2003-01-17 00:38:43	1080.000	ACS	W
☐	J8D713BOQ	ANY	14 02 56.63	+54 24 48.2	5	2003-01-16 01:45:15	2003-01-16 01:53:31	494.000	ACS	H
☐	J8D713030	NGC5457-5	14 02 29.60	+54 24 08.0	5	2003-01-15 23:15:03	2003-01-16 00:54:27	1080.000	ACS	W
☐	J8D713020	NGC5457-5	14 02 29.60	+54 24 08.0	5	2003-01-15 23:06:08	2003-01-16 00:45:32	1080.000	ACS	W
☐	J8D712V5Q	NGC5457-4	14 02 30.78	+54 27 25.9	5	2003-01-15 11:14:17	2003-01-15 11:24:27	600.000	ACS	W
☐	J8D712V6Q	ANY	14 02 57.85	+54 28 06.1	5	2003-01-15 11:14:28	2003-01-15 11:22:44	494.000	ACS	H
☐	J8D712010	NGC5457-4	14 02 30.78	+54 27 25.9	5	2003-01-15 06:56:40	2003-01-15 10:14:22	1080.000	ACS	W

Figure 1.4: HST Archive dataset preview page.

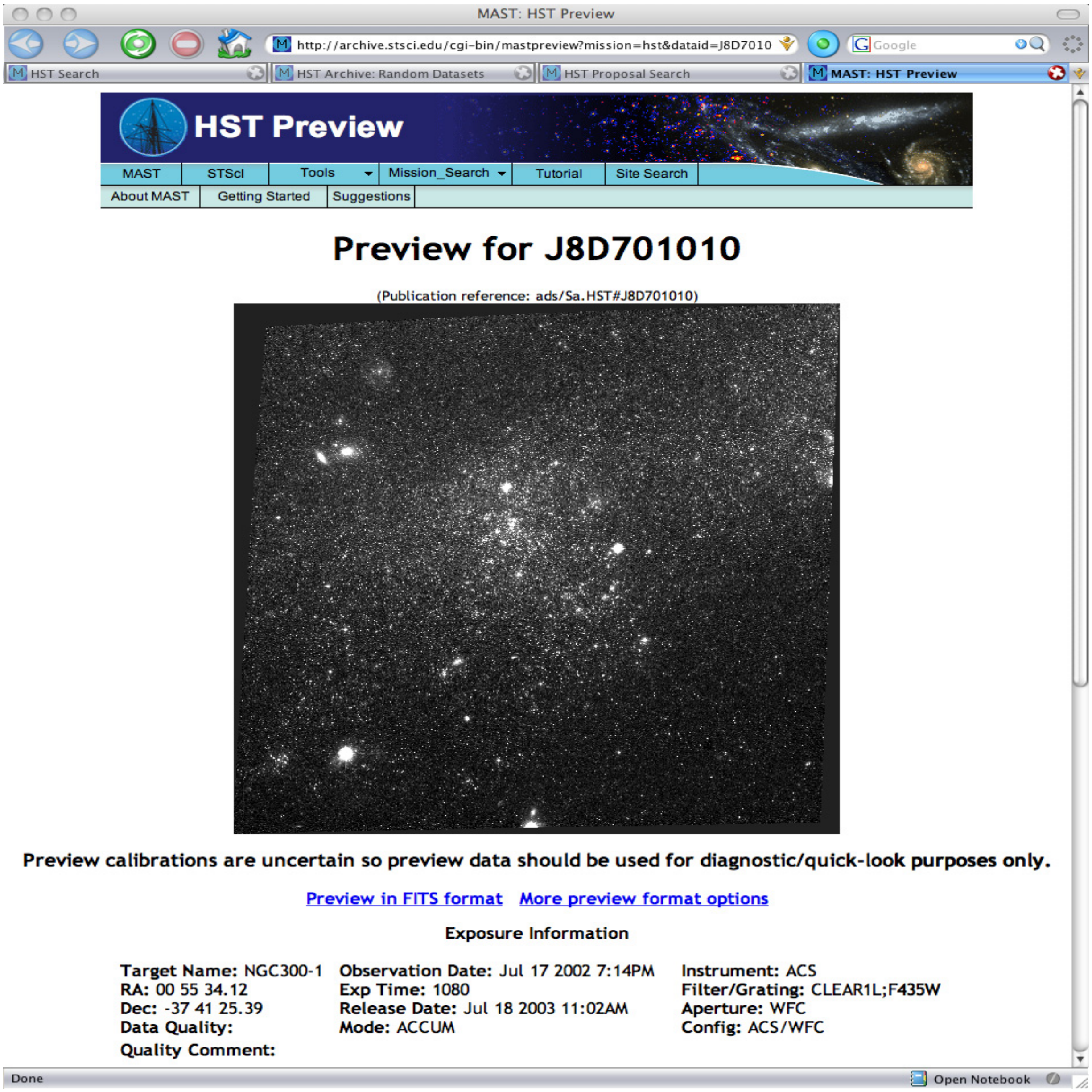
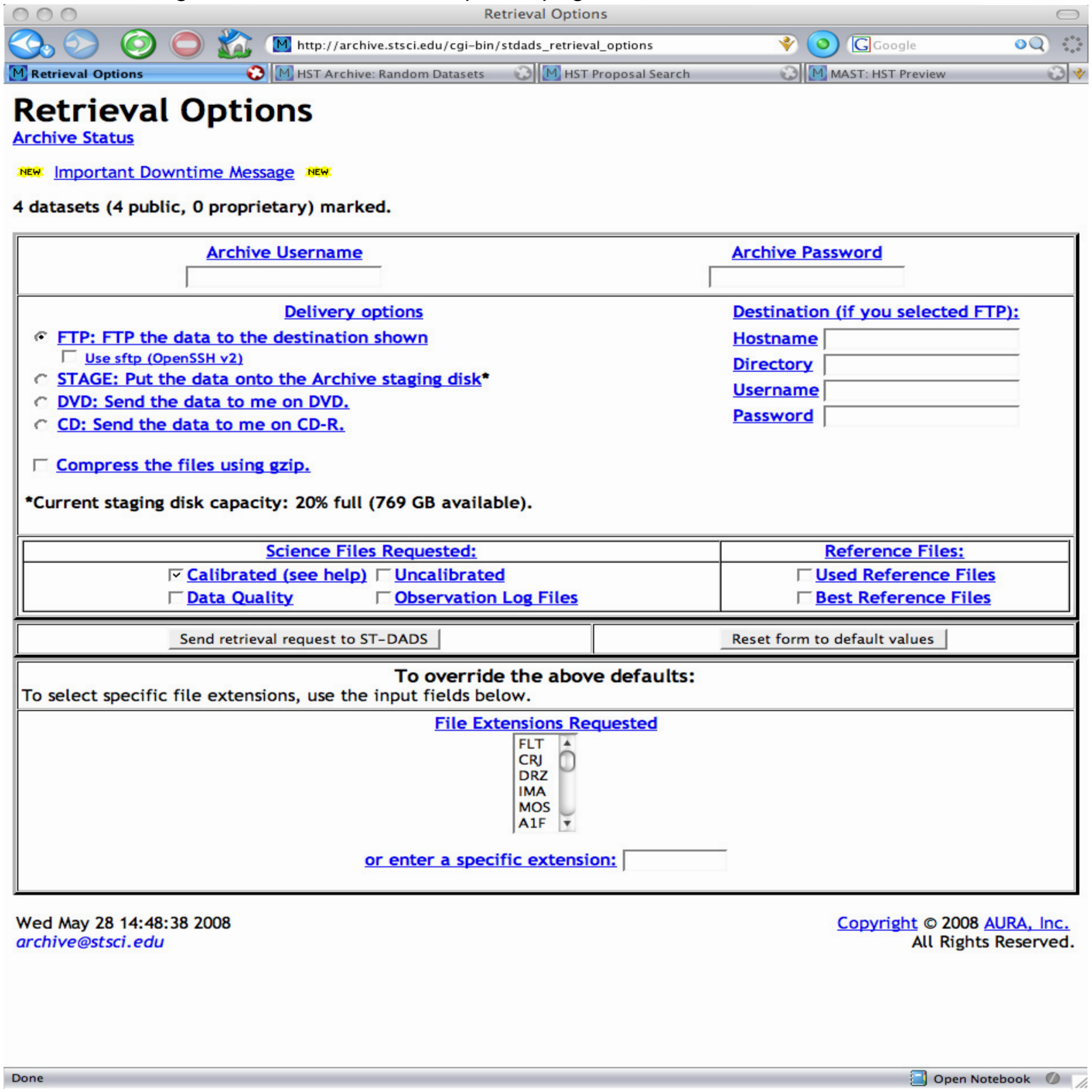


Figure 1.5: HST retrieval options page.



1.2.2 Hubble Legacy Archive

The Hubble Legacy Archive (HLA) is designed to enhance science from the Hubble Space Telescope by augmenting the data collected by the HDA and by providing advanced browsing capabilities. The primary enhancements are:

1. The data is online and available for immediate access.
2. A footprint service makes it easier to browse and download images.
3. More extensive "composite images" (e.g., stacked, color, mosaics) are being developed.

4. The absolute astrometry has been improved from 1 - 2 arcsec to ~0.3 arcsec.
5. Source lists are available for many fields.

The HLA is a joint project of the STScI, the European Coordinating Facility (ST-ECF), and the Canadian Astronomy Data Centre (CADC).

The HLA is in its Data Release 2 (DR2) phase since September 2008. This encompasses enhanced HLA products for most of the ACS and WFPC2 non-proprietary data, and access to MAST products for NICMOS, STIS, FOS, and GHRS. DR2 provides both DAOphot and SExtractor lists for ~70% of ACS images. In addition, user-interface improvements have been added with DR2, including the capability for faster downloads of multiple files, an enhanced plotting tool, GSC2, 2MASS, SDSS, and FIRST catalog overlay, user-defined search lists, and much faster footprints.

Future releases will include "ACS-like-products" (i.e., cosmic-ray rejected, MultiDrizzle-combined) for other existing data such as NICMOS and STIS images, as well as for the new instruments WFC3 and COS. A variety of spectroscopic products are also being considered (e.g., coadded multi-visit spectra, multi-wavelength spliced spectra), and a more advanced user interface (e.g., quick look spectral extractions, line identification lists). Some of the more general goals of the HLA are to make HST data compatible with the Virtual Observatory (VO), to move toward a sky atlas user-view rather than a collection of datasets, and to develop an "all-HST-sky" source list. Figures 1.6, 1.7, and 1.8 show examples of the HLA Web pages.

The HLA can be accessed at: http://hla.stsci.edu.

Figure 1.6: HLA example page with ACS WFC broad band and coadded (color) images.

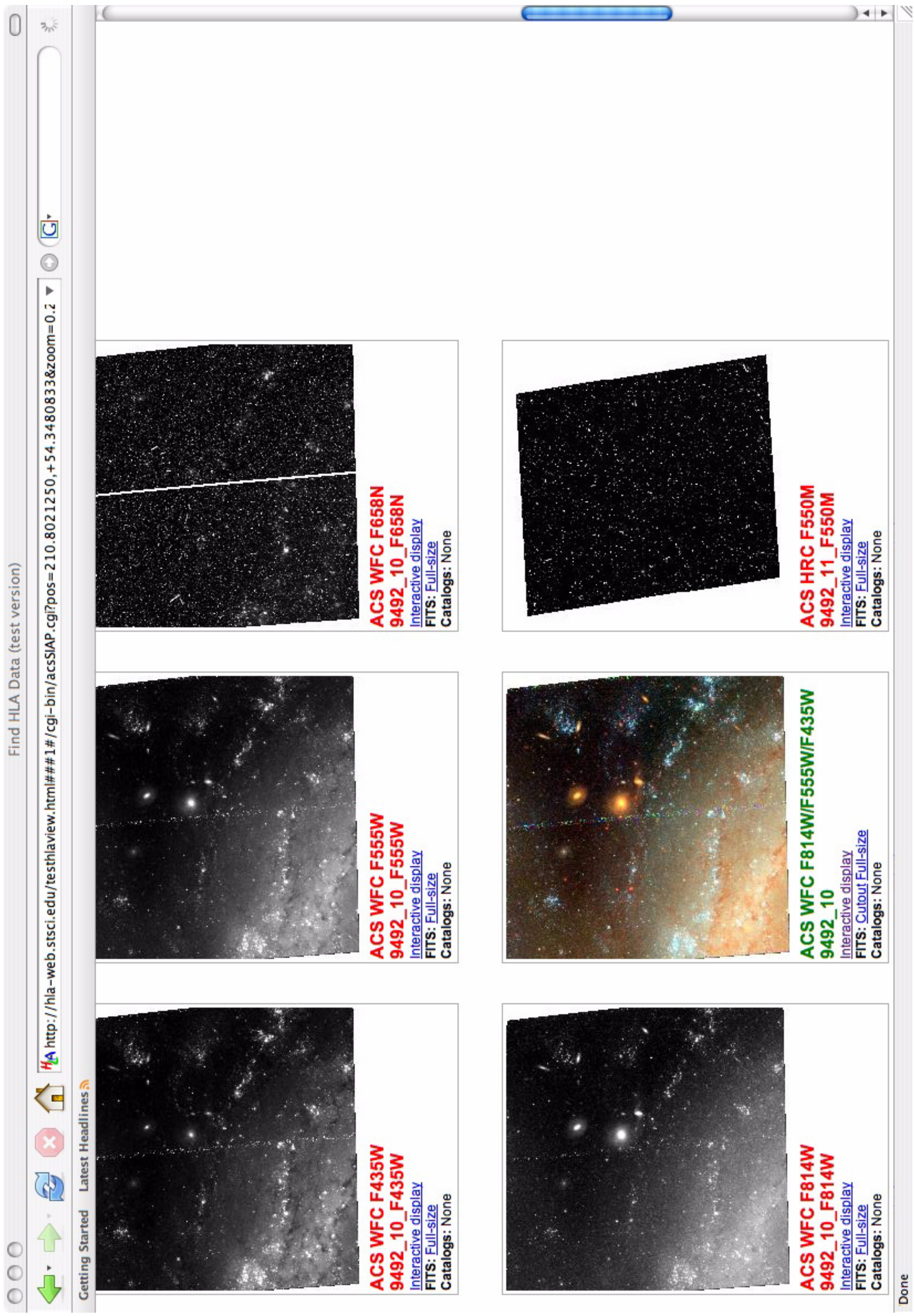


Figure 1.7: STIS Eta Car spectral image and segments of a wavelength calibrated spectrum.

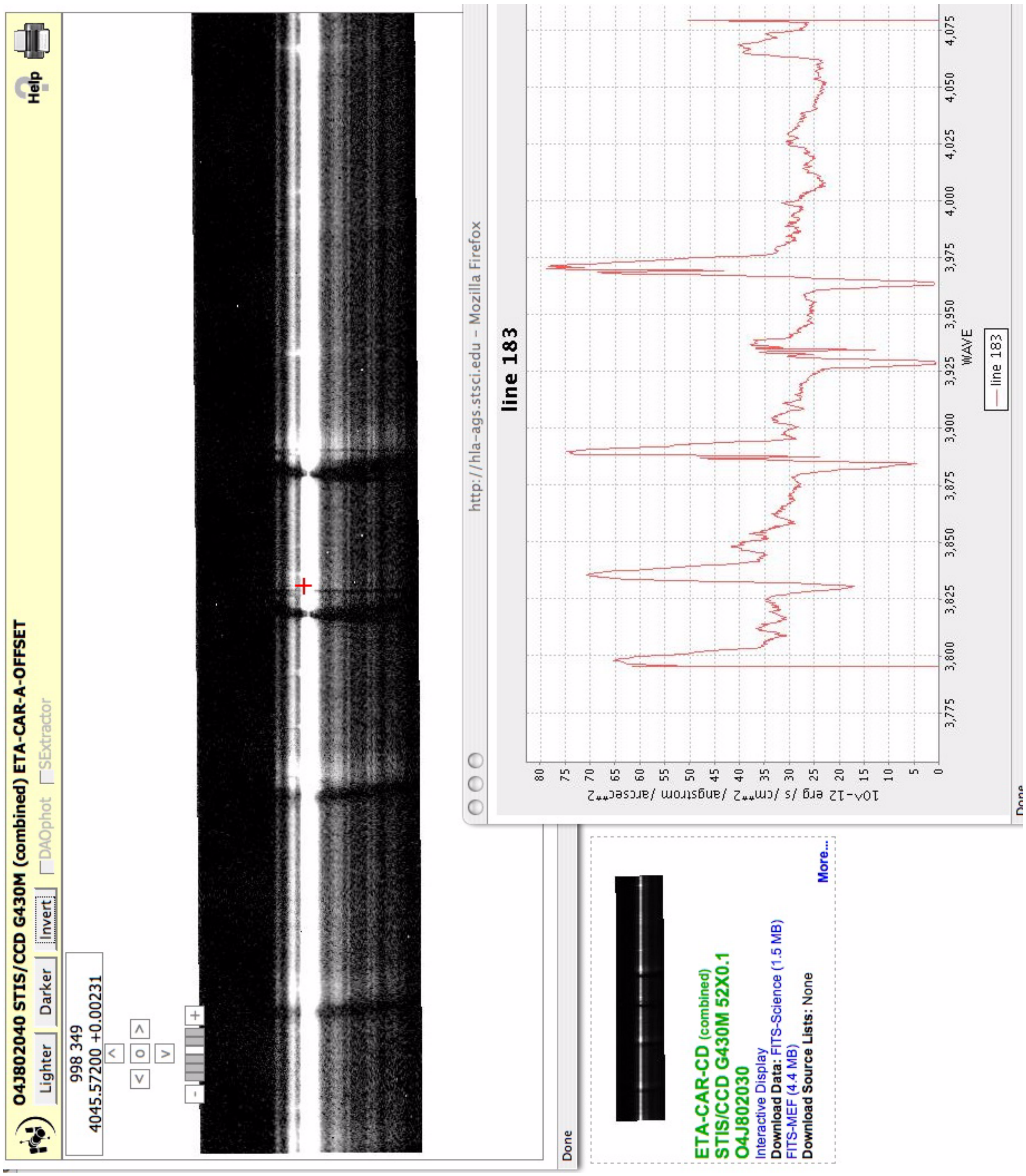
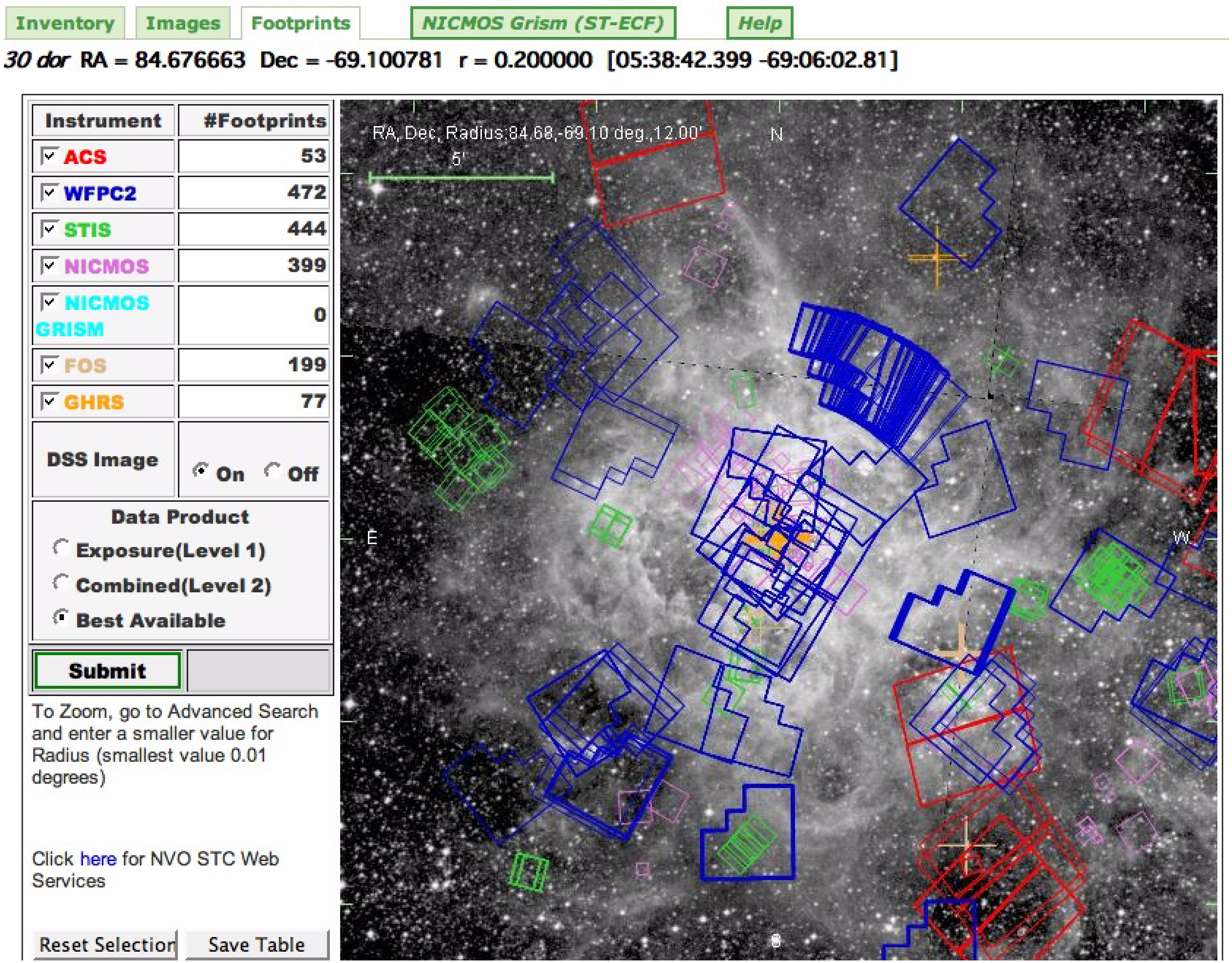


Figure 1.8: FOV of all HST instruments. Observation FOVs are overlaid on 30 Doradus.



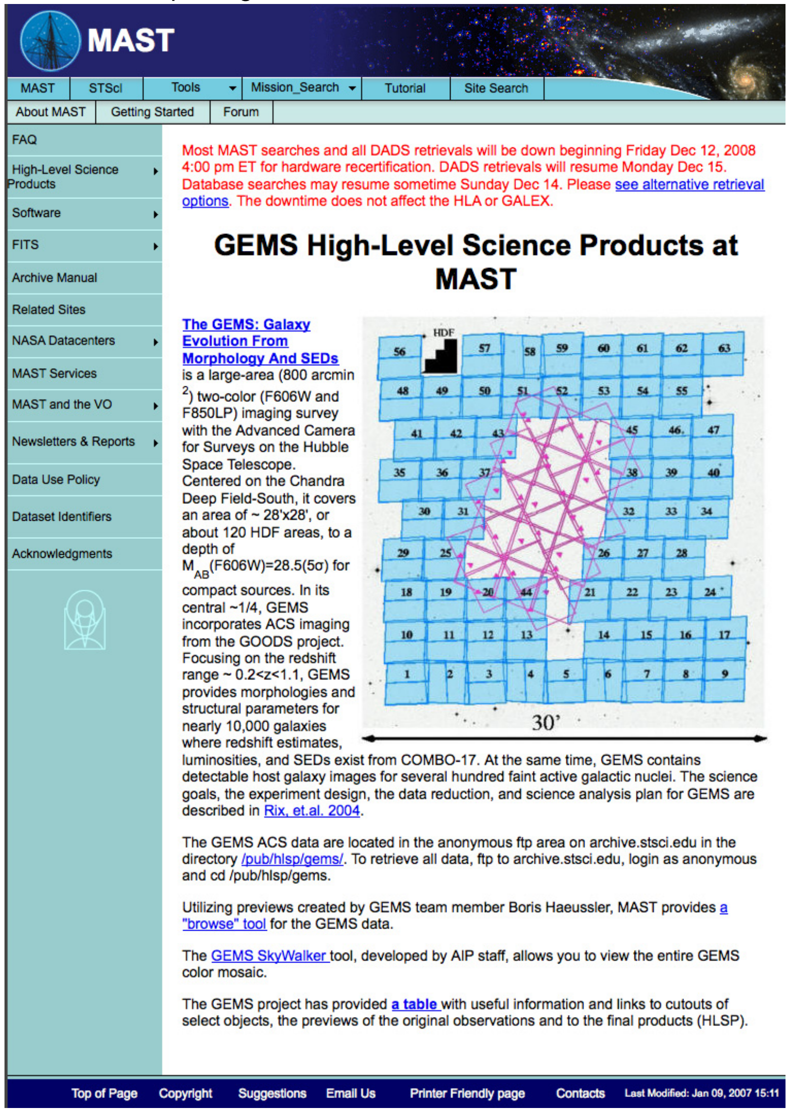
1.2.3 High-Level Science Products

MAST also contains a number of High-Level Science Products (HLSP), which are accessible at http://archive.stsci.edu/hlsp/index.html. High-Level Science Products are fully processed (reduced, coadded, 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) from Cycle 11 onward, but contain contributions from smaller HST programs and other MAST missions.

A screen shot of the Web page for the ACS product Galaxy Evolution from Morphology and SEDs (GEMS) is shown in Figure 1.9.

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 to get the latest version). Furthermore, they are asked to contact the archive scientist involved as soon as they start working on the data.

Figure 1.9: Example High-Level Science Product: GEMS.



1.3 Obtaining Data with StarView

The Java version of StarView is no longer supported by STScI. The changes necessary for StarView to support WFC3 and COS queries will not be made. However, the Archive Sciences Branch of STScI is migrating StarView's functionality to the Web. This project is still new, but a first version will be available for SM4 Servicing Mission Orbital Verification (SMOV). Documentation for this version will be available at the following URL:

http://archive.stsci.edu/

CHAPTER 2:

HST File Formats

In this chapter...

2.1 HST Data Files

The STScI pipeline automatically processes and calibrates all the data received from *HST* and assembles them into a form suitable for most scientific analyses.

Data from instruments installed after the 1997 servicing mission (ACS, COS[1], NICMOS, STIS, and WFC3[1]) are made available to observers as files in Multi-Extension FITS (MEF) format, which is directly readable by most **PyRAF/IRAF/STSDAS** tasks.

For first and second generation instruments (FGS, FOC, FOS, GHRS, HSP, WF/PC-1, and WFPC2), the data are in a format known as Generic Edited Information Set (GEIS). Byte ordering for binary data in GEIS files is machine-dependent. Therefore, for purposes of archiving and file transfer, GEIS data are converted to a modified FITS format known as "waiver FITS." It is necessary to convert waiver FITS files back to GEIS format for data processing and analysis using **IRAF/STSDAS** tasks.

As part of the reprocessing effort mentioned in Section 1.1.1, all WFPC2 data will be available[2] in either waiver FITS or MEF format. The user will be able to specify in which format the data should be retrieved from the HDA. **STSDAS** support for the analysis of WFPC2 MEF data

1. Available after installation on *HST* during Servicing Mission 4.

2. As of January 2009, some of the WFPC2 data are available in MEF format. Reprocessing will continue well into 2009.

files has recently been completed, therefore WFPC2 data, in either GEIS or MEF formats, can now be fully processed with **STSDAS** tasks.

For older instruments (FOC, FOS, FGS, GHRS, HSP, WF/PC-1, and WFPC2) data are provided in waiver FITS format, which needs to be converted back to GEIS format before processing. Note that WFPC2 data will soon be available in MEF format from the HDA. Newer instruments (ACS, COS, NICMOS, STIS, and WFC3) generate and store files in FITS format and should not be converted to GEIS.

Table 2.1: HST Instrument File Formats.

Instrument	Status	Format from HDA	Format for STSDAS Analysis
ACS	reactivated after SM4	MEF	MEF
COS	active after SM4	MEF	MEF
FGS	active	waiver FITS	GEIS
FOC	decommissioned SM3B	waiver FITS	GEIS
FOS	decommissioned SM2	waiver FITS	GEIS
GHRS	decommissioned SM2	waiver FITS	GEIS
HSP	decommissioned SM1	waiver FITS	GEIS
NICMOS	active	MEF	MEF
STIS	reactivated after SM4	MEF	MEF
WFC3	active after SM4	MEF	MEF
WFPC2	decommissioned during SM4	waiver FITS & MEF[1]	GEIS & MEF[1]
WFPC1	decommissioned SM1	waiver FITS	GEIS

1. HDA now also stores WFPC2 data in MEF format.

This chapter describes these two HST file formats, GEIS and MEF, in more detail. ACS, COS, NICMOS, STIS, and WFC3 observers should pay particular attention to the section on MEF files, which shows how to identify and access the contents of these files and covers some important conventions regarding header keywords. Users of the heritage instruments should consult the material on data groups and conversion from waiver FITS to GEIS format found in Section 2.3.1 before proceeding to Chapter 3.

Throughout this document, references to FITS files will always mean HST Multi-Extension FITS format files. References to waiver FITS files will always be explicitly stated.

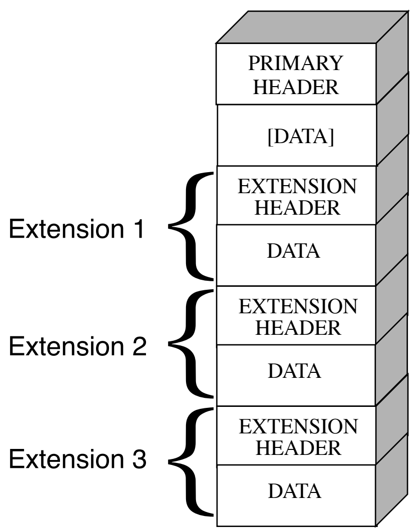
2.2 Multi-Extension FITS File Format

Flexible Image Transport System (FITS)[3] is a standard format for exchanging astronomical data, independent of the hardware platform and software environment.

A 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 *keywords* that describe the data in a particular HDU and the data component may immediately follow the header.

For HST FITS files, the first HDU, or primary header, contains no data. The primary header may be followed by one or more HDUs called extensions. Extensions may take the form of images, binary tables, or ASCII text tables. The data type for each extension is recorded in the `XTENSION` header keyword. Figure 2.1 schematically illustrates the structure of a FITS file and its extensions.

Figure 2.1: FITS file structure.



Each FITS extension header contains the required keyword `XTENSION`, which specifies the extension type and has one of the following values:

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

IMAGE, BINTABLE, and TABLE, corresponding to an image, binary table, and ASCII table, respectively.

Table 2.2: HST Header Keyword Descriptions.

HDU Keyword	Description	Values
XTENSION	Data type for extension	• IMAGE • BINTABLE (binary table) • TABLE (ASCII table)
EXTVER	Imset number (see Table 2.3)	Integer
EXTNAME	Extension names that describe the type of data component	• SCI (science image) • ERR (error image) • DQ (data quality image) • SAMP[1] (number of sample) • TIME[1] (exposure time) • EVENTS[2] (photon event list) • GTI[2] (good time interval) • WHT (weight image) • CTX (context image)
PIXVALUE[3]	Allowed for any extension except SCI, and used for an image with uniform value for all pixels	Real number

1. Only found in NICMOS and WFC3 data.
2. Only found in COS and STIS data.
3. When an image has the same value at all pixels (e.g., data quality value), the extension has no data component. Instead, the constant pixel value is stored in the header keyword PIXVALUE.

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.

2.2.1 Working with Multi-Extension FITS Images in IRAF

The FITS image kernel included in **IRAF** version 2.12.2 and higher treats each extension like a standard **IRAF** image. This subsection provides an overview of Multi-Extension FITS file syntax and keyword options needed to extract specific types of data. 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.

In order to work with ACS, NICMOS, or STIS data, users will need to upgrade to at least IRAF 2.12.2 and STSDAS 3.2. Processing of WFC3 data will require instead STSDAS 3.8 or higher. COS will require STSDAS 3.9 or later and its pipeline runs only under PyRAF (Section 3.2).

Generating a FITS File Listing

Once you have downloaded any FITS files from the HDA, 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. For example, Table 2.3 illustrates the first 11 lines generated by **catfits** from a NICMOS `MULTIACCUM` FITS file containing only images. The output columns from **catfits** contain the following information:

- The first column lists the extension numbers. Note that the primary header extension number is 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.
- The fourth column lists the imset number, given in the EXTVER keyword.

Several **STSDAS** tasks can work with entire imsets (see Section 3.4.3), but most operate on individual images. See the "Data Structure" chapters in Part II of this Data Handbook for more information on the contents of a particular instrument's imsets.

Table 2.3: 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*`][`*image section*`]`

or

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

Specifying the extension number is the most basic method of accessing an HDU in a FITS file, but it is not necessarily the most useful syntax. 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

ACS, COS, NICMOS, STIS, and WFC3 data files use an **IRAF** image kernel convention that allows HDU extensions, under certain circumstances, to *inherit* keywords from the primary header. 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. Note, 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 global keyword inheritance by using the NOINHERIT keyword option in the file specification. For example:

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

The resulting `outfile.fits` contains no global keywords from `fitsfile.fits`, except for keywords which were present in the extension header. In the first example, where the FITS image uses an absolute extension number, `noinherit` is the only entry needed in the

FITS option field. In the second command, the `noinherit` option is bracketed with the `EXTNAME` and `EXTVER` keyword. 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 MEF FITS files:

If the specified output file does not yet exist, it is created containing only the specified extension of the original file. For example, to copy the contents of the primary header of `fitsfile.fits` to a new FITS file named `outfile.fits`:

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

Note that **imcopy** will yield an error if an extension is not specified in the command. If the specified output file already exists and you want to append a new extension to it, you must include the APPEND option in the output file specification. The example below appends extension `[sci,2]` of `fitsfile.fits` to the existing file `outfile.fits`, while retaining the original EXTNAME and EXTVER of the extension. The `noinherit` keyword option prevents 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]
```

Note that the backslash is added to indicate that the remainder of the command follows on the next line, after the “>>>” prompt.

To change the EXTNAME or EXTVER of the appended extension, you can specify new values for these keywords in the output extension:

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

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

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

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

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

2.2.2 Working with FITS Table Extensions

FITS tables are used to store certain types of data from ACS, COS, NICMOS, STIS, WFC3, and WFPC2. For these instruments, OPUS produces association tables that list the exposures used to construct association products. In addition, reference data may be stored in FITS tables. This section describes:

- 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 in, **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. The syntax for accessing a specific FITS table is similar to the syntax for accessing FITS images (see Section 2.2.1), with the following exceptions:

- The FITS table interface does not support header keyword inheritance.
- FITS tables must reside in a FITS table extension, in either ASCII form (XTENSION=TABLE) or binary form (XTENSION=BINTABLE).

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 (e.g., **tedit**) can modify an existing FITS extension, and tasks that create a new table (e.g., **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, the 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 and COS spectra take advantage of this feature. Each column of a STIS or COS spectral table holds data values corresponding to a particular physical attribute, such as wavelength, net flux, or background flux. For STIS, 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 **sgraph** and **igi** (which have been appropriately modified to handle three-dimensional data tables), you will need to extract the desired arrays from the three-dimensional table. Two **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. The task **tscopy** will copy rows, columns, and subsets of tables. 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). To obtain a list of the column names, you can run the **tlcol** task (type "`tlcol infile.fits`").

A row selector parameter can be used to specify a certain row in the table. 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 `infile.fits`, specifically from the columns labeled 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=SCI and EXTVER=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`".

2.3 GEIS File Format

GEIS format[4] is the standard format for reducing data from FGS, FOC, FOS, GHRS, HSP, WF/PC-1, and WFPC2. Data from these instruments are distributed by the HDA in waiver FITS files and must be converted to GEIS format. Note that WFPC2 data will soon be available from the HDA in waiver FITS and MEF format files[2].

All HST images in GEIS format consist of two components: a *header file* (with suffix ending in "h"), and a separate *binary data file* (with suffix ending in "d"). Both files must reside in the same directory for processing.

GEIS header files (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.

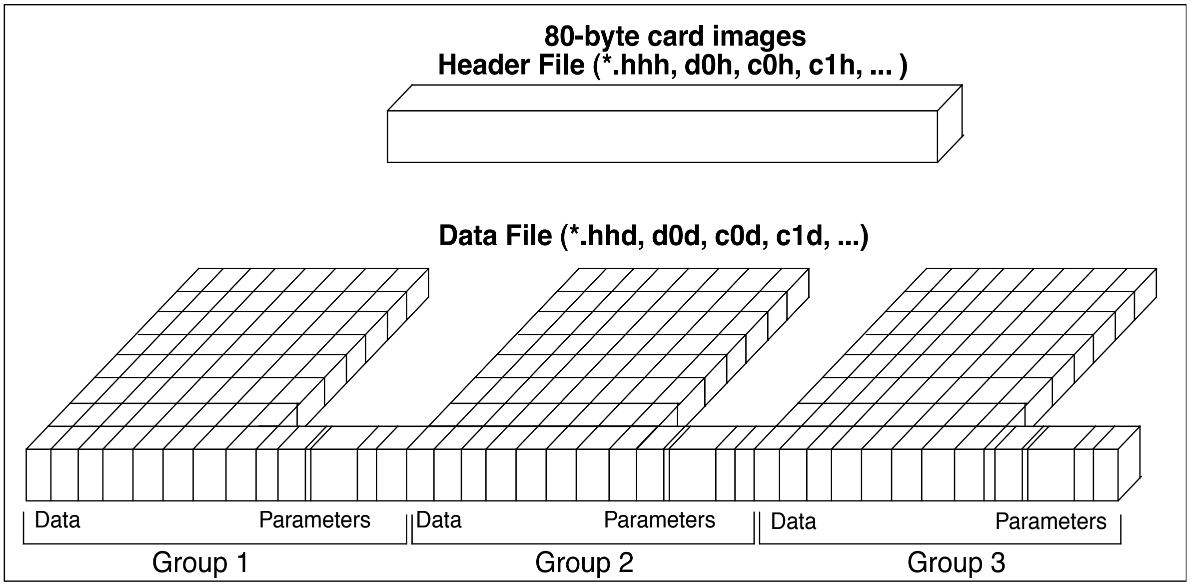
GEIS binary data files, (e.g., `w0lo0105t.c1d`), contain one or more *groups* of binary data. Each group comprises a data array followed by an associated block of binary parameters called the Group Parameter Block (GPB). Each group of a GEIS file has identical array sizes, data types, and group parameters. Figure 2.2 depicts the structure of a GEIS data file graphically.

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."

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

4. GEIS files are also commonly referred to as **STSDAS** images.

Figure 2.2: GEIS file structure.



2.3.1 Converting Waiver FITS to GEIS

The HDA stores and distributes datasets from FGS, FOC, FOS, GHRS, HSP, WF/PC-1, and WFPC2 in waiver FITS format.

We highly recommend that users convert waiver FITS datasets back into their native GEIS format before processing them.

Your data must be in GEIS format for you to use 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 waiver FITS format to the GEIS format. A special convention is used to map GEIS format to waiver 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 multi-group 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
```

For example, the waiver FITS WFPC2 dataset `u6n20101m_clf.fits` can be converted using **strfits** to created two GEIS files: `u6n20101m.clh` (a header file) and `u6n20101m.cld` (a data file).

2.3.2 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, a single FOS or GHRS dataset may comprise many spectra. The data corresponding to each CCD (for WF/PC-1 or WFPC2), or each readout (FOS) or bin (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 2.4 lists the *contents* and the number of groups in the final calibrated image for the most commonly-used modes of each instrument that uses the GEIS data format.

Table 2.4: 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 data(`.d0h`, `.c0h`), digital sky data (`.d1h`, `.c1h`), analog star data (`.d2h`, `.c2h`), or analog sky data (`.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.

2.3.3 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 primary header file and the parameter block associated with a group. Because GEIS header files are composed solely of ASCII text, they are easy to view or print using standard Unix text-handling facilities. However, the group parameters are stored in the binary data file. To access them you need to use an **IRAF** task such as **imheader**, as shown in the section titled "Printing Header Information".

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 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 is useful for updating header keywords containing calibration switches and calibration reference files.

Always edit headers using tasks like hedit, eheader, groupmod, or 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. Most of these tasks operate on only one image at a time, so you usually need to specify the GEIS file group to be processed. If you do not specify a group, the task will operate on the first group by default.

Specifying a Group

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

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

(Always refer to a GEIS file by its header file name, with suffix ending in “h”, even though mathematically you are operating on the data portion.)

The command above 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 multi-group GEIS file results in an output file containing a single group.

Specifying an Image Section

If you wish to process only part 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 it 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 two commands 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 for MEF files, the task **imheader** extracts and prints information about a GEIS image. This task prints 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 multi-group GEIS file which contains 64 groups of data, 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
```

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

Other Group-Related Tasks

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

2.3.4 The waiver FITS Format

File formats for the first and second generation HST instruments (FGS, FOC, FOS, HSP, WF/PC-1, GHRS, and WFPC2) were developed before the standardization of MEF format. The waiver FITS format was developed in response to the need for a machine independent storage format for these data and was based on the idea of stacking multi-group GEIS data as a new dimension in a FITS image.

For example, a WFPC2 science data GEIS file with four groups has four 800x800 pixel images in its data file. When this GEIS file is converted to a waiver FITS file (using the **IRAF** task **stwfits**), the resulting FITS file has the dimensions of 800x800x4 (a three-dimensional image!) in its primary HDU. Similarly, an FOS GEIS file may have 40 groups, each group being a one-dimensional image (spectrum) that is 2064 pixels in length. The waiver FITS file equivalent of this FOS GEIS file has one 2D image of the size 2064x40 as its primary HDU.

In the case of a 4-group WFPC2 image, the first extension of the waiver FITS file is a table containing four rows. Each row represents a group. Each column in the table will correspond to a group keyword. Each element in the table contains keyword values for a specific group. This can be viewed using the **tread** command:

```
st> tread u2eo030ft_c0f.fits[1]
```

You can also display the values of specific keywords using a command like **tdump**, which in the example below, writes the values to a file called "params.txt":

```
st> tdump u2eo030ft_c0f.fits[1] columns="PHOTMODE, CRVAL1,\
CRVAL2, BIASEVEN, BIASODD" datafile=params.txt
```

```
File "params.txt"
WFPC2, 1, A2D15, F487N CAL  204.716  70.286  295.994  295.966
WFPC2, 2, A2D15, F487N CAL  204.667  70.283  318.476  318.966
WFPC2, 3, A2D15, F487N CAL  204.680  70.304  303.804  303.966
WFPC2, 4, A2D15, F487N CAL  204.742  70.300  306.539  306.966
```

The data component of a multi-group GEIS file, when converted to waiver FITS, is stored in the primary HDU of the waiver FITS image as a multi-dimensional image. The task **display** can be used to view one group image at a time. For instance, to view group 2 of a 4-group waiver FITS WFPC2 image, type:

```
st> display u2eo030ft_c0f.fits[0][*,*,2]
```

The **display** task reads the image from the primary HDU, and specifies the image using three-dimensional syntax, where the “*” represents all pixels in *x* and *y*.

If you want to view the central 400x400 section of the WFPC2 image, you can use the following command:

```
t> display u2eo030ft_c0f.fits[0][200:600,200:600,2]
```

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

CHAPTER 3:

Analyzing HST Data

In this chapter...

3.1 Analysis Options for HST Data

HST data can be manipulated with several different software packages. In this section we introduce a few of the software language options.

3.1.1 IRAF/STSDAS

STSDAS is an **IRAF** package developed by STScI for the reduction and analysis of HST data. The package contains 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. Sections 3.2 through 3.7 introduce the basics of **STSDAS**, illustrating how to display HST data, presenting some simple data manipulations, and pointing you towards more sophisticated tasks.

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**/**PyRAF**, consult the **IRAF**/**PyRAF** Primer in Appendix A before reading further.

3.1.2 PyRAF

PyRAF is a 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 as the **IRAF** CL. 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
- a GUI epar parameter editor with help displayed in a separate window
- IDL-like capabilities
- true error handling for scripts (shows which line of a script fails 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

3.1.3 Python

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

- numpy: IDL-style array manipulation facilities
- PyFITS: read and write FITS files to and from arrays
- matplotlib: plotting and image display package
- numdisplay: display arrays to Saoimage, DS9, and Ximtool
- **PyRAF**: 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. Tools are available to read (but currently not write) GEIS files.

Python can make use of **PyRAF** to allow access to **IRAF** tasks. Tutorials are available which illustrate the use of Python for interactive data analysis in astronomy (in much the same style as is now popular with IDL). The initial focus of these tutorials is the use of interactive tasks for the novice user. The more advanced tutorials focus on teaching the details of Python programming. The tutorials can be downloaded from:

`http://www.scipy.org/Topical_Software`

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`

3.1.4 Interactive Data Language (IDL)

IDL is an array-based, interactive programming language that provides many 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, it is very popular in the astronomical community with many astronomers using it for their analysis of HST data.

It can be obtained from ITT Visual Information Solutions (http://www.ittvis.com/idl/), for a fee. Libraries for reading HST data are part of the freely available ASTRON library (http://idlastro.gsfc.nasa.gov) which has links to other IDL astronomy libraries.

3.1.5 Fortran and C

For those who wish to write their own Fortran or C applications, we recommend using 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 GEIS format directly so users will need to use the waiver FITS format obtained from the archive and manually extract the needed information.

3.1.6 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). These libraries do not support GEIS format, but can handle waiver FITS format.

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.
- `examples` *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 current **STSDAS** package structure. The current version of **IRAF** (v2.14 as of September 2007) and TABLES (v3.9 as of November 2008) must be installed on your system in order to use **STSDAS**. It is always a good idea to make sure you have the most recent versions of the software as they often include important code updates. Newer versions of **IRAF**, **STSDAS**, and TABLES are periodically released. Please check the following Web site for the latest information:

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

Table 3.1: **STSDAS** Version 3.8 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.**
acoadd	Image coaddition using lLucy Method with optional acceleration
gaussfit	Least squares and robust estimation program
plucy	Multiple channel photometric image restoration
slitless	Extract a spectrum from a direct/Grism image pari
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 and GEIS input/output for HST 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.
mstools	General-purpose tasks that handle NICMOS imsets
hstcos	Tasks for calibrating COS 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.
wfc3	Tasks for calibration WFC3 data
wfpc	Tasks for calibrating WF/PC-1 and WFPC2 data.
w_calib	Tasks for deriving the WF/PC-1 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 (V1.2) 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.

3.2.2 Packages of General Interest

For 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 **IRAF** package includes general tasks for copying (**imcopy**), moving (**imrename**), deleting (**imdelete**), displaying (tv), and examining (**imexamine**) 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 image statistics, rotating and magnifying images, and registering and dewarping images.
- **stsdas.toolbox.imgtools**: This package contains tools for working with 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. Also of note are the tasks **imcalc** for performing image arithmetic, and **rd2xy** and **xy2rd** for converting between RA, Dec and x,y pixel coordinates. Many of these tasks will also work with single group or waiver FITS format files.
- **stsdas.toolbox.imgtools.mstools**: This package contains tools for working with FITS image files, in particular ACS, NICMOS, STIS, and WFC3 image sets (imsets). **Msstatistics**, for example, will print statistics on each image set (imset) in an image file. Similarly, **msarith** can be used for image arithmetic and uses information from the Data Quality (DQ), Error (ERR), and if available, the TIME and SAMP extensions in the calculation.
- **stsdas.analysis.dither.multiDrizzle**: This routine is provided as a “one-touch” interface for performing all the tasks necessary for registering dithered HST images, performing cosmic ray rejection, removing geometric distortion and performing the final image combination using “**drizzle**”.
- **stsdas.analysis**: This package contains general tasks for image analysis, such as Fourier analysis (**fourier**), dithering (**dither**), and **fitting**.

For 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 only for input.) 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

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 DS9, in a separate window from your **IRAF** session, either from a different xterm window or as a background job before starting **IRAF**. To start DS9, type the following in a Unix window:

```
> ds9 &
```

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/web/projects/x11iraf/.

2. Make sure that **IRAF** has been set to expect the size of the image you wish to display. This is controlled with the `stdimage` keyword. As long as this is set to the largest image size you would like to display, the entire image will be available in your viewer. Otherwise, **IRAF** will only display up to the currently set limit. If you would like a general setting that will display images up to 8192 × 8192, set `stdimage` equal to `imt7`, otherwise, set `stdimage` equal to the length of one side of the image. Other types of images may require larger formats, like `imt4096` for ACS and WFC3 images, for example, while `imt1024` is often adequate for WFPC2 and NICMOS images. To set the size of the image buffer to 1024 × 1024, or the size of the image you would like to display, type the following in the **IRAF** window:

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

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

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

4. Display the image in frame 1 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. Figure 3.1 shows how to display chip 1 of an ACS/WFC 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.

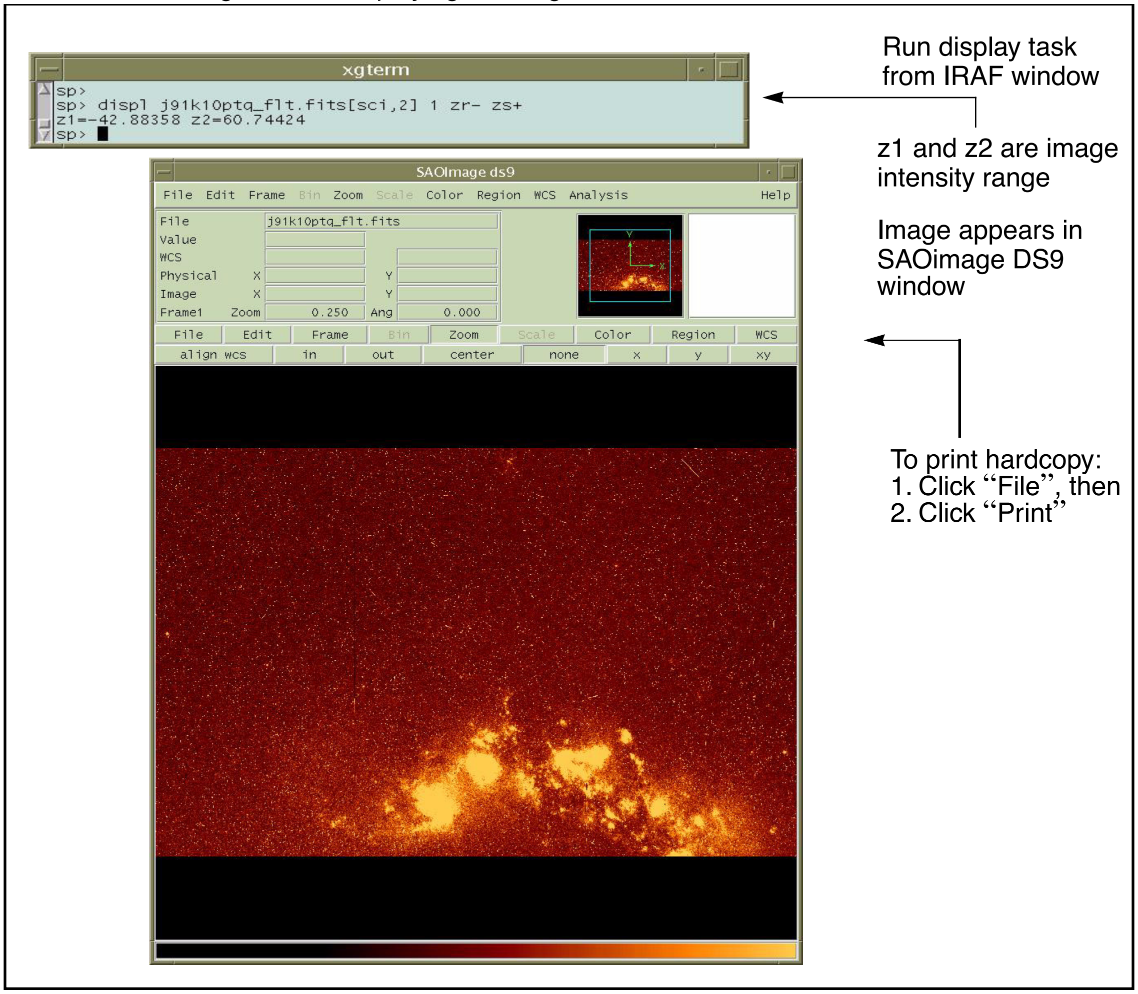
Modifying the Display

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 DS9 window, you can use the SAOimage commands displayed near the top of the image window to manipulate or print your image. The *SAOimage User's 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. Online help is also available at the system level: type `man saoimage` in Unix.

Figure 3.1: Displaying an image.



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 intensity scaling using the `zscale`, `zrange`, `z1` and `z2` parameters. The `zscale` parameter toggles the auto scaling. Setting `zrange+` (and `zscale-`) tells the task to display the image using the minimum and maximum values in the image. To customize your minimum and maximum intensity display values, set `z1` to the minimum value and `z2` to the maximum value that you want displayed. You must also set `zscale-` and `zrange-` to disable these parameters. To ensure the entire image is displayed, you should set the `fill+` option. For example:

```
im> disp j91k10ptq_flt.fits[sci,2] 1 zrange- zscale- z1=2.78
z2=15.27 fill+
```

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 DS9 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 a 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, you can simply use:

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

3.3.2 Working with Image Sections

To display only a portion of an image, use 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, type:

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

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

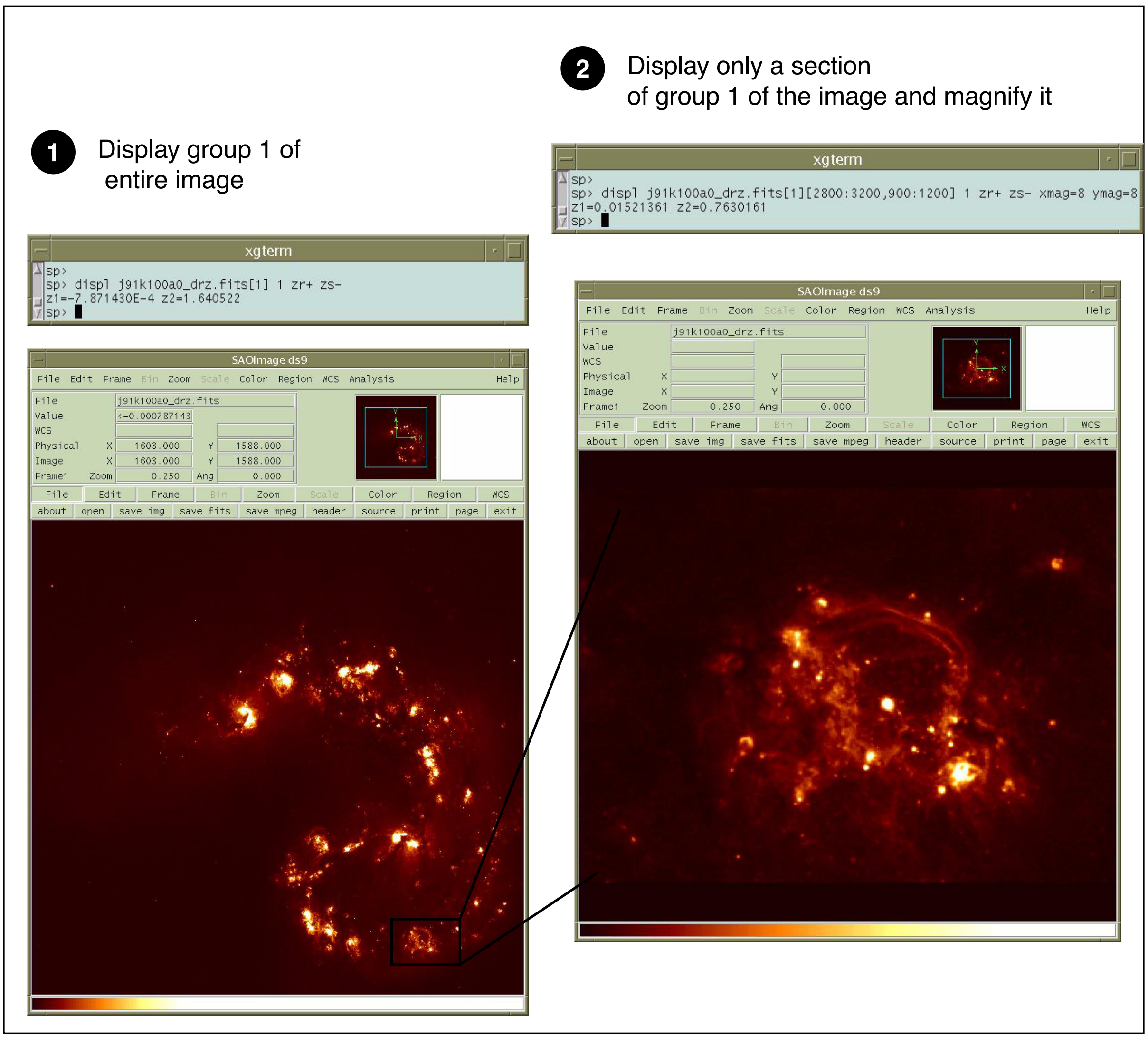
```
tv> display image.fits[sci,2][101:200,*] 1
```

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

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 number followed by the image section.

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

Figure 3.2: Displaying sections and groups of an image.



3.4 Analyzing HST Images

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

- Relating your image to sky coordinates
- Examining and manipulating your image
- Working with Multi-Extension FITS imsets
- Converting counts to fluxes

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 contours and grid of RA and Dec. 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.) DS9 displays the current *x,y* pixel location of the cursor in the upper-left corner of the window. These tasks will only give accurate results when they are run on images which have been corrected for distortion. 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, minutes, and seconds.

Observers should be aware that *these tasks do not correct for geometric distortion*. Only ACS, FOC, STIS, WFC3[2], and WFPC2 images currently undergo geometric correction during standard pipeline processing. Table 3.2 lists some additional tasks that make use of the standard astrometry keywords.

Table 3.2: Additional **IRAF** and **STSDAS** Astrometry Tasks.

Task	Package	Purpose
compass	stsdas.graphics.sdisplay	Plot north and east arrows on an image.
imexamine	cl.images.tv	(rimexamine) Mutli-purpose tool for examining images - statistics, photometry, and astrometry.
north	stsdas.hst_calib.ctools	Display the orientation of an image based on keywords.
rimcursor	cl.lists	Determine RA and Dec of a pixel in an image.
wcslab	cl.images.tv	Produce sky projection grids for images.

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. Generally, observations obtained during the same visit using the same guide star acquisition are well-registered. Observations separated by one or more guide star acquisitions will typically have small shifts. 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 find its absolute position from the Guide Star Catalog II (GSC2), which is on the IAU recommended International Celestial Reference Frame, and has a typical error of 0.3". Contact the Help Desk if you require further assistance.

2. After installation on *HST* during Servicing Mission 4.

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. Table 3.4 lists many useful **IRAF/STSDAS** tasks for manipulating images. The list is not exhaustive, just a sample of what is available.

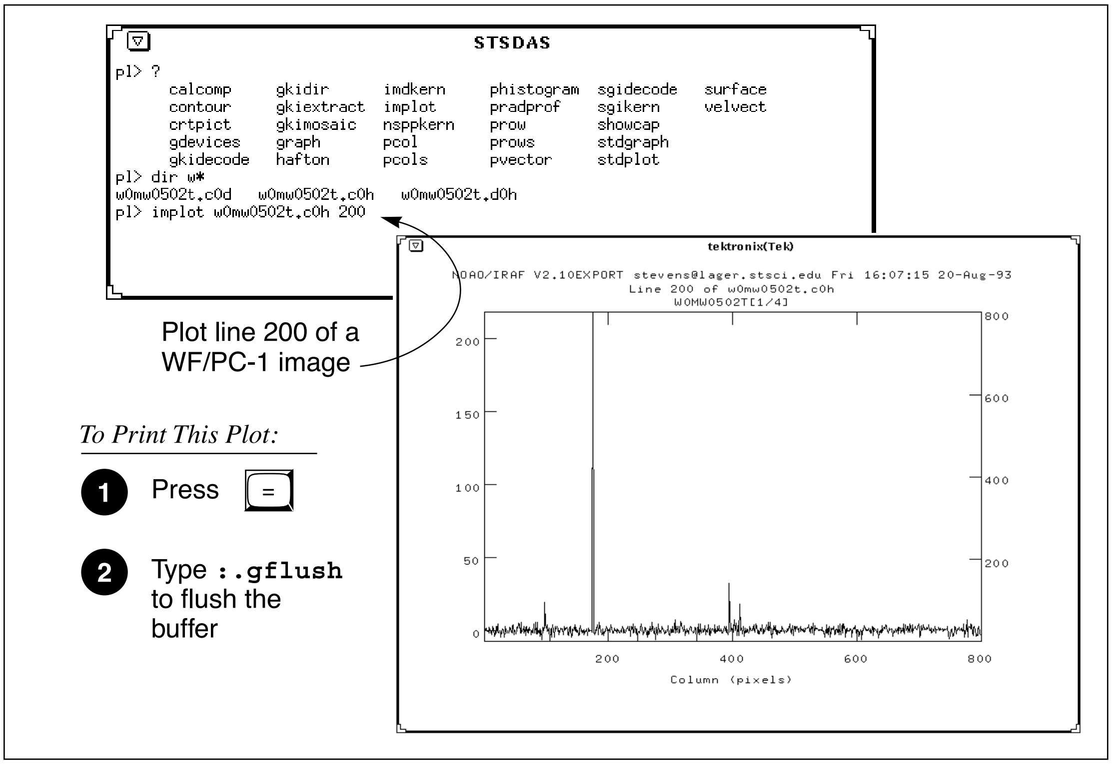
implot

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 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 online help, but those most commonly used are listed in Table 3.3. Figure 3.3 shows an example of how to use the **implot** task to plot a row of data.

Table 3.3: Basic implot Commands.

Keystroke	Command
?	Display online 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 (step 2 only works in **IRAF**).



imexamine

The **imexamine** task (in the **images.tv** package) is a powerful **IRAF** 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 FITS Imsets

ACS, COS, NICMOS, STIS, and WFC3 data files contain groups of images, called *imsets*, associated with each individual exposure. See Table 2.1 and the Data Structures chapters in Part II 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
fmedian	images.imfilter	box median filter a list of images
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]
grplot	stsdas.graphics.stplot	Plot lines of a group-format 1D GEIS image (spectrum)[1]
gstatistics	stsdas.toolbox.imgtools	Compute image statistics[1]
histogram	stsdas.graphics.stplot	Plot or list a histogram of a table column, image, or list
igi	stsdas.graphics.stplot	Interactive Graphics Interpreter plotting package
imcalc	stsdas.toolbox.imgtools	Perform general arithmetic on images
imcombine	images.immatch	Combine images using various algorithms
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 Section imexamine)
implot	plot	Plot lines and columns of images (see Section implot)
magnify	images.imgeom	Magnify an image
msarith	stsdas.toolbox.mstools	Performs basic arithmetic on HST FITS imsets[1]
mscombine	stsdas.toolbox.mstools	Extension of **gcombine** for HST FITS imsets[1]
msstatistics	stsdas.toolbox.mstools	Extension of **gstatistics** for HST FITS imsets[1]
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) Useful for masking of images.
rotate	images.imgeom	Rotate an image
saodump	stsdas.graphics.sdisplay	Make image and color map files from SAOimage display
sgraph	stsdas.graphics.stplot	Graph image sections or lists
siaper	stsdas.graphics.stplot	Plot science instrument apertures of HST
xregister	images.immatch	Register 1D or 2D images using cross-correlation

1. Will process all groups of a multi-group GEIS file.

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 ACS, COS, NICMOS, STIS, and WFC3 data files. It separates each imset into its basic components (e.g., SCI, ERR, DQ, SAMP, TIME). 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 (NICMOS and WFC3/IR data only).

Once **gcombine** has finished, **mscombine** then reassembles the individual output images into imsets and outputs them as one data file. The output images and error maps from **gcombine** form the SCI and ERR extensions of the output imset(s). The DQ extension will be a combination of the masking operations and the rejection algorithms executed in **gcombine**. TIME extensions 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 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 feature is the inclusion of the error and data quality information included with HST FITS images in computing statistical quantities.

In addition to the standard statistical quantities (min, max, sum, mean, standard deviation, median, mode, skewness, kurtosis), two additional quantities have been added to take advantage of the error information: the weighted mean and the weighted variance of the pixel distribution. If x_i is the value at the *i*-th pixel, with associated error σ_i, the weighted mean and variance used in the task are:

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

and:

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

The data quality information in the imset 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 FITS 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 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 other 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 flux.
- Instructions on how to use header keyword information to convert HST image units 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 to Units of Flux or Magnitude

Calibrated HST images obtained from the HDA store signal in various units. Table 3.5 lists a selection of HST instrument image units. Refer to the instrument specific Data Handbooks for instruments not included in this list.

Table 3.5: Examples of Calibrated HST Image Units

Instrument	Calibrated Data Units	Drizzled Data Units
ACS	e^-	e^-
COS	DN/s	N/A[1]
NICMOS	DN/s	DN/s
STIS	DN	N/A[1]
WFC3/UVIS	e^-	e^-
WFC3/IR	e^-/s	e^-/s
WFPC2	DN	DN/s

1. No drizzled products from the pipeline.

The pipeline calibration tasks do not alter the 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 "counts" may refer to DN 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 DN or electrons (e.g., STIS or WFPC2), 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 signal rates (e.g., 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 and WFC3/IR image headers also contain the keyword PHOTFNU in units of Janskys. Multiplying these images 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 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. However, for WFPC2, new header keywords provide corrections for each chip, in units of magnitude, for contamination (ZP_CORR) and CTE (CTE1E2, CTE_1E3, CTE1E4). 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 × 512. Consult the instrument section (Part II) of the Data Handbook for more information.

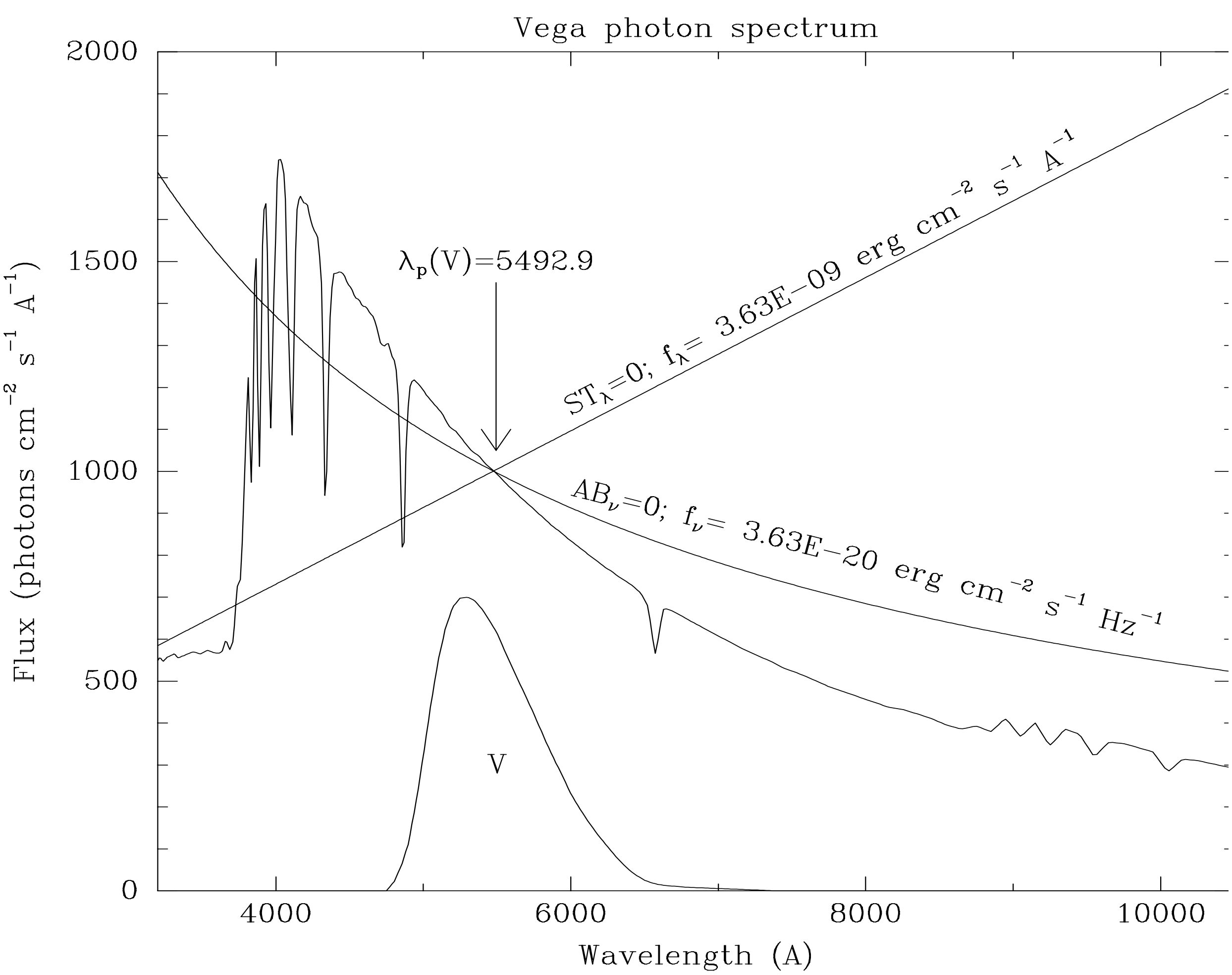
If your HST image contains a source whose flux you know from ground based measurements, you may choose to calibrate the final photometry of your HST image from the counts observed for this source.

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:

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

where the value of the PHOTZPT keyword is the zero point of the ST magnitude (STMAG) scale. The STMAG system is based on 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 Figure 3.4; Koornneef et al., 1986; Horne 1988; and the *Synphot User's Guide*).

Figure 3.4: Standard photometric systems illustrated.



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 sections of this Handbook (Part II).

Synphot

The **STSDAS** synthetic photometry package, called **synphot**, can simulate HST observations of astronomical targets with known spectra. It makes use of a data set that contains throughput curves of all HST optical components, such as mirrors, filters, gratings, apertures, and detectors, and it can generate passband shapes for any combination of these elements. It can also generate synthetic spectra of many different types of sources, including stellar, blackbody, power-law and H II regions, 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. See the *Synphot Data User's Guide* and Appendix A.5 for more information on how to use and obtain the **synphot** data set, 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 is done 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), 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** and has been tested on a representative set of commonly-used ACS, NICMOS, STIS, and WFPC2 observing modes.

The only required input to **MultiDrizzle** is a list of calibrated science images. The user may also choose to provide additional input such as bad pixel masks, a delta-shift file, or a reference image. The script performs the following steps:

- Calculate and subtract a background sky value for each exposure.
- Search for additional bad pixels that are strongly negative, which may not have been flagged in the data quality arrays.
- Determine shifts from the WCS information in the image headers. If the user has supplied a delta-shift file, those shifts are applied in addition to the header offsets. The user may also choose to provide the absolute shifts to use for each image, ignoring the header information.
- Drizzle each input image onto a separate output image. All images remain in the same reference plane such that any pixel (x,y) is the same logical coordinate in each image. During the drizzling process, each image is corrected for any geometric distortion that might be present.
- Combine the separately drizzled images to create a median image, in order to obtain an estimate of the cleaned final image.
- Transform sections of the median image, corresponding to the location of each input image, back to the original distorted input frames and calculate the derivative of each image.
- Compare the original input image against the transformed median and its derivative in order to create a cosmic ray mask for each exposure.
- Combine the cosmic ray masks with any additional user-supplied bad pixel masks and finally drizzle all the original input exposures 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 already been run. 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 operation. However, this may not be the optimal scientific image for a given set of exposures, therefore access is provided to parameters of **drizzle** and other steps for fine-tuning the results.

Please refer to the MultiDrizzle Handbook v3.0 (Fruchter & Sosey et al. 2009), and the online help documentation for MultiDrizzle within **PyRAF**, for further information about the various parameters for the script. The handbook also contains basic examples for each of the current instruments.

In general, the code has been tested on a wide variety of the most commonly used observing modes and should produce useful results with the default parameters. Since the software is actively being improved, it is important to check for updates if you find problems with a particular dataset. Users are encouraged to send e-mail to help@stsci.edu for assistance.

3.5 Displaying HST Spectra

This section describes how to plot the most common HST spectra (COS, FOS, GHRS, and STIS) in **IRAF/STSDAS** for a quick first look.

We will not discuss ACS, NICMOS, or WFC3 grism or prism data. The tools for extracting, displaying and analyzing spectra from these instrument modes are discussed in the instrument sections in Part II (see sections on **aXe** for ACS and WFC3, and **NICMOSlook** for NICMOS).

3.5.1 Specview

Specview[3] is a very useful tool for displaying and analyzing spectra from most HST instrument configurations in their native archival format, as well as data from a variety of other spectral instruments. It is a Java application for 1D interactive spectral visualization and analysis.

Specview was written at STScI in Java (Busko 1999) and is distributed in standalone application and applet formats. The application version requires that either the Java Development Kit (JDK) or Java Runtime Environment (JRE) be installed in your system and accessible from your path.

3. Specview is a product of the Space Telescope Science Institute, which is operated by AURA for NASA.

Specview is capable of overplotting spectra from different instruments, measuring, modelling, and fitting spectral features, spectral line identification, and it allows somewhat elaborate plot annotation. More information about Specview, together with screen shots, demos, and the software for download are available at:

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

3.5.2 COS and STIS Spectra

Both COS and STIS data files retrieved from the HDA can contain spectra in two different forms: as spectral images in FITS IMAGE extensions or as extracted spectra in FITS BINTABLE extensions.

Plotting COS and STIS Imaging Spectra

You can use **sgraph** in the **graphics.stplot** package of **STSDAS** to plot spectral images by specifying the image section that contains the spectrum. For example, to plot the entire *x*-range of the calibrated two-dimensional STIS 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]
```

Similarly, to plot the calibrated two-dimensional COS spectrum in the first extension of the file `l61h54cxr_flt_a.fits`, averaging rows 451 through 480, you would type:

```
st> sgraph l61h54cxr_flt_a.fits[1][*,451:480]
```

Displaying a 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 COS and STIS Tabular Spectra

To plot COS or STIS spectra in BINTABLE extensions, you first need to understand how the spectra are stored as binary arrays in FITS table cells. Section 2.2.2 discusses this format and describes the *selectors* syntax used to specify these data arrays. 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.

COS Row Selector Examples

COS tabular spectra contain two or three rows corresponding to either the FUV segments of the NUV stripes. For example, to select the

WAVELENGTH array corresponding to segment A of the FUV spectrum in extension 1 of `cos_fuv.fits`, you would specify the file as either:

```
cos_fuv.fits[1][c:WAVELENGTH][r:segment=FUVA]
or
cos_fuv.fits[1][c:WAVELENGTH][r:=1]
```

To select the WAVELENGTH array corresponding to stripe C of the NUV spectrum in extension 1 of `cos_nuv.fits`, you would specify the file as either:

```
cos_nuv.fits[1][c:WAVELENGTH][r:segment=NUVC]
or
cos_nuv.fits[1][c:WAVELENGTH][r:=3]
```

STIS Row Selector Examples

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]
```

Plotting Tasks

The **sgraph** task and the **igi** plotting package, discussed below, both understand the row *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"
```

Similarly, to plot flux vs. wavelength in COS segment FUVA, you could type

```
st> sgraph "cos.fits[1][r:segment=FUVA] WAVELENGTH FLUX"
```

Remember to include the quotation marks, otherwise, **sgraph** will complain about too many 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.

3.5.3 FOS and GHRS Spectra

Before working 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 2.3.1 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.

3.5.4 Producing Hardcopy

This section shows how to generate hardcopies of plots directly and describes **igi**, the Interactive Graphics Interpreter available in **STSDAS**. If you are working in the Python/**PyRAF** environment, the plotting library **matplotlib** is available. It uses most of the MATLAB syntax.

Direct Hardcopies

To print a quick copy of the displayed plot from the cl window:

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`.

From a **PyRAF** window, making hardcopies is simpler: just select `print` from the menu at the top of the graphics window.

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:

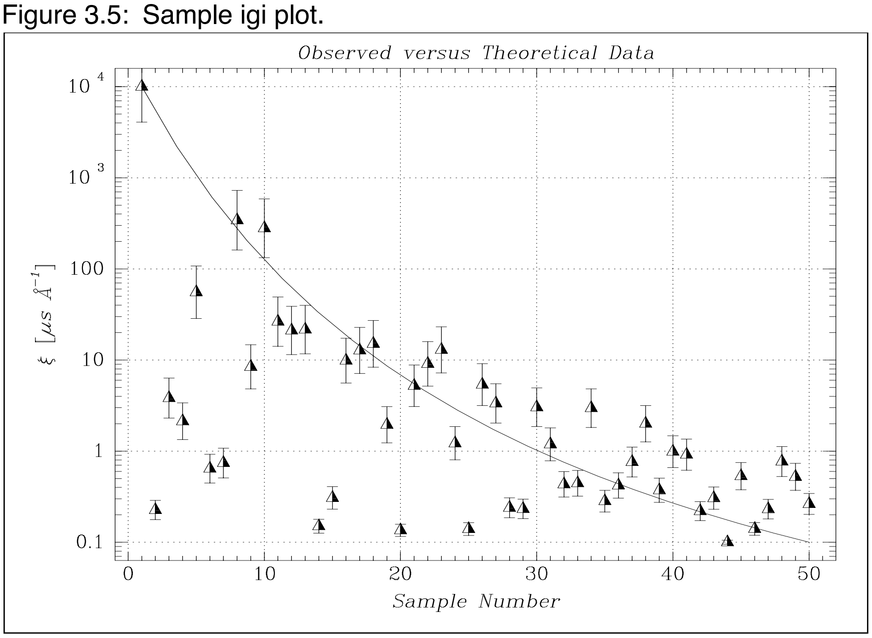
```
st> sgraph o43balbnm_x2d.fits[1][*,100:1000] device=psi_land
st> gflush
/tmp/pskxxx
```

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

igi

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

Figure 3.5: Sample igi plot.



3.6 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 specially prepared data (except for STIS two-dimensional spectra). Before discussing these tasks we will first describe how to recast your data into forms that are more generally accessible.

Alternatively, many other useful tools are now available for the analysis of HST spectra which operate outside of **IRAF**; among them is **specview**, a Java tool for viewing and analyzing spectra from a variety of instruments. It is briefly described in the previous section, "Displaying HST Spectra" on page 25.

3.6.1 Preparing COS and STIS Spectra for Analysis in IRAF or PyRAF and STSDAS

Calibrated spectra emerge from the pipeline either as two-dimensional spectral images (`x2d` STIS files) or as one-dimensional spectra stored in tabular form (`x1d` files for COS and STIS). You can analyze calibrated two-dimensional spectra in **IRAF** as you would analyze any other spectral image, because their headers already contain the necessary wavelength information.

Tabulated COS and STIS spectra can be analyzed directly using **STSDAS** tasks that understand the *selectors* syntax described in Section 2.2.2. 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 analog to **mkmultispec** useful for working with three-dimensional tables. It extracts COS and 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 three-dimensional table cells and places them in conventional two-dimensional tables for easier access.
- **tximage**: This task extracts specified data arrays from a 3-D 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 row (or several) from a COS or 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 3-35). 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 rows from the 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.3.6). OIF format files have a header component (suffix `.imh`) and a binary data component (suffix `.pix`).

If you want to select particular rows, rather than writing all the rows 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, type:

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

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

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

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

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

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 three rows are selected. This restriction results from a limit to the number of keywords that can be used to store the dispersion relation.***

txtable

COS and STIS spectra are stored as data arrays within individual cells of FITS binary tables (see Section 2.2.2). 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
```

Similarly, for a file `cos_nuv.fits` which contains a COS NUV spectrum and you want to extract only the wavelength and flux arrays for all three stripes, type:

```
tt> txtable "cos_nuv.fits[1][c:WAVELENGTH,FLUX][r:row=(1:3)]" \
>>> out_table
```

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, due to the different functional form of the prism-mode dispersion solution. For FOS 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. **tedit** can be used to edit selected rows of a table.

3.6.3 Photometry

Photometric correction of COS, FOS, GHRS, and STIS spectra is done by the pipeline during spectral extraction, resulting in flux-calibrated spectra. For detailed information see Part II of the Data Handbook specific to each of these instruments.

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 3.6 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 3.6: Spectral Analysis Tasks in **IRAF/STSDAS**.

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
calcspec	stsdas.hst_calib.synphot	N/A	Create a synthetic spectrum
continuum	noao.onedspec	Image	Continuum normalize spectra
fitprofs	noao.onedspec	Image	Non-interactive Gaussian profile fitting to features in spectra and image lines
fitspec	stsdas.hst_calib.synphot	table	Fit a model spectrum to an observed spectrum
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	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
plspec	stsdas.hst_calib.synphot	table	Plot calculated and observed 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 3.5.3)
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 3-39)

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 purpose 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 do not 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 3-35).

The **splot** task has *many* available options described in detail in the online help. Table 3.7 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 3-32), 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 3.7: 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 3.8, 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 3-36).

Table 3.8: 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 redraw the plot window, 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 S command) over which the fit will be computed so that the task will not try to fit the entire spectrum.
- Define an initial guess for the baseline coefficients by placing the cursor at two baseline locations (one on either side of the feature to be fitted) using the B keystroke.
- Use the R keystroke to redraw the screen and see the baseline that you've just defined.
- Set the initial guesses for the Gaussian centers and heights by placing the cursor at the peak of each feature and typing P.
- Press F to compute the fit once you've marked all the features you want to fit.

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

Note that when the **ngaussfit** task is used in this way (i.e., starting with all default values), the initial guess for the FWHM of the features will be set to a value of one. Furthermore, this coefficient and the coefficients defining the baseline are held fixed by default during the computation of the fit, unless you explicitly tell the task through cursor *colon* commands[4] 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 3.6 and Figure 3.7 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 3.7.

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

Figure 3.6: Fitting Hβ and [OIII] emission features in NGC 4449.

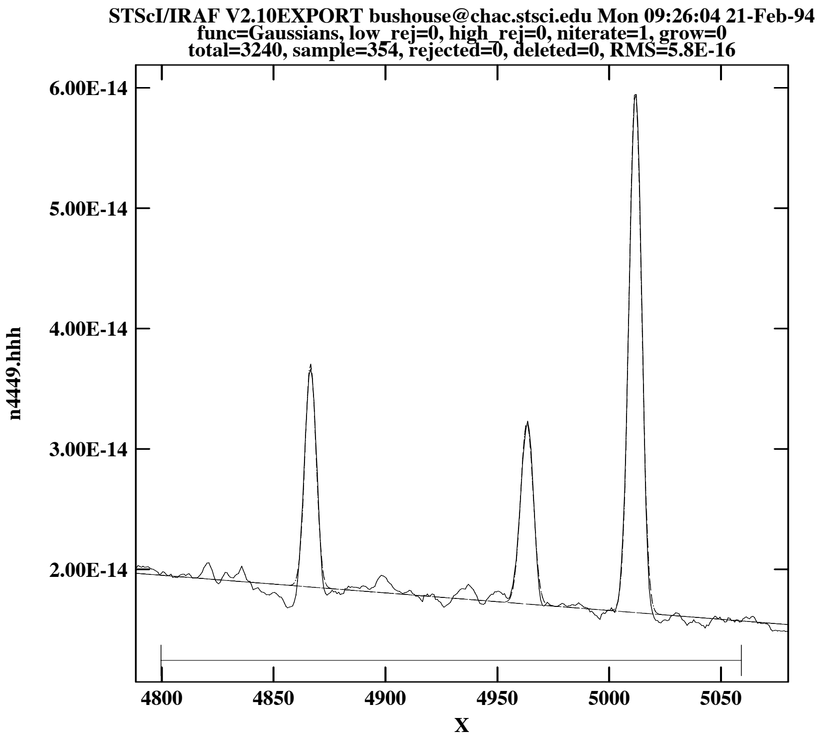


Figure 3.7: Coefficients and error estimates

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

3.6.6 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 at STScI by Gerard Kriss. Extensive online help is available to guide you through the task,[5] 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 User's Guide,* version 1.3, September 1994.
- *STSDAS Site Manager's Installation Guide and Reference*, version 3.9, November 2008.
- *Synphot User's Guide*, version 5.0, November 2005.
- *Synphot Data User's Guide*, version 1.2, June 2008.
- *IGI Reference Manual*, version 2.0, September 1998.

3.7.2 Available from NOAO

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

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

- *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. Massy and L. Davis.

3.7.3 Other References Cited in This Chapter

- Busko, I., 1999, "A Java Graphics Package to Support specview", available at http://specview.stsci.edu/design/design6.ps
- Busko, I. 2002, "Specview: a Java Tool for Spectral Visualization and Model Fitting", Astronomical Data Analysis Software and Systems XI, ASP Conference Series, v.281, p.120, ed. Bohlender,D.A., Durand, D., and Handley,T.H.
- Fruchter, A. S. & Hook, R. N. 2002, PASP 114, 144.
- Fruchter, A. S. & Sosey, M. et al. 2009, "The MultiDrizzle Handbook", v3.0.
- 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), p. 337.
- 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
- numpy.scipy.org/

PART II:

COS Data Handbook

In this part. . .

This handbook describes data from the Cosmic Origins Spectrograph (COS) to be installed on the Hubble Space Telescope (HST) in 2009, and how to manipulate, calibrate, and analyze those data.

Introduction

How to Use this Handbook

This handbook is designed to help users manipulate, process, and analyze data from the Cosmic Origins Spectrograph (COS) which will be installed on the Hubble Space Telescope (HST) during the 2009 servicing mission (SM4). It is designed for users familiar with HST data but new to COS. Users who wish to find more general information including instructions for retrieving data from the HST archive, a description of HST file formats, and a discussion of PyRAF/IRAF/STSDAS software for displaying and processing these data, are referred to Part I, the Introduction to the HST Data Handbook.

For detailed information on the capabilities of the instrument, and how to plan observations, users should refer to the COS Instrument Handbook. For further information and timely updates, users should consult the COS Web page (http://www.stsci.edu/hst/cos), especially the Document Archive link. In particular, the STScI Analysis Newsletters (STANs) highlight changes in code and calibration procedures and provide other instrument-related news. The Instrument Science Reports (ISRs) present in-depth characterizations of the instrument and detailed explanations of calibration code and procedures.

The current edition of the COS Data Handbook was completed in early 2009. At present, preparations are being made for Servicing Mission 4 to *HST*, which includes installation of COS.

Handbook Structure

The COS Data Handbook is organized in five chapters, which discuss the following topics:

- Chapter 1:COS Overview provides a brief overview of the instrument and its operational capabilities. If you are not already familiar with the details of COS, you should begin here.

- Chapter 2:COS Data Files describes the contents of COS data files, the meanings of selected header keywords, and the relationship of the data products to the original Phase II proposal. If you are not familiar with the filenames, header keywords, or contents of the data files from COS, you should read this chapter next.
- Chapter 3:COS Calibration describes how the calibration pipeline processes your observation, the content of refernce files used during calibration of your observation and how to run the calibration pipeline from home. If you are not familiar with the important characteristics of COS data and the standard procedures for reducing them, or do not know how your data have been calibrated, you should read this chapter.
- Chapter 4:COS Error Sources describes the sources of uncertainty and limiting accuracies of COS data, with brief discussions of possible on-orbit instrumental phenomena. COS observers should read this chapter to acquaint themselves with the limitations of the data that may remain after pipeline calibration.
- Chapter 5:COS Data Analysis describes certain **IRAF/PyRAF/STSDAS** tasks, and other software packages useful for optimizing data products and analyzing the data. In particular, it discusses software tools that can be applied to specific types of data and data formats. It describes how to analyze target acquisitions and guide star tracking. It provides descriptions of different kinds of data and gives detailed instructions on how to work with them; specifically: extracted spectra, and TIME-TAG data. Most observers will find this chapter useful when determining how they should reduce and analyze their data.

There are some important pieces of general information about *HST* data, the *HST* Archive, and the **IRAF** and **STSDAS** analysis software that are not specific to the COS, and which are therefore not discussed in the COS specific section, Part II, of this *Data Handbook*. We refer the reader to the most recent version of the companion Introduction to *HST* Data Handbook for this information. In particular, Chapter 1, Chapter 2 and Chapter 3 of the Introduction to *HST* Data Handbook describe how to retrieve and read *HST* data, *HST* file formats, and the basics of the **STSDAS** software package. Appendix A: IRAF Primer offers an **IRAF** primer. Appendix B: HST File Names describes *HST* file name conventions and exposure "associations". Appendix C: Observation Logs describes *HST* Observation Logs. Additional help with *HST* data is always available via email to the STScI Help Desk at `help@stsci.edu`.

Since many of the instrument characteristics may be revised over a short time frame after SM4, readers are advised to consult the COS Web pages (http://www.stsci.edu/hst/cos/) for the latest.

Typographic Conventions

To help you understand the material in the *COS Data Handbook*, we will use a few consistent typographic conventions.

Visual Cues

The following typographic cues are used:

- **bold** words identify an **STSDAS**, **IRAF**, or **PyRAF** task or package name.
- `typewriter-like` words identify a file name, system command, or response that is typed or displayed as shown.
- *italic* type indicates a new term, an important point, or a mathematical variable, or a task parameter.
- SMALL CAPS identifies a header keyword.
- ALL CAPS identifies a table column.

Comments

Occasional side comments point out three types of information, each identified by an icon in the left margin.

Warning: When you see this symbol, you could corrupt data, produce incorrect results, or create some other kind of severe problem.

Heads Up: Here is something that is often done incorrectly or that is not obvious.

Tip: No problems...just another way to do something or a suggestion that might make your life easier.

CHAPTER 1:

COS Overview

In this chapter...

1.1 Instrument Capabilities and Design

The Cosmic Origins Spectrograph (COS) is an *HST* fourth generation spectrometer, designed to enhance the spectroscopic capabilities of *HST* at ultraviolet (UV) wavelengths. COS was built by Ball Aerospace Corporation to the specifications of Dr. James Green, the Principal Investigator (PI), at the University of Colorado at Boulder in conjunction with the COS Instrument Definition Team (IDT). Designed to primarily observe faint point sources, COS is optimized for maximum throughput, and provides moderate and low resolution spectroscopy in the UV and limited imaging in the NUV.

COS is a slitless spectrograph that employs two circular 2.5 arcsec diameter science apertures, the Primary Science Aperture (PSA) and the Bright Object Aperture (BOA). The PSA is an open aperture and the BOA contains a neutral density filter to attenuate the flux of bright objects. COS also contains two calibrations apertures, the Wavelength Calibration Aperture (WCA) and the Flat-Field Calibration Aperture (FCA). Light from external sources does not reach these apertures. Instead they are illuminated by internal calibration lamps. The FCA is not available for observers, but the WCA can be used by observers to obtain wavelength calibration spectra. The WCA can be illuminated by one of two Pt-Ne wavelength calibration lamps. Similarly, the FCA can be illuminated by one of two deuterium flat-field calibration lamps, however, this is restricted to observatory calibration programs.

The instrument has two channels: a far-ultraviolet (FUV) channel that is sensitive across the 1150-2050 Å wavelength range and a near-ultraviolet (NUV) channel that provides wavelength coverage from 1750-3200 Å. The COS optical design achieves its high performance, particularly in the FUV, by minimizing the number of reflections in the optical path and the use of large format detectors which maximize the wavelength coverage per exposure. Each channel has its own photon-counting detector and a selection of gratings (Table 1.1). The NUV channel also has a mirror that can be used in two modes for imaging. The FUV channel uses a single reflection system where a high-efficiency, first-order, aspheric holographic grating completely corrects the beam in the dispersion direction but has low spatial resolution perpendicular to dispersion. *Only one channel may be used at a time*.

Table 1.1: COS Spectroscopic Modes

Grating	Normal wavelength range (Å)[1]	Bandpass per exposure (Å)	Resolving Power $R = \lambda/\Delta\lambda$	Dispersion (mÅ pixel^{-1})
FUV Channel				
G130M	1150 – 1450	300	20,000 – 24,000	9.4
G160M	1405 – 1775	370	20,000 – 24,000	11.8
G140L	1230 – 2050	>820	2,500 – 3,000	86.5
NUV Channel				
G185M	1700 – 2100	3 × 35	16,000 – 20,000	34
G225M	2100 – 2500	3 × 35	20,000 – 24,000	34
G285M	2500 – 3200	3 × 41	20,000 – 24,000	40
G230L	1700 – 3200	(1 or 2) × 400	1,700 – 3,200	389

1. Normal wavelength ranges are for the primary (default) central wavelength setting.

FUV Spectroscopy

The FUV channel employs a large format cross delay line (XDL) detector consisting of two 16384 x 1024 pixel segments, referred to as FUV segments A and B. The segments are separated by a physical gap of 9 mm, which makes it impossible to obtain a continuous spectrum across the two segments with a single setting. The supported central wavelength positions were selected to enable full wavelength coverage of the gap. Table 1.2 shows the wavelength ranges of both segments for all possible FUV grating and central wavelength combinations.

Table 1.2: Wavelength Ranges for FUV Gratings

Grating	Central wavelength setting (Å)	Recorded wavelengths	
		Segment B	Segment A
G130M	1291	1132 – 1274	1291 – 1433
	1300	1141 – 1283	1300 – 1442
	1309	1150 – 1292	1309 – 1451
	1318	1159 – 1301	1318 – 1460
	1327	1168 – 1310	1327 – 1469
G160M	1577	1382 – 1556	1577 – 1752
	1589	1394 – 1568	1589 – 1764
	1600	1405 – 1579	1600 – 1775
	1611	1416 – 1590	1611 – 1786
	1623	1428 – 1602	1623 – 1798
G140L[1]	1105	N/A[2]	1105 – 2253
	1230	<300 – 1095	1230 – 2378

1. It is not yet clear how much of the G140L segment B short wavelength (< 1095 **Å**) ranges will be available due to uncertainties in the HST OTA throughput. This will be investigated on-orbit.
2. The G140L grating and 1105 central wavelength setting moves the zero-order image onto segment B. Therefore, only segment A is available for this setting.

NUV Spectroscopy

To retain efficiency utilizing the square format of the NUV detector, three mirrors simultaneously image three, fully aberration-corrected, spectra on a single 1024 x 1024 Multi-Anode Micro-channel Array (MAMA) detector. Consequently, three separate regions of the spectrum are imaged onto the detector. These spectral regions, referred to as stripes A, B, and C, each span the physical length of the detector in the dispersion direction - but are not contiguous in wavelength space. The allowable grating positions were defined with two objectives: the capability of obtaining full spectral coverage over the NUV bandpass and maximizing scientific return with a minimum number of grating positions. As a result, several of the supported central wavelength positions were selected to maximize the number of diagnostic lines on the detector in a single exposure. Table 1.3 shows the wavelength ranges of the three stripes for all possible NUV grating and central wavelength combinations

Table 1.3: Wavelength ranges for NUV gratings

Grating	Central wavelength setting (Å)	Recorded wavelengths		
		Stripe A	Stripe B	Stripe C
G185M	1786	1670 – 1705	1769 – 1804	1868 – 1903
	1817	1701 – 1736	1800 – 1835	1899 – 1934
	1835	1719 – 1754	1818 – 1853	1916 – 1951
	1850	1734 – 1769	1833 – 1868	1931 – 1966
	1864	1748 – 1783	1847 – 1882	1945 – 1980
	1882	1766 – 1801	1865 – 1900	1964 – 1999
	1890	1774 – 1809	1872 – 1907	1971 – 2006
	1900	1783 – 1818	1882 – 1917	1981 – 2016
	1913	1796 – 1831	1895 – 1930	1993 – 2028
	1921	1804 – 1839	1903 – 1938	2002 – 2037
	1941	1825 – 1860	1924 – 1959	2023 – 2058
	1953	1837 – 1872	1936 – 1971	2034 – 2069
	1971	1854 – 1889	1953 – 1988	2052 – 2087
	1986	1870 – 1905	1969 – 2004	2068 – 2103
	2010	1894 – 1929	1993 – 2028	2092 – 2127
G225M	2186	2070 – 2105	2169 – 2204	2268 – 2303
	2217	2101 – 2136	2200 – 2235	2299 – 2334
	2233	2117 – 2152	2215 – 2250	2314 – 2349
	2250	2134 – 2169	2233 – 2268	2332 – 2367
	2268	2152 – 2187	2251 – 2286	2350 – 2385
	2283	2167 – 2202	2266 – 2301	2364 – 2399
	2306	2190 – 2225	2288 – 2323	2387 – 2422
	2325	2208 – 2243	2307 – 2342	2406 – 2441
	2339	2223 – 2258	2322 – 2357	2421 – 2456
	2357	2241 – 2276	2340 – 2375	2439 – 2474
	2373	2256 – 2291	2355 – 2390	2454 – 2489
	2390	2274 – 2309	2373 – 2408	2472 – 2507
	2410	2294 – 2329	2393 – 2428	2492 – 2527

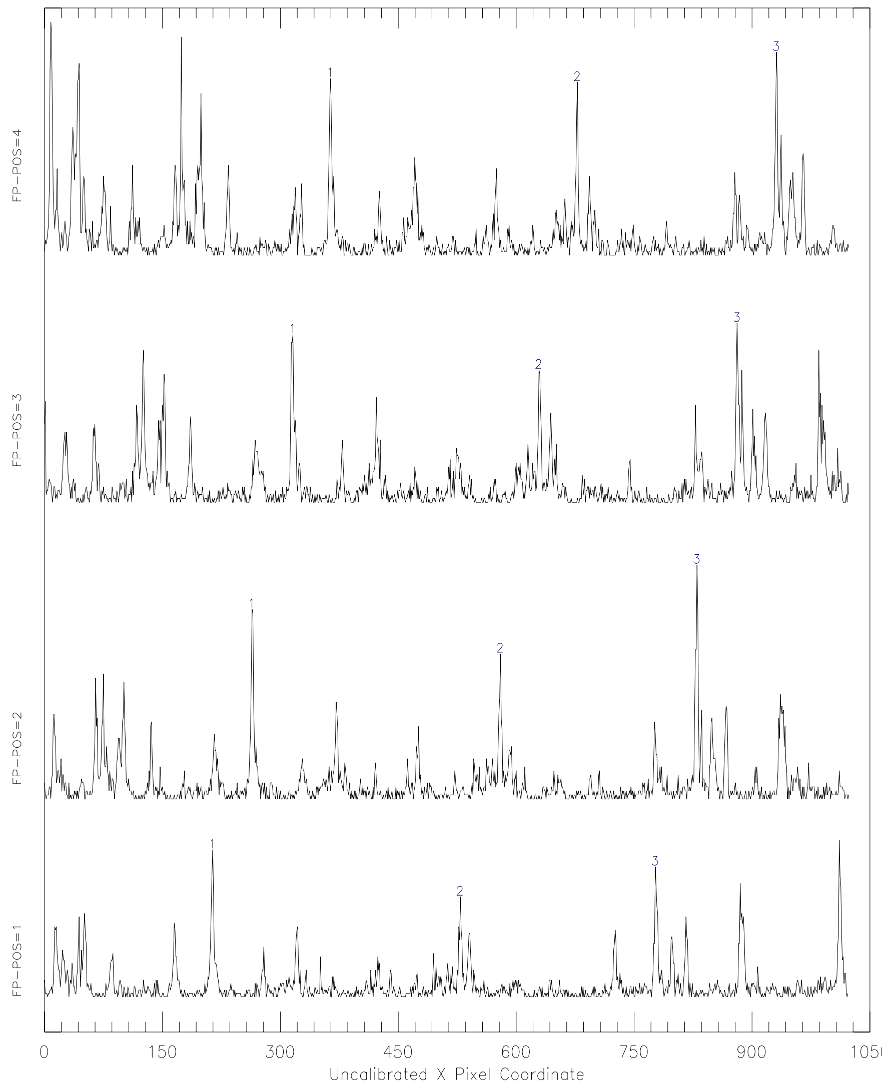
Grating	Central wavelength setting (Å)	Recorded wavelengths		
		Stripe A	Stripe B	Stripe C
G285M	2617	2480 – 2521	2596 – 2637	2711 – 2752
	2637	2500 – 2541	2616 – 2657	2731 – 2772
	2657	2520 – 2561	2636 – 2677	2751 – 2792
	2676	2539 – 2580	2655 – 2696	2770 – 2811
	2695	2558 – 2599	2674 – 2715	2789 – 2830
	2709	2572 – 2613	2688 – 2729	2803 – 2844
	2719	2582 – 2623	2698 – 2739	2813 – 2854
	2739	2602 – 2643	2718 – 2763	2837 – 2878
	2850	2714 – 2755	2829 – 2870	2945 – 2986
	2952	2815 – 2856	2931 – 2972	3046 – 3087
	2979	2842 – 2883	2958 – 2999	3073 – 3114
	2996	2859 – 2900	2975 – 3016	3090 – 3131
	3018	2881 – 2922	2997 – 3038	3112 – 3153
	3035	2898 – 2939	3014 – 3055	3129 – 3170
	3057	2920 – 2961	3036 – 3077	3151 – 3192
	3074	2937 – 2978	3053 – 3094	3168 – 3209
	3094	2957 – 2998	3073 – 3114	3188 – 3229
G230L	2635	1334 – 1733[1]	2435 – 2834	**1768 – 1967**[2]
	2950	1650 – 2050	2750 – 3150	**1900 – 2100**[2]
	3000	1700 – 2100	2800 – 3200	**1950 – 2150**[2]
	3360	2059 – 2458	3161 – 3560	**2164 – 2361**[2]

1. The wavelengths listed for central wavelength 2635 Å in stripe A are listed for completeness only and also in case a bright emission line falls onto the detector. Note that the NUV detector's sensitivity at these wavelengths is extremely low. To obtain a low-resolution spectrum at wavelengths below about 1700 Å we recommend G140L and the FUV channel.
2. The values in shaded cells are wavelength ranges as seen in second-order light. In these cases the achieved dispersion is twice that for first-order mode.

Grating Offset Positions (FP-POS)

For each central wavelength setting there are four grating offset positions (`FP-POS`=1-4) available to move the spectrum slightly in the dispersion direction. This allows the spectrum to fall on different areas of the detector to minimize the effects of small scale fixed pattern noise in the detector. Figure 1.1 shows the shifts in uncalibrated x pixel coordinates of the stripe B spectra for all four FP-POS positions.

Figure 1.1: Grating offset positions (FP-POS)



This figure shows spectra obtained at all four FP-POS positions using the G185M grating with a central wavelength setting of 1850. The individual plots show the collapsed counts from the stripe B spectra versus the uncalibrated x pixel coordinates. Note that the three features marked 1, 2, and 3, shift slightly for each FP-POS position.

NUV Imaging

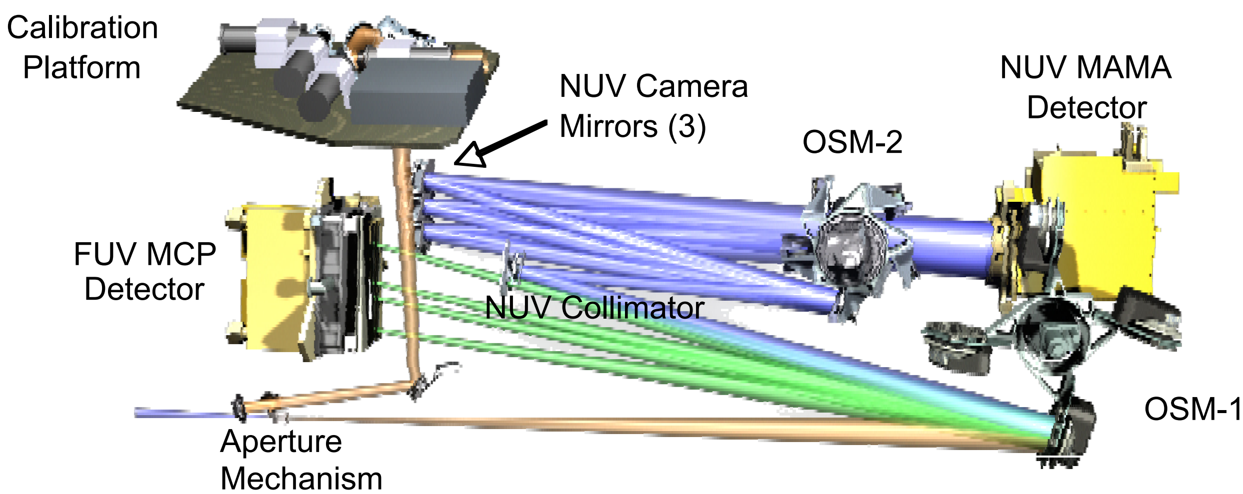
COS imaging may only be done with the NUV channel and the spectral coverage includes the entire NUV bandpass from ~1650-3200 Å. This mode utilizes a flat mirror with two available mirror settings, MIRRORA and MIRRORB. The first setting uses a primary reflection off the mirror surface, and the second setting provides an attenuated reflection. MIRRORB and/or the BOA may be used to obtain images of brighter objects, but MIRRORB produces a secondary image and the BOA produces an image with coma that degrades the spatial resolution (Figure 5.2 and Figure 5.3). While the spatial resolution of COS NUV MIRRORA images can be quite good, the field of view is very small. Furthermore, because the optics image the sky onto the detector – not the aperture – the image includes some light from sources out to a radius of about 2 arcsec. However, only point sources within about 0.5 arcsec of the aperture center have essentially all their light imaged, and so the photometric interpretation of a COS image can be inherently complex.

Data Collection Modes

COS has two modes of data collection, TIME-TAG and ACCUM, and only one mode can be used for a given observation. In TIME-TAG mode the position, time and, for FUV, pulse height of each detected photon are tabulated into an events list, while in ACCUM mode the photon events are integrated on-board into an image. TIME-TAG data has a time resolution of 32 ms, and can be screened as a function of time during the post-observation pipeline processing to modify temporal sampling and exclude poor quality data. COS is optimized to perform in TIME-TAG mode, although ACCUM mode is fully supported in the pipeline processing. ACCUM mode should be used primarily for UV bright targets that can not be observed in TIME-TAG mode due to high count rates. Users should note that FUV data taken in ACCUM mode use sub-arrays since the 18MB of on-board memory cannot hold a complete FUV image (containing both detector segments). The FUV ACCUM subarrays, which are 16384 x 128, sizes are shown in Figure 2.2.

1.2 COS Physical Configuration

Figure 1.2: The COS optical path and the locations of the mechanisms.

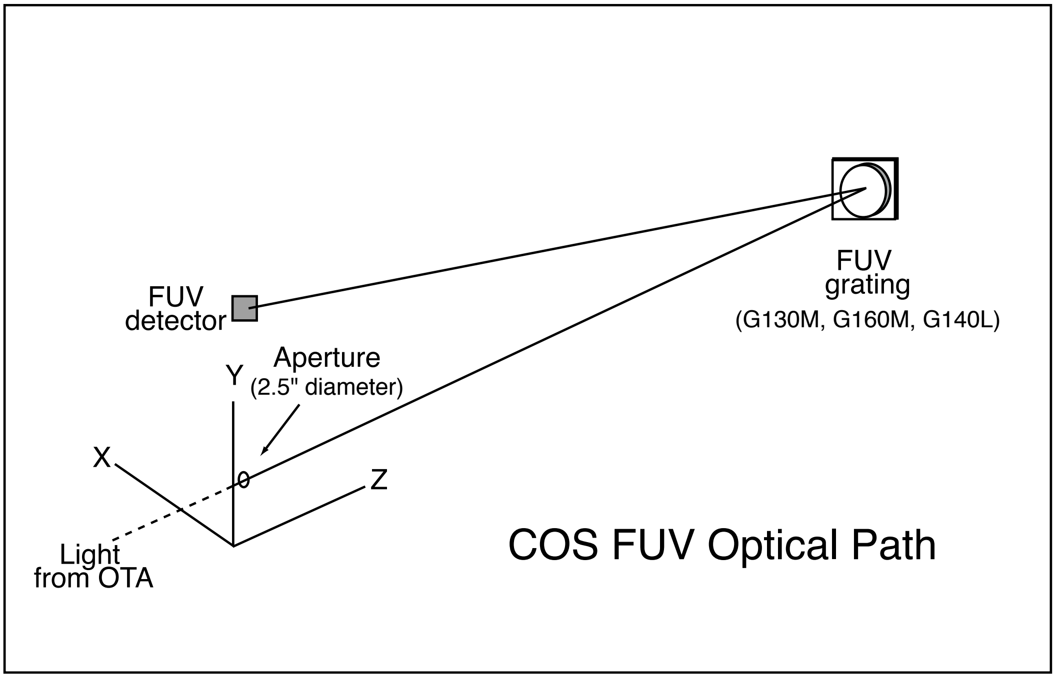


Drawn to scale, with all elements in proportion and in their correct relative locations.

The COS optical design includes an external shutter, two science apertures, two calibration apertures, two optic select mechanisms (OSM1 and OSM2), and separate NUV and FUV detectors. COS also has an independent calibration lamp assembly containing two Pt-Ne and two deuterium lamps, which can illuminate the detectors with a continuum spectrum or an emission line spectrum.

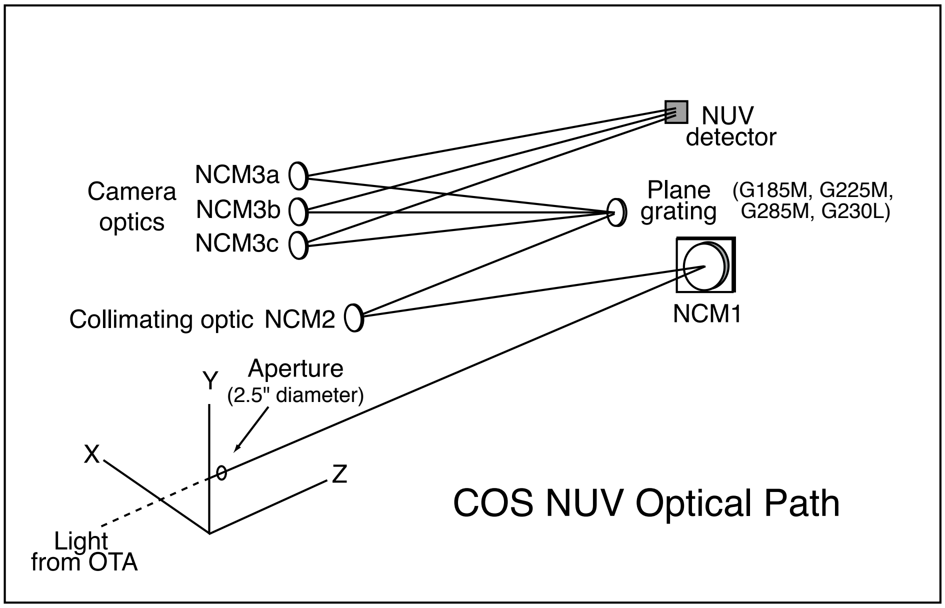
External light enters the aperture mechanism through either the PSA or the BOA and illuminates OSM1, which contains the three FUV gratings and a mirror. Each grating can be set to one of several positions, to obtain different wavelength ranges. The positioning of the OSM1 mechanism is not precisely repeatable, and this can cause small, but significant, variations in how the spectrum or image is projected onto the detector. This non-repeatability can be corrected in post-observation data processing using separate or concurrent (TAGFLASH) calibration lamp exposures (wavecals). The FUV gratings correct for aberration in the dispersion direction only, and disperse the incoming light onto the FUV XDL detector. The COS FUV channel optical path is illustrated in Figure 1.3

Figure 1.3: The COS FUV Optical Path.



If the OSM1 is set to the mirror position, the incoming light is directed to a collimating mirror, and then to OSM2, which contains a mirror for imaging and the four NUV gratings. Each grating offers multiple positions. As is the case with OSM1, the positioning of OSM2 does not repeat exactly, and the data needs to be corrected in post-observation data processing via either separate or concurrent wavecals. If a grating is in place on OSM2, the dispersed light is imaged onto the NUV detector by three separate parallel camera mirrors (NCM3a, b, c). This results in three spectra, or stripes, covering different wavelength ranges. Full wavelength coverage may be obtained through multiple observations with different grating positions. Alternatively, if the plane mirror is in place on OSM2, the undispersed light is sent to the middle camera mirror (NCM3b) and then imaged onto the NUV detector. The plane mirror on OSM2 may be used in either of two settings, designated as MIRRORA and MIRRORB. The MIRRORA setting employs a direct reflection from the plane mirror. For the MIRRORB setting, the plane mirror is slightly offset to provide primary reflection off the front surface of its coating and hence an attenuation factor of approximately 25 compared to the MIRRORA setting. The COS NUV channel optical path is illustrated in Figure 1.4

Figure 1.4: The COS NUV Optical Path.



A series of beam-splitters and fold mirrors direct light from the calibration lamp assembly (see Figure 1.2), through either the WCA or FCA and into the optical path. The calibration lamp assembly can provide continuum illumination to the NUV detector with its Deuterium lamps, and emission line illumination to both the NUV and FUV detectors with its Pt-Ne lamps. The Pt-Ne lamps may be operated during science exposures in order to produce concurrent wavelength calibrations (`TAGFLASH` mode).

1.2.1 The COS Detectors

COS uses two detectors, a FUV XDL and a NUV MAMA. Table 1.4 gives an overview of their characteristics.

Table 1.4: COS Detector Characteristics

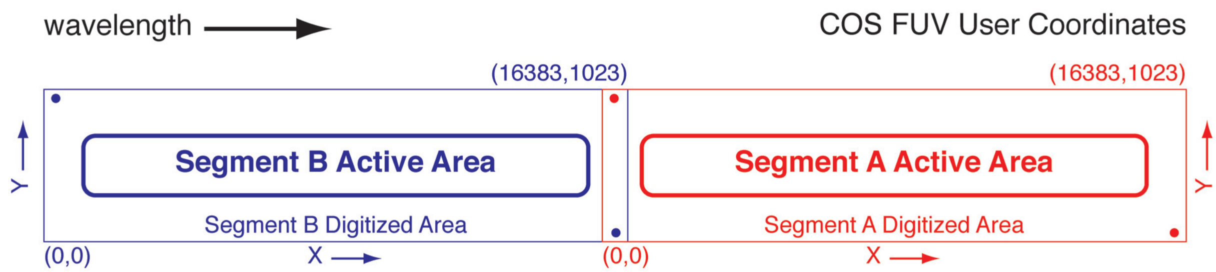
Detector Characteristic	FUV XDL	NUV MAMA
Photocathode	CsI (opaque)	Cs_2Te (semi-transparent)
Window	None	MgF_2 (re-entrant)
Wavelength range	1150 – 2050 Å	1700 – 3200 Å
Active area	85 × 10 mm[1]	25.6 × 25.6 mm
Pixel format (full detector)	16384 × 1024[1]	1024 × 1024
Image size recorded per spectrum	16384 × 128 (ACCUM)[1] 16384 × 1024 (TIME-TAG)[1]	1024 × 1024
Pixel size	6 × 24 µm 0.023 × 0.092 arcsec	25 × 25 µm 0.025 × 0.025 arcsec
Spectral resolution element size (= "resel")	6 × 10 pix	3 × 3 pix
Plate scale: Along dispersion (per resel)	0.13 arcsec	0.075 arcsec
Plate scale: Cross dispersion (per resel)	0.92 arcsec	0.075 arcsec
Plate scale: Imaging (per resel)	N/A	0.075 arcsec
Quantum efficiency	~26% at 1335 Å ~12% at 1560 Å	~10% at 2200 Å ~8% at 2800 Å
Dark count rate	~0.5 ct s^{-1} cm^{-2} ~7.2x10^{-7} ct s^{-1} pix^{-1} ~4.3x10^{-5} ct s^{-1} $resel^{-1}$	~34 cnt s^{-1} cm^{-2} ~2.1x10^{-4} cnt s^{-1} pix^{-1} ~1.9x10^{-3} cnt s^{-1} $resel^{-1}$

1. Sizes given are for an individual FUV segment.

FUV Channel

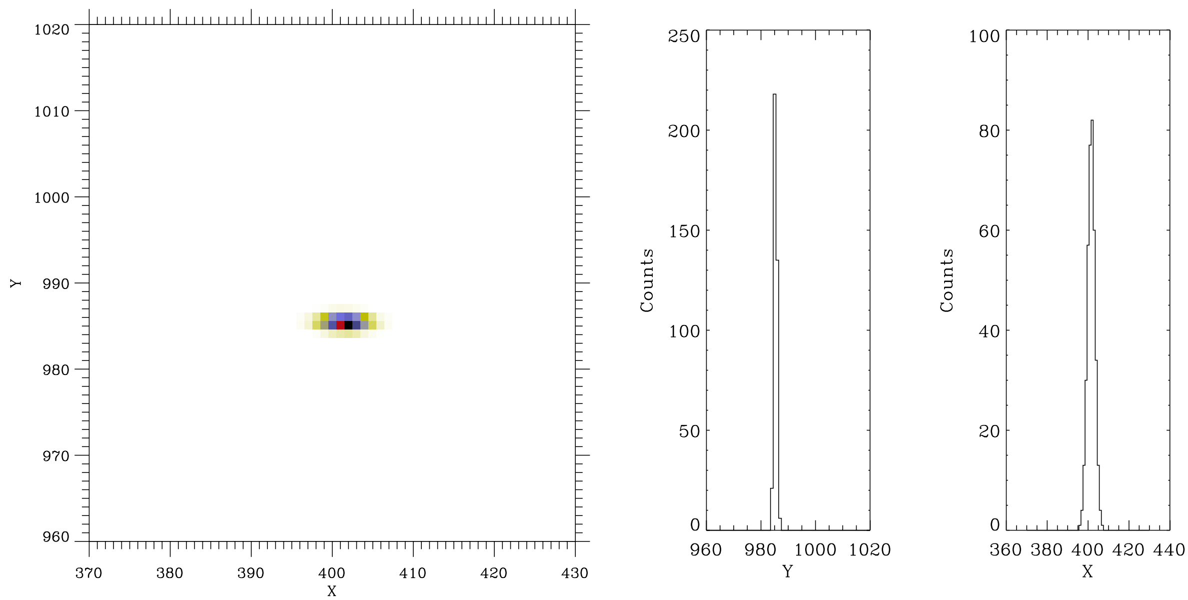
The FUV channel uses a large-format, windowless solar-blind cross delay line (XDL) detector. This is a two-segment photon-counting detector with microchannel plates feeding a XDL anode. The data are digitized to a 16384 x 1024 pixel format for each segment, however the active area is only 14200 x 540 for Segment A and 14150 x 400 for Segment B. Because there are no physical pixels, fiducial electronic pulses are recorded at specific times throughout an observation to permit alignment of data to a standard reference frame. These electronic pulses are referred to as "stim pulses". Figure 1.5 schematically shows the COS FUV XDL segments with the locations of the active areas and stim pulses. When active, the stim pulses emulate counts located near the edges of the anode, beyond the illuminated portions of the detector. A zoomed-in image of one of the FUV stim pulses on segment B is shown in Figure 1.6. An example of an FUV external science spectrum taken with Segment B is shown Figure 1.7, with a simultaneous wavelength calibration spectrum.

Figure 1.5: The FUV XDL Detector.



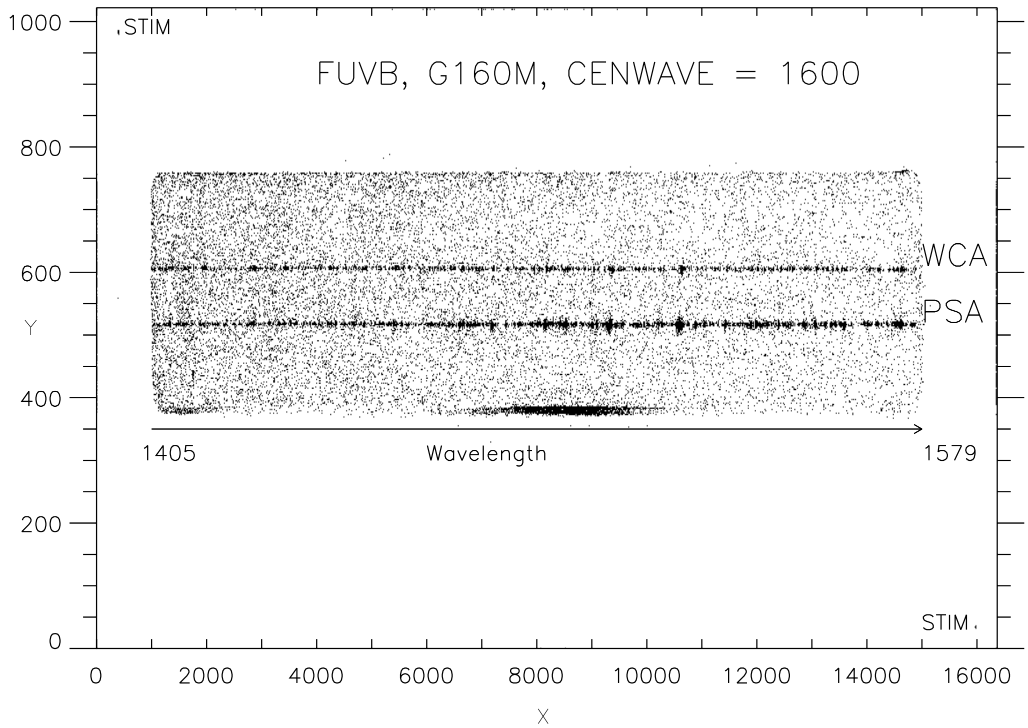
Drawn to scale. The slight curvature at the corners of the active areas is also present on the flight detectors. The red and blue dots show the approximate locations of the stim pulses. The numbers in parentheses are the pixel coordinates at the corners of the segment's digitized area.

Figure 1.6: COS FUV Stim Pulse



Left: A portion of an image in the FUV detector with a typical stim pulse is shown. Right: A histogram of the stim pulse profile in the x and y direction. The electronic stim pulses are used to remove thermal distortions and to map the XDL detector elements to a standard reference frame.

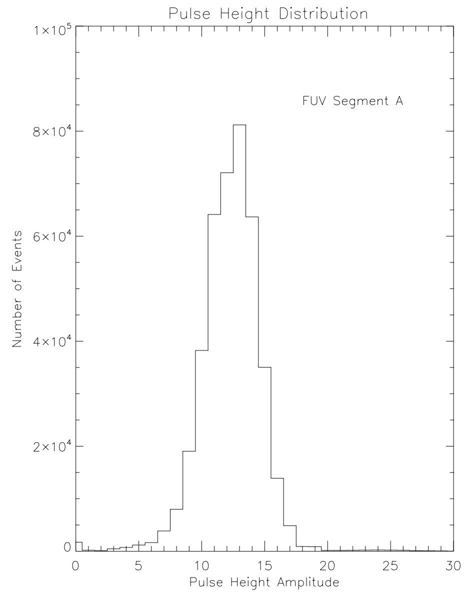
Figure 1.7: Example of a COS FUV Spectrum.



Wavelength calibration spectra for FUV segment B with G160M at 1600 obtained during ground testing. The upper spectrum is from the internal wavelength calibration lamp obtained through the WCA. The lower spectrum is from an external lamp obtained through the PSA. The bright streak at the bottom is due to an area of enhanced background on the detector segment. Note that the size of the active area is somewhat less than the overall digitized area, and that the *Y* axis has been stretched. The STIMs are also visible in the upper left and lower right corners.

With each recorded event on the XDL detector, the total charge in the associated electron cloud incident on the anode is recorded. For FUV TIME-TAG data this pulse height amplitude (PHA) is sent to the ground along with the position of the event and can be used during data analysis to identify non-photon events. For FUV XDL `ACCUM` mode data, only an integrated pulse height distribution (a histogram of the PHA data) for the entire segment is available, see Figure 1.8.

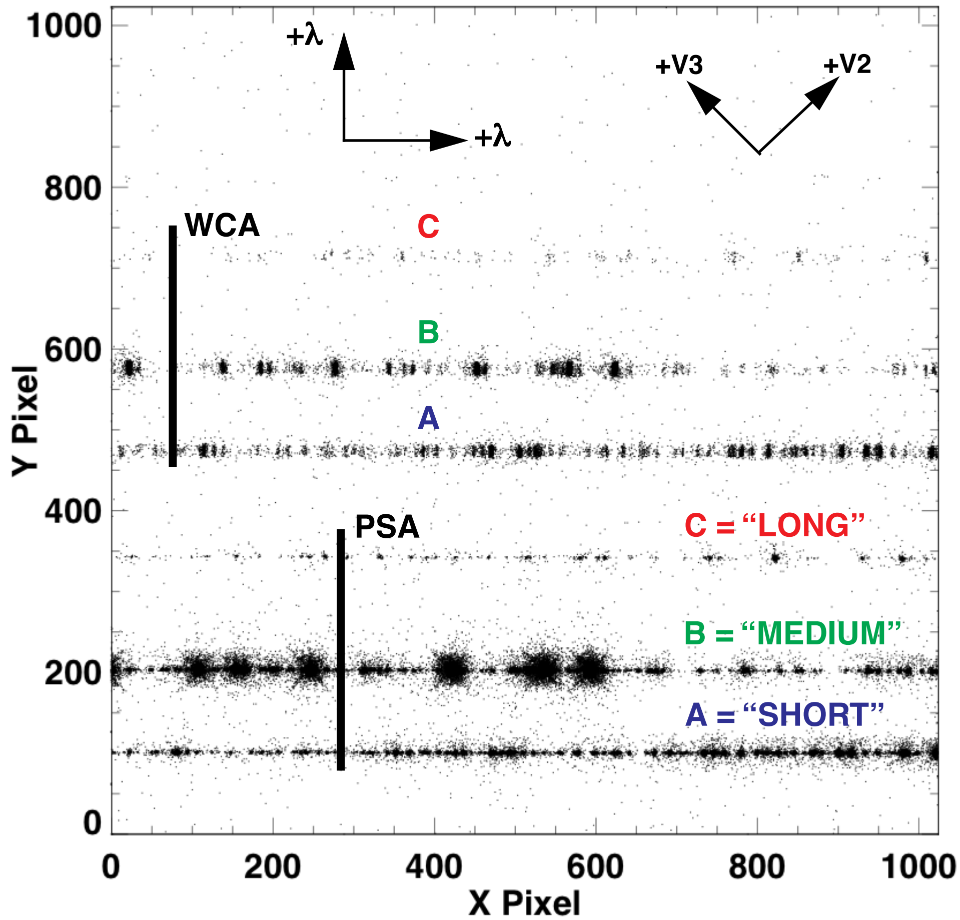
Figure 1.8: Example of a COS FUV Pulse Height Distribution



NUV Channel

The NUV channel uses a 1024 x 1024 pixel Multi-Anode Micro-channel Array (MAMA) detector. This has a semi-transparent cesium telluride photocathode on a magnesium fluoride window, which allows detection of photons with wavelengths from 1150 to 3200 Å. The NUV MAMA provides no pulse-height information, but may be used in both `ACCUM` and `TIME-TAG` mode. The NUV channel creates three spectrum stripes on the MAMA detector, resulting in three separate stripes for the science data and three for wavelength calibration data as shown in Figure 1.9.

Figure 1.9: Example of a COS NUV Spectrum.



Wavelength calibration spectra obtained from the internal source through the WCA (upper three stripes) and an external source through the PSA (lower three stripes). The stripes are designated A, B, and C, in going from bottom to top for each source. Wavelength increases from left to right in each stripe and from bottom to top (hence the SHORT, MEDIUM, and LONG designations).

1.3 Basic Instrument Operations

1.3.1 Target Acquisitions

The details of acquiring objects with COS are described in Chapter 7 of the *COS Instrument Handbook*. In brief, the COS flight software provides several methods for acquiring and centering a target in the aperture in both imaging and dispersed light modes. The simplest and fastest method uses the ACQ/IMAGE command to obtain a direct NUV image of the target field and then moving the telescope to the centroid of the measured light. This is the preferred method, but the target coordinates must be accurate enough to ensure that it falls within the aperture after the initial pointing of the telescope. With less accurate coordinates, or for all cycle 17 programs, a spiral search (ACQ/SEARCH) should be performed with either detector prior to other acquisition methods to ensure the target will fall within the aperture. The other COS acquisition methods (ACQ/PEAKXD and ACQ/PEAKD) use dispersed light from the target, and can also be performed with either detector.

1.3.2 Routine Wavecals

Routine wavelength calibration exposures, or wavecals, are needed by the COS calibration pipeline, **calcos**, to compensate for the effects of OSM drifts. All wavelength calibration exposures are taken in TIME-TAG mode. They may be obtained in either the TAGFLASH mode, where FLASH=YES for TIME-TAG science observations, or in separate wavelength calibration exposures that are either automatic or user-specified.

For TAGFLASH exposures, the wavecal lamp is turned on briefly at the start of an externally targeted exposure, and again at predefined intervals throughout the exposure. In this mode, photons from the external science target and the internal wavelength calibration source are recorded simultaneously on different portions of the detector; see Figures 1.7 and 1.9.

For TIME-TAG exposures not done in TAGFLASH mode, a separate wavecal exposure will be automatically performed (AUTO wavecal) for each set of external spectrographic science exposures using the same spectral element, central wavelength, and FP-POS value. These automatic wavecals are performed after the first such science exposure and after each subsequent science exposure if more than 40 minutes of visibility time has elapsed since the previous wavecal and the same spectrograph set-up has been in use over that time.

Observers also have the ability to insert additional wavecals by specifying TARGET=WAVE (GO wavecal). These exposures will use the same calibration lamp configurations and exposure times as the automatic wavecals. There only way to tell the difference between GO and automatic wavecal data is to look at the MEMTYPE header keyword, which will be discussed later in Table 2.6 of the “Association Tables (ASN)” Section.

1.3.3 Typical COS Observing Sequence

For most observations, the following sequence of events occurs:

- Acquire the object using `ACQ/IMAGE` with the NUV detector. This may be preceded by an `ACQ/SEARCH` if needed to scan a larger area of sky. In cycle 17 we recommend all acquisitions to start with an `ACQ/SEARCH` exposure.
- Obtain a spectra in `TIME-TAG` mode using `TAGFLASH` mode so that the spectra can be corrected for any OSM drifts, and with different `FP-POS` positions to enhance the signal-to-noise of the data.
- Obtaining more spectra during additional orbits as needed to achieve a desired signal-to-noise.

The typical COS observing sequence depends greatly on the type of observation specified. Typical COS observations use `TIME-TAG` mode and the PSA, with simultaneous wavelength calibrations taken via `TAGFLASH`. In this case, there was no reason to take multiple exposures unless the observer wanted multiple `FP-POS` positions to achieve a SNR > 30, or if the observing time required multiple *HST* visits. Multiple exposures are also used to cover the FUV detector gap, or to produce full wavelength coverage from the NUV wavelength stripes.

1.4 COS Coordinate System

References to multiple coordinate systems appear in the headers of COS data. These are tied to the OTA frame, the User frame, and the POSTARG frame. The following is a brief explanation of how these systems (shown in Figure 1.10) are related.

The three coordinate systems of interest are the:

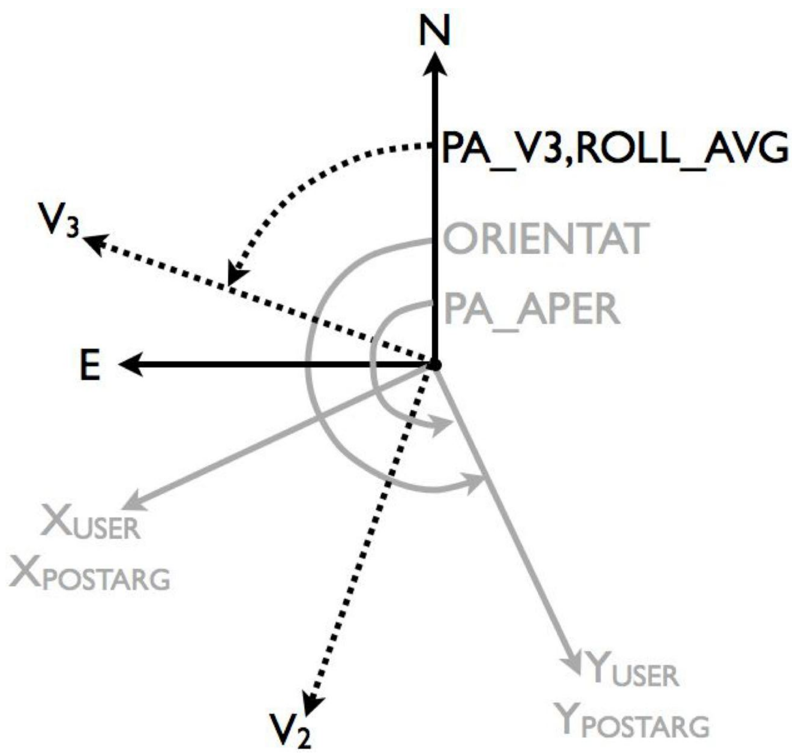
- The OTA or "V" Frame (V_1, V_2, V_3): The common coordinate system for Scientific Instruments and the FGSs. It is a distortion-free frame whose metric is arc seconds.
- User (or **IRAF**) Frame (X_{user}, Y_{user}): The frame associated with an OPUS undrizzled science image. It is aligned with the detector.
- POS-TARG Frame ($X_{POSTARG}$, $Y_{POSTARG}$): This is a distortion-free frame with units of arc seconds. Its origin coincides with the science aperture and its axes are closely aligned with the user frame.

The angles associated with these frames that appear in the headers of COS data files are:

- PA_V3: The position angle of the V_3 axis; the angle from North, towards East, to V_3, measured at the center of the *HST* focal plane (in the spt header).
- ROLL_AVG: The average angle from North towards East to V_3, measured at the position of the COS field in the *HST* focal plane (in the jit header, computed).
- PA_APER: The angle from North through East to $Y_{POSTARG}$ measured at the aperture reference (in the science header).
- ORIENTAT: The angle from North through East to Y_{user} measured at the aperture reference (in science header). For COS, PA_APER and ORIENTAT are equal, i.e., $Y_{POSTARG} = Y_{USER}$. *Note that this is not the same angle as the ORIENT specified in Phase II, which gives the position angle of the U3 axis, where U3 = -V3.*

Refer to ISR TEL2008-02 for a complete discussion of the COS reference frame geometry.

Figure 1.10: COS Coordinate Systems



CHAPTER 2:

COS Data Files

In this chapter...

2.1 Overview

Raw COS telescope data are processed through the STScI **OPUS** pipeline. The **OPUS** pipeline first processes the data through Generic Conversion, where the data bits from individual exposures are unpacked and combined into files containing raw, uncalibrated data. Next, the data are processed through the COS calibration pipeline, **calcos**, which performs image and spectroscopic reduction to produce output files that can be used directly for scientific analysis (see Chapter 3 for a more detailed description of the COS calibration pipeline). Finally, the data are ingested into the HDA through the Data Archive and Distribution System (DADS). This system populates a database containing header keywords which is accessible to users via the Multimission Archive at STScI (MAST). The data (both calibrated and uncalibrated) are then available for distribution by MAST to the user.

When COS data are requested from the Hubble Data Archive (HDA), they go through "On The Fly Reprocessing" (OTFR) which provides the best calibrated products by reprocessing the raw telemetry files "on-the-fly" each time data are requested. OTFR reprocessing uses the latest software versions and reference files available. The re-processed data are then distributed to the requestor.

The calibration reference files (e.g. flat fields, bad pixel tables) are also available from the *HST* Data Archive for users to download. Since reference files are frequently updated, OTFR may use different reference files depending on the date of reprocessing. In the event of an updated reference file or calibration software, users may re-calibrate their data in one of two ways. Once the updated reference files are released, the preferred method is for the user to re-retrieve their data from the HDA and let OTFR recalibrate the data with the default settings. Alternatively, the user can reprocess the data at home through **calcos** using the most recent reference files and software code (see “Run Calcos” in Section 3.6.1). *The second option will not include any changes in the data due to Generic Conversion updates but, will allow a customized calibration through the use of modified reference files or keyword switches*. Also, the user will need to manually edit the header keywords stating which reference files should be used by **calcos** (Table 2.15, and Section 3.6.1).

Once you have retrieved your data, you will need to understand:

- The naming conventions and file suffixes of the individual files (Section 2.2).
- The basic format in which the COS data are stored (Section 2.3).
- The structure and content of the individual files (Section 2.4).
- The size of the COS data files (Section 2.5).
- How to use the header keywords to identify the principal parameters of an observation and how to determine the calibration processing steps that were performed a dataset (Section 2.6).
- The meanings of the error and data quality arrays, which are propagated through the pipeline for each COS science observation (Section 2.7).

2.2 COS File Names

The naming convention for COS files is `rootname_*.fits`, where rootname follows the ippsoot naming convention (see Appendix B), and `*` is a three to nine character file suffix. The suffix identifies the type of data within the file. All FUV data files with the exception of the `x1d` and `x1dsum` files will have an additional suffix of `_a` or `_b` (e.g. `rootname_*_[a,b].fits`) to denote the detector segment. However, if `segment=A` is specified in the Phase II proposal there will be no corresponding `_b` files and vice versa. The FUV `x1d` and `x1dsum` files will always be segment combined and therefore will not have the additional suffix.

Table 2.1 lists the file suffixes for the COS data files and indicates which files are produced by the different types of observations. Depending on the type of observation, and the path it has taken through the calibration pipeline (see calibration flow charts; Figure 3.1-Figure 3.5), there will be an appropriate subset of these files in a given dataset. Note, the format of some of the COS files can be different depending on the observing mode; see Section 2.3 for more details.

COS data utilizes a modified naming convention from other HST instruments. In, particular COS FUV files can have TWO suffixes. The first suffix identifies the filetype and the second suffix if present identifies the FUV detector segment. For the remainder of this document the use of "suffix" will refer to the first suffix which identifies the filetype and will always include filetypes with the additional FUV segment suffix if they exist.

Table 2.1: Data Types and File Naming Conventions

Long Suffix	Data Format	Spectroscopic				Imaging		Related Page No.	Contents
		FUV		NUV		NUV			
		TIME-TAG	ACCUM	TIME_TAG	ACCUM	TIME_TAG	ACCUM		
									Uncalibrated Science Data
rawtag	table			•		•		9	Raw NUV TIME-TAG events list
rawtag_a, rawtag_b	table	•						9	Raw FUV TIME-TAG events list
rawaccum	image				•		•	8	Raw NUV ACCUM image
rawaccum_a, rawaccum_b	image		•					8	Raw FUV ACCUM image
rawacq	table or image	•		•		•	•	20	Raw acquisition file
pha_a, pha_b	image		••					10	Pulse height distribution
									Uncalibrated Support Data
asn	table	•	•	•	•	•	•	16	Association file
jit	table	•	•	•	•	•	•	23	Spacecraft pointing data averaged over 3 s intervals
jif	image	•	•	•	•	•	•	25	2-D histogram of the _jit file
spt	image	•	•	•	•	•	•	19	Support, planning and telemetry information
trl	table	•	•	•	•	•	•	18	Trailer file with a historical record of generic conversion processing
									Intermediate Data Products
trl	table	•	•	•	•	•	•	18	The raw trailer file is updated with a historical record and errors log of calibration pipeline processing[1]

Long Suffix	Data Format	Spectroscopic FUV TIME-TAG	Spectroscopic FUV ACCUM	Spectroscopic NUV TIME_TAG	Spectroscopic NUV ACCUM	Imaging NUV TIME_TAG	Imaging NUV ACCUM	Related Page No.	Contents
corrtag	table			•				11	NUV TIME-TAG events list with calibrated values
corrtag_a, corrtag_b	table	•						11	FUV TIME-TAG events list with calibrated values
flt	image			•	•	•	•	14	NUV flat-fielded science image
flt_a, flt_b	image	•	•					14	FUV flat-fielded science image
counts	image			•	•	•	•	13	NUV not flat-fielded science image
counts_a, counts_b	image	•	•					13	FUV not flat-fielded science image
lampflash	table	•[2]		•[2]				12	1-D extracted TAGFLASH (FLASH=yes) spectra
x1d	table	•	•	•	•			14	1-D extracted spectra for a single exposure
x1dsum<n>[3]	table	•	•	•	•			14	Averaged 1-D extracted spectra for multiple exposures with the same grating, central wavelength, aperture and FP-POS=<n>
									Final Data Products
fltsum	image					•	•	16	Summed flat-fielded image (imaging only). ***Final calibrated association product for all COS imaging datasets***
x1dsum	table	•	•	•	•			14	Final combined 1-D extracted spectra for multiple exposures with the same grating, central wavelength and aperture combining all FP-POS. ***Final calibrated association product for all COS spectroscopic datasets.***

1. Only updated during processing and ingestion by the HDA. When reprocessing data in a user's home environment the trl file will not be updated. Instead reprocessing will generate an ASCII tra file.
2. Only for TIME-TAG with FLASH=yes (TAGFLASH mode)
3. <n> can be 1,2,3,4 and denotes the FP-POS number.

2.3 COS File Structures

All COS data products are Multi-Extension FITS (MEF) format files and begin with a primary data unit which includes only a header with no data extension. The primary header stores keyword information describing global properties of the exposure in the file (e.g., the target name, target coordinates, exposure type, optical element, aperture, detector, calibration switches, and reference files used). The **catfits** task in **STSDAS** can be used to list the complete set of extensions and their data formats for the COS data files. For more information on working with MEF format files please refer to Chapter 2 in Part I.

2.3.1 COS FITS Table Extension Files

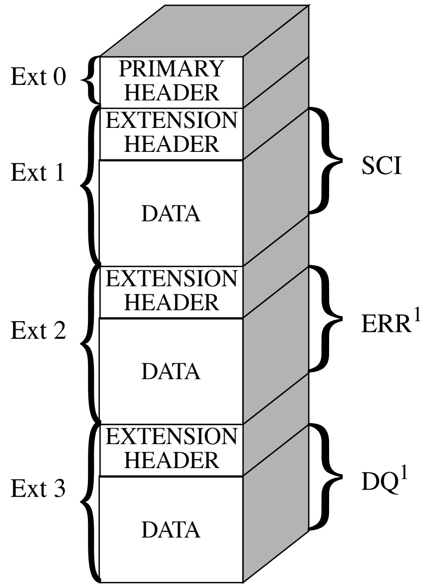
Tabular COS information, such as extracted one-dimensional spectra or the `TIME-TAG` mode event series, are stored as FITS binary tables. The tables can be accessed directly in the **PyRAF/IRAF/STSDAS** environment using tasks in the tables.ttools package as described in Chapter 2 and Chapter 3 of Part I of this document, or with other standard FITS tools.

2.3.2 COS FITS Image Extension Files

COS images and two-dimensional spectroscopic data are stored in FITS image extension files, which can be directly manipulated, without conversion, in the **PyRAF/IRAF/STSDAS** environment. Accessing images in the FITS image extension files in **IRAF** follows a simple convention explained in detail in Chapter 2 of Part I of this document. Figure 2.1 illustrates the structure of a COS FITS image extension file, which contains:

- A primary header that stores keyword information describing the global properties of the exposure in the file (e.g., the target name, target coordinates, exposure type, optical element, aperture, detector, calibration switches, reference files used).
- A set of image extensions, each containing header keywords with information specific to the given exposure (e.g., exposure time, world coordinate system) and a data array.

Figure 2.1: FITS Image Extension File for COS



1. Not all COS image extension files will contain the ERR and DQ extensions.

The following filetypes are stored in FITS image extension files with the particular format shown in Figure 2.1: `rawaccum`, `flt`, `counts`, `pha` and `rawacq`[1]. Each COS readout can generate one FITS image SCI extension or three FITS image extensions (SCI, ERR, and DQ) as explained below:

- The first extension type, SCI, stores the science values.
- The second extension type, ERR, contains the statistical errors, which are propagated through the calibration process. It is unpopulated in raw data files.
- The third extension type, DQ, stores the data quality values, which flag suspect pixels in the corresponding SCI data.

The error arrays and data quality values are described in more detail in Section 2.7. The value of the `XTENSION` keyword in the extension header identifies the type of data the extension contains; the value of this keyword may be determined using the IRAF **tables** tasks **catfits** or **thedit**.

1. Only ACQ/IMAGE files use the format shown in Figure 2.1. For more details on acquisition file formats see “Acquisition Files (`RAWACQ`)” in Section 2.4.4.

2.4 COS Data Products

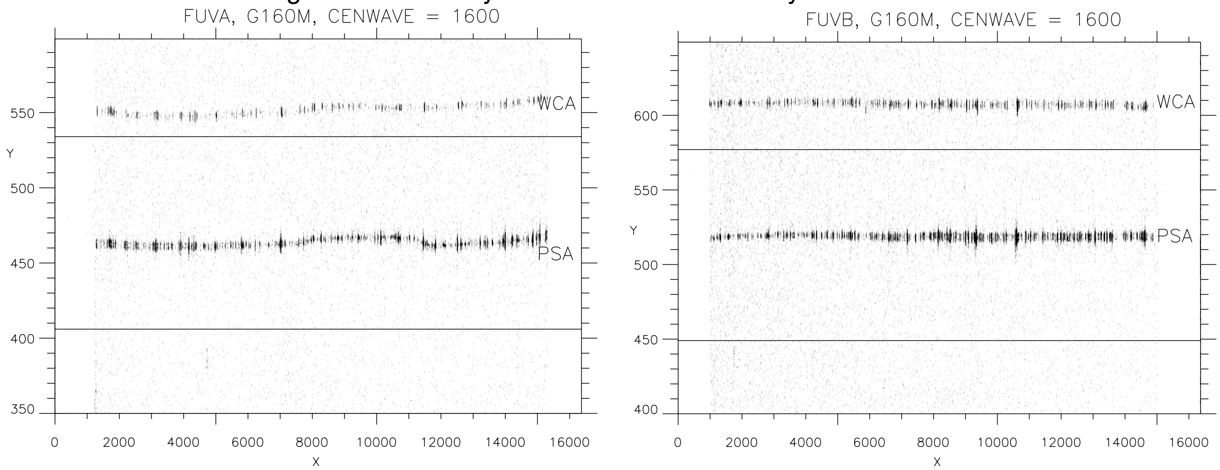
The following sections discuss the COS raw science data files, intermediate calibration products, final calibration products, and auxiliary data files. Uncalibrated science data includes all raw science data generated during Generic Conversion that have not been processed through the calibration pipeline. These raw files are the input files to the **calcos** pipeline, usually as part of an association (see “Association Tables (ASN)”). This results in both individual calibrated exposure files and a final combined product file.

2.4.1 Uncalibrated Science Data Files

*Raw ACCUM Images (*rawaccum*):*

For ACCUM data, the raw files contain a set of images, as shown in Figure 2.1, and have filenames with the suffix rawaccum for NUV data, or rawaccum_a and rawaccum_b for the two segments of the FUV detector. The SCI extension contains an image of the total accumulated counts during an exposure. For NUV data the ERR and DQ extensions have only a header with no data. For FUV data the ERR extension has only a header with no data, and the DQ extension is populated with data quality information only for pixels that are outside the subarray boundary. These extensions will be populated with data in the flt files after calibration pipeline processing. Even though FUV rawaccum_a[b] data are 16384 x 1024 images, only a portion of them contain actual data. These portions are called sub-arrays. Typically, three subarrays are used for each segment of an FUV ACCUM image. Two of these are centered on the STIM positions and the third is a stripe 128 pixels wide which is centered on the spectrum of the object. Figure 2.2 shows these spectral region sub-arrays superimposed on two FUV rawtag images.

Figure 2.2: Overlay of FUV ACCUM subarrays on FUV TIME-TAG data



The above figures shows FUV TAGFLASH data for both segments with the corresponding ACCUM subarrays noted by the dark lines. The data plotted here are the raw event locations prior to calibration processing. The distortion in the data, particularly for segment A, if very noticeable and discussed further in Section 3.4.7.

Raw `TIME-TAG` *Events Lists (*`rawtag`*)*

Raw events tables contain the locations and arrival times of individual photon events collected in `TIME-TAG` mode. These files have the suffix `rawtag` for NUV or `rawtag_a[b]` for the two FUV segments. Figure 2.3 shows the format of a `rawtag` table. The first extension contains the events list, in which each row of the table corresponds to a single event in the data stream and the columns of the table contain scalar quantities that describe the event. The second extension contains the good time intervals (GTI) table, where an uninterrupted period of time is considered as one good time interval. Interruptions in the data taking due to memory overflow could result in more than one GTI. Table 2.2 shows the columns of a `rawtag` table.

Figure 2.3: FITS File Format for Raw and corrected `TIME-TAG` Tables

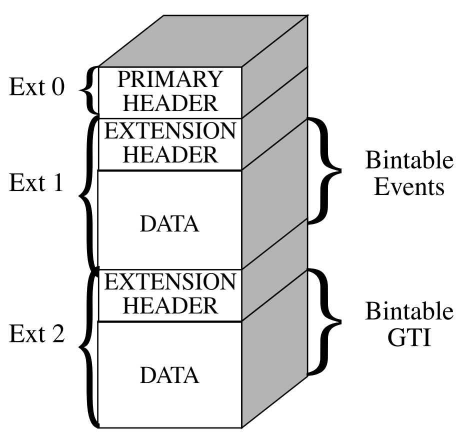


Table 2.2: Columns of a Raw TIME-TAG Data Table

Extension 1			
Column Name	**Units**	**Data Type**	**Description**
TIME	sec	float	Elapsed time in seconds since the exposure start time
RAWX	pixel	integer	Pixel coordinate along the dispersion axis
RAWY	pixel	integer	Pixel coordinate along the cross-dispersion axis
PHA[1]		byte	Pulse height amplitude (0-31)
Extension 2			
Column Name	**Units**	**Data Type**	**Description**
START	sec	float	Start good time interval since exposure start
STOP	sec	float	End good time interval

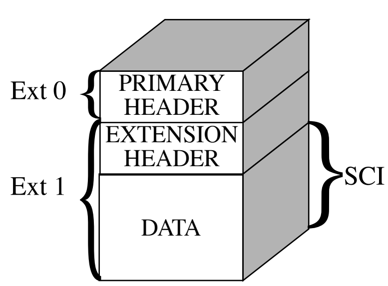
1. The PHA column is present in the NUV data *only* for symmetry with the FUV data columns. For NUV data the values in this column are set to 0, since no pulse height amplitudes are available.

For more information on working with TIME-TAG data see Section 5.4

Pulse Height Amplitude Files (pha):

For FUV ACCUM data only, a 7 bit pulse height amplitude histogram is accumulated in the detector electronics on-board. This information is placed in a file with the suffix pha. The pulse-height histogram files contain a primary header with no data and a single FITS image SCI extension containing a histogram of the pulse-height distribution during the exposure. The pulse height amplitude files do not contain an ERR or DQ extension, as shown in Figure 2.4. The pulse height distribution is an image array of length 128, corresponding to the number of photons with values from 0 to 127, corresponding to the pulse heights of 0-31 available in TIME-TAG data.

Figure 2.4: FITS Array Extension File for COS



2.4.2 Intermediate Science Data Files

Corrected Events Lists (`corrtag`):

The COS pipeline produces corrected `TIME-TAG` events lists and stores them in binary tables with suffix `corrtag`. These files have the same file format as the `rawtag` files, with a corrected events list and a good time interval extensions as shown in Figure 2.3. The corrected events table includes `X` and `Y` event locations that have been corrected for distortion, doppler shift, and offsets due to OSM motions in both the dispersion and cross-dispersion directions.

Table 2.3: Columns of a COS `corrtag` Table

Column Name	Units	Data Type	Description
TIME	sec	float	Elapsed time in seconds since the exposure start time
RAWX	pixel	integer	Pixel coordinate along dispersion axis (same as in rawtag file)
RAWY	pixel	integer	Pixel coordinate along cross-dispersion axis (same as in rawtag file)
XCORR[1]	pixel	float	RAWX corrected for distortion[1]
XDOPP	pixel	float	XCORR corrected for Doppler shift and for FUV only distortion
YCORR[1]	pixel	float	RAWY corrected for distortion[1]
XFULL	pixel	float	XDOPP corrected for offset in the dispersion direction, based on the wavecal spectrum
YFULL	pixel	float	YCORR corrected for offset in the cross-dispersion direction, based on the wavecal spectrum
EPSILON		float	Event weight based on flat field and deadtime
DQ		float	Data quality flag
PHA[2]		float	Pulse height amplitude
Extension 2			
Column Name	**Units**	**Data Type**	**Description**
START	sec	float	Start good time interval since exposure start
STOP	sec	float	End good time interval

1. Columns XCORR, YCORR, are present in the NUV data, only, for symmetry with FUV data columns. There is currently no distortion correction applied to the NUV data, and therefore the XCORR and YCORR columns are the same as the RAWX and RAWY columns for NUV data.
2. The PHA column is present in the NUV data *only* for symmetry with the FUV data columns. For NUV data this column is set to a default value of 0, since no pulse height amplitudes are available for NUV.

Lampflash Files *(`lampflash`):*

For `TAGFLASH` data, **calcos** produces an events list with suffix `lampflash`, that contains the extracted wavecal lamp flashes with one row for each unique segment or stripe and flash number (see Figure 2.6). The `lampflash` files have the format shown in Figure 2.5. The contents of the columns in a `lampflash` events list are listed in Table 2.4. Columns `TIME`, `LAMP_ON`, and `LAMP_OFF` are in seconds since the exposure start time; they are therefore in the same units and have the same zero point as the values in the `TIME` column of the `rawtag` or `corrtag` tables.

Figure 2.5: FITS File Format for Lampflash Table

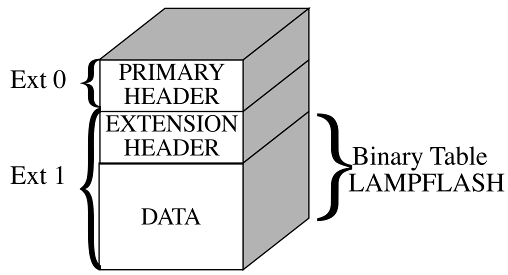


Table 2.4: Columns of a COS Lampflash Table

Column Name	Units	Data Type	Description
SEGMENT		String	FUV segment or NUV stripe name
TIME	sec	float	Median time of current flash
EXPTIME	sec	float	Duration of lamp flash in seconds
LAMP_ON	sec	float	Time since exposure start time of lamp turn on
LAMP_OFF	sec	float	Time since exposure start time of lamp turn off
NELEM		integer	Length of the WAVELENGTH, GROSS, NET, BACKGROUND, DQ, DQ_WGT, and ERROR arrays
WAVELENGTH	Å	double[nelem]	Wavelengths corresponding to fluxes
GROSS	counts s^{-1}	double[nelem]	Gross count rate
SHIFT_DISP	pixel	double[nelem]	Shift in the dispersion direction
SHIFT_XDISP	pixel	double[nelem]	Shift in the cross-dispersion direction
SPEC_FOUND		double[nelem]	Status of finding the spectrum (yes or no)

Figure 2.6: Example of COS Lampflash File

```
--> tread l9v213heq_lampflash.fits[1]                      \
columns="SEGMENT,TIME,EXPTIME,LAMP_ON,LAMP_OFF,SPEC_FOUND"

Column    1        2         3         4          5          6

Label SEGMENT  TIME     EXPTIME   LAMP_ON   LAMP_OFF  SPEC_FOUND

     1 NUVA    5.440     11.000   0.000     11.000    yes

     2 NUVB    5.440     11.000   0.000     11.000    yes

     3 NUVC    5.440     11.000   0.000     11.000    no

     4 NUVA    605.888   10.000   601.000   611.000   no

     5 NUVB    605.888   10.000   601.000   611.000   yes

     6 NUVC    605.888   10.000   601.000   611.000   no

     7 NUVA    2405.920  10.000   2401.000  2411.000  no

     8 NUVB    2405.920  10.000   2401.000  2411.000  yes

     9 NUVC    2405.920  10.000   2401.000  2411.000  yes

    10 NUVA    4805.120  8.000    4801.000  4809.000  yes

    11 NUVB    4805.120  8.000    4801.000  4809.000  yes

    12 NUVC    4805.120  8.000    4801.000  4809.000  yes
```

The above figure shows the 12 rows and only a subset of the columns of a NUV lampflash file containing four flashes. The tread task returns only the first element of each column for each row in the table. Note that for NUV data there will be three rows for each flash corresponding to the different stripes. Similarly, for FUV data there will be two rows for each flash corresponding to the two segments. For more information on STSDAS Table tasks see Section 5.1.2.

Counts Files (`counts`):

The counts images are an intermediate calibrated output product for both imaging and spectroscopic data with suffix `counts`. These files contain three extensions (SCI, ERR, and DQ) as shown in Figure 2.1. The data are in units of counts per pixel. For FUV data the images are 16384 columns by 1024 rows. The NUV images are 1274 x 1024 for spectroscopic data and 1024 x 1024 for data obtained in imaging mode. These are larger than the actual NUV detector in the x (wavelength) direction in order to accommodate Doppler and format shifts (due to OSM motions). The FUV images are not extended since the active area is less than the size of the detector, so these effects can be incorporated in to the images without the need to extend them.

Flat-Fielded Image Files (`flt`):

For *spectroscopic* data a flat-fielded image is an intermediate calibrated data file. The files has suffix `flt`, and contains three extensions (SCI, ERR, and DQ) as shown in Figure 2.1. The data are in units of the count rate. For FUV data the images are 16384 x 1024, and, like the `counts` images, the NUV images are 1274 x 1024 for spectroscopic data and 1024 x 1024 for data obtained in imaging mode. The `flt` images are corrected for flat field and deadtime effects, and this is what distinguishes them from the `counts` images.

2.4.3 Final Science Data Files (and Product Files)

The initial input files to **calcos** are the association tables with suffix `asn`. These files provide the calibration pipeline with information about how the data files are associated. In general, only exposures taken in sequence with the same spectral element, central wavelength (if applicable), and aperture at any `FP-POS` will be associated. For more information on COS association files see the "Association Tables (`ASN`)" portion of Section 2.4.4.

Processing of each individual exposure in the association produces a final calibrated result named with exposure rootname and suffix `x1d` (spectroscopy) or `flt` (imaging).

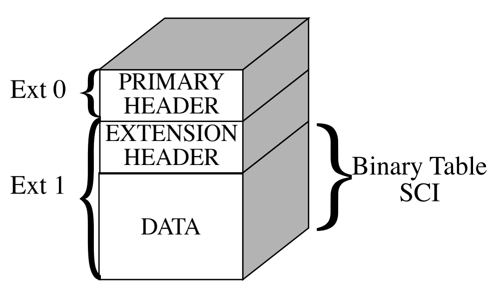
Next, for each `FP-POS` position <n> (where <n>=1,2,3, or 4), if there are *multiple spectroscopic* exposures in the association that use the same `FP-POS` position, **calcos** will combine them into a file named with the association rootname and suffix x1dsum<n>, where <n> is the integer `FP-POS` value.

Lastly, a final association product file is produced with association rootname and suffix `x1dsum` (spectroscopy) or `fltsum` (imaging) by combining *all science exposures* in the association. [Note: in the special case of associations with only one science exposure, the resultant exposure rootname `x1d` file and the association rootname product `x1dsum` file contain identical information.

One-Dimensional Extracted Spectra (`x1d, x1dsum`):

The COS pipeline produces extracted one-dimensional spectra and stores them in binary tables with suffix `x1d, x1dsum<n>` or `x1dsum`. Figure 2.7 shows the format of the 1-D extracted spectra table.

Figure 2.7: FITS File Format for 1-D Extracted Spectrum Table.



These COS extracted spectra tables are 3-Dimensional, with one row for each unique segment or stripe. For FUV data there are two rows labeled 1 and 2 which correspond to segments A and B respectively[2]. For NUV data there are three rows labeled 1,2, and 3 which corresponding to stripes A,B, and C respectively. Each table column can contain either a scalar value or an array of values, such as WAVELENGTH or FLUX. For example, NELEM will contain a scalar number, while the WAVELENGTH column will contain an array of wavelengths. Table 2.5 shows the contents of the different columns in an extracted spectrum table. A discussion of the data in COS extracted spectra is provided in Section 3.4.13.

Table 2.5: Columns of a COS Extracted Spectrum Table

Column Name	Units	Data Type	Description
SEGMENT		string	FUV segment or NUV stripe name
EXPTIME	sec	float	Corrected exposure time
NELEM		integer	Length of the array fields, such as the WAVELENGTH and GROSS arrays
WAVELENGTH	Å	double[nelem]	Wavelengths corresponding to fluxes
FLUX	erg s^{-1} cm^{-2} $Å^{-1}$	double[nelem]	Flux calibrated NET spectrum
ERROR	erg s^{-1} cm^{-2} $Å^{-1}$	double[nelem]	Internal error estimate
GROSS	counts s^{-1}	double[nelem]	Gross extracted spectrum count rate
NET	counts s^{-1}	double[nelem]	Difference of GROSS and BACKGROUND arrays
BACKGROUND	counts s^{-1}	double[nelem]	Background count rate
DQ_WGT		double[nelem]	Data quality weight
DQ		double[nelem]	Logical OR of data quality flags in extraction region

Flat-Fielded Image Files (`flt, fltsum`):

For NUV imaging observations, the `flt` and `fltsum` images are the final data products, with the latter being a simple sum of the individuals when several exposures are processed together. They are fully linearized and flat field corrected images. Unlike the `flt` files produced for the spectroscopic data (which are intermediate data products with a format of 1274 x 1024, see Section 2.4.2), the formats of the `flt` and `fltsum` files for imaging data is 1024 x 1024, since Doppler and OSM motions are not applied.

2. For FUV exposures taken with only segment B, the row labeled "1" will correspond to segment B, and there will be no row "2", and vice versa.

2.4.4 Auxiliary Data Files

Association Tables (`ASN`)

An association file is created for all COS observation sets, and has the suffix asn (e.g., `l9v221010_asn.fits`). This file holds a single binary table extension, which can be displayed with the IRAF tasks **tprint** or **tread**.

Calcos calibrates raw data from multiple science exposures and any contemporaneously obtained line lamp calibration exposures through the pipeline as an associated unit. Each individual science exposure in an *association* is fully calibrated in the process. See Appendix B for a general explanation of *HST* data associations. The information within an association table shows how a set of exposures are related, and informs the COS calibration pipeline how to process the data.

An example association table is shown in Figure 2.8. Note that all related COS exposures will be listed in an association table, with the exception of acquisitions, darks, and flats. Since the calibration pipeline processes the data through the association table, it is possible to have an association which contains only one exposure. The association file lists the rootnames of the associated exposures as well their membership role in the association. The exposures listed in an association table directly correspond to individual raw FITS files. For example, the association table can describe how wavecal exposures are linked to science exposures. Table 2.6 summarizes the different exposure membership types (`MEMTYPES`) used for COS association tables.

Table 2.6: Member types in COS associations.

MEMTYPE	Description
EXP-AWAVE	Input automatic wavelength calibration exposure
EXP-FP	Input science exposure
EXP-GWAVE	Input GO wavelength calibration exposure
PROD-FP	Output science product

Figure 2.8 illustrates the contents of the association table for a sequence of spectroscopic exposures for four FP-POS positions.

Figure 2.8: Sample Association Table l9v221010_asn

To display the association table for l9v221010_asn.fits:

```
cl>   tprint   l9v221010_asn.fits

# row MEMNAME    MEMTYPE        MEMPRSNT


   1  L9V221EUQ   EXP-FP         yes
   2  L9V221EWQ   EXP-AWAVE      yes
   3  L9V221EYQ   EXP-FP         yes
   4  L9V221F0Q   EXP-AWAVE      yes
   5  L9V221F2Q   EXP-FP         yes
   6  L9V221F4Q   EXP-AWAVE      yes
   7  L9V221F6Q   EXP-FP         yes
   8  L9V221F8Q   EXP-AWAVE      yes
   9  L9V221010   PROD-FP        yes
```

In the above figure, MEMTYPE describes the exposure membership type or role in the association. The column MEMPRSNT lists whether the member is present or not. A user could choose to change the association file to not include a member during processing by changing the MEMPRSNT to 'no'.

The association table above lists the names of the eight associated exposures (four external and four calibration) that are calibrated and combined to create the various association products which will have a rootname of l9v221010. This particular association is created from a single TIME-TAG spectroscopic APT specification with FP-POS=AUTO and FLASH=NO specified in the Phase II file, which leads to both a science exposure and automatic wavecal exposure taken at each FP-POS location. For example, the first entry in the table, l9v221euq, is the rootname of a single external science exposure taken with FP-POS=1. This exposure corresponds to the following rawtag files: l9v221euq_rawtag_a.fits, l9v221euq_rawtag_b.fits. The memtype of this exposure is EXP-FP which shows that it is an external exposure. The second entry in the table has a memtype of EXP-AWAVE. This denotes that the corresponding rawtag exposures, l9v221ewq_rawtag_a.fits and l9v221ewq_rawtag_b.fits, are wavecal exposures that will be used by the pipeline for wavelength calibration. Similar files correspond to the remaining three pairs of entries in the association file for data taken with the remaining three FP-POS positions. The pipeline will calibrate the members of an association as a unit, producing the calibrated data products for each individual exposure as well as the final combined association data product. For this particular association, the pipeline will produce a final combined association product, l9v221010_x1dsum.fits, which contains the final FP-POS combined, calibrated spectrum.

Trailer Files (TRL)

When COS data are processed through OTFR in the HDA, the output messages from generic conversion and the different calibration steps are stored in a FITS ASCII table known as the trailer file, with suffix trl. Each time the archive processes data before retrieval, the old trailer file is erased and a new one created using the results of the most recent processing performed. The archive will produce a trailer file for each individual exposure and association product. Association product trailer files contain the appended information from all the exposures in the association, in order of processing. The order of processing is the same as the order of exposures in the association table, with the exception of AUTO or GO wavecals which are always processed first.

In the trailer files from the HDA, the output messages from generic conversion appear first in the file. This section contains information relevant to the selection of the best reference files and the population of some of the header keywords. The second part of this file contains information from **calcos** processing. Each task in the **calcos** pipeline creates messages during processing which describe the progress of the calibration, and appear in the order in which each step was performed. These messages are quite relevant to understanding how the data were calibrated, and in some of the cases, to determining the accuracy of the products.

It is highly recommended to always examine the trailer files.

In this last section of the _trl file, the **calcos** steps are indicated by their module name. The **calcos** messages provide information on the input and output files for each step, the corrections performed, information regarding the reference files used, and in the case of FUV data, messages about the location of the stims, or shift correction applied to the data. **Calcos** also gives warnings when the appropriate correction to the data could not be applied. For more detailed information on the calibration steps and structure of **calcos**, please refer to Chapter 3.

Calcos Trailer Files (TRA)

When calcos is run in a user's home environment, **calcos** redirects the output of its steps to the STDOUT and an ASCII file with name rootname.tra. Note, the level of detail included in the output messages can be modified when running **calcos** (see "Run Calcos"). So, when run on a personal machine, **calcos** will *not* overwrite the trl file but rather will direct the output to STDOUT and an ASCII tra file. The tra file is formatted like the trl file but with two exceptions: the tra file will not contain the output messages from generic conversion, and the `tra` file is not converted to FITS format. Therefore, one must look at both the trailer (generic conversion messages) and `tra` (calibration messages) file when running **calcos** from a home environment. Each time calcos is run on a file, the STDOUT messages will be appended to the `tra` file if it already exists. Also, when running **calcos** on a personal machine there will be no `tra` created for the association products. Instead, the **calcos** messages for association products will be sent only to STDOUT.

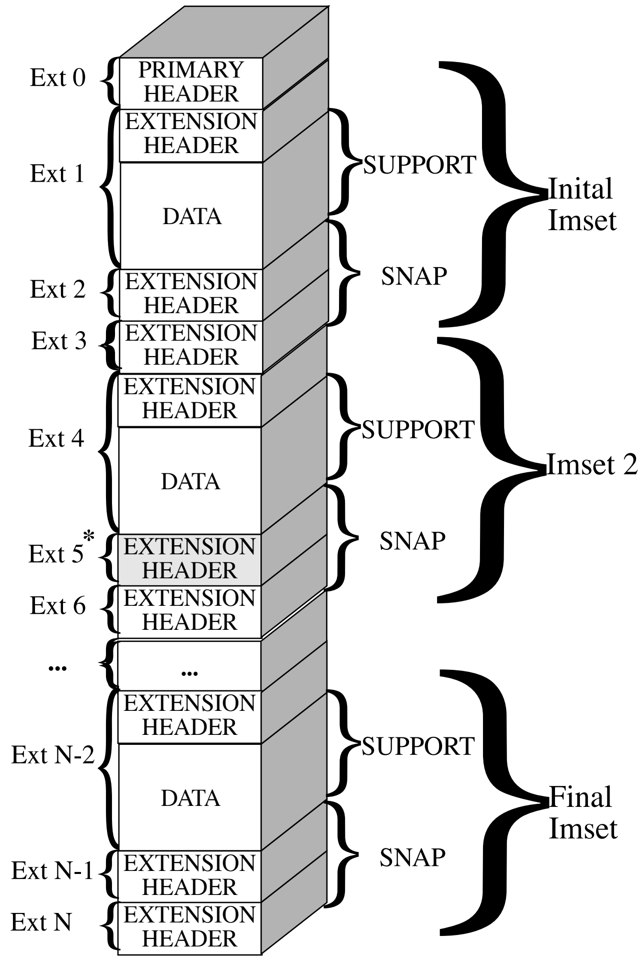
Support Files (SPT)

The support files contain information about the observation and engineering data from the instrument and spacecraft that was recorded at the time of the observation. A COS support file contains a primary header and at least three FITS image extensions. The first extension contains a header with the proposal information and an (16-bit) image array containing the data which populate the `spt` header keyword values. The image array element values are used by conversion software to populate the header keywords. Following the support extension, the COS `spt` files contain two engineering snapshot extensions. These extensions contain a readout of several instrument and telescope parameters from telemetry data at different times during the course of an exposure. The very first snapshot extension will always contain telemetry information from the beginning of an exposure. Depending on the length of the exposure, the support file may also contain one or several "imsets" which include a support extension and two snap extensions. These intermediate imsets will have only their second snapshot extension populated with telemetry data taken during the course of an exposure, while the first snapshot will be populated with default values. The very last imset of an `spt` file will have all three extensions (1 support and 2 snaps) populated with telemetry values at the completion of the exposure. Figure 2.9 depicts the structure of an N extension COS support file.

With several snapshots of COS telemetry values, one may track the instrument status periodically throughout an exposure. For a schematic listing of the spt headers with detailed information about the `spt` header keywords, See:

http://www.dpt.stsci.edu/cgi-bin/kdct-header?i=COS&s=20.1&db=Operational

Figure 2.9: COS Support File



* Extension 5, is not populated, and therefore all header keywords in this extension will be set to a default. Every other snapshot extension from extension 5 through N-4, will also not be populated.

COS support file with N extensions. The initial imset contains telemetry values at the start of the exposure. Extensions 3 through (N-3) contain imsets with telemetry values at intermediate times during the exposure. Notes the first snap extensions in these intermediate imsets are NOT populated. The final imset includes extensions N-2 through N and contains telemetry values at the end of the exposure. Both snap extensions are populated for the final imset.

Acquisition Files (RAWACQ)

All COS acquisition exposures will produce a single raw data file with suffix `rawacq`. Almost all COS spectroscopic science exposures are preceded by an acquisition sequence or exposure to center the target in the aperture. Keywords in the header of COS science data identify the exposure names of relevant acquisition exposures in each visit. In addition, there are several other useful keywords in the COS acquisition exposures that describe the acquisition parameters used, as well as the calculated centroid positions and slew offsets. Table 2.7 lists all the relevant acquisition keywords.

Table 2.7: Acquisition Header Keywords.

Keyword Name	Description
ACQSNAME[1]	Rootname of first acquisition search exposure
ACQINAME[1]	Rootname of first acquisition image exposure
PEAKXNAM[1]	Rootname of first cross-dispersion peakup exposure
PEAKDNAM[1]	Rootname of first dispersion peakup exposure
ACQ_NUM[1]	Total number of exposures in acquisition sequence
ACQCENTX	Calculated acquisition centroid X
ACQCENTY	Calculated acquisition centroid Y
ACQSLEWX	Slew offset in X
ACQSLEWY	Slew offset in Y
ACQMEASY	Measured computed Y position from wavecal
ACQPREFY	Desired computed Y position
TARGA2	Flight software located subarray axis2 coordinate of target
TARGA1	Flight software located subarray axis1 coordinate of target
TAFLAG	Target acquisition global status flag
CHECKBOX	Size of checkbox for finding algorithms
SCANSIZE	Number of dwell points per side square pattern
CENTER	Centering method used in Peakup or other
ACQTHRSH	Threshold value for peakup
LMPSUBX1	X1 of lamp subarray for acquisition location
LMPSUBX2	X2 of lamp subarray for acquisition location
LMPSUBY1	Y1 of lamp subarray for acquisition location
LMPSUBY2	Y2 of lamp subarray for acquisition location
TAXDCOR	Average cross-dispersion coordinate
TADCOR	Average Y coordinate
PEAKSTEP	Peakup scan stepsize in milli-arcseconds
PEAKNPOS	Number of dwell positions in search or peakup
PEAKCENT	Centering method used in Peakup acquisition

1. These keywords are also found in the COS science headers in addition to being in the acquisition headers.

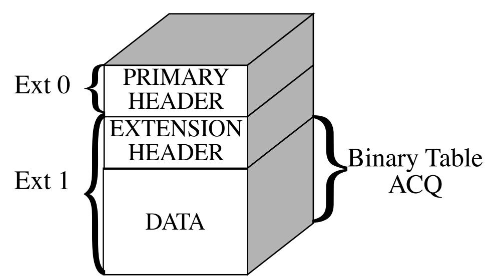
PEAKD and SEARCH Acquisitions:

Acquisition peakups in the dispersion direction (ACQ/PEAKD) and acquisition spiral searches (ACQ/SEARCH) both use the flux from exposures taken at different dwell points to center the target. For more information on these types of COS acquisitions see Sections 7.6.4 and 7.6.2 respectively of the COS Instrument Handbook. Data for these acquisitions contain one binary table extension which describes the acquisition search pattern dwell point locations and counts as shown in Table 2.8 and Figure 2.10

Table 2.8: Columns of an ACQ/SEARCH or ACQ/PEAKD table

Column Name	Units	Description
DWELL_POINT		Dwell point number in search pattern
DISP_OFFSET	arcsec	Offset in dispersion direction from the initial target pointing
XDISP_OFFSET	arcsec	Offset in the cross-dispersion direction from the initial target pointing
COUNTS	counts	Raw counts value at dwell point

Figure 2.10: FITS File Format for ACQ/SEARCH and ACQ/PEAKD data.



PEAKXD Acquisition:

Acquisition peakups in the cross-dispersion direction (ACQ/PEAKXD) use a TIME-TAG spectrum to center the target in the cross-dispersion direction. For more information on the ACQ/PEAKXD algorithm see Section 7.6.3 of the COS Instrument Handbook. An ACQ/PEAKXD exposure includes only a primary header and extension header. There are no data downlinked for this type of acquisition.

IMAGE Acquisition:

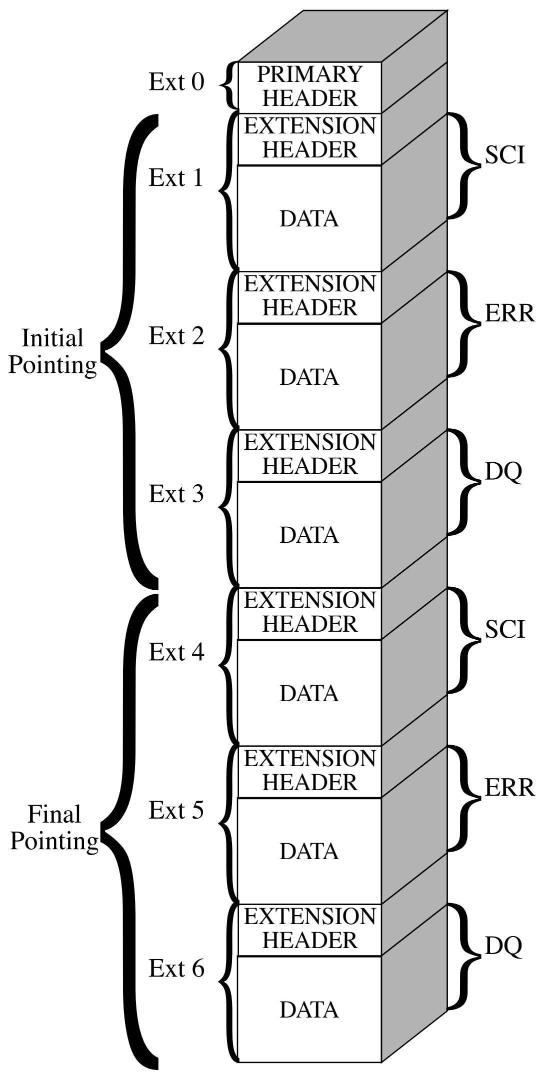
Acquisition images (ACQ/IMAGE) use a NUV image to center the target in the aperture. For more information on the ACQ/IMAGE algorithm see Section 7.5 of the COS Instrument Handbook. An ACQ/IMAGE exposure produces a raw data file containing two science image extensions corresponding to the initial and final pointing:

- [SCI,1] is an image of the initial target pointing.

- [SCI,2] is a confirmation image after the acquisition procedure has been performed.

See Figure 2.11 for the FITS file format for `ACQ/IMAGE` data.

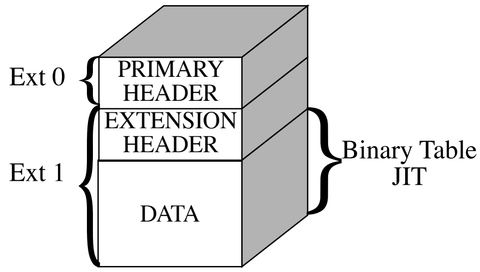
Figure 2.11: FITS File format for ACQ/IMAGE data



Jitter Files (`jit`)

The COS jitter files include 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 during the duration of the observation. The support files contain information about the observation and engineering data from the instrument and spacecraft that was recorded at the time of the observation. COS Jitter files utilize the file format shown in Figure 2.12 for all science observations, excluding acquisitions.

Figure 2.12: FITS File Format for JITTER data.



The jitter tables contain PCS data for each three second interval during the observation, as listed in Table 2.9. For more information on jitter file refer to Appendix C of Part III of this document.

Table 2.9: Columns of a jitter Table.

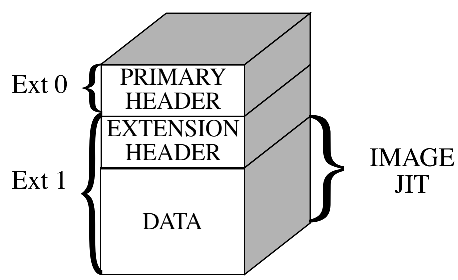
Column Name	Data Type	Units	Description
SECONDS	float	seconds	'Seconds' three second intervals from start
V2_DOM	float	arcsec	Dominant FGS V2 Coordinate
V3_DOM	float	arcsec	Dominant FGS V3 Coordinate
V2_ROLL	float	arcsec	Roll FGS V2 Coordinate
V3_ROLL	float	arcsec	Roll FGS V3 Coordinate
SI_V2_AVG	float	arcsec	Mean jitter in V2 over 3 seconds
SI_V2_RMS	float	arcsec	Peak jitter in V2 over 3 seconds
SI_V2_P2P	float	arcsec	RMS jitter in V2 over 3 seconds
SI_V3_AVG	float	arcsec	Mean jitter in V3 over 3 seconds
SI_V3_RMS	float	arcsec	Peak jitter in V3 over 3 seconds
SI_V3_P2P	float	arcsec	RMS jitter in V3 over 3 seconds
RA	double	degrees	Right Ascension of aperture reference
DEC	double	degrees	Declination of aperture reference
ROLL	float	degree	Position angle between North and +V3
LIMBANG	float	degree	Position angle between V1 axis and Earth limb
TERMANG	float	degree	Angle between V1 axis and terminator
LOS_ZENITH	float	degree	Angle between HST Zenith and target
LATITUDE	float	degree	HST subpoint latitude
LONGITDUE	float	degree	HST subpoint longitude
MAG_V1	float	Gauss	Magnetic field along V1

Column Name	Data Type	Units	Description
MAG_V2	float	Gauss	Magnetic field along V2
MAG_V3	float	Gauss	Magnetic field along V3
EARTHMOD	float	V mag * arcsec^{-2}	Model of Earth background light
SI_SPECIFIC	float	counts	Science instrument specific information
DAYNIGHT	string		Observation taken during the day (0) or night (1)
RECENTER	string		Recentering status flag, event in progress =1
TAKEDATA	string		Vehicle guiding status, nominal GS tracking =1
SLEWFLAG	string		Vehicle slewing status, slewing =1

2-D Spacecraft Pointing Histogram (`jif`)

The COS jif files are a 2-D histogram of the corresponding `jif` file (See "Jitter Files (`jit`)") and have the file format shown in Figure 2.13 for all science observations excluding acquisitions.

Figure 2.13: FITS File Format for jif data.



2.5 Data Storage Requirements

Users are reminded to consider the large size of `counts` and `flt` files when allocating disk space for storing and reprocessing COS data. Additionally, `corrtag` files with a large number of events can be quite large. These images serve as intermediate or final calibration products from the pipeline and have the file sizes given in Megabytes in Table 2.10. *Note, that these sizes are per exposure, and an associated observation set may have several exposures.*

Table 2.10: COS Pipeline Data Volumes per Exposure

File Type	FUVA	FUVB	Total FUV	Total NUV	Calibrated File
rawtag	9 bytes per photon	9 bytes per photon	9 bytes per photon (18MB per BUFFER-TIME)	8 bytes per photon (16 MB per BUFFER-TIME	
corrtag	35 bytes per photon	35 bytes per photon	35 bytes per photon (70 MB per BUFFER-TIME)	26 bytes per photon (52 MB per BUFFER-TIME)	•
rawaccum	64MB	64MB	128MB	2MB	•
flt	160MB	160MB	320MB	10MB	•
x1d	0.5MB[1]	0.5MB[1]	1MB[2]	<1MB	•
fltsum	N/A	N/A	N/A	10MB	•
x1dsum	0.5MB	0.5MB	1MB[2]	<1MB	•
counts	160MB	160MB	320MB	10MB	
lampflash	N/A	N/A	<1MB	<1MB	•

1. Values pertain to x1d_a or x1d_b files only. These files are temporary output products from **calcos** processing
2. Values are in addition to amounts given for each segment.

Similarly, users are reminded of the large cumulative size of calibrated COS spectroscopic datasets. Table 2.11 provides volume estimates for calibrated COS datasets.

Table 2.11: COS Pipeline Data Volumes per Calibrated Exposure

Detector	FUV		NUV	
Observation Mode	TIME-TAG	ACCUM	TIME_TAG	ACCUM
Pipeline-processed volume per exposure	650 MB + 44 bytes per photon	775 MB + 36 bytes per photon	25-35 MB + 34 bytes per photon	25-35 bytes
Standard calibrated files[1]	325 MB + 35 bytes per photon[2]	325 MB	15-25 MB + 36 bytes per photon[3]	15-25 MB

1. Minimum volume delivery option over the internet
2. Approximately 70 MB per BUFFER-TIME
3. Approximately 52 MB per BUFFER-TIME

2.6 Headers, Keywords, and Relationship to Phase II

As with previous *HST* instruments, the FITS header keywords in COS data files store important information characterizing the observations and telemetry received during the observations, and describe the post-observation processing of your dataset. Each keyword follows FITS conventions and is no longer than eight characters. Values of keywords can be integer, real (floating-point), boolean, and character strings. Several keywords are *HST* and COS specific. Knowledge of the keywords and where to find them is an important first step in understanding your data. By examining your file headers, using either **catfits, imhead, hselect, thselect** or **hedit,** in **STSDAS** you will find detailed information about your data including:

- Target name, coordinates, proposal ID, and other proposal level information.
- Observation and exposure time information such as observation start and duration.
- Instrument configuration information such as detector, grating, central wavelength setting, and aperture.
- Readout definition parameters such as subarray parameters.
- Exposure-specific information such as more detailed timing, world coordinate system information, and Fine Guidance Sensor identification.
- Calibration information such as the calibration switches and reference files used by the pipeline and parameters derived from the calibration, such as image statistics and wavelength shifts.

The keywords relevant for one COS data type will not necessarily be relevant to another. Accordingly, you will find that the header on a particular file type contains a unique combination of keywords appropriate for that type of observation. Long definitions for the keywords can also be accessed from the following Web page, which provides detailed explanations of the contents and algorithm for populating the keywords. This site also provides sample headers for different COS file types:

http://www.dpt.stsci.edu/keyword.

Keywords that deal with a particular topic, such as the instrument configuration, are grouped together logically throughout the headers. Table 2.12 lists a useful subset of these groups of keywords, indicates the name of the grouping, and where applicable, shows their relationship to the corresponding information from the Phase II proposal.

Table 2.13 summarizes the possible calibration switch keywords, and indicates whether they are present for a particular observation; it also indicates the reference file keyword corresponding to the particular calibration step. A calibration switch keyword is initially populated with values of OMIT, PERFORM or N/A in the raw uncalibrated science data. After each calibration step is executed in the COS calibration pipeline, **calcos**, will set the keyword switch to COMPLETE.

Table 2.12: Selected Header Keywords and Relationship to Phase II Parameters

Header Keyword	Phase II Equivalent	Description
General File Information (Primary Header)		
FILENAME		Name of the file
FILETYPE		Type of data found in the file (SCI, ACQ, SPT, ASN_TABLE)
NEXTEND		Number of extensions in the file.
DATE		Date file was created
Program Information (Primary Header)		
PROPOSID		4 or 5 digit program number.
PR_INV_L	PI Last Name	Last name of principal investigator
PR_INV_F	PI First Name	First name of principal investigator
PR_INV_M	PI Middle Initial	Middle name initial of principal investigator
LINENUM	Visit_Number, Exposure_Number	Indicates the visit and exposure number from the Phase II proposal: Visit_Number, Exposure_Number.
Target Information (Primary Header)		
TARGNAME	TargetName	Name of target.
RA_TARG	RA	Right ascension of the target (deg) (J2000).
DEC_TARG	DEC	Declination of the target (deg) (J2000).
POSTARG1	POSTARG	Postarg in axis 1 direction.
POSTARG2	POSTARG	Postarg in axis 2 direction.
Science Instrument Configuration (Primary Header)		
OBSTYPE		Observation type (IMAGING or SPECTROSCOPIC).
OBSMODE	Opmode	Operating mode (ACCUM, TIME-TAG).
EXPTYPE	Opmode	Exposure type (EXTERNAL/SCI, WAVECAL, PHA, DARK, FLAT, ACQ/IMAGE, ACQ/SEARCH, ACQ/PEAKD, ACQ/PEAKXD, ENG DIAG, OR MEM DUMP).
DETECTOR	Config	Detector in use (NUV or FUV).
SEGMENT	SEGMENT	FUV detector segment in use (FUVA, FUVB, BOTH, or N/A).
DETECTHV		FUV detector high voltage state (NOMAB, NOMA, NOMB, Off, Low).
SUBARRAY		Data from a subarray (T) or full frame (F).
LAMPUSED		Lamp status, NONE or name of lamp which is on (P1, D1, P2, or D2

Header Keyword	Phase II Equivalent	Description
LAMPSET		Spectral calibration lamp current value (milliamps).
LIFE_ADJ		Detector Life time adjustment position. (1-5)
OPT_ELEM	SpElement	Optical element in use (grating or mirror name).
CENWAVE	Wavelength	Central wavelength for grating settings.
APERTURE	Aperture	Aperture name.
PROPAPER	Aperture	Proposed aperture name.
APER_FOV		Aperture field of view description in mm.
FPPOS	FP-POS	Grating offset index (1-4) for spectrum dithers (FP-POS).
TAGFLASH	FLASH	Type of flashed exposures in TIME-TAG (NONE, AUTO, or UNIFORMLY-SPACED).
EXTENDED	Extended	Is the target extended (Yes or No).
NRPTEXP	NumberOfIterations	Number of repeat exposures in dataset: DEFAULT = 1.
EXP_NUM		Exposure number for repeated observations.
SHUTTER		External shutter position (OPEN or CLOSED).
Engineering Parameters (Primary Header)		
FPOFFSET		FP offset from nominal, in motor steps.
DEVENTA		Digital event counter, FUV segment A (counts s^{-1}).
DEVENTB		Digital event counter, FUV segment B (counts s^{-1}).
MEVENTS		NUV MAMA event counter (counts s^{-1}).
Target Acquisition Dataset Identifiers (Primary Header)		
ACQSNAME		Rootname of first acquisition search exposure.
ACQINAME		Rootname of first acquisition image exposure.
PEAKXNAM		Rootname of first x-dispersion peakup exposure.
PEAKDNAM		Rootname of first dispersion peakup exposure.
ACQ_NUM		Total number of exposures in acquisition sequence.
Archive Search Keywords (Primary Header)		
BANDWID		Bandwidth of the data.
SPECRES		Approximate resolving power at central wavelength.
CENTRWV		Central wavelength of the data.
MINWAVE		Minimum wavelength in spectrum.
MAXWAVE		Maximum Wavelength in spectrum.
PLATESC		Plate scale (arcsec/pixel).

Header Keyword	Phase II Equivalent	Description
Association Keywords (Primary Header)		
ASN_ID		Unique identifier assigned to association.
ASN_MTYP		Role of the member in the association.
ASN_TAB		Name of the association table.
Exposure Information (in Extension header 1 or greater)		
DATE-OBS		UT date of start of observation (yyyy-mm-dd).
TIME-OBS		UT time of start of observation (hh:mm:ss).
EXPTIME		Corrected exposure time (seconds).
RAWTIME		Exposure time of an individual raw exposure (seconds).
EXPSTART		Exposure start time (Modified Julian Date).
EXPEND		Exposure end time (Modified Julian Date).
EXPSTRTJ		Exposure start time (Julian Date).
EXPENDJ		Exposure end time (Julian Date).
PLANTIME	TimePerExposure	Planned exposure time (seconds).
NINTERPT		Number of exposure interrupts.
ORIENTAT		Position angle of image y-axis (degrees).
SUNANGLE		Angle between sun and V1 axis.
MOONANGL		Angle between moon and V1 axis.
SUN_ALT		Altitude of the sun above Earth's limb.
FGSLOCK		Commanded FGS lock (Fine, Coarse, Gyros, Unknown).
GYROMODE		Number of gyros scheduled for observation.
REFFRAME		Guide star catalog version.
Aperture Information (in Extension header 1 or greater)		
RA_APER DEC_APER		RA of reference aperture center. Declination of reference aperture center.
PA_APER		Position angle of reference aperture center.
SHIFT1A		Wavecal shift determined for FUV segment A or NUV stripe A in dispersion direction (pixels).
SHIFT1B		Wavecal shift determined for FUV segment B or NUV stripe B in dispersion direction (pixels).
SHIFT1C		Wavecal shift determined for NUV stripe C in dispersion direction (pixels).

Header Keyword	Phase II Equivalent	Description
SHIFT2A		Offset in cross-dispersion direction for FUV segment A or NUV stripe A (pixels).
SHIFT2B		Offset in cross-dispersion direction for FUV segment B or NUV stripe B (pixels).
SHIFT2C		Offset in cross-dispersion direction for NUV stripe C (pixels).
SP_LOC_A		Location of spectral extraction region for FUV segment A or NUV stripe A.
SP_LOC_B		Location of spectral extraction region for FUV segment B or NUV stripe B.
SP_LOC_C		Location of spectral extraction region for NUV stripe C.
SP_SLP_A		Slope of FUV segment A or NUV stripe A spectrum
SP_SLP_B		Slope of FUV segment B or NUV stripe B spectrum
SP_SLP_C		Slope of NUV stripe C spectrum
SP_WIDTH		Width (pixels) of extraction region
DPIXEL1A		Fractional part of pixel coordinate for FUV segment A or NUV stripe A (pixels). Average binning error.
DPIXEL1B		Fractional part of pixel coordinate for FUV segment B or NUV stripe B (pixels). Average binning error.
DPIXEL1C		Fractional part of pixel coordinate for NUV stripe C (pixels). Average binning error.
TIME-TAG Parameters (in Extension header 1 or greater)		
BUFFTIME	BUFFER-TIME	Onboard memory half-buffer-fill time.
OVERFLOW		Number of science data overflows.
NBADEVNT		Total number of events deleted in screening.
NEVENTS		Total number of events in raw data.
FUV TIME-TAG Parameters (in Extension header 1 or greater)		
TBRST_A TBRST_B		Time lost to bursts on FUV segment A (seconds). Time lost to bursts on FUV segment B (seconds).
TBADT_A TBADT_B		Time lost to BADTCORR screening on FUV segment A (sec). Time lost to BADTCORR screening on FUV segment B (sec).
NPHA_A NPHA_B		Number of events lost due to pulse height screening on segment A. Number of events lost due to pulse height screening on segment B.
NBRST_A NBRST_B		Number of events lost due to burst screening on segment A. Number of events lost due to burst screening on segment B.
NBADT_A NBADT_B		Number of events flagged by BADTCORR for segment A. Number of events flagged by BADTCORR for segment B.
NOUT_A NOUT_B		Number of events outside the active area for segment A. Number of events outside the active area for segment B.

Header Keyword	Phase II Equivalent	Description
NUV TIME-TAG Parameters (in Extension header 1 or greater)		
NBADT		Number of events flagged by BADTCORR.
NOUT		Number of events outside the active area.
TAFGFLASH Parameters (in Extension header 1 or greater)		
NUMFLASH		Integer number of flashes.
LMPDURi		Duration of flash i, seconds.
LMP_ONi		Lamp turn-on time for flash i, seconds since EXPSTART.
LMPOFFi		Lamp turn-off time for flash i, seconds since EXPSTART.
LMPMEDi		Median time of flash i, seconds since EXPSTART.
Velocity Reference Frame Conversion (in Extension header 1 or greater)		
V_HELIO		Geometric to heliocentric velocity.
V_LSRSTD		Heliocentric to standard solar LSR.
Doppler Correction Parameters (in Extension header 1 or greater)		
ORBITPER		Orbital period used on board for Doppler correction.
DOPPER		Doppler shift period (seconds).
DOPMAG		Doppler shift magnitude (low-res pixels).
DOPPMAGV		Doppler shift magnitude (Km/sec).
DOPPON		Doppler correction flag (T or F).
DOPPZERO		Commanded time of zero Doppler shift (MJD).
Instrument Doppler Correction Parameters (in Extension header 1 or greater)		
ORBTPERT		Orbital period used on board for Doppler correction.
DOPMAGT		Doppler shift magnitude (low-res pixels).
DOPPONT		Doppler correction flag (T or F).
DOPPZEROT		Commanded time of zero Doppler shift (MJD).
Global Count Parameters (in Extension header 1 or greater)		
GLOBRATE		Global count rate.
GLOBLIM		Was global linearity level exceeded.

Header Keyword	Phase II Equivalent	Description
Subbarray Readout Parameters (in Extension header 1 or greater)		
NSUBARRY[1]		Number of subarrays (1-8)
CORNER[N]X		Subbarray N axis1 corner pt in unbinned detector pixels.
CORNER[N]Y		Subarray N axis2 size in unbinned detector pixels.
SIZE[N]Y		Subbary N axis1 corner pt in unbinned detector pixels.
SIZE[N]X		Subarray N axis2 size in unbinned detector pixels.
Stim Pulse Parameters (in Extension header 1 or greater; for FUV data only)		
STIMRATE		Approximate STIM pulse injection rate.
STIMA_LX STIMA_LY STIMA_RX STIMA_RY		Segment A Left STIM pulse X centroid in raw data. Segment A Left STIM pulse Y centroid in raw data. Segment A Right STIM pulse X centroid in raw data. Segment A Right STIM pulse Y centroid in raw data.
STIMB_LX STIMB_LY STIMB_RX STIMB_RY		Segment B Left STIM pulse X centroid in raw data. Segment B Left STIM pulse Y centroid in raw data. Segment B Right STIM pulse X centroid in raw data. Segment B Right STIM pulse Y centroid in raw data.
STIMA0LX STIMA0LY STIMA0RX STIMA0RY		Reference location of Segment A Left STIM pulse X coordinate. Reference location of Segment A Left STIM pulse Y coordinate. Reference location of Segment A Right STIM pulse X coordinate. Reference location of Segment A Right STIM pulse Y coordinate.
STIMB0LX STIMB0LY STIMB0RX STIMB0RY		Reference location of Segment B Left STIM pulse X coordinate. Reference location of Segment B Left STIM pulse Y coordinate. Reference location of Segment B Right STIM pulse X coordinate. Reference location of Segment B Right STIM pulse Y coordinate.
STIMASLX STIMASLY STIMASRX STIMASRY		RMS width of Segment A Left STIM pulse X coordinate. RMS width of Segment A Left STIM pulse Y coordinate. RMS width of Segment A Right STIM pulse X coordinate. RMS width of Segment A Right STIM pulse Y coordinate.
STIMBSLX STIMBSLY STIMBSRX STIMBSRY		RMS width of Segment B Left STIM pulse X coordinate. RMS width of Segment B Left STIM pulse Y coordinate. RMS width of Segment B Right STIM pulse X coordinate. RMS width of Segment B Right STIM pulse Y coordinate.
Pulse Height Parameters (in Extension header 1 or greater for FUV data only)		
PHALOWRA PHALOWRB		Pulse height screening lower limit for segment A. Pulse height screening lower limit for segment B.
PHAUPPRA PHAUPPRB		Pulse height screening upper limit for segment A. Pulse height screening upper limit for segment B.

Header Keyword	Phase II Equivalent	Description
Image Statistics and Data Quality Flags (in Extension header 1 or greater for FUV data only)		
NGOODPIX		Number of good pixels.
SDQFLAGS		Serious data quality flags. (Can be modified as a **calcos** parameter see Section 3.4.17)
GOODMAX		Maximum value of good pixels.
GOODMEAN		Mean value of good pixels.
SOFTERRS		Number of soft error pixels (DQF=1).
Deadtime Correction Keywords (in Extension header 1 or greater for FUV data only)		
DEADRT DEADRT_A DEADRT_B		Count rate used for the NUV dead time correction (cps) Count rate used in the FUV Segment A dead time correction (cps) Count rate used in the FUV Segment B dead time correction (cps)
DEADMT DEADMT_A DEADMT_B		NUV Deadtime correction method (DATA, DEVENTS, or MEVENTS) FUVA Deadtime correction method (DATA, DEVENTS, or MEVENTS) FUVB Deadtime correction method (DATA, DEVENTS, or MEVENTS)
TIME-TAG Events Orientation Keywords (in Extension header 1 or greater)		
TCTYP2 TCTYP3		Axis type for dimension 1. Axis type for dimension 2.
TCRVL2 TCRVL3		Sky coordinates of 1st axis. Sky coordinate of 2nd axis.
TCDLT2 TCDLT3		Axis 1 degrees per pixels. Axis 2 degrees per pixels.
TCRPX2 TCRPX3		Axis 1 pixel of tangent plane direction. Axis 2 pixel of tangent plane direction.
TALEN2 TALEN3		Length of axis 1. Length of axis 2.
TC2_2 TC2_3 TC3_2 TC3_3		Partial of first axis coordinate with respect to x. Partial of first axis coordinate with respect to y. Partial of second axis coordinate with respect to x. Partial of second axis coordinate with respect to y.
TCUNI2 TCUNI3		Units of first coordinate value. Units of second coordinate value.
World Coordinate System and Related Parameters (in Extension header 1 or greater)		
WCSAXES		Number of Word Coordinate System axes.
CRPIX1 CRPIX2		x-coordinate of reference pixel. y-coordinate of reference pixel.
CRVAL1 CRVAL2		First axis value at reference pixel. Second axis value at reference pixel.

Header Keyword	Phase II Equivalent	Description
CTYPE1		The coordinate type for the first axis.
CYTPE2		The coordinate type for the second axis.
CD1_1		Partial of first axis coordinate with respect to x.
CD1_2		Partial of first axis coordinate with respect to y.
CD2_1		Partial of second axis coordinate with respect to x.
CD2_2		Partial of second axis coordinate with respect to y.
CUNIT1		Units of first coordinate value.
CUNIT2		Units of second coordinate value.
LTV1		Offset in X to subsection start.
LTV2		Offset in Y to subsection start.
LTM1_1		Reciprocal of sampling rate in X.
LTM2_2		Reciprocal of sampling rate in Y.

1. For FUV data subarrays 1-3 refer to segment A, and subarrays 4-7 refer to segment B.

Table 2.13: Spectroscopic Calibration Switch Keywords

EXPTYPE	EXTERNAL/SCI EXTERNAL/CAL				WAVECAL		DARK		FLAT	
DETECTOR	FUV		NUV		FUV	NUV	FUV	NUV	FUV	NUV
OBSMODE	TIME-TAG	ACCUM	TIME-TAG	ACCUM	TIME-TAG	TIME-TAG	TIME-TAG	TIME-TAG	TIME-TAG	TIME-TAG
Module										
BRSTCORR	Omit[1]	N/A	N/A	N/A	Omit[1]	N/A	Omit[1]	N/A	Omit[1]	N/A
BADTCORR	Perform	N/A	Perform	N/A	Perform	Perform	Perform	Perform	Perform	Perform
PHACORR	Perform	Perform	N/A	N/A	Perform	N/A	Perform	N/A	Perform	N/A
RANDCORR	Perform	Perform	N/A	N/A	Perform	N/A	Perform	N/A	Perform	N/A
RANDSEED	-1	-1	N/A	N/A	-1	N/A	-1	N/A	-1	N/A
TEMPCORR	Perform	Perform	N/A	N/A	Perform	N/A	Perform	N/A	Perform	N/A
GEOCORR	Perform	Perform	N/A	N/A	Perform	N/A	Perform	N/A	Perform	N/A
IGEOCORR	Perform	Perform	N/A	N/A	Perform	N/A	Perform	N/A	Perform	N/A
DOPPCORR	Perform	Perform	Perform	Perform	N/A	N/A	N/A	N/A	N/A	N/A
DEADCORR	Perform	Perform	Perform	Perform	Perform	Perform	Perform	Perform	Perform	Perform
FLATCORR	Omit[2]	Omit[2]	Perform	Perform	Omit[2]	Perform	Omit	Omit	Omit	Omit
DQICORR	Perform	Perform	Perform	Perform	Perform	Perform	Perform	Perform	Perform	Perform
WAVECORR	Perform	Perform	Perform	Perform	Perform	Perform	Omit	Omit	Omit	Omit
X1DCORR	Perform	Perform	Perform	Perform	Perform	Perform	Omit	Omit	Omit	Omit
BACKCORR	Perform	Perform	Perform	Perform	Perform	Perform	Omit	Omit	Omit	Omit
FLUXCORR	Perform	Perform	Perform	Perform	N/A	N/A	Omit	Omit	Omit	Omit
TDSCORR	Perform	Perform	Perform	Perform	N/A	N/A	Omit	Omit	Omit	Omit
HELCORR	Perform	Perform	Perform	Perform	N/A	N/A	Omit	Omit	Omit	Omit
STATFLAG	T	T	T	T	T	T	T	T	T	T

1. BRSTCORR is set to Omit until further information about FUV bursts has been determined on-orbit.
2. FLATCORR will be set to Omit for the FUV detector, until a sufficient on-orbit flat has been created.

Table 2.14: Imaging Calibration Switch Keywords

EXPTYPE	EXTERNAL/SCI EXTERNAL/CAL		WAVECAL	DARK	FLAT	ACQ/IMAGE
DETECTOR	NUV		NUV	NUV	NUV	NUV
OBSMODE	TIME-TAG	ACCUM	TIME-TAG	TIME-TAG	TIME-TAG	ACCUM
Modules						
BADTCORR	Perform	N/A	Perform	Perform	Perform	N/A
FLATCORR	Perform	Perform	Perform	Omit	Omit	Perform
DEADCORR	Perform	Perform	Perform	Perform	Perform	Perform
DQICORR	Perform	Perform	Perform	Perform	Perform	Perform
PHOTCORR	Perform	Perform	Perform	Omit	Omit	Perform
TDSCORR	Perform	Perform	Perform	Omit	Omit	Perform
STAFLAG	T	T	T	T	T	T

Table 2.15: Reference File Keywords

Reference File	Description
BRSTTAB	Burst parameter table
BRFTAB	Baseline reference frame reference table
BADTTAB	Bad time interval reference table
PHATAB	Pulse height discrimination reference table
GEOFILE	Geometric distortion table
DEADTAB	Deadtime reference file
FLATFILE	Pixel to pixel flat field reference file
LAMPTAB	Template calibration lamp spectra table
WCPTAB	Wavecal parameters table
DISPTAB	Dispersion coefficient reference table
BPIXTAB	Bad pixel table
XTRACTAB	1-D Spectral extraction information table
PHOTTAB	Photometric throughput table
TDSTAB	Time-dependent sensitivity correction table

2.7 Error and Data Quality Array

The COS pipeline propagates both statistical errors and data quality flags throughout the calibration process. These are then combined from both the science data and the reference file data to produce data quality information in the calibrated data.

2.7.1 Error Array

The error array contains an estimate of the statistical error at each pixel. In the raw file, the error array is empty, and in the calibrated files the error array is generated from Poisson statistics of the gross counts and propagation of the errors to account for those introduced by correcting the observed data for flat field and dead time effects.

2.7.2 Data Quality Flags

Data quality flags are assigned to each pixel in the data quality extension. Each flag has a true (set) or false (unset) state. Flagged conditions are set as specific bits in a 16-bit integer word. For a single pixel, this allows for up to 15 data quality conditions to be flagged simultaneously, using the bitwise logical OR operation. Note that the data quality flags cannot be interpreted simply as integers but must be converted to base-2 and interpreted as individual flags. Table 2.16 gives the specific conditions that are flagged, depending on the states of different bits (i.e., being on or off).

There are basically three types of DQ flags.

6. Temporal flags, which refer to time intervals that are pre-determined as bad for some reason, or time intervals that are determined to be contaminated by bursts.
7. Event flags, which involve events whose pulse heights lie outside of a pre-determined range, and are suspected to result from non-photon events.
8. Spatial flags, which refer to locations on the detector that have been determined to be bad, or suspect.

Screening for the first two types of flags is done by turning calibration switches on or off, or by altering reference files. Once a photon has been determined to have a bad temporal or event flag, it will never appear in a final data product unless the moduals which screen for it are turned off or the reference files which define them are changed. On the other hand, the screening for the spatial flags can be easily altered by changing the SDQFLAGS keyword in the header of the raw data file.

The raw data quality files will be filled only when there are missing (data lost) or dubious (software error) data. If no such errors exist, initialization will produce an empty data quality extension whose header has `NAXIS=0`. These flags are set and used during the course of calibration, and may likewise be interpreted and used by downstream analysis applications. See Section 3.4.12 for more information on the data quality initialization calibration module. STATFLAGS Section 3.4.17

Table 2.16: COS Data Quality Flags

FLAG Value	Bit Setting	Quality Condition Indicated
1	0000 0000 0000 000**1**	Error in the Reed-Solomon decoding (an algorithm for error correction in digital communications).
2	0000 0000 0000 00**1**0	Brush mark
4	0000 0000 0000 0**1**00	Grid shadow mark
8	0000 0000 0000 **1**000	Spectrum near an edge of the detector
16	0000 0000 000**1** 0000	Dead spot
32	0000 0000 00**1**0 0000	Hot spot
64	0000 0000 0**1**00 0000	Burst
128	0000 0000 **1**000 0000	Outside a subarray
256	0000 000**1** 0000 0000	Lost data replaced by fill values.
512	0000 00**1**0 0000 0000	Pulse Height too low
1024	0000 0**1**00 0000 0000	Pulse Height too high
2048	0000 **1**000 0000 0000	Data within bad time interval
4096	000**1** 0000 0000 0000	Wavelength is below 900A
8192	00**1**0 0000 0000 0000	Wrinkled appearance from detector flat field
16384	0**1**00 0000 0000 0000	Vertical S distortion on FUV segment A

CHAPTER 3:

COS Calibration

In this chapter...

3.1 Raw Data Compilation

The basic input to **calcos** is raw science data files. This section provides a brief overview of how these files are generated from raw spacecraft telemetry.

Telemetry containing COS science data is downlinked from the *HST* satellite through a Tracking and Delay Relay Satellite System (TDRSS) satellite to a ground station in White Sands, NM. From there it is sent to Goddard Space Flight Center where the data capture facility packet processor (PACOR) collects the downlinked science data and places them into telemetry "pod files". These pod files are then transmitted to STScI where they are saved to a permanent storage medium. The STScI ingest pipeline, OPUS, then unpacks the data, populates keyword values extracted from the telemetry stream, reformats the data, and repackages them into raw, uncalibrated, but scientifically interpretable data files.

The raw files are then processed by the **calcos** software to produce a variety of calibrated data files. The results of these procedures are used to populate the databases that form the searchable archive catalog at STScI describing the individual instrument exposures. At this point, the raw and calibrated data files generated from the pod files are normally discarded.

Only the pod files and the information placed in the archive databases are preserved. Each time a user requests data from the Hubble Data Archive via the "On The Fly Reprocessing" (OTFR) system, the raw files are regenerated from the original pod files, and then recalibrated with the latest reference files. A more detailed description of the OTFR system as it applies to COS and other *HST* instruments can be found in Swade et al, available on-line at

http://www.adass.org/adass/proceedings/adass00/P2-36/.

Although the OTFR system uses pod files for input, **calcos** uses the raw files (together with their association files) as its initial input. Figure 1.7 shows a raw image of one segment of the COS FUV XDL detector.

3.2 Pipeline Processing Overview

The calibration pipeline, **calcos**, has been developed by the Space Telescope Science Institute (STScI) to support the calibration of *HST*/COS data. Although the COS pipeline benefits from the design heritage of previous *HST* instruments and of the Far Ultraviolet Spectroscopic Explorer (FUSE), the **calcos** modules are tailored specifically to the COS instrument and based on data reduction algorithms defined by the COS Instrument Definition Team (IDT). As with other *HST* pipelines, **calcos** uses an association table (the `_asn` files) to specify the data files to be included, and employs header keywords to specify the calibration steps to be performed and the reference files to be used. **Calcos** is written in Python, which enables the pipeline and user to take advantage of an extremely productive, open-source, easy-to-read scripting language, with many libraries for data reduction and analysis. **Calcos** is in the **stsci_python** package, which is available for download from STScI:

http://www.stsci.edu/resources/software_hardware/pyraf/stsci_python/current/download

Calcos is designed with a common underlying structure for processing FUV and NUV channels which, respectively, use a cross delay line (XDL) and a Multi Anode Microchannel Array (MAMA) detector. The **calcos** calibration pipeline includes pulse-height filtering and geometric correction for the FUV channel, and flat-field, deadtime, and Doppler correction for both channels. It includes methods for obtaining an accurate wavelength calibration by using the on-board spectral line lamps. A background subtracted spectrum is produced and the instrument sensitivity is applied to create the final flux calibrated spectrum.

There are two basic types of raw data files: `TIME-TAG` photon lists and `ACCUM` images of the detector. In general, **calcos** must convert these into one dimensional calibrated flux and wavelength arrays. **Calcos** must be able to perform different types of calibration processes to accommodate the different input types.

The level of calibration performed depends upon the data type.

- Acquisition mode exposures (`ACQ/SEARCH, ACQ/IMAGE, ACQ/PEAKXD`, and `ACQ/PEAKD`) are not calibrated by **calcos**. Only the raw data from these modes are provided.
- All other science data are completely calibrated. This includes geometric and thermal correction for the FUV data, flat fielding, linearity corrections and pulse height filtering. The data flow and calibration modules for processing the data are described in detail in sections 3.3 and 3.4.
- Raw data taken in `TIME-TAG` mode are event lists (`rawtag` binary tables). The basic calibration is done on the tabular data, producing a calibrated (`corrtag`) events table. The events are then accumulated into an calibrated image (`flt`) by **calcos**.
- Raw data taken in `ACCUM` mode (`_rawaccum`) are binned to an image array on board the spacecraft.
- For spectral data, **calcos** extracts a spectrum from the flat-fielded image, computes associated wavelengths, and converts the count rates to flux densities, yielding a one-dimensional, background subtracted spectrum. For FUV data there will normally be two spectra, one from segment A and one from segment B. For NUV data there will normally be three spectra, one for each spectral "stripe".
- When multiple exposure comprise an observation, these are combined into a single, summed spectrum.

See Chapter 2 for the naming conventions of the various input, temporary, and output calibrated files.

3.3 Calcos: Structure and Data Flow

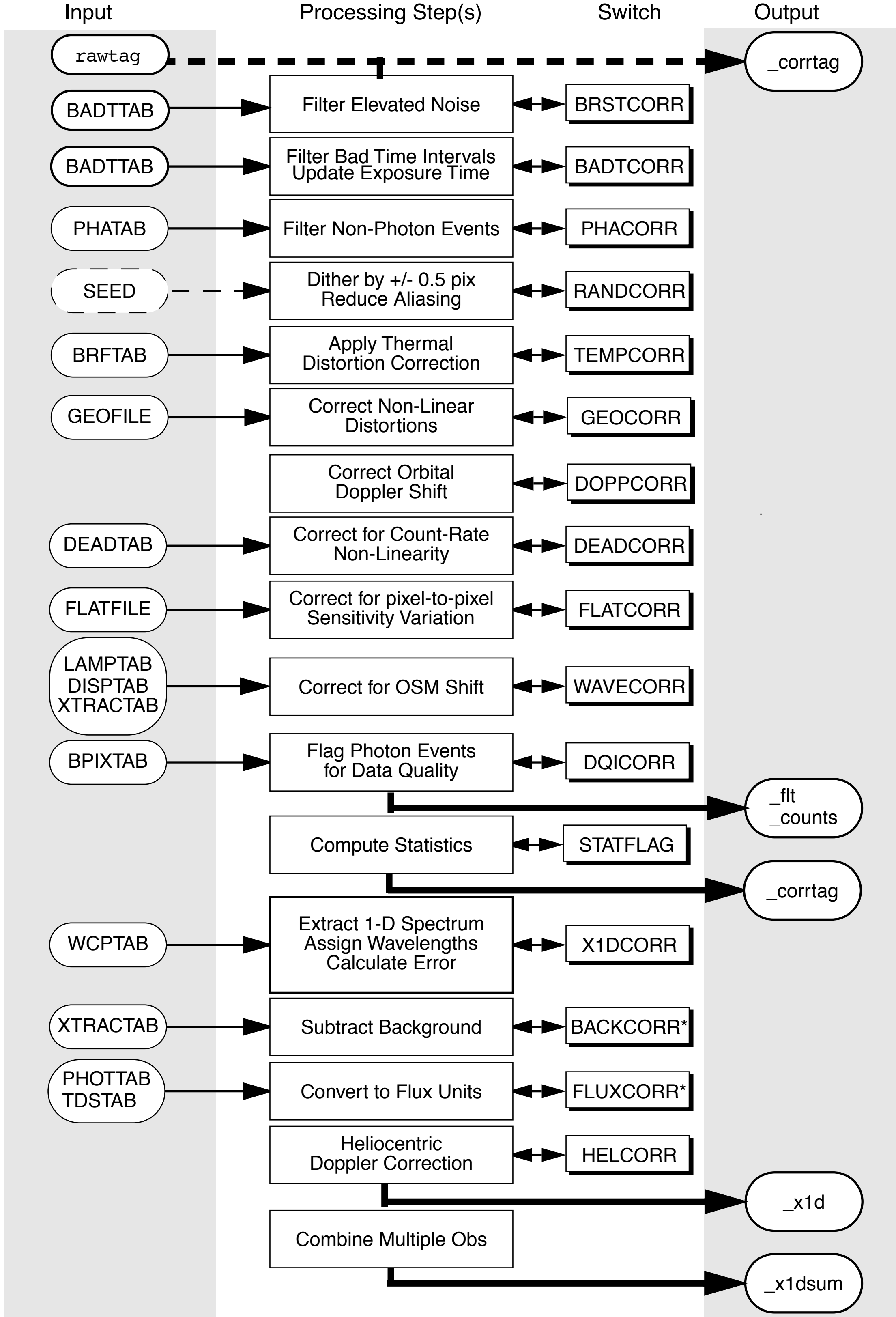
Calcos is comprised of three main components that provide calibration of the COS data. This structure incorporates modules that (1) correct the data for instrument effects (e.g. noise, thermal drifts, geometric corrections, pixel-to-pixel variations in sensitivity), (2) generate an exposure-specific wavelength-calibrated scale, and (3) extract and produce the final (one-dimensional) flux-calibrated (summed) spectrum for the entire observation. Both COS FUV and NUV `TIME-TAG` event lists and `ACCUM` images are fully calibrated by **calcos**. Target acquisition exposures are not calibrated by **calcos**, although the raw data from these observations are available through the data archive.

As with *HST* calibration pipelines for other instruments, the choice of which operations are performed during calibration is controlled by calibration switches, which are stored in the primary FITS header. OPUS sets the switches that are appropriate for a given data type to PERFORM for steps to be carried out by **calcos**, and then **calcos** changes them to COMPLETE in the calibrated files. When OPUS creates the raw data files, it also populates the headers with the names of the appropriate reference files for each calibration operation. Any calibration step may require zero, one, or more calibration reference files; the names of these files are also listed in the headers. Exactly how the data are processed depends on whether they are FUV TIME-TAG or ACCUM spectra, NUV TIME-TAG or ACCUM spectra, or NUV images. The names of the keywords containing the switches and reference file names were introduced in Table 2.13, and their roles in the data reduction and the calibration steps are described in the following sections.

Unlike many HST calibration pipelines, calcos was not designed so that its modules can be run independently, i.e. it is not re-entrant. The only ways to modify the pipeline flow are by changing the calibration switches or the contents of the reference files (see Section 3.6.1).

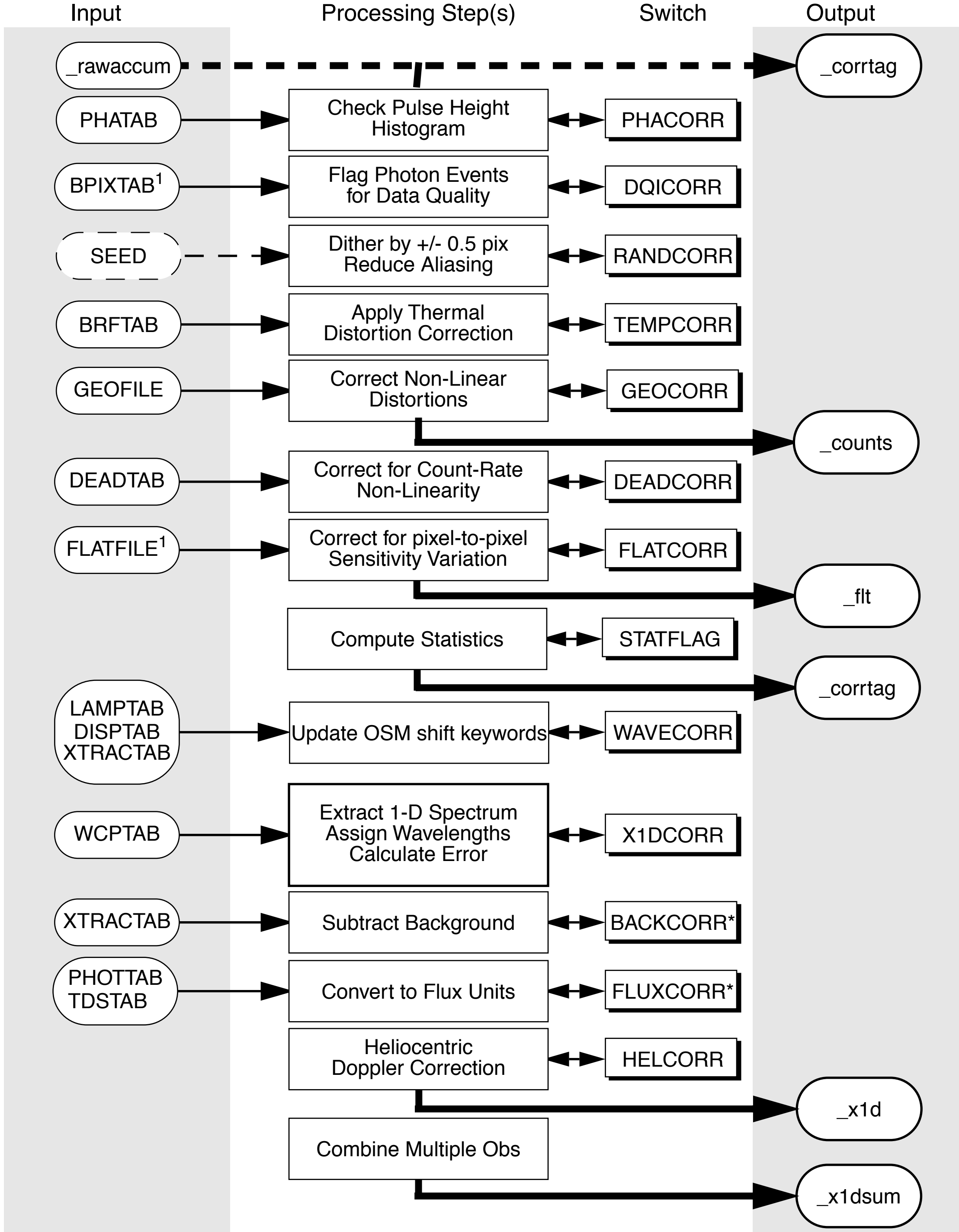
Figure 3.1 - Figure 3.5 show how a single raw file moves through the pipeline for FUV TIME-TAG, FUV ACCUM, NUV TIME-TAG and NUV ACCUM spectroscopic data, and for NUV images. Each Figure shows, from left to right, the input files, the processing steps performed by each module, and the output files. Note that in some instances, output files are created and then subsequently modified. In these cases, the file is shown at the end of a dashed arrow at the point it is created and again by a solid arrow at the point where it is finalized. Steps that apply only when spectra are extracted are marked with an * in Figure 3.1 through Figure 3.4. For ACCUM data, Doppler corrections are done on board. Consequently, for these spectra certain reference files are transformed to the coordinate system of the data, rather than the other way around. We note on Figure 3.2 and Figure 3.4 when this is done.

Figure 3.1: FUV TIME-TAG spectroscopic pipeline flow chart.



* These steps are only implemented if X1DCORR=PERFORM.

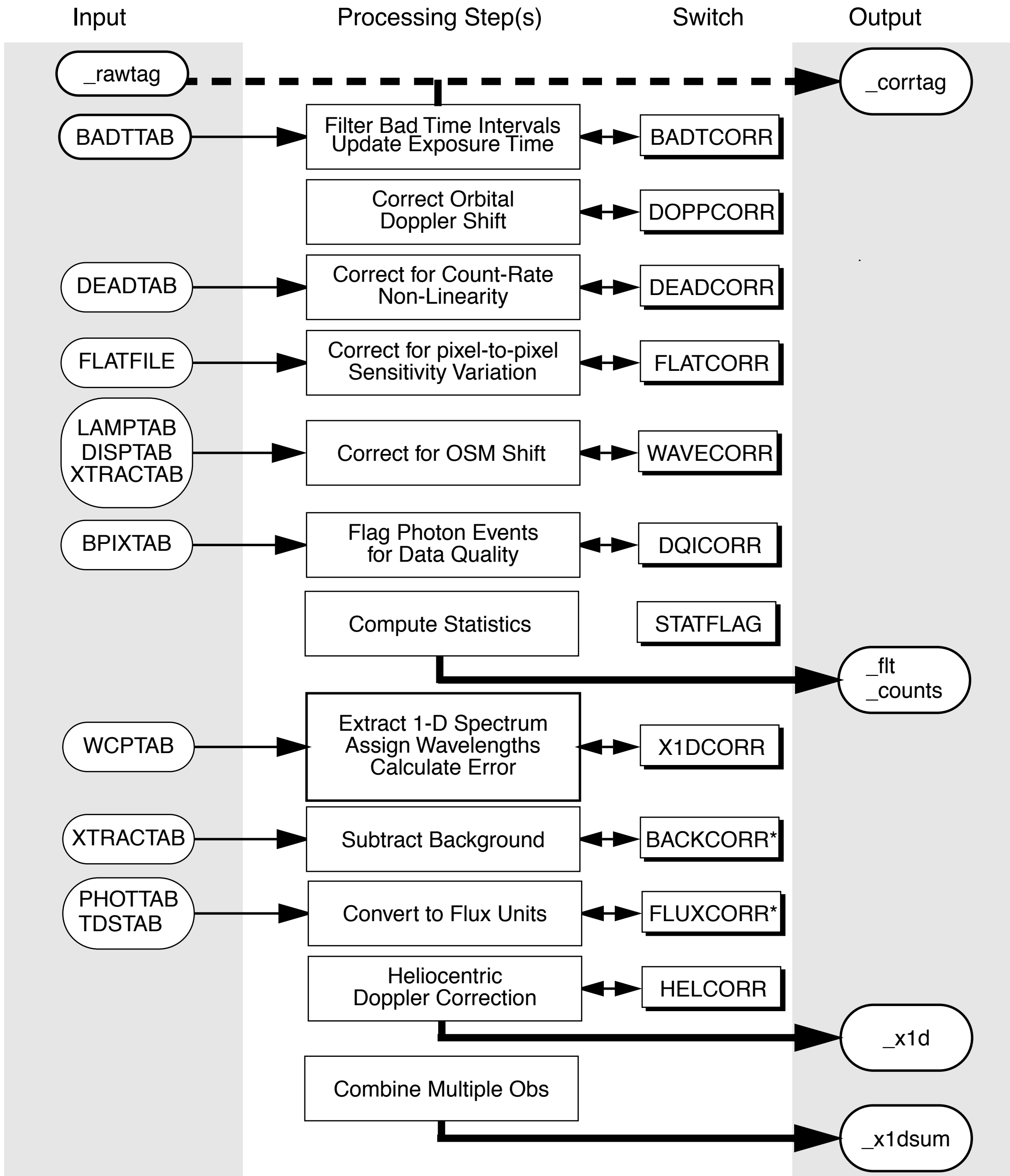
Figure 3.2: FUV ACCUM spectroscopic pipeline flow chart.



* These steps are only implemented if `X1DCORR=PERFORM`.

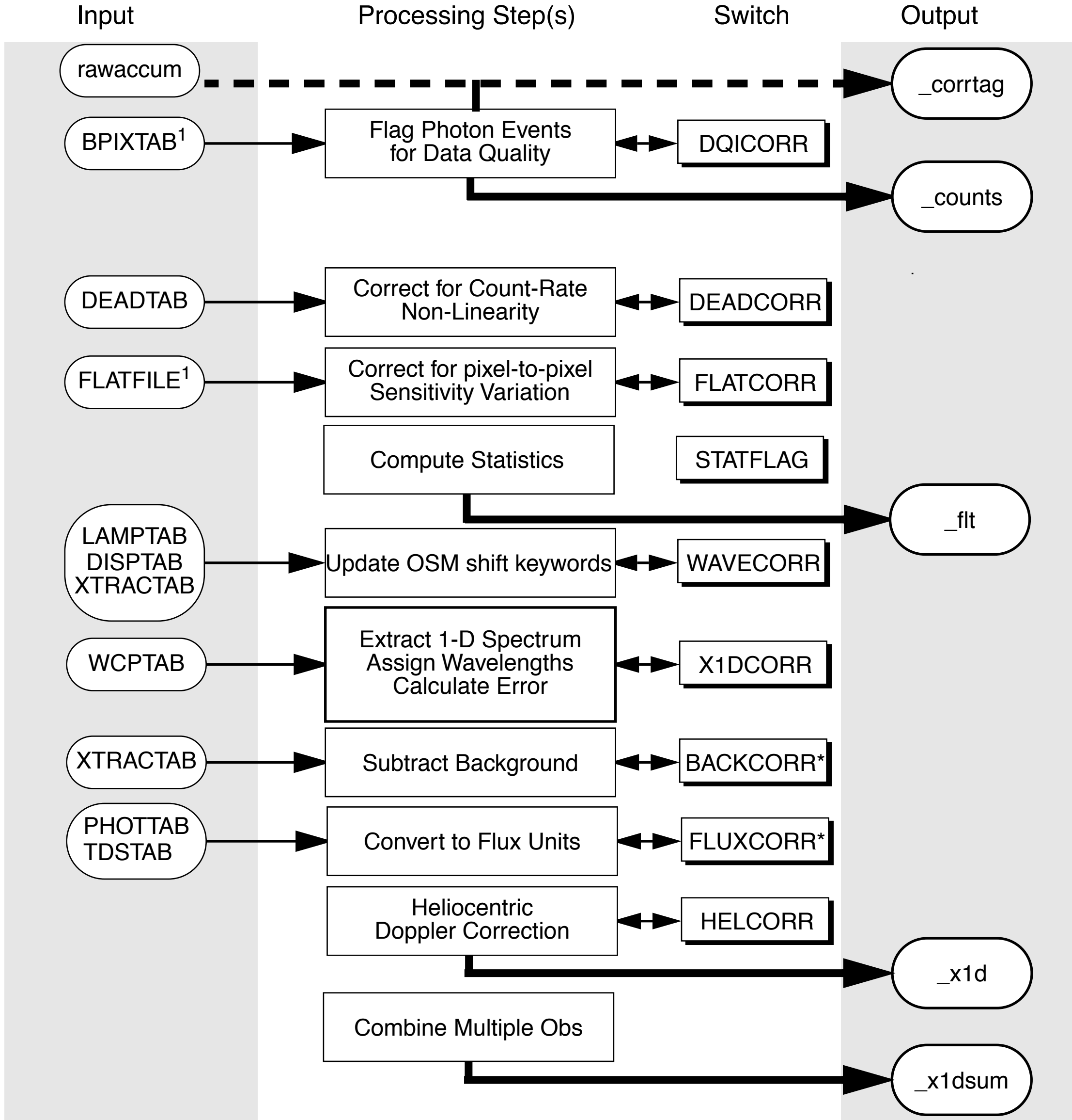
[1] Reference files that are transformed to the doppler corrected coordinate system of the data before being applied

Figure 3.3: NUV TIME-TAG spectroscopic pipeline flow chart



* These steps are only implemented if `X1DCORR=PERFORM`.

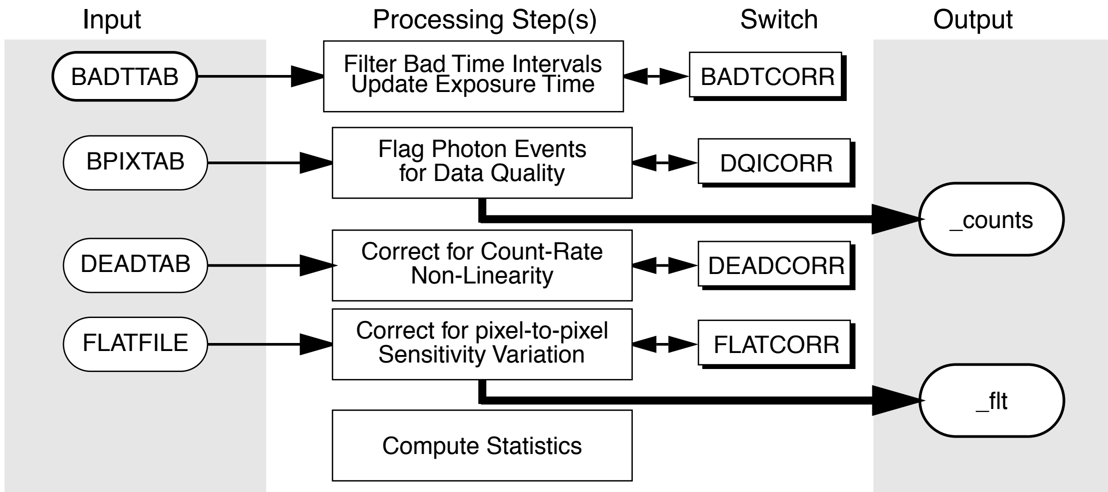
Figure 3.4: NUV ACCUM spectroscopic pipeline flow chart



* These steps are only implemented if `X1DCORR=PERFORM`.

[1] Reference files that are transformed to the doppler corrected coordinate system of the data before being applied.

Figure 3.5: NUV image pipeline flow chart



3.4 Descriptions of Spectroscopic Calibration Steps

This section provides a more detailed description of the calibration processing steps and algorithms applied by **calcos**, including the switches, reference file inputs, science file inputs, and the output products. Setting the calibration switch to PERFORM enables the execution of the corresponding pipeline calibration task.

Future modifications and updates to **calcos** will be announced in STScI Analysis Newsletters (STANs) and documented at the Web site:

http://www.stsci.edu/hst/cos/updates/calcos.

The calibration steps are described in order of their occurrence for FUV TIME-TAG data in the pipeline processing flow. When present, each sub-section header begins with the switch that activates the module.

3.4.1 Initialization

When the pipeline is initiated, it first opens an association file to determine which files should be processed together. For TIME-TAG data, it also creates a _corrtag file before anything else is done. The initial contents of this file is simply a copy of the contents of the _rawtag file. It is then updated throughout the running of the pipeline.

3.4.2 BRSTCORR: Search for and Flag Bursts

This module flags "event bursts" in the FUV TIME-TAG data for removal.

- Reference file: BRSTTAB, BRFTAB
- Input files: FUV _rawtag
- Header keywords updated: TBRST_A, TBRST_B (time affected by bursts in segments A and B), NBRST_A, NBRST_B (number of events flagged as bursts in segments A and B), EXPTIME.
- Updates _corrtag file for TIME-TAG data.

The COS FUV detectors are similar to the FUSE detectors, which experienced sudden, short-duration increases in counts while collecting data. These events, called bursts, led to very large count rates and occurred over the entire detector. It is still uncertain if COS will suffer from the same phenomenon on-orbit. As a precaution, the BRSTCORR module screens the data to identify bursts and flags the time intervals in which they occur. This module is only applied to FUV TIME-TAG data.

The first step in the screening process is to determine the count rate over the whole detector, including stim pulses, source, background, and bursts. This rate determines which time interval from the BRSTTAB table to use for screening.

Screening for bursts is then done in two steps. The first step identifies large count rate bursts by calculating the median of the counts in the background regions, defined in the XTRACTAB table, over certain time intervals (DELTA_T or DELTA_T_HIGH for high overall count rate data). Events with count rates larger than MEDIAN_N times the median are flagged as large bursts.

The search for small count rate bursts is done iteratively, up to MAX_ITER. This step uses a boxcar smoothing of the background counts (taking the median within the box) and calculates the difference between the background counts and the running median. The boxcar smoothing is done over a time interval MEDIAN_DT or MEDIAN_DT_HIGH. Elements that have already been flagged as bursts are not included when computing the median. For an event to be flagged as affected by a small burst the difference between the background counts and the running median has to be larger than the following quantities:

1. A minimum burst count value: BURST_MIN * DELTA_T (or DELTA_T_HIGH for large overall count rates),
2. A predetermined number of standard deviations above the background: STDREJ * square_root(background counts),
3. A predetermined fraction of the source counts: SOURCE_FRAC * source counts.

The source counts value in 3) is the number of events in the source region defined in the XTRACTAB table minus the expected number of background counts within that region.

All events that have been identified as bursts are flagged in the data quality column (DQ in the corrtag table) with data quality bit = 64. In addition **calcos** updates the following header keywords to take into account time and events lost to burst screening: TBRST_A and TBRST_B (time lost to bursts in segments A and B); NBRST_A, NBRST_B (number of events lost to bursts in segments A and B), and EXPTIME. For _x1dsum files the value of the EXPTIME keyword is that corresponding to the exposure time for segment A whenever this segment is present. The EXPTIME for each segment is contained in a table column in the first extension of the x1dsum file.

When running **calcos** a user can specify that the information about bursts be saved into a file. This output file contains four columns, one row per time interval (DELTA_T or DELTA_T_HIGH): Column 1 contains the time (seconds) at the middle of the time interval; column 2 contains the background counts for that time interval, column 3 = 1 for time intervals with large bursts and = 0 elsewhere, and column 4 = 1 for time intervals with small bursts and = 0 elsewhere.

The initial values in the burst parameters table (BRSTTAB) are based on expectations based on FUSE, and are likely to be modified once COS is installed in the Hubble Space Telescope and more is learned about the bursts (if any) seen by the FUV detector.

3.4.3 BADTCORR: Bad Time Intervals

This module flags time intervals in TIME-TAG data that have been identified as bad for some reason.

- Reference file: BADTTAB
- Input files: FUV and NUV rawtag
- Header keywords updated: EXPTIME, NBADT, or NBADT_A and NBADT_B (number of events flagged for NUV or FUVA and B, respectively) and TBADT or TBADT_A and TBADT_B (time lost to bad events in NUV or FUVA and FUVB, respectively).
- Updates corrtag file for TIME-TAG data.

The BADTTAB table lists zero or more intervals of time which will be excluded from the final spectrum for various reasons. This file is currently empty, but it could be populated by the COS team if events occur on orbit which they feel render data collected during specific time intervals not scientifically useful. It is also available for the convenience of the user. For example, the user may wish to eliminate observations obtained in the daytime portion of the orbit to minimize airglow contamination, or they may want to isolate a certain portion of an exposure. In these cases, modifying BADTTAB may be the most convenient means to accomplish this. Events in the rawtag file having times within any bad time interval in BADTTAB are flagged in the DQ column of the corrtag table with data quality = 2048. The exposure time is updated to reflect the sum of the good time intervals. This step applies only to TIME-TAG data.

3.4.4 PHACORR: Pulse Height Filter

This module operates on FUV data and flags events whose pulse heights are outside of nominal ranges.

- Reference file: PHATAB
- Input files: FUV rawtag, FUV rawaccum
- Header keywords updated: NPHA_A, NPHA_B, PHAUPPRA, PHAUPPRB, PHALOWRA, PHALOWRB

This module works differently for FUV TIME-TAG and ACCUM data. It is not used for NUV data.

For FUV TIME-TAG data, each photon event includes a 5 bit (0 - 31) Pulse Height Amplitude (PHA). The value of the pulse height is a measure of the charge produced by the microchannel plate stack, and can be used to identify events which are likely due to cosmic rays or detector background. The PHATAB reference file lists lower and upper pulse height thresholds expected for valid photon events for each detector segment. The PHACORR module compares each event's pulse height to these thresholds, and if the pulse height is below the Lower Level Threshold (LLT) or above the Upper Level Threshold (ULT), the event is flagged in the DQ column of the corrtag table with a data quality bit of 512 or 1024, respectively. The upper and lower thresholds are also written to the PHALOWRA (PHALOWRB) and PHAUPPRA (PHALOWRB) keywords in the output data files for segment A (B), while the number of events flagged is written to the NPHA_A and NPHA_B keywords.

Default values of the lower (LLT) and upper (ULT) thresholds have been chosen based on the properties of the detector and are implicit in data used when generating other reference files (e.g. FLUXTAB).

Modifying these threshold values could lead to incorrect results in the calibrated products, and should therefore be done with EXTREME caution.

For FUV ACCUM data, pulse height information is not available for individual events. However, a 7 bit (0 - 127) Pulse Height Distribution (PHD) array, containing a histogram of the number of occurrences of each pulse height value over the entire detector segment, is created onboard for each exposure. PHACORR compares the data in this pha file to the values in the PHATAB file. Warnings are issued if the peak of the distribution (modal gain) does not fall between the scaled values of LLT and ULT; or if the average of the distribution (mean gain) does not fall between the MIN_PEAK and MAX_PEAK values in PHATAB. The PHALOWRA and PHAUPPRA, or PHALOWRB and PHAUPPRB keywords are also populated in the output files with the LLT and ULT values from the PHATAB.

3.4.5 RANDCORR: Add Pseudo-Random Numbers to Pixel Coordinates

This module adds a random number between -0.5 and +0.5 to each x and y position of a photon detected by the FUV detectors.

- Reference file: none
- Input files: FUV rawtag, FUV rawaccum
- Header keywords updated: RANDSEED
- Updates corrtag file for TIME-TAG data and a virtual corrtag file for ACCUM data.

For FUV TIME-TAG data RANDCORR adds random numbers to the raw coordinates of each event, i.e.:

$$XCORR = RAWX + \Delta x$$

$$YCORR = RAWY + \Delta y$$

Where Δx and Δy are uniformly distributed, pseudo-random numbers in the interval $-0.5 < \Delta x, \Delta y \leq +0.5$.

The result of this operation is to convert the raw, integer pixel values into floating point values so that the counts are smeared over each pixel's area.

For FUV ACCUM data, a pseudo TIME-TAG list of x and y values is created with an entry for each recorded count. Next, a unique Δx and Δy are added to each entry and then the image is reconstructed.

If the RANDSEED keyword in the raw data file is set to its default value of -1, the system clock is used to create a seed for the random number generator. This seed value is then written to the RANDSEED keyword in the output files. Alternatively, an integer seed (other than -1) in the range -2147483648 to +2147483647 can be specified by modifying the RANDSEED keyword in the raw data file. Doing so will ensure that identical results will be obtained on multiple runs.

RANDCORR is only applied to events in the active area of the detector, as defined in the BRFTAB. Stim pulses, for example, do not have this correction applied.

3.4.6 TEMPCORR: Temperature-Dependent Distortion Correction

This module corrects for linear distortions of the FUV detector coordinate system that are caused by changes in the temperature of the detector electronics.

- Reference file: BRFTAB
- Input files: FUV rawtag, FUV rawaccum
- Header keywords updated: none
- Updates corrtag file for TIME-TAG data and flt file ACCUM.

The FUV XDL detector has virtual, not physical, detector elements that are defined by the digitization of an analog signal. The charge packet associated with a photon event is split and transported to opposite sides of the detector where the difference in travel time of the two packets determines the location of the photon event on the detector. Since the properties of both the delay line and the sensing electronics are subject to variations as a function of temperature, apparent shifts and stretches in the detector format can occur.

To measure the magnitude of this effect, electronic pulses (Figure 1.6) are recorded at two reference points in the image (“electronic stims”) at specified time intervals throughout each observation. TEMPCORR first determines the locations and separations of the recorded stim positions and then compares them to their expected locations in a standard reference frame (as defined in columns SX1, SY1, SX2, and SY2 of the BRFTAB file). The differences between the observed and reference stim positions are used to construct a linear transformation between the observed and reference frame locations for each event (or pseudo-event in the case of ACCUM data). TEMPCORR then applies this transformation to the observed events, placing them in the standard reference frame.

3.4.7 GEOCORR and IGEOCORR: Geometric Distortion Correction

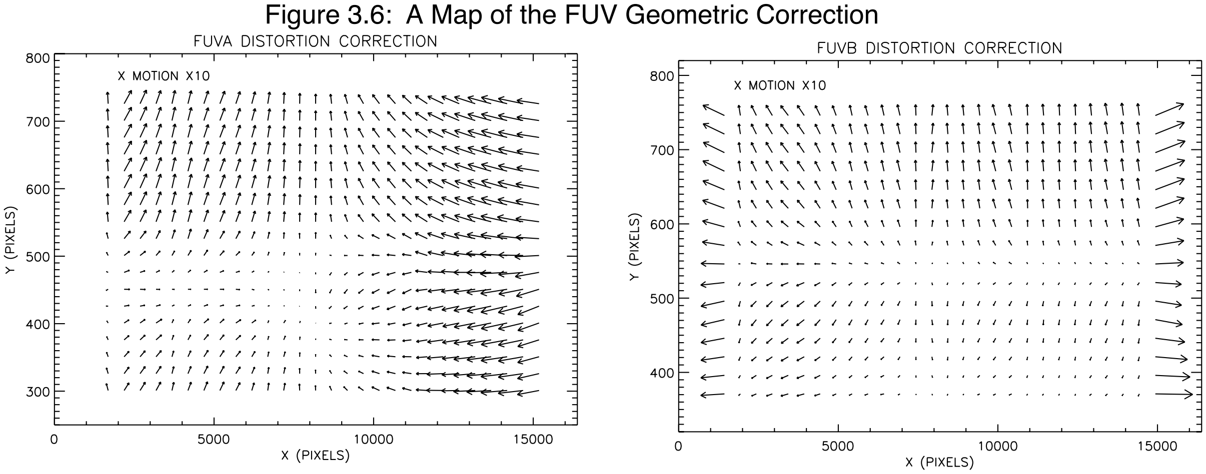
This module corrects geometric distortions in the FUV detectors.

- Reference file: GEOFILE
- Input files: FUV rawtag, FUV rawaccum
- Header keywords updated: none
- Updates corrtag file for TIME-TAG data and flt file ACCUM.

The GEOCORR module corrects for geometric distortions due to differences between the inferred and actual physical sizes of pixels in the FUV array (ground measurements indicated that geometric distortions in the NUV MAMA are negligible). It produces a geometrically corrected detector image with equal sized pixels. This is done by applying the displacements listed in the reference file, GEOFILE, which lists the corrections in x and y for each observed pixel location. Since RANDCORR has converted the pixel locations to fractional values, IGEOCORR interpolates the contents of GEOFILE to the fractional pixel locations. The default is IGEOCORR='PERFORM'.

`GEOFILE` was created by using a ray-trace analysis of the COS FUV optical system. A set of wavelength calibration exposures was taken while stepping the aperture mechanism in the cross-dispersion direction to create an image of dispersed line profiles. The ray trace and measured line positions were compared to determine the shift between the measured (uncorrected) and predicted (corrected) line positions (see Figure).

Figure 3.6: A Map of the FUV Geometric Correction



A map of the FUV geometric correction, scaled by a factor of 10 in the x-direction for illustration purposes, for detector segment A (left) and segment B (right) in user coordinates. The arrow points from the observed to the corrected position.

3.4.8 DOPPCORR: Correct for Doppler Shift

This module corrects for the effect that the orbital motion of *HST* has on the arrival location of a photon in the dispersion direction.

- Reference files: `DISPTAB`
- Input files: FUV/NUV `rawtag`, FUV/NUV `rawaccum`
- Header keywords updated: none

During a given exposure the photons arriving on the FUV and NUV detectors are Doppler shifted due to the orbital motion of *HST*. The orbital velocity of *HST* is 7.5 km/s, so spectral lines in objects located close to the orbital plane of *HST* can be broadened up to 7.5 km/s, which can be as much as 60% of a resolution element.

DOPPCORR is the **calcos** routine which corrects for the orbital motion of *HST*. It operates differently on TIME-TAG and ACCUM files:

For TIME-TAG files the raw events table contains the actual detector coordinates of each photon detected, i.e., the photon positions will include the smearing from the orbital motion. In this case DOPPCORR will add an offset to the pixel coordinates (the RAWX column) in the events table to undo the Doppler broadening. The corrected coordinates are written to the column XDOPP in the `corrtag` file for both FUV and NUV data.

For ACCUM files Doppler correction is applied onboard and is not performed by **calcos**. This means, however, that the pixel coordinates of a spectral feature can differ from where the photon actually hit the detector - a factor which affects the data quality initialization and flat field correction. Therefore for ACCUM images DOPPCORR shifts the positions of pixels in the bad pixel table to determine the maximum bounds that could be affected. It is also used to convolve the flat field image by an amount corresponding to the Doppler shift which was computed on orbit. The information for these calculations are contained in the following header keywords:

- DOPPONT: True if Doppler correction was done on-board.
- ORBTPERT: Orbital period of *HST* in seconds.
- DOPMAGT: Magnitude of the Doppler shift in pixels.
- DOPZEROT: Time (in MJD) when the Doppler shift was zero and increasing.

The "T" suffix at the end of each of these keywords indicates that they were derived from the on board telemetry, whereas the other keywords described below were computed on the ground from the orbital elements of *HST*. The two sets of keywords can differ by a small amount, but they should be nearly identical.

DOPPCORR assumes that the Doppler shifts vary sinusoidally with time according to the orbital movement of *HST*. The following keywords are used to perform the correction and are obtained from the science extension (SCI) header in the _rawtag or _rawaccum file:

- `EXPSTART` - start time of the exposure (MJD)
- `DOPPZERO` - the time (MJD) when the Doppler shift was zero and increasing (i.e., when *HST* was closest to the target)
- `DOPPMAGV` - The magnitude of the Doppler shift (km/s) to be applied to the data
- `DOPPMAG` - The number of pixels corresponding to the Doppler shift (used only for shifting the data quality flag arrays and flat fields)
- `ORBITPER` - the orbital period of *HST* in seconds

The data columns used in the correction are TIME (elapsed seconds since `EXPSTART`) and RAWX (position of photon along dispersion direction). The Doppler correction to be applied is then

SHIFT = -(DOPPMAGV/c*d)*lambda(RAWX)*sin(2*pi*t/ORBITPER)

where c is the speed of light (km/s), d is the dispersion of the grating used in the observation (Å/pixel), lambda(RAWX) is the wavelength at the RAWX position being corrected (obtained from the dispersion solution for that grating and aperture in the `DISPTAB` reference file) and t is defined as

t = (EXPSTART - DOPPZERO)*86400 + TIME

where the factor of 86400 converts from days to seconds.

The actual pixel shift applied at each position is obtained by rounding the SHIFT value to an integer.

3.4.9 DEADCORR: Nonlinearity Correction

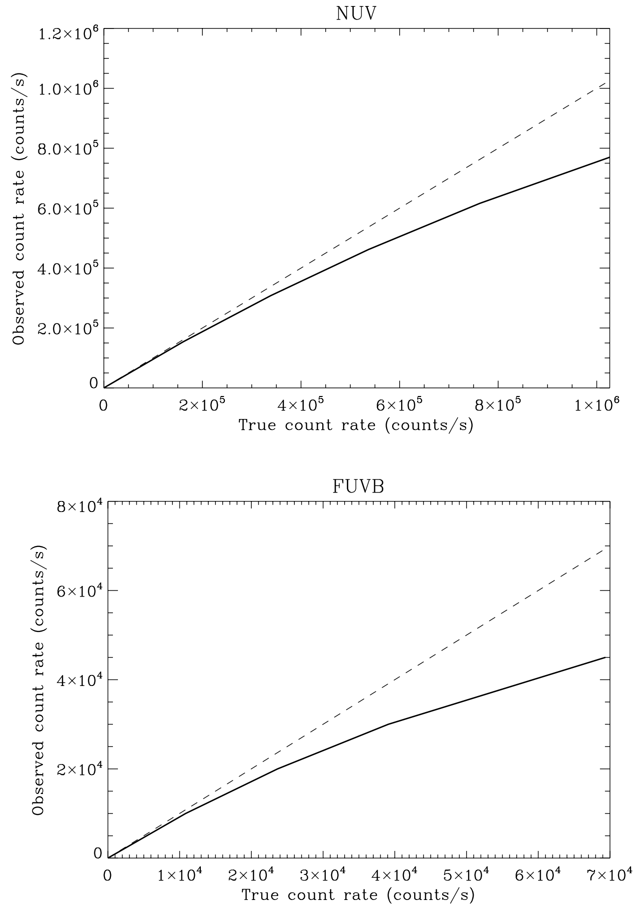
This module corrects for count rate dependent non-linearities in the COS detectors.

- Reference file: `DEADTAB`
- Input files: FUV/NUV `rawtag`, FUV/NUV `rawaccum, NUV images`
- Header keywords updated: none
- Updates `corrtag` file for TIME-TAG data and `flt` for ACCUM.

DEADCORR corrects for non-linear photon detection in the COS detector electronics. Both the FUV and NUV detector electronics have a small temporal overhead when counting events. This overhead becomes noticeable when the count rates become large.

The efficiency of the detector's photon counting is calculated as the ratio of the number of events counted and the number of events detected. This value is referred to as the deadtime. The deadtime for each detector is modeled and the reference file `DEADTAB` contains a lookup table of the correction for various count rates. Figure 3.7 shows how the measured count rates deviate from the actual count rates as a function of the actual count rate for the NUV detector, and segment A and B of the FUV detector.

Figure 3.7: FUV and NUV Deadtime



The solid curves are the observed count rates versus true count rates for the COS detectors and the dashed lines are for perfect detectors. TOP: The NUV MAMA. BOTTOM: Segment B of the FUV XDL detector (the curve for Segment A is nearly identical). Significant deviations from the true count rates occur at about 15,000 counts per second for the XDL detectors, and at roughly 10 times this rate for the MAMA.

For `TIME-TAG` data the deadtime correction is computed every 10 seconds. The observed count rate is the number of events within that time interval, and the deadtime factor is determined by interpolation within the values in `DEADTAB`. The values in the EPSILON column in the _corrtag file for events within that time interval will then be divided by the deadtime factor. For `ACCUM` data the observed average count rate is taken from a header keyword for the digital event counter. The deadtime factor is then found by interpolation within the `DEADTAB`, the same as for `TIME-TAG` data, and the science and error arrays will be divided by the deadtime factor.

3.4.10 FLATCORR: Flat Field Correction

This module corrects for pixel-to-pixel non-uniformities in the COS detectors.

- Reference file: FLATFILE
- Input files: FUV/NUV rawtag, FUV/NUV rawaccum, NUV images
- Header keywords updated: none
- Updates corrtag file for TIME-TAG data and flt for ACCUM.

The FLATCORR step corrects for high frequency, pixel-to-pixel sensitivity differences across the detector. It uses a flat field image located in the file specified by the FLATFILE header keyword. Figure 3.8 shows an NUV flat, and an example of an FUV flat is given in Section 3.4.12. For spectroscopic data, any wavelength dependence of the detector response or remaining low frequency spatial variations are removed by the flux calibration step (FLUXCORR, Section 3.4.15). Flat fielding is performed in geometrically corrected space, and because the pixel-to-pixel variations should be largely wavelength independent, only one reference image is used per detector or detector segment (NUV, FUVA, and FUVB). The flat field correction is applied differently for TIME-TAG and ACCUM mode data for both spectroscopic and imaging modes.

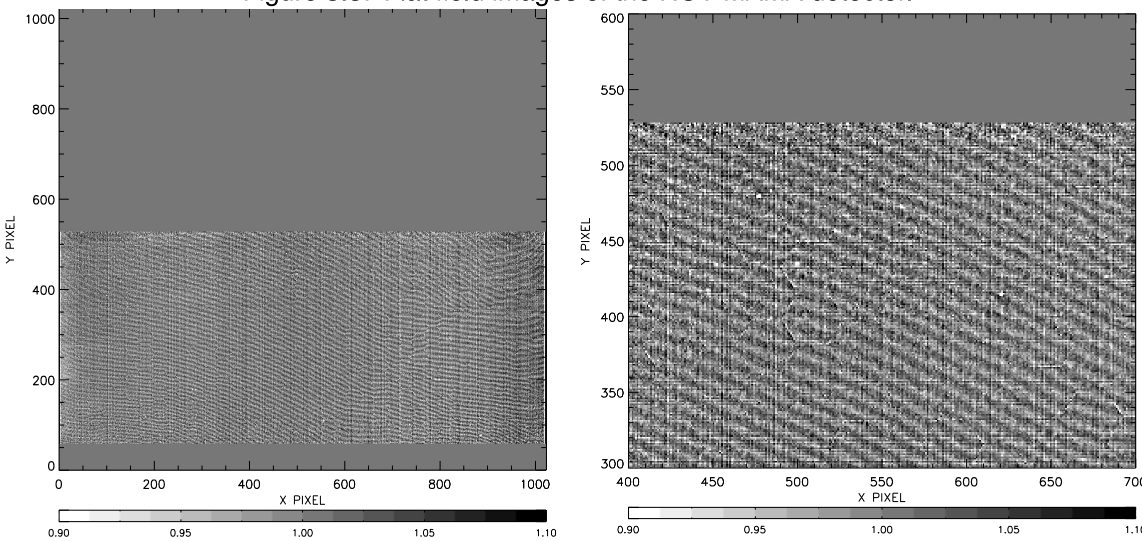
For spectroscopic TIME-TAG exposures, each photon in the events list is individually corrected. In the _corrtag file, the photon weight in the EPSILON column is divided by the flat field value at the event's detector location rounded to the nearest pixel (XCORR, YCORR for FUV; RAWX, RAWY for NUV).

For spectroscopic ACCUM mode data, photons are summed into an image on-board by the COS electronics. To compensate for the motion of *HST* during the observation, spectroscopic exposures are normally taken with Doppler compensation performed during the accumulation (science header keyword DOPPON=TRUE). During Doppler compensation, photon locations are shifted as the data are received, and the underlying flat field at each imaged pixel is an average of the original pixel position sensitivities. FLATCORR replicates this averaging for the flat field correction using the same control parameters as those on-board (DOPPMAGT, DOPZEROT, ORBTPERT) if DOPPCORR=PERFORM (Section 3.4.8). The convolved flat field image is divided into the rawaccum data, which is then divided by the exposure time to create the flt image file.

NUV images using the mirrors are not Doppler corrected. In this case, DOPPCORR=OMIT, and the input data are divided by the flat field without convolution.

For both the flt and counts files, error arrays are created based on counting statistics (Section 2.7), but they are not used in further processing.

Figure 3.8: Flat field images of the NUV MAMA detector.



The image at left shows the full detector, and the one on the right has been enlarged to illustrate structure in the flat field images. The hex structure associated with the microchannel plate is visible in both FUV and NUV flat fields.

3.4.11 WAVECORR: Wavecal Correction

For spectroscopic data, this module determines location of the wavelength calibration spectrum on the detector relative to a template, and then applies zero point shifts to align the wavecal and the template.

- Reference files: `LAMPTAB`, `WCPTAB`, `XTRACTAB`, `DISPTAB`
- Input files: FUV/NUV `rawtag`, FUV/NUV `rawaccum`
- Header keywords updated: `SHIFT1[A-C]`, `SHIFT2[A-C]`, `DUPLEXES-C]`, `SP_LOC_[A-C]`, `SP_SLOPE_[A-C]`, `LMP_ONi`, `LMPOFFi`, `LMPDURi`, `LMPMEDi`.
- Updates `corrtag` file for TIME-TAG data.

The wavecal step of **calcos** determines the shift of the 2-D image on the detector along each axis resulting from thermal motions and drifts within an OSM encoder position. This step applies only to spectroscopic data, `TIME-TAG` and `ACCUM`, for both the FUV and NUV detectors. The shifts are determined from one or more contemporaneous wavelength calibration observations of a spectral line lamp (wavecal) which must be obtained without moving the OSM between the science and wavecal exposures.

There are two types of wavecals. For `ACCUM` data the spectrum of the calibration lamp is contained in an exposure that is separate from that of the science (AUTO or GO wavecals). For `TIME-TAG` data the wavecals can also be separate exposures, but the default when observing with the PSA aperture is `TAGFLASH` mode. In the `TAGFLASH` mode the line lamp is turned on and off (flashed) one or more times during a single science exposure, producing a wavecal spectrum that is offset in the cross-dispersion direction from the science spectrum (See Figure 1.7, and Figure 1.9). The algorithm used to determine the shifts is the same in either case, but the way that the shift is determined at the time of the observation differs. Thus, we begin by describing how the offsets are found.

Determining the offsets:

For each wavecal the location of the spectrum in the cross-dispersion direction is determined by collapsing the spectrum along the dispersion direction using the extraction slope defined in the XTRACTAB table (SLOPE). The location of the brightest pixel, after boxcar smoothing, is taken as the spectrum location and that location is compared to the nominal position defined in the XTRACTAB table. The offsets from nominal positions for segments A and B (FUV) or stripes A, B, and C (NUV) are recorded in the keywords SHIFT2[A-C]. The two FUV segments are processed independently. Cross-dispersion shifts are determined for each NUV stripe and then the average is computed. This average value is applied to the three stripes and recorded in the SHIFT2[A-C] (thus, the three keywords have the same value). The sign of the SHIFT2 keyword is positive if the spectrum was found at a larger pixel number than the nominal location.

To determine the offsets in the dispersion direction, the wavecal spectrum is collapsed along the cross-dispersion direction and compared to the template wavecal (LAMPTAB) taken with the same grating and central wavelength. For the NUV the wavecal spectra for all the three stripes are added together to improve the signal-to-noise ratio and to increase the number of useful features for cross-correlating with the summed template wavecal spectra. The maximum amplitude for correlating the wavecal and template wavecal spectra is defined by the value of XC_RANGE in the WCPTAB table. **Calcos** takes into account the FPPOS of the wavecal spectrum by shifting it by FPOFFSET times STEPSIZE before cross-correlating it with the template wavecal. STEPSIZE, from the WCPTAB table, is the number of pixels corresponding to one OSM motion for a particular grating.

Applying the offsets:

The way the offsets are applied to the spectral data depends on whether the data were obtained with AUTO or GO wavecals or with TAGFLASH wavecals. For AUTO or GO wavecals, the wavecals are obtained at different times than the spectral data and temporal interpolation is done to determine the appropriate shifts. For TAGFLASH data, the wavecals are interspersed with the spectral data, allowing more precise and, consequently, more intricate corrections to be made.

AUTO or GO wavecals:

For ACCUM science exposures which are bracketed by AUTO or GO wavecal observations, the SHIFT1[A-C] and SHIFT2[A-C] values from the wavecals are interpolated (linearly) to the middle time of the science observation, and the interpolated values are assigned to the SHIFT1[A-C] and SHIFT2[A-C] keywords in the science data header. If there is just one wavecal observation in a dataset, or if there are more than one but they don't bracket the science observation, the SHIFT1[A-C] and SHIFT2[A-C] keywords are just copied from the nearest wavecal to the science data header.

For non-TAGFLASH TIME-TAG science exposures bracketed by AUTO or GO wavecal observations, the SHIFT1[A-C] and SHIFT2[A-C] values from the wavecals are interpolated (linearly) so that each event in the science exposure is shifted according to its arrival time. The SHIFT1[A-C] and SHIFT2[A-C] keywords recorded in the science data header are in this case the averages of the values applied. As in the ACCUM case, if there is only one wavecal observation in a dataset, or if there are more than one but they do not bracket the science observation, the SHIFT1[A-C] and SHIFT2[A-C] keywords are just copied from the nearest wavecal to the science data header.

TAGFLASH DATA:

A TAGFLASH wavecal is a lamp exposure that is taken concurrently with a TIME-TAG science exposure, and the photon events for both the wavecal lamp and the science target are mixed together in the same events table. In many respects, TAGFLASH wavecals are handled differently from conventional wavecals.

The nominal start and stop times for each lamp flash are read from keywords in the corrtag table. The actual start and stop times can differ from the nominal times, so **calcos** determines the actual times (restricted to being within the nominal start-to-stop intervals) by examining the number of photon events within each one-second interval in the wavecal region defined in the XTRACTAB table. A histogram of the count rate is constructed. The histogram is expected to have one peak near zero, corresponding to dark counts, and another at high count rate, due to the lamp illumination. The average count rate when the lamp is on is taken to be the count rate for the second peak of the histogram. The lamp turn-on and turn-off times are taken to be the times when the count rate rises above or sinks below half the lamp-on count rate.

Calcos uses the time of the median photon event within a lamp turn-on and turn-off interval to determine the moment of the flash. The keywords LMP_ONi and LMPOFFi (i is the flash number) are updated with the actual turn-on and turn-off times, in seconds, since the beginning of the science exposure. The keywords LMPDURi and LMPMEDi are updated with the actual duration and median time of the flash.

The location of the wavecal spectrum in the cross-dispersion direction is determined by collapsing it along the dispersion direction and by comparing this position with the nominal one in the XTRACTAB table, similarly to what is done for AUTO or GO wavecals. The wavecal spectrum is then collapsed along the cross-dispersion direction to produce a 1-D spectrum that is cross-correlated with the template spectrum so that the SHIFT1[A-C] offsets can be determined. Typically there will be more than one wavecal flash during a science exposure; in this case the shifts will be linearly interpolated between flashes. The SHIFT1[A-C] and SHIFT2[A-C] values that are recorded in the science data header are the average of the values found from the different flashes. During this step **calcos** produces a calibration lamp flash file with the extension lampflash.

Additional Functions: WAVECORR also corrects the _flt and _counts files which result from both ACCUM and TIME-TAG science data for the offsets in the dispersion and cross-dispersion directions. However, since these images are in pixel space they can only be corrected by an integer number of pixels. The _flt and _counts images are corrected by the nearest integer to SHIFT1[A-C] and SHIFT2[A-C]. DPIXEL1[A-C] is the average of the difference between XFULL and the nearest integer to XFULL, where XFULL is the column by that name in the _corrtag table. This is the average binning error in the dispersion direction when the _flt and _counts images are created from the _corrtag table. DPIXEL1[A-C] is zero for ACCUM data. This shift is used when computing wavelengths during the X1DCORR step.

For both ACCUM and TIME-TAG data **calcos** also updates the SP_SLOPE_[A-C] in the science data header with the SLOPE values from the XTRACTAB reference file for the science spectrum [A-C]. The keywords are SP_SLP_A, SP_SLP_B, SP_SLP_C.

Currently, **calcos** finds the offset in the dispersion direction for the NUV detector by coadding the wavecal spectra of all three stripes. This assumes that the three stripes move in lock step when the gratings are moved. Once COS has been installed on-orbit, these motions will be examined in more detail during some of the SMOV programs. **Calcos** may be modified, depending upon the results of these programs.

3.4.12 DQICORR: Initialize Data Quality File

This module identifies pixels which are suspect in some respect and creates the counts and flt images for science data.

- Reference file: BPIXTAB
- Input files: FUV/NUV rawtag, FUV/NUV rawaccum, NUV images
- Header keywords updated: none
- Creates the counts and flt files for all data and updates the corrtag file for TIME-TAG data.

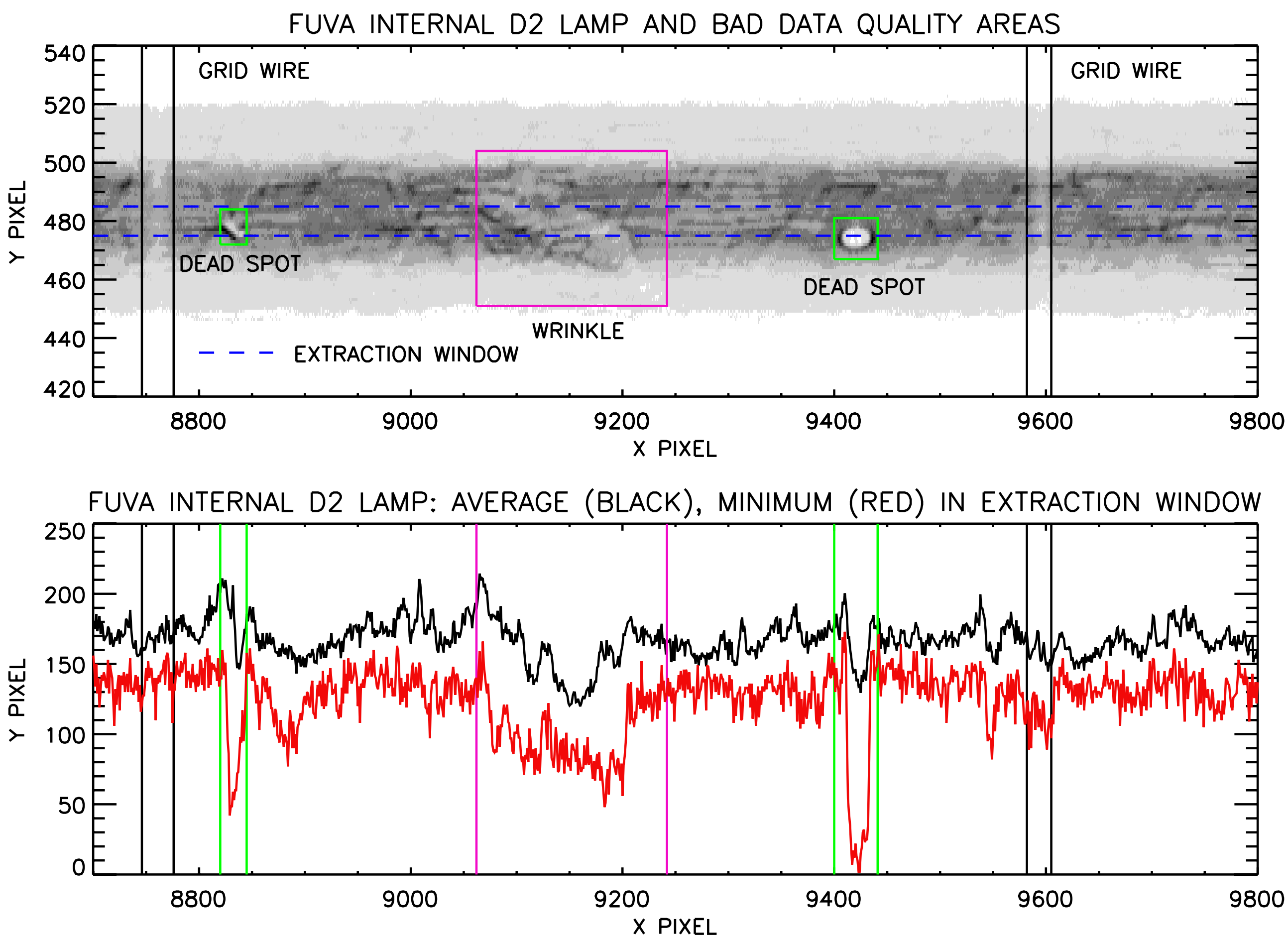
The DQICORR step maps data quality flags for suspect detector pixels listed in the BPIXTAB table to the science data. The COS data quality flags are discussed in Section 2.7.2 and are listed in Table 2.16. Figure 3.9 shows examples of the types of regions isolated by the DQ flags and the effect that they can have on an extracted spectrum. DQICORR proceeds differently for TIME-TAG and ACCUM mode exposures, but the flags in the flt and counts images are created similarly in preparation for spectral extraction. Consequently, we describe each mode separately.

TIME-TAG: For TIME-TAG exposures, DQICORR compares the XCORR, YCORR (FUV) or XRAW, YRAW (NUV) pixel values in the rawtag file to the rectangular regions containing detector defects in the BPIXTAB table. A flag for each photon event is created by performing a bitwise OR combination for all flags appropriate for the pixel location, as well as using the bad time flags set by BRSTCORR (Section 3.4.2) and BADTCORR (Section 3.4.3). The final flag is located in the DQ column of the corrtag file. When the flt and counts images are generated from the corrtag file, photons which arrived during bad times or bursts are omitted from the image and ERR array. For FUV data, events whose PHAs were flagged as out of bounds are omitted as well. However, data with spatial DQ flags are retained at this stage. Several of these blemishes are illustrated in Figure 3.9

The third FITS extension of the flt and counts files is an array of data quality values generated directly from the BPIXTAB table. If DOPPCORR=PERFORM, the BPIXTAB locations are Doppler-smeared and the flags from all neighboring pixels that contribute to the flt and counts image pixels are combined.

ACCUM: For ACCUM exposures, the rawaccum image file will already have a third FITS extension of data quality values if any pixel had been flagged when constructing the raw image. The extension will be a null image if all initial data quality flags are zero. This is usually the case for NUV data, but not FUV. For FUV ACCUM exposures, photons are collected for only part of the detector segment and an initial data quality array is created to mark the pixels outside those subimage boundaries (flag=128, out-of-bounds). When **calcos** creates the `flt` and `counts` images, it first converts the `rawaccum` image to a pseudo-time-tag array. It concatenates the raw data quality values within the `BPIXTAB` locations as for `TIME-TAG` mode, and ignores the regions flagged in Table 3.1 when generating the `flt` and `counts` files. As for `TIME-TAG` processing, the third extension of the `flt` and `counts` files contains a Doppler-smeared version of the `BPIXTAB` table, but it also includes the initial flag assignments in the `rawaccum` DQ extension.

Figure 3.9: The FUV flat field.



The FUV flat field is used to illustrate blemishes and regions of lower sensitivity for the detector. These regions are flagged in `BPIXTAB` according to the feature type.

3.4.13 X1DCORR: Locate and Extract 1-D Spectrum

This module extracts a one dimensional spectrum from the image of the spectrum on the detector.

- Reference files: `XTRACTAB`, `DISPTAB`
- Input files: `flt`, `counts`
- Header keywords updated: none
- Creates `x1d` files

A 1-D spectrum and its error array are extracted from the `flt` and `counts` images by summing the counts in the cross-dispersion direction within a band centered on the spectrum. The data are not resampled in the dispersion direction. Wavelengths are assigned by evaluating a polynomial function (dispersion relation) of the pixel coordinates. Background is subtracted (see BACKCORR) to get the net count rate, and the absolute flux is computed from the net count rate (see FLUXCORR).

This section provides the details of the spectral extraction process and the construction of the arrays which populate the `x1d` files. Table 3.1 lists these arrays along with others that are used to calculate them. The following discusses how each array is calculated in detail.

Table 3.1: Variables used in 1-D spectral extraction

Variable	Description
e[i]	Effective count rate, extracted from `_flt`
GC[i]	Gross count rate, extracted from `_counts`
BK[i]	Smoothed background count rate, extracted from `_counts`
eps[i]	(Effective count rate) / (gross count rate)
N[i]	Net count rate = eps[i] (GC[i] - BK[i])
ERR[i]	Error estimate for net count rate
FLUX[i]	Calibrated flux
WAVE[i]	Wavelength scale in Angstroms.
DQ_WGT[i]	Weights array
DQ	Bitwise OR of the DQ in the extraction region
snr_ff	The value of keyword SNR_FF from the flat field reference image
extr_height	The number of pixels in the cross-dispersion direction that are added together for each pixel of the spectrum
bkg_extr_heigh	The number of pixels in the cross-dispersion direction in each of the two background regions
bkg_smooth	The number of pixels in the dispersion direction for boxcar smoothing the background data
bkg_norm	Float (extr_height) / (2.0 float (bkg_extr_height))

Table 3.1 defines the variables used in the 1-D spectral extraction. Variables beginning with a capital letter are saved in the output `x1d` file. An "“[i]” represents array element i in the dispersion direction.

GROSS: The GROSS count rate spectrum is obtained from the counts file. The extraction is performed over a parallelogram, whose shorter edges are parallel to the image boundaries and longer (slightly sloping) edges are parallel to the spectrum (see, Figure 3.10). The columns within the parallelogram are summed in the cross-dispersion direction to obtain each element of the GROSS spectrum. Note that in some cases the spectral lines are obviously tilted, i.e., not aligned with the columns, but the sum ignores any such tilt. The location of the parallelogram in the cross-dispersion direction is taken from column B_SPEC in the XTRACTAB; if a conventional wavecal was used, the offset (keyword SHIFT2A, SHIFT2B, or SHIFT2C) are added to the nominal extraction location.

Extraction routines of this sort are referred to as a "boxcar" extractions because they do not weight the elements of the spectrum in the cross-dispersion direction.

BACKGROUND: Two background regions are sampled on the counts array to obtain a mean background count rate spectrum. For the FUV data, these are on above and below the spectrum, and for the NUV data they are above stripe C and below stripe A (see, Figure 3.10 and Figure 3.11). The background regions are extracted in the same way as the spectrum. The values in the two background regions are added, boxcar smoothed in the dispersion direction, and scaled by the sizes of their extraction regions before being subtracted from the science spectrum. Details of the background extractions are given in Section 3.4.14.

NET: The NET spectrum is the difference between the GROSS spectrum and a properly scaled BACKROUND spectrum multiplied by an array which accounts for flat field and dead time effects. This array is eps[i] = e[i]/GC[i], where GC[i] is an element in the GROSS spectrum described above and e[i] is an element in an array extracted from the _flt file in exactly the same way as the GROSS spectrum is extracted from the counts file. Consequently, this factor corrects the NET spectrum for flat field and dead time effects.

DQ: The DQ array in the x1d file is the bitwise OR of the members of the DQ array, contained in the third FITS extension of the counts file, for all of the points in the counts image that contribute to an element of the GROSS spectrum. Consequently, if anything is flagged within the extraction region, it is reflected in the x1d DQ array.

DQ_WGT: The DQ_WGT array has one point for each extracted point in the spectrum. It is 0 or 1 depending on whether the DQ for a given point is allowed according to the header keyword, SDQFLAGS. The default value for SDQFLAGS is 184. Thus, it sets DQ_WGTS to 0 for events that are near the edge of the detector, dead spots, hot spots or outside of the subarray (see, Table 2.16). Otherwise, DQ_WGTS = 1. The DQ_WGTS array is used to construct the x1dsum file discussed in Section 3.4.18.

ERROR: The ERROR array is calculated from a combination of variables needed to track the detected counts and the different scale factors which multiply them. The raw ERROR array is calculated as follows

The error array involves elements from both the _flt and the _counts files. It is calculated as follows:

$$term1[i] = (N[i]*exptime/(extr\_height*snr\_ff))^2$$

$$term2[i] = eps[i]^2\ exptime*(GC[i] + BK[i]*(bkg\_norm/bkg\_smooth))$$

$$ERR[i] = sqrt\{(term1[i] + term2[i])\}/exptime$$

The ERROR array contained in the _x1d file differs from this one only in the sense that it has had the absolute flux calibration applied (see Section 3.4.15).

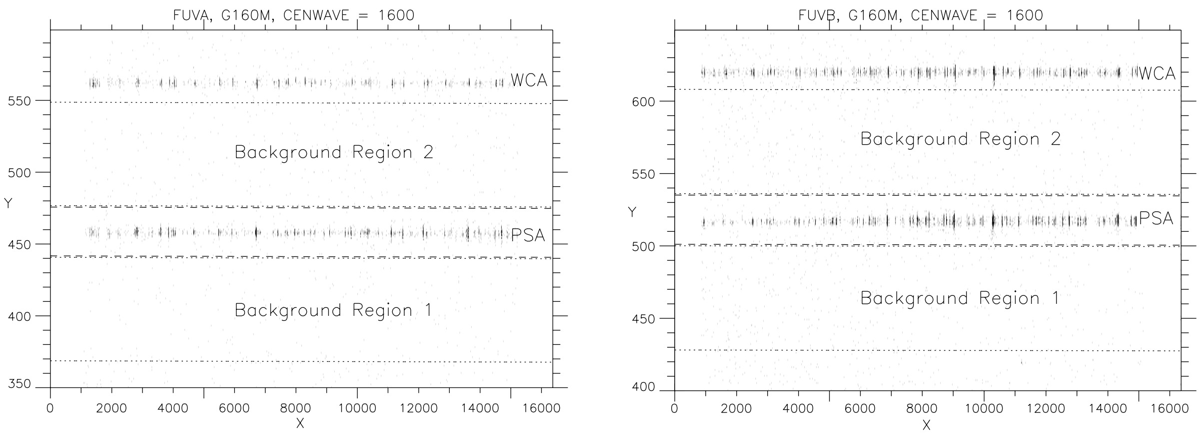
WAVELENGTH: As part of the spectral extraction, **calcos** assigns wavelengths to pixels in the extracted spectra using dispersion coefficients from the reference table `DISPTAB`. For each grating, central wavelength, and aperture, the `DISPTAB` table contains the dispersion solution with respect to the template spectral lamp table that was used in the WAVECORR step. The dispersion solution has the following form:

$$lambda = A_0 + A_1\ s + A_2\ s^2 + A_3\ s^3$$

where lambda is the wavelength in Angstroms, s is the pixel coordinate in the dispersion direction, and A_i are the dispersion coefficients. Offsets due to an OSM shift determined from WAVECORR (Section 3.4.11) are corrected for by applying a linear offset after all other corrections have been made. For all modes, small, residual offsets occur because of thermal drifts and drifts in the positioning of the OSM. The precision of the OSM positioning and the impact of small offsets in the spectra are discussed further in Chapter 4:COS Error Sources

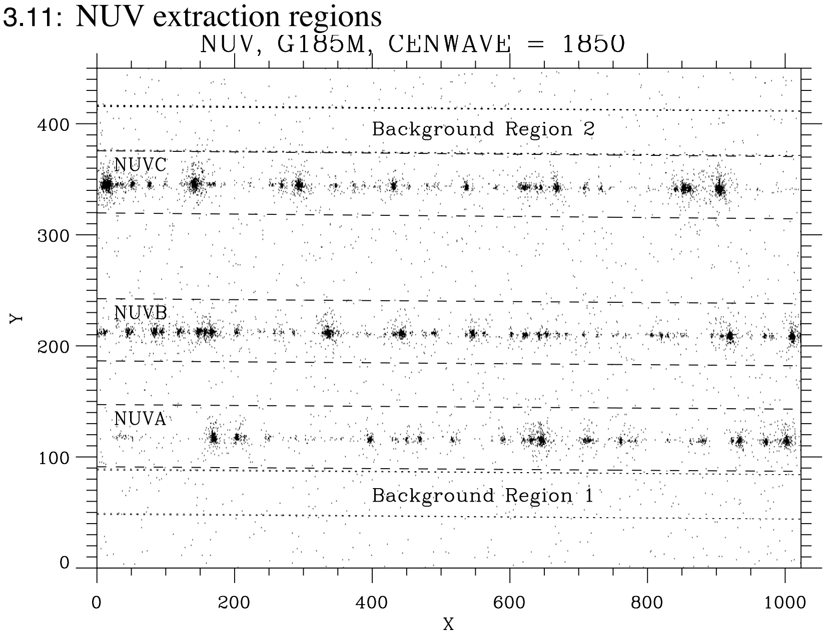
FLUX: The FLUX array in the `x1d` file is the NET spectrum corrected by the sensitivity curve. The details of this process are discussed in Section 3.4.15.

Figure 3.10: FUV extraction regions



Portions of the undistorted images of the FUV detector segments (compare to Figure 2.2) illustrating the regions used to extract the spectrum and define the background. The dashed lines indicate the spectral extraction window, and the dotted lines define the background extraction region.

Figure 3.11: NUV extraction regions



Portion of the NUV detector showing spectral extraction regions used for the three non-contiguous PSA spectra (dashed lines) and for the background (dotted lines).

3.4.14 BACKCORR: 1D Spectral Background Subtraction

This module determines the background contribution to the extracted spectrum and subtracts it.

- Reference file: XTRACTAB

- Input files: `flt`, `counts`
- Header keywords updated: none
- Updates the `x1d` file

The BACKCORR module computes the number of counts in the background regions, scales them by the ratio of sizes of the spectral extraction region to background regions, and then subtracts that value from the extracted spectrum at each wavelength. There are two background regions defined. For FUV data, there is one above and one below the object spectrum (see Figure 3.10). For the NUV spectra, the two regions are above and below the three stripes (see Figure 3.11). Each background region is a parallelogram with the same slope used to define the object extraction region, but with different y-intercepts. The parameters of the background extraction region in the FUV are

- HEIGHT: The full height (along the cross-dispersion) of the object extraction region in pixels
- BHEIGHT: The full height (along the cross-dispersion) of the background region in pixels
- BWIDTH: The full width (along the dispersion) of the box-car average performed on the background.
- B_BKG1: y-intercept of upper background region
- B_BKG2: y-intercept of lower background region

The centers of background regions 1 and 2 in the cross-dispersion (Y) direction follow a linear function in the dispersion (X) direction according to the function

$$Y = mX + b$$

where m is the slope of the background (keyword SLOPE), and b is the Y-intercept of the background region (B_BKG1 or B_BKG2). At the i-th pixel along the dispersion direction (X) the background is computed by first summing all of the counts in pixels in the cross-dispersion within +/- (BHEIGHT/2) of the central Y pixel of the background box. All DQ spatially related DQ flags are ignored (note that counts which occur during bad time intervals or which have out of bounds PHAs, never make it to the `counts` file). Once this is done for all X pixels, the result is averaged over +/- BWIDTH/2 pixels along the dispersion direction. This gives a local average background (with known anomalous pixels such as dead spots or strong hot spots excluded). The background regions below and above the object spectrum are both computed in this way, and then they are summed and divided by 2 to yield an average background rate. This average is then scaled to the number of pixels in the object extraction box by multiplying it by the factor (HEIGHT / (BHEIGHT * BWDITH)). The result is the background count rate BK[i] in Table 3.1, which is written to the

BACKGROUND column in the x1d file. The background-subtracted count rate (corrected for flat field and deadtime) is written to the NET column in the x1d table.

3.4.15 FLUXCORR/TDSCORR: Conversion to Flux

This module converts extracted spectrum into physical units, and allows for time dependencies in the conversion.

- Reference files: PHOTTAB, TDSTAB
- Input file: x1d
- Header keywords updated: none

If FLUXCORR=PERFORM, FLUXCORR divides the NET and ERROR columns by the appropriate sensitivity curve read from the PHOTTAB reference table, which converts them to flux units (erg cm^{-2} s^{-1} $Å^{-1}$). The NET divided by the sensitivity is written to the FLUX column of the x1d file, while the ERROR column is modified in-place.

The sensitivity curves read from the reference files are interpolated to the observed wavelengths before division. The flux calibration is only appropriate for point sources and has an option to accommodate time-dependent sensitivity corrections.

If TDSCORR=PERFORM, then the module TDSCORR will correct for temporal changes in the instrumental sensitivity relative to the reference time given by REF_TIME keyword in the FITS header of TDSTAB. TDSTAB provides the slopes and intercepts needed to construct a *relative* sensitivity curve. The curve for the epoch of the observation is determined by piecewise linear interpolation in time using the slopes and intercepts closest to the time of the observation. The sensitivity may be discontinuous at an endpoint of a time interval. Different piecewise linear functions may be specified for each of the wavelengths listed in the Table. This process results in a relative sensitivity at the epoch of the observation, at the wavelengths given in the reference table. Interpolation between these wavelengths to the observed wavelength array is also accomplished by piecewise linear interpolation.

Since COS is a new instrument, the contents of the reference file for this aspect of the flux calibration results in no correction being applied. However, there will be continuing COS calibration programs that will monitor the time dependence of its response functions and populate TDSTAB as needed.

3.4.16 HELCORR: Correction to Heliocentric Reference Frame

This module converts the observed wavelengths to Heliocentric wavelengths.

- Reference file: none
- Input files: `rawtag`, `x1d`
- Header keywords updated: `V_HELIO`

In addition to the Doppler smearing from *HST* orbital motion, the photons acquired during an observation are also Doppler shifted due to the orbital motion of the Earth around the Sun (V ~ 29.8 km/s). The sign and magnitude of the Doppler shift depend on the time of the observation as well as the coordinates of the target (i.e., the position of the target relative to the orbital plane of the Earth).

The HELCORR module in **calcos** transforms wavelengths of a spectrum to the heliocentric reference frame. It is applied to the extracted 1D spectrum during the operation of X1DCORR, by utilizing the keyword `V_HELIO`, which gives the heliocentric velocity (km/s) for the observed target during the observation. This is the component of the Earth's velocity away from the target. It is computed by **calcos** and written to the science data header of the output corrtag file before spectral extraction is performed.

The shift at each wavelength is

$$\mathrm{Lambda}_{\mathrm{Helio}} = \mathrm{lambda}_0\ [1 - (V_{\mathrm{Helio}}/c)\ \mathrm{lambda}_0]$$

where $\mathrm{Lambda}_{\mathrm{Helio}}$ is the corrected wavelength (Å), c is the speed of light in km/s and lambda_0 is the wavelength before the Heliocentric correction.

The velocity vector of the Earth is computed in the J2000 equatorial coordinate system, using derivatives of low precision formulae for the Sun's coordinates in the Astronomical Almanac. The algorithm does not include Earth-Moon motion, Sun-barycenter motion, or the Earth-Sun light-time correction.

3.4.17 STATFLAG: Report Simple Statistics

This module computes some statistical measures that provide general information about COS science observations.

- Reference file: `XTRACTAB`, `BRFTAB`
- Input files: `flt`, `counts`, `x1d`, `lamptab`
- Header keywords updated: `NGOODPIX`, `GOODMEAN`, `GOODMAX`

STATFLAG enables the reporting of statistics for COS observations. STATFLAG is enabled by default for all science observations and operates on `x1d`, `counts`, and `flt` data products. STATFLAG is intended to provide a very basic statistical characterization of the events and locations on the detectors that are known to be good. By default, all of the following data quality (DQ) flags must be zero for an event or pixel to be considered good: DQ_NEAR_EDGE, DQ_DEAD, DQ_HOT and DQ_OUT_OF_BOUNDS (see Table 3.10 for definitions). This behavior is modified by the SDQFLAGS header keyword (Serious Data Quality FLAGS), which indicates which DQ values should be excluded from the statistical calculations. Numerically, the default value of SDQFLAGS is 184.

To select an alternative definition of SDQFLAGS, the user should modify the _rawtag or _rawaccum header and reprocess the file with calcos.

STATFLAG reports the following statistics:

- NGOODPIX: The number of good pixels or collapsed spectral columns For the `counts` and `flt` images, this is the number of pixels in the spectral extraction or imaging region. For the `x1d` file, each 'Y' column in the spectral extraction region of the `flt` file is combined to produce the one-dimensional spectrum. The DQ of each column is the logical OR of the DQ flags of the individual pixels. Only collapsed spectral columns that pass the DQ conditions indicated by SDQFLAGS are considered good for purposes of calculating statistics.
- GOODMEAN: The mean of the good bins in counts per bin. For the `counts` and `flt` files, a bin is an individual pixel, while for `x1d` files, a bin is a collapsed spectral column.
- GOODMAX: The maximum of the good bins in the same units as the mean.

3.4.18 Finalization (making the `sum` files)

The final data products for spectroscopic and NUV imaging are different and are discussed separately.

SPECTROSCOPY: Once the processing is complete, an `x1d` file is written for each spectroscopic exposure. This file includes spectra from both segments A and B for the FUV detector, and from all three stripes for the NUV detector (see, Section 2.4.3). In addition, one or more `x1dsum` files are created. This is done even if only one spectrum was obtained.

The `x1dsum` files differ from the `x1d` files in one important respect. When an `x1dsum` file is created the DQ_WGT array (Section 3.4.18) is used to determine whether a point is good or bad. When only a single file contributes to the `x1dsum` file, if DQ_WGT = 0 for a pixel, then the counts, net and flux arrays for that point are set to zero. If the `x1dsum` or `x1dsum<n>` (for FP-POS observations) includes several `x1d` files, then, for each point in the spectrum, only those files with a DQ_WGT = 1 at that point are included (weighted by the individual exposure times), and the DQ_WGT array in the `x1dsum` file is updated to reflect the number of individual spectra which contributed to the point. If the updated value of DQ_WGT for a particular point is 0, then the value of the spectrum at that point is set to 0 in the `x1dsum` file.

NUV IMAGING: The end product for NUV imaging is an `fltsum` file. Like the `x1dsum` files, and `fltsum` file is created even if only one exposure is processed. However, since no shifting is performed for imaging observations (see, Figure 3.5), the `fltsum` file is a simple exposure time weighted mean of the individual `flt` files (and it is identical to the `flt` file if only one exposure contributed to it). The DQ flags image of the `fltsum` file and, for that matter, all of the individual `flt` images, are identical. This is because the only data which make it into an `flt` or `counts` image are free of temporal or event flags (see, Section 2.7). Consequently, in the absence of shifting, all of their spatial flags should be identical.

3.5 Descriptions of Imaging Calibration Steps

The processing of image data is depicted in Figure 3.5. It is an abbreviated version of the pipeline that only involves those steps which identify bad data and linearize the initial counts. No absolute flux calibration is performed and no background is identified or subtracted.

The final data products for image data are the `flt` and `fltsum` files described in Section 2.4.2.

3.6 Customizing COS Data Calibration

Sometimes the pipeline calibration performed shortly after the data were obtained from the telescope is not the best possible calibration for your science program. There are a number of reasons why it may be desirable to recalibrate your data. The most likely reasons include:

- More appropriate reference files have become available since the data were originally obtained.
- Some steps need to be repeated with different input parameters. For example, you may wish to re-perform the 1-D spectral extraction with a smaller height in order to minimize the background, or you may wish to cut a `TIME-TAG` exposure into sub-exposures, in order to study time variability.

The simplest way to recalibrate your data with the most appropriate reference files is to request the data from the archive again. However, to tailor the calibration to your individual preferences, it may be beneficial to run **calcos** yourself on your local machine, or to use tasks that improve the reference files or allow customized treatment of the data. **Calcos** can be imported and executed when running **PyRAF** or **Python**.

Be sure you are using the latest version of the* calcos *and STSDAS software. See www.stsci.edu/resources/software_hardware/stsdas for information about the latest release.

Calcos contains provisions for recalibrating raw data. Users can specify the pipeline processing steps to be performed and select the associated reference files. However, **calcos** was not designed to run its various modules independently, i.e. it is not re-entrant. The pipeline flow is modified by setting calibration switches or reference file names and then rerunning the entire pipeline. The calibration switches in the headers of the calibrated data files will reflect the operations performed on the calibrated data and the reference files used.

3.6.1 Mechanics of Tailored Recalibration

If you chose to recalibrate your COS data on your local machine, there is a certain amount of set up required for **calcos** to run properly. The operations mentioned in the checklist below will be described in detail in the following subsections:

- Set up a directory with the required reference files.
- Determine which reference files are needed and retrieve them from the Archive.
- Set the environment variable `lref` to point to your reference file directory. *Note:* you must do this before starting a **PyRAF** session!
- Update the input data file headers (including reference file names). In **IRAF**, this would be done using **thedit**. Update the input association files.
- Set the calibration switches in the headers of the raw data files to perform the needed steps.The default calibration switches are listed in table 2.13.
- Edit the association file.
- Run **calcos**.

Set up the Directory Structure for Running calcos

Before running **calcos**, you will need to define an environment variable to indicate the location of the directory containing the needed calibration reference files. The names of the calibration files are preceded with the logical path name "`lref$`" in the COS science headers. Ordinarily you would define this directory in an **PyRAF** session to be, for example, "`/data/vega3/cos/lref/`" using the **set** command:

```
cl> set lref "/data/vega3/cos/lref/" # Won't work!
```

Note the trailing slash (/). However, **calcos** and all of its modules are actually foreign tasks and as such do not access **PyRAF** environment variables. Therefore, *before invoking the* **cl**, you will need to define an environment variable from the host command line (see below) that is appropriate to your host machine. For Unix systems, the appropriate command for the example above is:

```
% setenv lref /data/vega3/cos/cal_ref/
```

Note that an alternative to using the `lref$` variable is specifying the full pathnames to the reference files in the science headers.

When running* calcos *or any of its modules, you must define environment variables (such as* `lref$`*) before starting the cl. It is not possible to define them within IRAF using the set command, nor is it possible to define them with an escape to the host level, such as:

```
!setenv lref /data/vega3/cos/lref/
```

Retrieve Reference Files

To recalibrate your data, you will need to retrieve the reference files used by the different calibration steps to be performed. The names of the reference files to be used during calibration must be specified in the primary header of the input files, under the section "CALIBRATION REFERENCE FILES." Note that the data headers will be populated already with the names of the reference files used during pipeline calibration at STScI.

Chapter 1 of Part 1 of this handbook describes how to retrieve data and reference files via the World Wide Web. To retrieve the best reference files via MAST, check "Best Reference Files" in the "Reference Files" section of the Retrieval Options form.

The COS reference files are all in FITS format, and can be in either IMAGE or BINTABLE extensions. The names of these files along with their corresponding primary header keywords, extensions, and format (image or table), are listed in Chapter 2. The rootname of a reference file is based on the time that the file was delivered to the Calibration Data Base System (CDBS).

Edit the Calibration Header Keywords

To edit file headers in preparation for recalibration, use the **STSDAS** task **thedit**. The **thedit** task takes several input parameters: the name(s) of the raw data files to be edited, the header field to be edited, and the new value of the header field. It can be used to change the values of any calibration switches, reference files or tables to the values you wish to use for recalibrating your data. To edit the calibration keyword values:

1. Run the **thedit** task, specifying a list of files in which you want to change calibration keyword values. You may specify more than one file (using wildcards, for example) to be updated. For example, you could change the flat reference file to be used for all COS raw science files in the current directory using the command:

```
ct> thedit *raw*.fits[0] flatfile 'lref$n9n201821_flat.fits' up+
```

Similarly, to turn off the FUV burst calibration switch use the command:

```
ct> thedit *raw*.fits[0] brstcorr 'OMIT' up+
```

If you are changing keywords that reside in the FITS primary header unit with* hedit *or* thedit*, be sure to explicitly specify the primary header by appending "[0]" to the FITS file name.

The task* chcalpar *will be available for use with COS data with STSDAS release 3.10. This task can be used to efficiently edit calibration header keywords.

Edit the Input Association File

Users may find it necessary to edit the input association file for **calcos**. Reasons for editing an association file might include the use of a different wavecal or to remove a compromised exposure from an association. One way to update an association file is to use the **STSDAS** task, **tedit**. For example, use the **PyRAF** task **tprintf** to first look at the contents of association table, `l9v221010_asn.fits.`

```
-->tprint l9v221010_asn.fits prparam=no prdata=yes
# MEMNAME        MEMTYPE        MEMPRSNT
  L9V221EUQ      EXP-FP         yes
  L9V221EWQ      EXP-AWAVE      yes
  L9V221EYQ      EXP-FP         yes
  L9V221F0Q      EXP-AWAVE      yes
  L9V221F2Q      EXP-FP         yes
  L9V221F4Q      EXP-AWAVE      yes
  L9V221F6Q      EXP-FP         yes
  L9V221F8Q      EXP-AWAVE      yes
  L9V221010      PROD-FP        yes
```

To quickly see basic exposure information for each exposure listed in the association use the **thselect** command:

```
--> thselect l9v221*raw*fits[0]                                    \
filename,detector,aperture,opt_elem,cenwave,exptype,obsmode,fppos 'yes'


l9v221euq_rawtag_a.fits   FUV  PSA   G130M  1309  EXTERNAL/SCI   TIME-TAG  1
l9v221euq_rawtag_b.fits   FUV  PSA   G130M  1309  EXTERNAL/SCI   TIME-TAG  1
l9v221ewq_rawtag_a.fits   FUV  WCA   G130M  1309  WAVECAL        TIME-TAG  1
l9v221ewq_rawtag_b.fits   FUV  WCA   G130M  1309  WAVECAL        TIME-TAG  1
l9v221eyq_rawtag_a.fits   FUV  PSA   G130M  1309  EXTERNAL/SCI   TIME-TAG  2
l9v221eyq_rawtag_b.fits   FUV  PSA   G130M  1309  EXTERNAL/SCI   TIME-TAG  2
l9v221f0q_rawtag_a.fits   FUV  WCA   G130M  1309  WAVECAL        TIME-TAG  2
l9v221f0q_rawtag_b.fits   FUV  WCA   G130M  1309  WAVECAL        TIME-TAG  2
l9v221f2q_rawtag_a.fits   FUV  PSA   G130M  1309  EXTERNAL/SCI   TIME-TAG  3
l9v221f2q_rawtag_b.fits   FUV  PSA   G130M  1309  EXTERNAL/SCI   TIME-TAG  3
l9v221f4q_rawtag_a.fits   FUV  WCA   G130M  1309  WAVECAL        TIME-TAG  3
l9v221f4q_rawtag_b.fits   FUV  WCA   G130M  1309  WAVECAL        TIME-TAG  3
l9v221f6q_rawtag_a.fits   FUV  PSA   G130M  1309  EXTERNAL/SCI   TIME-TAG  4
l9v221f6q_rawtag_b.fits   FUV  PSA   G130M  1309  EXTERNAL/SCI   TIME-TAG  4
```

To remove a member from association, `l9v221010_asn.fits`, and to use a different wavecal file, use the following commands in **PyRAF**.

```
cl> tedit l9v221010_asn.fits
# row   MEMNAME         MEMTYPE         MEMPRSNT
  1     L9V221EUQ       EXP-FP          yes
  2     L9V221EWQ       EXP-AWAVE       yes
  3     L9V221EYQ       EXP-FP          yes
  4     L9V221F0Q       EXP-AWAVE       yes
  5     L9V221F2Q       EXP-FP          yes
  6     L9V221F4Q       EXP-AWAVE       yes
  7     L9V221F6Q       EXP-FP          yes
  8     L9V221F8Q       EXP-AWAVE       yes
  9     L9V221010       PROD-FP         yes

Change the MEMNAME of row 8 to L9V221FSQ
Change the MEMPRSNT of rows 1 and 2 to no

cl> tprint l9v221010_asn.fits
# MEMNAME         MEMTYPE         MEMPRSNT
  L9V221EUQ       EXP-FP          no
  L9V221EWQ       EXP-AWAVE       no
  L9V221EYQ       EXP-FP          yes
  L9V221F0Q       EXP-AWAVE       yes
  L9V221F2Q       EXP-FP          yes
  L9V221F4Q       EXP-AWAVE       yes
  L9V221F6Q       EXP-FP          yes
  L9V221FSQ       EXP-AWAVE       yes
  L9V221010       PROD-FP         yes
```

Finally, reprocess the data by running **calcos** on the updated association file. See the following section for syntaxes of running **calcos**.

Run Calcos

Users may choose any of three ways to run **calcos** using Python, **PyRAF**, or from the UNIX command line. The input arguments and examples for each case are as follows:

1. To run **calcos** in Pyraf from the **hstcos** package:

```
--> stsdas.hst_calib.hstcos
--> calcos filename_asn.fits outdir=new
```

Table 3.2: Arguments for running **calcos** in PyRAF

Argument	Values	Default	Description
input	filename		Association table (`asn`) or individual raw file (`rawtag`, `rawaccum`) to be processed
outdir	directory name		The name of the ouptut directory
verbose	yes or no	no	Verbose output from calcos
quiet	yes or no	yes	Turns off most output to `STDOUT`
savetmp	yes or no	no	Save temporary files: `x1d_a, x1d_b, lampflash_a, and lampflash_b`
stimfile	filename		If specified, the stim positions will be written to (or appended to) this text file
livetimefile	filename		If specified, the livetime factors will be written to (or appended to) this text file
burstfile	filename		If specified, burst information will be written to (or appended to) this text file

2. To run **calcos** in Python:

```
>>> import calcos
>>> calcos.calcos('filename_asn.fits', verbosity=2, \
outdir="new")
```

Table 3.3: Arguments for running **calcos** in Python:

Argument	Values	Default	Description
asntable	'filename'	' '	Association table (asn) or individual raw file (rawtag, rawaccum) to be processed
outdir	directory name	None	The name of the ouptut directory
verbosity	0, 1, 2	1	0=quiet, 1=verbose, 2=very verbose
save_temp_files	True or False	False	Save temporary files: x1d_a, x1d_b, lampflash_a, and lampflash_b
stimfile	"filename"		If specified, the stim positions will be written to (or appended to) this text file
livetimefile	"filename"		If specified, the livetime factors wil be written to (or appended to) this text file
burstfile	"filename"		If specified, burst information will be written to (or appended to) this text file

3. To run **calcos** from the UNIX command line:

```
% calcos -o new --stim stim.txt fileaname_asn.fits
```

Table 3.4: Command-line options for running **calcos** in UNIX:

Option	Description
-q	Quiet
-v	Very verbose
-s	save temporary file
-o outdir	Output directory
-stim filename	Append stim locations to filename
-live filename	Append livetime factors to filename
-burst filename	Append burst information to filename

To redirect the **calcos** `STDOUT` to a file use the following command:

```
% calcos -v -o new filename_asn.fits > log.txt
```

While users are recommended to run **calcos** on association files, it is possible to run **calcos** with a single raw file as the input. In this mode, **calcos** will always automatically process both segment files for FUV data if they both exist. For example if `rootname_rawtag_a.fits is the input for` **calcos**, then `rootname_rawtag_b.fits` will automatically be processed. The data from both segments will be calibrated and combined to create the final product, `rootname_x1d.fits`

3.6.2 Using GO Wavecals

Through the use of associations, **calcos** also contains a provision to select wavecals other than the default for calibration of the science exposures. To use an exposure other than or in addition to the default wavecal, the user can add a row to the association table. The rootname (case insensitive) should be given in the MEMNAME column, the string EXP-GWAVE in the MEMTYPE column, and the value in the boolean MEMPRSNT column set to true (e.g. yes if you use the **IRAF tedit** task). Make sure that the `WAVECORR` keyword in the primary header of the raw science file is set to `PERFORM`, and then run **calcos** as normal. Note GO Wavecals can only be used with non `TAGFLASH` data.

3.7 Reference Files

This section contains a description of the COS reference files. See Figure 3.1 - Figure 3.5 for which modules use these files and Section 3.4 for explanations of how their contents are applied by those modules.

3.7.1 BRSTTAB: Burst Parameters Table

- File Suffix: `_burst`

The `BRSTTAB` file provides the parameters needed to identify bursts. It consists of a primary header extension and a binary table extension with the columns listed in Table 3.5. Details of the burst rejection routine are given in Section 3.4.2.

Table 3.5: `BRSTTAB` table contents

Column Name	Data Type	Description
SEGMENT	String	Segment name, FUVA or FUVB
MEDIAN_N	Double	Factor above the median count rate for a time interval to be identified as a burst
DELTA_T	Double	Normal sampling time for large burst detection (s)
DELTA_T_HIGH	Double	High count rate sampling time for large burst detection (s)
MEDIAN_DT	Double	Time interval used to search for localized bursts (s)
BURST_MIN	Double	Minimum threshold rate for small bursts (counts/s)
STDREJ	Double	Number of standard deviations above background noise for small bursts
SOURCE_FRAC	Double	Minimum factor small bursts must be above source counts.
MAX_ITER	Long	The maximum number of iterations used to re-evaluate the median to detect a localized burst
HIGH_RATE	Double	Total count rate threshold to use DELTA_T_HIGH instead of DELTA_T (counts/s)

3.7.2 BADTTAB: Bad Time Interval Table

- File Suffix: _badt

The BADTTAB reference file lists the start and end times of known bad time intervals. It is used by the BADTCORR calibration module to flag events in TIME-TAG events lists which occur during a bad time interval. In later processing the flagged events will be removed from the final calibrated data, and the exposure time header keyword, EXPTIME, updated. The bad time interval table consists of segment, start, and end columns (see, Table 3.6). The segments columns can be populated with either FUVA, FUVB or ANY. The start and end columns are in Modified Julian Date.

Table 3.6: BADTTAB table content

Column Name	Data Type	Description
SEGMENT	String	Detector segment, FUVA, FUVB or ANY
START	Double	Bad time interval start time in MJD
END	Double	Bad time interval end time in MJD

3.7.3 PHATAB: Pulse Height Discrimination Table

- File Suffix: _pha

The PHATAB reference file is only valid for FUV data, and is applied during the PHACORR step of **calcos** to filter non-photon events. The file consists of two extensions, the first being the primary header, and the second a binary table (see Table 3.7). The table lists the lower and upper thresholds for valid individual pulse heights in TIME-TAG mode. In TIME-TAG mode, each detector event has an associated pulse-height of 5 bits with values ranging from 0 to 31, The table also gives the minimum and maximum values for the location of the mean value of the pulse height distribution used in ACCUM mode. In ACCUM mode, a pulse height distribution histogram is generated for the whole exposure and downloaded as part of the science data file. The histogram includes all the digitized events for each segment independently of the currently defined subarrays. Note in ACCUM mode the pulse height is a 7 bit number with values ranging from 0 to 127.

Table 3.7: PHATAB table contents

Column Name	Data Type	Description
SEGMENT	String	Segment name, FUVA or FUVB
LLT	Long	Lower limit threshold (TIME-TAG)
ULT	Long	Upper limit threshold (TIME-TAG)
MIN_PEAK	Float	Lower limit for location of mean (ACCUM)
MAX_PEAK	Float	Upper limit for location of mean (ACCUM)

3.7.4 BRFTAB: Baseline Reference Frame Table

- File Suffix: _brf

The BRFTAB reference file is only applicable to FUV data and is used during pipeline processing in the TEMPCORR module to apply the thermal distortion correction. The FUV detector does not have physical pixels like a CCD. Instead, the x and y positions of detected photon events are obtained from analog electronics, which are susceptible to thermal changes. Electronic stim pulses are normally commanded during integration and are used as physical position reference points. To return the FUV data to a known physical space, the BRFTAB defines the stim positions.

The BRFTAB file consists of a primary header extension and a binary table extension. The table lists the stim locations, stim search regions, and the active detector areas (Table 3.8).

Table 3.8: BRFTAB table contents

Column Name	Data Type	Description
SEGMENT	String	Segment name, FUVA or FUVB
SX1	Double	X pixel coordinate (zero indexed) of stim1[1]
SY2	Double	Y pixel coordinate (zero indexed) of stim1
SX2	Double	X pixel coordinate (zero indexed) of stim2[2]
SY2	Double	Y pixel coordinate (zero indexed) of stim2
XWidth	Long	Half width of search region for stims
YWidth	Long	Half height of search region for stims
A_Left	Long	X pixel of left side of active region
A_Right	Long	X pixel of right side of active region
A_Low	Long	Y pixel of lower side of active region
A_High	Long	Y pixel of upper side of active region

1. Stim 1 is located in the upper left corner
2. Stim 2 is located in the lower right corner

3.7.5 GEOFILE: Geometric Correction File

- File Suffix: `_geo`

This file is only used for FUV data. The `GEOFILE` is used by the GEOCORR calibration module to perform the geometric correction. From the nature and construction of the XDL detectors, the physical size of the pixels vary across the detector. The geometric distortion maps are used to correct for this variation and to transform the data into a constant physical pixel size early in the data reduction calibration process. After the thermal correction has been applied and the detector digital span and position are adjusted to their reference values, as defined in the reference table, the geometric correction can be applied. This implies that all the files used to determine the geometric correction were initially thermally-corrected.

Each geometric correction reference file contains four IMAGE extensions. There are two for each segment, and for each segment, there is one for each axis. At a given (X,Y) location in the uncorrected COS data, the value at that location (corrected for binning and offset) in the geometric correction image gives the distortion to be subtracted from the X or Y coordinates.

3.7.6 DEADTAB: Deadtime Table

- File Suffix: `_dead`

The `DEADTAB` reference file is used in the DEADCORR module, to obtain the true number of events received compared to the number of events counted by the detector electronics.

There is one `DEADTAB` reference file for the NUV and FUV detectors. They consist of a primary header extension and a binary table extension which contains the livetime values for a given observed count rate and segment. The livetime is defined as:

livetime = observed rate / true rate

and can be used to calculate the true count rate.

3.7.7 FLATFILE: Flat Field File

- File Suffix: `_flat`

`FLATFILE` provides a flat field image which is used by the pipeline to remove the pixel-to-pixel variations in the detector. The FUV `FLATFILE` consists of a primary header and two 14000 x 400 IMAGE extensions, one for each segment. The NUV `FLATFILE` consists of a primary header and a 1024 x 1024 IMAGE extension.

Currently, there is no usable FUV flat field reference file from pre-flight testing, and a dummy file of all ones is being used. A plan is in place to obtain FUV flat-field data in orbit from standard stars for every grating and possibly every central wavelength of the FUV detector. Until these data are available, the FUV flat field processing will use a file consisting of ones, which leaves the data unchanged.

The NUV flat field is a combination of internal and external deuterium flat field lamp exposures from thermal-vacuum testing which illuminate the portion of the detector that will receive all of the incoming external light on orbit. The data cover the following pixel region of the detector: x (dispersion): 0 to 1023, and y (cross-dispersion): 495 to 964. The rest of the detector, where flat field data are not available, has a value of 1.0. The bottom four and top three rows of the detector do not fit well with the rest of the detector and they are flagged in the data quality table.

3.7.8 BPIXTAB: Bad Pixel Table

- File Suffix: `_bpix`

The data quality initialization table identifies rectangular regions on the detectors that are known to be less then optimal. The feature type describes the type of detector blemish enclosed within the bounding box and q is the quality value assigned to all events detected within the box. The regions were identified by visual inspection of the combined flat field data for each detector (and segment). The BPIXTAB files consist of a primary header and a binary table extension which consists of the columns listed in Table 3.9.

Table 3.9: BPIXTAB table content

Column Name	Data Type	Description
SEGMENT	String	Segment name, FUVA, FUVB, or ANY for NUV
LX	Long	X coordinate of lower left corner of region
LY	Long	Y coordinate of lower left corner of region
DX	Long	Width of region in X
DY	Long	Width of region in Y
DQ	Long	Data quality value to assign to current region
TYPE	String	Comment regarding current region

In the BPIXTAB table, the DQ field may have several different values, each associated with a unique issue as shown in Table 3.10.

Table 3.10: Data Quality Flag Values

Flag	Name	Description
0	DQ OK	No anomalous condition noted
1	DQ Softerr	Reed-Soloman error
2	DQ_Brush Mark	Brush mark
4	DQ Grid Shadow	Grid shadow mark
8	DQ Near Edge	Spectrum near an edge of the detector
16	DQ Dead	Dead spot
32	DQ Hot	Hot spot
64	DQ Burst	Count rate implies a burst (FUV only)
128	DQ Out of Bounds	Pixel is outside the subarray
256	DQ Data Fill	Data fill due to the telemetry drop-out
512	DQ PH Low	Pulse height is below cutoff
1024	DQ PH High	Pulse height is above cutoff
2048	DQ Bad Time	Time is inside a bad time interval
4096	DQ Bad Wavelength	Wavelength is out of bounds
8192	DQ Divots	Wrinkled appearance from detector flat field
16384	DQ Sdistortion	Vertical S distortion seen on FUVA

3.7.9 LAMPTAB: Template Calibration Lamp Spectra Table

- File Suffix: `_lamp`

The `LAMPTAB` files consist of a primary header extension and a binary table extension which contains an extracted 1-D spectrum from the internal PtNe calibration lamp through the WCA aperture, for each grating and central wavelength setting. It is used in the **calcos** pipeline to determine the pixel offset of the observed data. The structure of the template calibration lamp spectra table is shown in Table 3.11.

Table 3.11: `LAMPTAB` table contents

Column Name	Data Type	Description
SEGMENT	String	Segment name: FUVA, FUVB, NUVA, NUVB, NUVC
OPT_ELEM	String	Grating name
CENWAVE	Long	Central wavelength (Angstrom)
WAVELENGTH	Double	Array of wavelength (Angstrom)
INTENSITY	Float	Spectrum array

3.7.10 WCPTAB: Wavecal Parameter Table

- File Suffix: `_wcp`

The `WCPTAB` file contains information relevant for the wavecal pipeline processing. It consists of primary header extension and a binary table extension which is described in Table 3.12. A fixed RESWIDTH value of 6.0 pixels (per resolution element) is used for the FUV detector and a fixed RESWIDTH value of 3.0 pixels (per resolution element) is used for the NUV detector. The FUV STEPSIZE is measured by calculating the displacement in pixels from a PtNe spectrum obtained at a position of `FPOFFSET=0` to the position `FPOFFSET=-2` for segment A from the WCA (and dividing by 2). The NUV STEPSIZE is measured by calculating the displacement in pixels from a PtNe spectrum obtained at a position of `FPOFFSET=0` to the position `FPOFFSET=-2` for stripe B of the WCA (and dividing by 2). The XC_RANGE was estimated as 110% of the STEPSIZE for both FUV and NUV.

Table 3.12: `WCPTAB` table contents

Column Name	Data Type	Description
OPT_ELEM	String	Grating name
XC_RANGE	Long	Maximum Lag (amplitude) for cross correlation
RESWIDTH	Double	Number of pixels per resolution element in the dispersion direction
MAX_TIME_DIFF	Double	Defines 'close in time' for wavecals
STEPSIZE	Long	One step of OSM is this many pixels

3.7.11 DISPTAB: Dispersion Coefficient Table

- File Suffix: `_disp`

There are two `DISPTAB` files with similar formats, one for the NUV, and one for the FUV. They consist of a main header and a binary table in the second HDU. These tables provide the dispersion relations for each segment, aperture, optical element and central wavelength. Each file has the format given in Table 3.13.

Table 3.13: DISPTAB table format

Column Name	Data Type	Description
SEGMNT	String	Segment name, FUVA, FUVB, NUVA, NUVB, NUVC
OPT_ELEM	String	Grating name
APERTURE	String	Aperture name
CENWAVE	Long	Central wavelength of setting
NELEM	Long	Number of non-zero coefficients in the polynomial
COEFF	Double	Coefficients.
DELTA	Float	Offset for the science data

For P_x = the Doppler corrected pixel value in the dispersion direction, the associated wavelength for a specific segment, optical element, aperture, and central wavelength is given by

$$\text{lambda}(P_x) = \text{COEFF}[0] + \text{COEFF}[1]*P_x + \text{COEFF}[2]*P_x^2 + \text{COEFF}[3]*P_x^3 + \text{DELTA}$$

3.7.12 XTRACTAB: 1-D Spectral Extraction Table

- File Suffix: _1dx

There are two XTRACTAB files with similar formats, one for the NUV and one for the FUV. They consist of a main header and a binary table in the second HDU. These tables provide the information needed to extract the spectrum from a geometrically corrected image of the detector for each optical element and central wavelength. Each file has the format given in Table 3.14.

Table 3.14: XTRACTAB table format

Column Name	Data Type	Description
SEGMENT	String	Segment name, FUVA, FUVB, NUVA, NUVB, NUVC
OPT_ELEM	String	Grating name
CENWAVE	Long	Central wavelength setting
APERTURE	String	Aperture name
SLOPE	Double	Slope of the spectrum
B_SPEC	Double	Intercept of the spectrum
B_BKG1	Double	Intercept of the background below the spectrum
B_BKG2	Double	Intercept of the background above the spectrum
HEIGHT	Long	Height of the extraction window for the spectrum
BHEIGHT	Long	Height of the extraction window for the background
BWIDTH	Long	Width of the boxcar filter used to smooth the backgrounds

The spectral extraction of a source is performed by collapsing the data within a parallelogram of height HEIGHT that is centered on a line whose slope and intercept are given by SLOPE and B_SPEC. Similarly, two background spectra are determined by collapsing the data within a parallelogram of height BHEIGHT centered on the lines defined by SLOPE and B_BKG1 and SLOPE and BKG2. The background spectra are then smoothed by a boxcar of width BWIDTH. These are then scaled and subtracted from the source spectrum.

3.7.13 PHOTTAB: Photometric Throughput Table

- File Suffix: `_phot`

There are two `PHOTTAB` files with similar formats, one for the NUV, and one for the FUV. They consist of a main header and a binary table in the second HDU. These tables provide the information needed to convert from corrected detector counts to flux units of erg $s^{-1}cm^{-2}A^{-1}$ for each segment, optical element, aperture and central wavelength. Each file has the format given in Table 3.15.

Table 3.15: PHOTTAB Table Format

Column Name	Data Type	Description
SEGMENT	String	Segment Name
OPT_ELEM	String	Name of optical element
CENWAVE	Long	Central wavelength of the setting
APERTURE	String	Name of the aperture
WAVELENGTH	Double	Wavelength array in Angstroms
SENSITIVITY	Float	Sensitivity array

The units of the Sensitivity array are (count s^{-1} $pixel^{-1}$)/(erg s^{-1} cm^{-2} $Angstrom^{-1}$). For each segment, optical element, central wavelength setting, and aperture, these files contain arrays of wavelengths and sensitivities which can be interpolated onto the observed wavelength grid. The net counts can then be divided by the sensitivity curves to produce flux calibrated spectra.

3.7.14 TDSTAB: Time Dependent Sensitivity Table

- File Suffix: `_tds`

There are two such files, one for the FUV and one for the NUV. They are only used for spectroscopic data. The files contain the information necessary to determine the relative sensitivity curve at any given time by interpolating between relative sensitivity curves given at fiducial times which bracket the observation, or else extrapolate the results from the last curve if the observation date is more recent than the last fiducial date. Interpolation data are provided for each segment, optical element, and aperture (see Table 3.16).

Table 3.16: TDSTAB Table Format

Column Name	Data Type	Description
SEGMENT	String	Segment Name
OPT_ELEM	String	Name of optical element
APERTURE	String	Name of the aperture
NWL	Long	Number of wavelength points
NT	Long	Number of time points
WAVELENGTH	Double[NWL]	Wavelength array in Angstroms
TIME	Double[NT]	Fiducial times in MJD
SLOPE	Double[NWL, NT]	Percent per year
INTERCEPT	Double[NLW, NT]	Ratios of current curve to original curves

For an observation obtained at time T, which lies between TIME[j] and TIME[j+1], the sensitivity curve used to calibrate the spectrum will be corrected by the following factor:

(T - REF_TIME) SLOPE[i,j]/(365.25*100) + INTERCEPT[i,j].

where REF_TIME is a general reference time given in the header of the FITS extension.

CHAPTER 4:
COS Error Sources

In this chapter...

4.1 Error Sources Associated with Pipeline Processing Steps

In this section, we discuss sources of error that are associated with major steps in the COS calibration pipeline (**calcos**). Note that these steps themselves were already described in Chapter 3 and will not be repeated here; this section will only describe specific issues related to the error budget of the resulting data which were not described before.

4.1.1 Dark Count

Dark counts arise from a combination of detector effects and external sources. **Calcos** will remove the effects of detector background (which includes dark, scattered light, etc.) in the `BACKCORR` module. This is done after the `X1DCORR` converts the detector image to a 1D extracted spectrum. Here, we discuss the instrumental contribution, since it can be the limiting factor in the error budget for very faint sources.

NUV-MAMA Dark Count

On the ground, the intrinsic dark rate of the COS MAMA was measured at 3.8×10^{-6} counts s^{-1} $pixel^{-1}$, or about 4 counts s^{-1} over the entire detector. On orbit, the total dark rate will also include contributions from other sources, such as cosmic rays and *HST*'s proximity to the SAA. As a result, complete characterization of the actual dark rate cannot be made until after launch. Figure 4.1 shows a ground dark for the NUV MAMA.

The dark rates for the STIS MAMAs have been extensively characterized and modeled on orbit (see the STIS Data Handbook). The processes that added to the dark rates on that instrument (window phosphorescence, detector glow) are not expected to be a problem for COS, but only on-orbit data will verify this.

Figure 4.1: NUV Dark

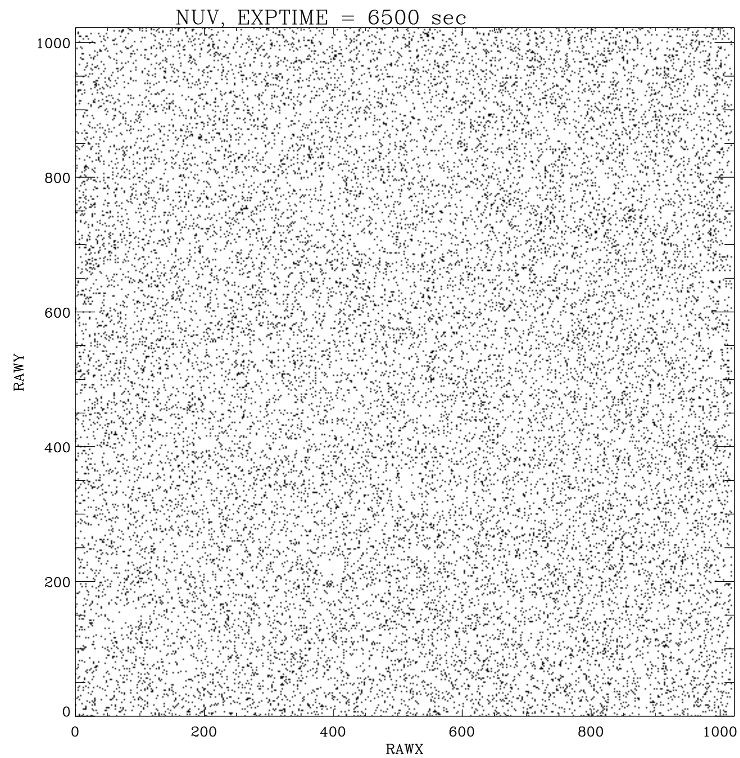


Image of a 6500 s dark for the COS NUV MAMA detector.

FUV-XDL Dark Count

The COS XDL FUV detector dark rate was measured on the ground to be approximately 9.6 and 10.4 counts s^{-1} for segments A and B, respectively. These amount to an average of less than 2×10^{-6} counts s^{-1} $pixel^{-1}$ over the active areas of the detectors. However, unlike the NUV detector, the dark in the two FUV detector segments has some structure, as can be seen in Figure 4.2. As with the NUV detector, this rate will increase on orbit, and will also vary with time and position in the orbit, so a final determination of the dark properties will have to wait until the detector is on orbit

Figure 4.2: FUV Darks.

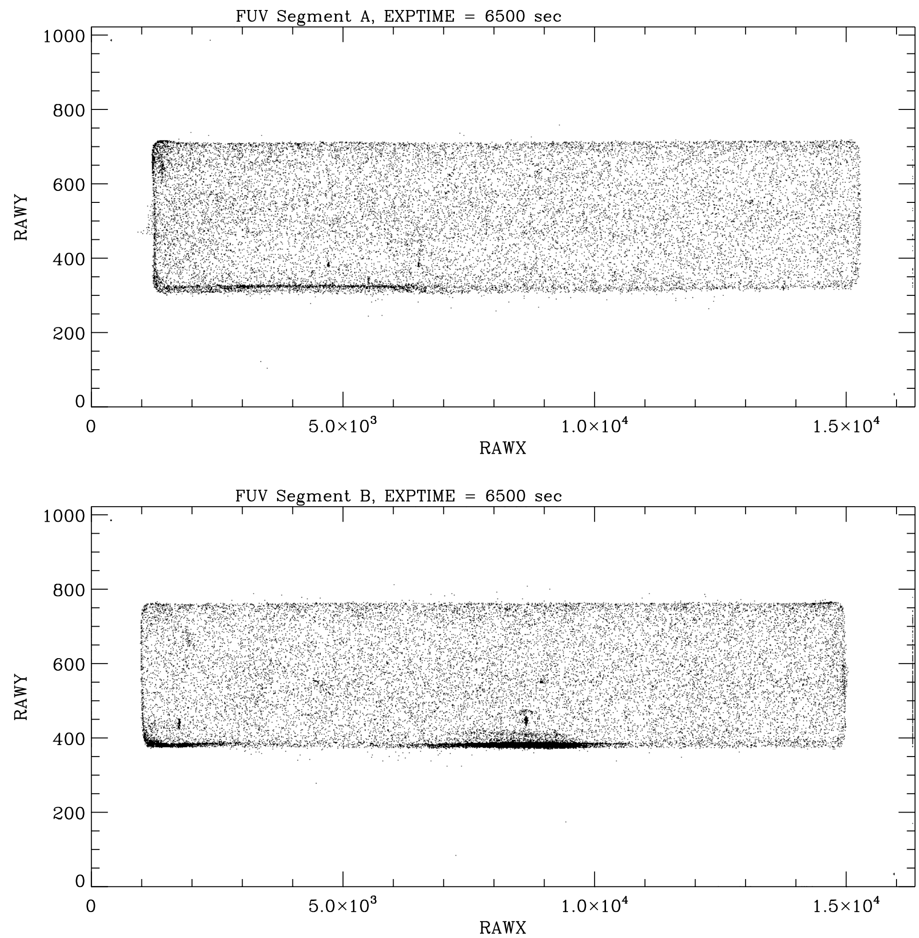


Image of a 6500 s dark exposures for segments A (top) and B (bottom) of the FUV detector. Notice that, unlike the NUV dark, these have considerable structure. Specifically, isolated hot spots and hot bands across the bottom of the active area, which is especially pronounced on segment B.

The FUSE mission used detectors with a design that was similar to the COS FUV XDL. The FUSE detector was subject to sudden, large increases in the background count rate which could last for many minutes. Although the likelihood of a similar problem on COS is unknown since the source was never understood on FUSE, the BURSTCORR routine has been added to **calcos** in order to detect and, if possible, exclude data taken whenever bursts occur. If bursts are seen after launch, BRSTCORR's burst-detecting algorithms will be adjusted based on the characteristics of the events seen.

4.1.2 Flat Fields

NUV-MAMA Flat Fields

The STIS MAMA flat fields are dominated by a fixed pattern that is a combination of several effects including "beating" between the micro-channel plates and the anode pixel array and variations in the charge cloud structure at the anode. Similar effects are present in the COS MAMA. Intrinsic pixel-to-pixel variations measured on the ground for the COS NUV-MAMA are 5.2% rms. The initial **calcos** flat field reference file was created from these data. Testing during SMOV will determine if this ground flat will be appropriate for on-orbit data or whether additional data will have to be obtained.

FUV-XDL Flat Fields

The FUV XDL detector has considerable fixed pattern noise, including dead spots and a honeycomb pattern due to the manufacturing process used to produce the MCP. The flat field obtained from internal ground tests did not improve the signal-to-noise expected to be achievable by the detector. Instead, a flat field will be determined from a set of observations of an external target during SMOV. `FP-POS` observations will be obtained at several positions orthogonal to the dispersion and these will be used to produce an FUV flat field image.

4.1.3 FUV XDL Thermal Drifts

The XDL centroiding electronics are sensitive to thermal effects. The `TEMPCORR` module of **calcos** measures the location of the stim pulses in order to determine the shift and stretch of the detector format and correct for any changes; `TEMPCORR` applies a linear correction based on the position of these stims. The accuracy of this correction will influence the ability to properly register the flat field corrections and may influence the final error budget.

4.2 Factors Limiting Flux and Wavelength Accuracy

4.2.1 Flux Accuracy

The accuracy of the absolute flux calibration of COS spectroscopic data is limited by several factors including:

- The accuracy of the absolute sensitivity calibration of the grating and central wavelength setting. The on-orbit absolute sensitivity calibration is determined by observing a standard star, with known absolute flux, well centered in both wavelength and cross dispersion direction within the aperture. The COS spectrum of the standard star is then extracted and the sensitivity required to produce the known flux of the star is determined as a function of wavelength. Deviations from nominal locations in the apertures or on the detectors may result in small errors. Once sensitivities based upon on-orbit observations are obtained, it is anticipated that COS sensitivity accuracies will be similar to the 1-2% typical of STIS FUV and NUV MAMA observations.

- The extraction procedure used to produce the spectrum. Initially, **calcos** will use a "boxcar" extraction procedure. This ignores the instrumental profile perpendicular to the dispersion, and treats all elements within the extraction box equally. To ensure that all of the signal is included, the width of the extraction box is typically larger than the spectrum and, as a result, includes more background signal than necessary. In order to avoid introducing artifacts into the extracted spectrum, the algorithm rejects all data in a column that contains a single bad pixel, thereby throwing away some of the data. Once on-orbit data enable the profile perpendicular to the dispersion to be characterized, an optimal extraction algorithm will be adopted which will provide improved signal-to-noise, especially for faint targets.
- The accuracy of the calibration of the time dependence of the sensitivity of the grating and wavelength region being used. Since COS is a new instrument, initially there will be no information on how the sensitivity may degrade with time. Characterization of this process will be the subject of an ongoing calibration campaign, beginning in SMOV and continuing throughout cycle 17 and beyond.
- Due to on-board Doppler corrections, a given pixel in `ACCUM` data will contain data from nearby pixels, which will cause a slight smearing of the fixed pattern noise.

4.2.2 Wavelength and Spectral Resolution Accuracies

There are several issues that may affect the COS wavelength calibration and spectral resolution, and these will be characterized on orbit. Some of these are:

- Because the dispersion relations are determined by a low (second or third) order polynomial, small-scale deviations from the relationship may be present for FUV XDL data. This was the case with FUSE and such small scale structure in the mapping from pixels to wavelength may also be present in the COS XDL detector although the geometric corrections should eliminate most of it. As a result, they may be present in early data supplied to the user.
- The NUV wavelength calibration procedure may not be optimal. Currently, all three stripes are assumed to move by identical amounts in pixel-space when grating motions occur. This assumption will be examined during SMOV. Should additional degrees of freedom be

required to obtain an accurate wavelength solution, early data could contain small, systematic errors in the mapping of detector pixels to wavelengths.

- OSM motions, or drifts, can cause the spectrum to shift in the dispersion direction. Ground-based testing indicates the scale of these drifts were occasionally 2-3 pixels (~1 resolution element for NUV, approximately one-half resolution element for FUV) in the first 20 minutes of motion, and rarely up to 6 pixels in the first hour of motion. Drifts tended to be negligible (< 1 pixel) after the first hour of motion. Although the drifts observed to date may be related to gravitational effects, in order to correct for this effect should it occur in zero-gravity on orbit, `TAGFLASH` wavecals have been implemented as the default mode for `TIME-TAG` observation. Ground-based testing indicates that corrections based on cross-correlation of `TAGFLASH` lamp flashes with a reference spectrum typically result in correction accuracies ≤ 0.5 pixel. Further, it is only possible to correct `ACCUM` data for the mean OSM motion that occurred during the exposure. In some circumstances, this may result in a slight degradation in the spectral resolution of `ACCUM` data.
- The accuracy to which the source is centered in the science aperture along the dispersion direction can result in small displacements in the absolute wavelength scale corresponding to the plate-scales of 0.22 arcsec per FUV pixel and 0.25 arcsec per NUV pixel. Initial expectations for `ACQ/IMAGE` centering accuracies are of the order of 0.05 arcsec, and accuracies of other types of COS acquisition are anticipated to be of the order of 0.1 arcsec. One can calculate the resulting wavelength accuracy using the plate-scale and dispersion given in Table 1.4 and Table 1.1 respectively.
- As discussed in the COS Instrument Handbook, the BOA degrades the target image, resulting in a reduction of the spectral resolution by a factor of three or more.

Once significant quantities of SMOV and cycle 17 calibration data are in hand, all of these effects will be examined, and a refinement of the wavelength accuracy will be determined.

CHAPTER 5:

COS Data Analysis

In this chapter...

5.1 Data Reduction and Analysis Applications

Most of the software tools for operating on COS FITS files are contained in two **STSDAS** packages:

- **hst_calib.hstcos**: Contains COS specific tasks to calibrate and interpret data, including **calcos**. Many of these tasks are described in Chapter 3. A complete listing is presented in Section 5.1.1.
- **toolbox.ttools**: Contains tasks to read and operate on FITS tables, which is the format used for COS spectral extractions, `TIME-TAG` data, and most COS reference files. These tasks and their syntax are described in Section 2.2.2 of the *HST* Introduction. We give specific examples of their use here in Section 5.1.2.

In addition to the above packages, most basic image manipulation software (e.g., **display**, **daophot**, **imexamine**, **contour**) and spectral analysis software (e.g., **splot**, **tables**, **specfit**, **igi**) available in **PyRAF/IRAF/STSDAS** can be used on COS data, either directly through the **PyRAF** or **IRAF** FITS interface or by converting data to another format. Chapter 3 of the *HST* Introduction includes information about how to display COS images and extracted spectra, as well as how and when to convert data formats, and a description of spectral analysis tasks. We present a brief summary of spectral display and analysis tasks in Section 5.1.3.

5.1.1 COS-Specific STSDAS Tasks

In Chapter 3, we gave detailed discussions of the use of the data reduction pipeline **calcos**. This task is contained in the **STSDAS** package **hst_calib.hstcos**. Other tasks useful for reducing and analyzing COS data are contained in this package as well. A complete listing and brief description of these tasks is given in Table 5.1. All of these tasks can be run in **PyRAF**. Consult the on-line help for each task for more information. Some of these tasks will be discussed in greater detail in the remainder of this chapter.

Table 5.1: COS-Specific Tasks

Task	Description
	calcos tasks
calcos	Process COS data through the calibration pipeline.
cc-edit[1]	GUI to display COS data
wplot[1]	Plots wavecal spectrum and overplots simulated spectrum from lamp line list.

1. These tasks will not be available until STSDAS release 3.9 expected in spring of 2009.

For the most up-to-date list of COS specific tasks as well as examples please refer to the COS Instrument Web site:

http://www.stsci.edu/hst/cos/updates/dhb.

5.1.2 FITS Table Tasks

COS spectral extractions, `TIME-TAG` data, and most COS reference files are stored in FITS tables. (See Section 2.4.1 for a description of the structure of the table extension files for spectral extractions and `TIME-TAG` data.) Tasks designed to handle this format can be used to examine and extract information from the tables. Here we give specific examples of the use of routines in **ttools** to help you get started. A sample output is given after each command.

To find out what information is given in the columns of a FITS table (the parameters listed here are discussed in depth in Section 5.4) using the **tlcol** task:

```
cl>tlcol filename_x1d.fits
SEGMENT         CH*4       %-4s ""
EXPTIME         D          %8.3f s
NELEM           I          %6d ""
WAVELENGTH      D[16384]   %25.16g angstrom
FLUX            R[16384]   %15.7g"erg/s/cm*2/angstrom"
ERROR           R[16384]   %15.7g"erg/s/cm*2/angstrom"
GROSS           R[16384]   %15.7g "count /s"
NET             R[16384]   %15.7g "count /s"
BACKGROUND      R[16384]   %15.7g "count /s"
DQ_WGT          R[16384]   %15.7g ""
DQ              S[16384]   %11d ""
```

To use **tread** to look at the contents of the table:

```
cl> tread filename_x1d.fits
Column  1      2         3
Label __SEGMENT__ ___EXPTIME___ _______NELEM________
   1    FUVA        249.760          16384
```

Note that the number of columns displayed is limited by the width of the window that you are working in. To see more columns, you can start **PyRAF** or **IRAF** in a wider window or populate the task parameter *columns* in **tread** with some of the column names reported by **tlcol**:

```
cl> tread filename_x1d.fits \
columns='WAVELENGTH,FLUX,ERROR'
Column     1         2         3
Label ____WAVELENGTH____ ____FLUX____ ___ERROR____
   1  1585.584808800369       0.          0.
```

The **tlcol** output indicates that some of the columns contain arrays of 16384 elements rather than a single value. For those columns, **tread** displays the value of the first element in the array. e.g., the initial wavelength in this `x1d` extraction is 1585.58 Angstroms. To plot an entire array, or plot one array against another, you can use a routine that operates on the table data directly:

```
cl> sgraph "filename_x1d.fits[1] WAVELENGTH FLUX"
```

You may want to change the parameter values in the supporting parameter sets **axispar**, **dvpar**, and **pltpar** to adjust the plotting. Alternatively, you may find it easier to work with FITS images. To make FITS image files of the arrays:

```
cl> tximage "filename_x1d.fits[1][c:WAVELENGTH]" filename_x1d_wave
cl> tximage "filename_x1d.fits[1][c:FLUX]" filename_x1d_flux
```

The images can then be displayed or operated on with standard FITS image handling routines such as IDL, e.g.:

```
IDL> filename_x1dwave=readfits('filename_x1d_wave.fits',0)
IDL> filename_x1dflux=readfits('filename_x1d_flux.fits',0)
IDL> plot,filename_x1dwave,filename_x1dflux

or

IDL> filename_x1d=mrdfits('filename_x1d.fits',1)

IDL> plot, filename_x1d.wavelength, filename_x1d.flux
```

Reference file FITS tables generally have many rows, with each row characterizing a specific operating mode, location on the detector, value of a parameter to be used in the reduction, etc.:

```
cl> tread lref$s7g1700h1_disp.fits
Column   1      2       3       4     5            6

Label  SEGMENT OPT_ELEM APERTURE CENWAVE NELEM COEFF

    1 FUVA   G130M   PSA    1291   2   1272.64958352037

    2 FUVA   G130M   WCA    1291   2   1272.21660012193

    3 FUVA   G130M   BOA    1291   2   1272.64958352037

    4 FUVB   G130M   PSA    1291   2   1119.38291962259

    5 FUVB   G130M   WCA    1291   2   1118.98191864602

    6 FUVB   G130M   BOA    1291   2   1119.38291962259

    7 FUVA   G130M   PSA    1300   2   1282.84550637193

    8 FUVA   G130M   WCA    1300   2   1282.41252297349

    9 FUVA   G130M   BOA    1300   2   1282.84550637193

    ....
```

5.1.3 General Spectral Display and Analysis Tasks

Table 5.2 lists some of the more useful **PyRAF/IRAF/STSDAS** applications for displaying and analyzing COS spectral data.

Table 5.2: Spectral Analysis Tasks

Task	Input Formats	Purpose
stsdas.analysis.fitting.nfit1d	2-D & 3-D tables, images	General 1-D feature fitting; part of the STSDAS fitting package.
stsdas.graphics.stplot.igi	2-D & 3-D tables, images	General presentation graphics; supports world coordinates.
stsdas.graphics.stplot.sgraph	2-D & 3-D tables, images	General 1-D plotting; supports world coordinates.
stsdas.contrib.spfit.specfit	1-D images, ASCII tables	General 1-D spectrum modelling package.
stsdas.hst_calib.ctools.tomultispec	3-D table	Converts spectra from rows of a 3-D table to an IRAF multispec image.
noao.onedspec.splot	multispec images	General 1-D spectral analysis.

Some of the tasks are intended for browsing data or producing publication quality plots: the **igi** and **sgraph** tasks are described in Chapter 3 of the *HST* Introduction.

Specfit is a powerful spectral fitting routine that provides the user with many options for modelling emission and absorption features, continuum, and extinction. The fitting parameters can be linked to each other (e.g., fixed wavelength ratio or fixed flux ratio for two emission lines) and constrained. Different algorithms can be selected to first explore chi-square space, then rapidly achieve convergence in the deepest minimum. The application is fully described in a help file.

Plotting COS Spectral Images (`flt`)

The FUV data consists of separate files for each of the two FUV detector segments, segment A and segment B. In general, each segment contains one target spectrum and one wavecal. But, for the NUV data there are three disjoint portions of the spectrum present on the image for both the target and the wavecal. The following examples illustrate the use of the **sgraph** task to plot COS FUV and NUV spectra from the 2-D `flt` files.

```
# target spectrum, FUV segment A:
--> sgraph l61h54cxr_flt_a.fits[1][*,451:480]

# wavecal spectrum, FUV segment A:
--> sgraph l61h54czr_flt_a.fits[1][*,551:580]

# target spectrum, NUV stripes A, B, C:
--> sgraph l61h57ahr_flt.fits[1][*,91:110]
--> sgraph l61h57ahr_flt.fits[1][*,196:215]
--> sgraph l61h57ahr_flt.fits[1][*,338:357]

# wavecal spectrum, NUV stripes A, B, C:
--> sgraph l61h57ajr_flt.fits[1][*,465:484]
--> sgraph l61h57ajr_flt.fits[1][*,569:588]
--> sgraph l61h57ajr_flt.fits[1][*,710:729]
```

Plotting COS Tabular Spectra (`x1d`)

COS spectra in tabular format are very similar to STIS spectra. The following examples illustrate the use of the **sgraph** task to plot COS FUV and NUV tabular spectra using a row selector and specifying the columns (e.g., wavelength and flux) to plot.

```
# target spectrum, FUV segments A and B (respectively):
--> sgraph "l61h54cxr_x1d.fits[1][r:row=1] wavelength flux"
--> sgraph "l61h54cxr_x1d.fits[1][r:row=2] wavelength flux"

# wavecal spectrum, FUV segments A and B (respectively):
--> sgraph "l61h54czr_x1d.fits[1][r:row=1] wavelength net"
--> sgraph "l61h54czr_x1d.fits[1][r:row=2] wavelength net"
```

5.2 Evaluating Target Acquisitions and Guiding

COS target acquisitions and the options available to the observer are fully described in the COS Instrument Handbook. If you are examining COS observations that were specified by another observer, please refer to the Instrument Handbook to understand the options and parameters that may have been used.

Virtually all COS observations start with one or more acquisition exposures. The purpose of the acquisition is to ensure that the object observed is well centered in the COS aperture being used so as to avoid throughput losses and to produce a reliable wavelength zero point. Examining the acquisition data should allow you to detect significant errors in the centering of the target. Note that target acquisition data are always uncalibrated.

5.2.1 Types of Target Acquisitions

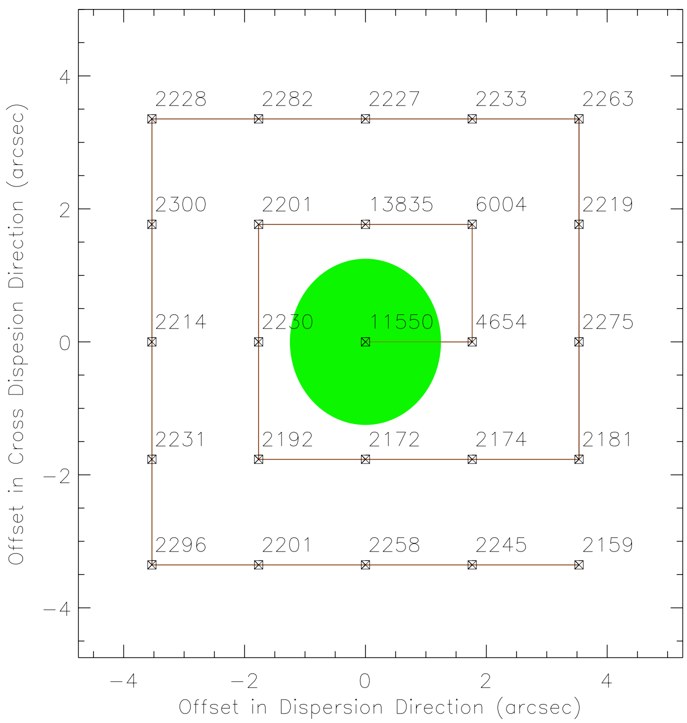
There are two types of COS acquisitions: imaging and dispersed-light. In an imaging acquisition, the COS NUV channel is used to obtain an image of the target in the COS aperture. This image is then analyzed by the COS flight software, the object's centroid is calculated, and the object is centered in the aperture. A dispersed-light acquisition directly analyzes the spectrum of the object being acquired and determines how to center the object so as to maximize throughput. Both types of COS acquisitions are intended to work on point sources or point-like sources.

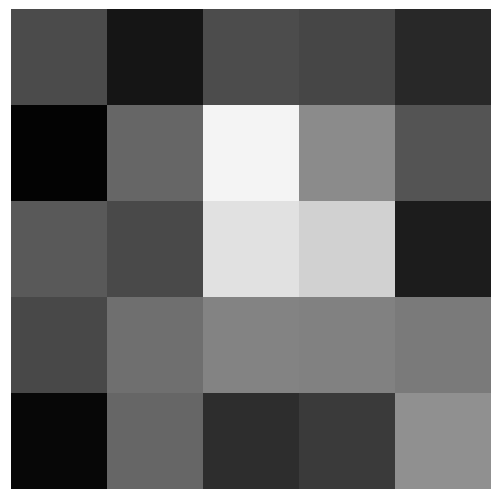
ACQ/SEARCH

A sequence of observations may begin with an `ACQ/SEARCH`, either in imaging mode or in dispersed-light mode (see the COS Instrument Handbook for a full description). The optical element selected will appear in the header: either a grating (and central wavelength) for dispersed light, or a mirror (`MIRRORA` or `MIRRORB`) for imaging. In either case, the telescope is commanded to move in a square spiral pattern, and at each dwell point an exposure is taken. The `STEP-SIZE` parameter sets the spacing between dwell points; the default is 1.767 arcsec, the optimum size to ensure that no area of sky is missed. The `SCAN-SIZE` parameter sets the number of dwells on each side of the square, and the choices are 2, 3, 4, or 5. If an even number of points is used (`SCAN-SIZE` = 2 or 4), the first point is offset by half the `STEP-SIZE` in both directions so that the overall pattern remains centered on the initial pointing.

The data from an `ACQ/SEARCH` exposure consists of a header and a binary table data extension which contains the accumulated counts at each dwell point, see Table 2.8. This array of counts was processed by the flight software to calculate the centroid and the telescope was then commanded to move to that centroid. A *quick verification* that an ACQ/SEARCH exposure was successful would be to find the values of the XDISP_OFFSET and DISP_OFFSET columns of the ACQ/SEARCH data table corresponding to the maximum counts value at a single dwell point. Then, compare the XDISP_OFFSET and DISP_OFFSET values to the ACQSLEWX and ACQSLEWY header keyword values (see Table 2.7). Similar, the data can be easily plotted for quick visual verification (see Figure 5.1).

Figure 5.1: Example of an ACQ/SEARCH exposure





Left: An example of an ACQ/SEARCH spiral pattern showing the offsets in the dispersion and cross dispersion directions from the initial pointing for each dwell point with the counts at each dwell point overplotted. The green circle in the center represents the science aperture and the initial pointing. Right: Linearly scaled image of the counts at each dwell point for the same 5 x 5 ACQ/SEARCH exposure.

Undispersed Light (Imaging) Acquisitions (ACQ/IMAGE)

When the ACQ/IMAGE command is used, two ACCUM exposures in imaging mode are taken for the specified exposure time, using the NUV channel of COS. The first exposure is taken after the initial pointing by *HST* and is used by the flight software to determine the centroid of the object and the amount of pointing change needed to center the object. The second image is taken after the object is centered to confirm that proper centering occurred. Each of the two images uses a sub-array of the size 816 × 345 on the COS NUV MAMA. The commanded motions of the telescope in x and y are provided in the ACQ/IMAGE header. The _rawacq file contains the initial target image as a 1024 x 1024 array, followed by the confirmation image, another array of the same size.

The appearance of the image of a point source recorded by COS in ACQ/IMAGE mode will depend on the aperture used (PSA or BOA) and the mirror (MIRRORA or MIRRORB). The best optical quality is achieved with the PSA used with MIRRORA, in which case a diffraction-limited image is created with a tight core. If MIRRORB was used instead to attenuate the source, two images of the source are produced (Figure 5.2). If the BOA was used, a neutral-density filter attenuates the source, but that filter has a slight wedge shape that degrades optical quality. Figure 5.3 shows images of point sources obtained with the BOA using MIRRORA and MIRRORB. Profiles of images taken with various combinations of (PSA, BOA) and (MIRRORA, MIRRORB) are shown in the COS Instrument Handbook.

The data produced by ACQ/IMAGE can be used to confirm proper acquisition of an object, by direct comparison of the two images.

Figure 5.2: Example of an image using the PSA and MIRRORB.

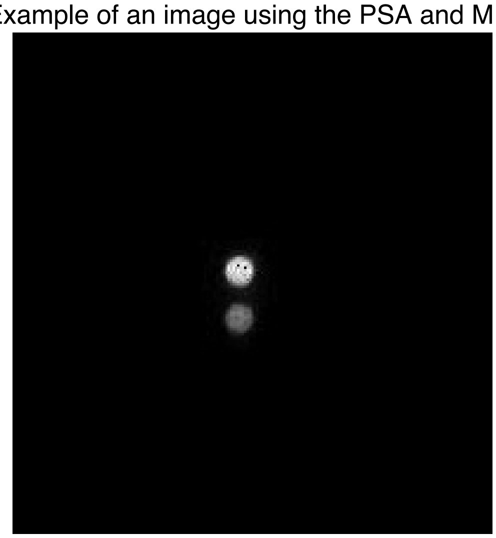
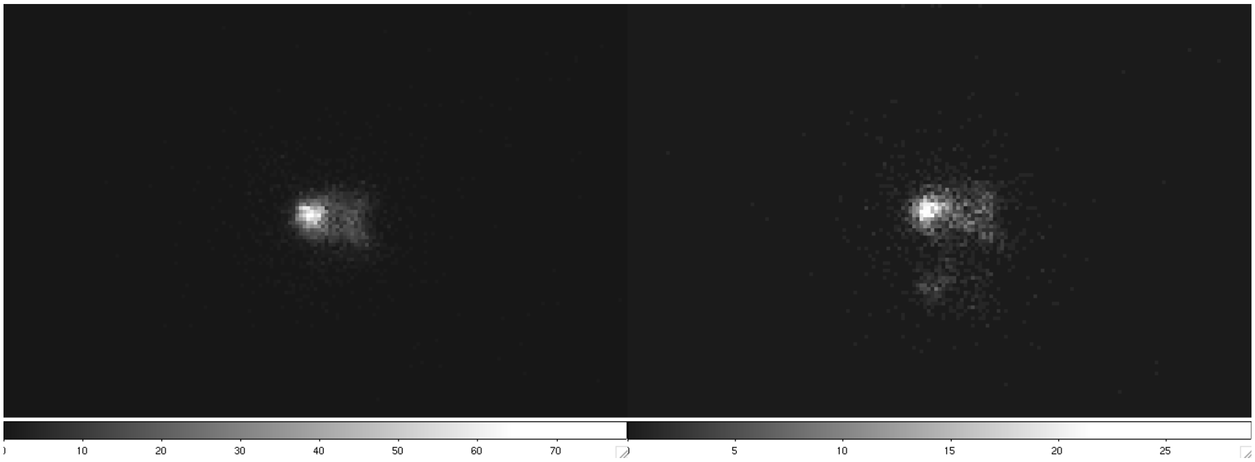


Figure 5.3: BOA and MIRRORA (left), and the BOA and MIRRORB (right).



Example of an image using the BOA and MIRRORA (left), and the BOA and MIRRORB (right)

Dispersed-Light Acquisitions (ACQ/PEAKD and ACQ/PEAKXD)

As noted above, an ACQ/SEARCH exposure can be performed in dispersed light. In that case, the file header will show a grating and central wavelength for the optical element chosen. As for ACQ/IMAGE, any acquisition performed in dispersed light can use either aperture: the PSA or BOA. In addition to ACQ/SEARCH, two other commands are available to improve the centering of an object in dispersed light: ACQ/PEAKXD and ACQ/PEAKD.

An ACQ/PEAKXD should always precede an ACQ/PEAKD if both were performed. ACQ/PEAKXD centers the spectrum in the cross-dispersion direction by obtaining a short exposure, calculating the centroid, and moving the telescope by that amount. Users will only receive files with headers containing the commanded movement of the telescope for ACQ/PEAKXD exposures. A *quick verification* that an ACQ/PEAKXD exposure was successful would be to compare the ACQSLEWY and (ACQPREFY - ACQCENTY) header keyword values (see Table 2.7).

ACQ/PEAKD centers the spectrum along the dispersion direction by executing a series of short exposures with the telescope moving the source in a line for a specified number of points (SCAN-SIZE), spaced by STEP-SIZE arcsec (effectively a 1-D ACQ/SEARCH). A centroid is calculated, and the same options available for ACQ/SEARCH are also available for ACQ/PEAKD. Following the centroid calculation, the telescope is moved to center the source, and the counts at each dwell point are recorded in a table, see Table 2.8. Users may compare the offsets associated with the dwell point containing the maximum counts to the telescope slews recorded in the header. A *quick verification* that an

ACQ/PEAKD exposure was successful would be to find the value of the DISP_OFFSET column of the ACQ/PEAKD data table corresponding to the maximum counts value at a single dwell point. Then, compare the DISP_OFFSET value to the ACQSLEWX header keyword values (see Table 2.7). The data can also be easily plotted for a quick visual verification, similar to what is shown in Figure 5.1 for the ACQ/SEARCH example.

5.2.2 Guiding Errors for Single-Guide-Star Mode

Tracking on two guide stars should provide pointing accuracy sufficient to keep targets centered in the COS aperture for several orbits. However, in some cases, observations are made using only a single guide star instead of the usual two. Either the General Observer has consented to this in consultation with the Program Coordinator when two suitable guide stars could not be found, or one Fine Guidance Sensor failed to acquire its guide star during the guide star acquisition/reacquisition. See Table 2.6 for keywords to check for the status of the guide star acquisition. In this situation, the roll of the telescope is under GYRO control, which may allow a slow drift of the target on a circular arc centered on the single guide star. The rate of the drift of the radius of this circle depends on the characteristics of the pointing for any particular observation, but typical values are expected to be in the range of 1.0 to 1.5 milliarcsec/sec (possibly, but very rarely, as large as 5 milliarcsec/sec).

To calculate the approximate magnitude of the drift of the target on the detector, you will need to find the distance of the target from the acquired guide star. The primary header of the observation log file jif identifies the acquired guide star (GSD_ID) and gives its right ascension (GSD_RA) and declination (GSD_DEC) in degrees. For example, for a target 10 arcmin from the guide star, a drift of the target around the guide star of 1 milliarcsec/sec during a 1,000 second exposure would cause the target to move 0.0029 arcsec on the detector. The direction of the motion on the detector can be deduced from header keywords in the science data describing the position angle of the detector (e.g. PA_APER) in combination with the direction perpendicular to the radiant. In many cases, the drift will be a small fraction of a pixel, although in some cases an image exposure may appear smeared.

5.3 Working with Extracted Spectra

Here we discuss ways of customizing the extraction of spectra and modifying reference files.

5.3.1 Working With x1d Files in IDL

While STScI does not support IDL tasks, the FITS files generated by the **OPUS** and **calcos** pipelines can be read directly into IDL by using the **mrdfits.pro** task. This task and many other are available as part of a library of astronomical IDL routines available from:

`http://idlastro.gsfc.nasa.gov/homepage`

5.3.2 Working With x1d Files in IRAF/PyRAF

When calibrating a single spectroscopic exposure, the **calcos** pipeline creates a one-dimensional extracted spectra file, with suffix `x1d` and a filename such as "`l9v220eqq_x1d.fits`".

COS `x1d` files are MEF format files and their data contents and extension formats are discussed in Section 2.4.3. As with other COS data files, the primary [0] extension will contain only header information, but no data. The extracted spectra are stored in a single [SCI] extension as *multi-dimensional* binary table. 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 COS spectra take advantage of this feature and are *three-dimensional*.

Using the "Selectors Syntax" to work with 3-D tables

In order to analyze COS tabular spectral data with **STSDAS** tasks other than **sgraph** and **igi** (which have been appropriately modified to handle three-dimensional data tables), you will need to extract the desired arrays from the three-dimensional table. Two 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. The task **tscopy** will copy rows, columns, and subsets of tables. 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). To obtain a list of the column names, you can run the **tlcol** task (type "`tlcol infile.fits`").

A *row selector* can be used to specify a certain row in the table. For example, if you specify:

```
infile.fits[1][c:WAVELENGTH,FLUX][r:SEGMENT=FUVA]
```

IRAF will extract data from the table stored in the first extension of `infile.fits`, specifically from the columns labeled WAVELENGTH and FLUX, and will restrict the extraction to the row containing segment A data. Part I Section 2.3.2 describes the selectors syntax and provides a few examples using the selector syntax to plot COS spectra.

Dumping x1d data to an ASCII File

In **PyRAF,** it possible to dump the arrays of an x1d file to an ASCII file by using the tasks **txtable** and **tdump**. For example, to extract the WAVELENGTH, FLUX and ERROR columns of FUV `x1d` file `l9v221fkq_x1d.fits`, first use the **txtable** task to convert the 3-D `x1d` table to a 2-D table:

```
--> txtable l9v221fkq_x1d.fits[1][r:row=1:2] l9v211fkq
l9v221fkq_x1d.fits[1][r:row=1:3] row=1  -> l9v211fkq_r0001
l9v221fkq_x1d.fits[1][r:row=1:3] row=2  -> l9v211fkq_r0002
```

This will create two new 2-D tables, `l9v221fkq_r0001.tab` and `l9v211fkq_r002.tab`, containing the FUVA and FUVB data respectively.

If using txtable to extract rows of 3-D COS tables, always use the row selector syntax of [r:row=1:50] to extract all possible rows. Even if there are less than 50 rows (as is always the case), txtable will return the correct number of files without error. This can be useful for dealing with the lampflash tables that can have varying number of rows depending on the number of flashes.

Then use the **tdump** task to dump the WAVELENGTH, FLUX and ERROR columns of the 2-D tables into one ASCII file using the following commands:

```
--> tdump l9v221fkq_r0001.tab columns="WAVELGNTH,FLUX,ERROR,DQ_WGT" \
>> l9v221fkqtest.tab


--> tdump l9v221fkq_r0002.tab columns="WAVELGNTH,FLUX,ERROR,DQ_WGT" \
>> l9v221fkqtest.tab
```

Plotting COS x1d Data

Each row of each of the science extensions in an `x1d` file will contain the columns listed in Table 2.5; a similar table, including array dimensions, can be displayed by using the task **tlcol** (see Section 2.3.2 Part I).

When using many **IRAF** and **PyRAF** routines with `x1d` files as input, it will be necessary to specify the extension number of the file. For example, to plot flux vs. wavelength in an `x1d` file, using the **sgraph** task, the command needed would be:

```
cl> sgraph "l9v220eqq_x1d.fits[1][r:row=1] wavelength flux"
```

For FUV data, the `x1d` files contain both[1] segments A and B. For example, to plot flux vs. wavelength in an FUV `x1d` file for segment A, using the **sgraph** task, the command needed would be:

```
cl> sgraph "l9v220eqq_x1d.fits[1][r:row=1] wavelength flux"
or
cl> sgraph "l9v220eqq_x1d.fits[1][r:SEGMENT=FUVA] wavelength flux"
```

To plot flux vs. wavelength in an FUV `x1d` file for segment B, using the **sgraph** task, the command needed would be:

```
cl> sgraph "l9v220eqq_x1d.fits[1][r:row=2] wavelength flux"
or
cl> sgraph "l9v220eqq_x1d.fits[1][r:SEGMENT=FUVB] wavelength flux"
```

For NUV data, the `x1d` files contain the three stripes A, B, and C. For example, to plot flux vs. wavelength in an NUV `x1d` file for stripe A, using the **sgraph** task, the command needed would be:

```
cl> sgraph "l9v220efq_x1d.fits[1][r:row=1] wavelength flux"
or
cl> sgraph "l9v220efq_x1d.fits[1][r:SEGMENT=NUVA] wavelength flux"
```

To plot flux vs. wavelength in an NUV `x1d` file for stripe B, using the **sgraph** task, the command needed would be:

```
cl> sgraph "l9v220efq_x1d.fits[1][r:row=2] wavelength flux"
or
cl> sgraph "l9v220efq_x1d.fits[1][r:SEGMENT=NUVB] wavelength flux"
```

To plot flux vs. wavelength in an NUV `x1d` file for stripe C, using the **sgraph** task, the command needed would be:

```
cl> sgraph "l9v220efq_x1d.fits[1][r:row=3] wavelength flux"
or
cl> sgraph "l9v220efq_x1d.fits[1][r:SEGMENT=NUVC] wavelength flux"
```

---

1. For FUV x1d files, both segments A and B will be present as along as the individual raw data from both segments were available at the time of processing. If only one segment was present during processing, then a row selector of row=1 will point to the data from that segment. Similarly, a row selector of row=2 will result in an error.

x1d files can also be displayed and analyzed using the **splot** routine in the **noao.onedspec** package of **IRAF** or **PyRAF**. This requires converting the x1d binary table first into a standard **IRAF** format (or OIF) which can be read by **splot**. The standard format is usually .imh format, containing the flux and wavelength information. The most convenient way of extracting the wavelength and flux information from the x1d table is to use **tomultispec** in the **stsdas.hst_calib.ctools** package. **tomultispec** takes rows from the x1d table and converts them into standard **IRAF** form. It fits a dispersion solution to each wavelength in the array and stores the solution in the header of the output **IRAF** spectrum.

If a COS x1d table contains spectra from segments A and B of the FUV channel (as would be the case if **calcos** were ran with both _rawtag_a and _rawtag_b files present), then the spectrum output by **tomultispec** will consist of two spectra, one from each channel. Running **tomultispec** on an NUV x1d file will always produce a multispec file with three separate spectra, one for each stripe.

The conversion can be done using the following command:

```
cl> tomultispec l9v220eqq_x1d.fits l9v220eqq_x1d.imh
```

This command will write the spectra from both channels to a single .imh file. However, a single channel can also be written if desired by specifying the row to be written. For example, to write spectra from FUV segment A use the following syntax:

```
cl> tomultispec l9v220eqq_x1d.fits[r:row=1] l9v220eqq_x1d.imh
```

Segment B would be denoted by row=2. For NUV data the three stripes are denoted as rows 1, 2, and 3 for Stripes A, B, and C respectively. The FLUX column is extracted by default, though other columns such as GROSS, NET, or ERROR could also be extracted.

The **IRAF** spectrum can now be displayed using **splot**:

```
cl> splot l9v220eqq_x1d.imh
```

Within **splot**, individual emission lines can be fit with commands like 'k' and composite lines can be deblended using 'd'.

5.3.3 Redoing Spectral Extraction

The **x1dcorr** module in **calcos** is designed to extract flux calibrated 1-D spectra from two-dimensional COS spectral images (flt files). This module is called by **calcos** as part of standard pipeline processing; its functioning in that role is described in Section 3.4.13.

Correcting for Shifts Along the Dispersion Direction

Properly aligning the spectrum along the dispersion direction is important not only for obtaining the correct wavelength solution, but also for properly applying the flux calibration. Incorrect registration of the spectrum will result in the wrong sensitivity being applied at each wavelength. This is especially important for low resolution spectra, since at some wavelengths the sensitivity changes rapidly with wavelength.

For `rawtag` exposures the wavecal lamp exposures are taken either concurrently with the science `rawtag` spectra (TAGFLASH) or they are acquired as separate `rawtag` spectra (AUTO or GO wavecals). For all science `rawaccum` exposures the wavecals are acquired as separate `rawtag` exposures.

The wavecal exposures are used by **calcos** to determine the location of both the wavecal image and the corresponding science image on the detector. The location may vary in a non-repeatable manner due to non-repeatability of the COS grating positions. When auto-wavecals are acquired as separate exposures they are taken close in time to the science exposures, with the grating in the same position as during the science exposure.

After processing data through **calcos**, you may decide that you need to shift the spectrum along the dispersion direction to correct offsets in the wavelength calibration. For example, wavelength calibration offsets may occur due to offsets of the target from the center of the PSA aperture (which can occur if the target acquisition was imperfect), or from repositioning of the grating due to thermal flexures. Assuming that **calcos** has been run on the data and a residual wavelength offset has been found in the calibrated spectrum, the offset can be corrected by first calculating the number of pixels corresponding to the offset, then subtracting it from the raw position of coordinates along the dispersion direction. The shift is applied to the RAWX column of the `rawtag` or `rawtag_(a,b)` file. In **IRAF** and **PyRAF** the **tcalc** task from the **ttools** package can be used to apply the shift:

```
cl> tcalc "l9v220eqq_rawtag_a.fits[1][r:RAWY=Y1:Y2]" "RAWX" "RAWX + SHIFT"
```

In the example above, the RAWX positions of the science spectrum in an FUV A `rawtag` file have been moved by "SHIFT" pixels. The shift is only applied to the parts of the science spectrum located between the pixels RAWY=Y1 and RAWY=Y2 along the cross-dispersion direction.

Adjusting the Background Subtraction

For spectra, a background region offset from the extraction region is used to determine the background. You can adjust the default parameters for this background region by first copying the `_1dx` reference file listed under the `XTRACTAB` keyword in the primary header to a local directory, then adjusting the background parameters within the local version of the `_1dx` reference file. Once you have adjusted the parameters to your satisfaction, edit the primary header of the `_rawtag` file (with an **IRAF** task such as **hedit**) to indicate the path to the local version of the `_1dx` file. You can then run **calcos** with the updated background subtraction parameters.

The background parameters available for editing in the `_1dx` file are:

- SLOPE: the slope of the line tracing the centers of both the spectrum and background regions
- B_BKG1: the y-intercept of the line tracing the center of the background region lying below (smaller Y- coordinate) the science spectrum
- B_BKG2: the y-intercept of the line tracing the center of the background region lying above (larger Y- coordinate) the science spectrum
- BHEIGHT: the total cross-dispersion height, in pixels, of the background extraction region. The upper and lower edges of the background are defined as +/- (BHEIGHT-1)/2 pixels from the line tracing the center of the background extraction region
- BWIDTH: the width of the background extraction region along the dispersion direction
- HEIGHT: the total cross-dispersion height, in pixels, of the source extraction region. The upper and lower edges of the source extraction region are defined as +/- (HEIGHT-1)/2 pixels from the line tracing the center of the source spectrum.

The values of these parameters in the local `_1dx` file can be edited with the **tedit** task in **IRAF/PyRAF**.

5.3.4 Splicing Extracted Spectra

The task **splice** can be applied to combine overlapping extracted COS spectra (e.g. spectra taken with different central wavelengths). It takes into account the error (ERR) array as well as the data quality (DQ) array. Handling of the DQ array is important as it helps **splice** perform the combination properly and avoid bad or noisy data in the output file arising from the large changes in throughput at the edges of the detector.

```
cl> splice obs1_x1d.fits,obs2_x1d.fits output_splice.fits
```

Please refer to the **splice** task help file for more useful information. If a multispec format spectrum is preferred for further analysis, the task **tomultispec** can be run on the output file of the **splice** task.

Running **splice** as mentioned above (rather than transforming individual `x1d` fits tables into multispec format before combining them) has important advantages: it keeps the science data, error, and DQ arrays intact allowing for easier error analysis, and it does not have a limitation on the number of segments or wavelengths to include, a problem with the multispec format due to the limit on the size of the FITS header which requires fitting the wavelength scale with a function.

5.4 Working with TIME-TAG Data

COS detectors can be used in ACCUM or TIME-TAG modes, as described in Chapter 5 of the *COS Instrument Handbook*. In TIME-TAG mode, the position and detection time of every photon is recorded in an events list. Detection times are recorded with 32 millisecond precision, although events may be buffered for as long as 32 milliseconds prior to assignment of a detection time.

For TIME-TAG datasets, the *HST* archive returns a raw events list in a file with a _rawtag suffix. The _rawtag file is a FITS file with two binary table extensions. The first extension contains the events list and the last extension a list of good time intervals, indicating time intervals when events are valid.

An events list in a _rawtag file is a FITS binary table extension named EVENTS, containing four columns named TIME, RAWX, RAWY, and PHA. Note only FUV data will include the PHA columns in the _rawtag files.

The TIME in the events extension contains the time when each event was recorded, relative to the start time (MJD) of the exposure given in the EXPSTART keyword of the primary FITS header.

In TIME-TAG the RAWX column contains the pixel coordinate along the spectral axis where each event was recorded. Corrections to remove Doppler shifts introduced by the orbital motion of *HST* are applied by **calcos** and placed in the corrtag file. The correction depends on optical element and the projected orbital velocity of *HST*, which varies over the course of an observation. In ACCUM mode, this Doppler compensation is applied on orbit during an observation and is included in the RAWX column, but in TIME-TAG mode the uncorrected positions are downlinked and Doppler compensation is applied during ground processing. The RAWY column contains the pixel coordinate along the spatial, or cross-dispersion, axis. No Doppler compensation is applied. The PHA column (for FUV data only) contains the pulse height amplitude for each event as an integer on a 5-bit scale.

After all EVENTS extensions in a _rawtag file, there will be one final binary table extension named GTI, containing columns named START and STOP. There will be associated start and stop times for every uninterrupted observing interval during a planned exposure. For most datasets, there will be only one START and one STOP time encompassing all buffer dumps in an exposure. Multiple good time intervals are possible, however - for example, if guide star lock is lost. Times in START and STOP are expressed in seconds since the start time (MJD) of the exposure given in the EXPSTART keyword of the primary FITS header. The exposure start time (JD) is also provided in the EXPSTARTJ keyword of the primary FITS header. The start time is also provided in the In **IRAF**, good time intervals can be examined using the **tprint** task in the **tables** package:

```
cl> tprint rootname_rawtag.fits[GTI]
```

where rootname must be replaced by the rootname of the `_rawtag` file being examined.

Useful **IRAF** tasks for analyzing and manipulating data taken in the `TIME-TAG` observing mode are listed in Table 5.3.

Table 5.3: Useful IRAF Tasks for Reducing TIME-TAG Data

Task	Purpose
tpar, tprint	Read the value for table header keywords
sgraph	Display the 1-D spectrum
tomultispec	Convert `x1d` spectrum into IRAF .imh spectrum
splot	Display an IRAF .imh spectrum
splice	Splice spectra together

5.4.1 Displaying TIME-TAG Data in DS9

To view a TIME-TAG file (`_rawtag_(a,b)` in the FUV, `_rawtag` in the NUV), open **ds9**, then choose 'open' from the menu bar at the top. The image will load but, save for a few pixels registering a value of 1, the remaining pixels will be zero.

Once the image is loaded, go to the menu item 'bin' and open the pull-down menu from that. From that pulldown menu you can choose the size of the image to view – generally you should make it as big as possible: 8192X8192 pixels for FUV data, 1024X1024 pixels for NUV.

NOTE: for the instructions below, the changes will not take effect until you click on the 'Apply' button.

Now choose 'Binning Parameters' from the 'bin' pulldown menu. This will open a new window with the binning parameters listed. You will notice right away that the Bin Columns are listed as TIME and RAWX. These are what is currently being displayed by **ds9** (which is why the image looks so strange when initially loaded). However, what you really want is RAWX vs. RAWY, so change that in the pulldown menu under Bin Columns.

You can also set the blocking size of the image in the 'Binning Parameters' window – just type in '2' in the Block field next to RAWX. By blocking this way along the dispersion direction, you can now see virtually all of the 16384 pixels along the dispersion direction. If you are looking at NUV data, then no additional blocking is needed – just leave the blocksize as 1, but choose the image size as 1024 pixels from the 'bin' pulldown menu.

Spectroscopic Data:

Next, from the 'Binning Parameters' window choose the part of the spectrum to be centered on the middle of the dispersion direction by clicking on the button marked 'center of data'. Now press 'Apply' on the binning parameters window to update the **ds9** display.

The spectrum should now be displayed, with the dispersion direction running from left to right. To better see the data, choose 'scale' under the main **ds9** menu bar, and from that pulldown menu choose a square root stretch and min/max range. You can now pan your cursor over the image, while holding the right button down on your cursor, until the contrast looks just right. If you would like to smooth the data a bit (this can be useful for bringing out fainter features and increasing signal to noise along the display), choose the 'Analysis' menu item under the main **ds9** menu bar and select 'smooth parameters'. A dialogue box will open, and from there you can set the number of pixels to smooth. Finally, you can also click on the 'Color' item on the **ds9** menu bar and choose 'invert color map' to get an inverted color map.

You can also load a `_corrtag_(a,b)` table in **ds9**, but in this case the appropriate columns to display are XCORR and YCORR. Otherwise, the same **ds9** commands apply as for `_rawtag` files. For both `TIME-TAG` and `ACCUM` spectroscopic data the `_flt` and `_counts` spectral images will load as simple 2-D images in **ds9**.

Imaging Data:

For both `TIME-TAG` and `ACCUM` imaging data The `_flt` and `_counts` images will load as simple 2-D images in **ds9**.

TIME-TAG Animation:

You can assign events registered during each time interval to a separate image in **ds9**, thereby creating a sequence of images which can be played as an animation. This can be useful in verifying the occurrence of lamp flashes in `TAGFLASH` data, in searching for the appearance of bursts in raw data, and so on. To bin the images in time, set up the image as described above – with RAWX and RAWY chosen in the 'Binning Parameters' dialogue box. At the bottom of the 'Binning Parameters' box is a parameter called 'Bin 3rd Column'. Set the value of this parameter to TIME. Next, choose the number of bins you would like to divide the event file into under the 'Depth' parameter. Setting this value to 10, for example, will create 10 separate images, with the first one showing all events registered during the first (`EXPTIME`/10) seconds, the next one showing all events registered between (`EXPTIME`/10) and (2*`EXPTIME`/10) seconds, the next showing all events registered between (2*`EXPTIME`/10) and (3*`EXPTIME`/10), and so on up to `EXPTIME`. The 'Min' and 'Max' parameters let you choose the range of values in time to display – usually this is pre-set to 0 and EXPTIME, and can be left unchanged to bin the entire image as above. Select 'Apply' to do the binning.

Note that some time will be required to create the sequence of images, and that binning the events in time in **ds9** is very memory intensive, and that it is easy to make **ds9** crash if EXPTIME is large (for example >1000 seconds) and the number of bins in 'Depth' is set to a large value (for example 30). It is best to start with a small value for 'Depth' that works, then increase the value if needed.

After the binning is done, a new dialogue box will appear called 'Data Cube'. Numbered from left to right will be the enumeration of the bins (in the example above from 1 to 10), along with a slider underneath. Click on 'Play' in that window to start the animation – it will play each of the binned images sequentially in the **ds9** window. Again, the spacing between each of the bins will be (EXPTIME/10) in seconds, or (EXPTIME/Nbin), where Nbin is the number of bins.

In the animation, it should be possible to see the TAGFLASH spectrum appear and disappear as the sequence progresses. Obviously the sequence will show the flashes only if the keyword TAGFLASH=AUTO or TAGFLASH="UNIFORMLY SPACED" is in the header of the event file.

To exit from the animation, close the 'Data Cube' window, and then set the 'Depth' parameter in the 'Binning Parameters' dialogue box to zero, and click 'Apply'. That will reset the image in **ds9** to show all of the data again.

It is possible to bin in other parameters as well, such as PHA. The logic is the same as above.

5.4.2 Manipulating TIME-TAG Data for Variability

Users may wish to process only sub intervals of TIME-TAG events, to look for variability in the data. One way to do this would be to divide an exposure up into several sub-exposures before re-processing.

If the following example is used for TAGFLASH data users are warned to be wary of splitting the lamp flashes between sub-exposures. Inappropriately sub-dividing TAGFLASH data could adversely affect the wavelength calibration in the final products.

For example, to split the exposure, `l61h9002r_rawtag.fits`, into two equally spaced sub-exposures use the task **tcopy**.

```
cl> tcopy "l61h9002r_rawtag.fits[1][r:time=0:60]" /
l61h90021_rawtag.fits

cl> tcopy "l61h9002r_rawtag.fits[2]" /
l61h90021_rawtag.fits

cl> hedit l61h90021_rawtag.fits[0] rootname l61h90021

cl> hedit l61h90021_rawtag.fits[1] exptime=60

cl> hedit l61h90021_rawtag.fits[0] filename   /
l61h90021_rawtag.fits


cl> tcopy "l61h9002r_rawtag.fits[1][r:time=60:]" /
l61h90022_rawtag.fits

cl> tcopy "l61h9002r_rawtag.fits[2]" /
l61h90022_rawtag.fits

cl> hedit l61h90022_rawtag.fits[0] rootname l61h90022

cl> hedit l61h90022_rawtag.fits[1] exptime 60

cl> hedit l61h90022_rawtag.fits[0] filename
l61h90022_rawtag.fits
```

Next, the two sub-exposures should be reprocessed through **calcos**. To reprocess the data, a new association file should be created to input to **calcos**. The original exposure, `l61h9002r_rawtag.fits`, is a member of association, `l61h90010_asn.fits`. To list the members of this association use the task **tprint**:

```
cl> tprint l61hh90010_asn.fits prdata=yes

# row   MEMNAME    MEMTYPE     MEMPRSNT

  1     l61h9002r  EXP-FP      yes

  2     l61h7006r  EXP-AWAVE   yes

  3     L61H90010  PROD-FP     yes
```

Then, a new association file should be created and edited by using the IRAF commands **copy** and **tedit**:

```
cl> tcopy l61h90010_asn.fits l61h9001A_asn.fits

cl> tedit l61h9001A_asn.fits
```

In **tedit,** manually change the `MEMNAME` of row 1 to `"L61H90021"`. Then, insert a new row after row 1 with `MEMNAME=L61H90022`, `MEMTYPE=EXP-FP`, and `MEMPRSNT=YES`. Finally, change the `MEMNAME` of the last row to "L61H9001A". Exit **tedit** by typing **Control-D,** then **q**.

The resulting association file should have the following content:

```
cl> tprint l61hh0101_asn.fits
# row   MEMNAME     MEMTYPE     MEMPRSNT
  1     L61H90021   EXP-FP      yes
  2     l61H90022   EXP-FP      yes
  3     L61H7006R   EXP-AWAVE   yes
  4     L61H9001A   PROD-FP     yes
```

Theses three files could then be reprocessed by **calcos** using the following command in IRAF:

```
cl> calcos l61h9001A_asn.fits
```

The resulting calibrated extracted spectra:

- `l61h90021_x1d.fits`
- `l61h90022_x1d.fits`

would each contain the calibrated sub-exposure information from `l61h9002r_rawtag.fits` and could be used to search for any variability in the data.

PART III:

Appendixes

In this part. . .

APPENDIX A:

IRAF Primer

In this appendix...

A.1 IRAF Overview

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 from storage media to producing plots and images. Most astronomers will already be familiar with **IRAF**, but we provide this tutorial for HST observers who are beginners in **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 online help facility.

Additional information on **IRAF**, in particular *A Beginner's Guide to Using **IRAF*** is available through the NOAO **IRAF** page at:

`http://iraf.noao.edu`

A.2 Initiating IRAF

This section explains:

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

We assume that your site has IRAF and STSDAS installed. If not, you should obtain and install the software. See Appendix A.4 for details.

A.2.1 Setting Up IRAF in Unix/Linux

Before running **IRAF** or **PyRAF** for the first time you need to:

1. Create an **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 put it in their user home directory (i.e., the default directory that you are in when you log in to the system). The **IRAF** home directory does not need to be in your home 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 `setenv` and `source` lines)

The directory name is site-dependent—check with your system staff (`/usr/stsci/iraf/`)

The "`mkiraf`" command initializes **IRAF** by creating a `login.cl` file and a subdirectory called `uparm`. After typing the "`mkiraf`" command, you will see something like 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**. Generic terminal types that will work for most users are:

- `xgterm` for sites that have installed X11 **IRAF** and **IRAF** v2.10.3 BETA or later.
- `xterm` for most workstations running under X-Windows.
- `vt100` for most terminals.

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 message 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 for 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 and FITS format images.

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 only have to do it again when you install an updated version of **IRAF**. It is also a good idea at this time to make sure your standard image size is set to the largest size of the images you may want to display. This is controlled by the `stdimage` keyword. **IRAF** will only display images in a visual tool up to this size. To enable image sizes up to 8192 × 8192, set `stdimage` equal to `imt7`.

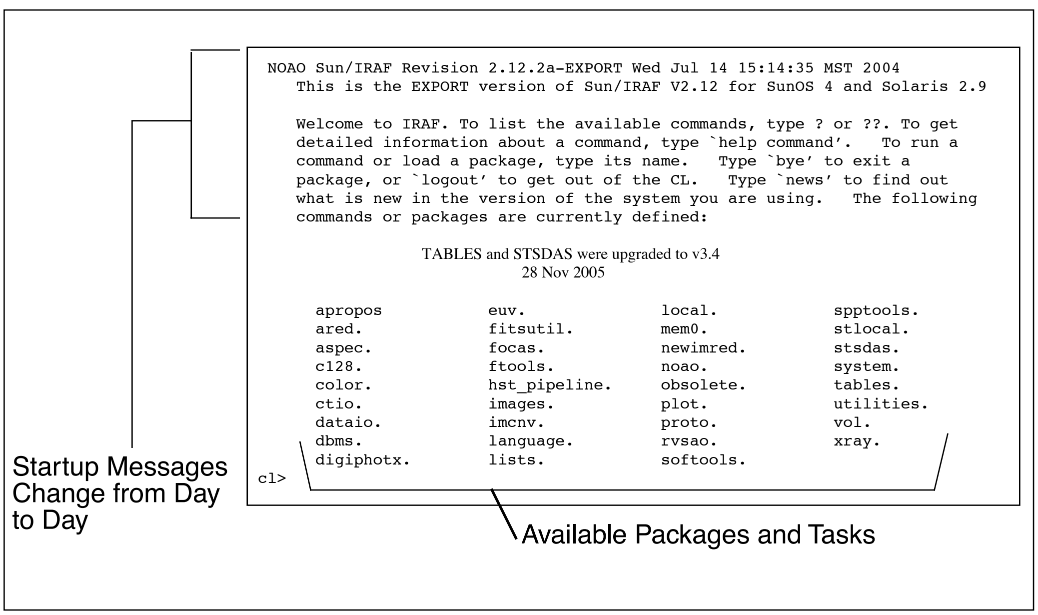
A.2.2 Starting and Stopping an IRAF Session

To start an IRAF session:

1. Move to your **IRAF** home directory.
2. Type `cl` at the command line.

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.



To quit an IRAF session:

Type "`logout`".

A.3 IRAF Basics

This section describes basic **IRAF** techniques such as:

- Loading packages
- Running tasks and commands
- Getting online help
- Viewing and setting parameters (see Appendix A.3.4)
- Setting and using environment variables (see Appendix A.3.5)
- File management
- Troubleshooting

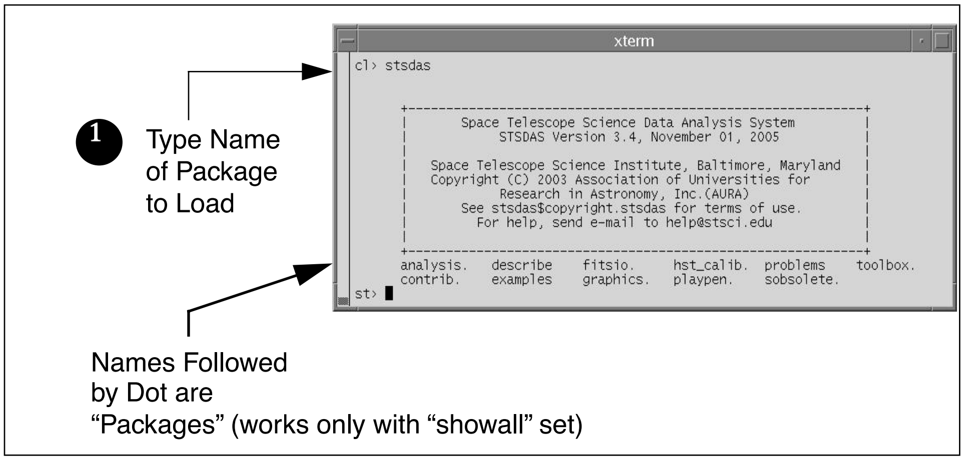
A.3.1 Loading Packages

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.

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**).

The most frequently used packages can be added to your `login.cl` file. 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
- `package` - Lists all packages loaded
- `bye` - Exits the current package

A.3.2 Running Tasks

This section explains how to run 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*. Alternatively, you can specify the values of the required *parameters* on the command line when you run the task. For example, if you want to print header information on `myfile.hhh`, type:

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

IRAF does not require you to specify 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 command) 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*, 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 text to the screen, you may 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 so:

```
st> taskname &
```

A.3.3 Getting Help

This section describes:

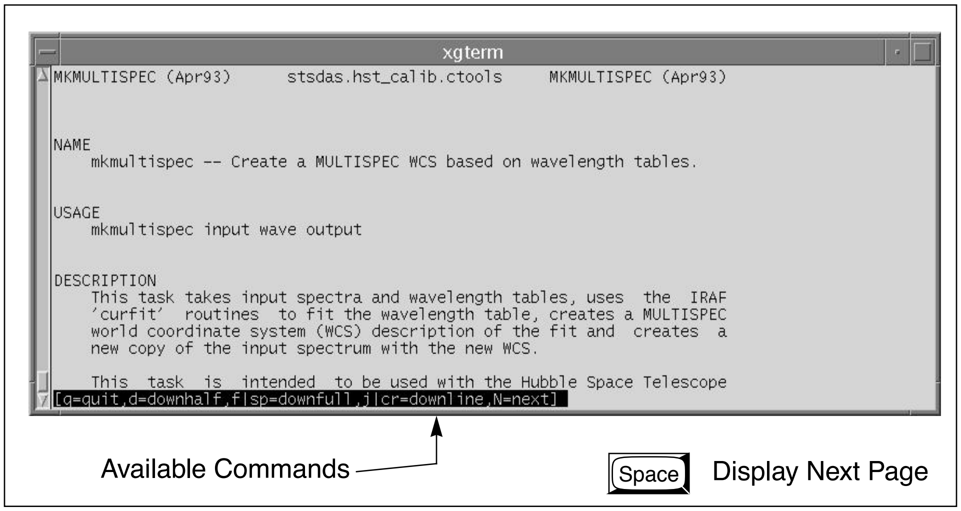
- How to use **IRAF**'s online help facility
- How to find a task that does what you want
- **IRAF** tutorials and CL tips

Online Help

You can get online help with any **IRAF** task or package by using the **help** command,[1] which takes as an argument the task or package name about which you want help. Wildcards are supported. To display the online help for the **STSDAS mkmultispec** task, type:

```
fi> help mkmultispec
```

Figure A.3: Displaying online help.



Two **STSDAS** tasks that display particular 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.

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

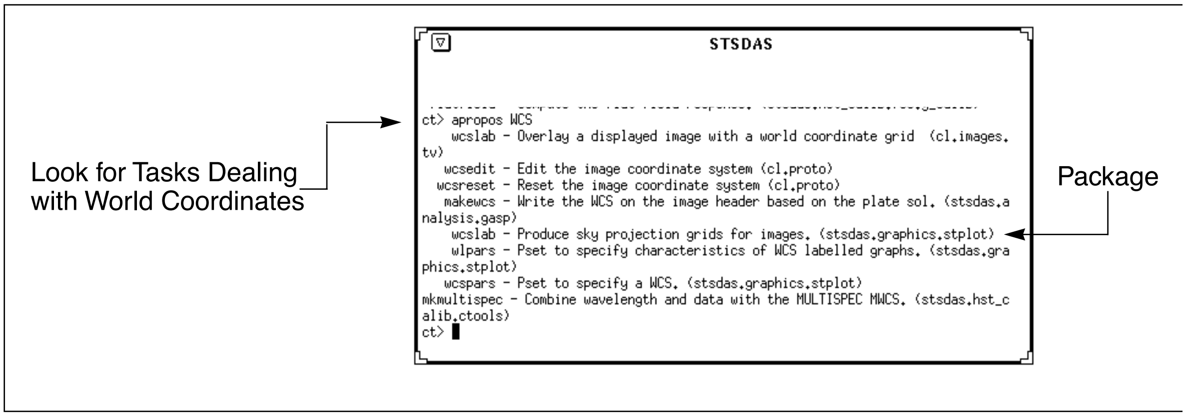
IRAF Tutorials and CL tips

Hints and tutorials are available on the NOAO Web site to help you become more familiar with **IRAF**. See the following Web sites:

http://iraf.noao.edu/iraf/web/tutorials/tutorials.html

http://iraf.noao.edu/tips/cl.html

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



A.3.4 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.

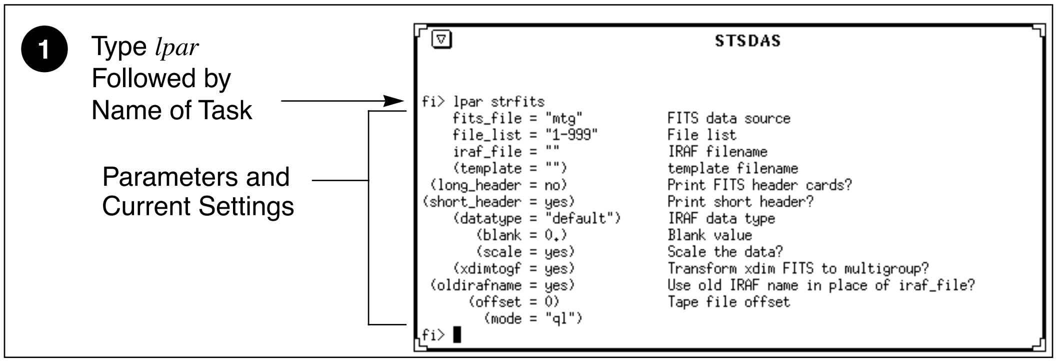
The two most useful commands for handling task parameters are:

- **lparam** to display the current parameter settings (abbreviated **lpar**)
- **eparam** to edit parameters (abbreviated **epar**)

Viewing Parameters with lparam

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

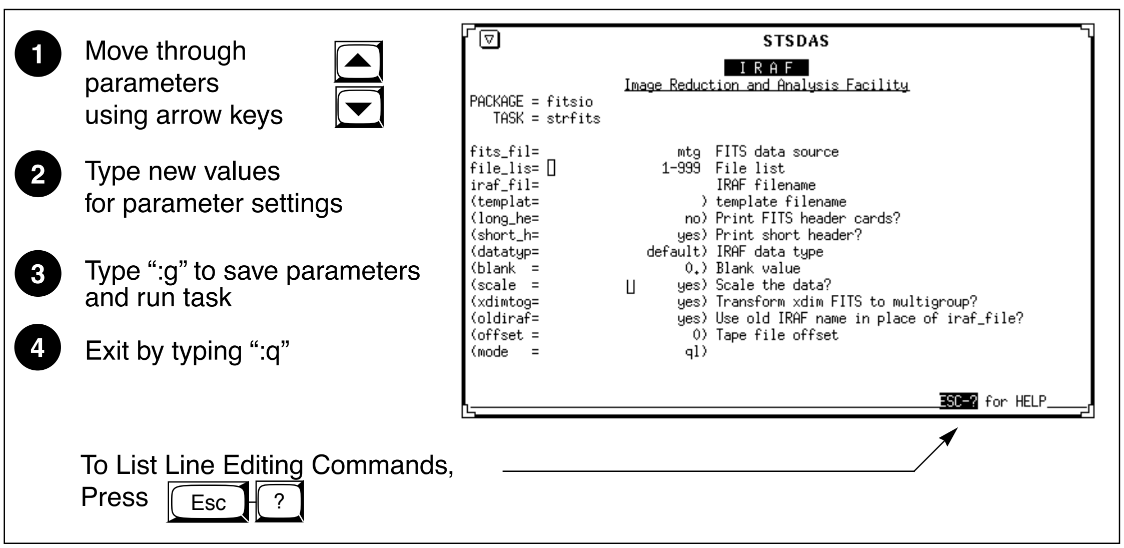
Figure A.5: Displaying parameter settings with **lpar**.



Setting parameters with eparam

Epar is an interactive parameter editor. It displays all of the parameters and their current settings. You can move around the screen using the arrow keys and type new settings for any parameters you wish to change. Figure A.6 shows what you will see when typing `epar strfits` when using the CL interface.

Figure A.6: Editing parameters with **epar** in **IRAF**.



Parameter 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. 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 allowed. Some tasks accept special features when specifying file names, including "@" lists, **IRAF** networking syntax, and image section or group syntax (see Appendix A.3.6).
`Integer`	Whole number. Often the task will specify minimum or maximum values (consult 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 a string.
`Pset`	Parameter set.

Restoring Parameter Default Values

Occasionally, **IRAF** might get confused by your parameter values. You can restore the default parameters with the **unlearn** command. You can use **unlearn** either on a task or an entire package. Help on using **unlearn** can be found online:

http://stsdas.stsci.edu/cgi-bin/gethelp.cgi?unlearn

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.3.5 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 with environment variables. Environment variables are defined 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 (Set this to the largest image you will mostly want to display. `imt7` will enable images up to 8192x8192.)	`set stdimage = imt1024`
`stdgraph`	Default graphics device	`set stdgraph = xgterm`
`clobber`	Allow or prevent overwriting of files	`set clobber = yes`
`imtype`	Default image type for output images. "`hhh`" is used for GEIS format and "`fits`" for FITS format. ("`imh`" is used for original **IRAF** format (OIF).)	`set imtype = "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.3.6 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*,[2] 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 `.q1d`.

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. See Chapter 2 for details on FITS files.

When working with IRAF (OIF) or STSDAS (GEIS) 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. 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., `j91k10pt*.fits`
  - ? Matches any single character, e.g., `z01x23x.c?h`

When using wildcards to specify GEIS files 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`.

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

- **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 User's 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: `nemesis!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.
- **back** - Revert to directory last visited.

A.3.7 Troubleshooting

There are a couple of easy things you can do to make sure that you do not 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 a few times.
- Occasionally, you may need to logout of the CL, restart **IRAF**, and try your command again.

If you still have a problem with an **IRAF** task, contact the NOAO Help Desk at `iraf@noao.edu`.

A.4 PyRAF Capabilities

PyRAF is a Python package that runs **IRAF** tasks from Python. All the features of Python are available when you run **PyRAF**. In addition, you can run **IRAF** tasks using the same syntax that you would use in the **IRAF** CL.

To start a PyRAF session:

1. Move to your **IRAF** home directory.
2. Type "`pyraf`" at the command line.

Like **IRAF**, several lines of introductory text will be displayed, and a prompt will appear at the bottom of the screen.

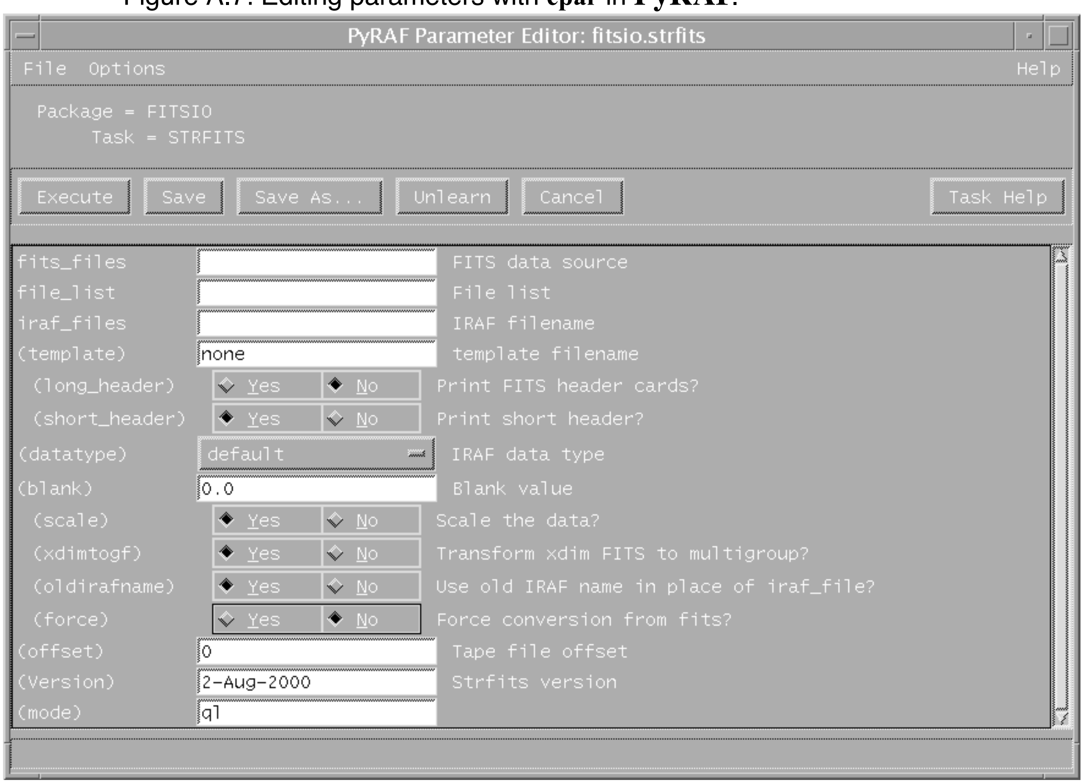
To quit a PyRAF session:

Type "`.exit`".

Setting parameters with eparam in PyRAF

In **PyRAF**, the **epar** editor uses a graphical user interface (GUI), as shown in Figure A.7. The **PyRAF epar** includes features like file browsing and pop-up help, which are not available in the **IRAF** CL.

Figure A.7: Editing parameters with **epar** in **PyRAF**.



Advantages of using PyRAF

There are many advantages of **PyRAF** over the **IRAF** CL. The error handling of **PyRAF** is superior. When using CL scripts and encountering an error, **PyRAF** reports the line number in the CL script where the error occurred. Python has excellent string handling capabilities and other features. Scripts can be written in Python, although you can still write and use CL scripts.

PyFITS can be used to read a FITS image or table to an in-memory array (a numpy array). Arithmetic manipulation can be performed with numpy arrays and results written to FITS images or tables. Although, there are certainly cases where it would be difficult to duplicate the features of an **IRAF** task using simple in-memory array arithmetic, tasks such as **imcombine**.

IRAF tasks are easily run from **PyRAF**. Interactively, there is no noticeable difference from the **IRAF** CL, but in a Python script you would preface the name of *any* **IRAF** task with "`iraf.`" (including the period). For example: "`iraf.phot`".

The following Python script runs the **IRAF** task **imstatistics**. It is saved in a file called "`imstat_example.py`".

```
#! /usr/bin/env python

import sys
from pyraf import iraf

def run_imstat(input):
       iraf.images()
       for image in input:
             iraf.imstat(image)

if __name__ == "__main__":
      run_imstat(sys.argv[1:])
```

This script can be run in two ways:

- from Python or **PyRAF**:

```
--> import imstat_example
--> imstat_example.run_imstat(["im1.fits[1]","im2.fits[1]"])
```

- from the Unix shell and if the file "`imstat_example.py`" is executable:

```
%> imstat_example.py im1ex1.fits im2ex1.fits
```

In the first example, the argument to the script is a list of two different images, each specifying the first extension of each FITS file. In the second example, each of the files passed as a parameter to the script contains the first extension of the FITS images used in the first example. For further information, see the **PyRAF** Programmer's Guide, which can be downloaded from:

`http://stsdas.stsci.edu/pyraf/stscidocs/index.html`

A.5 Accessing IRAF, PyRAF, and STSDAS Software

IRAF, PyRAF, and **STSDAS** are provided free of charge to the astronomical community. You can request the **IRAF** software by sending e-mail to: `iraf@noao.edu`. Instructions on how to download and install **PyRAF** can be found online:

`http://www.stsci.edu/resources/software_hardware/pyraf/current/download`

You must have **IRAF** or **PyRAF** installed on your system in order to run **STSDAS**. Detailed information about downloading and installing **STSDAS** is found at the following page:

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

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

When you download STSDAS, you should also download the TABLES package. TABLES must be installed prior to STSDAS. You must have IRAF or PyRAF installed on your system in order to install TABLES and STSDAS.

A.5.1 Accessing the Synphot Data Set

This handbook sometimes refers to the **synphot** data set, which must be available in order to run tasks in the **STSDAS synphot** package (see Section 3.4.4). The package is included with the **STSDAS** installation, but the corresponding data set must be retrieved and installed separately. Instructions on how to obtain and install the **synphot** data set can be found at:

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

Users should note that there are two separate synphot-related documents:

- the *Synphot User's Guide*, which includes instructions on how to use the tasks in the package, and
- the *Synphot Data User's Guide*, which includes instructions on how to set the different HST keywords.

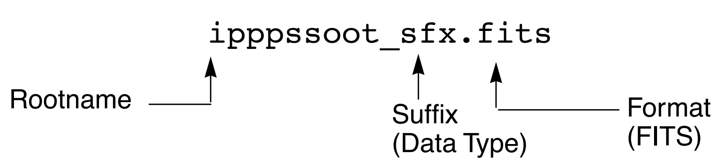
APPENDIX B:

HST File Names

In this appendix...

B.1 File Name Format

HST data file names encode a large amount of information about the files themselves. Datasets retrieved from the HDA, as described in Chapter 2 of the HST Introduction (Part I), consist of multiple files in Multi-Extension FITS format, each with a name that looks like this:



- ***Rootname***: The first part of the file name (`ipppssoot`) is the *root-name* of the dataset to which the file belongs. All files belonging to a given dataset share the same rootname.
- ***Suffix***: The part of the filename between the "_" and the ".fits" is called the *suffix (sfx)*, and it indicates the type of data the file contains. All science instruments except for COS, have data file suffixes with three characters. COS data file suffixes are between three and eleven characters long.

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

For example, a STIS data file named `o8v502010_x1d.fits` is a FITS file that belongs to the dataset with rootname "`o8v502010`", and its "`_x1d`" suffix indicates that it contains calibrated science spectra.

GEIS Files

In order to use **PyRAF/IRAF/STSDAS** tasks to work with data from the instruments FOC, FOS, GHRS, HSP, WF/PC-1, and WFPC2[1] you will want to convert these waiver FITS files into GEIS format. See Section 2.2 in the HST Introduction for instructions on how to convert FITS files to GEIS files using **strfits**. Like MEF format files, the names of GEIS files also derive from a file's rootname and suffix, and they have the form:

```
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 waiver 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 2.2.1 in the HST Introduction). For clarity, this handbook will use the term "extension" when referring to a component of a MEF format file and the term "suffix" when referring to the three character identifier in a filename.

1. Note that the HDA also distributes WFPC2 data in MEF format files and those files do not need to be converted to GEIS format.

B.2 Rootnames

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

Table B.1: IPPPSSOOT Root Filenames.

Character	Meaning
I	Instrument used, will be one of: ***e*** - Engineering data ***f*** - Fine Guidance Sensors ***i*** - Wide Field Camera 3 ***j*** - Advanced Camera for Surveys ***l*** - Cosmic Origins Spectrograph ***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 possible combinations). 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*** *- ACS, COS, NICMOS, or WFC3 association sub-product*

B.3 Suffixes of Files Common to All Instruments

The suffix of an HST file identifies the type of data that it contains. Several types of file suffixes are common to all instruments. These common suffixes are briefly described below. Please refer to the appropriate instrument's "Data Structures" chapters (in Part II) for the definitions of the instrument-specific suffixes.

Trailer Files

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

Support Files

Support files (suffix `spt`) contain information about the observation and engineering data from the instrument and spacecraft recorded at the time of the observation.

Observation Logs

Observation Logs (suffix `ji* or cm*`, aka "obslogs") are FITS files that contain information describing how the HST spacecraft behaved during a given observation. More information on these files is given in Appendix C.

B.3.1 Historical File Structures

Old Version Observation Logs

The engineering telemetry, which is the basis of the observation logs, does not get processed with OTFR. Thus any significant change in its processing results in a different type of observation log. Such a change occurred in February 2003, when the existing Observation Monitoring System (OMS) was streamlined to reduce the software maintenance costs. Obslogs produced prior to this time have `cm*` suffixes and contain science specific information, in addition to the pointing information. Obslogs produced after this date have `ji*` suffixes and do not contain any instrument specific information.

Obslogs headers, which you can read with the **IRAF** task **imheader** (see Section 2.3.3 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. Obs logs tables and images record spacecraft pointing information as a function of time. For more information on these files, 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 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 obs logs files. PDQ files were discontinued on May 9, 2002.

OCX Files

The suffix `ocx` denotes Observer Comment Files which were 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 discontinued on May 9, 2002.

B.4 Associations

The ACS, NICMOS, STIS, and WFC3 calibration pipelines sometimes produce single calibrated images from *associations* of many exposures. For example, a NICMOS, ACS, or WFC3 observer might specify a dithering pattern in a Phase II proposal. Those instruments would then take several exposures at offset positions, and the pipeline would combine them into a single mosaic (suffix "`_mos`" for NICMOS; suffix "`_drz`" for ACS and WFC3). 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 a single image, free of cosmic rays, that would be the association product (suffix "`_crj`").

The COS calibration pipeline instead uses associations to process *all* of COS science data. Please refer to the Chapter 2 (in Part II of the COS Data Handbook) for more details.

When you search the HDA 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 HDA, you will receive both the association product and the exposures that went into making it. The corresponding association table, stored in the file with suffix "`asn`" and the same rootname as the association product, lists the exposures or datasets belonging to the association. You can read this file using the **STSDAS tprint** or **tread** tasks (see Section 2.2.2 in the HST Introduction, Part I). 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 for that exposure (see Table B.1). For association products and sub-products, the last character will be a number between 0-8.

In practice, STIS stores the exposures belonging to associations differently than the other instruments. The exposures belonging to a STIS association all reside in the same file, while the exposures belonging to an ACS, COS, NICMOS, or WFC3 association reside in separate data files. See the relevant Data Structures chapters (in Part II) for more details.

APPENDIX C:

Observation Logs

In this appendix...

C.1 Observation Log Files

Observation Log Files, also known as *jitter* 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.

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). The `jit` table accompanies the `jif` header. The `jit` table is the three-second average pointing data. The `jif` header is a two-dimensional histogram of jitter excursions during the observation which includes the header plus some image related keywords. For these files an orbit model is used to recalculate the orbit position and velocity at the exact time needed.

A detailed description of the old observation log files can be found online: 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 at: 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 Jitter File Contents

The EDPS jitter files are limited to the engineering data that describe 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 EDPS jitter files contain:

- *rootname*`j_jif.fits`*:* The FITS header contains keywords providing information regarding the file structure, observation details, modeled background light, point control system, jitter summary, orbital geometry, and problem flags and warnings. The extension 0 header will contain parameters relating to the entire association or dataset, ending at the "ASSOCIATION KEYWORDS" block. The header values for extensions 1 and beyond will contain all the group 0 keywords as well as additional blocks of parameters relating to that particular exposure in the association.
- *rootname*`j_jit.fits`: This is the 3-second average jitter table. It contains the reconstructed pointing, guide star coordinates, derived jitter at the SI aperture, pertinent guiding-related flags, orbital data (e.g., latitude, longitude, limb angle, magnetic field values, etc.) and FGS_flags. There are examples of headers and tables in the online Observation Logs documentation located at:

  http://www.stsci.edu/hst/observatory/obslog/OL_3.html#HEADING2

C.1.2 Observation Log File Contents

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 and Figure C.2 show 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 useful information for deconvolution 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.1.3 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 ACS, NICMOS, and STIS association files, see Appendix B and Part II of each instrument's Data Handbook. 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.

```
SIMPLE  =                    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: A 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	Take data status
SlewFlag	0,1 flag	Vehicle slewing status

C.1.4 Jitter File Contents (February 2003 Version)

The current format of the jitter files produced by the EDPS is 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 describe 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, 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 identification) 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 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 Instruments Division at

`http://www.stsci.edu/hst/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 files "`.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

A total of 43 OMS Obslog keywords were dropped for the new jitter image file.

Changes in Table File

The jitter table files "`.jit`" or "`.jwt`" have 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 to BrightLimb. EarthMag was an estimate of stray light from the Earth, based on a model that is no longer supported by EDPS. BrightLimb is a short integer value, where 1 indicates 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 FGSs. 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 suffixes, "`.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

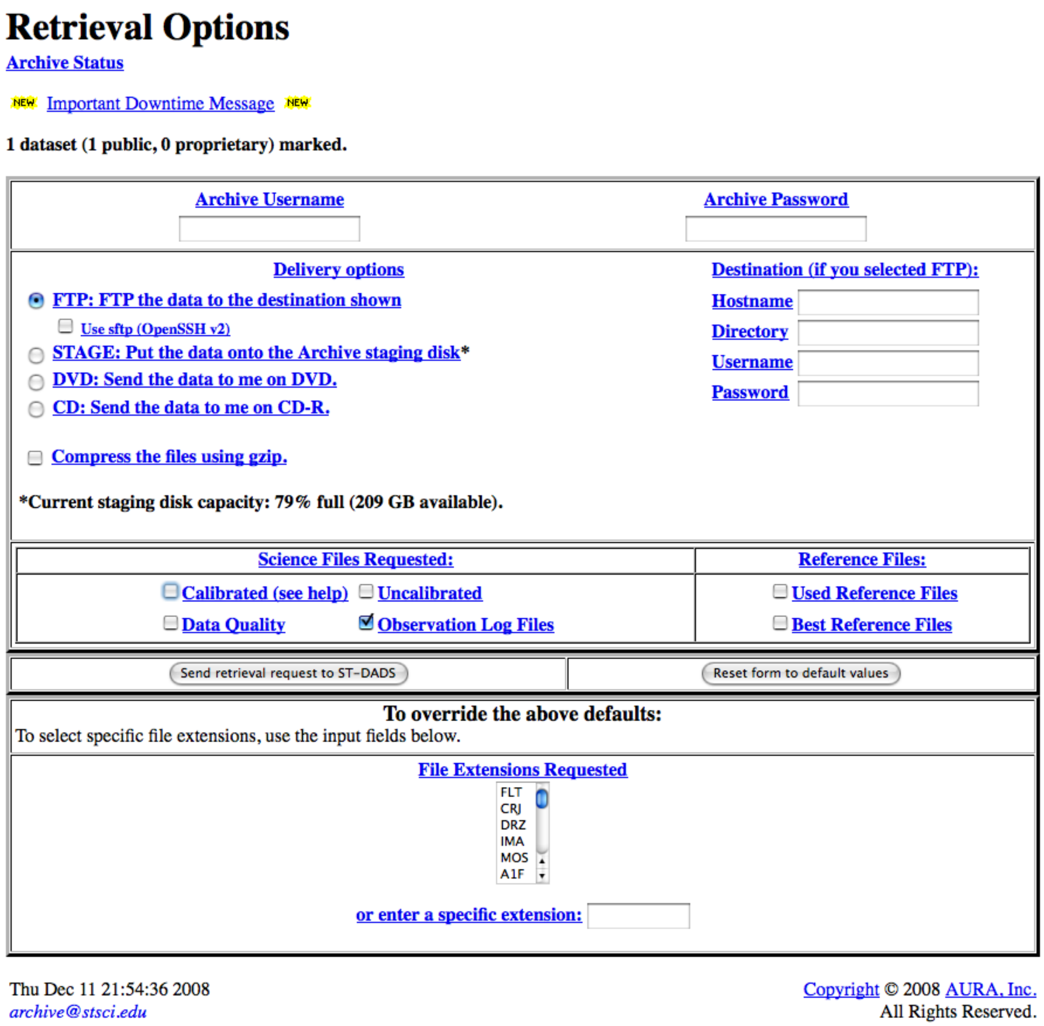
`http://www.ess.stsci.edu/teams/dsb/`

C.2 Retrieving Observation Logs

You can retrieve observation log files from the HDA, located at http://archive.stsci.edu/cgi-bin/dataset_input/. Unlike science data, which generally have a one-year proprietary period, observation log files become public as soon as they are archived.

The easiest way to get observation log files is to identify the observation of interest and then proceed with your request, as described in Chapter 2.9 of the ***STScI Archive Manual***, until you reach the "Retrieval Options" screen, shown in Figure C.3. You then need to select "Observation Log Files" in the "Science Files Requested" section of the page. The HDA will then deliver the associated observation log files. You will receive the "`_jif.fits`" and "`_jit.fits`" files.

Figure C.3: Choosing observation log files in MAST.



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 will not 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 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 researchers 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 observation log keywords is an example of two star guiding. These keywords are found in the `jif` file or, for older data, the `cmh` file.

```
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
```

The guide star identifications, GSD_ID and GSR_ID, are different for the two Guide Star Catalogs: GSC2 IDs are 10-characters in length, like those of GSC1, but consist of both letters and numbers. GSC1 IDs consist entirely of numbers.

The GSC2 catalog is the default catalog since Cycle 15 (June 2006). The keyword REFFRAME in the primary science header indicates which catalog was in use for an observation. This keyword is included in all Cycle 15 and later observations, and is either "GSC1" for Guide Star Catalog 1, or "ICRS" for International Celestial Reference System, upon which GSC2 coordinates are based. The same information is added to the HST Archive catalog file "shp_refframe" of the "shp_data" database table since June 2006. For more information about the catalogs and their astrometric accuracy, see: http://www-gsss.stsci.edu/Catalogs/Catalogs.htm

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 GHRS observations can span several orbits. If during a multiple orbit observation the guide star reacquistion 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 observation log 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 exposures 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 observation dataset spans multiple orbits, the guide star will be reacquired, 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.

In two-gyro mode, all observations are scheduled with two guide stars. Proposers cannot request the use of only one guide star.

Guide star handoffs are not allowed in two-gyro mode.

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-axes) 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 OMS 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-axes. The following list of header keywords, found in the `jif` or older `cmh` files, 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)
```

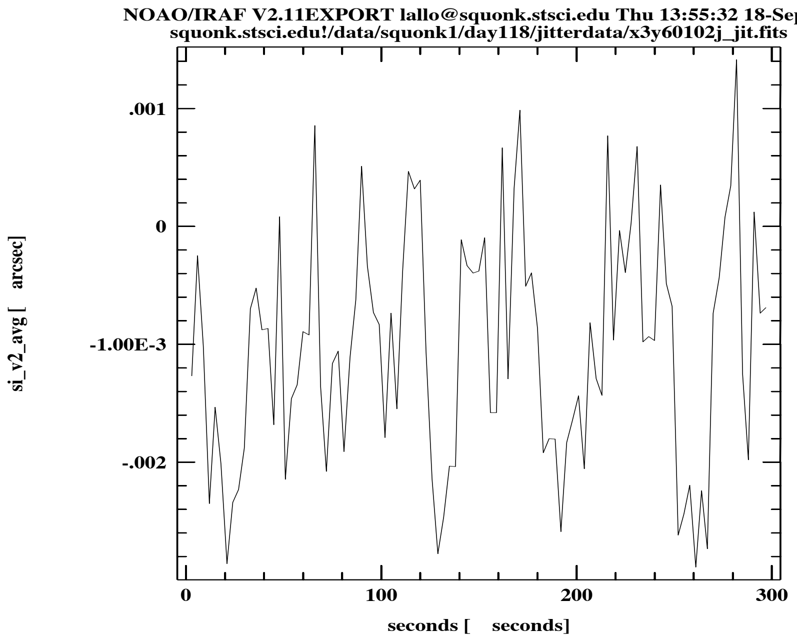
Recentering events occur when the spacecraft software decides that shaking is too severe to maintain lock. The FGS will release guide star control and within a few seconds reacquire the guide stars. It is assumed the guide stars are still within the FGS field of view. During the recentering time, INDEF 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 will start after the guide stars are acquired (or reacquired), and the observation may stop before the "Loss-of-Lock" or "Recentering" event that occurred at the end of an OMS window.

The **sgraph** command 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"
```

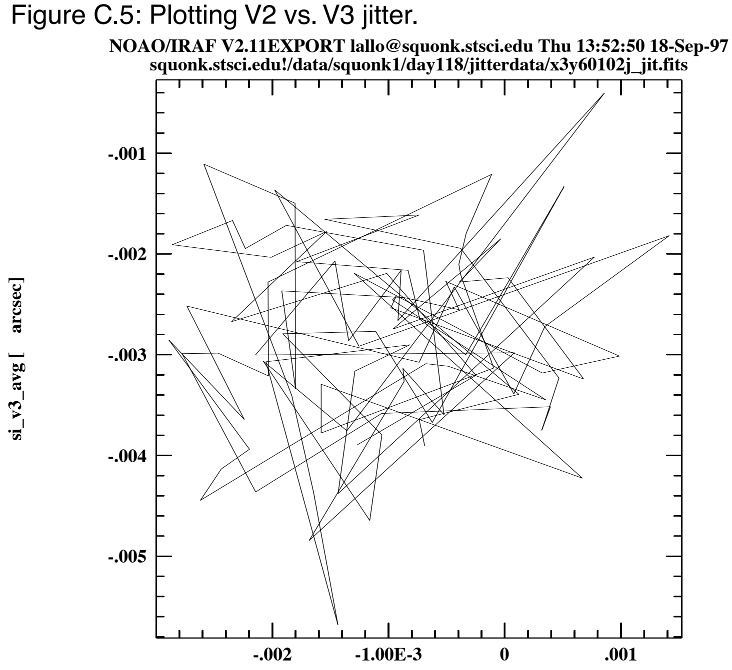
Figure C.4: Plotting jitter along V3-axis.



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.



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 HST 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. Documentation is available via the Web at: http://www.stsci.edu/hst/observatory/documents

Index

H

I

J

K

L

U

V

W

X