



FIRS data reduction code manual

July 2013 C. Beck

Update: Jan 2014 C. Beck

Contacts:

Data reduction software: Christian Beck cbeck@nso.edu; Serena Criscuoli scriscuo@nso.edu

Instrument PI and instrument control software: Haosheng Lin lin@ifa.hawaii.edu

Technical: Haosheng Lin lin@ifa.hawaii.edu; Christian Beck cbeck@nso.edu; Serena Criscuoli scriscuo@nso.edu; lvtt@nso.edu

Contents

1	Introduction	2
1.1	Software installation	3
1.2	Software compatibility	3
2	General data properties	4
2.1	Data format	4
2.2	Header information	4
2.3	Observation types	4
3	Data reduction steps (general)	5
3.1	Data overview	5
3.2	Flat fielding	5
3.3	Polarimetric calibration	5
3.4	Reduction of observations	6
3.5	Utilities and data visualization	6
4	Data reduction with the code	7
4.1	Data overview routine get_firs_dataprop.pro	7
4.2	Flat field routines firs_1083_ff.pro and firs_630_ff.pro	8
4.3	Polarimetric calibration routine firs_cal.pro	12
4.4	Routine for reduction of observations firs_1565_obs.pro and firs_630_obs.pro	15
4.5	Data visualization/control of results with firs_view.pro	16
5	*Creation of overview archive	17
6	General comments on programs	18
7	Troubleshooting	19
A	File header example	20

B DST ICU retardance measurements	21
C Call to code for reducing data of one day	21

1 Introduction

This manual describes the IDL data reduction code provided by the National Solar Observatory (NSO) for data of the Facility Infrared Spectropolarimeter (FIRS; University of Hawaii; see <http://kopiko.ifa.hawaii.edu/firs/>). The set of IDL routines is aimed at performing the basic steps of the data reduction to obtain science-ready data. The following points were the guidelines in its development:

- As robust in the execution as possible.
- As few interpolations of (raw) data as possible.
- Use a modular set of routines to allow to exchange subroutines and/or to improve individual subroutines without modifying the global structure of the programs.
- Use separate subroutines for any major tasks that are needed repeatedly in the execution of the code or which could be useful outside of the FIRS data reduction.
- Use IDL procedures/functions for major/minor tasks.
- As automatic as possible. Include interaction with the user only when necessary. At the same time, the user can request full control again by deactivating the automatic processing via keywords.
- Allow control of execution by plotting all relevant quantities. Store any important quantities derived during the execution in separate save files for an eventual separate, detailed investigation.
- Stick to the convention of using parameters for all quantities that are required to run a routine and keywords for all parameters that are optional (not obeyed consistently).
- Provide documentation/explanations in the source code (not obeyed consistently).

The following points are considered in the basic data reduction:

- Dark current correction.
- Flat fielding, i.e. removal of persistent small-scale/large-scale intensity inhomogeneities. This does NOT include: a) removal of time-varying interference fringes, b) correction for intensity trends in the spectral dimension (e.g. prefilter transmission curves).
- Polarimetric calibration including correction of $I \rightarrow QUV$ cross-talk. Residual $QUV \leftrightarrow QUV$ cross-talk is only corrected optionally with user-provided values.

Additionally, these points are NOT considered in the data reduction:

- Removal of spectral curvature.
- Removal of trends of the global intensity/light level with time.
- Correction for interference fringes.
- Correction for non-linearity of the CCDs.
- * Relative distortions of the two orthogonally polarized beams (spatial or spectral axis).
- * Explicit correction for dead pixels (IR camera).

All points with an * might eventually be implemented at a later time.

The reasons for not including some of the points above were a conflict with the required robustness (e.g. avoiding the possibility to fail under certain circumstances such as for observations close to/off the limb), the necessity to adjust settings to each individual data set (e.g. fringes), or the general complexity of any implementation of the correction in a robust way.

We note that most of the points not taken into account can be considered as “data cosmetics” because they mainly affect the intensity Stokes I and are only important for any physical parameter derived from the intensity, such as temperature, Stokes I line-of-sight velocities, etc... . Apart from the correction for the dark current (subtraction), all steps applied to the observations are done multiplicative, i.e. they preserve the ratio QUV/I throughout the full data reduction.

The following sections explain the general structure of FIRS data and the different observation types (Sect. 2), the theory and execution of individual data reduction steps (Sect. 3), the usage of the programs (Sect. 4) and a list of problems often encountered during usage of the data reduction code (Sect. 7).

! Important !

The routines for 630 nm have not been updated and might not work with recent FIRS data in that channel (change of modulation sequence in 2013).

The IDL programs cal_261013_1565.pro and cal_200113_1083.pro execute the data reduction for the corresponding days and were tested with the current version of the software to process correctly. Please have a look at these two in case you get stuck.

! Important !

1.1 Software installation

Unpack the firs_soft.tar file into your home directory or any other folder. The folder will contain all IDL routines used in the code apart from those implemented in IDL itself, a README.TXT file with a short summary of the code and its usage, an html help page created with mk_html_help that lists the routines alphabetically and shows their program headers, all images necessary for the (on-line) html data overview archive, the most recent telescope matrix parameter file for the DST, and this manual.

Read the README.TXT file and change the paths as described, i.e. the file paths in the routine **laodct.pro** and the default telescope matrix file at the start of the routines **firs_630_obs.pro** and **firs_1565_obs.pro** (in case you do not want to provide this file name explicitly every time).

Depending on your IDL version and installation, you will also most likely need to add the firs_soft folder among the “preferred paths” of IDL before running the code. You can add the directory to your IDL path, e.g. using `!path = '~/firs_soft:'+!path` inside IDL.

1.2 Software compatibility

The code has been developed and meant to run under Linux OS. Most likely it will not work under Windows OS because of slightly different handling of window properties and strings. It has not been tested yet on MAC OS. A version for Windows is under development.

For any questions, comments or suggestions, please refer to the list of contacts provided above.

2 General data properties

2.1 Data format

FIRS data are stored in individual FITS files following the standard FITS convention. Each file contains four (one) images corresponding to the four modulation states observed in the spectropolarimetric (spectroscopic) mode. For data taken after April 2013, eight images are stored in polarimetric mode.

The current (July 2013) filename convention is:

firs.x.yyyymmdd.hhmmss.iiii.oooo

where x = 1 (630 nm channel) or 2 (IR channel), yyyy = year, mm = month, dd = day, hh = hour, mm = min, ss = seconds, iiii = counter of steps in the observation, and oooo = counter of repetitions of the observation sequence. A typical observation sequence of, e.g., 149 scan steps produces a list of files such as firs.2.20130120.112041.0000.0000, firs.2.20130120.112041.0001.0000,..., firs.2.20130120.112041.0149.0000.

The file name convention and data format of FIRS has changed over the years. If you would like to reduce data not stored in the format above and cannot modify the routines accordingly by yourselves, please contact cbeck@nso.edu or scriscuo@nso.edu.

FIRS files can be read with any FITS read routines. For the set of routines provide here, a slightly modified version of fxread.pro from the Solar Soft library (?) is used (called ffxread.pro; catching read errors was improved).

2.2 Header information

Appendix A shows an example of the information included in the FITS file headers. The header can be read in into IDL with, e.g., ffxread,file,data,header,/NODATA . The header string can be parsed with, e.g., exp_time = FXPAR(header,'EXP_TIME'). All header entries starting with “DST_” contain information provided by the DST telescope control system. Some of these are needed in the data reduction to calculate the polarimetric correction for the telescope polarization. Currently the type of the observation (dark current, flat field, solar observation, target, etc...) can only be found in the “Comment” field of the header in an sometimes inconsistent format and thus has to be determined manually.

An extended documentation on all header entries used by the camera control system(s) at the DST is available, but not included here because of its size. In case you need more information, please contact me (cbeck@nso.edu).

2.3 Observation types

In a standard observation day, the following types of data are taken with FIRS:

- Flat field (ff): taken with random motion of telescope and defocussed telescope/AO mirror. Default number of images/files is 32.
- Dark current (dc): taken with port-4 dark shutter inserted. Default number of images/files is 16.
- Polarimetric calibration (cal): taken with port-4 instrument calibration unit (ICU), one full rotation of the ICU retarder in 10 deg steps with the polarizer fixed at 0 deg and 45 deg (two sets of each 36 images/files). In case you are missing cal data at 45 deg (older data sets), the present code runs with the set at 0 deg alone.
- Solar observations: variable number of steps/files depending on settings chosen by the PI.

* Air force target, grid line target, dot grid target: on request by the PI, usually taken at least once per campaign. Variable number of steps/files depending on setup (single- or multi-slit). These data are usually taken in spectroscopic mode with the modulation deactivated.

The data reduction requires one dc and one ff measurement, and one set (36 images) of cal measurements. In case any of these data is not available from the same day, the data reduction code can use files from a different day. This is not recommended at all for dc and ff, and only marginally recommended for cal data.

3 Data reduction steps (general)

FIRS can take data in spectroscopic (polarimeter deactivated) or spectropolarimetric mode. The description given in the following section refers mainly to spectropolarimetric data, while the data reduction code can deal with both types of data internally.

FIRS can be operated with a single, double, or quadruple slit. All slits are automatically treated separately like being individual single-slit observations. The location of the spectra corresponding to individual slits is determined as first step of the flat fielding procedure.

The FIRS reduction code processes the data in four main steps that are described below (Sects. 3.1 to 3.4). Some utility routines are also provided for data visualization purposes (Sect. 3.5).

3.1 Data overview

Currently, the FIRS FITS file headers do not allow to extract the type of data automatically. The user has to manually identify which file is what as input for the data reduction code. To this extent a routine exists that parses all files in a given folder (without subdirectories) to extract the necessary header information. It provides a list of the so-called “base files”, i.e. the unique file names of each data sequence, their settings, number of steps pertaining to the sequence, etc..., and writes the collected information to a separate text file.

3.2 Flat fielding

From the flat field data, a gain table is constructed to correct for static small-scale and large-scale spatial inhomogeneities in the spectra. The method implemented uses average spectra from the ff data as reference and thus is not able to correct for intensity trends in wavelength, fringes or any other feature that is present in the average spectra as well. The gain table is derived from a division of a single profile, corresponding to one detector row, with an average profile. A set of average profiles along the slit is used to take variations in the spectral shape along the slit into account. Large-scale intensity trends along the slit are taken into account by normalizing the average profiles to each other. For the division, the spectral position of the average profile relative to a single profile is optimized by an iterative method to minimize residuals of spectral lines in the gain table. The same methods are used for IR and 630 nm channel, but for 630 nm some settings are chosen such that the telluric line at 630.20 nm is used explicitly in some steps.

3.3 Polarimetric calibration

3.3.1 Instrument calibration

The calibration sequence of FIRS with a fixed polarizer and a full rotation of the ICU retarder decouples the instrumental calibration from the telescope polarization (cf. Beck et al. 2005, A&A, 437). The telescope polarization can thus be ignored at this step. The calibration sequence leads to a simple linear input-output system of measured polarization states and created polarization input. The only critical issue in this context is the knowledge of the retardance of the ICU retarder to construct the input polarization during the calibration sequence. The current version of the code cannot determine the ICU retardance from the single set of cal measurements (36 images). The retardance can be determined explicitly if four additional measurements of $\pm QU$ are taken manually after the standard calibration (request of PI to observers needed). Standard retardance values for the ICU are about 87 deg at 630 nm, 83 deg at 1083 nm, and 110 deg at 1565 nm (cf. Appendix B, or the curve provided in Socas-Navarro et al. 2011, A&A, 531, 2; their Fig. 5). The linear input-output system is solved by a matrix inversion (cf. Beck et al. 2005, A&A, 437, Appendix A.2).

3.3.2 Telescope calibration

For correcting the polarization introduced by the telescope optics, i.e. all optics upfront of the port-4 ICU polarizer, the polarimetric telescope model of the DST is used (Skumanich et al. 1997, ApJS, 110; Socas-Navarro & Elmore 2011, A&A, 531, 2). The corresponding part of the code was developed from the already existing data reduction software of the IBIS instrument and uses routines pertaining to the IBIS data reduction pipeline (courtesy of A. Tritschler). The telescope model is based on the physical parameters of telescope properties that are determined regularly about once per year from SPINOR/DSLIP data. The resulting set of parameters is stored in an IDL save file that should be located in the FIRS software directory (currently Tmatrix_May2010_4708-14125.idl, to be updated). The offset angle between telescope model and ICU was set manually to 0 degrees for FIRS (keyword `offang = 0.`, see below).

3.4 Reduction of observations

The reduction of observations requires that beforehand a gain table was calculated (locations of beams in spectra, creation of dc save file, gain table) and a polarimetric response function was derived (X-matrix save file). After determining the locations of spectral lines and continuum wavelength ranges in the first step, the code processes all other steps/files of one observation sequence automatically. Any input gain, dc, and X-matrix files can be used even if they are from a different day than the observations to be processed. Spectroscopic data is processed by adding three dummy variables to simulate spectropolarimetric data. The dummy variables are then not written out at the end.

The observational data are

- corrected for the dc
- divided by the gain table
- rotated to compensate the CCD orientation relative to the spectral dispersion direction
- demodulated and multiplied with the inverse response function
- corrected for the telescope polarization
- corrected for $I \rightarrow QUV$ cross-talk
- written out as unformatted floating arrays to a single file.

The final data format was chosen because the file size can exceed 1 GB when stored as IDL save files, which can lead to problems on computers with small RAM. In the unformatted output format, one can read in individual steps. The endianness of the respective operating system may require to apply a `swap_endian` when reading the data in again on a different machine.

Note that the original file header information is NOT maintained in the reduced data.

3.5 Utilities and data visualization

***HTML overview archive** From the reduced FIRS data, overview maps can be generated for an inclusion in an HTML-based (online) archive. The archive is structured by years and days and is usually helpful for keeping track of specific data sets taken in extended observation campaigns. The archive can be browsed with any standard browser and can be simply used as off-line stand-alone feature.

***Display routine** There exists one routine for displaying reduced FIRS data that allows one to investigate the data quality in individual spectra or across the observed field of view. The display routine currently uses one of the files created by the archive software, but can be modified easily to skip this.

4 Data reduction with the code

This section explains in detail the usage of the four main routines that execute the four main data reduction steps described above. These routines are usually sequentially called by a main procedure, an example of which is given in Appendix C because some routines require the results of a previous step. Re-running individual steps can then be done in any order as long as the necessary results/save files from the previous steps are available.

From past experience, it is recommended to use a separate main procedure to reduce the data of one day, with a suited file name such as e.g. 'cal.date_wavelength.pro' (some examples inside the `firs_soft/` folder). This allows one to keep track of specific settings used for data sets. During an observing campaign, one can create main procedures for other data/days by copying and modifying an old file.

If not already done, you have to first open the routine `laodct.pro` in the `firs_soft/` folder, and change the variable `filepath` in the routine to the folder with the `firs_soft/` routines and the corresponding save files (`filepath = '.../firs_soft/'` in line 4 of `laodct.pro`) before using the code. This routine provides two more rainbow-like color tables (tables 41 and 42) that are not part of the standard IDL installation in most cases. It can be used like the standard `loadct.pro` otherwise (`laodct,nm; nm = 0...42`).

4.1 Data overview routine `get_firs_dataprop.pro`

Calling sequence:

```
input_dir = '/export/cbeck/obs2013/200113/firs/' ; e.g.
get_firs_dataprop,input_dir,change_save =change_save
```

This routine parses over all files located in `input_dir` and extracts their properties from the FITS file header information. All FIRS data files of the day should be located in the same folder without subdirectories. Also, do not change the standard FIRS file names. If the keyword `change_save` is not specified, it creates one text and one save file in `input_dir` whose names are derived from the date of the observations (e.g., 2013113.1.sav, 2013113.1.txt). In case that you do not have write permission in `input_dir` or you want store the output files elsewhere, use the keyword `change_save` to deviate the output files. The output save file '... .sav' contains information on the file name in the variable `bbasefile1` that is needed by all following routines, while the other variables stored here contain timing information.

Next, check the output text file with any text editor. It should look like this:

No.	File base	date	start - end time	?	exp.	naccum	steps	comment
0 Op	firs.2.20131003.073417.	2013/10/3	13:34:17 - 13:34:38	0	125	4	4	dark coadd 4, loop 16
1 Op	firs.2.20131003.083345.	2013/10/3	14:33:45 - 14:40:28	0	125	10	32	sft ID 517
2 Op	firs.2.20131003.084131.	2013/10/3	14:41:31 - 14:44:48	0	125	10	16	dark ID 517
3 Op	firs.2.20131003.085058.	2013/10/3	14:50:58 - 15:33:12	0	125	10	190	scan 1 ID 517
4 Op	firs.2.20131003.093616.	2013/10/3	15:36:16 - 15:43:0	0	125	10	32	sft ID 517
5 Op	firs.2.20131003.095002.	2013/10/3	15:50:2 - 16:32:24	0	125	10	190	scan 2 ID 517
6 Op	firs.2.20131003.103302.	2013/10/3	16:33:2 - 17:15:35	0	125	10	190	scan 3 ID 517
7 Op	firs.2.20131003.111627.	2013/10/3	17:16:27 - 18:30:59	0	125	20	190	scan 4 ID 517
8 Op	firs.2.20131003.123320.	2013/10/3	18:33:20 - 18:40:3	0	125	10	32	sft ID 517
9 Op	firs.2.20131003.125158.	2013/10/3	18:51:58 - 18:55:13	0	125	10	16	dark ID 517
10 Op	firs.2.20131003.125601.	2013/10/3	18:56:1 - 19:1:49	0	125	20	16	dark - coadd 20 ID 517
11 Op	firs.2.20131003.130806.	2013/10/3	19:8:6 - 19:16:34	0	125	10	36	pcal - Polarizer at 0 degrees ID 517
12 Op	firs.2.20131003.131726.	2013/10/3	19:17:26 - 19:26:0	0	125	10	36	pcal - Polarizer at 45 degrees ID 517
13 Op	firs.2.20131003.133047.	2013/10/3	19:30:47 - 19:34:10	0	125	2	190	targ ID 517
14 Op	firs.2.20131003.133550.	2013/10/3	19:35:50 - 19:39:19	0	125	2	190	lgrd ID 517
15 Op	firs.2.20131003.135533.	2013/10/3	19:55:33 - 20:6:30	0	1000	2	32	lft ID 517
16 Op	firs.2.20131003.140753.	2013/10/3	20:7:53 - 20:18:50	0	1000	2	32	lamp dark ID 517

The column “?” is a placeholder for the eventual automatic detection of the observation type. Now, identify the types of observations manually from either the file size or any notes that you took during the observations. In case you varied the exposure times, note also which observations share the same settings. As input for the data reduction, you will mainly need the corresponding number (IDL counting) of the base file listed in the first column. In the example above, observation “0” was a solar observation (or any type of target scan), “1” a ff (32 images), “2” a dc (16 images), “3” and “4” observations, “5” a second ff, “6” a second dc,..., and “11” the first polarimetric calibration set (36 images, polarizer at 0 deg), and “12” the second polarimetric calibration set (36 images, polarizer at 45 deg). In case you cannot decide the type from the file size, you will have to read in some files belonging to the corresponding base file and

decide the type either from the header entries or the appearance of the spectra. Either leave the text file open or note down the identification somewhere (see the example routine in Appendix C that reduces the data of that day).

Update Jan 2014:

The comment field of the header should contain (2013 or newer) a coded description of the data type:

sfft: solar flat field data, random walk of telescope, AO mirror defocussed
dark: dark current
scan: science observation
pcal: polarization calibration series
targ: air force target scan
lgrd: line grid scan
dgrd: dot grid scan
lfft: lamp flat field

The routines have not been modified to make use of these entries because also aborted calibrations or bad flat fields can be present, so it is not reliable without filtering out all usable data before reduction. You can use the additional information, however, to figure out which data set is what.

Note:

The save file created by `get_firs_dataprop.pro` has to be the FIRST save file in the corresponding folder when its contents are listed alphabetically with the `ls` command. If written to `input_dir` with the FIRS raw FITS files, this works fine (usually no second save file there), and in `output_dir` it usually works because the filename starts with a number opposite to all other files that the FIRS reduction code creates. If this save file is not found correctly, all of the following routines will crash because the base file names are stored in it. Sorry, this is a slight flaw that I saw too late. If necessary, modify the name of this file to an unique version and update the other routines accordingly to find it again (lines below “; restore file with data information” in the main routines described below).

4.2 Flat field routines `firs_1083_ff.pro` and `firs_630_ff.pro`

Note: the `firs_1083_ff.pro` is used for BOTH IR wavelengths, also for 1565 nm data. `firs_1083_ff.pro` and `firs_630_ff.pro` differ slightly internally because the code makes explicit use of the telluric line at 630.20 nm for 630 nm data, but the calling sequence is identical. The routines were kept separate to facilitate easier improvements of one of them.

Calling sequence:

```
firs_1083_ff,input_dir=input_dir,output_dir=output_dir,no_av = no_av,beam_ang = beam_ang,plt = plt,  
verbose = verbose, int_limit = int_limit,fringes = fringes,filepath = filepath,flat_file = flat_file,dark_file  
= dark_file,avdc_only = avdc_only,sspeed = sspeed,change_save = change_save,swap = swap,quit_column  
= quit_column,yrange = yrange,pickx = pickx,check_lines = check_lines,nopause = nopause,sshift =  
sshift,shifts_manual = shifts_manual,help = help
```

The parameters and keywords are detailed in the HTML help file or the header at the beginning of the routine that can be displayed by `firs_1083_ff/help` in IDL. You will not need most of the keywords. Keywords that are given in the program header text with a “/” such as `/plt` are only on-off switches, while all others can have one or more values.

A commonly used call is:

```
firs_1083_ff,change_save = change_save,flat_file = 1,dark_file = 2,/plt,input_dir = input_dir,output_dir =  
output_dir,int_limit = .1,/quit_column
```

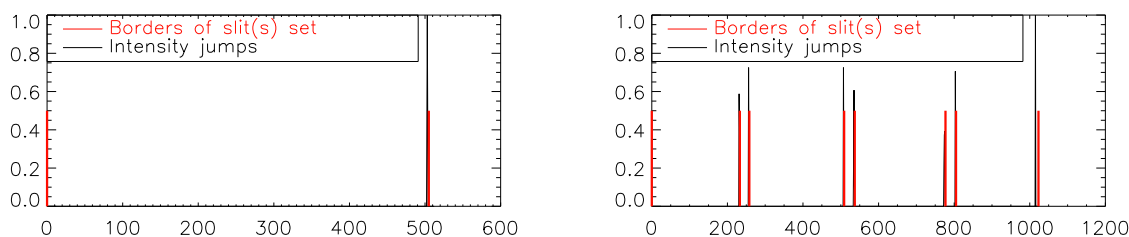



Figure 1: Successful detection of a single slit (left) and a quadruple slit (right) in the spectral dimension.

`flat_file = 1` and `dark_file = 2` denote the number of the corresponding base file in the list generated in the previous step (see above). It is recommended to keep `/plt` set all the time to see intermediate and final results. The keyword `/nopause` suppresses all stop/pause commands in the code, but the code will still execute all plot/display commands.

Setting the keyword `int_limit` might be required for the automatic detection of the slits and beam locations. If the automatic detection fails, change `int_limit` in steps of 0.05 (fraction of maximal intensity $0 < \text{int_limit} < 1$). If no settings produce a correct location of the slits/beams, overrule the automatic detection by explicitly providing the height of the two beams along the slit with e.g. `yyrange = [0,511,512,1022]/2.` and `/pickx` (manually select location of slits in spectral direction). With `/pickx`, you will be asked for the number of slits by the code and will have to mark the left and right border of the spectra belonging to each of the n slits in a full image.

Update Jan 2014:

Added keyword `/shifts_manual` provides an overrule over the shift/alignment derived from correlating the two beams. It was needed because of some dead pixels that screwed up the correlation in one beam in 1565 nm 4-slit data. If using the keyword, accept the shifts from the correlation, and then you will be prompted for the correct values directly afterwards.

With the above call to the routine, the code will first average the dc and ff files. The averages are stored in two separate IDL save files. Running the call with the keyword `/no_av` skips the averaging and uses previously created save files. The keyword `/avdc_only` only averages the dc files and then aborts.

The code will then try to determine automatically the number of slits in the data (x-direction in the spectra) using the intensity threshold given by `int_limit` (fraction of maximum intensity). This can be overruled with `/pickx`. Figure 1 shows how a successful automatic detection of the number of slits should look like for the case of single and quadruple slit data.

In the next step, the location of the two beams along the slit(s) are determined (overrule by providing `yyrange`). The combined result of the location of slits and beams should look like in Fig. 2 (quadruple slit example in 630 nm). The upper and lower beams of each slit are then clipped to have all the same size.

A possible beam rotation is determined next. The rotation angle provided by the code has turned out to be unreliable sometimes, so the user must judge whether the returned number makes sense. Note that the rotation angle value is not used further in this step of the code (`firs_XXX_ff.pro`) nor the next main step (polarimetric calibration; `firs_cal.pro`). It is instead an input for the final step (reduction of observations) and can there be provided manually by the user.

Update Jan 2014:

The default setup for FIRS now (2013 and later) uses a hairline in front of the slit. You can use the hairline in the spectra to determine the rotation angle with high precision outside of the data reduction itself, and provide the angle explicitly. Please check the sign if you use it.

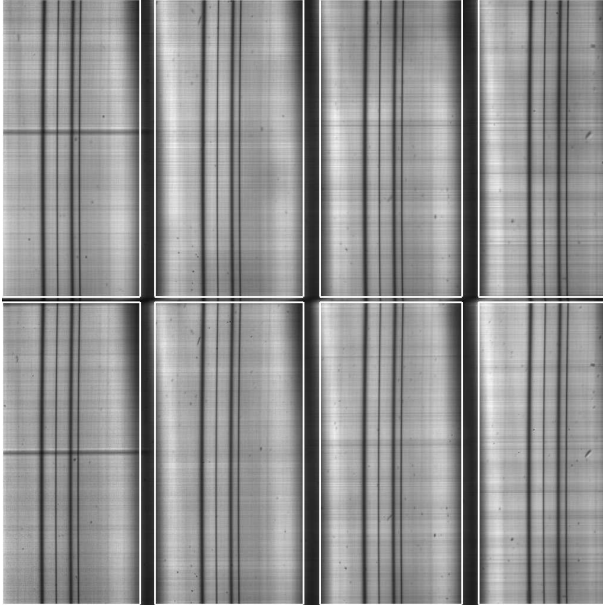


Figure 2: Successful location of slits and beams in x and y for a quadruple slit (630 nm data).

Next, for each slit, the upper and lower beam are aligned to each other with pixel precision. In case of more than one slit and a difference in the derived vertical shift between any of the slits of more than 10 pixels, an initial shift value for the correlation is requested. Set it to the most likely/most common number (e.g. if the shifts are 10, 20, 20, 21 \rightarrow enter 20 as initial value).

The equally-sized, aligned beams are then cut out and treated individually to obtain a gain table correction for each of them. The positions and dimensions of the beams are stored. The same numbers are later applied to the calibration data and observational data. They provide a reliable pixel-precision alignment of the two beams.

To calculate the gain table, the code will then try to identify spectral lines in the the spectrum. If this fails or you have set the `/check.lines` keyword, you will be requested next to mark spectral lines manually. The lines are flipped upside down for that purpose. You have to mark 2/4 lines (IR/630 nm channel). You do not have to mark the line core with high precision because some range around the marked position will be used. The first/second line (IR/630 nm channel) from the left border that you click onto will be used to determine the line-core position along the slit to deskew the spectra, to align average and individual profiles, and to mark the left end of the spectral range used in the iterative improvement of the gain table. The last/last-but-one line (IR/630 nm channel) towards the right that you click onto denotes the other end of the spectral range used in the iterative improvement of the gain table. The best correction is usually obtained inside the spectral range used for optimization, i.e. inbetween the two lines marked (IR) and the first and third line marked (630 nm).

For the IR channel, use the Si I line at 1082.7 nm as first line (alignment of spectra), and then encompass an as large as possible spectral range by clicking on the last line to the right covered in the spectrum. For 1565 nm, click either on 1564.8 nm as first line or pick one of the lines more to the left (if covered in the spectrum). As second line, pick 1565.2 nm or any line more to the right (if covered in the spectrum). For 630 nm, always click on Fe I 630.15, O₂ 630.20, Fe I 630.25, and O₂ 630.27 nm. Figure 3 shows how the line identification should look like for a 1083 nm single-slit (large spectral range) and a 630 nm quadruple-slit observation (small spectral range).

Since the spectral lines in each slit can be/are displaced from each other, you will have to mark the lines again once per slit. The rest of the calculation of the gain table is automatic, but takes its time. You will get one gain table per slit and beam. The results are all stored together in one save file. The calculation of the gain table has only to be done once for each ff sequence.

Figure 4 shows examples of gain tables for single-slit 1083 nm data and quadruple-slit 630 nm data (top row). The line residuals in the IR gain table are acceptable, while in 630 nm there are as good as none.

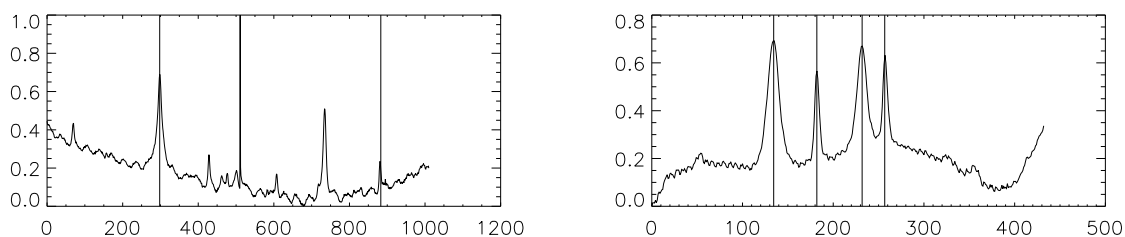


Figure 3: Correct line identification in single-slit 1083 nm data (left) and a quadruple-slit 630 nm data (right).

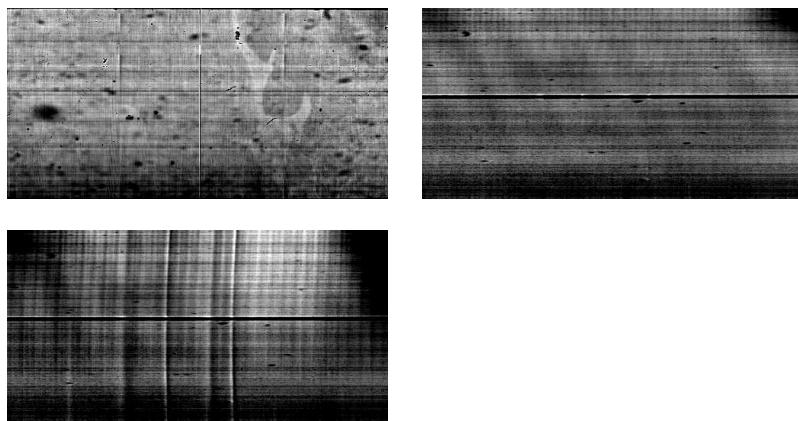


Figure 4: Gain table for one beam and one slit in single-slit 1083 nm data (top left, acceptable; display range 0.85 to 1.15) and a quadruple-slit 630 nm data (top right, good to excellent). Bottom: “bad” gaintable for quadruple-slit 630 nm data.

The lower panel shows a “bad” and unacceptable gain table. In case you get such strong line residuals, you can/should

- reduce the value of `speed` to 0.1 or similar
- make sure that the leftmost line you click on is correct (in the IR channel; for 630 nm, O₂ 630.20 nm should be the second line from the left you marked) and can be used for spectral alignment
- vary (extend or reduce) the range used for the optimization (location of second line marked in IR channel; 630 nm data has no options, but usually is fine).

In the end of the gain table calculation and with `/plt`, the input ff data, the corrected ff data and the corresponding gain table are displayed. Check the corrected image for homogeneity of the intensity along the slit as second characteristic of a good gain correction.

Notes:

1. The average profiles are calculated for a number of overlapping ranges along the slit. The corresponding sizes of the individual subdivisions along the slit and their overlap are currently hard-wired inside `create_gaintables_firs_630.pro` (routine is used for both IR and 630 nm channel) in the variables `rr` (location of subdivisions) and `sz` (overlap). Providing the `row` keyword to `create_gaintables_firs_630.pro` overrides the default values. In case you get an uneven intensity along the slit or persistent line residuals that cannot be mitigated by any of the points above, you can try your luck with the `row` keyword. The settings for the subdivisions are actually taken from the POLIS software library and have not been changed since 2003.
2. `create_gaintables_firs_630.pro` with its subroutines is a stand-alone routine that calculates a gaintable from a ff image, dc, the position of a single spectral line, and some spectral range for the iterative improvement. Feel free to use it for other data if you want to or find it useful.
3. In the calculation, as a first step a “coarse” gain table is calculated. This uses the average profile

along the full slit without subdivisions. The coarse gaintable is stored as well in the gain table save file. Under certain circumstances, the coarse gain table can be even better than the “fine” gain table. The coarse gain table has the advantage of providing a perfectly uniform intensity along the slit because only one average profile is used. It is, however, not used by default in the current code. The “bad” gain table in Fig. 4 is actually the coarse gain table at 630 nm, while the coarse gain table for the IR data in Fig. 4 had slightly smaller line residuals than the “fine” one shown.

The routine creates the following outputs (the sizes of some variables can vary depending on the number of slits, and whether the polarimeter was used in the observation or not):

```
dc.firs. ... .sav:
dc: average dark current image

ff.firs. ... .sav:
ff_img: average flat field image

gain.firs. ... .sav:
aangle: beam rotation angle
cchi: chi square values of determination of rotation angle
dx, dy: size of beams in image
posx, posy: position of beams/slits in image
dx_final, dy_final: size of clipped and aligned beams in image
posx_final, posy_final: position of clipped and aligned beams/slits in image
gaintable: gain table for all beams/slits
gaintablecoarse: 'coarse' gain table for all beams/slits
n.slits: number of slits
ppos_ref: reference position of spectral line used for alignment in beams/slits
mean.slits: average intensity in beams/slits
mean.slitscoarse: average intensity in beams/slits when coarse gain table is used
mean.slits1: average intensity in beams/slits after balancing of intensity
int.only: 0 polarimetric 1 spectroscopic
```

4.3 Polarimetric calibration routine firs_cal.pro

The same routine is used for all wavelengths. Many of the keywords are identical to before. The routine is fully automatic and offers little options for influencing the result apart from the retardance of the ICU retarder and the number of subdivisions along the slit.

Calling sequence:

```
firs_cal,spat_res=spat_res,no_av=no_av,plt=plt,ilimit=ilimit,retard=retard,theta_err=theta_err,theta_pol=
theta_pol,gainindex=gainindex,darkindex=darkindex,cal_file=cal_file,change_save=change_save,input_dir=
input_dir,output_dir=output_dir,flat_file=flat_file,swap=swap,add_cal=add_cal,nverbose=nver-
bose,cchi=cchi,n_separations=n_separations,nopause=nopause,quit_column=quit_column,help=
help
```

Example call:

```
firs_cal,/plt,input_dir=input_dir,output_dir=output_dir,cal_file=11,darkindex=2,change_save=change_save,theta_err=
5,retard=82.
firs_cal,/plt,input_dir=input_dir,output_dir=output_dir,cal_file=12,darkindex=2,change_save=change_save
,theta_err=5,theta_pol=45.,retard=82.
```

The most important difference in the correct call to the routine is that now the numbers given in `dark-index` or `gainindex` do not refer to the list of files in `input_dir`, but the dc and gain save files in `output_dir`

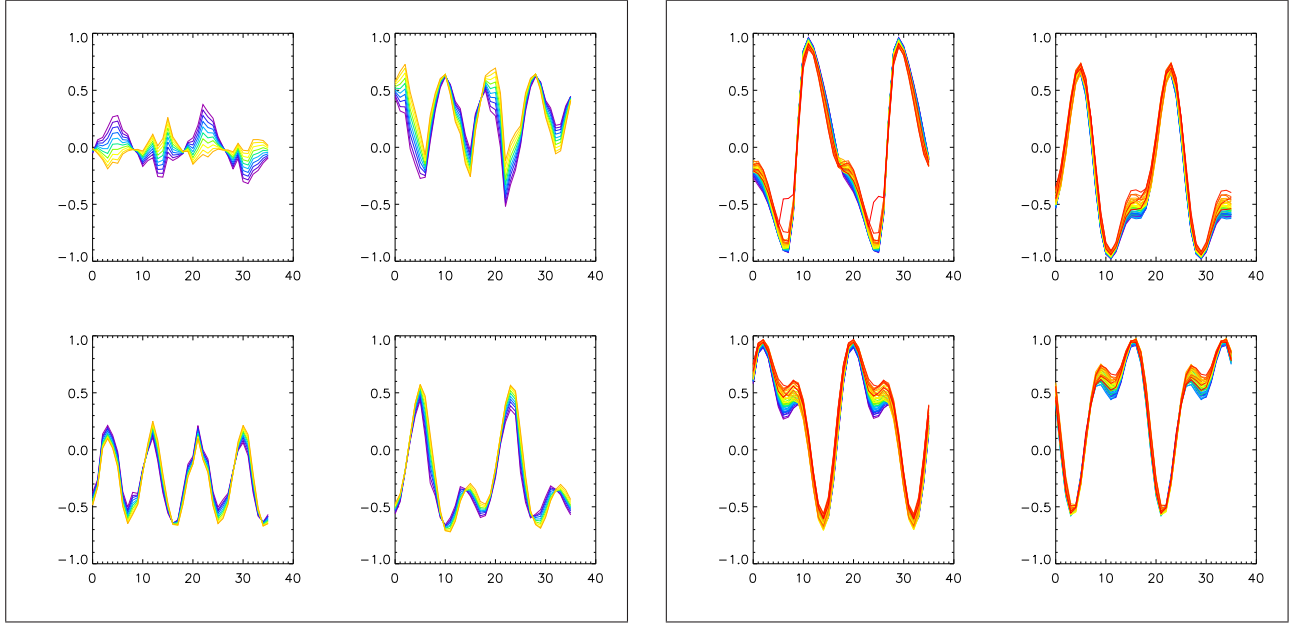


Figure 5: Calibration measurements with 10 subdivisions along the slit for single-slit 1083 nm data (left four panels) and quadruple-slit 630 nm data (right). The curves show the corresponding values for the 37 ICU positions during the calibration sequence. After changing to the usage of demodulated data, the upper left panel shows the intensity now (2013 and later; image was not created anew).

instead. E.g., in the call above the 3rd dc file in `output_dir` is used (IDL counting). Pick the dc/gain that is closest in time to the calibration data. Note that if the polarizer angle is not correct, the result will be wrong. (* One actually could read the file headers to get it; maybe I'll add that somewhen).

`theta_err` is the offset angle between polarizer and retarder. Its value can be guessed rather easily from the displacement between observed and best-fit curves. Change in steps of 1–5 deg if necessary.

`retard` is the retardance of the ICU retarder at the wavelength of the observations (cf. Sect. 3.3.1 above and Appendix B for the appropriate values).

The code reads in the calibration data, cuts out the beams as defined in the gain save file used, and extracts the calibration curves. Figure 5 shows two examples, where the left panel is a corrupt calibration, while the right panel is a valid calibration. If any of the four modulation states does not show a systematic variation (upper leftmost panel in Fig. 5) or the curves in any two of the states are identical/very similar, something went wrong with the modulation in the observations. In the best case, the four modulation states show completely differently shaped curves. The larger the amplitude, the better was the efficiency of the modulation.

Update Jan 2014:

The code was changed to using demodulated data (2013 or later). Thus *IQUV* is used instead of modulation states. The intensity curve should show minor variation around a roughly constant value and is given in counts or normalized to average intensity, respectively.

To additionally control the polarimetric response, use `/plt` and have a look at the last two windows that compare the Stokes input with X^{-1} · the observed output and the observed output with X · the Stokes input. The curves should match halfway well, apart from Stokes I (see Fig. 6). A more direct estimate of the quality of the polarimetric calibration is provided by the efficiencies for measuring a given polarization state that are printed on the terminal. Efficiencies around or above 0.3 (= 30 %) are acceptable, those below 0.2 are suspicious in general, and values of 0.1 or lower are unacceptable and indicate some serious

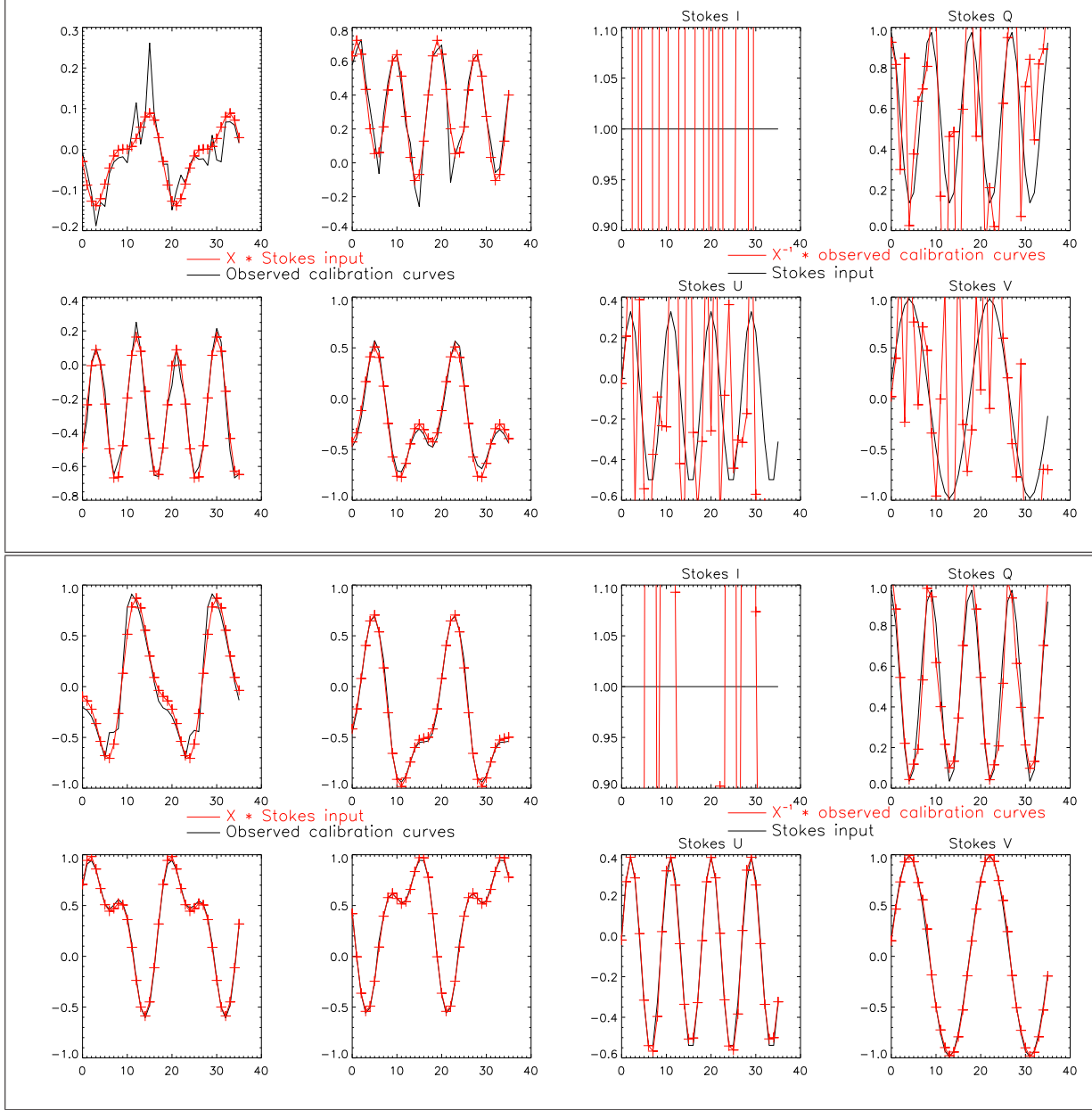


Figure 6: Calibration measurements for single-slit 1083 nm data (top panels) and quadruple-slit 630 nm data (bottom panels). The curves show the corresponding values for the 37 ICU positions during the calibration sequence. The 1083 nm calibration is flawed and gives a really bad fit in the backward direction ($X^{-1} * \text{observations}$, upper right panels). The forward direction looks okay even for the flawed 1083 nm calibration ($X * \text{input}$) because that is the direction whose χ^2 is minimized.

problem. In the reduction of the data, the inverse of the response is applied, which, e.g., for the bad calibration multiplies the observed images by $1/\sim 0.1 > 10$ (!), blowing up the noise incredibly.

When reducing the second calibration data set with the polarizer at 45 deg, the resulting response functions and efficiencies have to be nearly identical to those of the first set.

The routine creates the following outputs:

```
polvec.firs. ... .sav:
cal_curves: observed values during the calibration sequence

xmat.firs. ... .sav:
theta_err: offset angle between retarder and polarizer of the ICU
n_separations: number of subdivisions along the slit
yyr: location of the subdivisions along the slit
xxmat: X matrix for n_separations subdivisions
retard: retardance value used for the ICU retarder
```

4.4 Routine for reduction of observations `firs_1565_obs.pro` and `firs_630_obs.pro`

The main reduction routine uses again several of the keywords of before. It has several keywords for fine-tuning that usually are not needed. `firs_1565_obs.pro` is used for both IR wavelength ranges. Data taken after April 2013 can have a hairline in the spectra for alignment purposes. For older data without a hairline, the keyword `/no_align` has to be set.

Calling sequence:

```
firs_630_obs,plt = plt,cont_wl = cont_wl,v2q = v2q, v2u = v2u,beta = beta,obsindex = obsindex,gainindex
= gainindex,scan_range = scan_range,verbose = verbose,angle = angle,darkindex = darkindex,cal_file =
cal_file,change_save = change_save,tel_cal = tel_cal,input_dir = input_dir,output_dir = output_dir,check_lines
= check_lines,ccal_file = ccal_file,flat_file = flat_file,swap = swap,tmat_save = tmat_save,outfile = oout-
file,wavelength = wavelength,notelcor = notelcor,pos_lines = pos_lines,offang = offang,no_align = no_align,
quit_column = quit_column,hair_up = hair_up, hair_down = hair_down,check_hairlines = check_hairlines,average_cal
= average_cal,catch_dead = catch_dead,smooth_width = smooth_width,disp_scale = disp_scale,help = help
```

Several of the keywords overrule default settings:

```
ccal_file, flat_file = use IDL save file with response function / gain table from a different day (including
full filepath to the file)
/tel_cal = do not apply telescope and  $I \rightarrow QUV$  correction for telescope calibration data
/notelcor = do not apply telescope correction
outfile = use different output file name (including full filepath to the file)
The values that should be set are
```

```
wavelength = 1083. or 1565. for the IR data
angle = rotation angle of beams, if in doubt set to 0
tmat_save = file name of parameter file for telescope model (including full filepath to the file)
/no_align = in this case, neither hairlines nor spectral lines are used for subpixel alignment, but a corre-
lation of the two beams instead
```

Example call:

```
firs_1565_obs,/plt,input_dir = input_dir, output_dir = output_dir,cal_file = 11,darkindex= 0,gainindex
= 0,obsindex = 0,tmat_save = '/home/cbeck/firs_soft/Tmatrix_May2010_4708-14125.idl',wavelength =
10830.,offang = 0.,angle = 0.
```


For running the reduction of other data on the same day, only the values of [darkindex](#), [gainindex](#), and [obsindex](#) have to be modified. The keyword [pos_lines](#) allows one to provide the position of the spectral lines so that the code can run fully automatically over all data of one day.

! Important !

For data taken during the first cycle of the DST service mode in Jan/Feb 2013, the calibration unit was vignetting the beam during calibration. You have to set the keyword [/average_cal](#) to use only the response of the upper part of the beam.

! Important !

The final reduced data is written out as unformatted floating. For reading them in again correctly, check the beginning of `firs_view.pro`. You have to restore the save file that is created by `firs_1565_obs.pro` or `firs_630_obs.pro` for each reduced data set to get the dimensions of the data (`dx_final + 1`, `dy_final + 1`) and the number of slits (`n_slits`). If the polarimeter was activated, the value of `int_only` is 1, otherwise it is 0.

Example:

```
if int_only eq 0 then begin
if n_slits eq 1 then a = ftarr(xdim,ydim,4) else a = ftarr(n_slits,xdim,ydim,4)
endif else begin
if n_slits eq 1 then a = ftarr(xdim,ydim) else a = ftarr(n_slits,xdim,ydim)
endelse

openr,in_unit,infile,/get_lun
readu,in_unit,a
...
free_lun,in_unit
```

Note: the data reduction routine does not sort the slits, so the first image contains the data from the first scan step and all slits. See `firs_view.pro` for how to sort them out again, you're clever guys.

The routine creates the following outputs:

firs.dat: reduced data file, unformatted floating numbers

firs.dat.sav:

ic: map of continuum intensity

totq, totu, totv: maps of wavelength-integrated unsigned *QUV* signal

sshifts: storage array of applied shifts for beam alignment (obsolete)

llinepos: line-core position of spectral line that was used for alignment

pos_solar: wavelength region that was used for creation of *QUV* maps

ccont_wl: wavelength region that was used for $I \rightarrow QUV$ cross-talk correction

4.5 Data visualization/control of results with `firs_view.pro`

FIRS data reduced with this set of routines can be read and displayed with the program `firs_view.pro`. Because of the total size of some reduced data files, the routine only reads in single scan steps by default, but has a keyword to load the full data file to memory (if it is sufficient).

Calling sequence:

```
firs.view,infile,swap = swap,full_data = full_data,width = width,binning = binning,v2q = v2q,v2u
= v2u,test_cal = test_cal,spec_width = spec_width,limits = limits
```

Example call:

```
firs.view,'firs.2.20131003.095002.reduced.dat'
```

The keywords `v2q`, `v2u` can be used to test the residual cross-talk from $V \rightarrow Q, U$. The correction is only applied to the spectra and profiles of the scan step that is active and not to the 2D $|QUV|$ maps if the complete file is not loaded into memory (`/full_data`). `width` and `spec_width` control a spatial and spectral smoothing by x pixels. `binning` bins the the data by its value to fit better on small screens. The program is controlled by clicking into the window labeled “Logarithmic maps”. The left mouse button displays the profiles of the corresponding location in the map and the scan step corresponding to the x-value. The right mouse button quits the program, and the middle mouse button switches between the continuum and a line-core image.

5 *Creation of overview archive

For the creation of the archive files, the archive has to be prepared before. The structure MUST be one folder per year, named with the year:

```
.../FIRS_archive/yyyy/
```

and subfolders for every day:

```
.../FIRS_archive/yyyy/ddmmyy/
.../FIRS_archive/yyyy/ddmmyy/
...
```

E.g., `/export/cbeck/FIRS_archive/2013/200113/`, `/export/cbeck/FIRS_archive/2013/210113/`, etc. ...

The html page is structured then automatically by years, month and days. Differences to the naming convention above will prevent the routine from working. The top level folder can be named arbitrarily.

With the keywords from above, the calling sequence is:

```
inpath = outpath
outpath = '/export/cbeck/FIRS_archive/2013/200113/'
generate.firs_archive_files,inpath = inpath, outpath = outpath,twice = 4,wavelen = 1083.,pollimit = .03
```

This will create one text and one gif file in outpath for each reduced data set , and one save file in inpath for each reduced data set. The corresponding save file is needed to run the program `firs.view.pro` to display one data set. You may have to select again lines as above in the ff procedure, but always only two solar lines. The archive software creates maps of line-core velocity for the two lines you mark and Stokes V polarity for one of them.

```
inpath = outpath
firs.create.html,filepath = inpath
```

This creates a html page from all gif and text files in inpath.

The next routine requires to know the location of the archive `inpath = '.../FIRS_archive/'` specifying the top level folder, and the correct output file `outfile = '.../FIRS_archive/FIRS_archivemain.html'`. You

have to copy the following files from the `firs_soft/` directory to your `.../FIRS_archive/` before: `example.gif`, `fears1.gif`, `fears.gif`, `NSO.gif`, and `example.ps`.

`firs.create_mainhtml,inpath=inpath,outfile=outfile`

The html pages are static html with all links relative to the top level folder `.../FIRS_archive/`.

The routines create the following output:

`generate_firs_archive_files:`

`... .reduced.dat.gif:` gif image of overview maps
`... .reduced.dat.txt:` header information
(one for each map)

`firs.create_html:`

`Overview.ddmmyy.html:` overview page for one day from all gif and txt files

`firs.create_mainhtml:`

`FIRS_archivemain.html:` main html file with links to all individual overview pages

6 General comments on programs

The following rules are roughly valid throughout the programs:

Variables consisting of double letters (e.g., `xx`) or starting with double letters are usually intended to be temporarily used only and might be recycled (exemptions: `tt`, `yy`).

Loop variables are `k`, `kk`, `kkk`, ... with usually counting loops from inside out with less letters, i.e. main loop is `k`, first sub-loop `kk`, etc.

Variable names usually correspond to the content of the variable, e.g., `gaintable` = gain table, `ff.img` = flat field image etc...

The programs are roughly documented inside the code itself.

Program header information might be incomplete, I haven't checked all later modifications.

Some of the keywords might have become redundant with the modifications of the code and the instrument. I have not removed them for the code, in case they might be again needed for some specific data sets.

IDL save files are indicated by the ending `'.sav'`. Unformatted data is denoted by `'.dat'`.

7 Troubleshooting

TBD. Need user input.

Problem	Reason	Solution
General:		
Code stops with messages like:		
'Can not read from/write to file : xx Please enter valid filename and hit return :'	A critical file check failed.	Check the file name and directory. You might also have chosen a non-existent file number in a list of files. The code will not proceed until a valid file name is given.
!read_directory: Sorry, can not read directory xx ! Directory does not exist or is a file !	A directory read failed.	Check the input/output folders. Code might proceed, but is likely to crash afterwards.
Terminal output of 'ls: No match.'	A file search found no match.	Usually uncritical, but check file names. Could lead to a subsequent crash.
Questions like 'rm: remove regular filetmp.remove_me ?' appear	rm is set to interactive mode	change the settings of rm on your session, or change spawn,'rm '+... to spawn,'rm -f '+... in read_directory.pro and check_existence.pro
Data reduction:		
Calibration data: '!! firs_cal: Warning !! Efficiencies above 0.866. Seems fishy.' or: ' Warning : Unphysical matrix !! '	Efficiencies or response function seem screwed up.	Check dc and cal file, and the retardance value. Check the locations of the beams cut out in the ff routine
Code stops in flat field or observational data reduction with: 'Warning ! Very high mean shift value of xx pixels.'	Shift values between beams have exceeded hard-wired limit in data alignment.	Re-run with other spectral line/hair line positions. Code will proceed, but the message usually indicates a fatal flaw. In case of reducing observations, try the /no_align keyword.
You get messages like: % SAVE: Undefined item not saved: MEAN_SLITS.	Variable was not used in data reduction.	Ignore. Some variables appear only in special data (e.g. multi-slit mode).
% Illegal subscript range: XYZ. % Execution halted at: DESKEW1 (routine may vary)	Array dimensions were exceeded.	Usually caused by bad/wrong definition of locations of spectral lines. Check line positions and/or spatial shifts.

A File header example

SIMPLE =	T /	Written by IDL: 20-Jan-2013 10:17:56.00
BITPIX =	16 /	16-bit twos complement binary integer
NAXIS =	3 /	
NAXIS1 =	1024 /	
NAXIS2 =	1024 /	
NAXIS3 =	4 /	
DATE_OBS=	'2013/01/20' /	observing date
TIME_OBS=	'10:17:56' /	observing time
OBS_STAR=	'2013-01-20T17:17:42.593' /	frames UTC start time
OBS_END =	'2013-01-20T17:17:56.015' /	frames UTC end time
EXP_TIME=	125 /	exposure time in msec
SUMS =	10 /	No. of images summed
LCVR0 0 =	3.22300 /	Liq. crystal variable retarder 0 setting #0 (mv)
LCVR1 0 =	1.19300 /	Liq. crystal variable retarder 1 setting #0 (mv)
LCVR0 1 =	3.22300 /	Liq. crystal variable retarder 0 setting #1 (mv)
LCVR1 1 =	2.19600 /	Liq. crystal variable retarder 1 setting #1 (mv)
LCVR0 2 =	2.22600 /	Liq. crystal variable retarder 0 setting #2 (mv)
LCVR1 2 =	1.63400 /	Liq. crystal variable retarder 1 setting #2 (mv)
LCVR0 3 =	2.22600 /	Liq. crystal variable retarder 0 setting #3 (mv)
LCVR1 3 =	3.51500 /	Liq. crystal variable retarder 1 setting #3 (mv)
INST ID =	'FIRS.2 ' /	
WAVELENG=	'10830 ' /	Observing wavelength
CAMERA =	'SEIR1K ' /	
NO. LOOP=	150 /	Total number of loop
LOOP =	106 /	Current loop Counter
NOREPEAT=	1 /	Total number of repeat
REPEAT =	0 /	Current repeat counter
SLITWIDTH=	0.0400000 /	Slit width in [mm]
FEED F/#=	'F/36 ' /	Feed Optics Configuration
FSM X0 =	-0.222250 /	Field Scanning Mirror Start Postion [mm]
FSM DX =	0.00363891 /	Field Scanning Mirror Step Size [mm]
FSM CP =	0.163474 /	Field Scanning Mirror Current Position [mm]
DST_TIME=	'2013-01-20T17:17:48.879' /	When DST info was last updated (UTC)
DST_AZ =	1.479162000000E+02 /	DST turret azimuth (degrees)
DST_EL =	3.044120100000E+01 /	DST turret elevation (degrees)
DST_TBL =	9.231033200000E+01 /	DST table position (degrees)
DST_GDRN=	5.358401700000E+01 /	DST guider angle (degrees)
DST_XGDR=	-4.612561200000E+02 /	DST guider X-axis position (arcsec)
DST_YGDR=	1.168439500000E+02 /	DST guider Y-axis position (arcsec)
DST_SLAT=	-1.511029400000E+01 /	Stonyhurst Heliographic solar lat. (deg)
DST_SLNG=	2.777134000000E+01 /	Stonyhurst Heliographic solar long. (deg)
DST_HPA =	2.477989800000E+02 /	Heliocentric position angle (degrees)
DST_GPA =	2.404290100000E+02 /	Geocentric position angle (degrees)
DST_RV =	4.878687800000E-01 /	Heliocentric radius vector
DST_PEE =	-7.369975100000E+00 /	Solar P angle (degrees)
DST_BEE0=	-5.108332400000E+00 /	Solar B0 angle (degrees)
DST_ELL0=	9.510758800000E+01 /	Solar L0 angle (degrees)
DST_SDIM=	1.950628200000E+03 /	Solar diameter (arcsec)
DST_LLVL=	5.209960900000E+00 /	DST guider light level
DST_SEE =	5.029296900000E-01 /	Seykora scintillation monitor value (arcsec)
DST_SAZ =	1.477630400000E+02 /	Solar azimuth (degrees)
COMMENT	FIRS map 2, South 15.2, West 27.5, 150 steps	

Table 1: Retardance measurements for the DST ICU.

Source	589	630	656	854	1083	1565
Elmore/design	86	83	82	82	93	111
Beck/FIRS	—	88	—	—	82	116
Beck/SPINOR	—	—	—	88	83	105
Sarah/Tom(?)/FIRS	—	77	—	—	87	102
Ali/IBIS	87	86	84	83	—	—

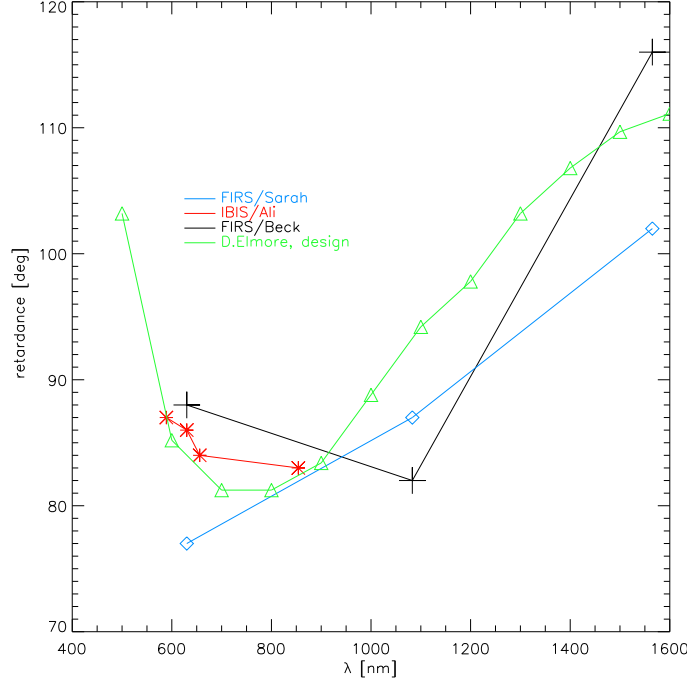


Figure 7: Retardance measurements for the DST ICU. *Black line/crosses:* FIRS data, Beck. *Red line/asterisks:* IBIS data, Ali. *Blue line/diamonds:* FIRS data, Sarah (or Tom ?). *Green line/triangles:* possible design, David.

B DST ICU retardance measurements

C Call to code for reducing data of one day

```

pro cal_031013_1565

; directory to input raw data
input_dir = '/data/smocycle2/03_october_2013/coconut/'

; directory for output data
output_dir = '/data/cbeck/031013/'

; change location of save file with data information if you do not have write
; permission in input_dir
change_save = '/data/cbeck/031013/'

pp = 1
if pp eq 1 then begin

get_firs_dataprop,input_dir,change_save =change_save

```

```

; check output text file for the numbers of flat, dc, cal, observations
; edit file numbers below accordingly

stop

; output created in change_save:
; 2013103.2.sav, 2013103.2.txt

endif

; coding for eventual automatic reduction, ignore
; 16 cal 2 map 1 ff 0 dc

; 2013 10 03
types1 = [0,1,0,2,1,2,2,2,1,0,0,16,16,2,2,1,0]

pp = 1
if pp eq 1 then begin

yyrange = [5,500,515,1016]/2.

firs_1083_ff,change_save = change_save,flat_file = 1,dark_file = 2,/plt ,input_dir =input_dir, output_dir
= output_dir,/swap,yyrange = yyrange,/pickx,/quit_column,sspeed = .5

firs_1083_ff,change_save = change_save,flat_file = 4,dark_file = 2,/plt ,input_dir =input_dir, output_dir
= output_dir,/swap,yyrange = yyrange,/pickx,/quit_column,sspeed = .5

firs_1083_ff,change_save = change_save,flat_file = 8,dark_file = 9,/plt ,input_dir =input_dir, output_dir
= output_dir,/swap,yyrange = yyrange,/pickx,/quit_column,sspeed = .5

firs_1083_ff,change_save = change_save,flat_file = 8,dark_file = 10,/plt ,input_dir =input_dir, output_dir
= output_dir,/swap,yyrange = yyrange,/pickx,/quit_column,sspeed = .5,/avdc_only

; output created in output_dir:
; dc.firs.2.20131003.084131.sav, dc.firs.2.20131003.125158.sav, dc.firs.2.20131003.125601.sav
; ff.firs.2.20131003.083345.sav, ff.firs.2.20131003.093616.sav, ff.firs.2.20131003.123320.sav
; gain.firs.2.20131003.083345.sav, gain.firs.2.20131003.093616.sav, gain.firs.2.20131003.123320.sav

firs_cal,/plt,input_dir =input_dir, output_dir = output_dir,cal_file = 11,darkindex= 1,gainindex = 1,change_save
=change_save, retard = 116./,swap,n_separations = 10./,quit_column,theta_err = 5.

firs_cal,/plt,input_dir =input_dir, output_dir = output_dir,cal_file = 12,darkindex= 1,gainindex = 1,change_save
=change_save, retard = 116./,swap,n_separations = 10./,quit_column,theta_pol = 45.,theta_err = 5./,nopause

; output created in output_dir:
; polvec.firs.2.20131003.130806.sav, polvec.firs.2.20131003.131726.sav
; xmat.firs.2.20131003.130806.sav, xmat.firs.2.20131003.131726.sav

endif

pp = 1
if pp eq 1 then begin

wavelength = 15650.

```



```

tmat_save = '/home/cbeck/firs_soft/Tmatrix_May2010_4708-14125.idl'
cal_file = 11

firs.1565_obs,/plt,input_dir=input_dir, output_dir=output_dir,cal_file=cal_file,darkindex= 0,gainindex= 0,change_save=change_save,obsindex= 3,tmat_save=tmat_save,wavelength=wavelength,angle=.0,/swap,offang= 0./,quit_column,pos_lines=pos_lines,/catch_dead

firs.1565_obs,/plt,input_dir=input_dir, output_dir=output_dir,cal_file=cal_file,darkindex= 0,gainindex= 1,change_save=change_save,obsindex= 5,tmat_save=tmat_save,wavelength=wavelength,angle=.0,/swap,offang= 0./,quit_column,pos_lines=pos_lines,/catch_dead

firs.1565_obs,/plt,input_dir=input_dir, output_dir=output_dir,cal_file=cal_file,darkindex= 1,gainindex= 1,change_save=change_save,obsindex= 6,tmat_save=tmat_save,wavelength=wavelength,angle=.0,/swap,offang= 0./,quit_column,pos_lines=pos_lines,/catch_dead

firs.1565_obs,/plt,input_dir=input_dir, output_dir=output_dir,cal_file=cal_file,darkindex= 2,gainindex= 2,change_save=change_save,obsindex= 7,tmat_save=tmat_save,wavelength=wavelength,angle=.0,/swap,offang= 0./,quit_column,pos_lines=pos_lines,/catch_dead

firs.1565_obs,/plt,input_dir=input_dir, output_dir=output_dir,cal_file=cal_file,darkindex= 1,gainindex= 2,change_save=change_save,obsindex= 13,tmat_save=tmat_save,wavelength=wavelength,angle=.0,/swap,offang= 0./,quit_column,pos_lines=pos_lines,/catch_dead

firs.1565_obs,/plt,input_dir=input_dir, output_dir=output_dir,cal_file=cal_file,darkindex= 1,gainindex= 2,change_save=change_save,obsindex= 14,tmat_save=tmat_save,wavelength=wavelength,angle=.0,/swap,offang= 0./,quit_column,pos_lines=pos_lines,/catch_dead

; output created in output_dir:
; firs.2.20131003.085058.reduced.dat, firs.2.20131003.085058.reduced.dat.sav, firs.2.20131003.095002.reduced.dat
; firs.2.20131003.095002.reduced.dat.sav, firs.2.20131003.103302.reduced.dat, firs.2.20131003.103302.reduced.dat.sav
; firs.2.20131003.111627.reduced.dat,firs.2.20131003.111627.reduced.dat.sav, firs.2.20131003.133047.reduced.dat
; firs.2.20131003.133047.reduced.dat.sav, firs.2.20131003.133550.reduced.dat, firs.2.20131003.133550.reduced.dat.sav

endif

inpath = output_dir
outpath = '/export/cbeck/FIRS_archive/2013/031013/'

generate_firs_archive_files,inpath = inpath, outpath = outpath,twice = 2,wavelen = 15650.,pollimit = .001

; output created in inpath:
; firs.2.20131003.085058.reduced.dat.velpol.sav, firs.2.20131003.103302.reduced.dat.velpol.sav
; firs.2.20131003.133047.reduced.dat.velpol.sav, firs.2.20131003.095002.reduced.dat.velpol.sav
; firs.2.20131003.111627.reduced.dat.velpol.sav, firs.2.20131003.133550.reduced.dat.velpol.sav

; output created in outpath:
; firs.2.20131003.085058.reduced.dat.gif, firs.2.20131003.085058.reduced.dat.txt
; firs.2.20131003.095002.reduced.dat.gif, firs.2.20131003.095002.reduced.dat.txt
; firs.2.20131003.103302.reduced.dat.gif, firs.2.20131003.103302.reduced.dat.txt
; firs.2.20131003.111627.reduced.dat.gif, firs.2.20131003.111627.reduced.dat.txt
; firs.2.20131003.133047.reduced.dat.gif, firs.2.20131003.133047.reduced.dat.txt
; firs.2.20131003.133550.reduced.dat.gif, firs.2.20131003.133550.reduced.dat.txt

```

```
inpath = outpath
firs.create.html,filepath = inpath

; output created in inpath:
; Overview.031013.html

inpath = '/export/cbeck/FIRS_archive/'
outfile = inpath + 'FIRS_archivemain.html'

firs.create.mainhtml,inpath = inpath,outfile = outfile

; output created in inpath:
; FIRS_archivemain.html

stop

end
```