WFCCD Multi-Slit Reduction [v1.0]

I. Requirements

- IDL v5.4 or later
- >500M RAM
- >2G hard drive space
- The full software package, including the **djs** and **idlspec2d** codes
- Hopefully a 300MHz processor or better
- Slit mask information (ideally from Andy's maskgen program)
- Log sheets [somewhat optional]

II. Suggested Calibrations

- 2 QTZ Lamps at position of science frames
- 1 Arc on each side of the science frames
- (Optional) Bias frames to check CCD performance

III. Initial Setup (Repeat for each night)

(a) Create a new directory for the night and enter it

(b) Create a Raw/ directory and put all the raw data in it.

(c) Create a pro/ directory and copy the files from $XIDL/Spec/WFCCD/pro into it (e.g. *setcrds.pro, proc_night.pro, extract_night.pro*). **These files can be used to run most of the steps that follow.**

(d) Launch idl in the directory above Raw/

(e) **wfccd_strct** :: Create the wfccd structure. Again, I use the *proc_night.pro* file to do this. This structure organizes the entire night of data and is the most important file created. The structure is listed on the next page [v1.1].
The routine creates a few things: (1) an IDL structure in memory with whatever name you choose (e.g. night1); (2) the file 'struct.fits' which is a fits version of the structure; (3) the file 'wfccd.list' which is an ASCII version of the fits file which lists the values of some but not all tags. To view the structure outside of IDL (or even in general), I highly recommend the program 'fv' which I think stands for fitsview. It allows you to examine binary fits tables.
Example: IDL> **wfccd_strct**, night1, /MKDIR
Time : < 1s per image

(f) Modify the WFCCD 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 impossible, however, to automate all of the values for the tags and therefore the user must do a fair amount of this by hand. The file *setcrds.pro* is an extensive example. Note the IDL routine **x_fmidx** converts the frameno of an image to the index in the structure.
The obvious tags to modify are:

  i. Obj :: Object name
  ii. flg_anly :: Include in analysis (defaulted to 1 for yes)
  iii. type :: ARC, FLT, etc. (see the struct definition)
  iv. masknm :: Generally a two character string giving the unique def
  v. mask_id :: Long integer identifying a given mask. This is the most important
  vi. msk_fil :: Mask file describing the slits. You will always need to set this tag. Ideally, the output from Andy's *maskgen* program. I tend to put these files in the Masks/ directory.

(g) Updating the WFCCD structure :: 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 **write_wfccdstr**.
Example: IDL> **write_wfccdstr**, night1, FITS='name'
Time : fast

(h) OV region :: By default, the package will assume 32 columns of overscan. If you use less, the program will behave fine. If you use more, they will be ignored.

wfccdstr

Tag	Type	Comment
frame	0	FRAME Number
flg_anly	0	Analysis flag 0=Don't Analyse
Obj	' '	Object Name
type	' '	ObjTyp : OBJ, STD, DRK, ZRO, FLT, ARC, MSK, ALG
mask_id	0L	Mask ID
masknm	' '	Mask Name
exp	0.d	Exposure time
filter	' '	Filter : U,B,V,R,I, C
filtpos	0	Filter Position
grism	' '	Grism : BG, RG, NO
aperpos	0	Aperture position
casspos	0.	CASS pos angle
AM	0.	Airmass
CCD	' '	CCD
TEL	' '	Telescope
gain	0.	Gain
readno	0.	Read Noise
date	0.0d	Date of Obs
UT	' '	UT
RA	' '	RA
DEC	' '	DEC
Equinox	0.	EQUINOX
rootpth	' '	Path of the Root
img_root	' '	Root name (usually in Raw directory)
flg_ov	0	OV FILE? 0=No, 1=Yes
img_ov	' '	Name of OV file (with directory)
flg_msk	0	Mask FILE? 0=No, 1=Yes
img_msk	' '	Name of Mask file
flg_final	0	Final File? 0=No
img_final	' '	Name of Final img
nslits	0	Num of slits
ystrt	0L	Column for initiating the trace
msk_fil	' '	Name of the Mask info file (fits)
slit_fil	' '	Name of the Slit info file (fits)
arc_fil	' '	Name of the Arc image file (fits)
map_fil	' '	Name of the Map image file (fits)
flat_fil	' '	Name of the Flat image file (fits)
obj_fil	' '	Name of object structure (fits)

IV. Create bias frame (optional)

(a) As a check you might create a bias frame. At the moment the reduction process does not include the bias frame.

(b) Run **wfccd_bias**. This routine outputs an image 'Bias.fits' in the Bias/ directory. The routine creates and then deletes overscan frames for each image.
Example: IDL> **wfccd_bias**, night1
Example: IDL> **proc_night**, night1, /MKBIAS
Time : 3-5min

(c) To check the Bias frame : **xatv**, 'Bias/Bias.fits'

0. Mask Reduction

- The following routines all apply to a single mask. In general, this means multiple science exposures of one designed mask. One could break these up too, however.
- I suggest you create a .pro file like *proc_night.pro* to guide the process. It provides a convenient way of stepping through the reduction.
- Most of the following routines take only the WFCCD structure and the *maskid* as inputs.

I. Combine Flats

- Generally, one has at least 2 flats per mask. To remove CR, if nothing else, I combine the images.
- Run **wfccd_mkflat** :: This routine uses **x_addtwoflats** to combine 2 flats if there are only two or **x_combine** for more than 2 flats.
- Output is 'Flats/Flat_*maskid*.fits'
  Example: IDL> **wfccd_mkflat**, night1, maskid
  Example: IDL> **proc_night**, night1, maskid, /MKFLAT
  Time : < 1min
- Check the Flat: **xatv**, 'Flats/Flat_*maskid*.fits'

II. Create the Map

- **wfccd_mkmap** :: This step produces a fits file which contains information relating the original data frame to one where the slits run straight accross the image. The 'straightened' image has the same number of total rows as the original.
- The main routine **x_traceflat** takes the following steps:
  (a) transposes the flat image
  (b) creates a sawtooth image (shifts and subtracts) to highlight slit edges
  (c) Keys on column (row) 400 to find all significant peaks in the sawtooth image
  (d) Uses **trace_crude** to trace the slit edges
  (e) Outputs a trace structure **tracestrct**
- The user then interactively checks the various traces with the routine **x_ydtortgui**.
  - You can resize this gui
  - LMB : Stretches the contrast and brightness.
  - RMB : Edit the ends of each trace. The program assumes you have chosen the nearest edge and changes good to bad or vice-versa.
  - DELTRC : Delete a given trace altogether.

  I advise you extend the RHS of traces where it is sensible as there is generally little signal out at these edges. This program is one of the most user intensive of the entire reduction package. I intend to automize it entirely
- Finally, I run **x_fit2dtrace** to fit a 2D polynomial to the traces and create a map of the transformation. I also create the inverse map.
- Output is 'Maps/Map_*maskid*.fits' and 'Maps/Trc_*maskid*.fits. The map files are saved as double images and take 67M memory.
  Example: IDL> **wfccd_mkmap**, night1, maskid
  Time : 5min to check the trace + 5min for the Map creation

- Check the Map
  (a) Read in the flat: flat = mrdfits('Flats/Flat_*maskid*.fits')
  (b) Read in the map: map = mrdfits('Maps/Map_*maskid*.fits')
  (c) Rectify: rflat = x_rectify(flat, map)
  (d) Examine: **xatv**, rflat

III. Create the Slit Structure

(a) **wfccd_setslits** :: This step creates a slit structure which describes the slits in a mask in greater detail than the WFCCD structure. The user can interactively set the slit edges. This is important for dealing with slit overlap. Here is the structure:

mslitstrct

Tag	Type	Comment
flg_anly	0	Analyse? (1=yes, 0=no)
id	0L	ID number
field	' '	Field name
length	0.	(arcsec)
width	0.	(arcsec)
pa	0.	
arcpix	0.	Arcsec per pix
xyqso	fltarr(2)	x offset from the QSO (arcsec)
xpos	0.	x position of silt
nobj	0	Number of objects in slit
ypos	fltarr(10)	y pos of obj relative to slit edges (%)
yedg	fltarr(2)	Slit edges relative to slit mask cen (pix)
yedg_flt	fltarr(2)	y-Edges of slit on CCD (Undistorted FLAT)
yedg_orig	fltarr(5000,2)	Array of slit edges in orig
yedg_sky	fltarr(5000,2)	Edges of slit for sky subtraction

(b) Parse the slit file listed in the WFCCD structure under the 'msk_fil' tag. You will need to have put your mask file in the appropriate place beforehand (e.g. 'Masks/'). The routine is **wfccd_prsslits**. The slit structure is created in memory.

(c) Straighten the Flat with the Map file (uses **x_rectify**).

(d) **x_setslits** :: Guess at the slit positions using the info in 'msk_fil'

(e) **x_stsltgui** :: Unless NOVERIFY is set, launch a gui to interactively check the slits. It is important to set the silt edges about right. In particular, overlapping slits leads to a real problem with sky subtraction and they are best dealt with by identifying the slit edges properly.
  - You can resize the display screen
  - I would examine the LHS of the image where there is more flux
  - The slit width is always preserved
  - Red (blue) lines indicate the top (bottom) of a slit
  - z (Z) zooms in (out)
  - LMB sets contrast/brightness
  - s (S) shifts all slits below the selected one (inclusive) and keys on top (bottom) of slit
  - b (t) shifts one slit and keys on bottom (top) of slit

(f) Map the slits into the original frame using **x_origslit**. The slit edge traces are written into 'yedg_orig' in the slit structure.

(g) Output the Slit structure to 'Slits/Slits_*mask_id*.fits'
Example: IDL> **wfccd_setslits**, night1, maskid
Time : < 5min to check the slits

IV. Normalize the Flat

- **wfccd_normflat** :: Collapses (median) the slits in the flat using the slit edges from the previous step. Fits a high order bspline to the final result (with 5sig rejection).
- The main driver is **x_normflat**.
- The output is 'Flats/FlatN_*mask_id*.fits'. The WFCCD structure is updated to reflect the new flat. At this point the file 'Flats/Flat_*mask_id*.fits' is unnecessary (33M). I'd check the normalized flat before ever deleting it, however.
  Example: IDL> **wfccd_normflat**, night1, maskid
  Time : < 3min

V. Create the Wavelength Image

(a) This sequence of steps (i) processes the arc images, (ii) creates a wavelength solution for the central row of each slit in the straightened frame; and (iii) finds the wavelength solution for each pixel and maps back to the original frame.

(b) **wfccd_procarc** :: Process each Arc image. OV subtract, flatten, straighten the slits. Output is 'Arcs/ArcR_ccd###.fits' where ccd### corresponds to the orginal ccd frame number.
Example: IDL> **wfccd_procarc**, night1, maskid
Time : < 1min

(c) **wfccd_allarcsol** :: Find the wavelength solution for the center of each slit to be analysed (alignment star boxes are skipped). This routine drives the program **wfccd_arcsol** which:

- Medians the 5 central rows of each slit
- Guesses the initial wavelength solution based on the slit position
- Calls **x_autoid** which is a χ^2 minimization routine that automatically finds a wavelength solution given an initial guess, an arc spectrum, and a line list. By default we use the linelist 'wfccdB_HeNe.lst' given in $XIDL/Spec/Arcs/Lists
- Output
  is an Arc structure **wfccdarcstr** which is output as a fits file 'Arcs/ArcS_ccd###.fits'. It has the wavlength solution, arc spectrum, and a wavelength array for the center of each slit in the straightened frame. Example: IDL> **wfccd_allarcsol**, night1, maskid
  Time : $\approx$ 10s/slit

(d) **wfccd_arcimg** :: Translate the wavelength solution from the center rows to the entire slit. This is the only procedure which is exposure sensitive. The main driver is **x_arcimage**:

- Identifies all 3σ arc lines and traces them across the slit.
- It then calculates a unique wavelength solution for each row.

Finally, the program maps the wavelength solution back into the original frame. The routine is called once per exposure for a given mask. It matches each exposure with the nearest Arc frame by considering the UT of the images. The output is an Arc image 'Arcs/ArcI_ccd###.fits'.
Example: IDL> **wfccd_arcimg**, night1, maskid, expsr
Time : $\approx$ 1min/exposure

(e) **wfccd_chkarc** :: Plots the arc line solutions from each slit in a simple gui. Shows the RMS solution (Å) and the red lines identify key arc lines. If something isn't right, you are in trouble!
Example: IDL> **wfccd_chkarc**, night1, maskid, expsr
Time : fast

(f) **wfccd_cleanarc** :: Zeros out bad wavelength values. These should only occur when two slits are too close or overlapping. This procedure helps with the sky subtraction and extraction.
Example: IDL> **wfccd_cleanarc**, night1, maskid, expsr
Time : < 1min

VI. Process the Science Images

- **wfccd_procobj** :: OV subtract, flatten and staple in the wavelength image.
- Produces one final fits image: 'Final/f_ccd###.fits'. This file has Flux, Var, Wave in the first 3 data units (50Mb).
Example: IDL> **wfccd_procobj**, night1, maskid
Time : fast

VII. CR Flagging

- **wfccd_objcr** :: Identifies CR in the object frames by comparing two exposures. Currently can only take two images at a time but this is easily updated. A CR must have a flux $> 3\times$ the pixel in the other frame and a flux > 200.
Example: IDL> **wfccd_objcr**, night1, maskid
Time : < 2min

VIII. Clean Up

- **wfccd_cleanup** :: Removes many of the unnecessary files that the package creates (e.g. the unnormalized flat). Don't run this unless you are quite confident your processing is complete.
Example: IDL> **wfccd_cleanup**, night1, maskid
Time : fast

0. Mask Extraction

- The following routines all apply to a single exposure
- I suggest you create a .pro file like *extrct_msk.pro* to guide the process. It provides a convenient way of stepping through the reduction.
- Most of the following routines take the WFCCD structure, the *maskid*, and the exposure number as inputs.

I. Create Object Structure

(a) **wfccd_mkobjstr** :: Creates the object structure which stores the spectra of all objects idenitified in the mask.

(b) Options:

- /NOSKY :: Will not modify the region for sky subtraction
- /NOCHK :: Will not run **x_setobjgui**
- /NOSLIT :: Will not modify the Slit structure
- /NOOBJ :: Will not modify the Obj structure
- SKYOFF :: Offset of sky edge from slit edge (default=2pix)

specobjstrct

Tag	Type	Comment
field	' '	Name of field
slit_id	0L	
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

(c) The program first runs **x_fndslitobj** which automatically looks for the main science object (as defined in the Mask output file) and any serendipitous objects.

(d) The user should then run **x_setobjgui** to confirm the object locations and set the slit edges to their 'final' locations for sky subtraction. Rough centroid for the objects is good enough. *In general, sky subtraction works best on regions $\approx$ 25 pix wide.* Beware of objects near the slit edge. Try to extend the sky regions beyond them.

- xatv-like display (use the Help window)
- Green (blue) lines trace science (serendipitous) objects

- Red crosses identifies the guess from the slit info
- 'm' moves the closest object to the cursor position
- 'd' deletes closest obj
- '[]' = pan up,down,left,right
- b (t) shifts the slit edge at the bottom (top) of slit

(e) Output is 'Extract/Obj_*maskid*.fits' and the Slit structure is updated.
Example: IDL> **wfccd_mkobjstr**, night1, maskid, expsr
Time : 3min/exposure for slit+obj checking

(f) **wfccd_editobjstr** :: Edit the Obj structure. Options: /NOOBJ, /NOSLIT.
Example: IDL> **wfccd_editobjstr**, night1, maskid, expsr
Time : fast

II. Sky Subtraction

- **wfccd_skysub** :: This procedure subtracts the sky from each slit, one by one. The main driver is **x_subskyslit** which does a LEGENDRE fit (the better of 2nd or 3rd order provided enough pixels) to each column after masking out the objects in the slit. It works on individual exposures. Currently, my routines do not handle objects right at the slit edge very well.
- Output is the sky subtracted 2D frame 'Extract/F_ccd###.fits' with the appended Sky image and the rms of the fit along each column for each slit (35M).
Example: IDL> **wfccd_skysub**, night1, maskid, exposure
Time : 20-30s/slit (roughly 10-15min)

III. Extraction

(a) **wfccd_extobjbox** :: This procedure extracts the objects from the slits one by one. It works on individual exposures seperately. The main driver is **x_extobjbox** which

i. Traces the object with **trace_crude**
ii. Smashes the object to determine its profile
iii. Determines an aperture which includes 95% of the flux
iv. Identify pixels included in the aperture and what fraction
v. Flags CR's by comparing the flux in a given column against the object profile. CR events have variance set to -1.
vi. Calls **x_rebin2dspec** which sums the flux into a 1D array with starting wavelength 3200Å and $\Delta v = 100$km/s pixels

(b) All of the output is written to the object structure
Example: IDL> **wfccd_extobjbox**, night1, maskid, expsr
Time : < 5min

(c) Run **wfccd_objsumm** to get a summary of the Fspec file.

IV. Check/Edit the Individual Spectra

- **wfccd_chkspec** :: This routine plots up all of the objects in the object structure for a quick glance. It is a very simple gui. Bad extractions are easily identified and can be removed by adjusting the object structure 'by hand'.
Example: IDL> **wfccd_chkspec**, night1, maskid, expsr
Time : fast

- **wfccd_pltobj** :: This routine plots the spectrum and the 2D image together (very nice). You can examine the 'reality' of various emission and absorption lines.
  Example: IDL> **wfccd_pltobj**, night1, maskid, expsr, [objnm]
  Time : 5s

- **wfccd_editspec** :: This routine allows the user to set the variance of regions of spectra to 0 to eliminate them from further analysis.
  Example: IDL> **wfccd_editspec**, night1, maskid, expsr
  Time : fast

V. Flux the Spectra

- **wfccd_fluxspec** :: This routine plots up all of the objects in the object structure for a quick glance. It is a very simple gui. Bad extractions are easily identified and can be removed by adjusting the object structure 'by hand'.
  Example: IDL> **wfccd_fluxspec**, night1, maskid, [expsr]
  Time : fast

VI. Combine the Spectra

- **wfccd_combspec** :: This routine combines multiple object structures (primarily their spectra) into one Final structure. For objects with multiple spectra, the spectra are matched to the same flux and summed. At the moment this routine only handles two spectra at a time.

- Output :: A structure containg all of the fluxed, coadded spectra with name : *'Extract/Fspec_maskid.fits'*. Example: IDL> **wfccd_combspec**, night1, maskid
  Time : fast

- Run **wfccd_fspecsumm** to get a summary of the Fspec file.
- Run **wfccd_chkfspec** to view the Final spectra
  Example: IDL> **wfccd_chkfspec**, 'Extract/Fspec_maskid.fits'
  Time : fast

VII. Solve for Redshifts

- **wfccd_zfind** :: This routine drives the SDSS **zfind** routines and uses the SDSS eigenspectra (rebinned to WFCCD pixel scale) to automate redshift identification. It is a χ^2 minimization routine and it works rather well. Like all χ^2 routines, however, it tends to fail when there are particularly bad regions of data. The ZANS structure in the Fspec structure is updated.
  Example: IDL> **wfccd_zfind**, night1, maskid
  Time : Long, $\approx$ 30s per obj

- Run **wfccd_chkfspec** with the /ZFIND switch to view the Final spectra +
  Example: IDL> **wfccd_chkfspec**, 'Extract/Fspec_maskid.fits', /zfind
  Time : fast