Kast Data Reduction [v0.1]

I. Kast Suggested Calibrations

- 10 Bias (0s) frames
- 10 flat images (per setup and side)
- 2 Arc Calibrations (one per side)
- one Standard star (*for each setup*)

II. Pre-Reduction Setup

- Computer Resources
  - >300 MHz processor
  - 1G disk space / night
  - 300M RAM
  - Linux or Solaris
- Solaris WARNINGS (These do not apply to Linux)
  - Solaris+IDL does not release memory until IDL is exited
  - Consider exiting IDL occasionally and monitor memory usage
  - IDL+Solaris freezes up on the CPU intermittently
    * Possibly bug in licensing software
    * Forced to kill IDL process and restart
- Software
  - gcc or cc
  - IDL v5.4 or higher
  - IDL packages
    (a) **djs** IDL package (Schlegel)
    (b) **idlspec2d** IDL package (Sloan public)
    (c) **coyote** IDL package
    (d) **astron** IDL package (Goddard)
    (e) **xidl** IDL package (JXP)
    (f) **Kast** Kast package (JXP,GEP)
- Log sheets [somewhat optional]
- This cookbook

III. Initial Setup (Repeat for each night)

(a) Create a new directory for the night (e.g. 06aug02) and enter it
(b) Create a 'Raw/' directory and put all the raw data in it.
(c) If the filenames do not read b(r)####.ccd, it is quite likely the code will be unhappy
(d) gzip the data (gzip *.fits)
(e) Launch idl in the directory above Raw/ (e.g. 06aug02)
(f) **kast_strct** :: Create the Kast structure.
  - This structure organizes the entire night of data and is the most important file created.

- The routine creates a few things: (1) an IDL structure in memory with whatever name you choose (e.g. kast); (2) the file 'kaststrct.fits' which is a fits version of the structure; (3) the file 'kast.list' which is an ASCII version of the fits file which lists the values of some of the tags. To view the fits structure outside of IDL, I recommend the program 'fv' which I think stands for fitsview. It allows you to examine binary fits tables.
- If the NOEDIT keyword is not set, a gui will launch which allows some editing of the kast structure (see below).
  Example: IDL> **kast_strct**, kast, /MKDIR, [/NOEDIT]
  Time : <1s per image

(g) **kast_editstrct** :: Modify the Kast structure. The previous step creates the structure and takes a guess at the initial values of many of the tags based on the header card info. It is difficult, however, to automate all of the values for the tags and therefore the user should carefully check the structure. Also, the user should set flg_anly = 0 for all of images which should be ignored during data reduction (bad flats, etc.). For most of the important tags, one can use **kast_editstrct**. The rest must be done from the command line by hand or through a simple IDL script (recommended). The obvious tags to modify are:

- Obj :: Object name (this propagates into the final spectra and **should have no spaces!**)
- flg_anly :: Include in analysis (defaulted to 1 for yes)
- type :: OBJ, STD, DRK, TWI, QTZ, ARC, ZRO, ACQ, SLT, etc.
- setup :: 1L, 2L, etc. (unique integer for each instrument setup including slit width)

Example: IDL> **kast_editstrct**, kast
Time : User interaction

(h) Writing the Kast structure to disk:: In IDL you can modify the values of any of the tags. You can then save the structure in fits form and rewrite the ASCII file with the routine **kast_wrstrct**.
Example: IDL> **kast_wrstrct**, kast, FITS='name'
Time : fast

(i) Reading the Kast structure from disk:: **kast_ar** If no name is given, the file looks for the first fits file starting 'kast' that contains a '_'.
Example: IDL> kast = **kast_ar()**
Example: IDL> kast = **kast_ar(**'kast_name.fits'**)**
Time : fast

IV. Setup

- **kast_specsetup** :: This routine examines the kast structure and looks for calibration files associated with the various setups. It groups together exposures with identical Obj name and sets the obj_id tags accordingly. A summary of the Kast exposures is put in 'kast_specsumm.txt'.
  Example: IDL> **kast_specsetup**, kast
  Time : Fast

V. Process Flats

- This routines process the QTZ flats to create a response image (pixel by pixel variations) for the CCD.
- **kast_mkflat** :: Creates a stacked, normalized QTZ flat to correct pixel response variations. The routine stacks by median averaging the images then normalizes by fitting a b-spline to the collapsed 1-D version of the stack.
  Example: IDL> **kast_mkflat**, kast, setup, [side]
  Time : 1s per flat

VI. Arc Images

- **kast_mkarc** :: Process the Arcs. This step combines Arc frames, flattens the resultant image and outputs the final frame into the Arcs/ directory.
  Example: IDL> **kast_mkarc**, kast, setup, [SIDE=side]
  Time : <5s per image

0. Extraction

- The following routines all apply to a single object, i.e. multiple exposures of that object will be reduced together.
- Most of the following routines take the *kast* structure, and the *setup, side* and *obj_id* tags.

I. Process the Image

- **kast_proc** :: Flat field the Raw image. This routine takes the index number of the kast structure as input or keywords setup and obj. This is the integer in the first column of the file 'kast.list'. Output is in 'Final/' and is a flattened flux and inverse variance fits file (one gzipped fits file with two exentsions per image).
  Example: IDL> **kast_proc**, kast, setup, side, obj_id
  Time : <5s per image
  Image Check: **xatv**, 'Final/f_b(r)#.ccd'

II. Identify the Object

(a) **kast_fndobj** :: Allows the user to interactively identify the science objects (default is interactive selection). It smashes a region of the spectrum and finds all peaks containing 4 continuous pixels with 3 sigma significance. The region is taken to be 20 pixels to either side of SCICLM, which can be set explicity but by default is set to 700 and 515 for the 452/3306 and 300/7500 grisings using the d46 splitter and 783 for the 452/3306 grising using the mirror. For other setups there are no defaults and SCICLM should be set explicitly. If not interactive, the program chooses the brightest object within 5 pixels of the center of the slit. Set the keyword /AUTO to auto-identify the object and an aperture to mask for sky subtraction (not recommended). It then creates an object structure which stores the spectra of all objects identified in the slit.

specobjstrct

Tag	Type	Comment
field	' '	Name of field
slit_id	0L	Used to store the order number (0-9) corresponding to physical order (15-6)
obj_id	' '	ID value (a=primary, b-z=serendip, x=NG)
flg_anly	0	0=No analysis
exp	0.	
xcen	0L	Column where obj was id
ycen	0.	
flg_aper	0	0=boxcar
aper	fltarr(2)	Widths of aperture, 0/1 = bottom/top (pixels)
skyrms	0.	RMS of sky fit
trace	fltarr(5000)	
npix	0L	
wave	fltarr(5000)	
fx	fltarr(5000)	
var	fltarr(5000)	<= 0 rejected pix
flg_flux	0	0=f_λ, 1=f_ν
flux	fltarr(5000)	Fluxed data
sig	fltarr(5000)	Err in fluxed data
date	0.0d	
UT	' '	
img_fil	' '	
slit_fil	' '	
instr_strct	' '	e.g. wfccdstr fits file

(b) The routine also does a crude trace to the object based on the identified object peak location. This trace is used to mask out the object for sky subtraction.

(c) If the object to find is a standard star and is labeled STD in the *kast* structure, the /STD flag should be used along with the structure, setup and side. This increases the default aperture size for smashing in sky subtraction.

(d) Output is 'Extract/Obj_*name*.fits'
Example: IDL> **kast_fndobj**, kast, setup, side, obj_id, [exp, REFWV= /NOCLOB, /AUTO, /CHK, /STD]
Time : fast

III. Extraction

- **kast_extract** :: This procedure performs an extraction of the object for each exposure. The defaut method uses the Sloan **idlspec2d** package **extract_image**, which extracts by fitting a Gaussian profile to the object along with a low order polynomial for the sky. Setting the keyword **BOXCAR** causes the procedure to perform a boxcar extraction of the object instead. When using **BOXCAR** it's recommended to follow this procedure up with a call to **kast_crays**, which identifies cosmic rays in the spectra and alters their variance arrays accordingly (works only when there exist multiple exposures of the same object).

- All of the output spectra are written to the Obj structure

- A sky subracted image is added to the final image file stored in the directory bf Final, use the keyword **CHK** to view the sky, sky-subtracted imgage and, in the default case, the y-model calculated by **extract_image**.
Example: IDL> **kast_extract**, kast, setup, side, obj_id, [/BOXCAR]
Example: IDL> **kast_crays**, kast, setup, side, obj_id
Time : <10s per image

- Examine the final product:
  IDL> **kast_pltobj**, kast, setup, side, obj_id, /noimg, /nowv
  OR
  IDL> **xatv**, 'finalimage_name', getsky=2

IV. Wavelength Calibrate Exposures (Required before fluxing images)

- **kast_wavesol** :: This procedure wavelength calibrates the extracted spectra of a given object. The default launches an interactive GUI to calibrate the Arc image. It is recommended to use the **/auto** keyword, which automates the process, is quite robust, and much less time consuming, working by cross-correlating an archived wavelength solution with the extracted ARC to 'reidentify' the lines. It then performs a fifth order fit to the lines.
  Example: IDL> **kast_wavesol**, kast, setup, side, obj_id, /auto
  Time : <10s per image Examine the output with: IDL> **kast_pltobj**, kast, setup, side, obj_id, /noimg

V. Combine Multiple Exposures

- **kast_combspec** :: This procedure adds up multiple exposures of the same object, weighting by S/N after scaling each exposure by median ratio to the first. It appropriately deals with rejected pixels (e.g. flagged CR's). For objects with more than 2 exposures, it clips 5 sigma outliers as well. Even if you have only a single exposure, you need to run this routine prior to fluxing and collapsing to a 1D spectrum as it creates the final spectrum structure.

- Output is a fits file in 'FSpec' named 'Fspec_b(r)#.ccd.gz'.
  Example: IDL> **kast_combspec**, kast, setup, side, obj_id
  Time : fast
  Examine the output with: IDL> **kast_pltobj**, kast, setup, side, obj_id, /noimg, /comb

VI. Flux the Spectra

- **kast_flux** :: This procedure fluxes each extracted spectra or combined spectra for an object. Generally, I'd suggest using the flux calibration that is defaulted.

- When the keyword **/comb** is specified the program uses the combined spectrum and overwrites the flux array in the file created by **kast_combspec**; this keyword also directs **kast_pltobj** to use the combined spectrum for plotting.
  Example: IDL> **kast_flux**, kast, setup, side, obj_id, [/comb]
  Time : fast

- Examine the final product:
  IDL> **kast_pltobj**, kast, setup, side, obj_id, /flux, [/noimg], [/comb]