SolarSoft XRT Analysis Guide

Hinode X-Ray Telescope

Revised by Lucas Guliano

July 2023

Prepared at:

CENTER FOR ASTROPHYSICS
HARVARD & SMITHSONIAN

July 2023 Revision:

Updated *Using xrt cat.pro* section

March 2022 Revision:

Updated *Stray Light Overview* section

Added *Stray Light Correction* section

November 2019 Revision:

Updated the *Coaligning XRT* section

Updated the *Known Issues* section

Updated the *Contamination* section

Updated *Stray Light Section* section

June 2018 Revision:

Added list of URLs after Table of Contents.

Added section: *Browsing Data* to Chapter 2.

Added section: *Known Issues* to Chapter 2.

Added **xrt_teem_ch.pro** to Section 2.11.

Added subsection: Using **xrt_dem_iterative2.pro** to Chapter 2.

May 2017 Revision:

Added section about the Point Spread Function.

Added link to the XRT Level 0 MPEG archive in Table 2.1.

October 2016 Revision:

Added information about the 14 June 2015 light leak to Section 2.4.4 *Light Leak*.

Updated the XRT paper references in Table 1.2.

Corrected the fits keyword definitions of `CRPIX1` and `CRPIX2` in the Appendix.

Previous revisions by:

Patrick McCauley, Paola Testa, and P. R. Jibben

Originally written by:

Monica Bobra & Mark Weber
and other members of the XRT team

Contents

1 Introduction **9**
- 1.1 Hinode and the X-Ray Telescope . . . 9
  - 1.1.1 Synoptic Data . . . 9
  - 1.1.2 Eclipse Season . . . 10
- 1.2 Overview of the Data Analysis Pipeline . . . 10
  - 1.2.1 Data Transport . . . 10
  - 1.2.2 Data Products . . . 11
- 1.3 References . . . 12
- 1.4 Contacts . . . 12

2 X-Ray Telescope Software Guide **13**
- 2.1 Browsing XRT Data . . . 13
- 2.2 Obtaining XRT Data . . . 13
  - 2.2.1 Darts/Hinode . . . 13
  - 2.2.2 Virtual Solar Observatory . . . 14
  - 2.2.3 Hinode Science Data Center Europe Archive . . . 18
  - 2.2.4 LMSAL Sungate Website . . . 21
- 2.3 Searching XRT Data with the XRT Catalog . . . 23
  - 2.3.1 Using **xrt_cat.pro** . . . 23
  - 2.3.2 Search Criteria Used to Select XRT Data . . . 24
- 2.4 Reading XRT Data . . . 25
  - 2.4.1 Using **read_xrt.pro** . . . 25
- 2.5 Calibrating XRT Data . . . 26
  - 2.5.1 Using **xrt_prep.pro** . . . 26
  - 2.5.2 Routines Used by **xrt_prep.pro** . . . 32
  - 2.5.3 Contamination . . . 36
  - 2.5.4 Light Leak Overview . . . 39
  - 2.5.5 Light Leak Correction . . . 40
- 2.6 Displaying XRT Data . . . 41
- 2.7 Coaligning XRT Data . . . 42
  - 2.7.1 Using **xrt_read_coaldb.pro** . . . 42
  - 2.7.2 Using **xrt_jitter.pro** . . . 45
- 2.8 Making Composite Images with XRT Data . . . 46

2.9 Making Movies with XRT Data 49
2.10 Writing XRT Data 49
2.10.1 Using **write_xrt.pro** 49
2.10.2 Using **write_png.pro** or **write_bmp.pro**: 50
2.11 Instrument Responses and Inferring Physical Quantities 51
2.11.1 Filter Responses 55
2.11.2 Calculating XRT Filter Response with Non-Standard Spectra 62
2.11.3 Using **xrt_teem.pro** or **xrt_teem_ch.pro** 66
2.11.4 Using **xrt_dem_iterative2.pro** 73
2.12 The XRT Point-Spread Function 79
2.12.1 Using **xrt_deconvolve.pro** 80
2.13 Known Issues 80

3 X-Ray Telescope Instrument Guide **82**
3.1 XRT System Overview 82
3.2 Telescope Performance 82
3.3 MDP/XRT Communications 83
3.4 CCD Camera System 83
3.4.1 Pointing 84
3.4.2 Exposures 85
3.5 XRT Mechanisms 85
3.5.1 Visible Light Shutter 86
3.5.2 Visible Light Imager 86
3.5.3 Focus Mechanism 86
3.5.4 Filter Wheel and Shutter Assembly 86

A Level 0 Header Keywords **88**

List of Figures

1.1 Hinode eclipse season. 10

2.1 The Virtual Solar Observatory (VSO) main menu. 15
2.2 Search form generated after selecting the **Instrument / Source / Provider** box on the main VSO website. Hinode XRT data is provided by SAO. . . . 16
2.3 VSO form describing the data available given the search parameters. The **CheckBox Tools** menu in the left column can be helpful when selecting a large amount of data. 17
2.4 VSO Data request page. 18
2.5 Hinode Science Data Center Europe Archive Search page. Search parameters relevant to XRT are highlighted green in this image. 19
2.6 Example search results from the Hinode SDC Europe Archive. 20
2.7 Lockheed Martin Solar & Astrophysics Laboratory's Sungate website. . . . 21
2.8 LMSAL Solarsoft XRT catalog webpage. 22
2.9 Effect of contamination layer, accumulating on the CCD between a bakeout and the next, on the XRT temperature response of Al_mesh (black curves), C_poly (red curves), and Be_thin (blue curves). The solid lines are responses calculated for 2009-09-24, right after a CCD bakeout, and the dashed curves are calculated for 2009-10-15, i.e. three weeks later and just before the following bakeout. The comparison shows that in the regime of regular bakeouts, adopted since mid-2008, only the thinnest filter shows any detectable, though minimal, change in response due to accumulation of contaminating material on the CCD between bakeouts. 37
2.10 An example Al_mesh image taken on 5-August-2010 at 10:01UT. *Left*: The prepped data without spot correction. *Right*: The output image from the routine **xrt_spotcor.pro**. The severity of the spots depends on wavelength with the Al_mesh filter being most affected. Spot correction can be done from within **xrt_prep.pro** and a spot map should be generated to locate affected pixels. 38
2.11 Comparison of Ti_poly images before and after light leak. 40
2.12 Examples displaying XRT data in IDL. 42

2.13 Sample coalignment between XRT and AIA images using **plot_map**. XRT images are on the left and AIA on the right. *Upper Left*: XRT Map using level-0 FITS header information without correction. *Lower Left*: Same data after the correction done by xrt_read_coaldb.pro. The image location has changed relative to the *x* and *y* axes. The red contour in the right panel show the position of the flaring loop in XRT images. 46
2.14 The total telescope throughput of the XRT for each of the nine X-Ray filter channels. *Figure 17 from Golub et al. (2007).* 54
2.15 Example of XRT temperature responses calculated for two different dates. The solid lines are responses calculated for 2007-03-01, before the first CCD bakeout, and the dashed curves are calculated for 2008-03-01, in the regime of regular bakeouts. The comparison shows how the sensitivity in the lower energy range, significantly decreased by the contamination material, has been recovered through CCD bakeout and maintained with regular CCD bakeouts. 62
2.16 Example of XRT temperature responses calculated using the default spectrum for AIA and a standard XRT spectrum: the solid lines are the responses calculated for the standard XRT spectrum, and the dashed curves are calculated using the default spectrum for AIA. Both are calculated for the date 2011 August 21. 67
2.17 Example temperature output from **xrt_teem.pro** using Al_mesh and Al_poly filters. The units have been changed from log K to Million K. 70
2.18 Example emission measure output from **xrt_teem.pro** using the Al_mesh and Al_poly filters. 71
2.19 Example output from **xrt_deconvolve** with and without using a saturation mask. 81

3.1 XRT filter wheels as viewed from the sun. 87

List of Tables

1.1 **XRT Data Types** 11
1.2 **XRT References** 12

2.1 **XRT Data Browsing Resources** 14
2.2 **Summary of Stray Light Events and Links to Analysis Webpages** 40

3.1 **X-Ray Performance** 82
3.2 **Optical Performance** 83

List of URLs

http://adsabs.harvard.edu/abs/1992PhyS...46..202F 95
http://adsabs.harvard.edu/abs/2007PASJ...59S.845S 95
http://adsabs.harvard.edu/abs/2007SoPh..243...63G 95
http://adsabs.harvard.edu/abs/2008SoPh..249..263K 95
http://adsabs.harvard.edu/abs/2011SoPh..269..169N 95
http://adsabs.harvard.edu/abs/2014SoPh..289.1029N 95
http://adsabs.harvard.edu/abs/2014SoPh..289.2781K 95
http://adsabs.harvard.edu/abs/2016AJ....152..107A 95
http://adsabs.harvard.edu/abs/2016SoPh..291..317T 95
http://fits.gsfc.nasa.gov 24
http://hinode.nao.ac.jp/en/for-researchers/qlmovies/ 14
http://hinode.nao.ac.jp/en/gallery/latest/ 13
http://sdac.virtualsolar.org/cgi-bin/search 14
http://sdc.uio.no/search/ 18
http://solar.physics.montana.edu/HINODE/XRT/SCIA/latest_month.html 10, 14
http://solar.physics.montana.edu/HINODE/XRT/lev0_mma/ 14
http://solar.physics.montana.edu/HINODE/XRT/xrt_contam/index.html 36
http://solar.physics.montana.edu/takeda/xrt_sl2015/xrt15_sl2.html 40
http://solar.physics.montana.edu/takeda/xrt_sl2017/xrt17_sl3.html 40
http://solar.physics.montana.edu/takeda/xrt_sl2018/xrt18_sl4.html 40
http://solar.physics.montana.edu/takeda/xrt_straylight/xrt_sl_summary.html 40
http://www.lmsal.com/solarsoft/ 13
http://www.lmsal.com/sungate/ 21
http://xrt.cfa.harvard.edu 12, 14
http://xrt.cfa.harvard.edu/focus_catalog/ 14
http://xrt.cfa.harvard.edu/missionops/snapview/snapview.html . . 14
http://xrt.cfa.harvard.edu/resources/documents/XAG/XAG.pdf 74
http://xrt.cfa.harvard.edu/xpow/ 14
http://ylstone.physics.montana.edu/yosimura/hinode/coalignment/ . 43
http://ylstone.physics.montana.edu/yosimura/hinode/coalignment/samples/ 45
http:/xrt.cfa.harvard.edu/flare_catalog/ 14
https://darts.isas.jaxa.jp/solar/hinode/ 13
https://helioviewer.org/ 14
https://hinode.isee.nagoya-u.ac.jp/thesis.input/ 12
https://www.lmsal.com/sdodocs/doc/dcur/SDOD0060.zip/zip/entry/index.html 64

Introduction

This document consists of a Software Guide (Chapter 2) and Instrument Guide (Chapter 3). The Software Guide describes how to analyze XRT data; the Instrument Guide gives a broad overview of the X-Ray Telescope's hardware components.

1.1 Hinode and the X-Ray Telescope

Hinode is a joint mission between the space agencies of Japan, United States, Europe, and United Kingdom. The craft carries three instruments, a Solar Optical Telescope (SOT), Extreme Ultraviolet Imaging Spectrometer (EIS) and X-Ray Telescope (XRT); together, they are designed to provide multi-wavelength data from the photosphere to the upper corona. The 875-kg craft was launched on September 23, 2006 into a polar, sun-synchronous orbit at 600 kilometers with an inclination of -98°, allowing 9 months of continuous observations and a 3-month eclipse season. Hinode provides approximately 7 GB of data daily.

The XRT images coronal plasmas from 1 to approximately 20 million K with 2″ resolution (≈ 1″ pixels). The XRT images through nine X-ray filters using two filter wheels. The XRT also contains a visible light optic that takes G-band images for alignment. The back-thinned CCD has 2048×2048 pixels and images over a 35′× 35′ field of view, though partial frame images of various sizes can be read from select areas of the CCD. XRT takes approximately 0.7 GB of data daily as lossless or one of nine forms of lossy JPEG compression.

The baseline duration of the mission is 3 years. Currently XRT, SOT and EIS are operated from the Institute of Space and Astronautical Science (ISAS) in Sagamihara, Japan. See the Instrument Guide (Chapter 3) for more information.

1.1.1 Synoptic Data

The Hinode craft points at sun center for 10 minutes 2-4 times a day, enabling XRT to take synoptic data in the form of full-resolution, full-disk images with both long and short exposures in one or more X-ray filters. The long and short exposures are taken to capture the dynamic range of the coronal plasma's X-ray emission and thus the saturated pixels from the long exposure are replaced with corresponding pixels in the short exposure pair. The pair is then available as one image in a form referred to as Level 2 data (See Section 1.2.2).

Synoptic Level 2 data is available at:
http://solar.physics.montana.edu/HINODE/XRT/SCIA/latest_month.html.

1.1.2 Eclipse Season

The Hinode craft is in a polar, sun-synchronous orbit at 600 kilometers with an inclination of -98°, allowing 9 months of continuous observations and a 3-month eclipse season. The eclipse season begins in May every year; eclipse durations are shown in Figure 1.1. Effects of atmospheric absorption will be visible in XRT images beginning ≈ 4 days before the season through to ≈ 4 days after the season, for a total of ≈ 100 days.

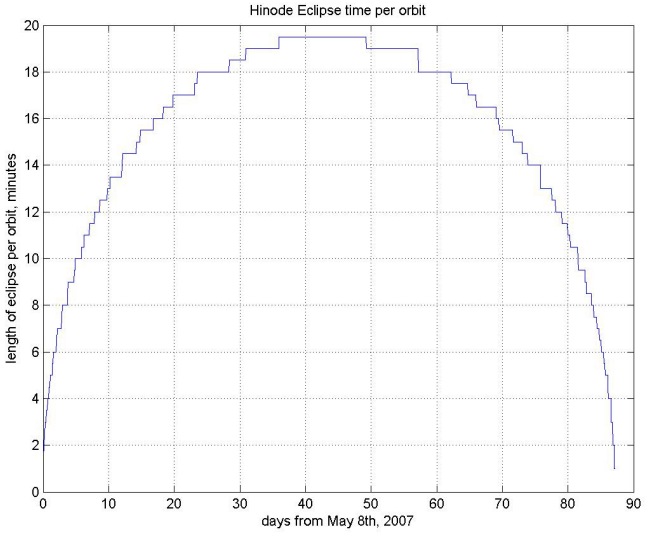


Figure 1.1: Hinode eclipse season.

1.2 Overview of the Data Analysis Pipeline

1.2.1 Data Transport

XRT downlinks 15 times daily at the Svalbard Ground Station, a member of the Norwegian Space Centre (NSC). Downlinks also occur up to four additional times daily at any one of the Japanese Aerospace Exploration Agency (JAXA) ground station network antennas, which include two Japanese sites, in the Kagoshima and Ibaraki prefectures, as well as three overseas sites: Maspaolomas, Canary Islands, Perth, Australia, and Santiago, Chile. Downlinked XRT telemetry is sent to the ISAS mission archive, the principal XRT data site, for reformatting to create Level 0 data. Both QuickLook and Level 0 data are then mirrored to the Smithsonian Astrophysical Observatory (SAO), in Cambridge, MA. Using the data retrieval techniques described in Section 2.2, users access machines at SAO to obtain XRT data.

1.2.2 Data Products

XRT data is available as FITS files, each of which include a data array and a metadata structure array containing a list of keywords (see Appendix 1). Several types of data products are available: QuickLook, Level 0, Level 1, and Level 2.

QuickLook data are expedited to ISAS so XRT operation team members can view images a few hours after the data has been taken. However, these data have not been completely reformatted and thus the images may not be whole and the FITS keywords will not be populated completely or correctly. XRT team members use this data to make operations decisions but these data products are not suitable for scientific purposes.

Level 0 data contains whole images; in addition, the FITS keywords have been populated correctly by the reformatter. Level 0 data cannot be created until all the housekeeping data for a particular observation has arrived at ISAS, which may take up to 7 days. Level 1 data has been calibrated by **xrt_prep.pro** and has units of instrumental Data Numbers. Level 2 data has been further processed into more physical units or into movies. The following table describes each of the XRT data products:

Table 1.1: **XRT Data Types**

Level	**Pixel values**	**File format**	**Purpose**
QuickLook	Data Number (DN)	FITS	Operations, Data Verification, QuickLook movies
0	DN	FITS	Basic science
1	DN/sec	FITS	Calibrated images
2	Physical units	Any	Short and long exposures summed into one image; Differential Emission Measure; Temperature Maps

XRT instrument data are available as single Level 0 FITS files with names in the the format **XRTYYYYMMDD_HHMMSS.S.fits**. The files stored in directories organized by hour beneath directories organized by day, month, and year. An example of a directory structure to access an individual FITS file is as follows:

YYYY/MM/DD/Hhh00/, or **2007/05/27/H1300**.

1.3 References

Table 1.2: **XRT References**

Instrument paper	Golub et al. 2007
Camera paper	Kano et al. 2008
Calibration paper	Kobelski et al. 2014
Measuring uncertainties	Kobelski et al. 2014
Temperature Response	Narukage et al. 2011 & Narukage et al. 2014
Synoptic Images	Takeda et al. 2016

XRT images in the popular media should be credited to SAO/NASA/JAXA/NAOJ. Please check the current Hinode data thesis projects webpage prior to publishing results.
https://hinode.isee.nagoya-u.ac.jp/thesis.input/

For more information about Hinode XRT visit the website:
http://xrt.cfa.harvard.edu.

1.4 Contacts

For data analysis problems, contact xrt_manager[at]cfa.harvard.edu.
For science discussions, contact xrt_science[at]cfa.harvard.edu.

X-Ray Telescope Software Guide

This chapter outlines how to analyze XRT data using software publicly available as part of SSWIDL (to add an instrument path to a SSWIDL tree, see http://www.lmsal.com/solarsoft/). This process involves obtaining, searching, reading, calibrating, coaligning and writing XRT data, as well as constructing Level 2 data products such as long-short composite images and various temperature analyses.

2.1 Browsing XRT Data

There are several resources available for browsing XRT data, all of which can be accessed at the XRT webpage. Table 2.1 provides links to the resource along with a brief description of what is available there.

The XRT data Snapview webpage presents a visual representation of XRT data that includes pointing and filter information and links to the VSO website. The XRT Synoptic Gallery provides daily full-disk solar images and the XRT flare catalog details every flare observed by XRT and links to flare data (VSO website). The Level 0 MPEG movie archive provides MPEG movies of the XRT Level 0 data. Users can quickly check data availability and quality before accessing the archive. New tools and catalogs are always being developed and users are encouraged to visit the XRT webpage to find the latest data products available.

2.2 Obtaining XRT Data

XRT data can be obtained from several outlets including the Darts/Hinode (ISAS, JAXA), the Virtual Solar Observatory (VSO), the Hinode Science Data Center Europe Archive, and the Lockheed Martin Solar & Astrophysics Laboratory (LMSAL) Sungate website.

2.2.1 Darts/Hinode

The DARTS/Hinode website is at https://darts.isas.jaxa.jp/solar/hinode/. Here users can find data and information for all three Hinode instruments, including XRT. Data can be downloaded from this website. A very useful feature at this website is the Hinode Latest Images.

Table 2.1: **XRT Data Browsing Resources**

XRT website	http://xrt.cfa.harvard.edu
Helioviewer	https://helioviewer.org/
XRT Synoptics Gallery	http://solar.physics.montana.edu/HINODE/XRT/SCIA/latest_month.html
XRT L0 MPEG Archive	http://solar.physics.montana.edu/HINODE/XRT/lev0_mma/
Hinode QL Movie Archive	http://hinode.nao.ac.jp/en/for-researchers/qlmovies/
XRT Flare Catalog	http:/xrt.cfa.harvard.edu/flare_catalog/
XRT Focus Catalog	http://xrt.cfa.harvard.edu/focus_catalog/
Snapview website	http://xrt.cfa.harvard.edu/missionops/snapview/snapview.html
XRT Picture of the Week	http://xrt.cfa.harvard.edu/xpow/

2.2.2 Virtual Solar Observatory

The VSO website, Figure 2.1, is at: http://sdac.virtualsolar.org/cgi-bin/search and allows users to search for solar data from several observatories. Users can select one or more ways to search for solar data by checking off the boxes they want and clicking the **Generate VSO Search Form** button located at the bottom of the screen. Currently, the VSO offers Level 1 XRT data as defined in Table 1.1. XRT data can be found in a variety of ways using this search form. If you are interested in searching XRT data based on the **Observable** option then you can check the **intensity** box after you click on the **Generate VSO Search Form** button. The **Instrument/Source/Provider** box creates a form to search for data based on an instrument or data archive. The other categories of interest to XRT data users are **Spectral Range** (soft X-Rays [1 - 100Å]), and **Nickname** (Soft X-Ray image).

The direct way of finding XRT data is by searching by **Instrument/Source/Provider**. This creates a list of solar data currently a member of the VSO. The VSO generated search form is displayed in Figure 2.2. Once the time and instrument have been selected, choose the **Search** button underneath the date. HINT: To change the date, change both years first, then both months, then both days. Otherwise the dates will automatically readjust.

Data for the dates of interest will appear in a form as seen in Figure 2.3. The left menu

bar provides users with several ways to select the data. A user can select data by checking the box next to the desired entry or by using the **CheckBox Tools** menu where data can be selected above or below a selected box. Once all of the data has been selected click on the **Request Data** button in the left menu.

Users are then directed to a page where they can select how to receive the data. Data can be downloaded via a URL or by creating a .tar file that will place the requested files in a staging area (see Figure 2.4). When selecting the tar file option, an email address may be required.

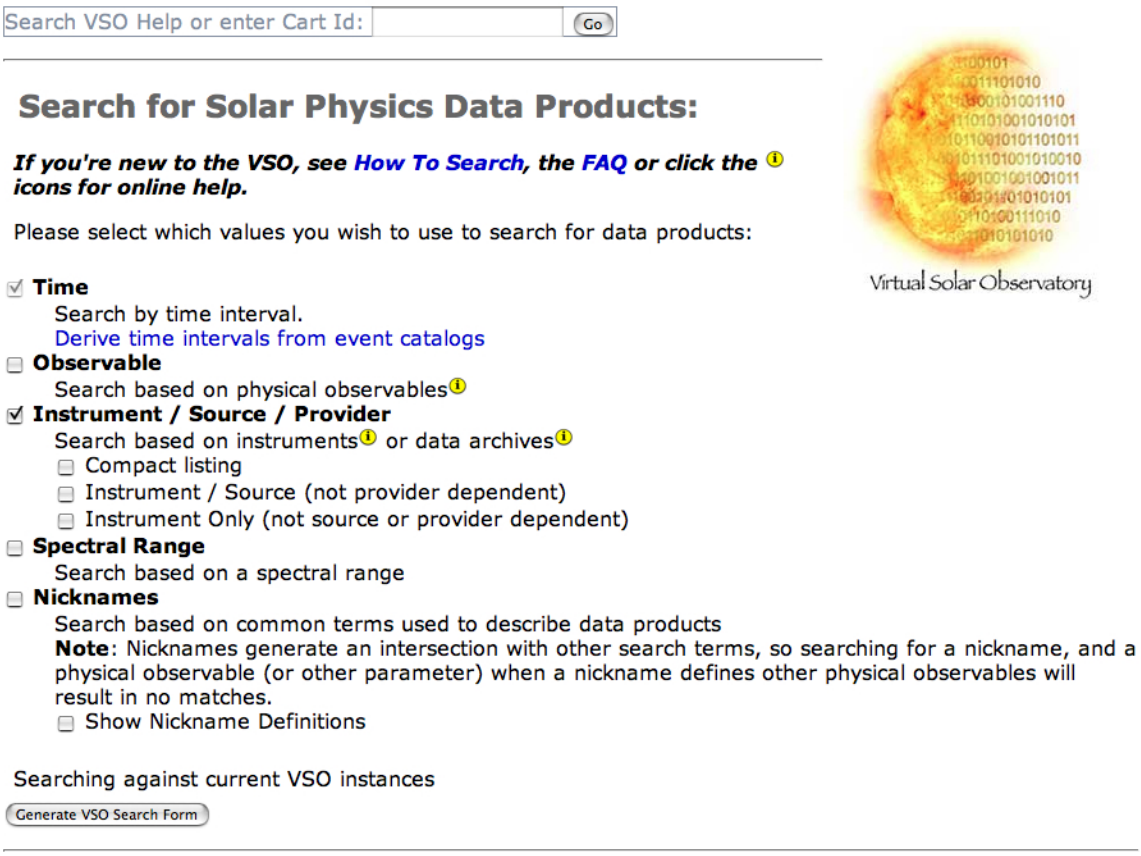
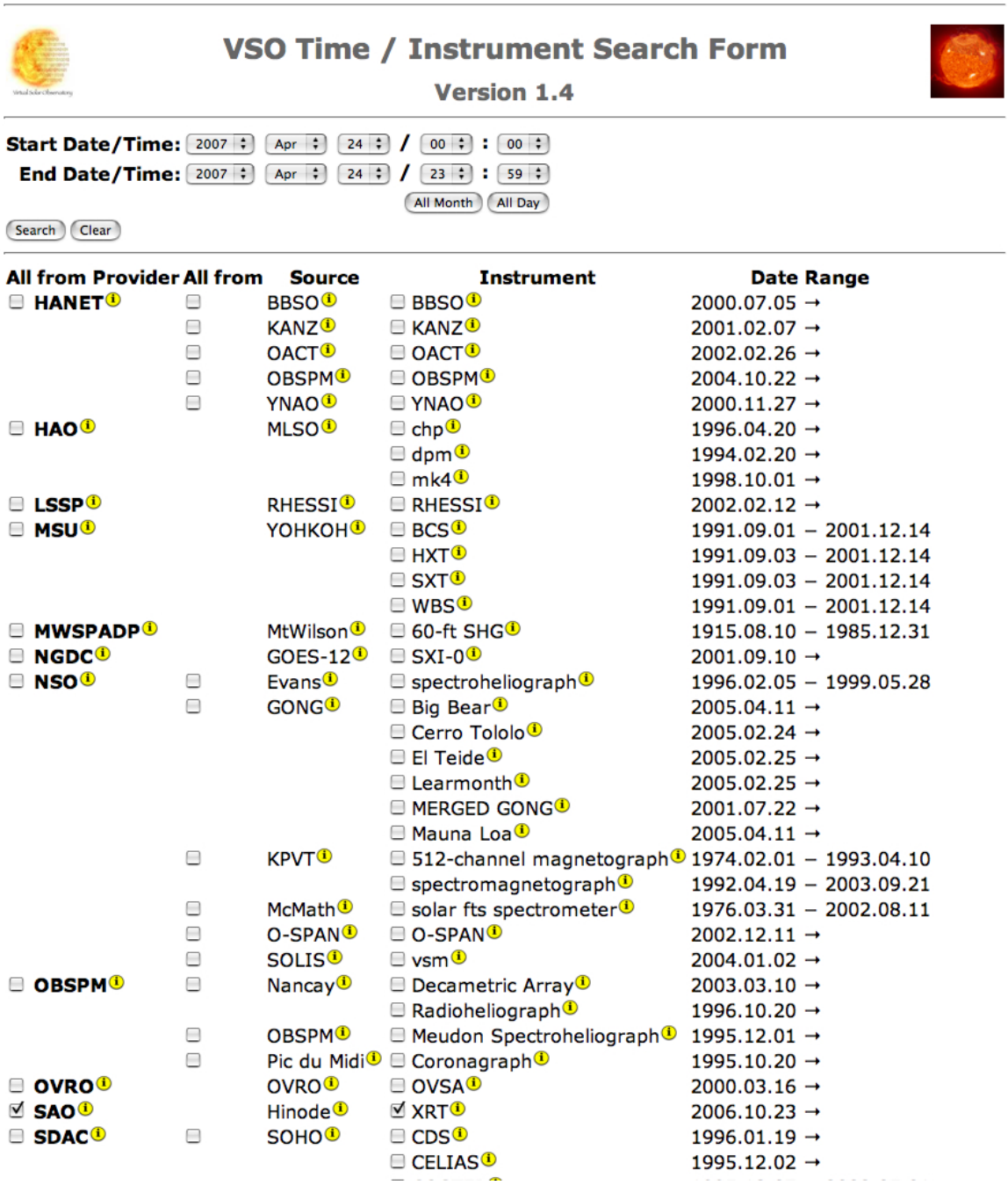


Figure 2.1: The Virtual Solar Observatory (VSO) main menu.



VSO Time / Instrument Search Form

Version 1.4

Start Date/Time: 2007 Apr 24 / 00 : 00

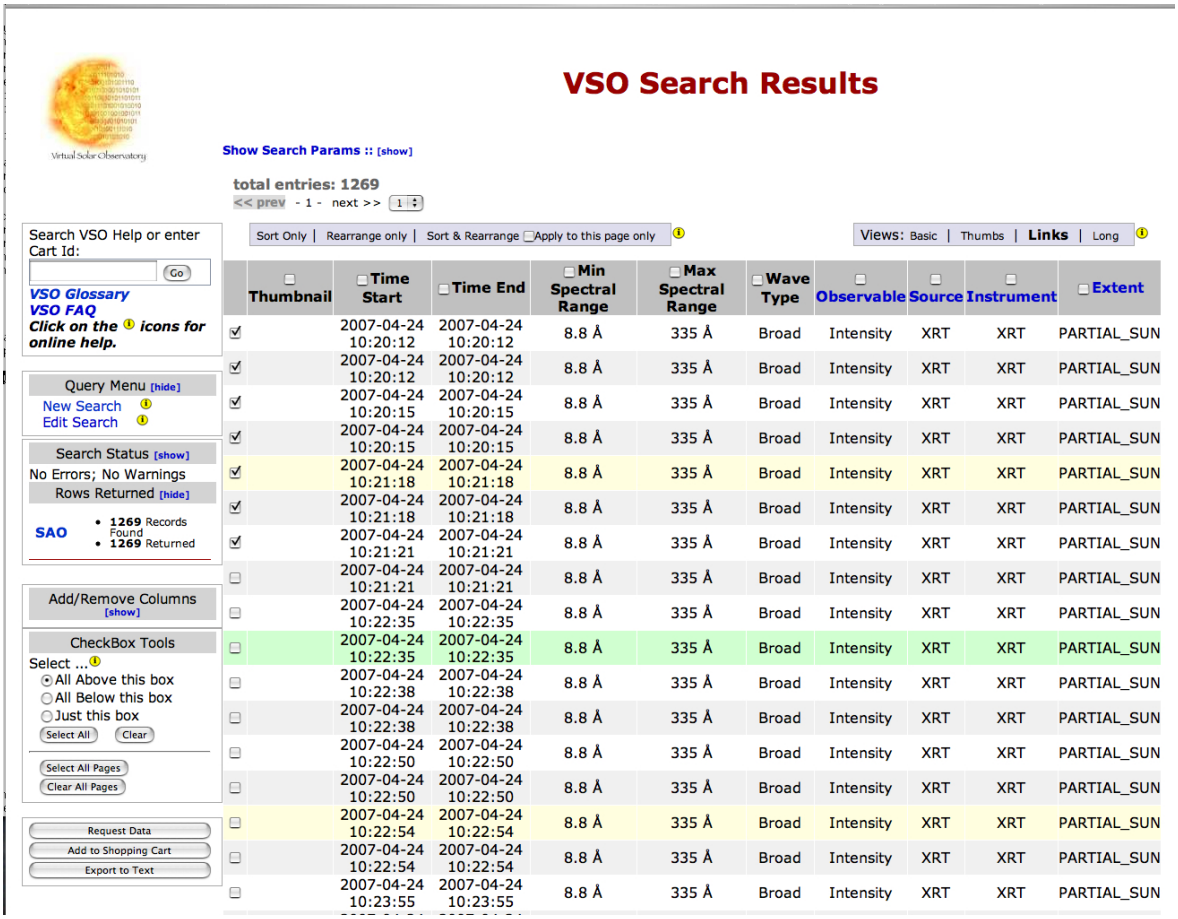
End Date/Time: 2007 Apr 24 / 23 : 59

All Month All Day

Search Clear

All from Provider	All from Source	Source	Instrument	Date Range
☐ HANET	☐	BBSO	☐ BBSO	2000.07.05 →
	☐	KANZ	☐ KANZ	2001.02.07 →
	☐	OACT	☐ OACT	2002.02.26 →
	☐	OBSPM	☐ OBSPM	2004.10.22 →
	☐	YNAO	☐ YNAO	2000.11.27 →
☐ HAO		MLSO	☐ chp	1996.04.20 →
			☐ dpm	1994.02.20 →
			☐ mk4	1998.10.01 →
☐ LSSP		RHESSI	☐ RHESSI	2002.02.12 →
☐ MSU		YOHKOH	☐ BCS	1991.09.01 – 2001.12.14
			☐ HXT	1991.09.03 – 2001.12.14
			☐ SXT	1991.09.03 – 2001.12.14
			☐ WBS	1991.09.01 – 2001.12.14
☐ MWSPADP		MtWilson	☐ 60-ft SHG	1915.08.10 – 1985.12.31
☐ NGDC		GOES-12	☐ SXI-0	2001.09.10 →
☐ NSO	☐	Evans	☐ spectroheliograph	1996.02.05 – 1999.05.28
	☐	GONG	☐ Big Bear	2005.04.11 →
			☐ Cerro Tololo	2005.02.24 →
			☐ El Teide	2005.02.25 →
			☐ Learmonth	2005.02.25 →
			☐ MERGED GONG	2001.07.22 →
			☐ Mauna Loa	2005.04.11 →
	☐	KPVT	☐ 512-channel magnetograph	1974.02.01 – 1993.04.10
			☐ spectromagnetograph	1992.04.19 – 2003.09.21
	☐	McMath	☐ solar fts spectrometer	1976.03.31 – 2002.08.11
	☐	O-SPAN	☐ O-SPAN	2002.12.11 →
	☐	SOLIS	☐ vsm	2004.01.02 →
☐ OBSPM	☐	Nancay	☐ Decametric Array	2003.03.10 →
			☐ Radioheliograph	1996.10.20 →
	☐	OBSPM	☐ Meudon Spectroheliograph	1995.12.01 →
	☐	Pic du Midi	☐ Coronagraph	1995.10.20 →
☐ OVRO		OVRO	☐ OVSA	2000.03.16 →
☑ SAO		Hinode	☑ XRT	2006.10.23 →
☐ SDAC	☐	SOHO	☐ CDS	1996.01.19 →
			☐ CELIAS	1995.12.02 →

Figure 2.2: Search form generated after selecting the **Instrument / Source / Provider** box on the main VSO website. Hinode XRT data is provided by SAO.



VSO Search Results

Show Search Params :: [show]

total entries: 1269

<< prev - 1 - next >> 1

Search VSO Help or enter Cart Id: Go

VSO Glossary
VSO FAQ
Click on the icons for online help.

Query Menu [hide]
New Search
Edit Search

Search Status [show]
No Errors; No Warnings

Rows Returned [hide]
SAO
- 1269 Records Found
- 1269 Returned

Add/Remove Columns [show]

CheckBox Tools
Select ...
All Above this box
All Below this box
Just this box
Select All Clear
Select All Pages
Clear All Pages

Request Data
Add to Shopping Cart
Export to Text

Sort Only | Rearrange only | Sort & Rearrange Apply to this page only

Views: Basic | Thumbs | Links | Long

Thumbnail	Time Start	Time End	Min Spectral Range	Max Spectral Range	Wave Type	Observable	Source	Instrument	Extent
	2007-04-24 10:20:12	2007-04-24 10:20:12	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:20:12	2007-04-24 10:20:12	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:20:15	2007-04-24 10:20:15	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:20:15	2007-04-24 10:20:15	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:21:18	2007-04-24 10:21:18	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:21:18	2007-04-24 10:21:18	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:21:21	2007-04-24 10:21:21	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:21:21	2007-04-24 10:21:21	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:22:35	2007-04-24 10:22:35	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:22:35	2007-04-24 10:22:35	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:22:38	2007-04-24 10:22:38	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:22:38	2007-04-24 10:22:38	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:22:50	2007-04-24 10:22:50	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:22:50	2007-04-24 10:22:50	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:22:54	2007-04-24 10:22:54	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:22:54	2007-04-24 10:22:54	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN
	2007-04-24 10:23:55	2007-04-24 10:23:55	8.8 Å	335 Å	Broad	Intensity	XRT	XRT	PARTIAL_SUN

Figure 2.3: VSO form describing the data available given the search parameters. The **Check-Box Tools** menu in the left column can be helpful when selecting a large amount of data.

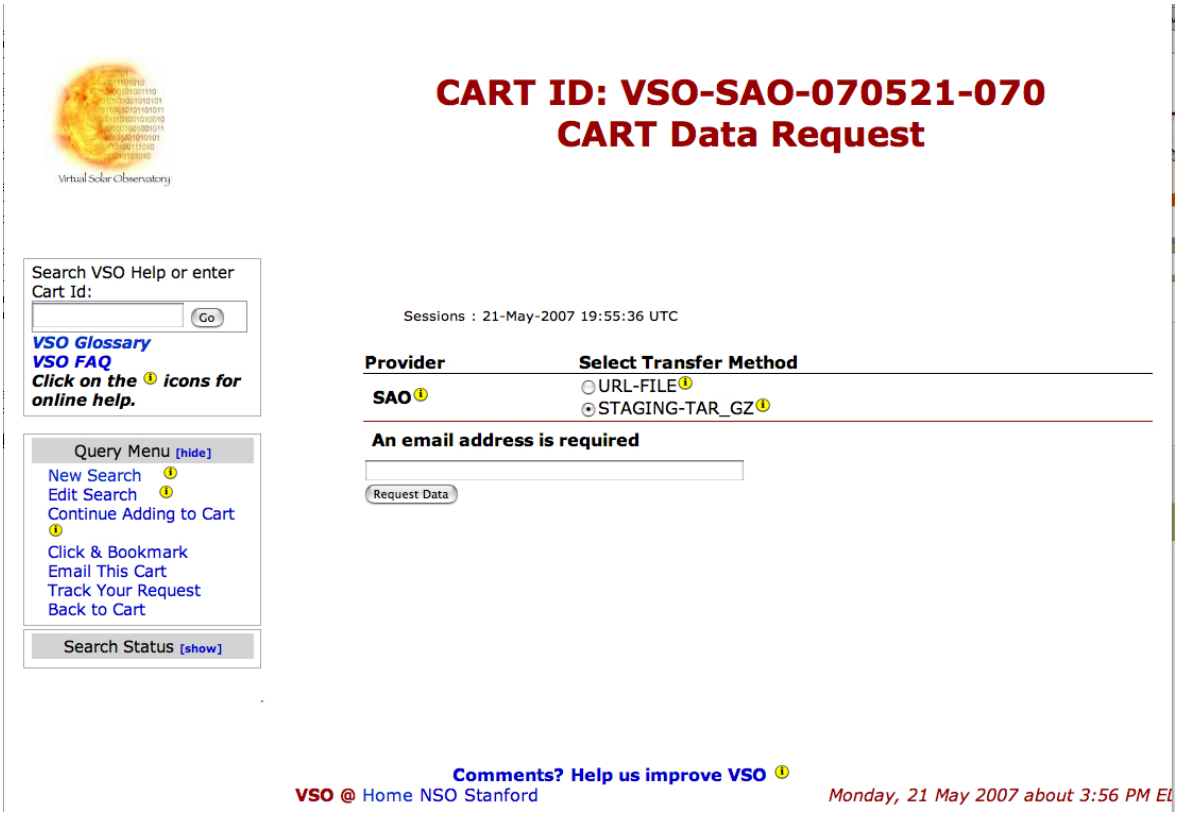


Figure 2.4: VSO Data request page.

2.2.3 Hinode Science Data Center Europe Archive

The Archive Search page at the Hinode SDC Europe, http://sdc.uio.no/search/, has data from Hinode's three instruments (XRT, EIS, and SOT). The main page is shown in Figure 2.5.

Figure 2.6 shows a typical results page output, which includes thumbnails (both a sample image from a dataset and its field of view superimposed on a context synoptic image). The 'Action' column contains several icons for selection of data to download. Once data are selected, they can be retrieved by clicking on the **Retrieve** button at the bottom of the page. You will need to enter an email address to which the system will send a notification and instructions on how to download the data once they are ready.

At the top of the webpage is a list of current instrument data in the archive. You can select XRT data here. In the **STATUS:** menu select 'Level 0' data to get unprocessed data. Finally, you must provide a range of dates by entering the date(s) in the EPOCH_START and EPOCH_END fields. Once the desired parameter values have been selected click the **Search** button.

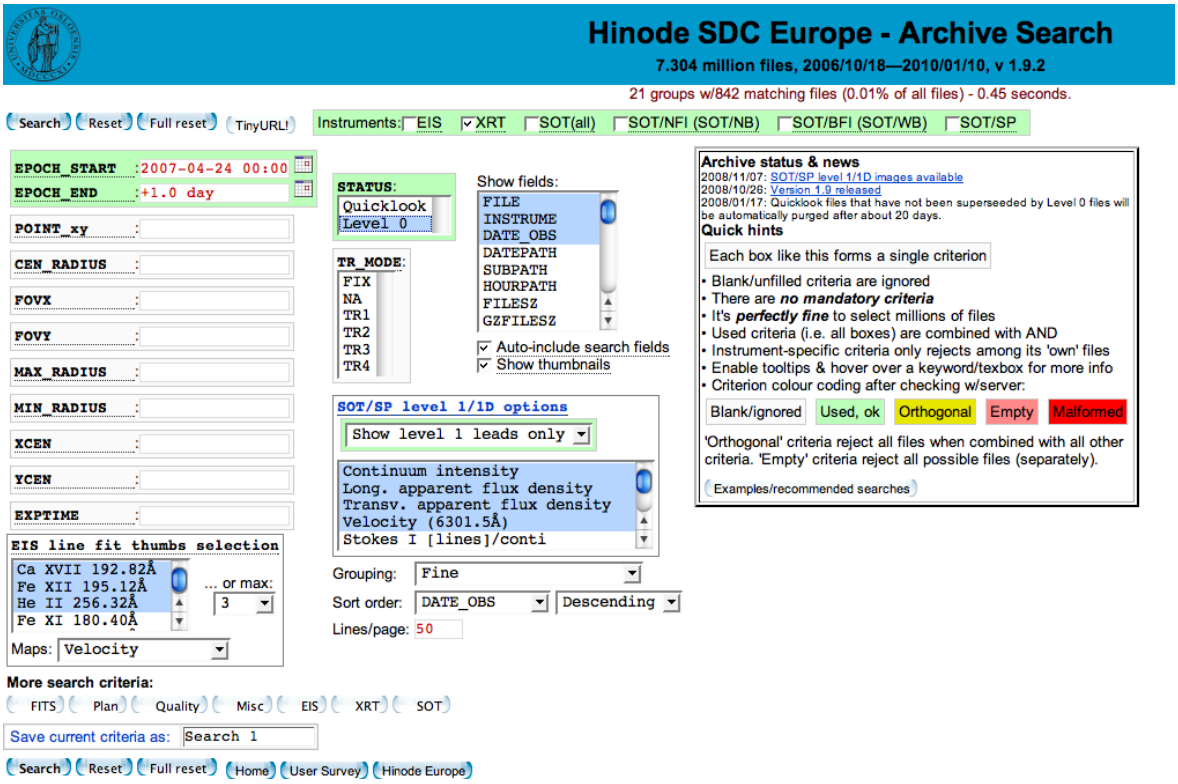


Figure 2.5: Hinode Science Data Center Europe Archive Search page. Search parameters relevant to XRT are highlighted green in this image.

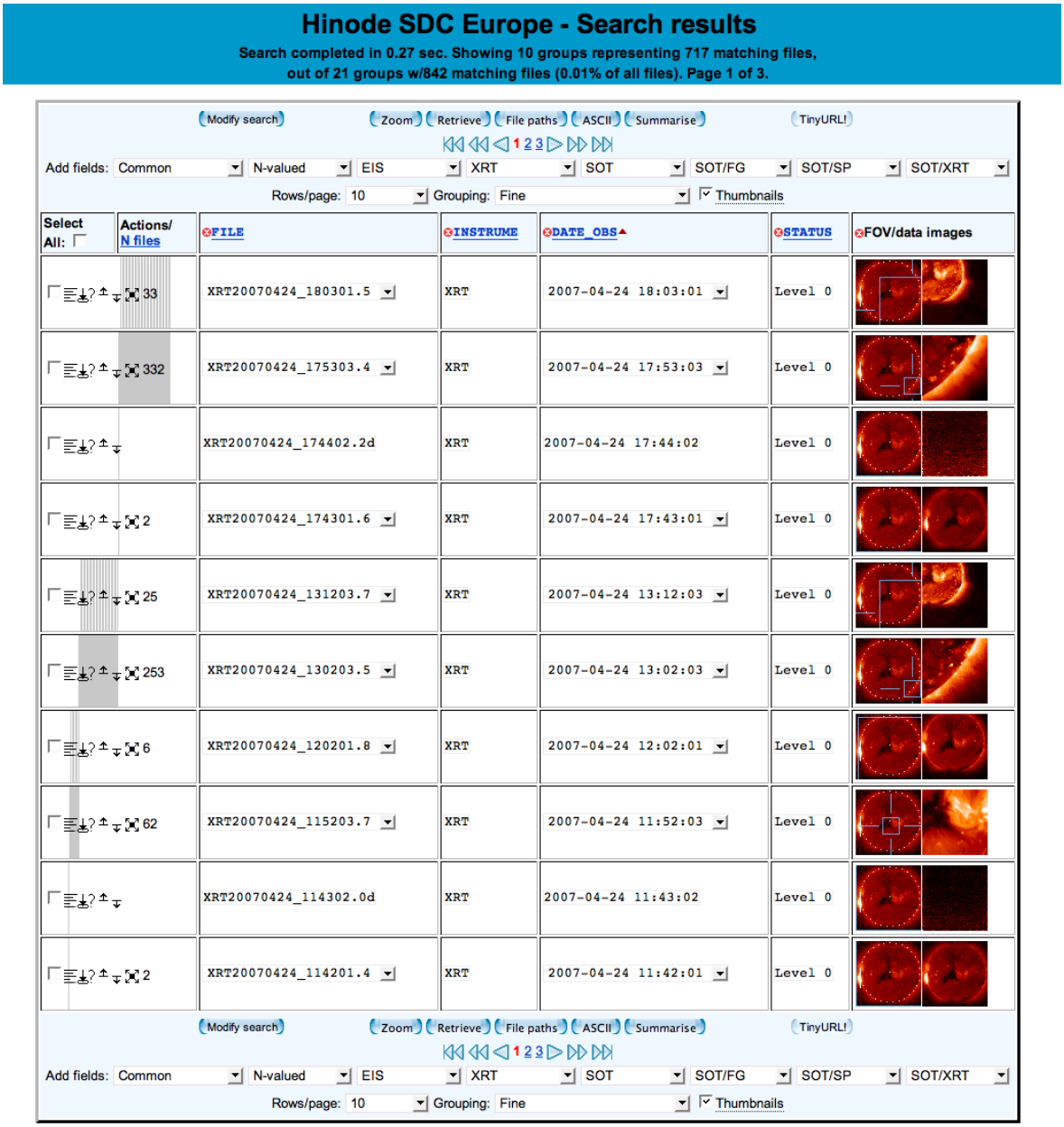


Figure 2.6: Example search results from the Hinode SDC Europe Archive.

2.2.4 LMSAL Sungate Website

Hinode XRT data may also be collected from the LMSAL Sungate webpage (Figure 2.7). http://www.lmsal.com/sungate/

There are multiple ways to search for XRT data. There is a direct link to the Hinode XRT catalog on this webpage as well as the Heliophysics Coverage Registry (HCR). Clicking the **Hinode XRT Archive** link will bring you to the search page shown in Figure 2.8.

The additional pages listed on the main page provide different routes to the data; each page is equipped with explanations.

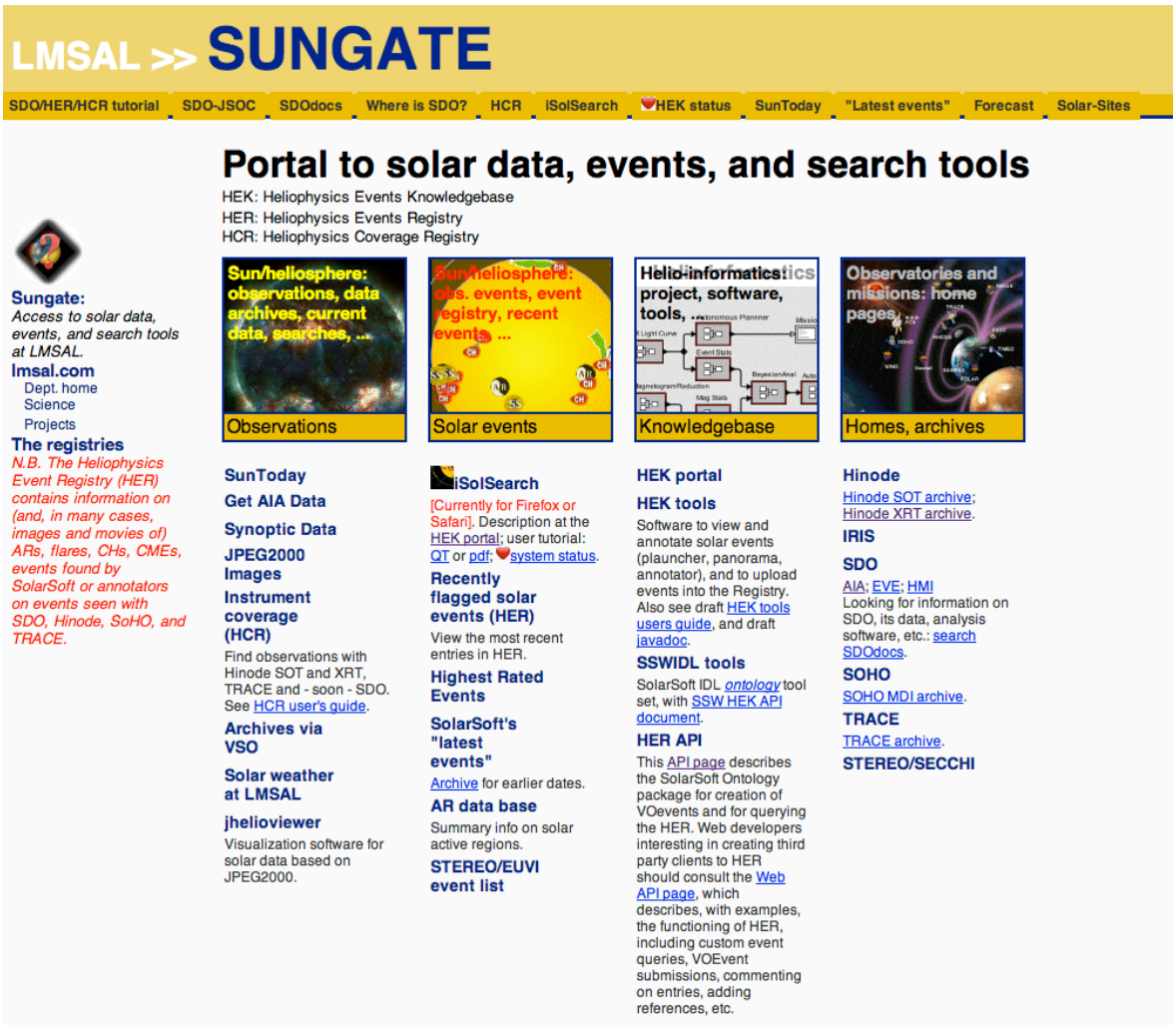


Figure 2.7: Lockheed Martin Solar & Astrophysics Laboratory's Sungate website.

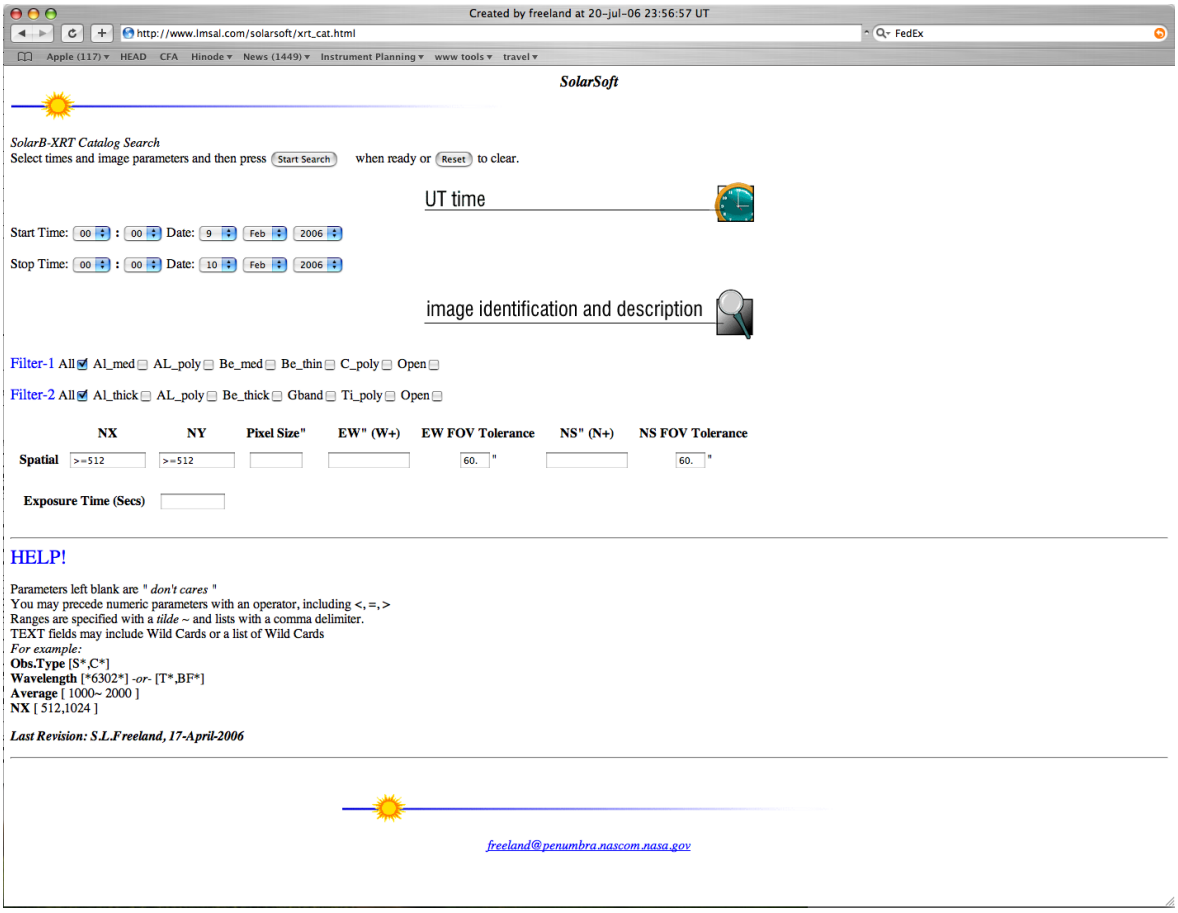


Figure 2.8: LMSAL Solarsoft XRT catalog webpage.

2.3 Searching XRT Data with the XRT Catalog

The XRT catalog contains a subset of XRT FITS header keywords, (such as field of view, filter positions, and image type), for every XRT observation. Using the catalog is a useful way to search data without downloading any of it. The catalog may be read and listed using the routine **xrt_cat.pro**.

In order to use this code, some ancillary XRT files are necessary and can be installed by running **sswdb_upgrade** from within SSWIDL and adding the **hinode/xrt** branch of the SSWDB tree.

2.3.1 Using xrt_cat.pro

This program returns XRT catalog records suitable for selecting data as a structure array. In the following example, the `OFILES` keyword generates a list of files using the subroutine **xrt_cat2files.pro**:

```
IDL> t0='2007-04-18T02:30:00'
IDL> t1='2007-04-18T12:30:00'
IDL> xrt_cat, t0, t1, catx, ofiles
IDL> help, catx
    CATX    STRUCT   = ->  <Anonymous> Array[311]
IDL> help, ofiles
    OFILES   STRING   = Array[311]

Or it can be run with a specific search array for selecting data:
IDL> sa = ['ec_fw1_= Be_thin', 'ec_fw2_ = Open', 'naxis1 = 1024',]
IDL> xrt_cat, t0, t1, catx, ofiles, /level0, /url, /verbose, search_array=sa
```

There are several input and output keywords that can be set to provide additional information.

Optional input keyword parameters:

`SEARCH_ARRAY`: (See **struct_where.pro**) Give a set of search parameters to further refine the data set.

`_EXTRA`: (Various possible types) Unknown parameters assumed to be in PARAM=VALUE pairs (see **struct_where.pro**).

`/QUICKLOOK`: (Boolean) If set, then return Quicklook data instead of Level 0 data. Overridden by `/level0`.

`/level0`: (Boolean) If set, then return Level 0 data. This is the default. Overrides `/QUICKLOOK`.

/level1: (Boolean) If set, then returns Level 1 data URLs. If set, it forces /level0 and /URLS to be set for the script to work, since it is really getting level0 URL links and replacing with the level1 version.

/URLS: (Boolean) If set, then return OFILES as URLs to a remote data server. (Default is to give local full paths.)

/REFRESH: (Boolean) If set, then refresh the cache (i.e., re-read the catalog).

/TEMP: (Boolean) If set, then use a temporary (i.e., offline/in-progress) catalog database.

/VERBOSE: (Boolean) If set, print out extra information. Overrides /QUIET. There are three levels of verbosity, in order of priority: (1) VERBOSE displays all errors and messages. (2) QUIET suppresses all errors and messages. (3) No keyword displays all errors and some messages.

/QUIET: (Boolean) If set, suppress messages.

/QSTOP: (Boolean) For debugging.

Optional output keyword parameters:

OFILES: (string array, [N_{img}]) List of full path filenames for the Level 0 data matching the catalog. If /URL, then instead returns data paths to a remote data server. This list is generated by a call to the program **xrt_cat2files.pro**.

ERROR: (scalar number) Returns the ERROR keyword value from a call to **read_genxcat.pro**.

QABORT: (Boolean) Indicates that the program exited gracefully without completing.

0: Program ran to completion.

1: Program aborted before completion.

2.3.2 Search Criteria Used to Select XRT Data

Further searching of XRT data can be done at the IDL command line using FITS keywords. All FITS values are in uninterpreted binary, with the exception that enumerated values are sometimes represented by string values instead of an integer. XRT follows with standard FITS file naming conventions, available at http://fits.gsfc.nasa.gov. A complete list of Level 0 XRT header keywords is provided in Appendix A. The following FITS file keywords may be of use.

EC_IMTY_: Image type; input string values of ‘dark’ or ‘normal’.

EC_FW1_: Filter Wheel 1 position; input string values of ‘Open’, ‘Al_poly’, ‘C_poly’, ‘Be_thin’, ‘Be_med’, ‘Al_med’

EC_FW2_: Filter Wheel 2 position; input string values of 'Open', 'Al_mesh', 'Ti_poly', 'Gband', 'Al_thick', 'Be_thick'

EC_VL_: Visible light shutter position; input string values of 'open' or 'closed'.

NAXIS1: General FITS keyword for pixel length in the *x*-direction; input integer value. XRT images are generally 384×384, 512×512 or 1024×1024; though any given area of the CCD can be read out.

NAXIS2: General FITS keyword for pixel length in the *y*-direction; input integer value.

Basic filter example:

```
IDL>ss=where(catx.ec_imty_ eq 'normal' and catx.naxis1 eq 512)
```

The following keywords are not useful for searching purposes, but are quite useful for interpretation purposes:

EXPTIME : This keyword represents the duration of the CCD exposure for normal images in seconds. EXPTIME is useful for sorting long/short pairs.

(DATE_OBS + DATE_END) / 2: Performing this calculation best represents the UT time the exposure was taken.

2.4 Reading XRT Data

2.4.1 Using read_xrt.pro

All XRT image headers and data can be read using the IDL routine **read_xrt.pro**, which calls the subroutine **mreadfits.pro**. Because XRT images are of various sizes, **read_xrt.pro** can read header and data information for images of various sizes using the keyword /FORCE. If the /FORCE keyword is used to read image data of various sizes, the data array will be given the maximum required dimensions. Smaller images will be embedded in the lowest corner of the respective 2D slice, with zero values in the buffer pixels. These small images will retain their original NAXIS1 and NAXIS2 keywords; for example, if a 512×512 image is embedded in a 2048×2048 array, the NAXIS1 and NAXIS2 keywords for that image will read 512 and 512, respectively. Continuing with the example in Section 2.3.

Basic call, to read headers and data:

```
IDL> ss=where(catx.ec_imty_ eq 'normal' and catx.naxis1 eq 2048)
IDL> read_xrt, ofiles[ss], index, data
IDL> help, ofiles[ss]
    <Expression>    STRING    = Array[5]
IDL> help, index
    INDEX           STRUCT    = ->   <Anonymous>  Array[5]
IDL> help, data
    DATA            INT       = Array[2048, 2048, 5]}
```

Basic call, to read headers only:
```
IDL> read_xrt, ofiles, index
```

To read headers and data for images of various sizes:
```
IDL> read_xrt, ofiles, index, data, /force
```

To suppress messages from mreadfits.pro:
```
IDL> read_xrt, ofiles, index, data, /force, /quiet
```

To display extra information:
```
IDL> read_xrt, ofiles, index, data, /force, /verbose
```

The `index.History` keyword is updated to reflect the origin file of the data:
```
IDL> print, index[0].History
    READ_XRT v2007-May-09: (22-May-2007 17:56:30)
    Read header only.
    Origin file: XRT20070418_111203.0.fits
```

2.5 Calibrating XRT Data

2.5.1 Using xrt_prep.pro

The routine **xrt_prep.pro** is similar in nature to **trace_prep.pro**, **eit_prep.pro** and **sxt_prep.pro**. The routine is intended to convert Level 0 data to Level 1 data. This process adds additional keywords to the Level 1 data FITS header.

It is important to note that if images of different sizes were read in using **read_xrt.pro**, the values for the `NAXIS1` and `NAXIS2` keywords are preserved. The data output from **xrt_prep.pro** defaults to 2*I for historical and memory conservation. Users are recommended to apply the `/FLOAT` keyword always unless they are constrained by available memory. The calibrations applied by **xrt_prep.pro** are discussed in the paper by Kobelski et al. (2014).

Basic call:
```
IDL> xrt_prep, input1, input2, index_out, data_out
```

Where `input1` and `input2` can take on one of two different values:

Case 1

Input 1: XRT FITS file list as a string scalar or array.
Input 2: The data set number(s) to extract and process.

Case 2

Input 1: The index structure for the file list as a structure array.

Input 2: The data array(s) as an integer array.

The basic calls for each case (*example data from Section 2.3*):

Case 1:

```
IDL> dset_arr = where(catx.ec_fw2_ eq 'Ti_poly' and $
     catx.naxis1 eq 2048 and catx.ec_imty_ eq 'normal')
IDL> xrt_prep, ofiles, dset_arr, index_out, data_out
IDL> help, ofiles
    OFILES     STRING   =    Array[311]
IDL> help, dset_arr
    DSET_ARR   LONG     =    Array[2]
IDL> help, index_out
    INDEX_OUT  STRUCT   =    -> <Anonymous> Array[2]
IDL> help, data_out
    DATA_OUT   INT      =    Array[2048, 2048, 2]
```

Case 2:

```
IDL> read_xrt, ofiles[dset_arr], index, data
IDL> xrt_prep, index, data, index_out, data_out, /float
IDL> help, index
    INDEX      STRUCT   =    -> <Anonymous> Array[2]
IDL> help, data
    DATA       INT      =    Array[2048, 2048, 2]
IDL> help, index_out
    INDEX_OUT  STRUCT   =    -> <Anonymous> Array[2]
IDL> help, data_out
    DATA_OUT   FLOAT    =    Array[2048, 2048, 2]}
```

The routine will output the updated index structure of the input images as a structure array $[N_{img}]$ and processed output images as an integer array $[N_x, N_y, N_{img}]$.
The executed steps (including possible options) are:

1. Read in raw FITS image(s) from a filelist or read in a datacube and structure.
2. Fill pixels of value = 0 (missing data) with a "missing data value" = -999.
3. Replace near-saturated pixels for values greater than some threshold (default: 2500 DN).
4. Option to remove radiation-belt/cosmic-ray hits and streaks.
5. Calibrate for read-out signals.
6. Locate missing pixels and replaces their values with a linear patch to improve Fourier filter performance.

7. Remove the CCD bias (pedestal), and dark current (using the subroutine **xrt_clean_ro.pro** which also calibrates the read-out signals).
8. Remove vignetting.
9. Option to normalize each image for exposure time.
10. Option to compute map of calibration uncertainties.
11. Option to cosmetically correct for contamination spots or dust.
12. Output the corrected image(s) in an updated structure and data cube.
13. Option to coalign XRT data using **xrt_read_coaldb.pro**.

Optional input keyword parameters:

`/NORMALIZE`: (Boolean) Set to normalize output image to DN per sec. The data output will not default to floating point. Use the `/FLOAT` keyword.

`/FLOAT`: (Boolean) Set if you want to return floating point (default is I*2). Users are recommended to always apply the `/FLOAT` keyword unless they are constrained by available memory.

`CLEAN_TYPE`: Type of Fourier cleaning to use (see subroutine **xrt_fourier_vacuum.pro** for details:

0 = prefilter on semi-fixed streaks, then remove remaining Fourier streaks, stars, and smudges [default].

1 = remove semi-fixed streaks only.

2 = remove Fourier streaks, stars, and smudges.

3 = no Fourier cleaning applied.

`BSUB_TYPE`: Type of (model) background subtraction to use:

0 = remove Nyquist ringing and large-scale background ramp [default].

1 = remove Nyquist ringing only.

2 = remove large-scale background "ramp" only.

3 = no (model) background subtraction applied.

Note: `DARK_TYPE`=1 resets `BSUB_TYPE`=3.

`DARK_TYPE`: Type of dark subtraction to use:

0 = if possible, use the model dark with median adjusted to median of cleaned darks nearby in time [default]. Otherwise, shift to `DARK_TYPE`=2.

1 = if possible, use the median of cleaned darks nearby in time; otherwise, as in the above case, it will shift to DARK_TYPE=2.

2 = use model darks without median adjustment (this corresponds to the old default of the **xrt_prep.pro** version prior to 'v2009-Jul-11').

Note: DARK_TYPE=1 resets BSUB_TYPE=3.

NSIGMA: Number of standard deviations beyond which an FFT pixel is considered bad in the Fourier Star removal (see **xrt_fourier_vacuum.pro** for details.) The default is 4.5. It is not recommended to go below 4.0.

NMED: Number of standard deviations above smoothed central minimal background to begin shielding (presumed) data from correction (see **xrt_fourier_vacuum.pro** for details). The default is 3.5. It is not recommended to go outside the range of 2.0 to 4.5.

/STOP_BLEED: (Boolean) Set if pixels corrected for saturation "bloom/bleed" at the edge of saturated region(s) are to be retained. The default is to revert the pixels to their input values.

/DESPIKE_DESPOT: (Boolean or integer) Set = N ($1 \leq N \leq 3$) for N passes to remove radiation belt and cosmic-ray spikes. This method uses convolution and thresholding to remove spikes and may remove small real features. It automatically makes a cosmetic correction to contamination spots (spline-based). See **xrt_tup_contam.pro** for an alternative, median-cap cosmetic correction. The default is not to remove spikes or spots. If set, it combines /ONLY_DESPIKE and /ONLY_DESPOT and overrides these too. These are considered cosmetic corrections and should not be used on data for quantitative analysis.

/ONLY_DESPIKE: (Boolean or integer) Set = N ($1 \leq N \leq 3$) for N passes of removal radiation belt/cosmic-ray spikes. This method uses convolution and thresholding to remove spikes and may remove small real features and/or reduce the overall sharpness of the image. The default is not to removes spikes from data. This is considered a cosmetic correction and should not be used on data for quantitative analysis.

/ONLY_DESPOT: (Boolean) Set to make a cosmetic correction to the contamination spots (spline-based). See **xrt_tup_contam.pro** for an alternative, median-cap cosmetic correction. The default is not to correct for spots. This is considered a cosmetic correction and affected pixels should not be used for quantitative analysis.

SENS_DESPIKE: (Float scaler) This number controls the aggressiveness of the despiking routine. Values in the range 1.1 to 1.5 work well. Values less than 1.0 are rejected for the default. Default = 1.4. See **xrt_despike2.pro** for more information.

SPOTTHRESH: (Float scaler) $0 \leq$ SPOTTHRESH ≤ 1, or -1. If value is between 0 and 1, the threshold fraction of a spotted binned pixel at which the binned pixel is treated as spotted [default=0.5]. Normally, the program only corrects spots if it determines they are sufficiently dark for speed. If = -1, force treatment of all spots regardless of how long it could take.

GRADE_TYPE: (Integer) Type of grade_map array to output. See /GRADE_MAP below, and **xrt_pixel_grade.pro**.

0 = byte array containing coded grade of pixel indicating pixel affected by saturation, saturation bloom/bleed, contamination spot, dust, hot pixel, or dust growth. See Section 2.5.3 for details about the effects of contamination on XRT data and implications for data analysis [default].

1 = double array, same as GRADE_TYPE=0 but with added encoding for the number of 1x1 pixels affected within each binned pixel (useful only for binned data).

2 = do not return a grade array.

COALIGN: (Boolean or integer) Set to type of coalignment to apply to XRT data.

0 = [default]: use UFSS data.

1 = use UFSS data plus cross-correlation with AIA 335 Å.

2 = No adjustment.

/QUIET: (Boolean) Set for no messages or errors. The default is some messages and all errors.

/VERBOSE: (Boolean) Set for all messages, errors, and intermediate data listings. Suppresses /QUITE.

/QSTOP: (Boolean) For debugging.

Optional output:

DATA_OUT2: (integer array, $[N_x, N_y, N_{img}]$ Processed output XRT images (data cube) like DATA_OUT but without cosmetic correction for contamination spots, dust, and dust growth. The default data type is I*2. See the /FLOAT keyword. This will only return an array if any of the following keywords are used: /ONLY_DESPIKE, /ONLY_DESPOT, or /DESPIKE_DESPOT.

The basic call for the optional output:

```
IDL> xrt_prep, index, data, index_out, data_out, data_out2, $
    /float, /despike_despot
IDL> help, index
    INDEX      STRUCT    =    -> <Anonymous> Array[5]
IDL> help, data
    DATA       INT       =    Array[1024,1024,5]
IDL> help, index_out
    INDEX      STRUCT    =    -> <Anonymous> Array[5]
IDL> help, data_out
    DATA       FLOAT     =    Array[1024,1024,5]
IDL> help, data_out2
    DATA       FLOAT     =    Array[1024,1024,5]
```

Optional output keyword parameters:

MISS_MAP: (byte array, $[N_x, N_y, N_{img}]$) This is a 2D Boolean map of each image.

0: mage pixel had data.

1: Image pixel was missing data and was replaced with the image average.

N_MISS_PIXELS: (long array, $[N_{img}]$) Number of missing pixels found in each image.

GRADE_MAP: (byte array or double array if GRADE_TYPE=1, $[N_x, N_y, N_{img}]$) This is a 2D map of each image with a value given by the sum of:

0: Image pixel was okay.

1: Image pixel was saturated and was replaced with a constant.

2: Pixel was affected by saturation bloom/bleed. If STOP_BLEED=1 it was replaced by a local median.

4: Pixel was affected by a contamination spot.

8: Pixel was affected by a dust speck.

16: Pixel was a hot pixel.

32: Pixel was a potential dust growth pixel.

If GRADE_TYPE=1, a fraction is added encoding the number of 1×1 spot, dust, and hot pixels per binned pixel. Fractions are relevant only for spot, dust, and hot grades.

For each of these grade types, two digits to the right of the decimal place are reserved for the number of 1×1 pixels within the binned pixel which are of the given type. The digits are assigned in the same order as listed above. For example, a 4×4 binned pixel that has 1×1 pixels of which 8 are okay, 3 are spotted, 2 are dust, and 4 are hot has a grade that is 2 + 8 + 16 = 26, plus 0.030204, or 26.030204. (This is potentially useful for thresholding pixels by the degree of "damage" - e.g., if only one 1×1 pixel in an 8×8 pixel is affected by a contamination spot, (i.e. only 1/64 spotted) the user may wish to treat it as okay for purposes of their further analysis. The grade map may be decoded into separate arrays (one map per grade type) using **xrt_pixel_grade.pro**.

N_GRADE_PIXELS: (long array, $[6, N_{img}]$) Number of graded pixels of each type (saturation, bloom/bleed, spot, dust, and hot respectively) found in each image.

SPIKE_MAP: (byte array, $[N_x, N_y, N_{img}]$) This is a 2D Boolean map of each image:

0: Image was okay.

1: Image pixel was found to be a particle hit and replaced by the local average.

N_SPIKE_PIXELS: (long array, $[N_{ing}]$) Number of spike pixels found in each image.

RUN_TIME: (Float scalar) The run time in seconds for **xrt_prep.pro**.

QABORT: (Boolean) Indicates that the program exited gracefully without completing.

0 = program ran to completion.

1 = program aborted before completion.

UNCERT_MAP: Returns an array of the photometric uncertainties within the XRT. These are non-statistical errors caused by JPEG compression, and uncertainties in vignetting, dark correction and filtering. No statistical/photometric errors are included. See the routine **xrt_cvfact.pro** to add in the photon statistics.

2.5.2 Routines Used by xrt_prep.pro

xrt_despike2.pro

The despike module is used to smooth out cosmic ray hits. Generally, this is intended as a cosmetic fix, not as a scientific recalibration of the spiked pixels. However, unspiked pixels are not affected and the SPIKE_MAP can be used to identify the corrected pixels. Therefore, this keyword should be used cautiously when prepping display images, and possibly should not be used at all for quantitative data analysis. This routine replaces the older version of **xrt_despike.pro**.

xrt_med_dark.pro

This program searches for full-frame darks in the XRT data archive, creates a full-frame median dark template and then generates an appropriate dark for the image being processed. Users can specify how many real darks are used to create a model dark and if a local archive does not exist the user can use **xrt_search_network.pro** to set the proper environment variables and retrieve the dark files to create the best possible dark correction. Users can create a single median dark based of the index of the data they want to process.

```
IDL> dark = xrt_med_dark(index)
```

This generates a single median dark that can be used for dark subtraction or baseline calibration. The output dark already has the Nyquist ringing removed but users have the option to recover the raw median dark with the ringing still included by adding it back via the commands:

```
IDL> data_nonyq = no_nyquist(data,index)
IDL> dark_raw = dark + (data - data_nonyq)
```

xrt_clean_ro.pro

The CCD camera has different bias voltages in odd and even pixel columns. As a result, XRT images show a prevalent sawtooth pattern in x at the Nyquist frequency, with a peak-to-peak

amplitude of $\approx$ 2.6 DN. There is also a low-level large-scale "ski-ramp" in y with a shape which can be approximated by an exponential decrease (amplitude $\approx$ 4.3 DN, e-folding width $\approx$ 185 pixels) and a weak linear increase on a base of $\approx$ 42 DN, which is best seen in dark frames.

The Fourier transform of an XRT dark shows many odd features, including "streaks" (narrow ridges of power spanning all y; many are semi-fixed in location), 2D Voigt profiles, and truncated streaks. The latter two features are typically variable in y position; all vary in amplitude. These features are present (though less clearly visible) in the Fourier transform of data as well. The routine **xrt_clean_ro.pro** addresses all of these issues while minimizing artifacts processing data creates.

The procedure removes the bias pattern Nyquist ringing (using **no_nyquist.pro**), the large-scale ramp (using **lsback_away.pro**), and the periodic features
(using **xrt_fourier_vacuum.pro**), in that order. The Fourier filtering can take a while on large images ($\approx$ .15 seconds on a 2048×2048 pixel image using a MacPro with dual 3GHz Xeon chips). Low frequencies on the transform are shielded from correction to avoid damaging the data; thus some low frequency read-out components will remain. Fourier patterns are not removed if a large part of the image (45% – 80% depending on image size) is saturated. The basic call is as follows:

```
IDL> xrt_clean_ro, image_in, image_out, hist=hist
```

Sometimes, the the default parameters cause the routine to correct some parts of the transform which contain a non-negligible data component in the Fourier power. In these cases, the read-out signals can be over-corrected. This can cause sync "ringing" around small, bright features in the corrected image. To reduce this, one can experiment with increasing the threshold for Fourier feature removal (`NSIGMA` [default=4.5]) and/or decreasing level above background in Fourier space to begin data shielding (`NMED` [default=3.5]). For example:

```
IDL> xrt_clean_ro, image_in, image_out, hist=hist, nsigma=5.0, $
    nmed=3.0
```

It may also be useful to experiment with the form of the Fourier analysis by changing `CLEAN_TYPE`. For example:

```
IDL> xrt_clean_ro, image_in, image_out, hist=hist, clean_type=1, $
     nsigma=5.0, nmed=3.0
```

Filtering out cosmic ray hits (see `/ONLY_DESPIKE`) may also be helpful in such cases.

The best available (default) dark subtraction method uses observed darks near in time to the observations to optimally adjust the model dark's base level (`DARK_TYPE=0`). After the high gain antenna failure January 2008, full-frame 1x1 binned darks were rarely taken, for telemetry reasons. The latest version of the software uses the 512×2048 strip darks (which

are taken regularly after August 2009) to emulate the full-frame 1x1 binned darks, and a note to that effect is placed in the history array.

Ordinarily, the program will search the users local copy of the XRT data archive to find and retrieve the required dark frames for the default dark subtraction method. Users without a local copy of the XRT archive have two options:

1. They may simply do nothing, in which case **xrt_prep** will search for the dark files locally, and, not finding them, will default to a pure model dark (DARK_TYPE=2). This may result in a slightly less accurate setting of the zero point in the reduced data, but may be sufficient for many purposes.

2. They may permit **xrt_prep** to copy the required darks from a remote data archive. Before running **xrt_prep**, the user must run a program called **xrt_search_network** that will set up the required paths, permit remote copying, and make a (temporary) local archive for the darks needed.

```
IDL> xrt_search_network,/ENABLE
```

When the **xrt_prep** session is finished, the paths and permissions can be reset by:

```
IDL> xrt_search_network,/DISABLE
```

The user should be aware that remote coping is network speed dependent and that this routine will make a new directory tree in the current directory emulating the structure of the data archive needed. The user may delete this tree after prepping, if desired.

xrt_pixel_grade.pro

This routine identifies the grade of an XRT pixel in an $N_x \times N_y$ array. It reports whether the pixel is affected by saturation, saturation bloom/bleed, contamination spots, dust/weakness (low QE), is "hot", or is a contamination growth on a dust speck and optionally (for binned data) the fraction of the pixel so affected. Users can change the mode of the program by setting the keyword INTERPRET = 1. This will interpret a grade array by generating an $N_x \times N_y \times 6$ array which gives the locations (and optionally, in the binned case when GRADE_FRAC = 1, the fractional contribution to the pixel) of saturation, bloom/bleed, contamination spot, dust/weakness, hot, or dust growth pixels.

nono_vignette.pro

This routine removes the vignetting function from the XRT images. The vignetting function only includes the effects due to the physical obstruction by parts of the telescope and does not include any wavelength dependent effects of the reflectivity of the mirror. For a complete discussion of the vignetting function see Kobelski et al. (2014).

xrt_unc_rel.pro

When a user wants to calculate the uncertainty of the XRT data based on the dark correction, Nyquist noise correction, Fourier filter correction, JPEG compression, and vignetting, **xrt_prep** calls this function. This function does not take binning into account in the JPEG analysis yet and it recovers the relative uncertainty squared. The actual uncertainty can be recovered by applying the following formula. Let u be the uncertainty, u_r represent the relative uncertainty and d be the prepped data, then

$$u^2 = u_r^2 * d^2.$$

xrt_prep applies this formula to generate the output uncertainty array. For further discussions about how the uncertainties in XRT data are calculated see the paper by Kobelski et al. (2014).

get_xrt_expdur.pro

If normalization of the data is requested in **xrt_prep.pro** this function retrieves the exposure duration of the image. The measured exposure duration of the image is stored in the FITS header keyword `E_ETIM` for normal images and `EXCCDEX` for dark images. These keywords will be updated when the data is normalized by **xrt_prep** and it is recommended that users update these keywords when they renormalize the data by any factor.

xrt_spotcor.pro

This routine makes cosmetic correction for CCD spots and dust on XRT data. It achieves this by using thin-plate splines. It is called using the keywords `/DESPIKE_DESPOT` or `/ONLY_DESPOT`. To use this routine, the image array must be composed of images of all the same size, binning, ccd location, and spot epoch. This routine does not work on Gband data. It also doesn't work well for 8×8 binned data but it does a reasonable job for lower binning levels.

The program assumes the timespan of the data in the cube is short enough that the evolution of the underlying structures is not important. It defines the spots which require correction based on the first image in each filter. Since the effect of the spots depends on the temperature of the underlying feature, their evolution over the timespan of the cube should be minimal for the spot correction to be optimal.

The correction is not photometric, though it should be reasonably good in most cases, provided the above guidelines are followed. The latest versions (after the February 2014 update) include a cosmetic correction for dust on the CCD and the growth seen in the dust after bakeouts post light leak (May 2012). The latter dust growth pixels are added to the output spot array but are separately designated (grade = 32) in the grade maps produced by **xrt_pixel_grade.pro**. The model for the dust growth will likely improve over time. See Section 2.5.3 for further discussions about cosmetic corrections to XRT data.

2.5.3 Contamination

Since launch, contaminating material has accumulated on the XRT CCD and focal plane filters (FPFs), causing a decrease in sensitivity. The effect of contamination is wavelength dependent, and the thickness of the contaminant deposited on the FPFs is different for each filter, therefore contamination affects the response of each XRT filter differently (see Narukage et al. (2011) and Narukage et al. (2014)).

The accumulation of contaminant on the CCD has significantly affected the XRT filter responses, especially for observations carried out with the thinner filters for which the longer wavelength contribution, more absorbed by the contaminating material, is higher. Therefore, in order to recover as much as possible the initial sensitivity of the instrument, a CCD bakeout has been performed to evaporate the accumulated material.

After each CCD bakeout, the contaminant layer on the XRT CCD gets thicker with time. Since the thickness of the layer impacts the temperature sensitivity of XRT, we routinely estimate the thickness by using Gband (and light leak) data. A repository of the layer thickness is distributed through SSW and updates happen about once every three weeks. Information regarding the contaminant thickness as a function of time is available online at `http://solar.physics.montana.edu/HINODE/XRT/xrt_contam/index.html`.

Users can determine when the latest update happened by updating their XRT branch of SolarSoft and checking the last entry of the file,
$SSW/hinode/xrt/idl/response/contam/xrt_contam_on_ccd.pdf.

Contaminant Spots on CCD

After this first bakeout, spots have appeared in both X-Ray and Gband XRT images, and they have been interpreted as due to contaminant material congealed in beads on the CCD surface during the bakeout process. A month-long bakeout has been performed soon after the first one, however it did not remove most of these spots. To avoid formation of additional permanent contamination spots the XRT team now performs CCD bakeouts on a regular schedule (at intervals of about 3 weeks), and this regime has proven successful in maintaining the CCD contamination layer thin enough to affect only minimally the filter responses (see Figure 2.9) and to be removed efficiently by the bakeouts without formation of additional spots. As of October 2009, the ratio of spot area to full CCD area is $\approx 5.2\%$.

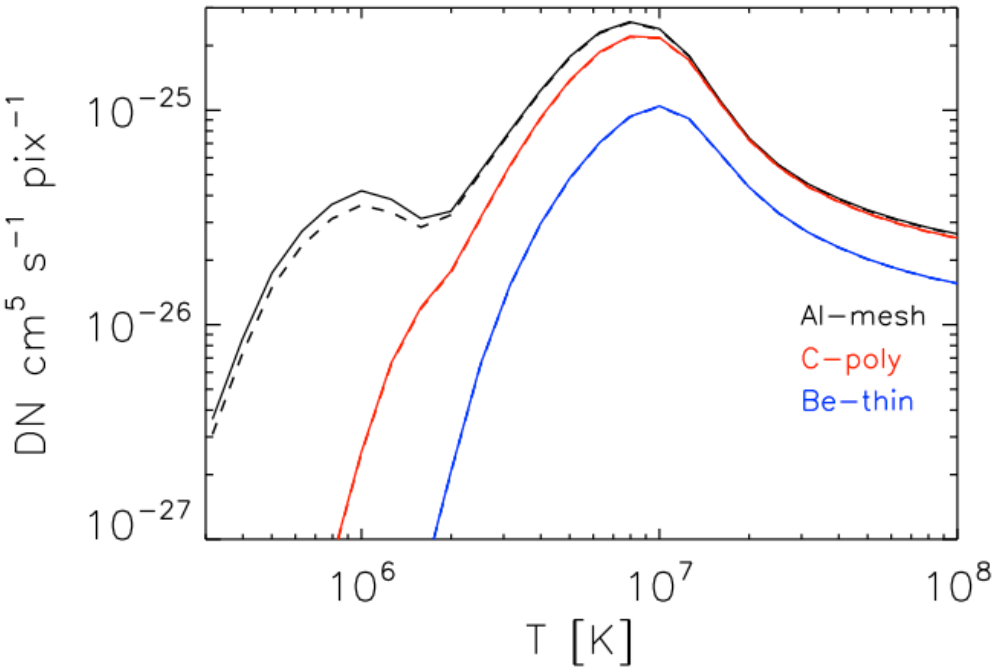


Figure 2.9: Effect of contamination layer, accumulating on the CCD between a bakeout and the next, on the XRT temperature response of Al_mesh (black curves), C_poly (red curves), and Be_thin (blue curves). The solid lines are responses calculated for 2009-09-24, right after a CCD bakeout, and the dashed curves are calculated for 2009-10-15, i.e. three weeks later and just before the following bakeout. The comparison shows that in the regime of regular bakeouts, adopted since mid-2008, only the thinnest filter shows any detectable, though minimal, change in response due to accumulation of contaminating material on the CCD between bakeouts.

The **xrt_prep.pro** routine thoroughly described in the above Section 2.5.1 provides a map of pixels affected by contamination spots, which should be excluded from analysis and has the option to cosmetically correct the data (see keyword parameters `GRADE_TYPE` and `GRADE_MAP`). In addition, there are two routines for cosmetic correction of the contamination spots available outside of **xrt_prep.pro**. **xrt_tup_contam** replaces each contamination spot with the median of the pixels at the edge of each spot.

Basic call to get correct a datacube and indices:

```
   IDL> xrt_tup_contam, ind_in, dat_in, ind_out, $
dat_out[,spotmaps=spotmaps]
```

The optional keyword `SPOTSMAPS` gets maps of which pixels were patched for each image.

xrt_spotcor uses thin-plate splines to smoothly patch over a spot.

Basic call to get a corrected array of image:

```
IDL> xrt_spotcor, index, data, index_out, data_out
```

Call to get a corrected array of image, spot arrays (spotmap), get spot statistics (numspots), and change spot threshold from default (=0):

```
IDL> xrt_spotcor, index, data, index_out, data_out, numspots, $
spotmap, thresh=thresh
```

The threshold allows the user to "accept" (treat as non-spotted) low levels of spot contamination in binned pixels. For example, one would set THRESH=1/16 if a single spot 1x1 pixel in a 4x4 binned pixel was acceptable. For **xrt_spotcor** to work successfully, the data must be of the same size, binning level, CCD location, and cover a reasonably small span of time. An example image processed with **xrt_spotcor.pro** is given in Figure 2.10.

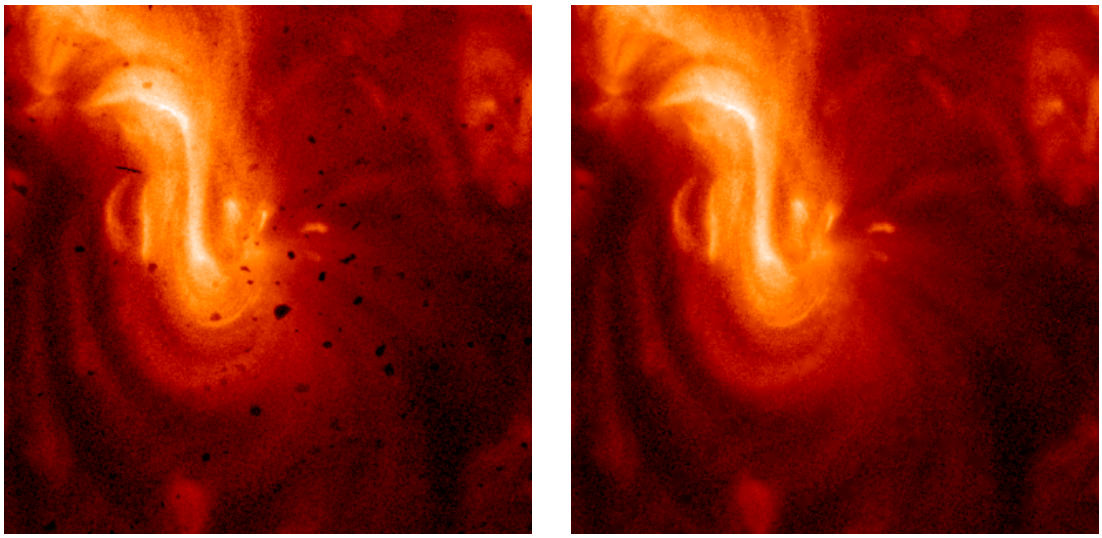


Figure 2.10: An example Al_mesh image taken on 5-August-2010 at 10:01UT. *Left*: The prepped data without spot correction. *Right*: The output image from the routine **xrt_spotcor.pro**. The severity of the spots depends on wavelength with the Al_mesh filter being most affected. Spot correction can be done from within **xrt_prep.pro** and a spot map should be generated to locate affected pixels.

The **xrt_tup_contam** routine is faster, though providing more coarse correction, in particular doing less well in cases where the spot lies over a region which has strong intensity gradients. **xrt_spotcor**, is slower, but does a better job when there are significant gradients in intensity; it is the default spot corrector in **xrt_prep**.

We restate that at present no method is available for quantitative correction to restore in the affected pixels the capabilities for quantitative analysis. Therefore, pixels affected by contamination spots should be excluded from any quantitative analysis.

Contaminant Layer on Focal Plane Filters

The contaminant layer deposited on the FPFs has different characteristics: it was accumulated in the first few months from the start of the mission, for each filter to an extent roughly proportional to their use, and it apparently remained constant after mid-June 2007 when the operational heaters were permanently turned on (and have stayed on until now, October 2009) keeping the temperature of the FPFs high enough ($\approx$ 20°C) to prevent further accumulation of contaminant.

The (wavelength dependent) effect of the layer of contamination on each of the FPFs on the temperature response is modeled and included in the instrument response (see Section 2.11.1 for further details). The effect of this contaminant layer on the FPFs is very limited for most filters, though it is significant for observations carried out with the Al_poly and (to a lesser extent) Al_mesh filters.

2.5.4 Light Leak Overview

The XRT periodically sees an increase in visible stray light contamination, often referred as either *stray light* or *light leak*; summarized in Table 2.2. The leading idea for this increase is a pinhole in the telescopes entrance filter, which allows for some transmission when the visible light shutter is closed. This has lead to significant visible light contributions to X-Ray images in some of the thinner XRT X-Ray filters. Figure 2.11 shows equivalently-scaled Ti_poly synoptic images taken before and after the first light leak; note the additional diffuse emission.

Since the Beryllium images and the thicker Aluminum images remain unaffected, and the Al_poly and Al_mesh images are correctable, XRT still retains the ability to make images in a full range of temperatures, and to distinguish plasmas of different temperatures via all the standard analysis techniques.

After 14 June 2015 the T_poly and C_poly filters should not be used for quantitative analysis, although they might be useful as context images. The XRT Team eliminated the C_poly and Ti_poly filters from all future observation programs. G_band images are still useful for calibration purposes and should be considered to be "engineering data". The Al_mesh and Al_poly images are marginally affected, and may be used with care. Their largest component is estimated to be at the 10 DN/s level, so the effect is negligible for active regions, but more important for dark features. The thicker filters are not significantly affected and may be used as before.

Users of XRT data are encouraged to visit the links provided in Table 2.2 for a detailed analysis of the stray light for each affected filter and to determine whether a data set needs correcting.

Table 2.2: **Summary of Stray Light Events and Links to Analysis Webpages**

2012-May-09, Phase 1
Gband intensity increased by a factor of 2. Significant visible light contributions to X-Rays in Ti_poly and C_poly and minor contributions to Al_mesh. http://solar.physics.montana.edu/takeda/xrt_straylight/xrt_sl_summary.html
2015-June-14, Phase 2
Significant visible light contributions to X-Rays in Ti_poly and C_poly and minor contributions to Al_mesh and Al_poly. http://solar.physics.montana.edu/takeda/xrt_sl2015/xrt15_sl2.html
2017-May-27, Phase 3
Ti_poly and C_poly images used for calibration and engineering purposes. Correctible contributions to Al_mesh and Al_poly. http://solar.physics.montana.edu/takeda/xrt_sl2017/xrt17_sl3.html
2018-May-29, Phase 4
Minor increase in overall intensity. Correctible contributions to Al_mesh and Al_poly. http://solar.physics.montana.edu/takeda/xrt_sl2018/xrt18_sl4.html

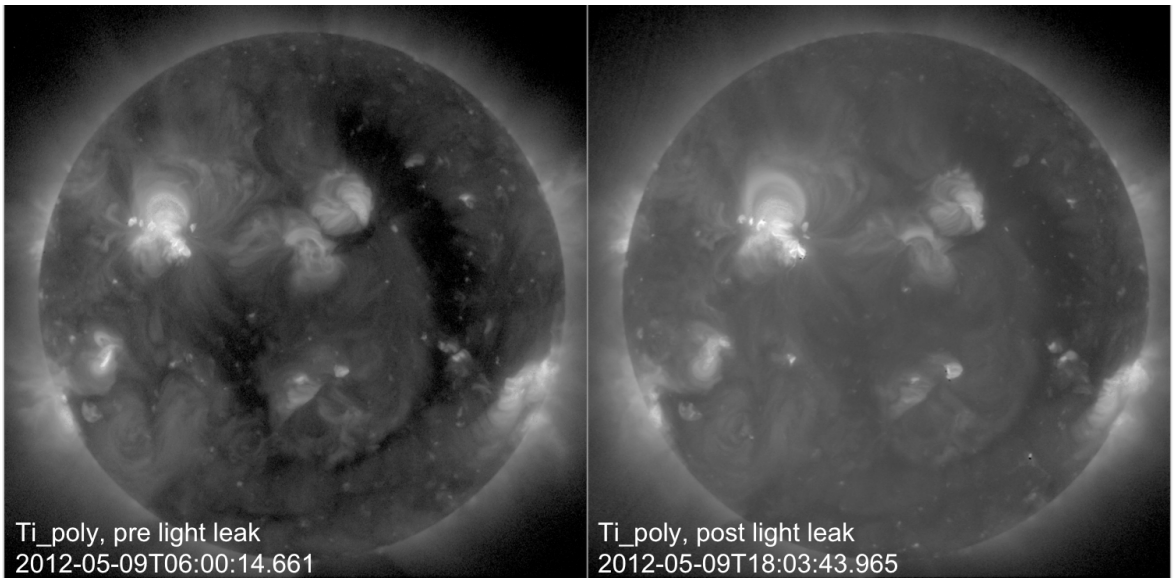


Figure 2.11: Comparison of Ti_poly images before and after light leak.

2.5.5 Light Leak Correction

In March of 2022, software was added to the SSW pipeline to perform light leak corrections. The light leak correction is done by simply subtracting the light leak image (visible stray light component included in each X-ray filter pair) obtained during the Hinode satellite's

eclipse season. This is performed with the following set of scripts available in SSWIDL:

xrt_synleaksub.pro: main program
check_sl_phase.pro: called from the main program
get_slcorfact_raw.pro: potentially called from the main program
leak_fits: directory containing processed leak images

The basic usage is as follows where out_idx and out_da will be the light leak corrected index and data array respectively:

```
IDL> file='./comp_XRT20200220_061539.6.fits'  ; sample XRT file (Al_mesh)
IDL> read_xrt, file, in_idx, in_da
IDL> xrt_synleaksub, in_idx, in_da, out_idx, out_da
```

The light leak pattern and intensity differ with each filter and satellite pointing. Leak pattern and intensity of the same filter and pointing are roughly constant during each stray light phase, but vary by 10 percent depending on the growth of the contamination layer on the CCD (that repeats gradual increase and jump down after CCD bakeouts. The image for the light leak correction should therefore be selected for the same filter, satellite pointing, stray light phase, and ideally be adjusted for the level of CCD contamination at the time of observation. However, as a practical matter, preparing the leak image for every possible pointing is hard to achieve, while we have good amount of light leak measurements at the disk center pointing. The full-disk composite images are therefore corrected most reliably for the light leak.

Intensity variation of the leak image due to the growth of CCD contamination layer is well determined for Ti_poly during stray light phase 1 by using the intensity correlation between Ti_poly and Al_mesh images (cg. Takeda et al. 2016, SolPhys. 291, p.317). The resulting k-factor is obtained with the function GET_SLCORFACT_RAW.PRO, and the leak image subtraction has been already performed only for the Ti_poly SCIA images at the phase 1 (as of March 2022).

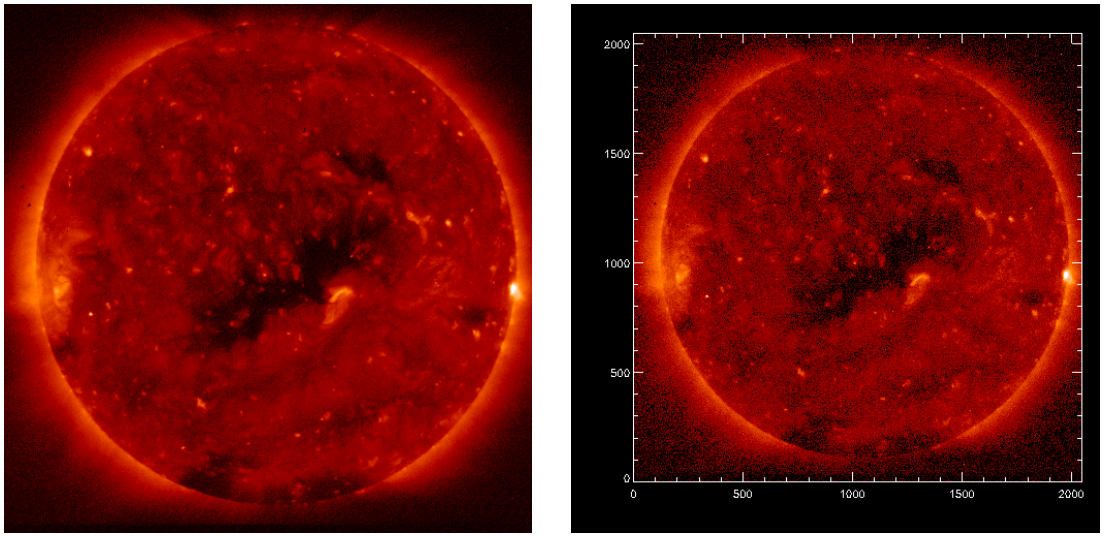
2.6 Displaying XRT Data

XRT has a large dynamic range. To display the data using IDL, it is often useful to logarithmically scale the data. If viewing a 2048×2048 image, which does not fit on most computer screens, it may be useful to rebin the image. The procedure below does pretty well with both raw and prepped images though you may have to change the gamma value of your display using **stretch.pro**. For example, display an image from the example started in Section 2.3. It is a 2048×2048 Ti_poly image that was prepped and normalized. See Figure 2.12.

```
IDL> image=sigrange(data_out[*,*,0], frac=.9999, range=range)
IDL> if range[0] gt 0 then imin=alog10(range[0]) $
    else imin=alog10(0.1)
IDL> imax = alog10(range[1])
IDL> image=alog10(image > 10.0^imin)
IDL> image = image < imax
IDL> loadct, 3
IDL> stretch, 0, 255, 1.1
IDL> wdef, 0, 512
IDL> tvscl, rebin(image, 512, 512)
```

Alternatively, one could use **plot_image.pro**:

```
IDL> plot_image, image
IDL> xdoc,'plot_image.pro'; for more information on plot_image.
```



Logarithmic scale and **stretch.pro** Using **plot_image.pro**

Figure 2.12: Examples displaying XRT data in IDL.

2.7 Coaligning XRT Data

The coalignment routine, **xrt_read_coaldb.pro**, is incorporated into **xrt_prep.pro**. Users can determine what type of co-alignment calibration was performed by reading the image history keyword in the output image header after running **xrt_prep**.

2.7.1 Using xrt_read_coaldb.pro

The **xrt_read_coaldb.pro** routine updates XRT level-0 FITS header keywords to achieve better alignment. The alignment database is kept in the $SSWDB tree and is updated weekly.

Users will have to update the $SSWDB/hinode/xrt/xrt_msu_coalign/ directory routinely to achieve the best alignment results. This routine does not alter XRT data and only updates the following index tags, or FITS keywords, typically used in coaligning images.

```
XCEN
YCEN
CRVAL1
CRVAL2
CRPIX1
CRPIX2
CDELT1
CDELT2
CROTA1
CROTA2
HISTORY
```

This procedure assumes there has been no spatial modification to the data array (e.g., no shift, no rotation, and no enlarge/shrink). If the data has been modified in one of these manners, you may need additional processing. If a user does not have access to the SSWDB database, they can download the database directly from the XRT Coalignment Database website: http://ylstone.physics.montana.edu/yosimura/hinode/coalignment/. Instructions for use are included at the webpage and it is also updated weekly. A paper detailing the generation of the alignment database is currently being prepared by Yoshimura and McKenzie.

The alignment routine has been included in the latest version of **xrt_prep** (v2014-Oct-20) by using the `/COALIGN` keyword. (See Section 2.5.1.)

Basic Call:

```
IDL> new_index=xrt_read_coaldb(index)
```

Inputs:

`INDEX`: (structure array) The XRT index structure.

Outputs:

`NEW_INDEX`: (structure array) The updated XRT structure.

Optional Inputs:

`/AIA_CC`: (Boolean) Set = 1 to utilize the calibration results from cross correlation between XRT thin-filter images and AIA 335 Å images. It is expected that you can get better results with this option, especially in the coalignment with data from SDO. This option is not the default, since sometimes you get "wrong" results with it due to some failures in the cross correlation process.

BASE_DIR: (string) The directory where XRT coalignment databases (coaldb_u, coaldb_c) should be found. [Default: $SSWDB/hinode/xrt/xrt_msu_coalign/].

Optional Outputs:

DB_VER: (String) Version of the coalignment database.

CALIBRATION_TYPE: (Integer) Set to return the resources used for the calibration. In general, smaller numbers mean better results. see below:

-1: No correction. Check if you have access to SSWDB on your system.

1: Cross correlation results (AIA and XRT). From the results of cross correlation between the XRT data and AIA 335A image observed close in time. You can use this resource type with /AIA_CC keyword.

2: Limb fitting results (G-band). If a full disk G-band solar image is available, it can improve the alignment by applying a limb fitting method. The correction using this method is robust.

3: UFSS data. The sun sensor on-board Hinode (UFSS) provides rather continuous output about Hinode's pointing information. We can utilize the information to get the XRT pointing with the correction of the offsets between the UFSS and XRT. This resource can cover most of the XRT data throughout the mission with good accuracy.

4: Limb fitting results (X-Ray). Limb fitting method is also applicable to X-Ray full disk images, though the accuracy of the results is not as good as those from G-band limb fitting.

6: Other methods

Once you get the corrected information for coalignment, you can quickly check the results by using the routine **plot_map**. Below is an example that demonstrates the coalignment results. Starting with an INDEX structure and XRT_DATA array we apply the correction and compare it with an AIA image. See Figure 2.13.

```
IDL> cor_index=xrt_read_coaldb(index)
IDL> new_xrt_data=rot(xrt_data,-cor_index.crota1,/cub)
IDL> cor_index.crota1=0 & cor_index.crota2=0
       ; where "xrt_data" is a single array of XRT data, and
       ; "cor_index" is a corrected index
       ; rotate XRT data and modify the tag of roll angle
IDL> index2map,cor_index,xrt_data, xrt_map
IDL> index2map,aia_index,aia_data, aia_map
       ; Prepare MAP structure
IDL> window,xs=512,ys=512 & mov=bytarr(512,512,2)
       ; Open window and prepare an array for movie
IDL> plot_map,xrt_map,/log & mov(*,*,0)=tvrd()
```

```
       ; display XRT image (log scale) and read it to the movie array
IDL> xr=!x.crange & yr=!y.crange
       ; Store x-,y-range of the plot to use the next plot
IDL> plot_map,aia_map,/log,xr=xr,yr=yr & mov(*,*,1)=tvrd()
IDL> stepper,mov
       ; play movie
```

The routine also works for the better coalignment of a time series of XRT images. Below is an example of making a dijittered movie:

```
 IDL> xrt_prep,old_index,old_data, new_index, new_data
       ; Prep to get corrected index and data
       ; Assuming all data have same x-,y-size and same binning
 IDL> delt_x = (new_index.xcen-new_index(0).xcen)/new_index.cdelt1
 IDL> delt_y = (new_index.ycen-new_index(0).ycen)/new_index.cdelt2
 IDL> shift_val = transpose([[delt_x],[delt_y]])
       ; Calculate offset values for each data referring to the first
       ; image in the sequence (i.e. new_data(*,*,0)).
 IDL> res_data = shift_img(new_data,shift_val)
       ; Shift images to correct their "jittering"
 IDL> stepper,res_data
       ; Watch the results to check
```

More examples can be found at the XRT coalignment webpage at MSU:
http://ylstone.physics.montana.edu/yosimura/hinode/coalignment/samples/

2.7.2 Using xrt_jitter.pro

The routine **xrt_jitter.pro** applies a satellite jitter and orbital drift correction to a time series of XRT images by using the sun-sensor signals and information from coalignment measurements taken regularly since February 2007. See Shimizu et al. (2007) for details.
Basic call to align images of a time series to a reference image (the default being the first image):

```
IDL> xrt_jitter, index, off
```

The output `OFF` is a float array, [2,N_{img}] containing the shift values, in arc seconds, applied to the images in the E-W and N-S direction. For removing jitter and orbital variation, perform:

```
IDL> data_ca = shift_img(data_x, off)
```

Optional keywords:

`REF`: Select the reference image frame [default = 0, using first image as reference].

`XRT_PIX`: Provides the offset values in units of XRT pixels instead of arc seconds.

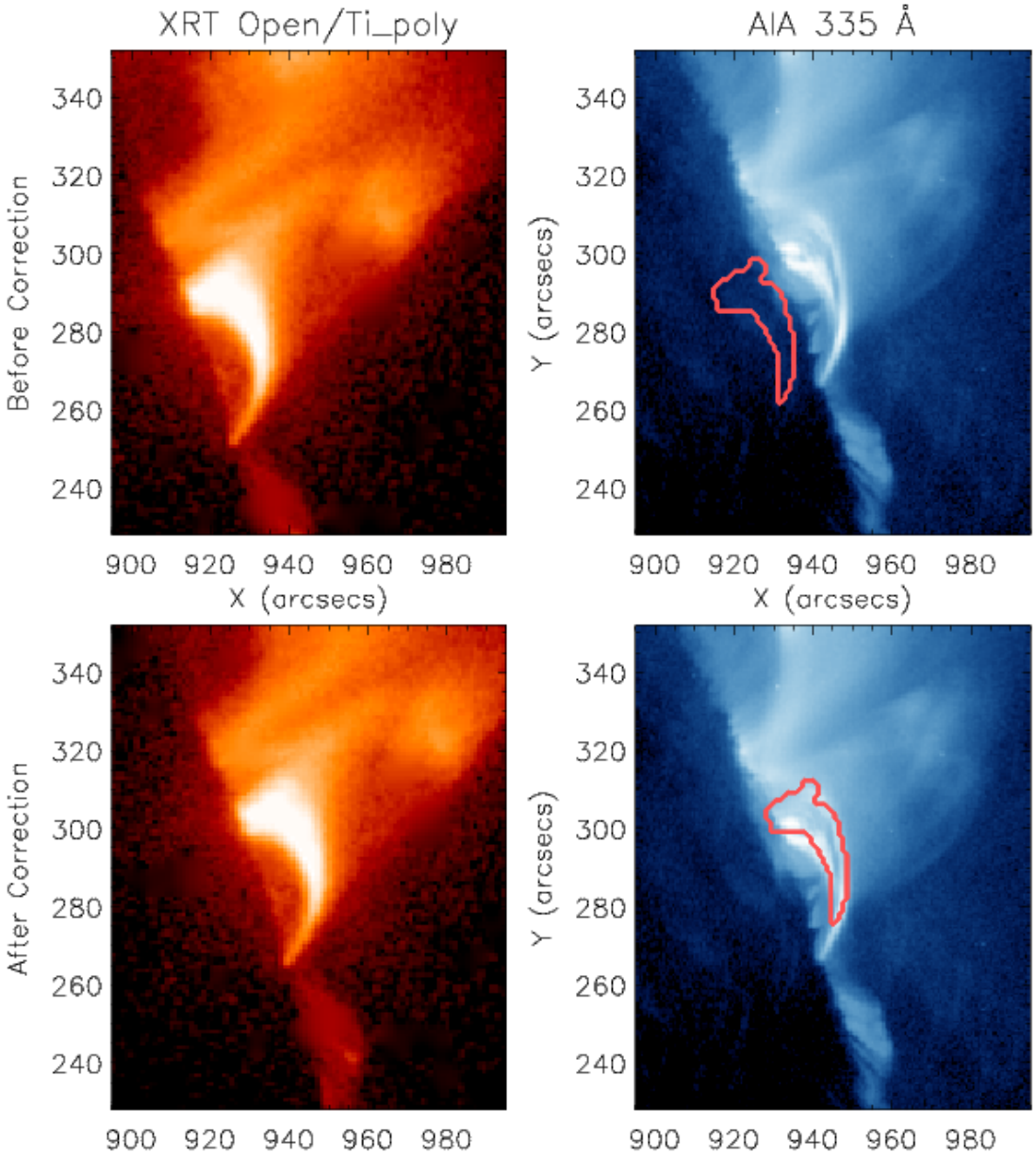


Figure 2.13: Sample coalignment between XRT and AIA images using **plot_map**. XRT images are on the left and AIA on the right. *Upper Left*: XRT Map using level-0 FITS header information without correction. *Lower Left*: Same data after the correction done by xrt_read_coaldb.pro. The image location has changed relative to the x and y axes. The red contour in the right panel show the position of the flaring loop in XRT images.

2.8 Making Composite Images with XRT Data

There are two routines available for making compositie images. The routine **mk_xrt_composite.pro** combines a single long and short exposure pair of images to increase

the dynamic range and to replace saturation with unsaturated data.

Basic call:

```
IDL> mk_xrt_composite, l_idx, l_da, s_idx, s_da, c_idx, c_da
```

Inputs:

`L_IDX`: (structure scalar) Long exposure index.

`L_DA`: (2D number array) Long exposure data.

`S_IDX`: (structure scalar) Short exposure index.

`S_DA`: (2D number array) Short exposure data.

Outputs:

`C_IDX`: (structure scalar) Composite index.

`C_DA`: (2D number array) Composite data.

Optional Keywords:

`DESPIKE`: (Boolean or integer) Set if 1 pass of despiking is wanted, or set to 1-3 for 1-3 passes of despike. [default, no despiking]

`LSAT_MAP`: (Byte array) Set to a Boolean map of the location of saturated pixels to be replaced. Use this if the data have already been **xrt_prepped** and the map was generated or if **xrt_prep** is not to be used. Array must be same size as images to be composited. 1 represents the pixel is to be replaced.

`/VERBOSE` (Boolean) If set, prints out extra messages.

The routine **xrt_batch_composite.pro** can be used to process multiple composite images and deposit them into an index structure array and data array consistent with **xrt_prep.pro** outputs and the like.

Basic call:

```
IDL> xrt_batch_composite, l_idx, l_da, s_idx, s_da, index_out, $
data_out
```

Inputs are simply arrays of **mk_xrt_composite.pro** inputs with outputs of the same form. The keywords of **mk_xrt_composite.pro** are also supported. Note that `LSAT_MAP` must now be an array of Boolean maps. See header for additional information.

It has become customary to take three exposures and generate a composite image from them. The routine **mk_xrt_comp3.pro** will take a set of long, medium and short exposures and turn them into a composite image.

Basic call:

```
IDL> mk_xrt_comp3, l_idx, l_da, m_idx, m_da, s_idx, s_da, c_idx, c_da
```

Inputs:

`L_IDX`: (structure scalar) Index for the long exposure.

`L_DA`: (2D number array, $[N_x, N_y]$) This is the 2D data array for the long exposure image. Can be either Level 0 or Level 1 data as long as it corresponds to `L_IDX`.

`M_IDX`: (structure scalar) Index for the medium exposure.

`M_DA`: (2D number array, $[N_x, N_y]$) This is the 2D data array for the medium exposure image. Can be either Level 0 or Level 1 data as long as it corresponds to `L_IDX`.

`S_IDX`: (structure scalar) Index for the short exposure.

`S_DA`: (2D number array, $[N_x, N_y]$) This is the 2D data array for the short exposure image. Can be either Level 0 or Level 1 data as long as it corresponds to `L_IDX`.

Optional Keywords:

`DESPIKE`: (Boolean or integer) Set if 1 pass of despike is wanted to remove radion-belt/cosmic-ray spikes, or set to 1-3 passes of despike [default is no despike].

`LSAT_MAP`: (byte array) Set to a Boolean map of the location of saturated pixels of the long exposure image. Use this if the data have already been prepped and the map was generated or if **xrt_prep** is not to be used. Array must be the same size as images to be composited.

`MSAT_MAP`: (byte array) Set to a Boolean map of the saturated pixels of the medium exposure image.

`/VERBOSE`: (Boolean) If set, print out extra messages.

Outputs:

`C_IDX`: (structural scalar) This is the index of the composite image. It will have the same content as `L_IDL`.

`C_DA`: (2D number array, $[N_x, N_y]$) This is the data array for the composite image.

2.9 Making Movies with XRT Data

Movies of XRT data can be made using a combination of IDL and software capable of stringing together a series of images, such as QuickTime Pro.

First, display the data on the screen (see Section 2.6). XRT movies are made by capturing images displayed on the monitor using the IDL function **tvrd**, which works in similar fashion to a screen-shot function. For this reason, image quality is dependent on the resolution of your computer monitor. To avoid this problem, port images into the IDL Z-buffer and read them from there:

```
IDL > set_plot, 'Z'
IDL > device, set_r = [512,512]
```

After determining the size of the data array, a movie can be made using a simple for loop:

```
IDL> help, data_out
    DATA_OUT            INT       = Array[512, 512, 64]
IDL> for i=0, 63 do begin &
IDL> tv, bytscl(alog(data_out[*,*,i])) &
IDL> xyouts, 0.03, 0.03, index[i].date_obs, /normal, size=2.0 &
IDL> opf = 'Hinode_XRT'+trim(i,'(i3.3)')+'.png' &
IDL> print, 'Saving image as: ', opf &
IDL> write_png, opf, tvrd(), red, green, blue &
IDL> endfor
```

Generate a movie using software capable of stringing together a sequence of images.

2.10 Writing XRT Data

2.10.1 Using write_xrt.pro

This program will write Level 1 FITS files for the input data (unless the PATHS_ONLY switch is used). There will be one image per file. By default, the files will be written into the current working directory, but file-paths can be modified with keywords.

Examples:
Basic call, to write to a user-specified directory:

```
IDL> write_xrt,index_out,data_out,outdir='/home/user/'
```

Write Level 1 data-cube into the site XRT archive:

```
IDL> write_xrt, index_out, data_out, /archive
```

Get the output filenames without writing the files:

```
IDL> write_xrt, index_out, data_out, /paths_only, out_paths
```

Optional input keyword parameters:

/ARCHIVE: (Boolean) If the /ARCHIVE switch is used, then the standard archive path and standard filename format will be followed. This overrides OUTDIR and OUTFILE. If /ARCHIVE is not used, then OUTDIR and OUTFILE are used if they are given. If OUTDIR is not given, then the current working directory is used. If OUTFILE is not given, then standard format filenames are used. The OUTDIR and OUTFILE keywords operate independently. If they are both given, then OUTDIR+OUTFILE will be used.

OUTDIR: (string scalar) Specifies the fully qualified output directory. Default = the current working directory. OUTDIR overrides ONLINE.

OUTFILE: (string array, [N_{img}]) Specifies the output filenames. Default = follow the XRT filename standard for Level 1 FITS files: L1_XRTyyymmdd_hhmmss.s.fits

/PATHS_ONLY: (Boolean) If set, just return OUT_PATHS but don't write any files.

/MAKE_DIR: (Boolean) If set, create implied directories if they do not exist.

/VERBOSE: (Boolean) If set, print out extra information. Overrides /QUIET

/QUIET: (Boolean) If set, suppress messages.

/QSTOP: (Boolean) For debugging.

Optional output keyword parameters:

OUT_PATHS: (string array, [N_{img}]) List of implied/derived full file-paths for the output.

QABORT: (Boolean) Indicates that the program exited gracefully without completing.

0: Program ran to completion.

1: Program aborted before completion.

2.10.2 Using write_png.pro or write_bmp.pro:

Furthermore, IDL routines will write out lossless image files (such as BITMAP) and lossy images files (such as PNG):

```
IDL> loadct, 3
IDL> tv, bytscl(alog(data_out[*,*,0])}
IDL> write_png, tvrd(), red, green, blue}, or
IDL> write_bmp, tvrd(), red, green, blue
```

2.11 Instrument Responses and Inferring Physical Quantities

Proper analysis of XRT data requires understanding the physical units of the relevant values and functions.

- d_c: data value in an image pixel for a filter channel c. Units are [DN pix^{-1}] (non-normalized) or [DN $pix^{-1}s^{-1}$] (exposure normalized).

  XRT data values, such as are given in XRT data files and as returned by **xrt_prep.pro**, are in units of [DN], for "Data Numbers". This is a measurement of the total photon energy accumulated in the respective pixel across all photon wavelengths over the duration of the image exposure. (Note that, alternately, data values might be in units of [DN s^{-1}] if the data has been normalized for the exposure duration, such as can be done with **xrt_prep.pro**. If normalization was applied with **xrt_prep.pro**, then there will be a sentence in the image `INDEX.HISTORY` tag that starts "XRT_RENORMALIZE Normalized from...".) This measured energy corresponds to emission from a field-of-view determined by the pixel size and binning level.

- `p`: pixel field-of-view size. Units are usually [arcsec].

  The size of the field-of-view of an image pixel is given by the image keywords `CDELT1` or `CDELT2` (length) and `CUNIT1` or `CUNIT2` (units), see Appendix A for a complete list. A full-resolution XRT pixel (i.e., no binning) has the size 1.02860 arc-seconds x 1.02860 arc-seconds. The binning (aka on-chip summing) level of the image is given by the image `CHIP_SUM` tag, but the `CDELT` and `CUNIT` tags should be consistent with that for any binning level.

- `g_port`: the camera gain; conversion factor from DN to N_photoelectrons. Units are [electron DN^{-1}].

  A data value in units of [DN] can be converted to the number of photoelectrons read from the CCD for the corresponding pixel. (This is NOT the same as N_e at the Sun!) The conversion factor is given by the gain setting of the camera's analog-to-digital converter and depends upon which CCD readout port was enabled. Use the function **get_xrt_gain.pro** to get the gains for an array of images. That routine has header documentation that explains its usage, and automatically takes care of accounting for the correct readout port for each image.

- `q`: photoexcitation conversion rate; conversion from N_photoelectrons to electron-volts. The value and units are 3.65 [eV $electron^{-1}$].

  By using the conversion factors `g_port` and `q`, one can convert data values from units of [DN] to total integrated photon energy [eV].

  XRT is a broadband instrument— Figure 2.14 illustrates that the effective areas of the filter-channels (i.e., the instrumental spectral responses) are sensitive to a wide range of

photon wavelengths. As a consequence, large errors may be introduced if one tries to directly convert data values from units of [eV] to [photon number] by assuming any single particular value of photon wavelength. The emitted solar spectrum simply has too many different atomic lines all throughout this broad spectral range of the instrument. Furthermore, since XRT's broadband response implies that individual lines are not distinguished, and hence, spectroscopic analysis cannot be readily performed, electron densities, Doppler velocities, and turbulence parameters cannot be directly inferred. So what *can* be done? We discuss using XRT data to infer electron temperatures and emission measures of the observed solar plasma. Finally, in a roundabout way and doing careful analysis, it is possible to infer photon numbers from the integrated photon energy per pixel that was discussed above.

- $R_c(\lambda)$: instrument effective area (i.e., the spectral response) per photon of wavelength λ for a filter-channel c. Units are [DN cm^2 sr ph^{-1} pix^{-1}] or, optionally, [electron cm^2 sr ph^{-1} pix^{-1}].

  The spectral response function $R_c(\lambda)$ indicates the detector signal (in DN or, optionally, in number of CCD photoelectrons) produced by a photon of given energy. It contains all of the information about the channel response as a function of wavelength. Section 2.11.1 discusses how to obtain XRT's spectral response functions (aka "wavelength responses", as distinguished from the "temperature responses"). Figure 2.15 overlays the spectral response functions for XRT's channels, both with and without the contamination layer for one epoch.

- $S(\lambda, T)$: solar spectrum emission model. Units are [ph cm^{3}s^{-1}sr^{-1}Å^{-1}].

- $F_c(T)$: instrument temperature response function, per emission measure, for a filter-channel c. Units are [DN cm^{5}s^{-1}pix^{-1}] or, optionally, [electron cm^{5}s^{-1}pix^{-1}]. Note that column emission measure has units of [cm^{-5}], so the units of the temperature response functions are, equivalently, [DN s^{-1}pix^{-1}EM^{-1}].

  However, the user will usually only need to obtain the spectral responses in order to then calculate the instrument temperature responses $F_c(T)$. (That process is also described in Section 2.11.1) The temperature response function indicates the detector signal that will be produced by unit emission measure of plasma for a given temperature.

  The spectral responses (effective areas) are purely functions of the instrument. But the temperature responses also depend upon some model of the solar emission spectra corresponding to different plasma temperatures. A spectral emission model $S(\lambda, T)$, such as one would get from ATOMDB/APEC or CHIANTI, is a set of plasma emission spectra for a set of temperatures. It is determined by information from all the assumptions (e.g., elemental abundances, ionization equilibrium state, et cetera) that go into modeling the x-ray radiation from a coronal plasma. The units are those of a photon intensity per column emission measure per area of plasma emitting into a steradian of solid angle. The XRT software will use an APEC solar model by default, but Section 2.11.1 describes how the user can instead apply a solar model that they have independently obtained.

The temperature responses are related to the spectral responses and a solar emission model in this way:

$$\mathrm{F}_c(T) = \int \mathrm{d}\lambda * \mathrm{S}(\lambda, T) * \mathrm{R}_c(\lambda).$$

Under common conditions and reasonable assumptions, it is possible to infer electron temperatures and emission measures for the observed plasma.

Under the assumption that the emitting plasma along the line of sight is of a uniform (or effectively uniform) temperature and emission measure, the technique of filter-ratios may be applied (Section 2.11.3) to infer the uniform temperature T_0 and corresponding emission measure $\mathrm{EM}|_{T_0}$ from measurements in two XRT channels. The observation d_c in each channel satisfies this relation:

$$\mathrm{d}_c = \mathrm{F}_c(T_0) * \mathrm{EM}|_{T_0}.$$

Under the assumption that the emitting plasma along the line of sight is composed of cells of thermalized plasma at various temperatures, each with uniform (or effectively uniform) emission measure, DEMs over some temperature range may be inferred.

- `DEM(T)`: column differential emission measure as a function of temperature T. Units are [$\mathrm{cm}^{-5}\mathrm{K}^{-1}$].

  See Section 2.11.4. for information about an XRT routine to calculate DEMs. DEMs are related to the temperature responses $F_c(T)$ and data values d_c in this way:

$$\mathrm{d}_c = \int \mathrm{d}T * \mathrm{F}_c(T) * \mathrm{DEM}(T).$$

  Do note, however, that an explanation of DEM analysis is well beyond the scope of this Guide— the software is provided for knowledgeable users who understand the assumptions and pitfalls of that sort of analysis.

- $\epsilon_c(\lambda)$: measured photon power as a function of photon wavelength. Units are [DN $\mathrm{s}^{-1}\mathrm{pix}^{-1}$]. (The factors `g_port` and `q`, described above, can be used to directly convert this to [eV $\mathrm{s}^{-1}\mathrm{pix}^{-1}$].)

- $\eta_c(\lambda)$: measured photon number rate as a function of photon wavelength. Units are [ph $\mathrm{s}^{-1}\mathrm{pix}^{-1}$].

- N_{ph}: total photon number rate corresponding to the XRT data value.

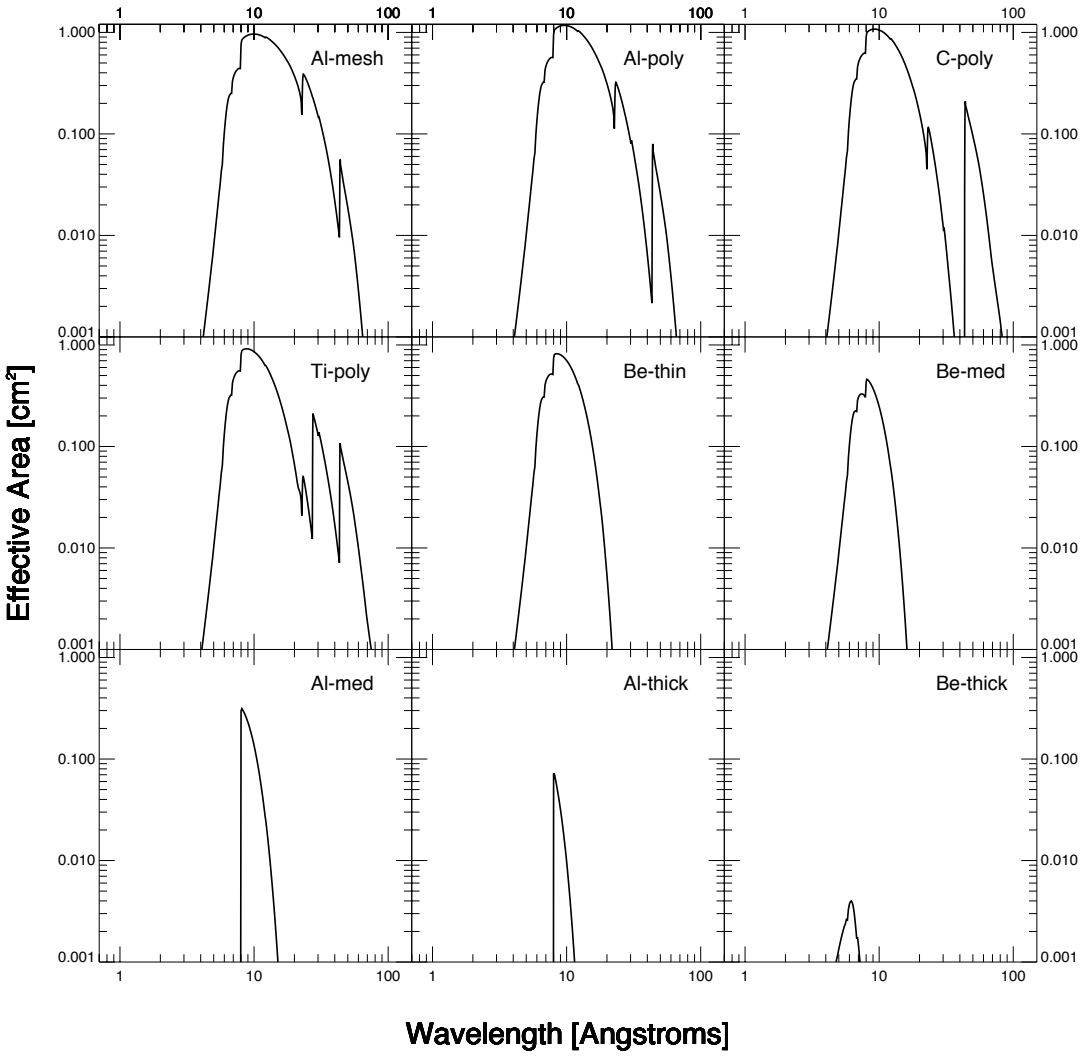


Figure 2.14: The total telescope throughput of the XRT for each of the nine X-Ray filter channels. *Figure 17 from Golub et al. (2007).*

We now return to the subject of inferring photon numbers from data values, which represent an integrated photon energy (units of [DN] or [eV]). An observation in a pixel is the total of all captured photon energies integrated over the solar spectrum. However, one cannot specify the solar spectrum until one has a solar emission model ($S(\lambda, T)$, which is a signal per emission measure as a function of wavelength) and a solution for (a) an emission measure EM at one temperature T_0 or (b) a DEM(T) over a range of temperatures. The relation for the solar spectrum captured in the channel c as a function of wavelength, $\epsilon_c(\lambda)$, is

(a) $\epsilon_c(\lambda) = d\lambda * R_c(\lambda) * S(\lambda, T_0) * EM|_{T_0}$, or

(b) $\epsilon_c(\lambda) = \mathrm{d}\lambda * \mathrm{R}_c(\lambda) \int \mathrm{d}T * \mathrm{S}(\lambda, T) * \mathrm{DEM}(T)$, respectively.

In either case, the net data value of the observation d_c may now be written as $\mathrm{d}_c = \sum_\lambda \epsilon_c(\lambda)$.

Once $\epsilon_c(\lambda)$ is expressed in units of [eV $\mathrm{s}^{-1}\mathrm{pix}^{-1}$] (applying the conversion factors of `g_port` and `q` as needed), the photon number as a function of photon wavelength is $\eta_c(\lambda) = \left(\frac{\lambda}{hc}\right) * \epsilon_c(\lambda)$, and the total captured photon number N_{ph} in channel c is $\mathrm{N}_{\mathrm{ph}} = \sum_\lambda \eta_c$.

Note that, in order to calculate a solar spectrum and then photon number, one relies on some solution for the emission measure or differential emission measure, each of which may only be solved from XRT data alone only by making several simplifying assumptions and avoiding the pitfalls of such analysis. Furthermore, this brief overview has not even touched upon the delicacies of proper error propagation. In short, deriving photon numbers from XRT data is not simple nor straightforward— the instrument has a *very* broadband response and was not designed specifically for inferring photon number. We recommend that users avoid it unless they already have ample expertise in the subject.

2.11.1 Filter Responses

This section describes the wavelength and temperature response of the XRT filters and how to obtain temperatures using the filter ratio method. For a full discussion see Narukage et al. (2011) and Narukage et al. (2014).

The routines providing the wavelength and temperature XRT responses are **make_xrt_wave_resp.pro** and **make_xrt_temp_resp.pro** respectively. The output of **make_xrt_wave_resp.pro** containing the instrument response is used as input to **make_xrt_temp_resp.pro** to calculate the temperature responses. The routine **xrt_teem.pro** will output the temperature and volume emission measure using the filter ratio method.

We note that the updated software should also be used for the analysis of older data (i.e., data taken before the new calibrations were released), as the response functions at all times are better modeled, in particular taking into account the effects of contamination*.

The routine **make_xrt_wave_resp.pro** (v2012-March-16) produces the effective areas and spectral responses for a set of XRT X-Ray channels, taking into account the thickness of the CCD contamination layer.

Basic call:

```
IDL> wave_resp = make_xrt_wave_resp(index=index, $
    contam_thick=contam_thick, contam_time=contam_time)
```

Optional input keyword parameters:

*Following the initial calibration, available at the beginning of the mission, the XRT team has released two significantly updated versions of the XRT calibration taking into account the effects of contamination, described in Section 2.5.3. The version released in September 2008 included an updated calibration able to adequately model the effects of the contamination on the CCD, while the most recent calibration, also takes into account the effect of the contamination layers on the focal plane filters.

`INDEX, CONTAM_THICK, CONTAM_TIME`: One, and only one, of these three input must be specified. `INDEX` must be a standard XRT data "index" array, such as the one produced by **xrt_prep.pro**. `CONTAM_THICK` (scalar) input keyword may be used to specify a single contamination layer thickness (in Angstroms). `CONTAM_TIME` input keyword must be in a time format that is recognizable by the SSWIDL program **anytim2utc.pro**. This input keyword may be used to specify a single "mission time", which will be used to specify a unique contamination thickness.

`CHN_FILENAME`: (Input) A filename of the form "xrt_channels_vNNNN.geny", where NNNN is a 4-digit version number. These files are the XRT channel configuration files distributed by the XRT Team. The default (i.e., if `CHN_FILENAME` is not used) is to get the latest, most correct file. The input may be a full or relative pathname. If it is a relative path name, the program will first look in $SSW_XRT/idl/response/channels/ and then look in the current working directory.

`OUTFILE`: Writes the constructed wave response structure to the specified file (using **save-genx.pro**);

`/EL_UNITS`: Sets the units of the spectral response (in the `SPEC_RESP` array output) to [el cm^2 sr ph^{-1} pix^{-1}]. Otherwise, the default is to produce the the response in units of [DN cm^2 sr ph^{-1} pix^{-1}].

The output structure contains data and information about the effective areas and spectral response functions for a set of XRT X-Ray channels. This output structure is the input to **make_xrt_temp_resp.pro** that describes the instrument response.

Outputs:

`WAVE_RESP`: This structure contains data and information about the effective areas and spectral response functions for a set of XRT X-Ray channels. This is the input that describes the instrument response.

Here is a listing and short description, of the fields of the `xrt_wave_resp` structure:

`WAVE_RESP` {`XRT_wave_resp_vNNNN`}:

- `TYPE`: [Default value = XRT_wave_resp.] This is the generic name of the structure that contains the wavelength response functions and information. The version of the structure definition might evolve, but the TYPE should remain constant. This field should NOT be changed.
- `WRS_STR_VERSION`: [Default value = XRT_wave_resp_vNNNN, where NNNN is a 4-digit number.] This is the version number of the xrt_wave_resp structure definition. In addition to being recorded in this field, this is also the "unique structure name" in the IDL sense. This field should NOT be changed.

- `WRS_STR_DESCR`: This is a sentence or two that describes the purpose of this structure type. This field should NOT be changed.

- `NAME`: This is a name for these wavelength responses for this channel and contamination thickness. On creation, it inherits the name of the `EFF_AREA` structure that was used to create it. This string concisely identifies these dependent wavelength responses (although it may not be unique). This field MAY be changed, but the default is recommended.

- `CONFG`: (structure scalar, {XRT_chn_config_ref_vNNNN}): This structure contains information about the instrument components and the history of their role in the processing that has been performed on this channel to produce this temperature response function. Large arrays (such as the channel transmission function) have been abridged, but there should be sufficient information in this structure to recover or reproduce the corresponding full channel structure (of type XRT_chn_config). The base set of channels is provided in the XRT SSWIDL tree by a file with a pathname like this: `$SSW_XRT/idl/response/channels/xrt_channels_vNNNN.geny`. A listing of the top-level fields of the XRT_chn_config structure, and further details, can be found in the routine header.

- `CONTAM`: (structure scalar, {XRT_contam_ref_vNNNN}): This structure contains information about the particular thickness of the CCD contamination layer (which affects the response of the channel) and the history of its processing. Large arrays (such as the contaminations transmission function) have been abridged, but there should be sufficient information in this structure to recover or reproduce the corresponding full contamination structure (of type XRT_contam). More information about the content of this XRT_contam_ref type of structure may be found in the program header of **make_xrt_ea.pro**. A listing of the fields of the XRT_contam_ref structure, and further details, can be found in the routine header.

- `EFFAR`: (structure scalar, {XRT_eff_area_vNNNN}): This structure contains information about the channel's effective area and about the history of its processing. More information about the content of this XRT_eff_area type of structure may be found in the program header of **make_xrt_ea.pro**. A listing of the fields of the XRT_eff_area structure, and further details, can be found in the routine header.

- `SPRSP`: (structure scalar, {XRT_spec_resp_vNNNN}): This structure contains information about the channel's spectral response and about the history of its processing. More information about the content of this XRT_spec_resp type of structure may be found in the program header of **make_xrt_sr.pro**. A listing of the fields of the XRT_spec_resp structure, and further details, can be found in the routine header.

- `HISTORY`: (string array, [3]): This text array is for XRT programs to record any instance in which they have modified the content of the XRT_temp_resp structure for this particular channel. On creation, a line is added for the creation of the XRT_temp_resp structure in **make_xrt_temp_resp.pro**. An entry string is composed of (a) a program name, (b)

the program's version/date, (c) colon separator, (d) Timestamp (in parentheses) of when the program completed the described action, and (e) a summation of what the program created or modified in the XRT_temp_resp structure. Values are inserted starting at the lowest address of this array. Remaining unused addresses are filled with the empty string. Filled strings should NOT be overwritten. Users should use the COMMENTS field to add their own notations to this structure.

- COMMENTS: (string array, [5]) A tag for adding notations regarding the temperature response described for this XRT channel. Users and programs are encouraged to put content in the lower addresses, and assign empty strings {"} to the higher, unused addresses.

Once the instrument response as a function of wavelength is calculated the routine **make_xrt_temp_resp.pro** can be used to derive the temperature responses of the XRT filters.

Basic call:

```
IDL> result = make_xrt_temp_resp(wave_resp, {[emiss\_model] OR
    [/apec_default] OR [/chianti_default]})
```

Input keyword parameters:

WAVE_RESP: Mandatory input structure containing the data and information about effective areas and spectral response functions for a set of XRT X-Ray channels. This structure array is the output of **make_wave_temp_resp.pro**

EMISS_MODEL, APEC_DEFAULT, CHIANTI_DEFAULT: One, and only one, of these three input must be specified. EMISS_MODEL (structure scalar, {XRT_emiss_model_vNNNN} [Nchannels]) This structure contains data and information about a plasma emission model, as a function of wavelength and temperature. This is the input that associates spectra with temperatures, which allows one to "convert" an instrument's spectral response to a temperature response. For a complete description, see the program header for **make_xrt_emiss_model.pro**.

APEC_DEFAULT : This input keyword switch may be used to indicate that the XRT default emission model from APED/APEC should be utilized. (Some of the details of this model are returned in the TRESP.EMISS substructure of the function output.) The default model assumes coronal abundances Feldman (1992).

CHIANTI_DEFAULT: This input keyword switch may be used to indicate that the XRT default emission model from CHIANTI should be utilized. (Some of the details of this model are returned in the TRESP.EMISS substructure of the function output.) The default model assumes coronal abundances Feldman (1992).

OUTFILE: Writes the constructed temperature response structure to the specified file (using savegenx.pro);

/EL_UNITS: Sets the units of the temperature response (in the TEMP_RESP array output) to [el cm^5 s^{-1} pix^{-1}]. Otherwise, the default is to produce the the response in units of [DN cm^5 s^{-1} pix^{-1}].

Output:

RESULT: Structure containing the data and information about the temperature response for each XRT X-Ray channel described by the WAVE_RESP input. A "channel" is primarily specified by a particular setting of the X-Ray analysis filters and a CCD contamination thickness.

A full listing of the fields of the **XRT_temp_resp.pro** structure, with a detailed description, can be found in the routine header. Here below we include a brief description of only a subset of the fields, and more information can be found in the header of **make_xrt_temp_resp.pro**:

TRESP {<Anonymous>}

TEMP: (Float array, [N_{temps}s]) The abscissa of the temperature response for this channel. The units are provided in the TEMP_RESP_UNITS field. The actual values may form a sequence of shorter length than the length of the TEMP array. The LENGTH field indicates the subarray of usable values, according to TEMP[0:LENGTH-1]. Unused elements are set to 0.0.

TEMP_UNITS: (string scalar) These are the units of the TEMP values, and MUST be equal to 'K' (degrees Kelvin).

TEMP_RESP: (Float array, [N_{temps}]) The temperature response function for this channel, this value indicates the level of CCD signal that is produced in one CCD pixel, in one second, by 1 cm^{-5} (column) emission measure of plasma at the corresponding temperature. The actual values may form a sequence of shorter length than the length of the TEMP_RESP array. The LENGTH field indicates the subarray of usable values, according to TEMP_RESP[0:LENGTH-1]. Unused elements are set to 0.0.

TEMP_RESP_UNITS: (string scalar) Units of the TEMP_RESP values. The default units are [DN cm^5 s^{-1} pix^{-1}] because XRT data are typically presented in DN. However, the /EL_UNITS keyword may be used to get TEMP_RESP in units of [electron cm^5 s^{-1} pix^{-1}].

LENGTH: (long scalar) 0 < LENGTH ≤ n_elements(TEMP) Indicates sublength of the TEMP and TEMP_RESP arrays that contains valid values, starting at the zero-th index. Only the range [0:LENGTH-1] of TEMP and TEMP_RESP will be utilized. Values outside this range may be zeroed. This value is set equal to the corresponding value in the EMISS_MODEL input. This field should NOT be changed.

CONFG: (structure scalar, {XRT_chn_config_ref_vNNNN}) This structure contains information about the instrument components and the history of their role in the processing that has been performed on this channel to produce this temperature response function. All of its values are inherited from the corresponding "CONFG" substructure

in WAVE_RESP (see description above). Large arrays (such as the channel transmission function) have been abridged, but there should be sufficient information in this structure to recover or reproduce the corresponding full channel structure (of type "XRT_chn_config").

CONTAM: (structure scalar, {XRT_contam_ref_vNNNN}) This structure contains information about the particular thickness of the CCD contamination layer (which affects the response of the channel) and the history of its role in the processing that has been performed on this channel to produce this temperature response function. All of its values are inherited from the corresponding "CONTAM" substructure in WAVE_RESP (see description above). Large arrays (such as the contamination's transmission function) have been abridged.

EFFAR: (structure scalar, {XRT_eff_area_ref_vNNNN}) This structure contains information about the channel's effective area and about the history of its role in the processing that has been performed on this channel to produce this temperature response function. Some of its values are inherited from the corresponding "EFFAR" substructure in WAVE_RESP (see description above). Large arrays (such as the effective area function) have been abridged.

SPRSP: (structure scalar, {XRT_spec_resp_ref_vNNNN}) This structure contains information about the channel's spectral response and about the history of its role in the processing that has been performed on this channel to produce this temperature response function. Some of its values are inherited from the corresponding "SPRSP" substructure in WAVE_RESP (see description above). Large arrays (such as the spectral response function) have been abridged.

EMISS: (structure scalar, {<Anonymous>}) This structure contains information about the plasma emission model (input EMISS_MODEL) and about the history of its role in the processing that has been performed to produce this temperature response function. Some of its values are inherited from the EMISS_MODEL input. For a complete description, see the header for **make_xrt_emiss_model.pro**. Large arrays (such as the model's spectra) have been abridged, but there should be sufficient information in this structure to recover or reproduce the corresponding full spectral response structure (of type "XRT_emiss_model").

HISTORY: (string array, [3]) This text array is for XRT programs to record any instance in which they have modified the content of the "XRT_temp_resp" structure for this particular channel.

COMMENTS: (string array, [5]) A tag for adding notations regarding the temperature response described for this XRT channel. Users and programs are encouraged to put content in the lower addresses, and assign empty strings {''} to the higher, unused addresses.

Example:

```
IDL> wave_resp_mar07 =
    make_xrt_wave_resp(contam_time='2007-Mar-01 09:00:00')
IDL> temp_resp_mar07 =
    make_xrt_temp_resp(wave_resp_mar07,/apec_default)
IDL> print,(temp_resp_mar07.history)(0)
    MAKE_XRT_TEMP_RESP v2008-Nov-21: (22-Jan-2010 19:44:11.07 UTC)
    Generated XRT_TEMP_RESP structure with name
    ''Al-mesh; XRT default emission model (APED/APEC 2007-May-17)"
IDL> help,(temp_resp_mar07.TEMP),(temp_resp_mar07.TEMP_RESP)
    <Expression>    FLOAT     = Array[50, 15]
    <Expression>    FLOAT     = Array[50, 15]
IDL> wave_resp_mar08 =
    make_xrt_wave_resp(contam_time='2008-Mar-01 09:00:00')
IDL> temp_resp_mar08 =
    make_xrt_temp_resp(wave_resp_mar08,/apec_default)}
```

The XRT temperature responses produced by the above commands are shown in Figure 2.15.

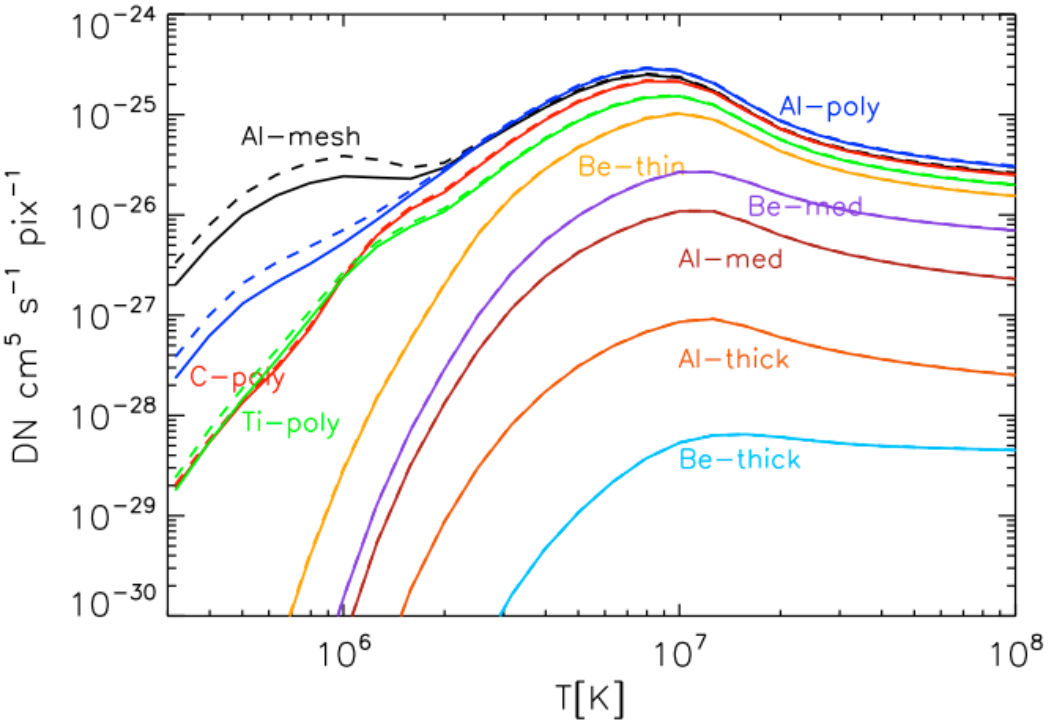


Figure 2.15: Example of XRT temperature responses calculated for two different dates. The solid lines are responses calculated for 2007-03-01, before the first CCD bakeout, and the dashed curves are calculated for 2008-03-01, in the regime of regular bakeouts. The comparison shows how the sensitivity in the lower energy range, significantly decreased by the contamination material, has been recovered through CCD bakeout and maintained with regular CCD bakeouts.

2.11.2 Calculating XRT Filter Response with Non-Standard Spectra

You can calculate the XRT filter response using non-standard spectra by using the routine **make_xrt_emiss_model.pro**; it puts a spectral emission model into a structure format that other XRT routines will expect.

Basic call:

```
IDL> emiss_model = make_xrt_emiss_model(model name, wave, temp, $
    spectrum, abund_model, ioneq_model, dens_model, $
    data_files = data_files)
```

Inputs:

`MODELNAME`: (string scalar) This will be the name/ID given to this spectral model.

`WAVE`: (Float array, [N_λ]) This is a 1D array of monotonically increasing wavelengths. Units are in angstroms. Must correlate to one dimension of the `SPEC` array. See `WLENGTH`.

TEMP: (Float array, [$N_{temperature}$]) This is a 1D array of monotonically increasing temperatures. Units are Kelvin. Must correlate to one dimension of the SPEC array. See TLENGTH.

SPEC: (Float array, [$N_{\lambda}, N_{temperature}$]) This is a 2D array of a spectral emission model. Units are ph cm^3 $s^{-1}sr^{-1}Å^{-1}$. This variable is dependent on WAVE AND TEMP. See WLENGTH and TLENGTH.

ABUND_MODEL: (string scalar) This is a brief description or reference for the abundance model that was used to calculate the spectra.

IONEQ_MODEL: (string scalar) This is a brief description or reference for the ionization equilibrium model that was used to calculate the spectra.

DENS_MODEL: (string scalar) This is a brief description or reference for the density or emission measure model that was used to calcite the spectra.

Optional Inputs:

WLENGTH: (long scalar, 0 < WLENGTH ≤ n_elements(WAVE)) Indicates sub-length of WAVE array that contains valid values, starting at the zeroth index. Only the range [0:WLENGTH - 1] of WAVE and of the corresponding dimension of SPEC will be utilized. Values outside this range may be zeros. Default behavior is to set this value equal to the length of the WAVE input array.

TLENGTH: (long scalar, 0 < TLENGTH ≤ n_elements(TEMP)) Indicates sub-length of TEMP array that contains valid values, starting at the zeroth index. Only the range [0:TLENGTH - 1] of TEMP and of the corresponding dimension of SPEC will be utilized. Values outside this range may be zeroed. Default behavior is to set this value equal to the length of the TEMP input array.

DATA_FILES: (string scalar or array, [N ≤ 5]) These are the source files for the emission model. It is up to the user whether they wish to provide them.

COMMENTS: (string scalar or array, [N ≤ 5]) There is a field in the generated structure to store any relevant comments.

OUTFILE: (string scalar) Write the constructed spectral emission model structure to the specified file using **savegenx.pro**. The extension ".geny" will be automatically appended to the filename. See "Return" output for structure description.

/VERBOSE: (Boolean) Print out extra information. Overrides /QUIET.

/QUIET: (Boolean) Suppress messages.

/QSTOP: (Boolean) For debugging.

Outputs a constructed spectral emission model structure. For example,

```
TYPE    STRING    'XRT_emiss_model'
EMS_STR_VERSION:    STRING    (anonymous)
EMS_STR_DESCR:    STRING    'Solar spectral emission model
        in XRT-compatible format.'
NAME:    STRING    'XRT default (APED/APEC)'
WAVE:    FLOAT    Array[5000]
WAVE_UNITS:    STRING    'Angstroms'
TEMP:    FLOAT    Array[50]
TEMP_UNITS:    STRING    'K'
SPEC:    FLOAT    Array[5000,50]
SPEC_UNITS:    STRING    'ph cm^3 s^-1 sr^-1 A^-1'
WLENGTH:    LONG    5000
TLENGTH:    LONG    50
ABUND_MODEL:    STRING    'Photospheric abundances'
IONEQ_MODEL:    STRING    'Mazotta et al. (1998)'
DENS_MODEL:    STRING    '1e10 g cm^-3'
DATA_FILES:    STRING    Array[5]
HISTORY:    STRING    Array[3]
COMMENTS:    STRING    Array[5]
```

Here is an example using the default emission model for AIA. To learn more about the AIA routines see the SDO Data Analysis Guide at https://www.lmsal.com/sdodocs/doc/dcur/SDOD0060.zip/zip/entry/index.html.

```
IDL> aia =aia_get_response(/emiss,/full)
IDL> help, aia, /str
** Structure <9000008>, 8 tags, length=501700960,
    data length=496215788, refs=1:
   DATE            STRING    '20130107_193807'
   GENERAL         STRUCT    -> MS_129742904002 Array[1]
   LINES           STRUCT    -> MS_129742904003 Array[548515]
   CONT            STRUCT    -> MS_129742904004 Array[1]
   FIXES           STRING    Array[3]
   TOTAL           STRUCT    -> MS_129742904005 Array[1]
   NOTES           STRING    Array[1]
   VERSION         INT              4
```

```
   IDL> modelname='AIA default spectrum'
   IDL> wave=aia.total.wave
   IDL> temp=10.^(aia.total.logte)
   IDL> spectrum = aia.total.emissivity
   IDL> abund_model= aia.general.abundfile
   IDL> ioneq_model = aia.general.ioneq_name
   IDL> dens_model = aia.general.model_name + ',p='+ $
       trim(aia.general.model_pe,1)
   IDL> emiss_model = make_xrt_emiss_model(modelname, wave, temp, $
       spectrum, abund_model, ioneq_model, dens_model, $
       data_files=data_files)
   IDL> help, emiss_model, /str
   ** Structure <4806e08>, 18 tags, length=3673200,
       data length=3673188, refs=1:
      TYPE            STRING    'XRT_EMISS_MODEL'
      EMS_STR_VERSION STRING    '<Anonymous>'
   EMS_STR_DESCR STRING 'Specifies a spectral emission model in X'...
      NAME            STRING    'AIA default spectrum'
      WAVE            FLOAT     Array[9001]
      WAVE_UNITS      STRING    'Angstroms'
      TEMP            FLOAT     Array[101]
      TEMP_UNITS      STRING    'K'
      SPEC            FLOAT     Array[9001, 101]
      SPEC_UNITS      STRING    'ph cm^3 s^-1 sr^-1 A^-1'
      WLENGTH         LONG              9001
      TLENGTH         LONG               101
   ABUND_MODEL STRING '~/ssw/packages/chianti/dbase/abundance/s'...
   IONEQ_MODEL STRING '~/ssw/packages/chianti/dbase/ioneq/chian'...
      DENS_MODEL      STRING    'Constant pressure,p=1E+15'
      DATA_FILES      STRING    Array[5]
      HISTORY         STRING    Array[3]
      COMMENTS        STRING    Array[5]
IDL> obs_date = '2011-08-21'
IDL> wresp_all = make_xrt_wave_resp(contam_time = obs_date)
IDL> help, wresp_all
WRESP_ALL       STRUCT    = -> XRT_WAVE_RESP_V0002 Array[15]
IDL> tresp_all = make_xrt_temp_resp(wresp_all, emiss_model)
```

```
IDL> help, tresp_all, /str
** Structure <4001c08>, 20 tags, length=5536,
   data length=5468, refs=1:
   TYPE            STRING    'XRT_TEMP_RESP'
   TRS_STR_VERSION STRING    '<anonymous>'
TRS_STR_DESCR STRING 'Provides temperature response function f'...
   NAME            STRING    'Al-mesh; AIA default spectrum'
   TEMP            FLOAT     Array[101]
   TEMP_UNITS      STRING    'K'
   TEMP_RESP       FLOAT     Array[101]
   TEMP_RESP_UNITS STRING    'DN cm^5 s^-1 pix^-1'
   LENGTH          LONG                101
   GAIN_USED       FLOAT          -1.00000
   WAVE_MIN        FLOAT           1.00000
   WAVE_MAX        FLOAT           400.000
   CONFG           STRUCT    -> XRT_CHN_CONFIG_REF_V0001 Array[1]
   CONTAM          STRUCT    -> XRT_CONTAM_REF_V0001 Array[1]
   FCONTAM         STRUCT    -> XRT_FILT_CONTAM_REF_V0001 Array[1]
   EFFAR           STRUCT    -> XRT_EFF_AREA_REF_V0001 Array[1]
   SPRSP           STRUCT    -> XRT_SPEC_RESP_REF_V0001 Array[1]
   EMISS           STRUCT    -> XRT_EMISS_MODEL_REF_V0001 Array[1]
   HISTORY         STRING    Array[3]
   COMMENTS        STRING    Array[5]
```

The variable TRESP_ALL is a structure that contains the XRT response functions calculated with the default AIA spectrum. A comparison of XRT filter responses calculated with the default AIA spectrum and with the standard XRT spectrum are given in Figure 2.16.

2.11.3 Using xrt_teem.pro or xrt_teem_ch.pro

XRT has two routine to generate XRT coronal temperature images and differential emission measure images using filter ratios. They are called **xrt_teem.pro** and **xrt_teem_ch.pro** (Chianti). The routines use the same methodology but the latter **xrt_teem_ch.pro** has spectra created using several versions of Chianti.

Basic call:

```
IDL > xrt_teem, index1, data1, index2, data2, te, em, et, ee
```

Input Parameters:

INDEX1: (Structure array) XRT index.

DATA1: (2 or 3-dim float array, [x,y,n_images]) XRT **non-normalized** level 1 data.

INDEX2: (Structure array) XRT index.

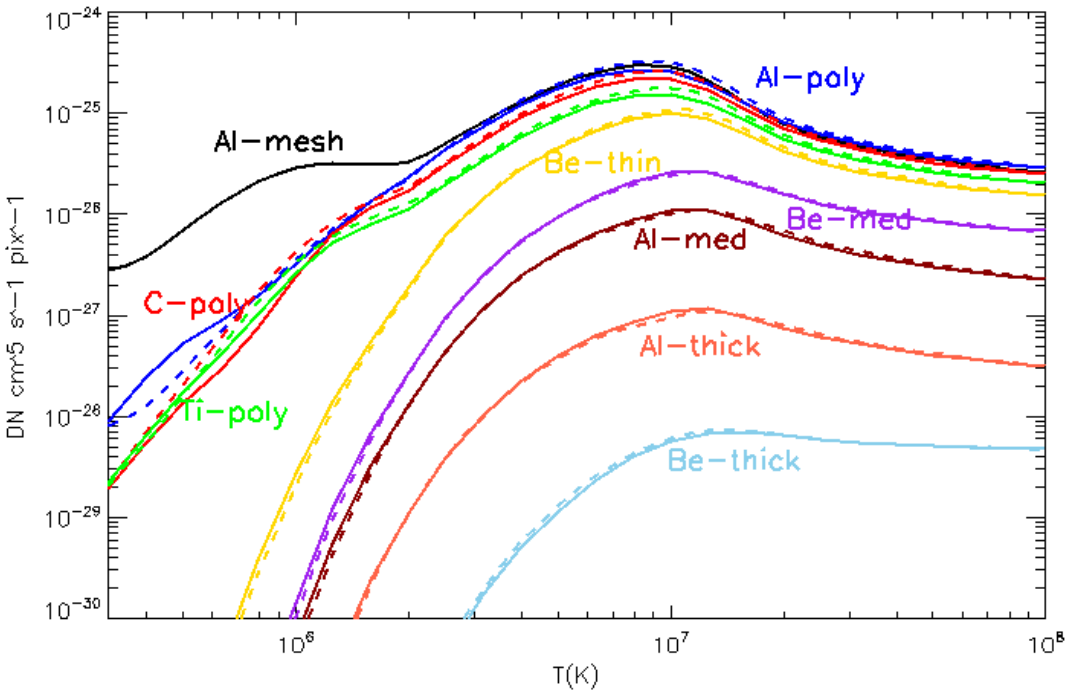


Figure 2.16: Example of XRT temperature responses calculated using the default spectrum for AIA and a standard XRT spectrum: the solid lines are the responses calculated for the standard XRT spectrum, and the dashed curves are calculated using the default spectrum for AIA. Both are calculated for the date 2011 August 21.

`DATA2`: (2 or 3-dim float array, [x,y,n_images]) XRT **non-normalized** level 1 data.

Optional Input keyword parameters:

`CAL`: [default `CAL` =2]

If set,

1= the coronal temperature is estimated based on the calibration in Narukage et al. (2011)

2 = the coronal temperature is estimated based on the calibration in Narukage et al. (2014).

`BIN`: (long) If set, data is binned using this value in spatial dimension.

`TE_ERR_THRESHOLD`: (Float) You can adjust the threshold ratio of the temperature error. (default is 0.1, i.e. 10%.)

`PHOTON_NOISE_THRESHOLD`: (Float) You can adjust the threshold ratio of photon noise to signal. (default is 0.1, i.e. 10%)

`/NO_THRESHOLD`: (Boolean) If set, no threshold is set.

`/INFO:` (Boolean) If set, information is shown.

`LAMBDA_SPECTRUM`: (Float array) wavelength for SPECTRUM in units of Å.

`TE_SPECTRUM`: (Float array) temperature for SPECTRUM in units of [log K].

`SPECTRUM`: (2-dim float array, [lambda,te]) photon number spectrum from solar plasma in units of [$cm^3\ s^{-1}\ sr^{-1}$]. We recommend that:

- The data point of the wavelength is from 1 to 400 [Å] with 0.1 [Å].
- The data point of temperature is from $10^{5.0}$ to $10^{8.0}$ [K] with $10^{0.05}$ [K] resolution.

`/APAC_SPECTRUM:` (Boolean) If set, solar spectrum calculated with APAC database. (default is the solar spectrum calculated with CHIANTI database.)

`EFF_AREA:` (structure) You can use the output from **make_xrt_wave_resp.pro**

Outputs:

`TE`: (2-dim float array, $[x, y]$) Derived temperature in log scale. [log K].

`EM`: (2-dim float array, $[x, y]$) Derived volume emission measure in log scale [$\log cm^{-3}$].

`ET`: (2-dim float array, $[x, y]$) Error in temperature in log scale. [log K].

`EE`: (2-dim float array, $[x, y]$) Error in the derived volume emission measure in log scale [$\log cm^{-3}$].

For the following example, the filter ratio will be taken on Al_mesh and Ti_poly full disk images taken on 2007 Mar 01 at 06UT. The order of the images does not matter as long as the index,data combination is the same for each filter. The two images are XRT20070301_060214.6.fits and XRT20070301_060310.1.fits. To display the images, an appropriate color table should be generated. Figure 2.17 is an example of the output of **xrt_teem.pro** with different threshold values. Figure 2.18 is the corresponding volume emission measures.

Example:

```
IDL> help, in, da
    IN            STRUCT    = -> <Anonymous> Array[2]
    DA             INT       = Array[2048, 2048, 2]
IDL> xrt_teem,in[0],da[*,*,0],in[1],da[*,*,1],te,em,et,ee,$
    /no_thresh
IDL> help,te
    TE              DOUBLE    = Array[2048, 2048]
IDL> help,em
    EM              DOUBLE    = Array[2048, 2048]
IDL> help,et
    ET              DOUBLE    = Array[2048, 2048]
IDL> help,ee
    EE              DOUBLE    = Array[2048, 2048]
IDL> te = 10.^(te)
IDL> te = te / 1000000. ; Convert to Million Kelvin
```

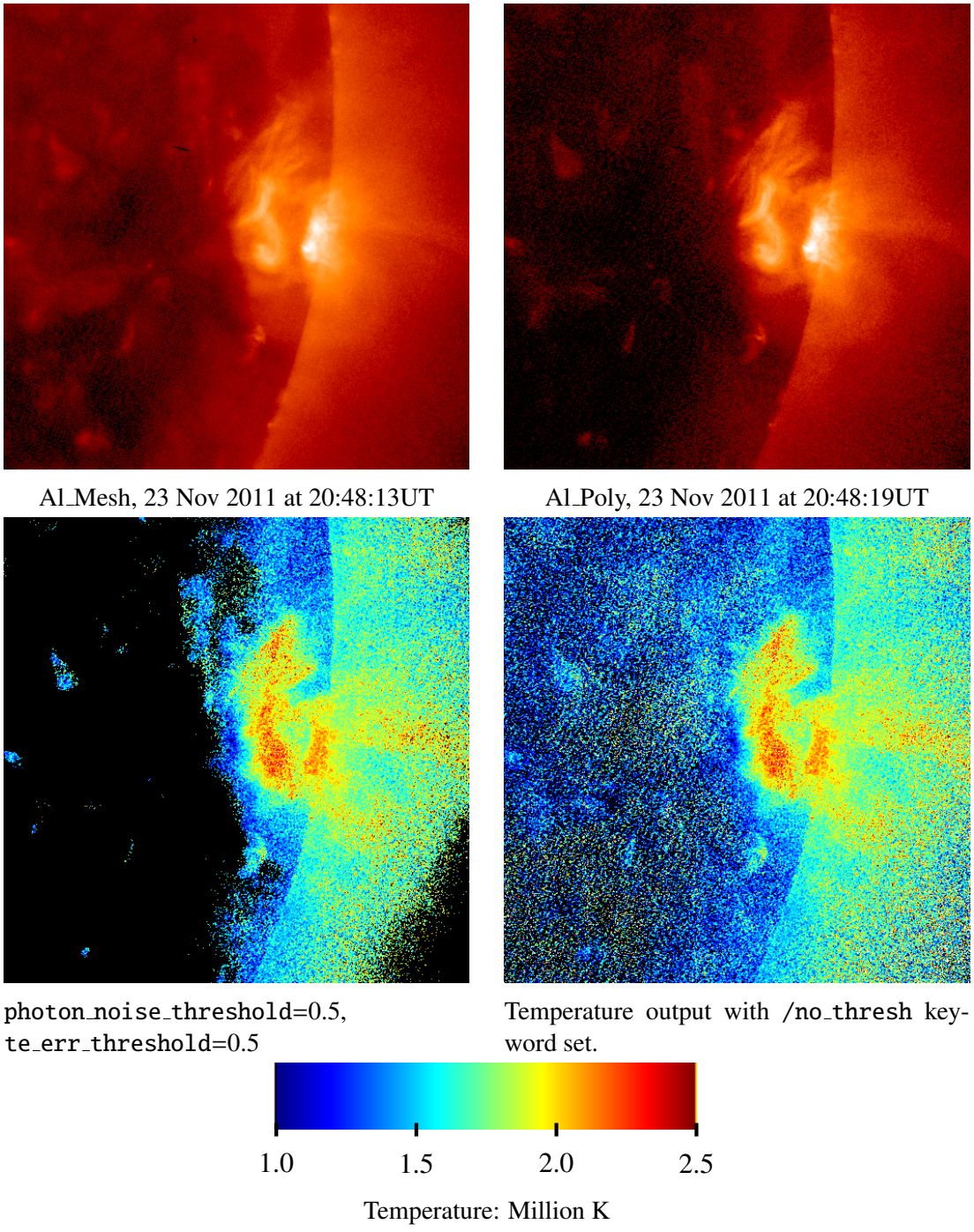


Figure 2.17: Example temperature output from **xrt_teem.pro** using Al_mesh and Al_poly filters. The units have been changed from log K to Million K.

Basic call:

```
IDL > xrt_teem_ch, index1, data1, index2, data2, te, em, et, ee
```

Input Parameters:

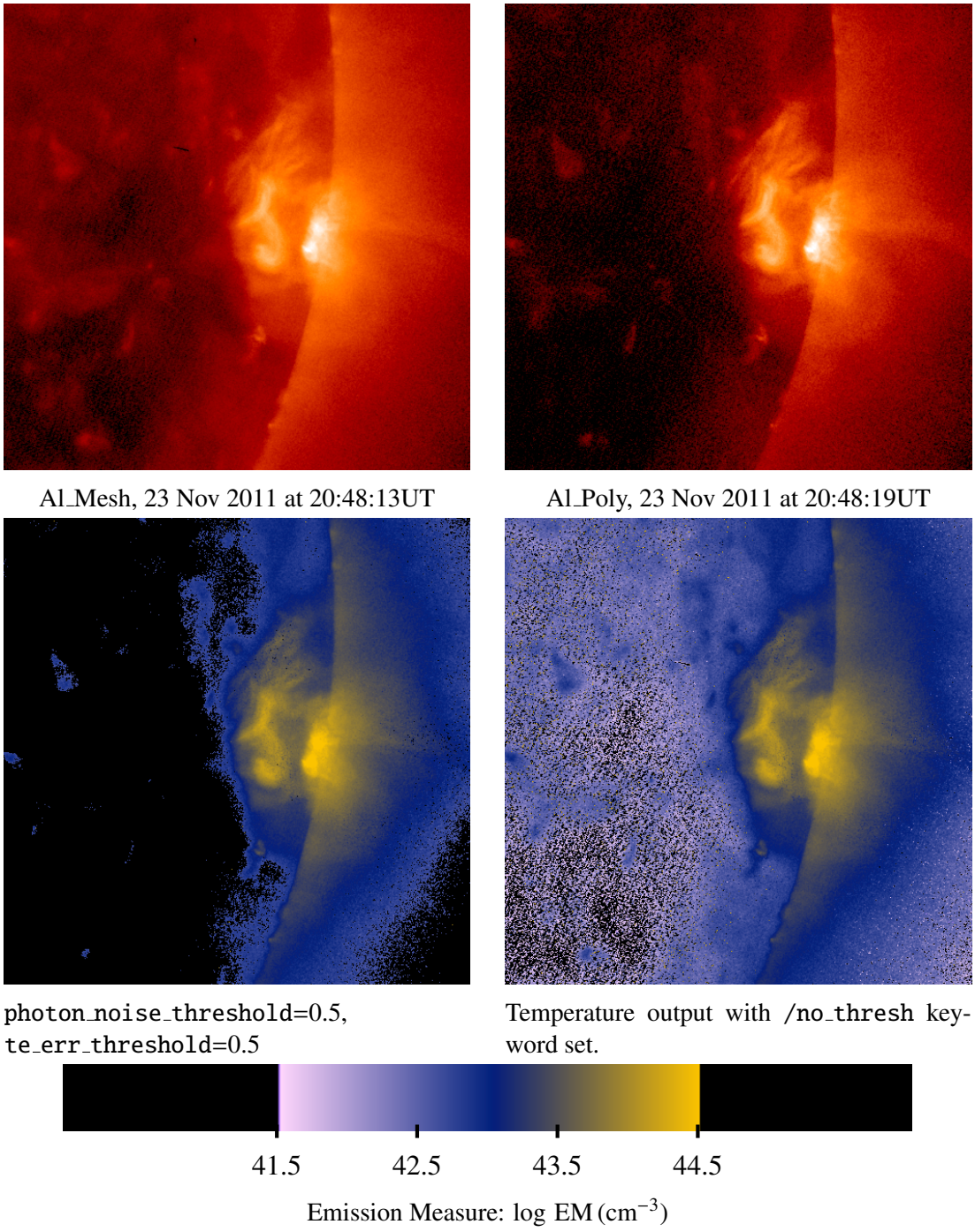


Figure 2.18: Example emission measure output from **xrt_teem.pro** using the Al_mesh and Al_poly filters.

`INDEX1`: (structure array) XRT index.

`DATA1`: (2 or 3-dim float array, [x,y,n_images]) XRT **non-normalized** level 1 data.

INDEX2: (structure array) XRT index.

DATA2: (2 or 3-dim float array, [x,y,n_images]) XRT **non-normalized** level 1 data.

Optional Input keyword parameters:

SPECTRUM_FILE: (scalar string) Specify the name of the solar spectrum files (with path).

CH_VERSION: (scalar string of length 4). Specify the desired version of solar spectrum files calculated with CHIANTI (Densities: 10^9[cm^{-3}]). By default, the routine will use the latest version of Chianti that is available. These files are stored in $SSW_IDL/XRT/idl/response/spectrum_files/. As of this writing, mid-2017 A.D., the versions of Chianti that are available for use by this routine are:

0601 = Version 6.0.1

0700 = Version 7.0.0

0710 = Version 7.1.0

0713 = Version 7.1.3

0800 = Version 8.0.0

list = List all available versions of Chianti. This will cause the routine to exit before calculating the temperature.

The ionization equilibrium: chianti.ioneq. The abundance: sun_coronal_1992_feldman_ext.

HYBRID [Optional input] (boolean) Switch to use hybrid spectrum.

PHOTOSPHERIC [Optional input] (boolean) Switch to use photospheric spectrum.

BIN: (long) If set, data is binned using this value in spatial dimension.

TRANGE: Same as **xrt_teem.pro**.

/NO_THRESHOLD: (Boolean) If set, no threshold is set.

TE_ERR_THRESHOLD: (float) You can adjust the threshold ratio of the temperature error. (default is 0.1, i.e. 10%.)

PHOTON_NOISE_THRESHOLD: (float) You can adjust the threshold ratio of photon noise to signal. (default is 0.1, i.e. 10%)

MASK1 [Optional input] Same as **xrt_teem.pro**.

MASK2 [Optional input] Same as **xrt_teem.pro**.

EFF_AREA same as **xrt_teem.pro**. You can input the output (effective area) from **make_xrt_wave_resp.pro**.

We recommend that:

- The data point of the wavelength is from 1 to 400 [Å] with 0.1 [Å].
- The data point of temperature is from $10^{5.0}$ to $10^{8.0}$ [K] with $10^{0.05}$ [K] resolution.

`/APAC_SPECTRUM:` (Boolean) If set, solar spectrum calculated with APAC database. (default is the solar spectrum calculated with CHIANTI database.)

`EFF_AREA:` (structure) You can use the output from **make_xrt_wave_resp.pro**

Outputs:

`TE`: (2-dim float array, $[x, y]$) Derived temperature in log scale. [log K].

`EM`: (2-dim float array, $[x, y]$) Derived volume emission measure in log scale [$\log \mathrm{cm}^{-3}$].

`ET`: (2-dim float array, $[x, y]$) Error in temperature in log scale. [log K].

`EE`: (2-dim float array, $[x, y]$) Error in the derived volume emission measure in log scale [$\log \mathrm{cm}^{-3}$].

2.11.4 Using xrt_dem_iterative2.pro

DEM estimation is a tricky business. This solver represents one algorithmic approach, and the default XRT temperature responses must necessarily make some assumptions (e.g., about the atomic physics, etc.). The XAG cannot possibly be a primer on DEM estimation, and the end-user is expected to know something about the topic. However, an effort has been made to keep the interface as straightforward and useful as possible. Furthermore, the temperature response codes (**calc_xrt_temp_resp.pro** and associated routines) are designed to be modular, allowing end-users to "roll their own".

This routine estimates a DEM(T) curve, given some observations d_c in channels c, and given the temperature response functions in every channel $F_c(T)$. These functions satisfy the equation:

$$\mathrm{d}_c = \int \mathrm{DEM}(T) * \mathrm{F}_c(T) \mathrm{d}T.$$

The inversion is ill-posed and technically fraught with perils. This routine employs a forward-fitting approach: A DEM is guessed and folded through the $F_c(T)$ to generate "model" observations. This process is iterated to reduce the chi-square between the actual and model observations. The DEM function is interpreted from some spline points, which are directly manipulated by the chi-square fitting routine **MPFIT.pro**. There are $N_c - 1$ splines, representing the degrees of freedom for N_c observations. Note that the number of temperature bins requested for the DEM solution are usually greater than N_c.

To estimate errors on the DEM solution, this routine provides for Monte-Carlo iteration. On each iteration, the observations are varied normally by their sigma error, and then solved for a DEM. According to Monte Carlo theory, the distribution of these DEM solutions

indicates the range of uncertainty in the DEM(T) that corresponds to the unvaried observations. This routine only works on one pixel $\times$ $N_{channels}$ at a time. This routine replaces **xrt_dem_iterative.pro**.

Basic Call:

```
XRT_DEM_ITERATIVE2, obs_index, obs_val, tresp, logT_out, dem_out
            [,obs_err=obs_err] [,base_obs=base_obs] [,mod_obs=mod_obs]
            [,chisq=chisq]     [,min_T=min_T]
            [,max_T=max_T]     [,dT=dT]             [,maxiter=maxiter]
            [,MC_iter=MC_iter] [,solv_factor=solv_factor]  [,/verbose]
            [,/quiet]          [,/qstop]              [,qabort=qabort]
```

Inputs:

OBS_INDEX: (Structure or string array, [N_{img}]). This parameter identifies the XRT X-Ray channel for the corresponding element of the OBS_VAL array. OBS_INDEX may take one of two forms:

a The "index" or "catalog" structure for the image from which this pixel OBS_VALUE came from.

b A string with the name of the XRT X-Ray channel or channel temperature response that corresponds to this OBS_VALUE. Here are the possible names for the default XRT channel names: 'Al-mesh', 'Al-poly', 'C-poly', 'Ti-poly', 'Be-thin', 'Be-med, 'Al-med', 'Al-thick', 'Be-thick', 'Al-poly/Al-mesh', 'Al-poly/Ti-poly', 'Al-poly/Al-thick', 'Al-poly/Be-thick'

OBS_VAL: (Float array, [N_{img}]) This is the set of XRT data values seen in the X-Ray channels identified by OBS_INDEX, respectively. Units = DN s^{-1} pix^{-1}, where "pix" means a one-arcsecond, full-resolution XRT pixel.

TRESP: ("temp_resp" structure array, $N_{channels}$) This structure contains data and information about the temperature responses for a set of XRT X-Ray channels. A "channel" is primarily specified by a particular setting of the X-Ray analysis filters and a CCD contamination thickness. This structure is obtained using **make_xrt_wave_resp.pro** and then **make_xrt_temp_resp.pro**. More information can be found in the headers of those programs and in the XRT Analysis Guide located in the Solarsoft directory, $SSW_XRT/docs/XAG.pdf and at: http://xrt.cfa.harvard.edu/resources/documents/XAG/XAG.pdf.

Outputs:

LOGT_OUT: (Float array, [N_{temp}]) The temperatures corresponding to the DEM solution. Units = log K. Runs from MIN_T to MAX_T with bin-width = DT.

DEM_OUT: (Float array, [N_{temp}, 1+MC_ITER]) A DEM solution is a 1D array of length = N_{temp}. If Monte Carlo runs were performed, then the the 2nd dimension spans these runs. DEM_OUT[$*$,0] is the solution for the original OBS_VAL. DEM_OUT[$*$,1:MC_ITER] correspond to the solutions for the MC runs. Units = cm^{-5} K^{-1}. See Note #3 for more about units.

Optional Inputs:

OBS_ERR: (Float array, [N_{img}]) This input may provide the Gaussian one-sigma errors for the values in OBS_VAL. The solver requires errors in order to calculate the least-squares. If the user does not provide these, then the default will be used. Default = (OBS_VALUE $\times$ 0.03). (Min = 2 DN.)

MIN_T: (Float scalar) Input the low end of the DEM temp. range. Units = 'log K'. Default = 5.5. See Note #2.

MAX_T: (Float scalar) Input the high end of the DEM temp. range. Units = 'log K'. Default = 8.0. See Note #2.

DT: (Float scalar) Input the bin-width of the DEM temp. range. Units = 'log K'. Default = 0.1. See Note #2.

MAXITER: (Float scalar) This program works by iterating a least-squares search. This keyword may be used to specify the maximum number of iterations for each DEM solution. Default = 2000.

MC_ITER: (Float scalar) Use this keyword to cause the program to perform Monte Carlo runs. The value of MC_ITER indicates how many runs. For each MC run, each of the OBS_VAL values are modified from their original value by a random normal amount using the respective OBS_ERR as a Gaussian sigma. Then, the DEM is solved again using the new set of OBS_VAL. See the descriptions of the output variables for how the results are reported. No MC runs are performed if this keyword is not used.

SOLV_FACTOR: (Float scalar) The least-squares solver is not completely insensitive to the order of magnitude of the numbers it is manipulating. SOLV_FACTOR is used to normalize the inputs to move the solver into a "sweet spot". The default choice is arbitrary but seems to work well (default = 1e21). Of course, the outputs are un-normalized at the end. It is recommended that users use the default.

/VERBOSE: (Boolean) If set, print out extra information. Overrides "/quiet" (see Note #1).

/QUIET: (Boolean) If set, suppress messages (see Note #1).

/QSTOP: (Boolean) For debugging.

Optional Outputs:

BASE_OBS: (Float array, [N_{img}, 1+MC_ITER]) These are the observations that a DEM solution corresponds to. BASE_OBS[∗,0] = OBS_VAL. BASE_OBS[∗,1:MC_ITER] correspond to the observations produced by adding random OBS_ERR noise to OBS_VAL for each MC run. You may think of these as the "actual" set of observations that each MC run was trying to solve the DEM for.

MOD_OBS: (Float array, [N_{img}, 1+MC_ITER]) These are the model observations which are produced by the corresponding DEM solution. BASE_OBS is the actual set, and MOD_OBS is how close the DEM can get to matching it.

CHISQ: (Float array, [1+MC_ITER]) These are the chi-square values for the DEM solutions. CHISQ[i] = total(BASE_OBS[∗, i]) − MOD_OBS[∗, i]/OBS_ERR[∗])^2

QABORT: (Boolean) Indicates that the program exited gracefully without completing. (Might be useful for calling programs.)

0: Program ran to completion.

1: Program aborted before completion.

Simplest possible usage:

```
IDL> help, obs_index, obs_val
    OBS_INDEX       STRING    = Array[13]
    OBS_VAL         DOUBLE    = Array[13]
IDL> xrt_dem_iterative, obs_index, obs_val, tresp, logT_out, dem_out
IDL> help, logT_out, dem_out
    LOGT_OUT        FLOAT     = Array[26]
    DEM_OUT         DOUBLE    = Array[26]
IDL> plot, logT_out, dem_out, psym=10, xtit='log T [log K]', $
         ytit='DEM [cm^-5 K^-1]'
```

Provide errors (obs_err) and perform 100 Monte Carlo runs:

```
IDL> help, obs_index, obs_val, obs_err
OBS_INDEX       STRING    = Array[13]
OBS_VAL         DOUBLE    = Array[13]
OBS_ERR         DOUBLE    = Array[13]
IDL>xrt_dem_iterative, obs_index, obs_val, tresp, logT_out, dem_out, $
IDL>obs_err=obs_err, base_obs=base_obs, mod_obs=mod_obs, $
IDL>chisq=chisq, MC_iter=100
IDL>help, logT_out, dem_out
    LOGT_OUT        FLOAT     = Array[26]
    DEM_OUT         DOUBLE    = Array[26]
IDL>help, base_obs, mod_obs, chisq
    BASE_OBS        DOUBLE    = Array[13, 101]
```

```
MOD_OBS         DOUBLE    = Array[13, 101]
CHISQ           FLOAT     = Array[101]
```

Notes:

1 There are three levels of verbosity.

verbose is highest priority. All errors and messages are displayed.

quiet is lower priority. No errors or messages are displayed.

neither is lowest priority. All errors and some messages are displayed.

2 The user may specify the temperature bins over which the DEM will be solved. However, this solution temperature range must lie within all of the T-ranges of the temperature responses, to permit interpolations. (The default XRT responses run logT = 5.5 to 8.0.) Also, the solver requires that the T-range be regular in 'log T' units. Therefore, the solution range is controlled with these three keywords: MIN_T, MAX_T, and DT. Although these temperatures are all presented in 'log T', the underlying integrals are performed over 'T'.

3 A discussion of units. It is easier to understand these units in the larger context. For a spectral model S of the solar radiance, such as is returned by **get_xrt_spec_genx**:

$$S(\lambda, T) \sim [\mathrm{ph\ cm^3 s^{-1} sr^{-1} \AA^{-1}}].$$

For a spectral instrumental response R, such as **calc_xrt_spec_resp.pro** returns:

$$\mathrm{R}(\lambda) \sim [\mathrm{DN\ cm^2 sr\ ph^{-1} pix^{-1}}],$$

which says something about how many DataNumbers are generated in the camera for a photon of a given wavelength. Then:

$$\mathrm{S}(\lambda, T) * \mathrm{R}(\lambda) \sim [\mathrm{DN\ \AA^{-1} s^{-1} pix^{-1} (cm^{-5})^{-1}}] \sim [\mathrm{DN\ \AA^{-1} s^{-1} pix^{-1} EM^{-1}}],$$

where EM $\sim$ cm^{-5} is the line of sight, or column emission measure. For a channel's temperature response F(T):

$$\mathrm{F}_c(T) = \int \mathrm{S}(\lambda, T) * \mathrm{R}(\lambda) * \mathrm{d}(\lambda) \sim [\mathrm{DN\ s^{-1} pix^{-1} EM^{-1}}]$$

Note that one must assume a spectral model of the solar radiance (per emission measure) in order to calculate a temperature response. This must combine an atomic physics model (such as from ATOMDB or CHIANTI) with assumptions about the nature of the solar plasma (N_e, T_e, P, abundances, etc.) along the line of sight.

Going just a bit further, for an EM at $T = T_0$, the signal d observed is $\mathrm{d}|_{T_0} = \mathrm{F}(T_0) * \mathrm{EM}|_{T_0} \sim [\mathrm{DN\ s^{-1} pix^{-1}}]$. For an emission measure continuously distributed over a range

of temperatures T, one uses the differential emission measure: DEM(T) ~ cm^{-5}K^{-1}. Now the net signal d is

$$\mathrm{d} = \int \mathrm{F}_c(T) * \mathrm{DEM}(T) * \mathrm{d}T \sim [\mathrm{DN\ s^{-1} pix^{-1}}].$$

Note that this program **xrt_dem_iterative.pro** just solves the equation:

$$\mathrm{d}_c = \int \mathrm{DEM}(T) * \mathrm{F}_c(T) * \mathrm{d}T$$

As long as the units are all consistent, it is less important what they are. So if the temperature response units were DN cm^{5}s^{-1}pix^{-1}, and the observations were in units of DN s^{-1}pix^{-1}, then DEM will still have units of cm$^{-5}K^{-1}$. The moral of the story is that a combination of temperature responses and observations can be solved, given sufficient care about units and cross-calibrations.

4 There is a program called **xrt_dem2obs.pro**, which may be used to generate a set of XRT observations from a DEM(T) curve. See that program for an explanation of its usage.

We can use these faux observations to illustrate the usage of the solver, since you can compare the "real" original DEM against the estimated DEM. Here is a simple tutorial using both programs. First, create a simple DEM curve.

```
IDL> demt = findgen(26)*0.1 + 5.5 ;; Temps are logarithms.
IDL> dem = demt * 0
 IDL> dem[15] = 1.3e22
 IDL> print, demt[15]
       7.00000
 IDL> wdef, 0, 1000
 IDL> plot, demt, dem, psym=10, /ytype, yr=[1e20,1e23]
```

Now generate the set of observations XRT would see.

```
 IDL> obs = xrt_dem2obs(demt, dem)
 IDL> help, obs                        ;; 14 XRT channels
 OBS             STRUCT    = -> <Anonymous> Array[14]
 IDL> help, obs[0], /st
** Structure <26648d4>, 3 tags, length=28, data length=28, refs=1:
    CHANNEL_NAME    STRING    'Al-mesh'
    OBS             FLOAT           8344.32
    OBS_UNITS       STRING    'DN s^-1 pix^-1'
```

Just to be clear, prepare input variables for the solver.

```
IDL> obs_val = obs.obs
IDL> obs_index = obs.channel_name
IDL> help, obs_index, obs_val
OBS_INDEX       STRING    = Array[14]
OBS_VAL         DOUBLE    = Array[14]
```

Run the solver.

```
IDL> xrt_dem_iterative, obs_index, obs_val, logT_out, dem_out
IDL> help, logT_out, dem_out
LOGT_OUT        FLOAT     = Array[26]
DEM_OUT         DOUBLE    = Array[26]
```

Overplot the solution.

```
IDL> oplot, logT_out, dem_out, psym=10, linesty=2
```

As a follow-up exercise, re-run the solver for 10 Monte Carlo iterations and over plot the results.

5 When the OBS_VAL values are smaller than the OBS_ERR uncertainties, there is a chance that the Monte Carlo random variations to the data set will generate a set of all-zero observations. The program checks for this case, and returns a DEM function equal to zero.

6 This routine deprecates and replaces the older routine **xrt_dem_iterative.pro**. The older routine is not compatible with the most recent response routines.

2.12 The XRT Point-Spread Function

A point spread function (PSF) was developed for the XRT based off of metrology data provided my the mirror manufacturer and on-orbit observations. The metrology data estimate the encircled energy profiles for two energies, 0.56 keV and 1.0 keV at best on-axis focus. The PSFs developed assume best on-axis performance and model scattering caused by the mirror. Other sources of scattering are not included in the model PSF. A complete discussion about the development of the PSF is provided in the Appendix of the paper, Afshari et al. (2016).

The PSF provided can be used with your favorite deconvolution routine. The PSF's are distributed in SolarSoft as fits files in the same location where the deconvolution routine is stored. The algorithm used to deconvolve XRT data is based off the SDO/AIA deconvolution routine called **aia_richardsonlucy_deconvolve.pro**. The routine **xrt_deconvolve.pro** is a wrapper for this program and processes the data prior to deconvolution.

The routine outputs the processed data and updated index structure. It process XRT images in the following ways:

1. For binned data the routine bins and renormalizes the PSF to be on the same plate scale as the images.
2. For small fields of view, the routine drops the image into a 2048x2048 zero array and extracts it after deconvolution.
3. The routines forces the input data to non-negative and less than 2500, the current saturation value of the data.

2.12.1 Using xrt_deconvolve.pro

```
IDL>xrt_deconvolve,in,da,new_in,new_da
```

Optional input keyword parameters:

/NITER: The number of iterations to perform. The default is 5.

/PSF1KEV: Use the 1keV PSF instead of the default 0.56 keV.

/QUIET: Suppress all output.

/VERBOSE: Output extra messages.

The routine does not take into account data saturated pixels or data spikes due to particle hits. These pixels can be excluded by multiplying the input image by a mask where the saturated pixels have value 0 and non-saturated pixels have value 1 in the mask. The mask can be created from either using the grade map output from **xrt_prep.pro** or by flagging the pixels yourself. There will be a ring around the mask in the output image but the masked region is ignored by the deconvolution routine.

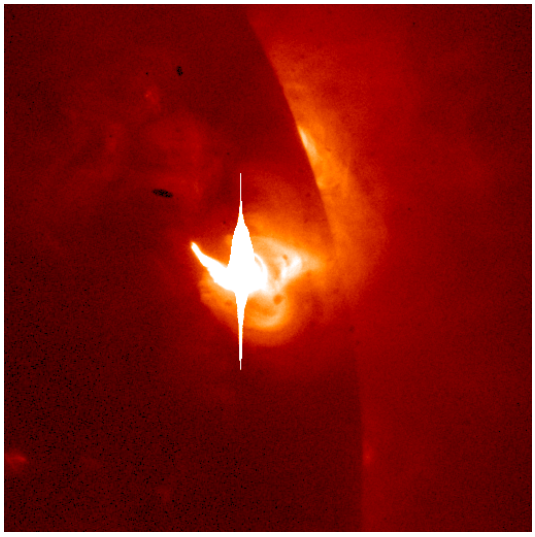
Basic call:

```
IDL>mask=intarr(512,512)+1  ; data=512x512 float array.
IDL>mask[where(data ge 2500)]=0
IDL>xrt_deconvolve,index,mask*data,new_in,new_data
IDL>new_data[where(data ge 2500)]=2500
```

2.13 Known Issues

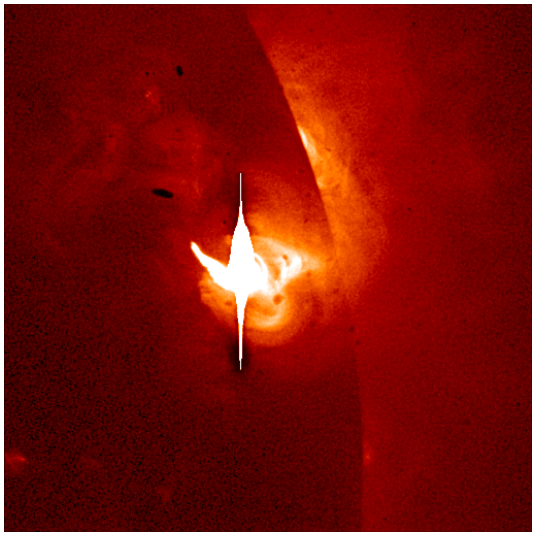
XRT Grade Map

Symptom: There is a minor bug in the grade map pertaining to the pixels that are affected by possible contamination growth. When processing a data cube, the grade map corresponding to the first image is correct. The subsequent grade map values for pixels

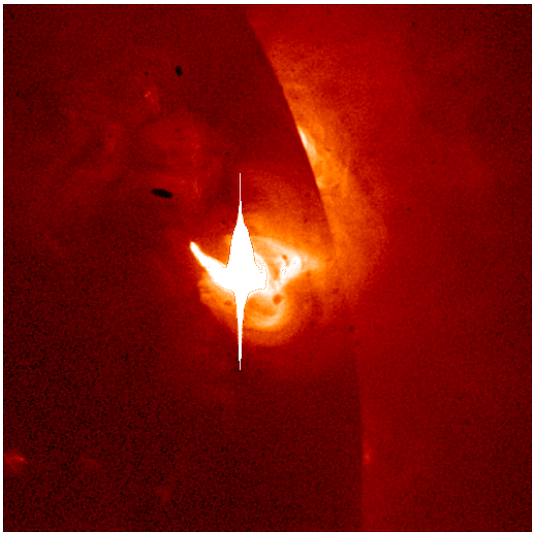


Al_Poly, 16 April 2013, 18:26:31 UT

Saturation Mask



Deconvolution Without Mask



Deconvolution With Mask

Figure 2.19: Example output from **xrt_deconvolve** with and without using a saturation mask.

affected by contamination growth are sometimes not correct. This issue will be dealt with in the next release of **xrt_prep.pro**.

Remedy: If the grade map is needed, then it is recommended that the images are processed as individual calls to **xrt_prep**.

X-Ray Telescope Instrument Guide

3.1 XRT System Overview

The XRT includes a single Wolter I X-ray optic and a companion visible light optic. The optics are supported at one end of a telescope tube. At the opposite end are two filter wheels with analysis filters, the main shutter, and a camera. There is an additional set of heat rejection filters in front of the optics, and a visible light shutter that allows light through the visible light optic. A focus mechanism controls the spacing between the optical elements and focal plane. See Golub et al. (2007).

The Mission Data Processor (MDP) coordinates the operation of the XRT and the CCD camera, in addition to the other components on the satellite. Camera image data is delivered directly to the MDP and is processed there. The only electrical connection between the XRT and the camera is the CCD_EXPOSE line, which initiates an exposure.

3.2 Telescope Performance

Table 3.1: **X-Ray Performance**

Focal Length	2707.2 mm ± 0.3 mm
Aperture Size	34.1 cm
Bandwidth	0.2 to 1.2 keV
Image Performance	68% encircled energy within 27 micron at 0.523 keV at the focal plane
Field of View	$35.11' \times 35.11'$
Visible Light Suppression	10^{11}
Effective Area	> 2.2 cm^2 at 0.523 keV

Table 3.2: **Optical Performance**

Focal Length	2708 mm ± 5 mm (adjusted to focus at the X-ray focal plane)
Central wavelength	4305 Å ±20 Å (G-Band)
Bandpass	< 170 Å FWHM
Resolution	< 2 arcsec half power diameter
Field of View	> 30 arcmin
Co-alignment	17 arc seconds
Confocality with X-ray optic	± 150 micron

3.3 MDP/XRT Communications

Communication occurs using a set of discretes (MDP-to-XRT signals) and bilevels (XRT-to-MDP signals), as well as serial commands sent from the MDP and status packets sent from the XRT. Discretes and bilevels propagate essentially instantaneously on the time-scale of interest. Serial commands consist of status requests, regularly spaced at 500 ms intervals, and other commands, interspersed between status requests, that perform specific actions. Status requests act as a heartbeat, resulting in receipt of a similarly spaced stream of status packets.

3.4 CCD Camera System

XRT uses a back-illuminated three-phase CCD with 13.5 μm pixel-size and 2048×2048 array, which was manufactured by E2V Technologies (Kano et al. 2008). The CCD has two identical read-out ports; R-port and L-port. XRT uses R-port as the default port, and L-port as a backup. From either port, an entire CCD image can be read. Characteristics of the camera are described in the table below.

CCD Specifications

CCD Type	Back-illuminated CCD (E2V/CCD 42-40)
Pixel Format	2048×2048 pixels
Pixel Size	13.5 μm × 13.5 μm
Field of View	35×35 arcmin
Pixel Binning Mode	1×1, 2×2, 4×4, 8×8
Dark Current	0.1 e^-/sec/pixel at -65°C
CCD Temperature	Passive cooling: < −43°C
CTE	Parallel > 0.999996, Serial > 0.999999 (−93°C < T < −50°C)
QE (X-rays/EUV)	0.93 at 13 Å, 0.61 at 45 Å, 0.46 at 116 Å, 0.56 at 304 Å
QE (Visible Light)	0.44 at 4000 Å, 0.66 at 5000 Å
Full-well capacity	2.0×10^5 e^-
Camera Gain Constant	57 e^-/DN
Camera System Noise	< 30 e^-
Output Data Resolution	12 bit

3.4.1 Pointing

Hinode can point anywhere on the solar disk. XRT, SOT and EIS planners decide on the day's pointing every morning. Hinode has four pointing tracks; thus it is not capable of tracking more than four objects over the course of one day.

XRT can offset its pointing from the spacecraft pointing using Region of Interest (ROI) tables. These tables allow the XRT Chief Observer to construct rectangular field of views, though the smallest scientifically valuable field of view is 256×256. The amount XRT can offset depends on the size of the image being read out from the CCD. If reading out a 256×256 image, XRT can offset up to 896 arc seconds from the Hinode pointing. Reading out images larger than 256×256 causes the offset amount to decrease. Reading out the full 2048×2048 CCD requires the XRT pointing to be equal to the spacecraft pointing.

Due to the position of the instrument on the spacecraft, XRT X-ray image pointings are inherently offset from the spacecraft pointing. The XRT X-ray images are offset from the SOT pointing by 40.0 arc seconds in the *x*-direction and 22.3 arc seconds in the *y*-direction. The XRT VLI images are offset from the XRT X-ray images by 32.4 arc seconds in the *x*-direction and 42.3 arc seconds in the *y*-direction. These offsets vary in time due to the

spacecraft orbit and jitter.

3.4.2 Exposures

XRT enables several algorithms to maximize the quality of observations. They are succinctly described below.

Automatic Exposure Control The Automatic Exposure Control (AEC) adjusts the exposure duration by analyzing the most recent X-ray image on board in a pipe-line manner. AEC is available only for the image with size smaller than or equal to 256 k pixels (i.e. a 512×512 pixel image). If an X-ray image does not achieve the proper exposure with the shortest exposure, AEC automatically changes the X-ray analysis filter to a pre-specified thicker filter.

Automatic Region Selector The Automatic Region Selector (ARS) is the function to automatically update ROI targets. This function does not work during the flare observation or during the passage of radiation belts. There are two functions in ARS. One is a global search to select the brightest region in the full frame of CCD and to update ROI1 table. The other is a local search to track a bright region by searching only around the current ROI location, and it updates ROI2, ROI3, and ROI4 independently.

Flare Detection Algorithm Because Hinode is not equipped with any dedicated X-ray detectors to detect solar flares, XRT has to do so by itself. The Flare Detection (FLD) is the function to detect the occurrence of a flare, to identify the flare location on CCD, and to raise a flare flag not only for XRT but also SOT and EIS. XRT takes full frame CCD images with 8 arcsec resolution (called FLD patrol images) at regular intervals. The intervals during the normal observation and during the flare observation can be set independently in the FLD Control Table. The baseline interval to take FLD patrol images is about 30 sec. Each FLD patrol image is first divided into 16×16 blocks, called macro-pixels. The macro-pixel image is created by summing the intensity in each macro-pixel. From a macro-pixel image, the MDP calculates a parameter which indicates the increase of the X-ray intensity normalized by the photon noise; if this parameter exceeds the threshold for the flare start in more than one macro-pixel, the MDP sets the flare flag and proceeds to calculate the flare position. When this parameter is smaller than the threshold for flare end, the MDP drops the flare flag.

3.5 XRT Mechanisms

The X-ray Telescope is composed of entrance filters, mirror, focal plane filters, visible light shutter, visible light imager, focus mechanism, filter wheel and shutter assembly system. Detailed descriptions are available in Golub et al. (2007). Short descriptions of the mechanisms relevant to those proposing joint observing programs are included below.

3.5.1 Visible Light Shutter

The VLS is opened to admit visible light to assist in aiming the telescope and aligning X-ray images with optical images from the SOT, but must be closed during X-ray exposures. The fail-safe mode for the VLS is closed, since X-ray exposures can only be made with the visible light shutter closed. It passes no more than 10-10 of visible light when closed. A stepper motor opens (positive direction) or closes the shutter. Moving the shutter from open to closed requires about 165 steps.

3.5.2 Visible Light Imager

The visible light imager (VLI) is mounted in the center of the Sun shield. X-rays pass through and are focused by the ring near the edge of the Sun shield. Visible light passes through the visible light shutter, if it is open, then through the visible light imager, and to the camera CCD array.

3.5.3 Focus Mechanism

After calibration, a position of 0 places the mechanism in the middle of the range. In this position, there is no stress on the CCD array. The stepping rate is approximately 300 actual steps/s. It takes about 67 s to move from one extreme to the other. Each 200 steps of the stepping motor corresponds to a single revolution of the motor shaft, and 40,000 steps (200 revolutions) are necessary to complete a cycle of the geared down cam. One half cycle, 20,000 steps, moves the focus rod from one focus extreme to the other, a range of ±2.433 mm. This is converted to ±1 mm camera deflection. There is a magnetic detent every fourth step. For this reason, position tracking remains reliable over time only if positions are specified in multiples of four focus motor steps. A focus step is defined as four motor steps. The focus odometer counts motor steps.

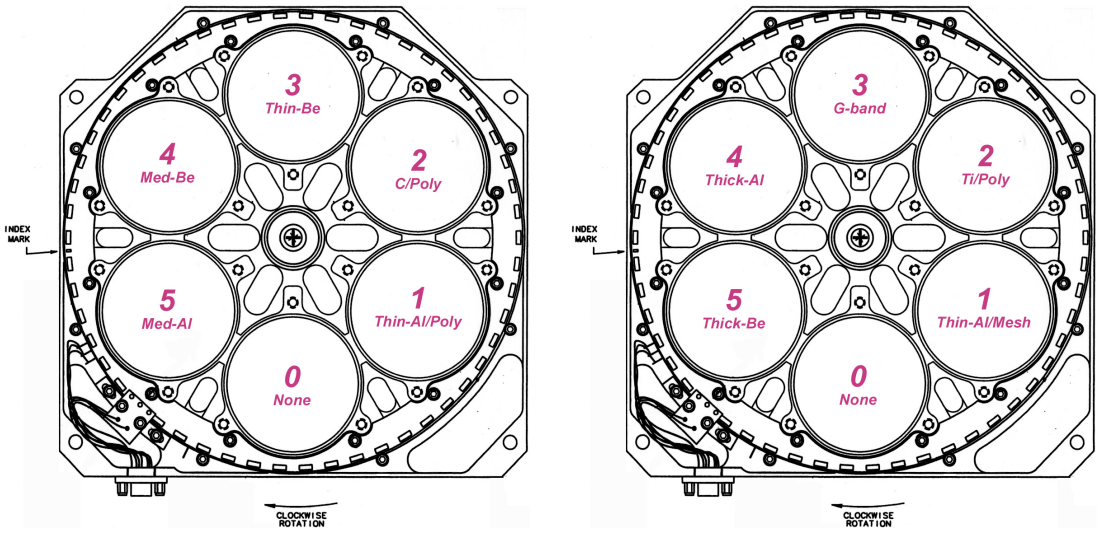
3.5.4 Filter Wheel and Shutter Assembly

The FSA subsystem consists of two filter wheels and the focal plane shutter, and the associated controllers. The focal plane shutter (FPS) is a disk containing narrow, medium and wide slots, shown here with the motor behind. The narrow (1.8°) and medium (14.4°) slots are swept over the camera aperture at a constant rate to make exposures. That is, the motor accelerates to a maximum speed before the slot is reached, then begins braking after the slot has passed. For wide (80°) exposures, the shutter may dwell in the open position as needed to complete the exposure. Thus the shutter brakes to a halt in the center of the wide slot, then reaccelerates to complete the pass. The shutter rotates either clockwise or counterclockwise to perform an exposure, so it is necessary to skip an intermediate slot only one third of the time. Exposure timing is somewhat asymmetric depending upon direction of motion. The shutter is not completely opaque to X-rays. The CCD is swept whenever it is not exposing to flush accumulated charge. A brushless DC motor operates the shutter. Timing marks

placed at 20° intervals determine the resolution with which shutter position is known to the controller.

The FSA subsystem operates two independently-positioned filter wheels. Each filter wheel has six slots, one of which has no filter. The filters provide a temperature resolution of 0.2 log T, 6.1 < log T < 7.5. They are numbered as shown in Figure 3.1.

Filter wheels require nearly 1.5 s to switch between adjacent positions, and thus can make up the longest part of an expose cycle. To minimize positioning time, select exposure order to reduce motion and/or use the expose command for filter wheel pre-positioning. Filter wheels are positioned sequentially, not in parallel, so time is additive (though typically both filter wheels are not used in the same exposure).



Filter Wheel 1 Filter Wheel 2

Figure 3.1: XRT filter wheels as viewed from the sun.

Level 0 Header Keywords

This appendix explains the 162 keywords in the XRT Level 0 FITS header.

XRT Level 0 FITS Header Keywords

Keyword	Value	Definition
SIMPLE	T	Conforms to FITS standard
BITPIX	8, 16, 32, -32, -64	Number of bits per pixel
NAXIS	2	Number of axes in the image
NAXIS1		Full image size in *x*
NAXIS2		Full image size in *y*
DATE	'YYYY-MM-DDThh:mm:ss.sss'	Date and time of file creation ('T' is character 'T')
DATE_RF0	'YYYY-MM-DDThh:mm:ss.sss'	Date and time of Level 0 reformatting ('T' is character 'T')
SATELLIT	0x21	Satellite number (from PDU header for images)
TELESCOPE	'SolarB'	Derived from above
INSTRUME	'XRT'; 'SOT/SP'; 'SOT/FG'	Name of the instrument; reformatter only retrieves XRT records
TIMESYS	'UTC'; 'TAI'	Time system of file header
MDP_CLK		MDP clock in units of $1/512^{th}$ seconds; same as E_SCLOCK
FILEORIG		Original filename used by level 0 reformatter
P1ROW	0 to N-1	*x*-coordinate of beginning, or lower left hand corner, pixel in image FOV; same as RPOS_ROW
P2ROW	0 to N-1	*x*-coordinate of ending, or upper right hand corner, pixel in image FOV; same as RPOS_ROW + RSIZ_ROW - 1
P1COL	0 to N-1	*y*-coordinate of beginning, or lower left hand corner, pixel in image FOV; same as RPOS_COL
P2COL	0 to N-1	*y*-coordinate of ending, or upper right hand corner, pixel in image FOV; same as RPOS_COL + RSIZ_COL - 1
TR_MODE	'TR1'; 'TR2'; 'TR3'; 'TR4'; 'FIX'	Tracking mode

XRT Level 0 FITS Header Keywords

Keyword	Value	Definition
IMG_MODE	1-3	Image mode (source of exposure trigger); 1: Table and manual, 2: Automatic Region Selection Patrol, 3: Flare Detection Patrol
AEC_FLG	'off' ; 'on'	Automatic Exposure Control; 1: Off, 2: On
AEC_TNUM		AEC table number; 0: Out of RB, 1: In RB
AEC_RSLT	0-3	Result of AEC calculation used to determine exposure time; 0: Normal, 1: Underexposure, 2: Overexposure, 3: No feedback
ORIGIN	'ISAS; NAOJ; MSSL; LMSAL; GSFC; 'SAO	Origin of the Chief Observer
DATA_LEV	0, 1	Data Level; running **xrt_prep.pro** will change 0 to 1
ORIG_RF0	'ISAS ; NAOJ ; MSSL ; LMSAL ; GSFC ; 'SAO	Where the level 0 file was created
VER_RF0		Version of Level 0 reformatter
PROG_VER	0-7	MDP observation table program version number
SEQN_VER	0-7	OT sequence table version number
PARM_VER	0-3	OT parameter table version number
PROG_NO		OT program number
SUBR_NO		OT subroutine number being executed
SEQN_NO		OT sequence table number
MAIN_CNT	0-7	OT number of times to repeat main loop
MAIN_RPT		OT current main-routine iteration
MAIN_POS		OT main-routine position
SUBR_CNT		OT sequential number of this subroutine in the main routine
SUBR_RPT		OT number of times current subroutine is repeated
SUBR_POS		OT loop count for current subroutine
SEQN_CNT		OT current sequence table repeat count
SEQN_RPT		OT sequence table repeat count
SEQN_POS		OT sequence table position
OBSTITLE		Title of observation
TARGET	'Active region' ; 'Quiet region' ; 'Coronal hole' ; 'Flare site'	Indicates the observation region. Flare site used when flare flag is set. Source of information observation planning database, or telemetry if flare flag is set.
SCI_OBJ		Up to 5 target phenomena selected from list. See Mission-Wide Keywords document, p. 26.

XRT Level 0 FITS Header Keywords

Keyword	Value	Definition
SCI_OBS		Target phenomena.
OBS_DEC		A few sentences describing the properties of the observation.
JOIN_SB	'ESX'; 'SX'; 'EX'; 'X'	Joint observation; E=EIS, S=SOT, X=XRT
OBS_NUM		Equal to OBS_ID
JOP_ID		Identifier of JOP
NOAA_NUM		AR Number as assigned by NOAA
OBSERVER		Name(s) of Chief Observer
PLANNER		Name(s) of Chief Planner
TOHBANS		Name(s) of Tohbans
DATATYPE	'SCI'; 'ENG'	Science or engineering data; darks and flats are considered engineering data
SAA	'In'; 'Out'	Indicates whether Hinode is in or out of a South Atlantic Anomaly region
HLZ	'In'; 'Out'	Indicates whether Hinode is in or out of Hight Latitude Zone region
FLFLG	'Flr'; 'Non'	Indicates if flare flag set or not
S_INSTRU	4	Instrument number
S_DAT_ID	1-3, 5-7	Type of status packet this record was created from: 0: Not used, 1: Normal status, 2: Normal and extended status, 3: Normal status and memory, 4: Not used, 5: Standard HDR only, 6: Extended status, 7: Memory
S_DAT_M	0, 1	0: Transfer to Kagoshima Space Center (KSC) and Sagamihara Satellite Operation Center (SSOC); 1: Monitor only at KSC
S_SP_SIZ	31-609	Status packet size; maximum value is 609, including header.
EC_ID	0-65535	Unique identifier, 'main ID'
EC_INDEX	0-35	Redundant to EC_EINDE; consider this keyword obsolete
EC_EINDE	0-35	Exposure Index
EC_CD_MO	0, 1	Cadence mode
EC_CD_M_	'safe' ; 'fast'	Cadence mode as name
EC_IMTYP	0, 1	Image type; 0: Normal, 1: Dark (closed shutter)
EC_IMTY_	'normal' ; 'dark'	Image type; a dark is taken with the shutter closed

XRT Level 0 FITS Header Keywords

Keyword	Value	Definition
EC_FW1	0-5	Filter Wheel 1 position
EC_FW1_	'Open' ; 'Al_poly' ; 'C_poly' ; 'Be_thin' ; 'Be_med' ; 'Al_med'	Filter Wheel 1 position as name
EC_FW2	0-5	Filter Wheel 2 position
EC_FW2_	'Open' ; 'Al_mesh' ; 'Ti_poly' ; 'Gband' ; 'Al_thick' ; 'Be_thick'	Filter Wheel 2 position as name
EC_VL	0, 1	Visible light shutter during exposure; 0: Closed, 1: Open
EC_VL_	'closed' ; 'open'	Visible light shutter during exposure as name
E_SCLOCK		Spacecraft clock of most recent status request prior to arrival of exposure command
E_LCLOCK	0-16777215	Time at which exposure command processing began, local clock, converted to μs
E_SH_OPE	0-16777215	Time CCD_EXPOSE and OPENOUT signals were raised (low 24 bits), converted to μs
E_SH_CLO	0-16777215	Time CCD_EXPOSE and OPENOUT signals were lowered (low 24 bits), converted to μs
EXCCDEX		Duration of CCD_EXPOSE in μs; this is the correct value to use for dark exposure times
OBT_TIME		Spacecraft clock time when CCD_EXPOSE was raised; this is E_SH_OPE converted to spacecraft clock time
OBT_END		Spacecraft clock time when CCD_EXPOSE was lowered; this is E_SH_CLO converted to spacecraft clock time
E_SH_POS		Shutter encoder position
E_SH_WA		Waiting position A
E_SH_WB		Waiting position B
E_SH_WC		Waiting position C
E_SH_CW		Waiting clockwise exposure time
E_SH_CCW		Waiting counterclockwise exposure time
E_VLO	0, 1	VLS open microswitch; 0: Off (VLS not fully open), 1: On (VLS fully open)
E_VLO_	'not fully open'; 'fully open'	State of VLS
E_VLC	0, 1	VLS closed microswitch; 0: Off (VLS not fully open), 1: On (VLS fully open)
E_VLC_	'not fully open'; 'fully open'	State of VLS
E_SH_ERR	0, 1	0: No error, 1: Shutter command error
E_FW1_PO		Filter Wheel 1 course position (internal diagnostic format)

XRT Level 0 FITS Header Keywords

Keyword	Value	Definition
E_FW1_ST		Filter Wheel 1 status
E_FW2_PO		Filter Wheel 2 course position (internal diagnostic format)
E_FW2_ST		Filter Wheel 2 status
E_ETIM_E		Exposure time (exponent); see also E_ETIM (Though the value for E_ETIM should be normalized if the data is normalized, E_ETIM_E should remain *unchanged* so the user can reconstruct the original exposure time.)
E_ETIM_M		Exposure time (mantissa); see also E_ETIM (Though the value for E_ETIM should be normalized if the data is normalized, E_ETIM_M should remain *unchanged* so the user can reconstruct the original exposure time.)
E_ETIM		Exposure time in μs, derived from above two fields; this number should be normalized if the data is normalized by exposure time
EXPTIME		Requested exposure time in seconds (calculated from EC_EINDE and exposure table)
E_TTN		Rev. number of exposure table
EXPMPAS	'single'; 'multi'	Single or multipass exposure
E_FW1_P	'Open'; 'Al_poly'; 'C_poly'; 'Be_thin'; 'Be_med'; 'Al_med'	Filter Wheel 1 position
E_FW2_P	'Al_mesh; 'Ti_poly'; 'Gband'; 'Al_thick'; 'Be_thick'	Filter Wheel 2 position
CCD_TEMP		CCD temperature; $t_c = -95.853 + 0.55376 t_{raw} + 5.9941 \cdot 10^{-5} t_{raw}^2$
CCD_TMPC		CCD temperature, derived from CCD_TEMP
CCD_READ	0, 1	CCD readout port; 0: right, 1: left
READPORT	'L'; 'R'	CCD readout port
CHIP_SUM	1, 2, 4, 8	On-chip pixel summation for CCD; 1: 1×1, 2: 2×2, 3: 4×4, 4: 8×8
CAL_INFO	0, 1	CCD image type; 0: Calibration image, 1: Observation image
CALIMAGE	'CAL' ; 'OBS'	CCD readout port (from CAL_INFO)
POS_COL		CCD column number of start of image (original value multiplied by 8 to get number of pixels)
POS_ROW		CCD row number of start of image (original value multiplied by 8 to get number of pixels)

XRT Level 0 FITS Header Keywords

Keyword	Value	Definition
ROI_H_SI		ROI horizontal size; 1: 64, 2: 128, 3: 192, 4: 256, 6: 384, 8: 512, 12: 768, 16: 1024, 24: 1540, 32: 2048; (original value multiplied by 64 to get number of pixels)
SIZ_COL		Horizontal size of ROI, derived from above; value is 0 if ROI_H_SIZE is reserved
ROI_V_SI		ROI vertical size; 1: 64, 2: 128, 3: 192, 4: 256, 6: 384, 8: 512, 12: 768, 16: 1024, 24: 1540, 32: 2048; (original value multiplied by 64 to get number of pixels)
SIZ_ROW		Vertical size of ROI, derived from above; value is 0 if ROI_V_SIZE is reserved
RECTIFY		Status of rectification to put solar south-east corner at the start of the CCD image
RPOS_COL		The rectified coordinate, equivalent to POS_COL, as though the image had been read out with this coordinate. If READPORT=R, RPOS_COL=POS_COL; otherwise RPOS_CPOS_COL.
RPOS_ROW		Rectified POS_ROW. Always the same as POS_ROW.
RSIZ_COL		Rectified SIZ_COL. Always the same as SIZ_COL.
RSIZ_ROW		Rectified SIZ_ROW. Always the same as SIZ_ROW.
EFFPORT		Rectified readout port
FOC_POS	-2500 to 2500	Focus position
BITCOMP1		Compression table keyword
IMGCOMP1		Compression table keyword
QTABLE1		Compression table keyword
BITC_VER	2	Bit compression lookup table version
ACHF_VER	76	AC Huffman table version
DCHF_VER	15	DC Huffman table version
QTAB_VER	0-7	Quantization table version
PCK_SN0		Data packet keyword
PCK_SN1		Data packet keyword
NUM_PCKS		Data packet keyword
HKTSYNC		True if fields derived from housekeeping data have been updated. (That is, they are not missing from the database.) Default is false.
DATE_OBS	'YYYY-MM-DDThh:mm:ss.sss'	UTC time when exposure began ('T' is character 'T')
TIME-OBS	'hh:mm:ss.sss'	Same value as DATE_OBS, but in a different format
CTIME	'DOW MON DD hh:mm:ss YYYY'	Example: 'Mon Mar 19 00:02:11 2007'; Same value as DATE_OBS, but in a different format

XRT Level 0 FITS Header Keywords

Keyword	Value	Definition
DATE_END	'YYYY-MM-DDThh:mm:ss.sss'	UTC time when exposure began ('T' is character 'T')
CRPIX1	(NAXIS1+1)/2	Horizontal location of the image center in pixel coordinates
CRPIX2	(NAXIS2+1)/2	Vertical location of the image center in pixel coordinates
SC_ATTX		Spacecraft attitude in *longitude*
SC_ATTY		Spacecraft attitude in *latitude*
CRVAL1		Number of arcseconds of the center of the sun from the reference position in the azimuthal direction (E-W); positive is to Solar West
CRVAL2		Number of arcseconds of the center of the sun from the reference position in the elevation direction (N-S); positive is to Solar North
CDELT1	PLATESCALE × SUMROW	Horizontal pixel size
CDELT2	PLATESCALE × SUMCOL	Vertical pixel size
CUNIT1		Horizontal units
CUNIT2		Vertical Units
CTYPE1		Type of units (label) of horizontal axis
CTYPE2		Type of units (label) of vertical axis
SAT_ROT		Difference between Solar north and *y*-axis of the satellite
INST_ROT		Difference between spacecraft *y*-axis and image *y*-axis
CROTA1		Angle between *x*-axis of image (same as *x*-axis of CCD) and E-W axis of heliocentric coordinates (SAT_ROT + INST_ROT)
CROTA2		Angle between *y*-axis of image and N-S axis of heliocentric coordinates (SAT_ROT + INST_ROT); CROTA1 and CROTA2 are identical for XRT
XCEN		*X*-coordinate of center of field of view
YCEN		*Y*-coordinate of center of field of view
XSCALE		Same as PLATESCL
YSCALE		Same as PLATESCL
FOVX		Width of field of view *x*-axis; equivalent to NAXIS1 × CDELT1
FOVY		Width of field of view *y*-axis; equivalent to NAXIS2 × CDELT2
PLATESCL		Platescale, in units of arcseconds per pixel

Bibliography

Afshari, M., Peres, G., Jibben, P. R., et al. 2016, AJ, 152, 107 [ADS]

Feldman, U. 1992, Physica Scripta, 46, 202 [ADS]

Golub, L., Deluca, E., Austin, G., et al. 2007, Sol. Phys., 243, 63 [ADS]

Kano, R., Sakao, T., Hara, H., et al. 2008, Sol. Phys., 249, 263 [ADS]

Kobelski, A. R., Saar, S. H., Weber, M. A., McKenzie, D. E., & Reeves, K. K. 2014, Sol. Phys., 289, 2781 [ADS]

Narukage, N., Sakao, T., Kano, R., et al. 2011, Sol. Phys., 269, 169 [ADS]

Narukage, N., Sakao, T., Kano, R., et al. 2014, Sol. Phys., 289, 1029 [ADS]

Shimizu, T., Katsukawa, Y., Matsuzaki, K., et al. 2007, PASJ, 59, 845 [ADS]

Takeda, A., Yoshimura, K., & Saar, S. H. 2016, Sol. Phys., 291, 317 [ADS]