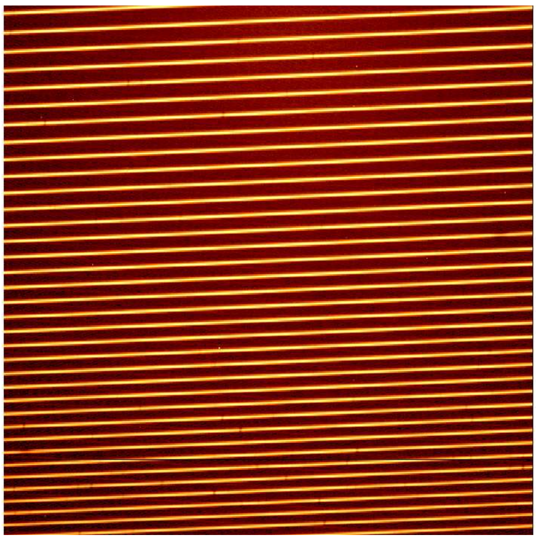
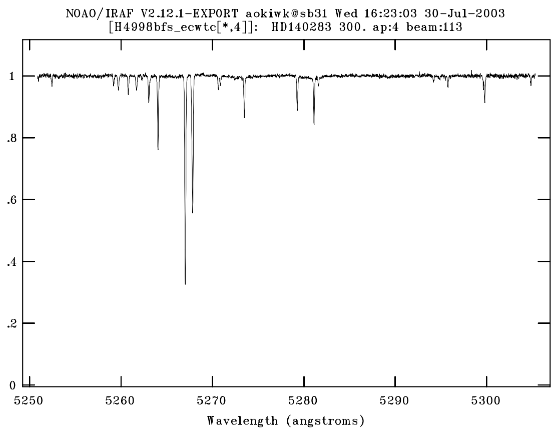
Data reduction of echelle spectra with IRAF

Version 1.2

October 2014




NOAO/IRAF V2.12.1-EXPORT aokiwk@sb31 Wed 16:23:03 30-Jul-2003
[H4998bfs_ecwtc[*,4]]: HD140283 300. ap:4 beam:113
1
.8
.6
.4
.2
0
5250
5260
5270
5280
5290
5300
Wavelength (angstroms)

Wako Aoki, Krzysztof Hełminiak

National Astronomical Observatory of Japan

Preface

This is a brief instruction for reduction of echelle data with IRAF (Image Reduction and Analysis Facility), in particular for data obtained with the High Dispersion Spectrograph at the Subaru Telescope. For details of the reduction with IRAF, readers are encouraged to refer to the IRAF web page[1] where manuals for the echelle package, as well as the CCD data reduction in general are available.

Examples of the parameter settings for IRAF tasks required for the echelle data reduction are given in this text. These will be useful for the first trial of the reduction for data from HDS and other spectrographs. However, the parameter setting is dependent on the data – please find better solution for yourselves.

Questions and comments for this text are welcome. The contact address is given below.

October 01, 2014

Wako Aoki
National Astronomical Observatory of Japan
2-11-1 Osawa Mitaka, Tokyo 181-8588, Japan
TEL: +81-422-34-3531 FAX:+81-422-34-3545
E-mail: aoki.wako@nao.ac.jp

Krzysztof "Kris" Hełminiak
Subaru Telescope, National Astronomical Observatory of Japan
650 N. A'Ohoku Pl., Hilo, HI 96720, USA
TEL: +1-808-934-7788 FAX:+1-808-934-5984
E-mail: xysiek@naoj.org

[1] http://iraf.noao.edu/

Contents

1 General reduction procedure

The goal of this text is to derive a wavelength-calibrated spectrum (Figure 2) from the two dimensional CCD image (Figure 1). The procedure includes calibrations of CCD data, extraction of the spectra, wavelength calibration, continuum normalization and combining spectra of individual echelle orders. An example of the flow of the reduction procedure is summarized as follows:

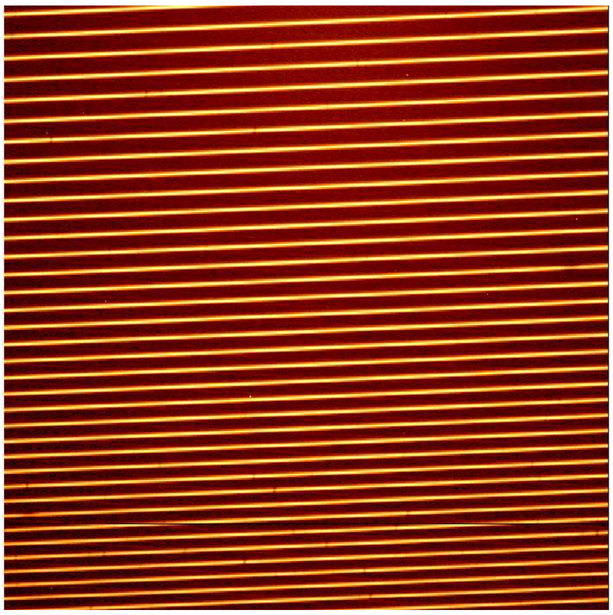


Figure 1: A two dimensional CCD image of object data. The horizontal and vertical axes are corresponding to the dispersion and slit directions, respectively.

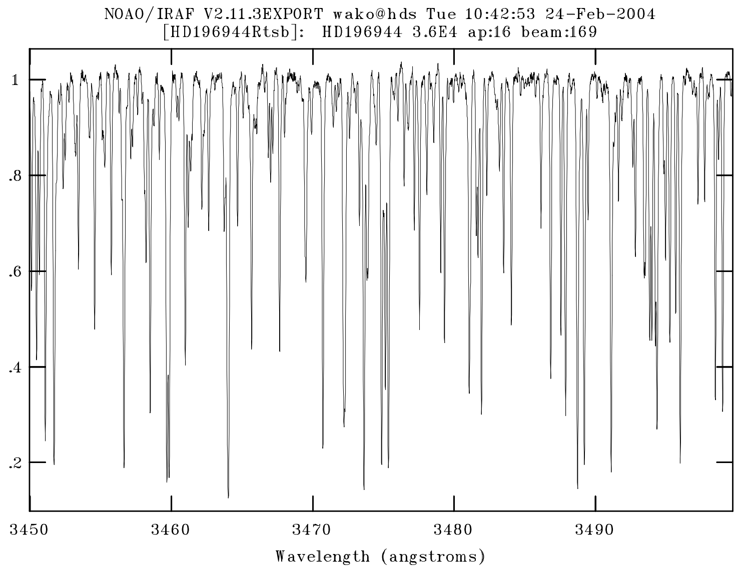
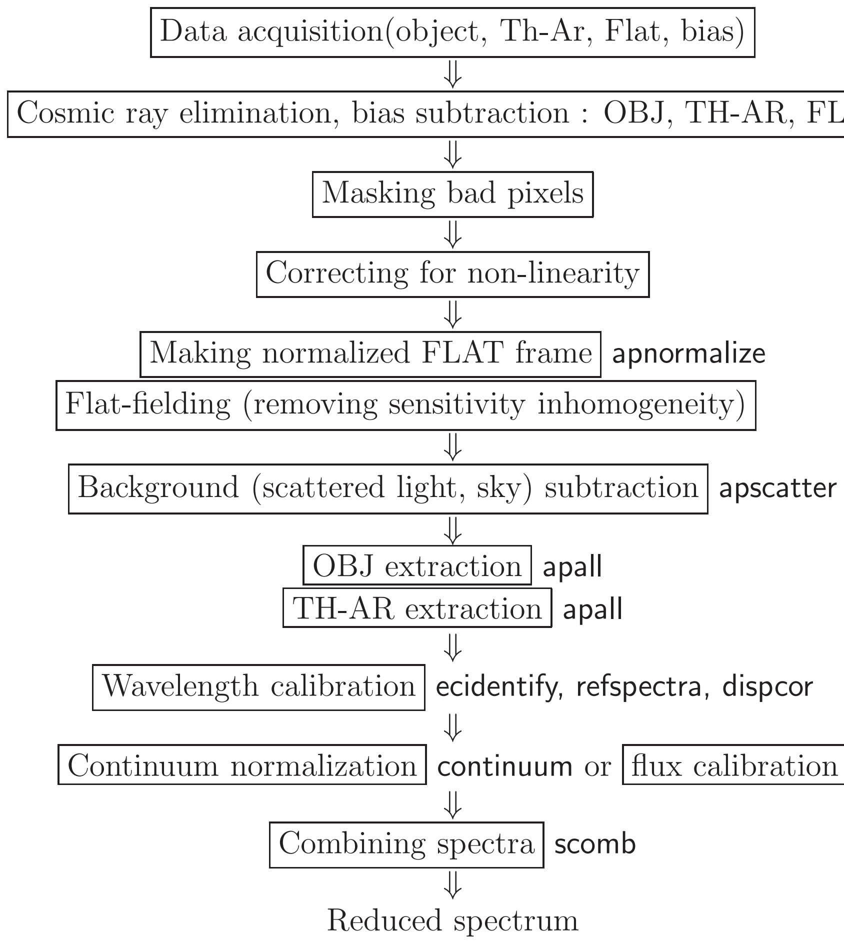


Figure 2: Portion of a one-dimensional, wavelength-calibrated, and continuum-normalized final spectrum.


Data acquisition(object, Th-Ar, Flat, bias)
Cosmic ray elimination, bias subtraction : OBJ, TH-AR, FLAT
Masking bad pixels
Correcting for non-linearity
Making normalized FLAT frame
apnormalize
Flat-fielding (removing sensitivity inhomogeneity)
Background (scattered light, sky) subtraction
apscatter
OBJ extraction
apall
TH-AR extraction
apall
Wavelength calibration
ecidentify, refspectra, dispcor
Continuum normalization
continuum or
flux calibration
Combining spectra
scomb
Reduced spectrum

2 Data preparation

In addition to the spectral data of objects, calibration data for bias subtraction, flat-fielding and wavelength calibration are usually obtained during observing runs. In the case of Subaru/HDS data, one may download data from the archive system (STARS and SMOKA). The data set should contain above frames. See Subaru's web page for details of the data archive system.

3 Setup of IRAF

First, a terminal (xgterm or xterm) is opened (the xgterm is recommended if available). Execute 'mkiraf' command, and then answer a few questions on the initialization of parameter files and setup of the terminal:

```
> mkiraf
Initialize uparm? (y|n): y
-- initializing uparm
Terminal types: xgterm,xterm,gterm,vt640,vt100,etc.
Enter terminal type: xgterm
A new LOGIN.CL file has been created in the current directory.
You may wish to review and edit this file to change the defaults.
```

Then, the file login.cl will appear on the directory you are working. This is a setup file for IRAF that can be modified manually. In the login.cl file one may find lines that look like this:

```
set     home            = "/home/username/IRAF_directory/"
...
set     uparm           = "home$uparm/"
```

The first one is the directory where login.cl is found, and the other defines the location of the catalog uparm, where current parameters of the tasks are stored. If one works on data from various instruments, we strongly suggest to create separate starting directories (separate login.cl-s and uparm-s) for each instrument.

It is convenient to set the default image type to FITS (*.fits) rather than IRAF format (*.imh and *.pix). In that case, modify the line on the image type from

```
#set     imtype          = "imh"
```

to

```
set     imtype          = "fits"
```

It is also possible to define packages and task that IRAF will load when starting, both already installed or user-made. Simply put their names at the end of the login.cl file.

```
imred
ccdred
echelle
...
task [name] = [path]/[filename].cl
```

The last line shows how to define additional tasks that are stored somewhere on the user's computer, outside the IRAF directory (usually *.cl files).

IRAF is started by the ecl or cl command which must be executed in the directory where the login.cl file exists. We recommend the ecl, as it allows for command callback (use arrows) and tab completion features.

```
> ecl
```

If no additional packages and tasks are loaded, the IRAF starts with a message similar to the following:

```
# LOGIN.CL -- User login file for the IRAF command language.
   NOAO/IRAF PC-IRAF Revision 2.16 EXPORT Thu May 24 15:41:17 MST 2012
      This is the EXPORT version of IRAF V2.16 supporting PC systems.

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

  Visit http://iraf.net if you have questions or to report problems.

  The following commands or packages are currently defined:

      apropos     images.     noao.       proto.      stsdas.     utilities.
      dataio.     language.   obsolete.   softools.   system.
      dbms.       lists.      plot.       spiral.     tables.
ecl>
```

Note that one can start IRAF in the directory where login.cl does not exist, but then no information written in the file is read by IRAF, and the performance of IRAF is significantly degraded.

The ecl> is the prompt of the IRAF system, and one can directly apply some UNIX command (e.g. cd, ls). Move to the directory where you hope to work using the cd command.

The reduction procedures are carried out interactively using IRAF 'tasks' (commands). The tasks are grouped into 'packages' depending on the roles. In order to execute a task, one first opens the package including the task by typing the package name[2]. For example, the task apall is located in the echelle package, which is located in the imred package. So, one first inputs the package names imred and echelle (or ec):

[2]A task can be executed by inputting only some part of the task name (or package name), if the name is distinguished from other tasks (or packages). For example, one can move to the echelle package by typing only eche or even ec.

```
ecl> imred
      argus.      crutil.     echelle.    iids.       kpnocoude.  specred.
      bias.       ctioslit.   generic.    irred.      kpnoslit.   vtel.
      ccdred.     dtoi.       hydra.      irs.        quadred.
im> ec
      apall         aprecenter    demos         refspectra    sflip
      apdefault@    apresize      deredden      sapertures    slist
      apedit        apscatter     dispcor       sarith        specplot
      apfind        apsum         doecslit      scombine      specshift
      apfit         aptrace       dofoe         scopy         splot
      apflatten     bplot         dopcor        sensfunc      standard
      apmask        calibrate     ecidentify    setairmass
      apnormalize   continuum     ecreidentify  setjd
ec>
```

One can go back to the previous package by bye. The package name that includes the task one hopes to apply is found by help. Here is the case of `apall` for example:

```
ecl> help apall
```

This results in the following message, where the package name `noao.twodspec.apextract` is given at the head. Other useful information is also included in the message.

```
APALL (Sep96)            noao.twodspec.apextract           APALL (Sep96)

NAME
    apall -- Extract one dimensional sums across the apertures


USAGE
    apall input


PARAMETERS

......
```

When one executes an IRAF task, some parameters along with the input and output files are required. The list of parameters is edited by the eparam task like eparam task-name. Examples are shown in the following sections.

4 Structure of HDS data and the over-scan regions

4.1 Characteristics of the HDS data

- Frame ID

  A serial number is assigned to each frame of HDS data. HDS produces two FITS files corresponding to the two CCDs by one exposure, for which two frame IDs are assigned. The ID consists of the instrument ID (HDSA) and the serial number (e.g., 00002480). In this case, the name of the FITS file is HDSA00002480.fits. Odd and even numbers correspond to the CCDs covering longer (red chip) and shorter wavelengths (blue chip), respectively. The number does not decrease: if the exposure is canceled and no FITS file is produced, the number is skipped.

- Characteristics of the FITS data produced with HDS

The HDS FITS file consists of the header unit, data unit, and ASCII extension tables and their header units. In the tables, the spectrum format of the obtained data (wavelength coverage of individual orders, position of the spectrum on the detector) is recorded. The format is calculated from grating angles etc.

4.2 Data format

The data unit contains the output of one CCD with 2048 (slit direction) by 4100 (dispersion direction) pixels, in the case that CCD on-chip binning is not applied, and the *over-scan* region. The *over-scan* indicates the additional readout to the CCD pixels exposed. The data in the over-scan region provides the bias level for the frame itself.

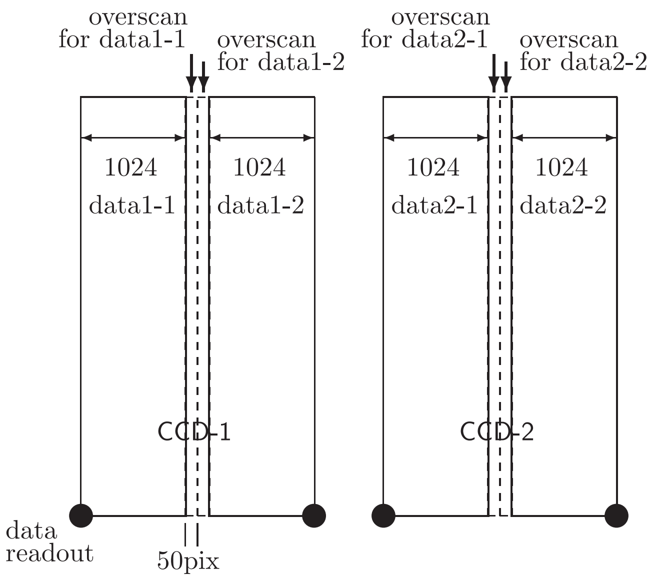


Figure 3: Schematic view of the HDS data format. The vertical and horizontal axes are corresponding to the dispersion and slit directions.

Figure 3 shows the data format of HDS. Since there are two readout ports for each CCD, the unit of data output originally consists of 1024 × 4100 pixels. The over-scan region (50 × 4100 pixels) is added to this data unit. One file consists of two units. Two files corresponding to the two CCDs (as shown in Figure 3) are obtained in one exposure. Note that the over-scan region of the same number of columns (50) is added to one data unit for the data with CCD on-chip binning. For instance, in the case of the 2×2 binning, the over-scan region of 50×2050 pixels is added.

The time variation of the bias level is corrected by using the data in the over-scan region. The method of the correction is explained in the next subsection.

4.3 Reduction method for HDS data (the first step)

1. Reading data neglecting ASCII extension tables

   The data unit of the FITS file is read by attaching [0] to the file name (e.g., *HDSA00000001.fits[0]*). Then the data and header units can be directly dealt with by IRAF.

   Alternatively, the task "rfits" reads the data unit from the original file and records it in a new FITS file:

   ex: rfits *input.fits* 0 *output.fits*

2. How to deal with the over-scan region

   As described in Section 4.2, the data of over-scan regions are attached to HDS data files. The over-scan data are useful to estimate the bias level for each CCD image. The bias level estimated from the over-scan region is subtracted from the real CCD image by the following procedure.

   - Calculate the average of the counts in the over-scan region (50×4100 pixels) for each unit of the CCD readout (1028×4100 pixels).
   - Subtract the above average from the data for each unit.
   - Multiply the data for each unit by the *gain*. The value of the gain for each output unit is given in Table 2.
   - Trim the effective data region off the whole data and combine to a single file (2048×4100 pixels).

   Note that the numbers of the pixels above are the case without CCD on-chip binning.

   The following table gives the values of gain (conversion factor) for individual readout units. These values are also given in the header unit of the FITS files (`H_GAIN1` for the data of longer wavelength, and `H_GAIN2` for the data of shorter wavelength).

   Table 1: Gain of CCD

unit of output	Gain (e^-/ADU)
CCD1, left(longer wavelength)	1.628
CCD1, right(shorter wavelength)	1.615
CCD2, left(longer wavelength)	1.782
CCD2, right(shorter wavelength)	1.665

   The IRAF script overscan.cl, which deals with the over-scan region as noted above, is available on the URL:

   `http://www.subarutelescope.org/Observing/Instruments/HDS/hdsql/overscan.cl`

   The task is defined as follows;

   ```
   ecl> task overscan=overscan.cl
   ```

   This script is executed with input and output file names as follows:

   ```
   ecl> overscan input.fits[0] output.fits
   ```

5 Displaying data

5.1 CCD image

A CCD image is shown by ds9 (SAOIMAGE). ds9 is started on an unix terminal with a FITS file name.

```
> ds9 filename.file &
```

Alternatively, the image is displayed from IRAF environment with ds9. The task name of this function is display. Following is an example to show the data of the file named H4998.fits:

```
ecl> display H4998.fits 1
z1=1420. z2=1967.862
```

The display ranges are adjusted by parameters of the task display.

```
ecl> epar display
```

Then, the following list of the parameters appears;

```
PACKAGE = tv
   TASK = display

image   =        ubc00478.fits  image to be displayed
frame   =                    1  frame to be written into
(bpmask =                 BPM) bad pixel mask
(bpdispl=                none) bad pixel display (none|overlay|interpolate)
(bpcolor=                 red) bad pixel colors
(overlay=                    ) overlay mask
(ocolors=               green) overlay colors
(erase  =                 yes) erase frame
(border_=                  no) erase unfilled area of window
(select_=                 yes) display frame being loaded
(repeat =                  no) repeat previous display parameters
(fill   =                 yes) scale image to fit display window
(zscale =                 yes) display range of greylevels near median
(contras=                0.25) contrast adjustment for zscale algorithm
(zrange =                 yes) display full image intensity range
(zmask  =                    ) sample mask
(nsample=                1000) maximum number of sample pixels to use
(xcenter=                 0.5) display window horizontal center
(ycenter=                 0.5) display window vertical center
(xsize  =                  1.) display window horizontal size
(ysize  =                  1.) display window vertical size
(xmag   =                  1.) display window horizontal magnification
(ymag   =                  1.) display window vertical magnification
(order  =                   0) spatial interpolator order (0=replicate, 1=linea
(z1     =                    ) minimum greylevel to be displayed
(z2     =                    ) maximum greylevel to be displayed
(ztrans =              linear) greylevel transformation (linear|log|none|user)
(lutfile=                    ) file containing user defined look up table
(mode   =                  ql)
```

The display ranges are adjusted by the values of the parameters xmag, ymag etc. In order to finish the parameter setting, press Ctrl+d, or type colon symbol (:) and subsequently q. If one types colon (:) and go, then the task itself (display in this case) starts its job.

5.2 Cross-cut image

A cross-cut image of the CCD data is useful to evaluate spectral data. This is displayed by the task implot. Here the data for flat fielding are shown as an example:

```
ecl> implot H4998.fits
```

A new window appears to show the cross-cut image of the data for the slit direction (Figure 4. Here, only a small portion of the data is shown. The range of the X-axis is from 0 to 2048, indicating the pixel number of the slit direction. In this case, the count (photon) level of the flat lamp is approximately 30,000. The slit length can also be measured from this image (about 30 pixels in this case). The display range is changed by inputting colon (:) and pixel numbers like 'x 500 1500', which results in a diagram showing the data between the given pixels. See help of the task implot for more details.

If one types 'c' on the window, a cross-cut image for the dispersion direction (i.e. direction that is perpendicular to the slit) is shown (c means column). A cross-cut image of the slit direction will be shown again by pressing 'l' (=line) on the window.

6 Basic corrections: bias, dark current, bad pixels and non-linearity

This chapter briefly summarizes the first reduction steps one should take when working on any kind of CCD data, not only spectroscopic. Some of them should be apply always, some are instrument-dependent. We will mention all, but focus on those suggested for the HDS.

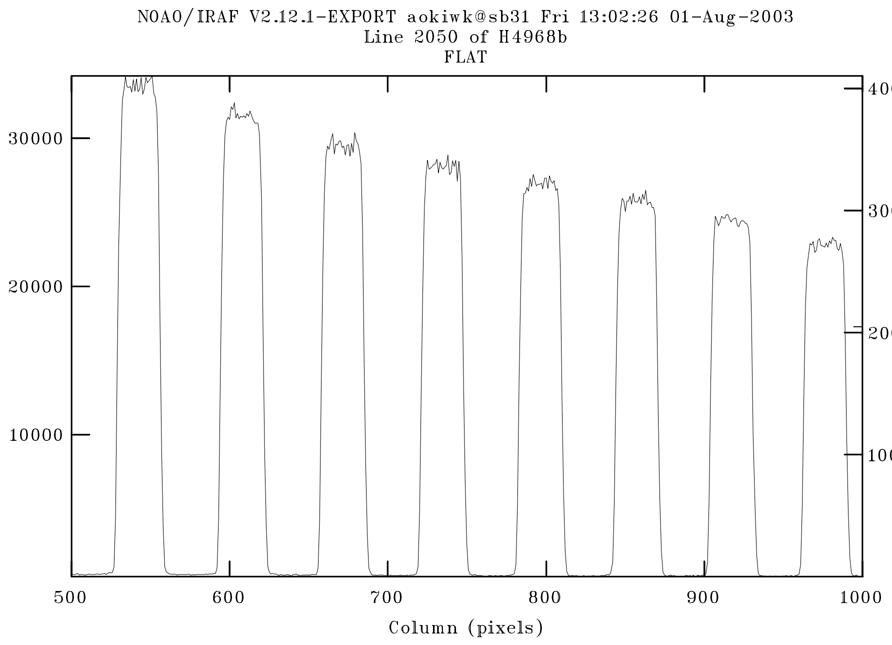


Figure 4: A portion of a cross-cut image of flat data for the slit direction. This is the case for the 2050th line (i.e. the center of the CCD image) for the dispersion direction.

6.1 Corrections for bias

CCD data obtained with no exposure will have some counts. These values are called *bias*. The values are dependent on the CCD pixels in general, though the differences are usually quite small. The corrections for bias are made using the frames obtained with no exposure. In order to increase the data quality, several frames should be combined by adopting the median of the values of each pixel (median, rather than average, is recommended because some pixels possibly show very high values caused by cosmic-ray noise). For this purpose, the task imcombine may be used. Following parameter table appears by executing 'eparam imcombine':

```
                                     I R A F
                    Image Reduction and Analysis Facility
PACKAGE = immatch
   TASK = imcombine

input   =             @bias_in  List of images to combine
output  =                 bias  List of output images
(headers=                     ) List of header files (optional)
(bpmasks=                     ) List of bad pixel masks (optional)
(rejmask=                     ) List of rejection masks (optional)
(nrejmas=                     ) List of number rejected masks (optional)
(expmask=                     ) List of exposure masks (optional)
(sigmas =                     ) List of sigma images (optional)
(logfile=               STDOUT) Log file

(combine=               median) Type of combine operation
(reject =                 none) Type of rejection
(project=                   no) Project highest dimension of input images?
(outtype=                 real) Output image pixel datatype
(outlimi=                     ) Output limits (x1 x2 y1 y2 ...)
(offsets=                 none) Input image offsets
(masktyp=                 none) Mask type
(maskval=                   0.) Mask value
(blank  =                   0.) Value if there are no pixels
```

```
(scale  =                 none) Image scaling
(zero   =                 none) Image zero point offset
(weight =                 none) Image weights
(statsec=                     ) Image section for computing statistics
(expname=                     ) Image header exposure time keyword

(lthresh=                INDEF) Lower threshold
(hthresh=                INDEF) Upper threshold
(nlow   =                    1) minmax: Number of low pixels to reject
(nhigh  =                    1) minmax: Number of high pixels to reject
(nkeep  =                    1) Minimum to keep (pos) or maximum to reject (neg
(mclip  =                  yes) Use median in sigma clipping algorithms?
(lsigma =                   3.) Lower sigma clipping factor
(hsigma =                   3.) Upper sigma clipping factor
(rdnoise=                   0.) ccdclip: CCD readout noise (electrons)
(gain   =                   1.) ccdclip: CCD gain (electrons/DN)
(snoise =                   0.) ccdclip: Sensitivity noise (fraction)
(sigscal=                  0.1) Tolerance for sigma clipping scaling correction
(pclip  =                 -0.5) pclip: Percentile clipping parameter
(grow   =                   0.) Radius (pixels) for neighbor rejection
(mode   =                   ql)
```

The input file names of the bias data like 'H4946,H4948,H4950,H4952,H4954' are given for input ('.fits' can be omitted), while a name like bias is given for output. Alternatively, for the input the file names can be given by an *input file* which gives a file name in each line. For example, the input file bias_in containing the above five file names of bias data is given for the input parameter as @bias_in, then the all bias data are dealt with input data for imcombine. The parameter combine should be median. Type :go, then the bias frame bias.fits is produced. The combined data should be confirmed using implot or some other tasks.

Input files should be previously trimmed and reduced for the over-scan. An alternative task – zerocombine – can do that automatically when producing the final bias image. This task first runs the procedure ccdproc, that processes each image according to the settings specified (epar ccdproc).

The bias frame is later subtracted from the object or other calibration frames. One can use the task imarith:

```
ecl> imarith H4998.fits - bias.fits H4998b.fits
```

Here the result is written in the new file H4998b.fits. Having many files to reduce, it is more convenient to use the task ccdproc, with a list of all the files to be reduced as the input:

```
ecl> ccdproc @list_of_files
```

One should pay attention to which reduction steps are set to 'yes', and to put correct names of calibration files.

6.2 Corrections for dark current

Even when no light falls on the detector, some counts can be recorded. This is called the *dark current*, and is a result of the thermal movement of the electrons in the chip and electronics. The dark current is estimated by the 'exposure' without opening the shutter. It's level is time-dependent, so in principle it is better to use 'dark' frames of the same or similar exposure time as the science frames. The 'dark' exposures should be trimmed first, then reduced for over-scan and bias. Preparation of the dark calibration image can be done with the task darkcombine, which again uses ccdproc. See help for more details.

The dark current of the HDS detectors is negligible and this step can be omitted. However, one can find dark frames under the following url: http://www.naoj.org/Observing/Instruments/HDS/dark.html.

6.3 Masking bad pixels

In the HDS CCD chips (especially in the Red), there are some strong bad columns. They could badly affect some reduction processes (order tracing, flat-fiedling etc.). In order to correct them, a *mask* frame should be created at the beginning of your reduction process.

When there are not many bad pixels, or there are bad rows or columns, one can prepare a simple text file. Each line should represent The file should have entries of the form "xbegin xend ybegin yend" for rectangular areas, or "x y" for isolated pixels, for example:

```
130 130 1 1024
873 300
```

Then, a task fixpix can be used:

```
                    Image Reduction and Analysis Facility
PACKAGE = proto
   TASK = fixpix

images  =        @list_of_files List of images to be fixed
masks   =             badpix.pl List of bad pixel masks
(linterp=                INDEF) Mask values for line interpolation
(cinterp=                INDEF) Mask values for column interpolation
(verbose=                   no) Verbose output?
(pixels =                   no) List pixels?
(mode   =                   ql)
```

or

```
ecl> fixpix @list_of_files badpix.pl
```

The 'mask' can also be made by extracting those pixels in BIAS frames, which have abnormal counts. For the HDS data one can use the script mkbadmask.cl, available at:
http://www.naoj.org/Observing/Instruments/HDS/hdsql/mkbadmask.cl.
This script requires another one:
http://www.naoj.org/Observing/Instruments/HDS/hdsql/wacosm11.cl,
and both have to be defined in the login.cl file.

```
PACKAGE = echelle
   TASK = mkbadmask

inimage =                 Bias  Input BIAS image
outimage=                 Mask  Output MASK image

(lower  =                 -100) Lower limit replacement window
(upper  =                  300) Upper limit replacement window

(clean  =                  yes) Clean up by wacosm11
(base   =                    1) Baseline for wacosm11
(mode   =                    q)
```

The correction itself can be done with the task ccdproc, with the parameter `fixpix` set to `yes`, and the name of the resulting file – Mask.fits – given in `fixfile`.

6.4 Corrections for non-linearity

Non-linearity is a defect in the CCD's response to the incoming flux of light. At high electron count rates, increase of the flux by a given factor results in the increase of counts by a different, slightly smaller factor. In other words, at high electron numbers the CCD is slightly less sensitive that at low.

While good linearity is confirmed in the HDS data for electron numbers less than 10,000 e^-(∼6,000 ADU), significant non-linearity, of the order of several per cent at 50,000 e^-, appears for higher electron numbers.

The HDS data, after over-scan, can be corrected for the non-linearity with this script (to be defined in login.cl):
http://www.naoj.org/Observing/Instruments/HDS/hdsql/hdslinear.cl.

6.5 Cosmic rays rejection

High-energy particles constantly hit CCD chips, producing significantly more counts in the pixels they fall on. In cases where images are combined (bias, flat, multiple exposures of the same object), these marks can be easily corrected during the combination with the task imcombine (see: Section 6.1). The parameter `refect` should be set to a different value (minmax, sigclip, crreject...). See help of this task for more details.

For single expositions one needs to use a dedicated procedure, many of which are available in IRAF, for example cosmicrays in the package crutil, lacos_spec/lacos_im in the package stsdas, or the aforementioned wacosm11. One needs to remember that such procedures usually look for a sudden increase in counts between adjacent pixels. They should not be used on high signal-to-noise ratio spectroscopic expositions of bright stars, as they tend to treat the spectrum as a big cosmic ray and affect the edges of the spectrum.

7 First extraction of spectra

For the further calibration of the spectroscopic data (e.g. flat-fielding, subtraction of scattered light) it is useful first to extract a spectrum from the data image using the task apall. This will produce reference data for the further calibration procedures. Before starting, make sure along which axis is the dispersion direction (horizontally or vertically). This can be set in the parameters of the echelle package (`epar echelle`) – set the parameter dispaxi to '1' in the horizontal case, and to '2', for the vertical orientation.

The parameters of apall are given as follows:

```
ec> epar apall

                                   I R A F
                    Image Reduction and Analysis Facility
PACKAGE = echelle
   TASK = apall

input   =               H4998b  List of input images
(output =            H4998b_ec) List of output spectra
(apertur=                     ) Apertures
(format =              echelle) Extracted spectra format
(referen=                     ) List of aperture reference images
(profile=                     ) List of aperture profile images

(interac=                  yes) Run task interactively?
(find   =                  yes) Find apertures?
(recente=                  yes) Recenter apertures?
(resize =                  yes) Resize apertures?
(edit   =                  yes) Edit apertures?
(trace  =                  yes) Trace apertures?
(fittrac=                  yes) Fit the traced points interactively?
(extract=                  yes) Extract spectra?
(extras =                   no) Extract sky, sigma, etc.?
(review =                  yes) Review extractions?

(line   =                INDEF) Dispersion line
(nsum   =                  100) Number of dispersion lines to sum or median

                                # DEFAULT APERTURE PARAMETERS

(lower  =                  -5.) Lower aperture limit relative to center
(upper  =                   5.) Upper aperture limit relative to center
(apidtab=                     ) Aperture ID table (optional)

                                # DEFAULT BACKGROUND PARAMETERS

(b_funct=            chebyshev) Background function
(b_order=                    1) Background function order
(b_sampl=          -10:-6,6:10) Background sample regions
(b_naver=                   -3) Background average or median
(b_niter=                    0) Background rejection iterations
(b_low_r=                   3.) Background lower rejection sigma
(b_high_=                   3.) Background upper rejection sigma
(b_grow =                   0.) Background rejection growing radius
```

```
                                # APERTURE CENTERING PARAMETERS

(width  =                   5.) Profile centering width
(radius =                  10.) Profile centering radius
(thresho=                   0.) Detection threshold for profile centering

                                # AUTOMATIC FINDING AND ORDERING PARAMETERS

nfind   =                   22  Number of apertures to be found automatically
(minsep =                   5.) Minimum separation between spectra
(maxsep =                1000.) Maximum separation between spectra
(order  =           increasing) Order of apertures

                                # RECENTERING PARAMETERS

(aprecen=                     ) Apertures for recentering calculation
(npeaks =                INDEF) Select brightest peaks
(shift  =                  yes) Use average shift instead of recentering?

                                # RESIZING PARAMETERS

(llimit =                 -10.) Lower aperture limit relative to center
(ulimit =                  10.) Upper aperture limit relative to center
(ylevel =                  0.1) Fraction of peak or intensity for automatic wid
(peak   =                  yes) Is ylevel a fraction of the peak?
(bkg    =                   no) Subtract background in automatic width?
(r_grow =                   0.) Grow limits by this factor
(avglimi=                  yes) Average limits over all apertures?

                                # TRACING PARAMETERS

(t_nsum =                   10) Number of dispersion lines to sum
(t_step =                   10) Tracing step
(t_nlost=                    3) Number of consecutive times profile is lost bef
(t_funct=             legendre) Trace fitting function
(t_order=                    3) Trace fitting function order
(t_sampl=                    *) Trace sample regions
(t_naver=                    1) Trace average or median
(t_niter=                    0) Trace rejection iterations
(t_low_r=                   3.) Trace lower rejection sigma
(t_high_=                   3.) Trace upper rejection sigma
(t_grow =                   0.) Trace rejection growing radius

                                # EXTRACTION PARAMETERS

(backgro=                 none) Background to subtract
(skybox =                    1) Box car smoothing length for sky
(weights=                 none) Extraction weights (none|variance)
(pfit   =                fit1d) Profile fitting type (fit1d|fit2d)
(clean  =                   no) Detect and replace bad pixels?
(saturat=                INDEF) Saturation level
(readnoi=                    8) Read out noise sigma (photons)
(gain   =                   1.) Photon gain (photons/data number)
(lsigma =                   4.) Lower rejection threshold
(usigma =                   4.) Upper rejection threshold
(nsubaps=                    1) Number of subapertures per aperture
(mode   =                   ql)
```

The input file name and an arbitrary name of the output file are given in the first two lines. This task first searches for spectra of individual echelle orders in the cross cut image of the data. Then the aperture position and size are determined and the result is displayed to provide an opportunity for manual corrections. In order to carry out these tasks, one specify the parameters find, recente, resize and edit to 'yes'.

For the determined aperture size and position, individual spectra are traced and the counts are summed along the direction of the slit. The parameters trace, fittrac and extract are set to 'yes'.

Other important parameters are as follows:

- # AUTOMATIC FINDING ... (nfind): this gives the number of spectra extracted
- # RESIZING PARAMETERS (peak and ylevel): If the parameter peak is yes, the aperture size is determined as the width of the cross cut image of each spectrum at the ylevel height of the peak. For example, if the ylevel is 0.2, the width at the 20% of the peak of the cross cut image is adopted to be the aperture size.

At the first step, the parameters extras and bkg should be no while the avglimi should be yes.

Execute the task and answer the questions, then a diagram like Figure 5 appears. This is a result of the automatic searches for the aperture position and size. The position and size of apertures are confirmed or corrected if necessary. Point the cursor close to an aperture and press 'd' to remove it, press 'm' to manually mark the aperture you want to trace and extract. See help of the task for more commands (type '?'). If the aperture searches are not satisfactory, one may stop the task by typing 'q' and answering no for subsequent questions, and execute the task again with different parameters.

A special care should be paid for the order numbers. The number should be from left to right without any gap or duplication. To sort the apertures, move cursor close to one, press o, and put the number you want this aperture to have in the sequence.

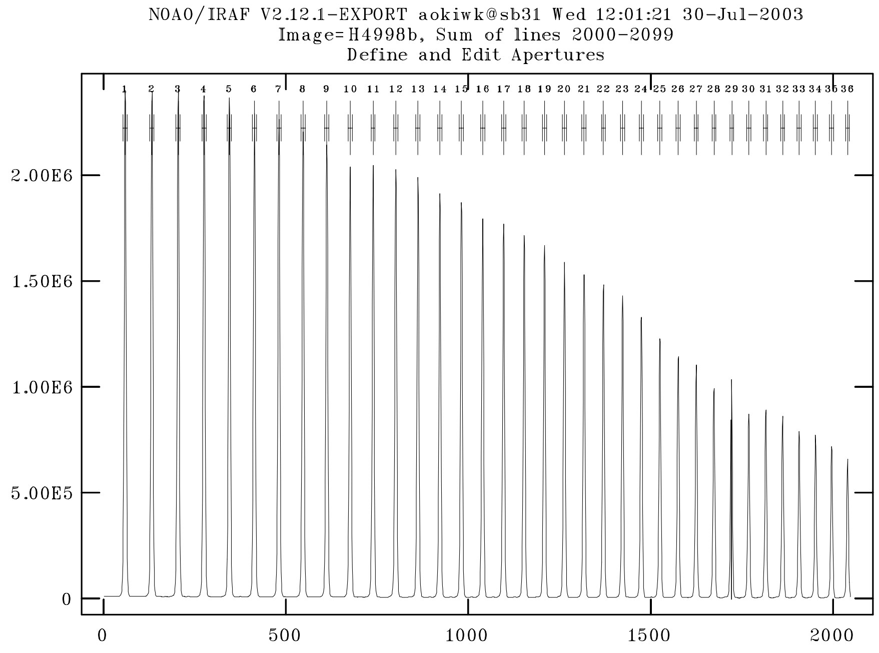


Figure 5: A cross cut image of the data along the slit direction. A result of the first step of the task apall that determines the aperture positions with the order numbers.

If the position, the size of the apertures, and the order numbering are satisfactory, type 'q' at the window of the diagram, then tracing process for each spectrum starts. A diagram like Figure 6 appears. This diagram shows a tentative result of the order trace where the horizontal and vertical axes indicate pixel numbers of the dispersion and slit directions, respectively. The detected peak of the spectrum is shown by the '+' symbol, while the fitting to these detected peaks are shown by a dashed line. Points automatically rejected from the fit are marked with diamonds.

If the fitting is insufficient, one may increase the order of the fitting function, and/or change the fitting range for the horizontal axis. The order of the fitting function is changed, for example, to 3 by typing ':order 3' (or ':o 3') on the display. The fitting range is changed by the key 's' at the two positions on the display (the fitting

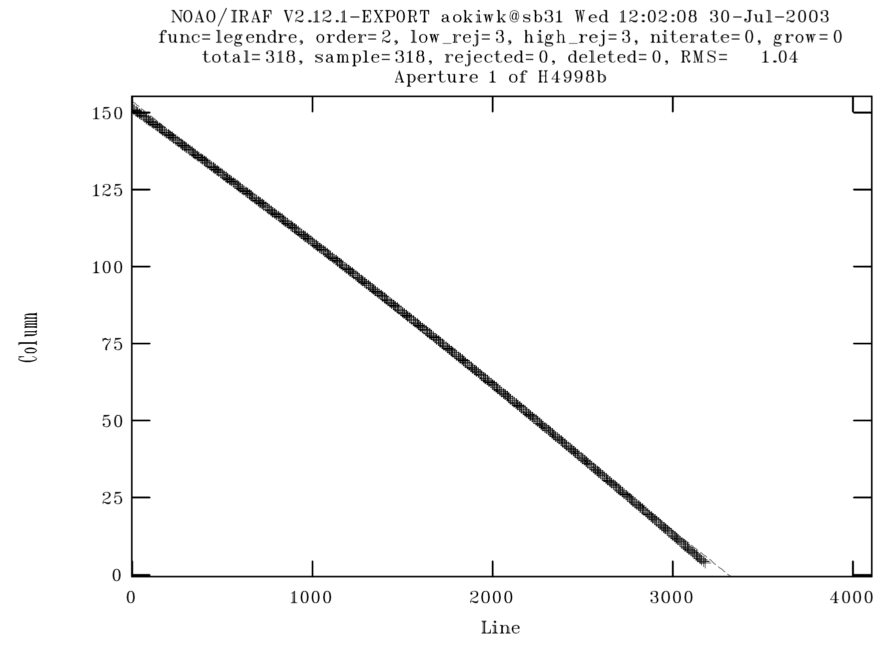


Figure 6: Order tracing for a spectrum. The horizontal axis means the dispersion direction and the vertical axis means the slit direction of the data (unit is the pixel number of the CCD). Note that this order does not stretch across the whole chip, but ends around pixel 3200.

range is initialized by pressing 't'). The fitting using new parameters is made by the key 'f'. Single points can be removed from the fit by pressing 'd' (while the cursor is close to one). They are marked with the '×' symbol. Press 'u' to include them back in the fit.

To evaluate the fit one can check the residuals, by pressing 'j', or the ratio of the location of the detected peaks to the fit with the key 'k'. Typing 'h' will show the points and the fit again. The commands described above work for almost every interactive fitting procedure in IRAF (in the task continuum for example).

If the fitting is satisfactory, type 'q' at the display, and apply a similar procedure to the next spectrum. After fitting for all spectra contained in the image, the extracted spectrum of the first echelle order appears as shown in Figure 7.

The task splot is useful to display spectra:

```
ec> splot H4998b_ec
```

See the last section for the details of splot.

8 Flat fielding

In order to correct the pixel-to-pixel inhomogeneity of the sensitivity of the detector, images of white light (a halogen lamp in case of the HDS) are obtained with the same setup of the spectrograph as the science spectra (Figure 4). These data provide an estimate of the sensitivity of the detector including their wavelength dependence.[3]

One usually makes a 'flat frame' from the median of several flat images, as in the case of bias frames, with the task imcombine. Here the name of the flat frame is given as 'flat.fits'.

Flat fielding of the object data are made by dividing the object frame by the flat frame. Before that, however, the flat frame is usually 'normalized' for the count, in order to keep the count of the object data. The normalization of the flat frame is made with the task apnormalize (or apflatten). The parameters for apnormalize are as follows;

[3]When the wavelength coverage of the spectrograph is so wide that the intensity of the flat lamp may not be sufficiently high in some wavelength ranges, flat data are obtained by changing the exposure time or filter setting. In the case of HDS, flat data are usually obtained for each CCD.

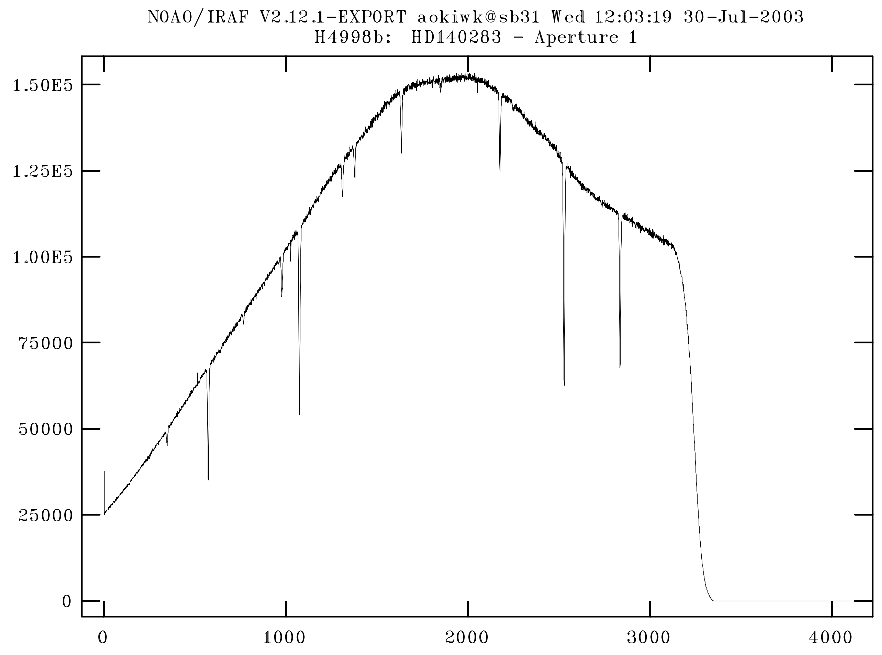


Figure 7: A spectrum obtained by the task apall. The unit of the horizontal axis is the pixel number of the CCD at this stage. The sudden drop after the pixel 3200 comes is because this order of the spectrum does not reach the end of the CCD.

```
                                 I R A F
                  Image Reduction and Analysis Facility
PACKAGE = echelle
   TASK = apnormalize

input   =                 flat  List of images to normalize
output  =                flatn  List of output normalized images
(apertur=                     ) Apertures
(referen=               H4998b) List of reference images

(interac=                  yes) Run task interactively?
(find   =                   no) Find apertures?
(recente=                  yes) Recenter apertures?
(resize =                  yes) Resize apertures?
(edit   =                  yes) Edit apertures?
(trace  =                   no) Trace apertures?
(fittrac=                   no) Fit traced points interactively?
(normali=                  yes) Normalize spectra?
(fitspec=                  yes) Fit normalization spectra interactively?

(line   =                INDEF) Dispersion line
(nsum   =                   10) Number of dispersion lines to sum or median
(cennorm=                   no) Normalize to the aperture center?
(thresho=                  10.) Threshold for normalization spectra

(backgro=                 none) Background to subtract
(weights=                 none) Extraction weights (none|variance)
(pfit   =                fit1d) Profile fitting type (fit1d|fit2d)
(clean  =                   no) Detect and replace bad pixels?
(skybox =                    1) Box car smoothing length for sky
(saturat=                INDEF) Saturation level
```

```
(readnoi=                     0.) Read out noise sigma (photons)
(gain   =                     1.) Photon gain (photons/data number)
(lsigma =                     4.) Lower rejection threshold
(usigma =                     4.) Upper rejection threshold

(functio=                spline3) Fitting function for normalization spectra
(order  =                      3) Fitting function order
(sample =                      *) Sample regions
(naverag=                      1) Average or median
(niterat=                      5) Number of rejection iterations
(low_rej=                     3.) Lower rejection sigma
(high_re=                     3.) High upper rejection sigma
(grow   =                     0.) Rejection growing radius
(mode   =                     ql)
```

These are similar to the parameters of the task apall, because the extraction of the flat spectra is once made for the normalization. This time, however, the object data (e.g. H4998), to which extraction was already applied, can be used as a reference image (parameter referen). Since the object and flat images are obtained with the same setup of the spectrograph, the positions of the recorded spectra on the CCD are the same in principle[4].

In order to refer to the object file, the name of the object file (e.g., H4998b) is given as the parameter referen. *The reference image is the two dimensional data before extraction, rather than the extracted one dimensional spectrum data (e.g., H4998b,ec).* The reference image is used to search for the apertures and to trace the spectra. In that case the parameters find, trace and fittrac are set to no. The detailed position and the size of apertures are determined for the flat frame, because they are different in general between the object and flat images. Therefore, the parameters recente and resize are set to yes. Unfortunately, the parameters peak and ylevel do not exist in the parameter list of apnormalize. They are given separately in the parameter list of apresize, which is called from apnormalize, as follows:

```
                                   I R A F
                    Image Reduction and Analysis Facility
PACKAGE = echelle
   TASK = apresize

input   =                         List of input images
(apertur=                       ) Apertures
(referen=                       ) Reference images

(interac=                     no) Run task interactively?
(find   =                    yes) Find apertures?
(recente=                     no) Recenter apertures?
(resize =                    yes) Resize apertures?
(edit   =                    yes) Edit apertures?

(line   =                  INDEF) Dispersion line
(nsum   =                      1) Number of dispersion lines to sum or median
(llimit =                   -20.) Lower aperture limit relative to center
(ulimit =                    20.) Upper aperture limit relative to center
(ylevel =                    0.4) Fraction of peak or intensity for automatic wi
(peak   =                    yes) Is ylevel a fraction of the peak?
(bkg    =                     no) Subtract background in automatic width?
(r_grow =                     0.) Grow limits by this factor
(avglimi=                    yes) Average limits over all apertures?
(mode   =                     ql)
```

[4] The positions of the spectra on the CCD image are recorded in the file like apH4998 below the directory database. So one should not delete or move this directory, and be careful for the correspondence between the data file itself, and the file in the database directory.

Extraction of flat spectra is made by apnormalize as that by apall for object spectra. However, since the reference data are used for tracing the spectra, there is no manual work in this case. Then the first extracted spectrum appears as Figure 8.

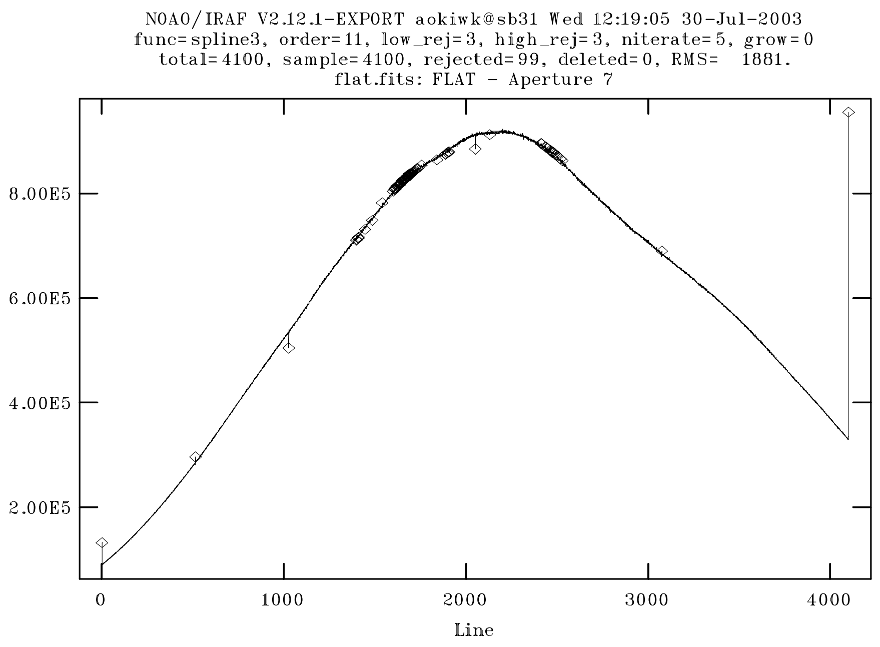


Figure 8: Fitting of a curve to an extracted flat spectrum

A profile fit to the data points is also shown in the diagram. In order to improve the fitting, one may chance the function and the order of the function by, for example, ':function spline3' and ':order 9', respectively. One may also change the fitting range by double 's' keys to avoid the inappropriate parts of the spectra for the fitting, e.g. pixels affected by bad columns. Then the key 'f' makes a re-fitting for the new parameter setup.

Pressing 'q' finishes the procedure for this spectrum, then the next one appears. This procedure is applied to all orders of the spectra.

The result is seen by implot. The normalized flat frame looks like Figure 9. The part where the count is exactly unity is the outside of the aperture, i.e. the region between the two adjacent spectra where (essentially) no light is detected. The scatter found within the apertures indicates the inhomogeneity of the detector sensitivity.

Flat fielding of an object frame is made by dividing the object frame by the normalized flat frame with the task imarith:

```
imarith H4998b / flatn H4998bf
```

where the normalized flat frame is flatn.fits and the flat-fielded object frame is H4998bf.fits.

9 Background subtraction

As can be seen in the cross cut image of the object frame (Figure 10), the counts of the background region between the two adjacent orders are not zero, due scattered light inside the spectrograph. Estimates of background is made by masking the apertures of the spectra and applying surface fitting to the other regions. The background subtraction is made only for object frames here[5].

The task apscatter is used for this purpose. An example of parameter setting for this task is given below. The parameters referen, find, recente, resize, edit, trace and fittrac are set as in the case of apnormalize. The reference file is again the object frame for which the order trace and extraction is once made. The aperture size is given in the parameter list of apresize.

[5]The background subtraction should be also made for the flat data before flat fielding. This process is skipped here because the effect is minor.

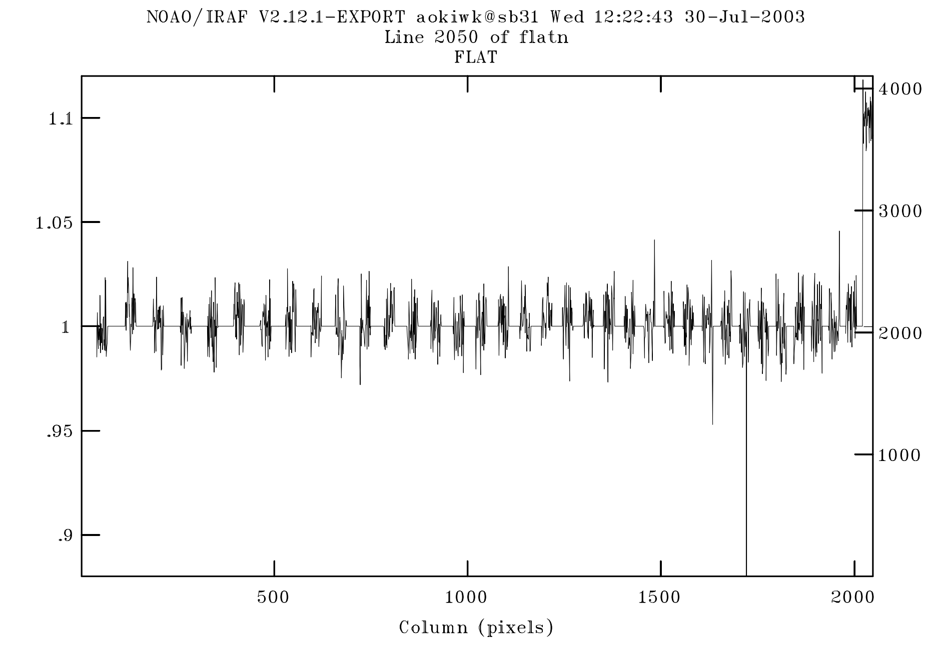


Figure 9: A cross cut image of the flat frame along the slit direction

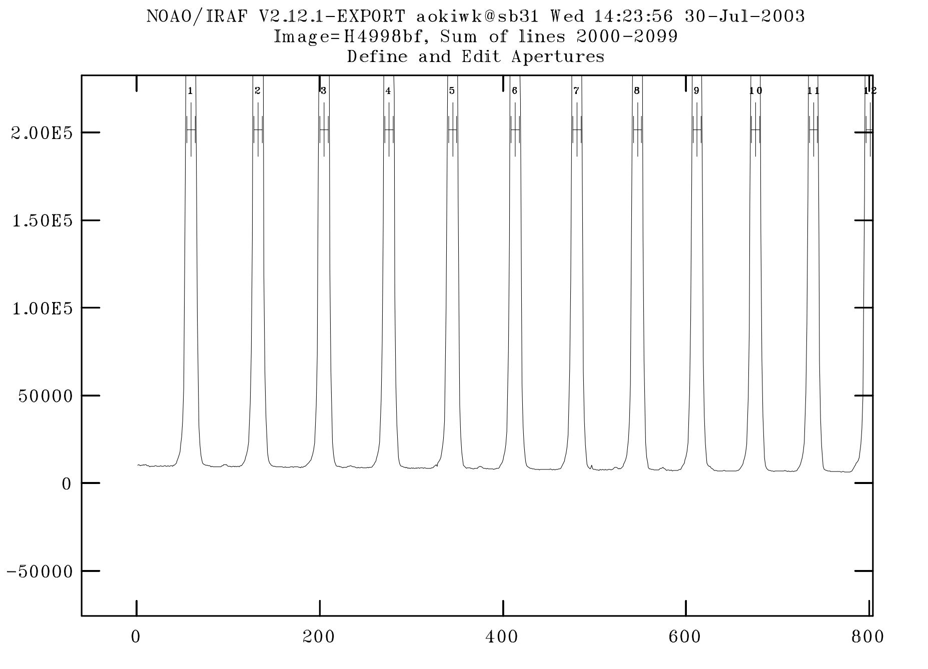


Figure 10: A cross cut image of the object frame obtained during the procedure of apscatter.

The first part of the task is carried out like that of apnormalize. After the (automatic) order tracing, a result of the surface fitting for the slit direction is displayed like in Figure 11. One may correct the fitting function and its order. The key 'q' finishes the fitting process for this cross cut image. Then one may see the fitting result for other lines by inputting, for example, 'line 200'. If the fitting for the slit direction looks satisfactory, type 'q' or 'quit', then the fitting for the dispersion direction like Figure 12 is shown (this may take a while). One may see the cross cut images of other orders by typing 'column 2500' etc. If the fitting is satisfactory, type 'q' or 'quit', then a background-subtracted image is obtained.

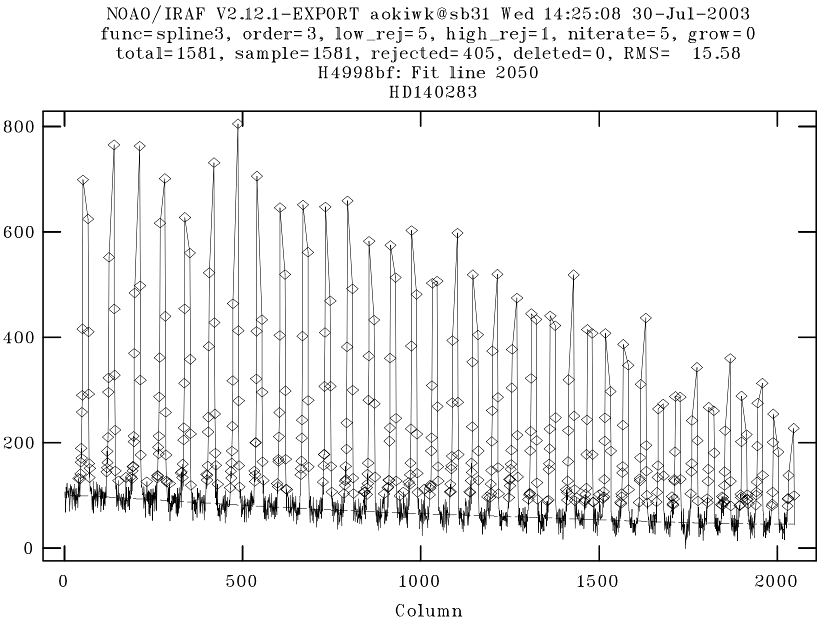


Figure 11: A cross cut image of the aperture-masked data for the slit direction. The result of surface fitting to the estimated background image is shown by the dashed line.

```
                              I R A F
                  Image Reduction and Analysis Facility
PACKAGE = echelle
   TASK = apscatter

input   =              H4998bf  List of input images to subtract scattered lig
output  =             H4998bfs  List of output corrected images
(apertur=                     ) Apertures
(scatter=                     ) List of scattered light images (optional)
(referen=              H4998b) List of aperture reference images

(interac=                 yes) Run task interactively?
(find   =                  no) Find apertures?
(recente=                 yes) Recenter apertures?
(resize =                 yes) Resize apertures?
(edit   =                 yes) Edit apertures?
(trace  =                  no) Trace apertures?
(fittrac=                  no) Fit the traced points interactively?
(subtrac=                 yes) Subtract scattered light?
(smooth =                 yes) Smooth scattered light along the dispersion?
(fitscat=                 yes) Fit scattered light interactively?
(fitsmoo=                 yes) Smooth the scattered light interactively?

(line   =               INDEF) Dispersion line
(nsum   =                 100) Number of dispersion lines to sum or median
(buffer =                  1.) Buffer distance from apertures
```

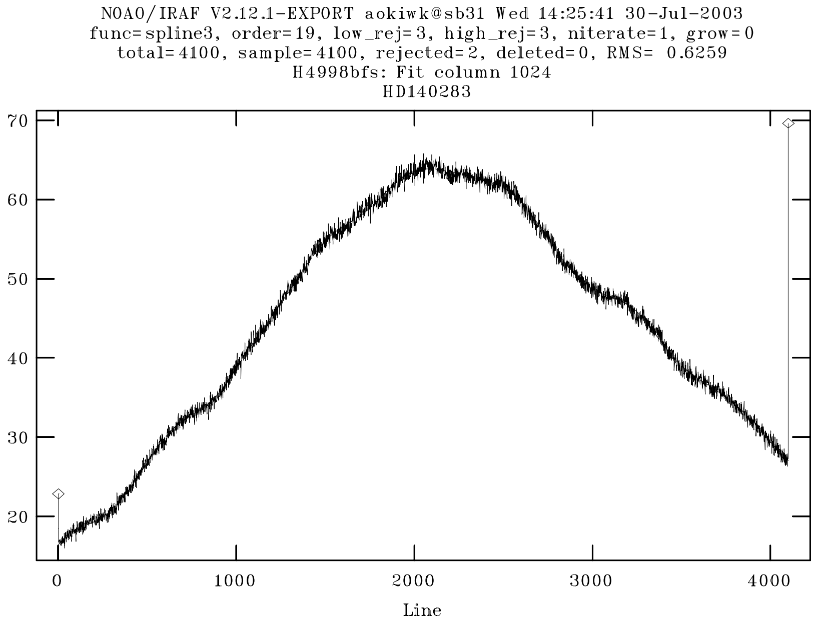


Figure 12: The same as Fig.11, but for the dispersion direction.

```
(apscat1=                    ) Fitting parameters across the dispersion
(apscat2=                    ) Fitting parameters along the dispersion
(mode   =                  ql)
```

10 Extraction of one dimensional spectra

The extraction of spectra from the flat-fielded and background-subtracted object frame is made again using the task apall. In this case, however, one can use the first object frame to which extraction was applied as a reference data. Followings are an example of the parameter setting for the first 16 lines:

```
                              I R A F
                    Image Reduction and Analysis Facility
PACKAGE = echelle
   TASK = apall

input   =            H4998bfs  List of input images
(output =         H4998bfs_ec) List of output spectra
(apertur=                    ) Apertures
(format =             echelle) Extracted spectra format
(referen=              H4998b) List of aperture reference images
(profile=                    ) List of aperture profile images

(interac=                 yes) Run task interactively?
(find   =                  no) Find apertures?
(recente=                 yes) Recenter apertures?
(resize =                 yes) Resize apertures?
(edit   =                 yes) Edit apertures?
(trace  =                  no) Trace apertures?
(fittrac=                  no) Fit the traced points interactively?
(extract=                 yes) Extract spectra?
(extras =                  no) Extract sky, sigma, etc.?
(review =                 yes) Review extractions?
```

This part does not have to be run interactively (set the parameter interac to 'no'), however, we suggest to keep the control on this step in case of low signal-to-noise data.

11 Wavelength calibration

The above procedure provides the one dimensional spectra for individual echelle orders as a function of the pixel number. The wavelength calibration is made using comparison spectra (spectra of Th-Ar arc lamp) obtained by the same setup of the spectrograph as applied to the object. The laboratory wavelengths of individual Th spectral lines are known. Here, the relation between the wavelength and the CCD pixel number is made using the Th-Ar spectra.

11.1 Identification of Th spectral lines

First, the Th-Ar spectra are extracted using apall as for the object data. In this case (a slit spectrograph), the aperture position and size should be just the same as those for the object data. That is, the parameters find, recente, ..., fittrac are specified to be no, and the object file (before extraction) is given as the reference file (the parameter referen). For fibre-fed spectrographs, the reference file can be the same as for other steps.

Then, a spectrum as shown in Figure 13 is obtained.

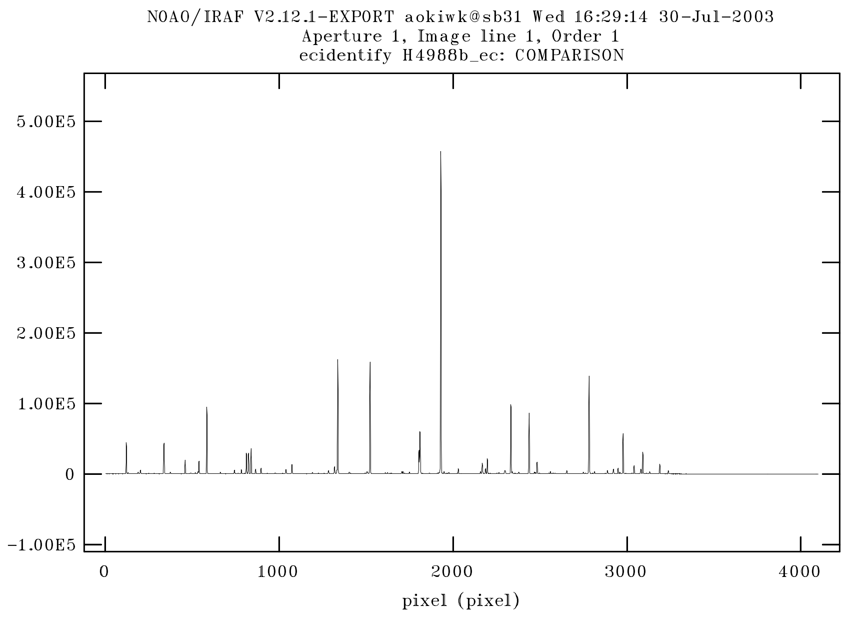


Figure 13: An example of comparison spectrum. This plot appears in the ecidentify procedure.

A number of Th (and Ar) lines appear in the comparison spectra. The next step is to identify the spectral lines and assign the wavelengths. For this purpose, it is useful to know the wavelength coverage of the data. The wavelength coverage calculated for the spectrograph setup is given in the FITS header as follows:

```
WAVELEN =               480.60 / Center wavelength of the center order (nm)
WAV-MAX =               540.17 / Maximum wavelength recorded (nm)
WAV-MIN =               414.20 / Minimum wavelength recorded (nm)
```

In this case, the frame approximately covers 414.2–540.17 nm.

On the other hand, an atlas of wavelengths for individual Th lines is provided for the HDS data reduction[6]. The wavelength of each Th line is identified by comparisons of the above plot and the atlas.

[6]Honda & Aoki 2001, http://www.naoj.org/Observing/Instruments/HDS/wavecal.html

11.2 Wavelength identification for Th lines and wavelength scaling

The wavelength identification for each Th line is made by the task ecidentify.

```
ec> epar ecident
```

The name of the comparison spectrum is given for the parameter images, while the coodli is set to 'linelists$thar.dat', which means the list of Th lines implemented in IRAF.

A spectrum like Figure 13 is displayed in the course of executing this task. One may magnify the plot using the 'window' mode (key 'w') as in the case of splot (Section 15). Point the cursor at one emission line, and press 'm' there. Then a mark appears above the emission line. Give its wavelength identified from the atlas of Th-Ar data. For each echelle order several lines should be identified, spread as evenly across the order as possible. One may delete mis-identified line by pointing the cursor on the line and pressing the key 'd'. (The mark may remain on the window. In that case, re-display the plot by the key 'r'.)

One may move to the next order of the spectra using the key 'k' (and go back to the previous order with 'j'). This procedure is made for every order of the spectra. In practice, one may skip some orders to save the manual job.

If the identification of lines for the whole data is completed, press 'f' to fit a function to the identified lines, providing relation between the line position on the detector and the wavelength. A plot of the fitting residual as a function of the pixel number of the dispersion direction is displayed as Fig. 14. The data point that significantly deviates from others may be deleted by the key 'd' (pointing it by the cursor). The function for the order of the fitting function are changed by inputting, for example, ':xorder 4' ':yorder 3', which are corresponding to the fitting functions for dispersion and slit directions. Figure 14 shows a result of the appropriate fitting (the number of data points are very large because this is a result of the automatic searching for the lines. See below for details). A plot for the slit direction is shown by the keys 'x' and 'o' (the keys 'x' and 'p' shows again the fitting for the dispersion direction).

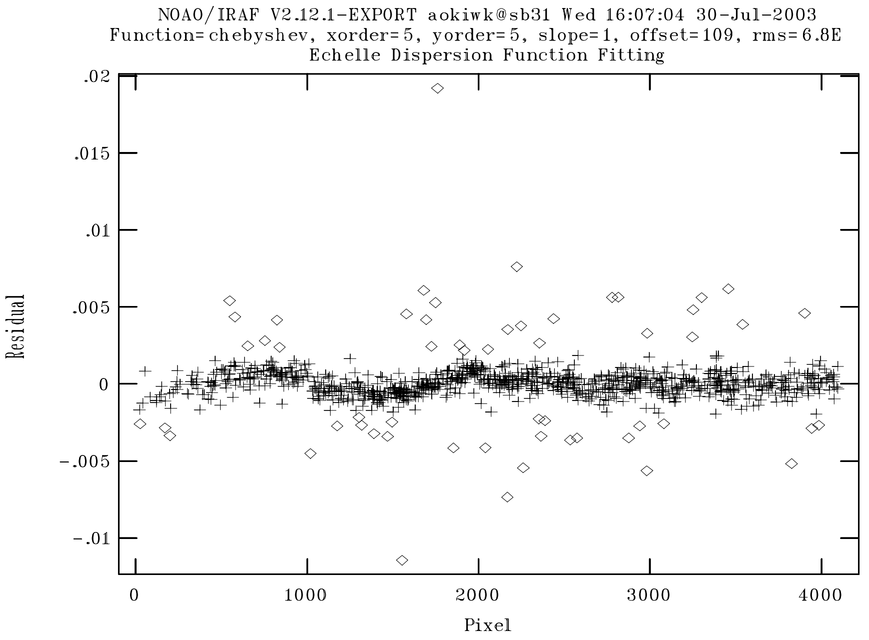


Figure 14: A result of the identification of Th lines and fitting. The residual of the fitting (the unit is Å) is shown as a function of the pixel number of the dispersion direction

If the fitting result is satisfactory (the rms of residual is usually smaller than 0.01 for the xorder=yorder=3), exit this plot with the key 'q' (the plot like Figure 13 appears again). The key 'l', executes automatic searches for Th lines using the Th line list included in the IRAF. The maximum number of automatic identification is given in the parameter file of ecidentify as maxfeat. The fitting result is shown again by the key 'f': this time, many lines are identified as found in Fig. 14. Change the parameters xorder and yorder to obtain satisfactory fit. For typical HDS data xorder=yorder=4 provides good results. The task is finished by pressing the key 'q'.

If there are problems in the first manual identifications for Th lines, the automatic fit possibly produces incorrect results. The resulting plots of the line identifications and fitting should be carefully examined.

The wavelength solution is in general different for each spectrograph setting, but within one setting should be repeatable. To apply it to many Th-Ar spectra, one should use the task ecreidentify.

```
ec> ecreid H5008b_ec refe=H4988b_ec
```

Here, the H5008b_ec.fits is the new Th-Ar exposure we want to find the solution for, and H4988b_ec.fits is the one we found the solution previously. An output like this should appear:

```
ECREIDENTIFY: NOAO/IRAF V2.12.1 aokiwk@sb31 Wed 16:15:48 30-Jul-2003
  Reference image = H4988b_ec, Refit = yes
               Image    Found     Fit Pix Shift  User Shift  Z Shift       RMS
           H5008b_ec  784/784 784/784    0.0381      0.0634  1.06E-7   6.3E-4
```

This means that the procedure has found and fit all the lines that were used for the reference Th-Ar spectrum. One can re-examine and correct the solution for the new spectrum with ecidentify.

11.3 Wavelength calibration of the science spectra

Now the wavelength scale produced by the ecidentify or ecreidentify is applied to the stellar spectra. First, the stellar spectra are linked to the above comparison data by the task refspectra as follows:

```
on> refspec H4998bfs_ec refe=H5008b_ec
```

Here, the H4998bfs_ec.fits is the stellar spectrum data, while H5008b_ec.fits is the comparison data. The default setting for the parameters sort and group of this task are jd. However, jd does not appear in the HDS fits header, while MJD is given. Delete jd from these two lines, or give MJD (or UT) for sort.

In many scientific cases, the spectrum of the target is adjacent to two Th-Ar exposition, one taken just before and the other just after the science spectrum. In refspectra it is possible to use more than one reference:

```
ec> refspec H4998bfs_ec refe=H5008b_ec,H4988b_ec
```

In such case, parameters sort and select determine how the final calibration is done, i.e. which reference spectrum is taken with which weight. See the help of refspectra for more details.

Finally, the wavelength calibration for the spectra is made by the task dispcor (short of 'dispersion correction'):

```
ec> dispcor H4998bfs_ec H4998bfs_ecw
```

Then, the wavelength-calibrated spectrum H4998bfs_ecw.fits is obtained (Figure 15).

12 Continuum normalization

The next step is to make normalized spectra by fitting curves. Echelle spectra usually have the efficiency peak near the center of the CCD, and the count level is sometimes very low at the edges. An example is shown in Figure 15, where the count of the shorter wavelength is quite low. In such cases, some portion that has very low count may be trimmed from the spectrum. The trimming, the first 600 pixels for example, can be done with the task imcopy as follows:

```
imcopy H4998bfs_ecw[601:4100,*] H4998bfs_ecwt
```

The normalization is carried out by the task continuum.

```
ec> continuum H4998bfs_ecw H4998bfs_ecwc.fits
```

Figure 16 shows an example of the fitting of a function to the spectrum, where the absorption lines are excluded from the fitting. The fitting function and its order are changed by inputting, for example, 'func spline3' and ':order 5', respectively. Fitting ranges are also given by the key 's' as in the case for apnorm. The fitting is updated by the key 'f'. Keys 'h', 'j', and 'k' show the spectrum with the fit, the residuals, and the ratio (a continuum-normalized spectrum!), respectively.

If the fitting is satisfactory, type 'q', then the spectrum of the next order appears. The procedure is repeated for every order. Figure 17 shows an example of the normalized spectrum.

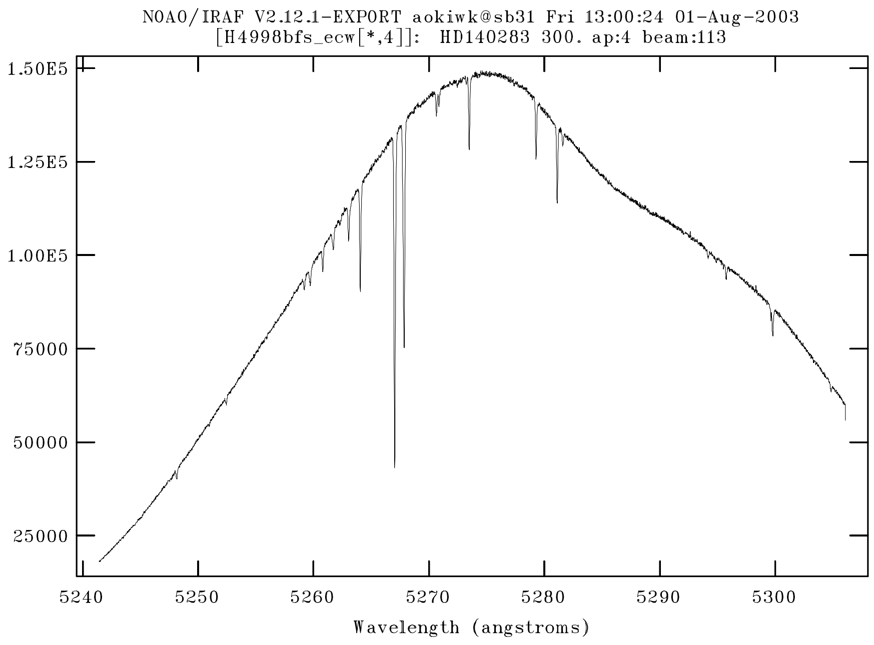


Figure 15: Wavelength-calibrated spectrum obtained by dispcor.

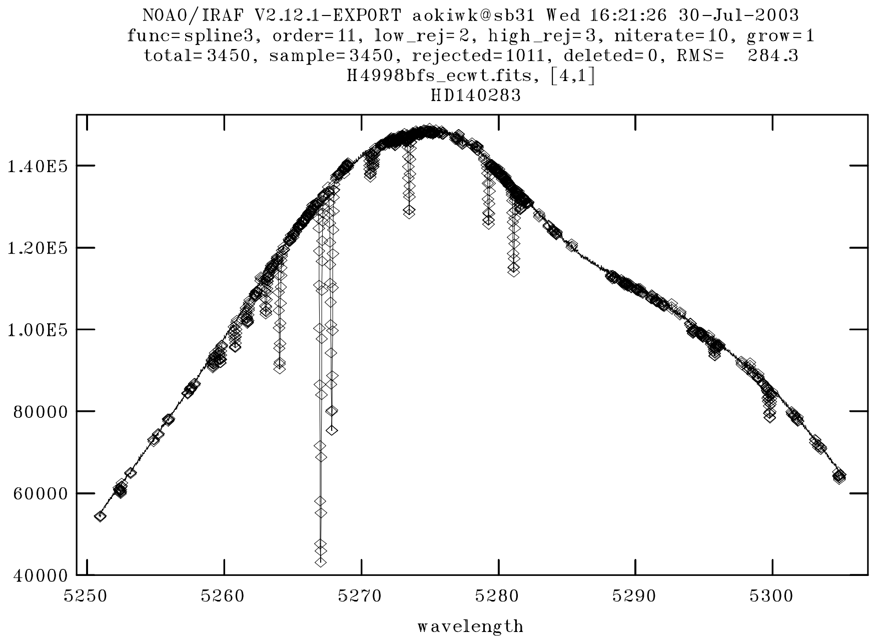


Figure 16: An example of continuum fitting

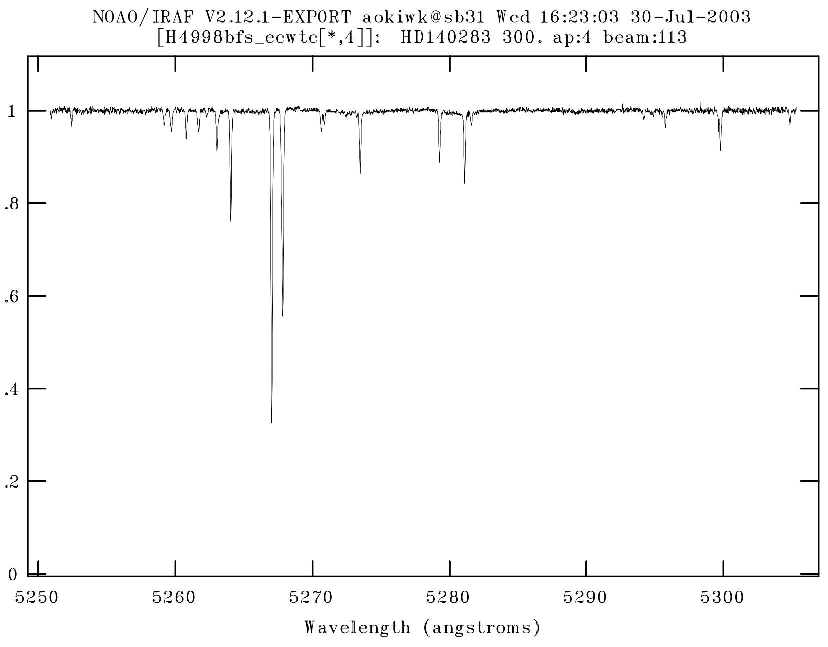


Figure 17: An example of the normalized spectrum

13 Making a combined spectrum

The multi-order spectra are merged by the task scombine[7]. A simple method is to average the normalized spectra for the overlapped range (the portion which is covered by adjacent orders). In this case, the parameters of this task group and combine are set to images and average, respectively.

However, this simple method results in a significantly degraded spectrum in the merged region, where the photon counts at the edge of one spectrum are quite low compared with the other, as seen in Fig. 18. This is because the noise of such portion is magnified by the normalization process. Figure 19 shows an example of the result by the above simple method. The spectrum includes ranges where the quality is significantly low (i.e. around 3468 Å and 3488 Å).

This problem is avoided by the following procedure. First, normalized spectra are made by the task continuum, as in the previous section. Then, the spectra are divided by the normalized spectra using the task smarith[8]. In such way the spectra of the continuum are obtained, as shown in Figure 20. They are just the functions that were fit to the spectra in the normalization process.

Next, the spectra are merged by the task scombine with the parameter combine of sum as follows;

```
(group  =                images) Grouping option
(combine=                   sum) Type of combine operation
```

Then, the summed spectrum is obtained, as shown in Figure 21.

The same procedure is applied to the continuum spectra. The result is shown in Figure 22. Finally, the summed object spectrum is divided by the summed continuum spectrum using sarith, and the normalized and merged spectrum is obtained (Figure 23).

[7]The scombine is used for adding several spectra obtained with the same setup. In this case, the input files are given like "input=A.ec,B.ec,C.ec", while the parameters group and combine are set to aperture and sum, respectively.

[8]The task sarith is preferred over the imarith, as it works in the dispersion coordinate space (typically wavelength) rather than logical pixel space.

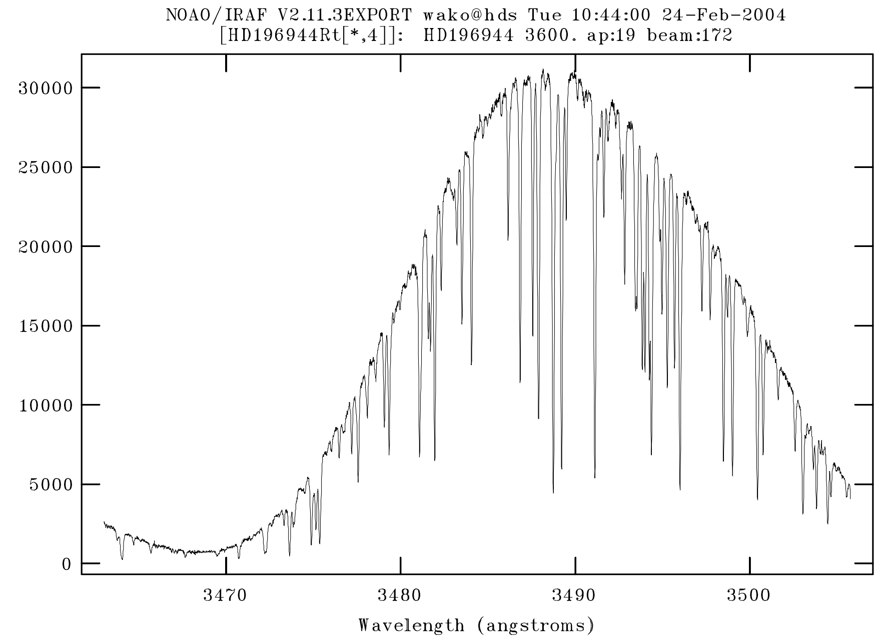


Figure 18: An example of wavelength-calibrated spectra. The count at the edge of the spectrum is quite low.

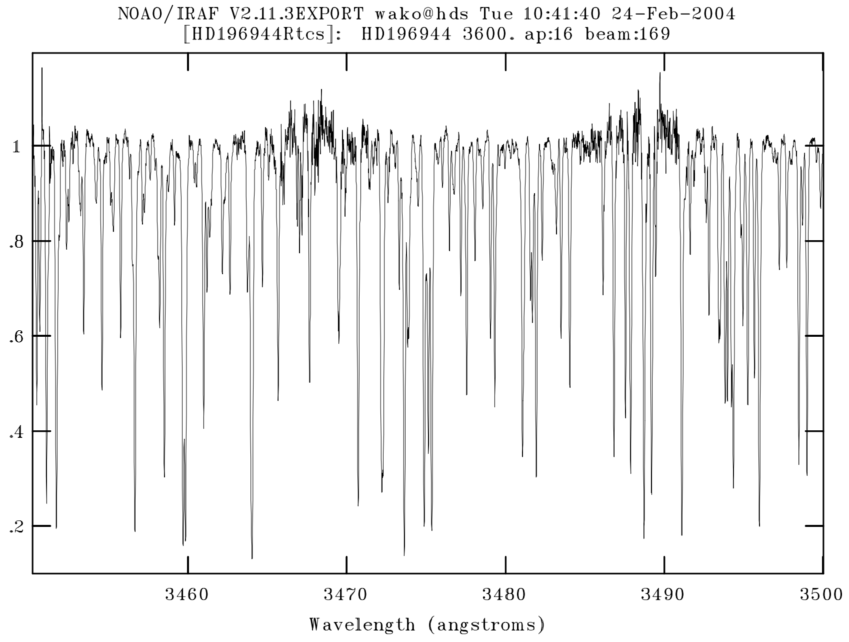


Figure 19: A combined spectra by applying a simple average of normalized spectra. The quality of the data is significantly low at the wavelengths corresponding to the edges of each spectra.

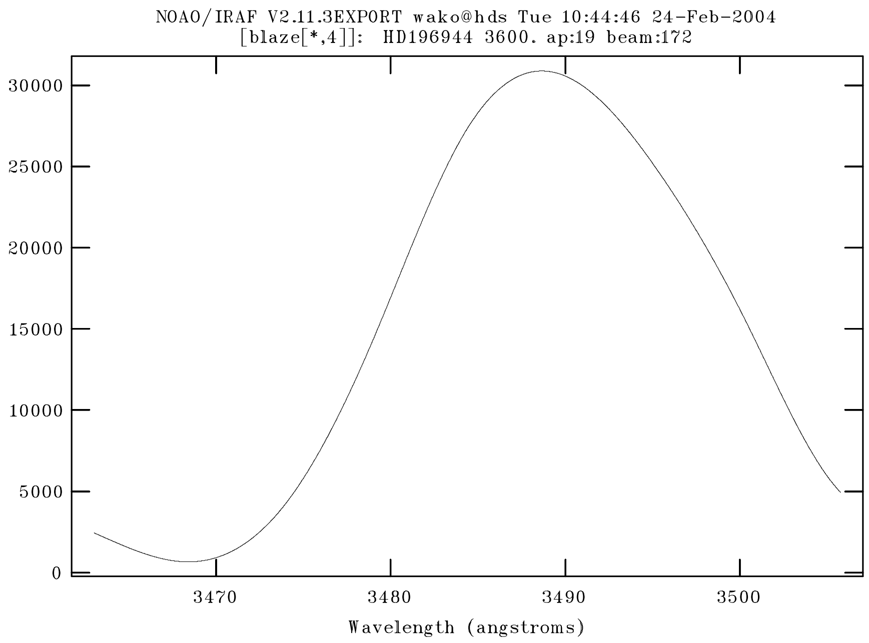


Figure 20: Spectra of continuum.

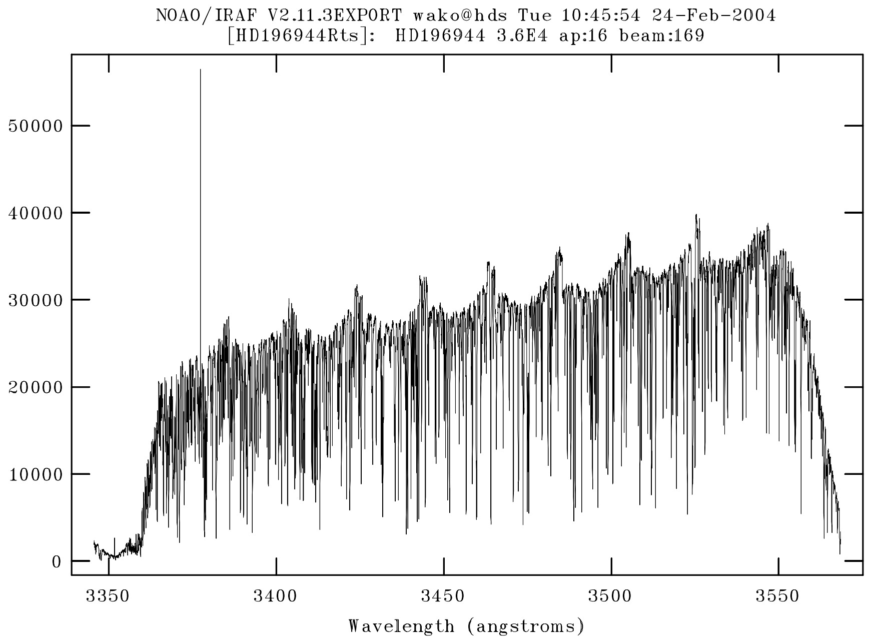


Figure 21: A spectrum combined by simple sum

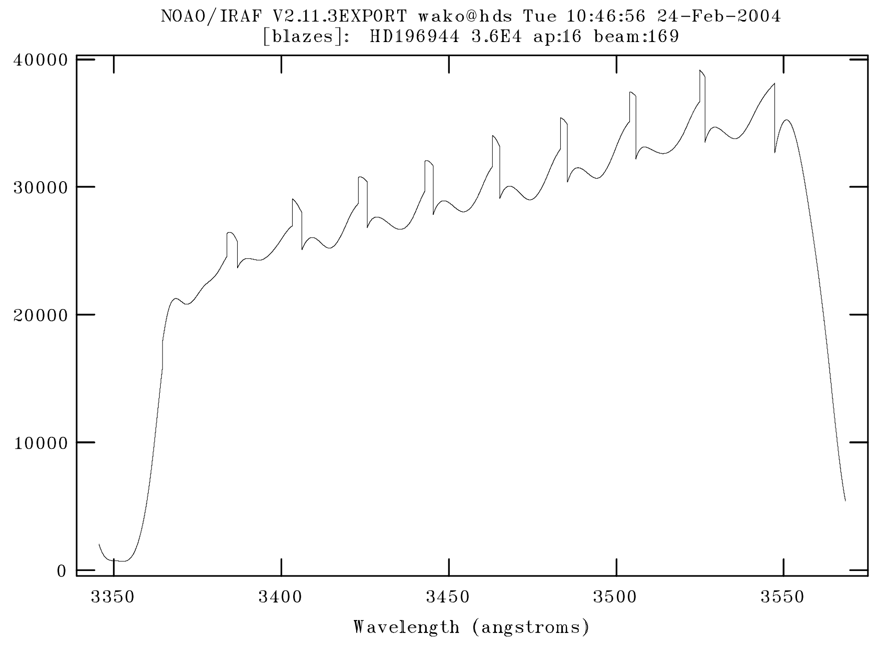


Figure 22: The same as Figure 21, but for the continuum spectrum.

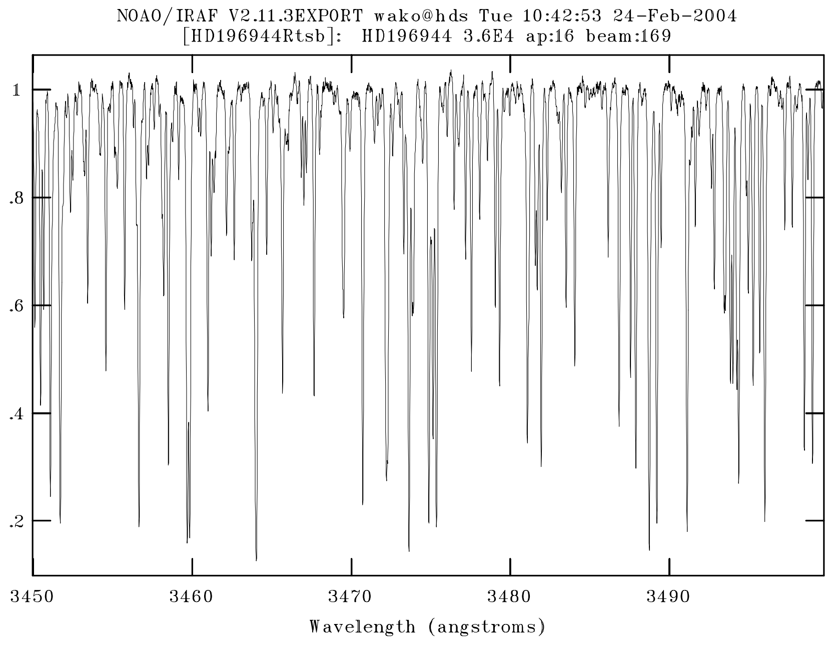


Figure 23: The spectrum obtained by dividing the object spectrum (Figure 21) by the continuum spectrum (Figure 22)

14 Useful tasks for analyses of obtained spectra

Here some useful tasks to analyze the obtained spectra are briefly introduced. Please see help or IRAF manual for details.

14.1 Statistics

Statistics of data (e.g. average, standard deviation) are obtained with the task imstatistics:

```
ecl> imstat object
#              IMAGE      NPIX      MEAN    STDDEV       MIN       MAX
              OBJECT   92822    0.8684     10.98    -2594.     219.1
```

The range to which the task applied is given for the input data name like `object[1000:3000,*]`.

14.2 Measurements of spectral lines

The task splot may be used to measure equivalent widths, FWHM of lines, line positions etc. Gauss, Lorentz and Voigt profile fitting, as well as line deblending, are available for these purposes. Measurements can be done both in wavelength and velocity domain. See the help of this task for more details.

14.3 Doppler correction for observers motion

The correction from the apparent radial velocity to the helio-centric radial velocity (or others) can be calculated using the task rvcorrect in the package noao.astutil. Before that, information of the observatory is given by changing parameters for observatory in the package noao as follows:

```
                               I R A F
                 Image Reduction and Analysis Facility
PACKAGE = noao
   TASK = observatory

command =                  set  Command (set|list|images)
obsid   =              obspars  Observatory to set, list, or image default
images  =                       List of images
(verbose=                   no) Verbose output?

(observa=              subaru) Observatory identification
(name   =              Subaru) Observatory name
(longitu=      155.47611111111) Observatory longitude (degrees)
(latitud=      19.825555555556) Observatory latitude (degrees)
(altitud=                4139.) Observatory altitude (meters)
(timezon=                  10.) Observatory time zone
override=                       Observatory identification
(mode   =                   ql)
```

Below is an example of the parameters for rvcorrect, where the apparent radial velocity is given to vobs. The parameter observa is set to obspars, then the information currently given by the task observatory is applied.

```
                               I R A F
                 Image Reduction and Analysis Facility
PACKAGE = astutil
   TASK = rvcorrect

(files  =                     ) List of files containing observation data
(images =                     ) List of images containing observation data
(header =                  yes) Print header?
(input  =                   no) Print input data?
```

```
(imupdat=                  no) Update image header with corrections?

(epoch  =               2000.) Epoch of observation coordinates (years)
(observa=             obspars) Observatory
(vsun   =                 20.) Solar velocity (km/s)
(ra_vsun=                 18.) Right ascension of solar velocity (hours)
(dec_vsu=                 16.) Declination of solar velocity (degrees)
(epoch_v=               2000.) Epoch of solar coordinates (years)

(year   =                1996) Year of observation
(month  =                   1) Month of observation (1-12)
(day    =                  18) Day of observation
(ut     =              18.969) UT of observation (hours)
(ra     =              8.7278) Right ascension of observation (hours)
(dec    =             -7.2336) Declination of observation (degrees)
(vobs   =               12.34) Observed radial velocity
(hjd    =     2450101.2953037) Helocentric Julian Day (output)
(vhelio =     20.483996801402) Helocentric radial velocity (km/s) (output)
(vlsr   =     5.3777605335745) Local standard or rest radial velocity (km/s) (o
(mode   =                  ql)
```

Followings is an example of results, where the value “VHELIO” and others are listed.

```
# RVCORRECT: Observatory parameters for OAO
#       latitude = 34.5738889
#       longitude = -133.5963889
#       altitude = 372.
##   HJD          VOBS   VHELIO     VLSR   VDIURNAL   VLUNAR  VANNUAL   VSOLAR
2450101.29530    12.34    20.48     5.38     -0.268    0.007    8.404  -15.106
```

14.4 FITS to ASCII, and ASCII to FITS conversion

The task wspectext in the package onedspec is used to write the FITS spectral data to an ASCII file.

```
ec> oned
      aidpars@       dopcor         reidentify     sensfunc       specplot
      autoidentify   fitprofs       rspectext      setairmass     specshift
      bplot          identify       sapertures     setjd          splot
      calibrate      lcalib         sarith         sfit           standard
      continuum      mkspec         sbands         sflip          telluric
      deredden       names          scombine       sinterp        wspectext
      dispcor        ndprep         scoords        skytweak
      disptrans      refspectra     scopy          slist
on> wspectext H4998bfs_ecwtcs HD140283.txt
```

The spectral data (wavelength and count) are listed with the header unit of the FITS data.

```
BITPIX  =                    8  /  8-bit ASCII characters
NAXIS   =                    1  /  Number of Image Dimensions
NAXIS1  =               110420  /  Length of axis
ORIGIN  = 'NOAO-IRAF: WTEXTIMAGE'  /
IRAF-MAX=                   0.  /  Max image pixel (out of date)
IRAF-MIN=                   0.  /  Min image pixel (out of date)
IRAF-B/P=                   32  /  Image bits per pixel
IRAFTYPE= 'REAL FLOATING      '  /  Image datatype
OBJECT  = ' HD140283          '  /
FILENAME= 'H4998BFS_ECWTCS    '  /  IRAF filename
```

```
....

4110.51308358849  0.9839716
4110.52538877657  0.990428
4110.53769396465  0.9826592
4110.54999915273  0.978662
4110.56230434081  0.9812679
4110.57460952889  0.9803735
4110.58691471697  0.974153
4110.59921990505  0.9779771
4110.61152509312  0.9771149
4110.6238302812  0.9719001
4110.63613546928  0.9783365
4110.64844065736  0.9836422

....
```

A 1D spectrum can be produced from a text file that has a similar structure as the example above, using the task rspectext. The wavelength solution can be either stored in the (optional) header part, or can be reproduced – rspectext calls the task dispcor for this purpose.

15 Appendix

15.1 The HDS quick-look reduction

In addition to this data reduction manual, there is an IRAF procedure that allows for almost-on-the-fly HDS data reduction and examination, while doing the observations:

`http://www.naoj.org/Observing/Instruments/HDS/hdsql-e.html`

It is specifically designed for the HDS, but follows the general approach described here. It is also possible to run it outside the observatory. Once the preparations are made, it reduces *only one* spectrum at a time. Although it is prepared to be as good as possible, there is no guarantee that the quality of the output will be optimal. This manual was intended to be more general and explanatory.

15.2 Other useful tasks and functions

- Task history

  When starting IRAF with cl, the tasks executed previously are selected and modified by e. When starting with ecl, up and down arrows can be used.

- `@` Files

  When a large number of files are dealt with by the same task with the same parameter setting, the names of input and output files can be given in text files, in which one file name is given in each line. The files are referred to with `@` at the head. For instance, when the object files are divided by a flat frame (e.g., flat.fits), the names of object files may be given in the file "input_list" like

  ```
  object1
  object2
  object3
  ...
  ```

  The names of output files are also given in the file "output_list" like

  ```
  object1f
  object2f
  object3f
  ...
  ```

The process to divide the object frames by the flat is carried out as follows:

```
imarith @input_list / flat @output_list
```

- hedit

  The task hedit ('header editor') can be used to quickly modify the FITS header by adding, deleting or appending selected keywords. The input is a FITS file or list of files, names of fields (keywords) to be edited (may be new keywords), and the value one wants to put. In case of deleting certain keywords one does not put any value, and changes the task's parameter delete to 'yes'.

  The header editor is capable of performing global edits on entire image databases wherein the new value of each field is computed automatically at edit time and may depend on the values of other fields in the image header. See the help for more details.

- hselect

  To extract values of certain keywords from the header one can use the task hselect from the package imutil. The input is a FITS file or list of files, names of fields (keywords) to be extracted and a boolean expression used to select files for extraction – only files that make the expression true will be taken into account, typing 'yes' will make the task work on all files. For example:

  ```
  >ec hselect @file_list OBJECT,$I "EXPTIME > 300"
  ```

  will list all the names of observed objects (header keyword `OBJECT`) and the names of corresponding fits files (special variable `$I`), for which the exposure time (header keyword `EXPTIME`) was longer than 300 seconds.

  See the help for the previous task (hedit) for more information about the expressions and other special variables that can be used.

- splot

  Spectral data are plotted by splot. The spectrum of the next or previous echelle order is displayed by inputting "shift + 0" or "shift + 9", that is, "`(`" or "`)`", respectively.

  The display ranges are changed as follows: one may shift to the so-called 'window mode' by inputting 'w' on the display. In this mode the key 'j' trims the left hand side of the data from the position of the cursor. Similarly, the keys 'k', 't', and 'b' trim the right hand side, upper part, and the lower part of the data, respectively, from the position of the cursor. The key 'a' shows the whole range of the data again. (The window mode is applied to only one key. So one use the above keys with 'w', e.g. 'w+j', 'w+t'.)

  The plot is printed by typing ":.snap" on the display. That is saved in an eps file by ":.snap epsf".

- implot

  This is useful to see a cross-cut image of a two-dimensional CCD data. The display ranges are changed by inputting, for example, ":x 1000 2000" or ":y 0 5000".

- spectplot

  To plot multiple spectra in one graph use the task specplot. These can be either different orders of the same echelle spectrum, or 1D (or a selected echelle order) spectra of different objects.

15.3 Some special attentions

Some special attentions for data reduction with IRAF are given in order to avoid frequently found troubles.

- IRAF should be started by ecl or cl command *in the directory in which login.cl file exists.*
- *Do not delete the "irafterm" window (in which spectra and other images are shown)* using the "close" (×) button of the window. If deleted, the IRAF must be re-started after killing the previous processes.

- HDS has two CCDs to cover longer and shorter wavelengths. As a result, two separate FITS files are produced by one exposure. The odd and even numbers in the file names correspond to the CCDs covering longer and shorter wavelengths, respectively. The data reduction procedures are applied separately to each file.

- The directory database: The detailed information for the aperture determination by apall and the wavelength calibration by ecident is recorded in "apXXXX" and "ecXXXX" files, respectively, in the directory database. When the reference data for apall and ecident are copied to other directory, the corresponding files in the database directory should also be copied to the corresponding directory.