

MULTISPEC

**A code for the extraction of
slitless spectra in crowded fields**

v2.0.1

March 2007

Jesús Maíz Apellániz

Instituto de Astrofísica de Andalucía

1 A quick summary

MULTISPEC is a code for the automatic extraction of multiple spectra from slitless exposures of dispersing elements such as objective prisms, grisms, or gratings. The code has been optimized for crowded fields composed of point sources such as resolved stellar clusters. MULTISPEC obtains the spectra by multiple profile fitting and is a spectral analog of a profile-fitting photometry package such as DAOPHOT or HSTPHOT. The first versions of MULTISPEC (1.0.x, 1.1.x) could only be used with HST/STIS NUV-MAMA objective-prism data. Version 1.2 added support for HST/STIS FUV-MAMA G140L data. For version 2.0 the full code was rewritten in order to generalize it to arbitrary slitless data. The current version includes the calibration files for the modes cited above plus a generic uncalibrated mode. Future versions will add calibration files for more modes.

2 Installing MULTISPEC

Start by creating a directory where the program will be located. We will call the path to that directory `MULTISPEC_DIR` and it should be included in your `IDL_PATH` system variable before you invoke IDL (if you need help on setting up `IDL_PATH`, see <http://www.linuxhelp.net/forums/lofiversion/index.php/t4853.html>). Now, go to the web site <http://www.iaa.es/~jmaiz> and download the file `multispec.tar.gz` into that directory. At the `unix shell >` prompt in `MULTISPEC_DIR` type:

```
unix shell > gunzip multispec.tar.gz
unix shell > tar xfv multispec.tar
... lots of files ...
unix shell > ln -s multispec_6.3.sav multispec.sav
```

And with that you are (almost) ready to start using MULTISPEC! But before we do that, here is some additional information:

- MULTISPEC is written in IDL and requires that program to be installed with a license.
- If your version of IDL is earlier than 6.3, substitute 6.3 by 6.0 in the `ln` command. IDL versions prior to 6.0 are not supported, sorry.
- You should not need the tar files anymore so if disk space is an issue, go ahead and delete them.
- MULTISPEC requires CHORIZOS to be correctly installed. CHORIZOS is available from the same web site as MULTISPEC.

3 Some background on slitless spectroscopy

First-order spectroscopy using a small slit is an inefficient use of a bidimensional detector since only a small fraction of the pixels play a role in the detection of the signal. A long slit is an improvement if one needs to measure several sources in the same field but, unless all of your targets are aligned, is still quite inefficient since several exposures are needed to measure all the sources. Several alternatives (e.g. multiple fibers, integral field units, micromirrors, or microshutters) are present in some current or planned telescopes to maximize the simultaneous collection of spectral data from multiple sources, but in others (e.g. HST) the only available alternative is the use of slitless exposures and a dispersing element, be it a grating, an objective prism, or a grism.

The use of slitless exposures invokes several possible complications. The first one, spectra overlapping due to crowding, is illustrated in Fig. 1, which shows slitless exposures of a single star, a sparse field, and a crowded one. In the first case (one source) it is straightforward to assign the origin of each detected count to its source. In the second case (few tens of sources)

we have the same situation for most stars in the field (though one of the bright sources in this specific example turns out to be a binary and there is a significant overlap between the two spectra). In the third case (few hundreds of sources) most sources overlap to some degree with neighboring ones and the assignment of an origin to a given detected count can be quite complicated.

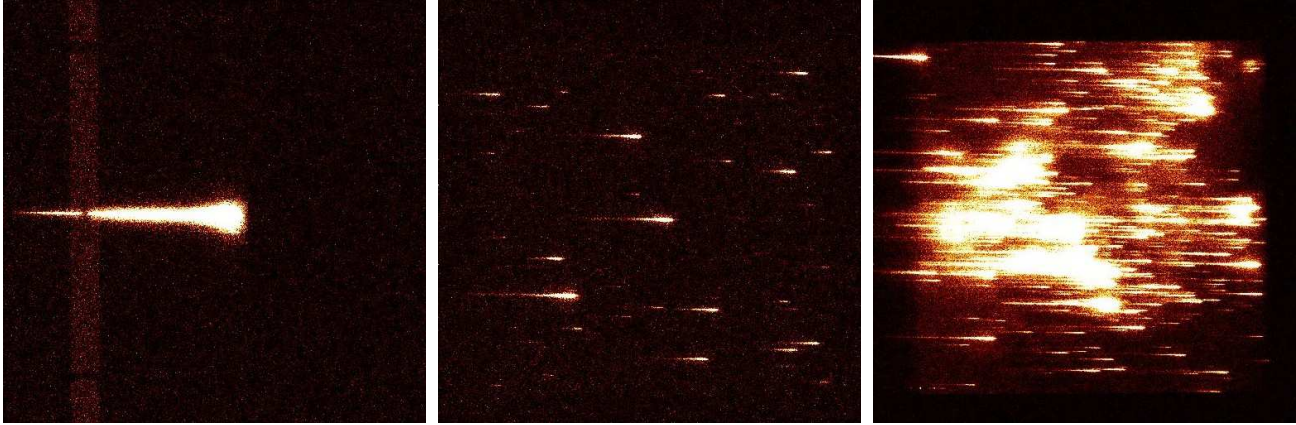


Figure 1: STIS objective-prism exposures of (left) a single star, (center) an sparse field, and (right) a crowded field. The spectral direction is nearly parallel to the x axis.

A second problem is the treatment of extended objects, either sources or background¹. An extended source observed in slitless mode will be in general spread out in both the dispersion and the cross-dispersion directions. The latter is only a minor complication if there is no significant velocity structure in the extended emission; it can be solved by integrating over a larger window (if using aperture extraction) or convolving a model of the spatial profile with the instrumental cross-dispersion profiles (if doing profile fitting). The former is more serious, since it translates into a degrading of the spectral resolution. An extended background is an added problem for slitless exposures as compared to ones with a slit. We can distinguish two different origins for the background: astronomical objects, such as nebular emission from an H II region or a planetary nebula or the continuum from an unresolved stellar population; and foreground sources, such as geocoronal emission lines, atmospheric emission lines and continuum, zodiacal light, or earthshine. Foreground sources are usually rather uniform and they only complicate the extraction by degrading the S/N ratio for weak sources. Astronomical objects, on the other hand, can be a more serious problem, since they are likely to be spatially variable and to introduce confusion and larger uncertainties in the extraction.

A third problem, related to the first one, is the identification and positioning of the individual sources. If crowding is severe, as in the right panel of Fig. 1, it may be hard not only to separate different spectra but even to identify how many there are. But even if crowding is not severe, as in the middle panel of Fig. 1, the question remains as to how one measures the offset (or zero position) in the spectral direction.

MULTISPEC is a code that can be used to extract multiple spectra from slitless exposures that addresses the above problems. This document describes how that is done, the different modules that make up the code, and the requirements to install and use MULTISPEC. The original version of the document was published by STScI as Maíz Apellániz (2005).

4 Techniques

The goal of extracting multiple spectra from a slitless exposure is, in several ways, similar to that of extracting the photometry of a collection of point sources in an image. In the latter case, when one is analyzing a sparse field the most straightforward method is aperture photometry with a small radius. However, for a crowded field the aperture photometry of a star can easily be contaminated by counts produced by neighbors, and a profile-fitting (sometimes called crowded-field) code such as DAOPHOT, DOPHOT, or HSTPHOT does a better job. For slitless spectroscopy, crowding is a larger problem than for photometry

¹We use the term background here in the general meaning of anything that is not the source, even though those photons may actually originate behind, at the same distance as, or in front of the source.

because a given source produces a significant number of counts over a larger number of pixels. Hence, given the same number of sources, chances of having an overlap are higher. Therefore, I decided to use profile fitting as the technique for MULTISPEC instead of aperture extraction, the spectroscopic equivalent of aperture photometry.

There are several issues which need to be addressed in fitting profiles to multiple spectra:

- A difference between a photometry- and a spectroscopy-profile-fitting code is that the former has no preferred direction in the detector and, for that reason, it uses 2-D profiles as its fundamental unit. Spectroscopic exposures, on the other hand, have a distinct asymmetry in that one direction shares the dual character of being an angular – or spatial – and a wavelength – or spectral – coordinate (that direction will be called here the dispersion direction). Furthermore, the position of the object in the perpendicular angular coordinate (the cross-dispersion direction) is fixed over its extent. For those reasons, it makes more sense to fit 1-D profiles in the cross-dispersion direction once we have determined the position of the object on the 2-D detector (i.e. its trace).
- Another difference between the two types of code is the need for auxiliary exposures. A photometry package can use one image to identify the sources and to extract the magnitudes of the objects. Doing so in a spectroscopy code is more difficult, given the possible extensive overlap among sources. For that reason, an image (or images) of the same field is a preferred alternative for source identification.
- The use of two different types of observations, an image (or images) for source identification and a slitless spectral exposure for the measurement of spectra, introduces an added complication: the need to accurately know the relative geometric distortion between the two in order to produce a one-to-one correspondence between the positions in one and the other throughout the field covered. These geometric distortions can be calculated by obtaining calibration spectral slitless exposures and images of a crowded field. The images are used to generate a simulated slitless exposure of the field without any distortion which can then be compared with the real exposure. Note that, due to the degeneracy between spatial and spectral coordinates in the dispersion direction, the precision of the geometric distortion solution in that direction is likely to be poorer than in the cross-dispersion direction unless the presence of absorption/and or emission lines is ubiquitous in the sources, their intensities and wavelengths are well known, and the dispersing element can resolve them.
- As is also the case for crowded-field photometry, a detailed knowledge of the spatial (cross-dispersion) profile² is required for a crowded-field spectroscopy code. The PSF is likely to be wavelength dependent and, if it is not well known, systematic errors can be introduced, which would be small for bright stars and caused by an incorrect aperture correction but possibly large for dim stars, especially those close to bright ones.
- The ability of a profile-fitting code to accurately deconvolve the fluxes of a close pair is expected to depend heavily on the S/N ratio of the data (Porter et al. 2004). Therefore, we should expect different uncertainties for stars which have the same number of counts at the same wavelength depending on the presence or absence of neighbors.

With those issues in mind, the problem can be established in the following way. For the sake of notation simplicity, we will assume that the dispersion direction is parallel or nearly-parallel to the x axis (this is the case for the HST/STIS modes). Then, the 1-D profile fitting can be done in the y direction and we can work directly with column data. Let $C(i, j)$ be the number of counts at the pixel with coordinates (i, j) after the background has been subtracted. We are ultimately interested in measuring the spectral fluxes $F_k(\lambda)$, ($k = 1, N$) of the N stars present in the exposure but our immediate goal is to find the contribution $C_k(i, j)$ of star k to pixel (i, j) subject to the restriction:

$$C(i, j) = \sum_{k=1}^N C_k(i, j). \quad (1)$$

$C_k(i, j)$ is the result of the discretization in the vertical (cross-dispersion) direction of:

$$\frac{F_k(\lambda_k(i)) \cdot t_{\text{exp}} \cdot P_\lambda(y - y_k(i))}{s(\lambda_k(i))}, \quad (2)$$

²By analogy with the imaging case, from now on we will call the spatial profile PSF.

where $\lambda_k(i)$ is the relationship between wavelength and the horizontal pixel index i for star k (which depends on the dispersion relation, the position of the star in the reference image, and the geometric distortion³), $P_\lambda(y - y_k(i))$ is the normalized PSF centered at the continuous vertical coordinate $y = y_k(i)$, $y_k(i)$ is the position of the center of the profile of the star k at the pixel with horizontal index i , t_{exp} is the exposure time, and $s(\lambda)$ is the sensitivity function in units of $(\text{erg s}^{-1} \text{ cm}^{-2} \text{ \AA}^{-1})/(\text{cts s}^{-1} \text{ pixel}_\lambda^{-1})$. $P_\lambda(y)$ is a function that has to be determined from high S/N observations that allow for a determination of the PSF at the sub-pixel level. Finally, $y_k(i)$ is calculated from [a] the position of the star in the image, [b] the relative geometric distortion between the image and the spectral exposure (plus any possible offsets and/or rotations), and [c] the trace produced by the spectrum of a star centered on the detector (described by a function $y(x)$), which we will assume does not vary for stars located off-center (apart from the relative displacement in x and y derived from [a] and [b]).

If we now select the column defined by i , we will find that the traces of N' stars ($N' \leq N$) cross it (for some dispersing elements, e.g. the STIS objective prism, the trace of an individual star does not span all the columns in the detector). Then, for that column we have N_j equations (where N_j is the number of pixels in a column) defined by Eq. 1 and N' unknowns (the $F_k(\lambda_k(i))$ for the N' stars). If we impose the reasonable condition that $N' < N_j$ (certainly, some crowding may be expected but it would be hard to measure more than one spectrum per pixel!), we find that the system is overdetermined. We can solve for such a system minimizing the χ^2 -like function:

$$g_i = \sum_{j=1}^{N_j} \left(C(i, j) - \sum_{k=1}^{N'} C_k(i, j) \right)^2 = \sum_{j=1}^{N_j} \left(C(i, j) - \sum_{k=1}^{N'} \frac{F_k(\lambda_k(i)) \cdot t_{\text{exp}} \cdot \text{discretize}_j[P_\lambda(y - y_k(i))]}{s(\lambda_k(i))} \right)^2 \quad (3)$$

for the N' unknowns $F_k(\lambda_k(i))$. In Eq. 3 we discretize the PSF by integrating the sub-pixel precision $P_\lambda(y - y_k(i))$ over the extent of a vertical pixel and, in that way, we take into account the different centering possibilities (e.g. pixel-centered or edge-centered vertical profiles). This is the equivalent of using ePSFs for crowded-field photometry (Anderson and King 2000).

Minimizing a χ^2 -like function which is as highly-non-linear and has a potentially large number of parameters as the one in Eq. 3 is easy to program but dangerous to apply blindly. The main reason is that accurate initial guesses for the parameters are required: otherwise, the algorithm can easily end up in a local minimum which is not the correct one. Furthermore, the existence of noise or incorrect background subtraction or knowledge of the PSF can also lead to unphysical (i.e. negative) results for $F_k(\lambda_k(i))$. These issues can be avoided by:

- Providing as good an initial guess for the parameters as possible. In MULTISPEC, any input spectral energy distribution (SED) can be used and normalized to the measured magnitude, with the additional possibility of extinguishing it, to estimate $F_k(\lambda_k(i))$. Regarding how to select the best SED that matches the observed photometry, the reader is referred to CHORIZOS Maíz Apellániz (2004), which is also available from <http://www.iaa.es/~jmaiz>.
- Using a code to solve Eq. 3 capable of placing constraints on the parameters in order to detect unphysical or unrealistic cases and to minimize the consequences on the calculated fluxes of neighboring sources.

Another issue is the treatment of the background, which needs to be subtracted before Eq. 3 is solved. The algorithm I use to calculate the background in a potentially crowded field involves the following steps:

1. Identification of the region of the detector where sources provide the dominant contribution to the total number of counts in a given pixel. That zone will be called the source region and the complementary zone the background region.
2. An optional estimation of the residual effects of the neighboring sources on the background region and a subtraction of that estimate from the original data pixel by pixel.
3. Calculation of a background map in the background region, possibly using spatial smoothing.

³Throughout this discussion I will assume for simplicity that $F_k(\lambda)$ is a quantity already discretized in wavelength and I will refer to discretization only in the spatial y coordinate.

4. Calculation of the background in the source region by interpolating from the background calculated in the previous step.
5. An optional iteration to improve the definition of regions in the first step and the contamination from sources in the second one.

If the background is complex, the implementation of the above algorithm is far from trivial and it is likely to require fine tuning of the region selection and smoothing parameters in order to be accurate. Furthermore, if the background is rapidly varying and bright close to some sources, a precise solution may not be possible from the data (this is a larger problem for slitless spectroscopy than for photometry) and we may have to settle for an approximate one. In the next section I will discuss the options available in MULTISPEC for fine-tuning the background calculation.

Even after all the above measures have been taken, it is possible that some sources are nearly exactly aligned in the dispersion direction so that it will be impossible to unambiguously deconvolve their individual fluxes using that exposure. For that reason, a code should be able to identify those cases and warn the user of the problem. Also, it would be convenient to have a post-processing tool that combines the data from exposures with different orientations in order to use alternative data when such a circumstance arises.

5 MULTISPEC modules and their use

MULTISPEC is a profile-fitting crowded-field slitless-spectroscopy code that has been written in IDL and which consists of five different modules: PREPDATA, GENGUESS, CALCBACK, MAINFIT, and WRITEOUTPUT. Each module can be called separately or, alternatively, a wrapper named MULTISPEC itself can be used to call each of them sequentially. Figure 2 shows the basic flow diagram for the five modules that make up MULTISPEC and briefly describes the tasks performed in each one of them. Figure 3 shows the public (non-hidden) parameters and keywords available for the wrapper and each of the five modules.

MULTISPEC was originally written for the HST/STIS NUV-MAMA objective prism and in its current version it also includes support for FUV-MAMA G140L data and for an uncalibrated (in wavelength) generic mode. Other observing modes can be included with the addition of the proper calibration files (see next section) by the user. My plans are to add more calibration files for other HST and non-HST modes in the future.

The current version of MULTISPEC can handle dithered observations analyzed individually at each dither position, since it allows for the treatment of a number of image/slitless spectral exposure pairs (note, however, that MULTISPEC does not use the drizzled data itself, see below). The minimum information that is needed for a MULTISPEC run in any of the modes with existing calibration files are:

- **A 2-D FITS file containing a slitless or long-slit exposure.** Note that MULTISPEC expects the 2-D FITS file to be in the original, geometrically distorted reference frame, in order to correctly handle count statistics. If one is using MULTIDRIZZLE to clean cosmic rays and combine several dithered HST spectral exposures, it is recommended that: [a] the SCI_COR output (cosmic-ray cleaned, geometrically-distorted data) is used as input for MULTISPEC; [b] each spectral exposure is processed separately; and [c] the resulting 1-D spectra for each star are combined a posteriori.
- **An ASCII table with the file names of the spectral exposure(s)** (and, optionally, associated images). The file is described in more detail in the next subsection (see `input_files`).
- **An ASCII table with the ID, positions, input SEDs, and magnitudes for each star.** The file is described in more detail in the next subsection (see `input_phot`).

5.1 The wrapper MULTISPEC

We start describing the wrapper MULTISPEC, a wrapper that can be used to sequentially execute all the modules. Note that the first two entries for PREPDATA in Fig. 3 are IDL positional parameters that can also be used for the wrapper. The

MULTISPEC MODULES DESCRIPTION

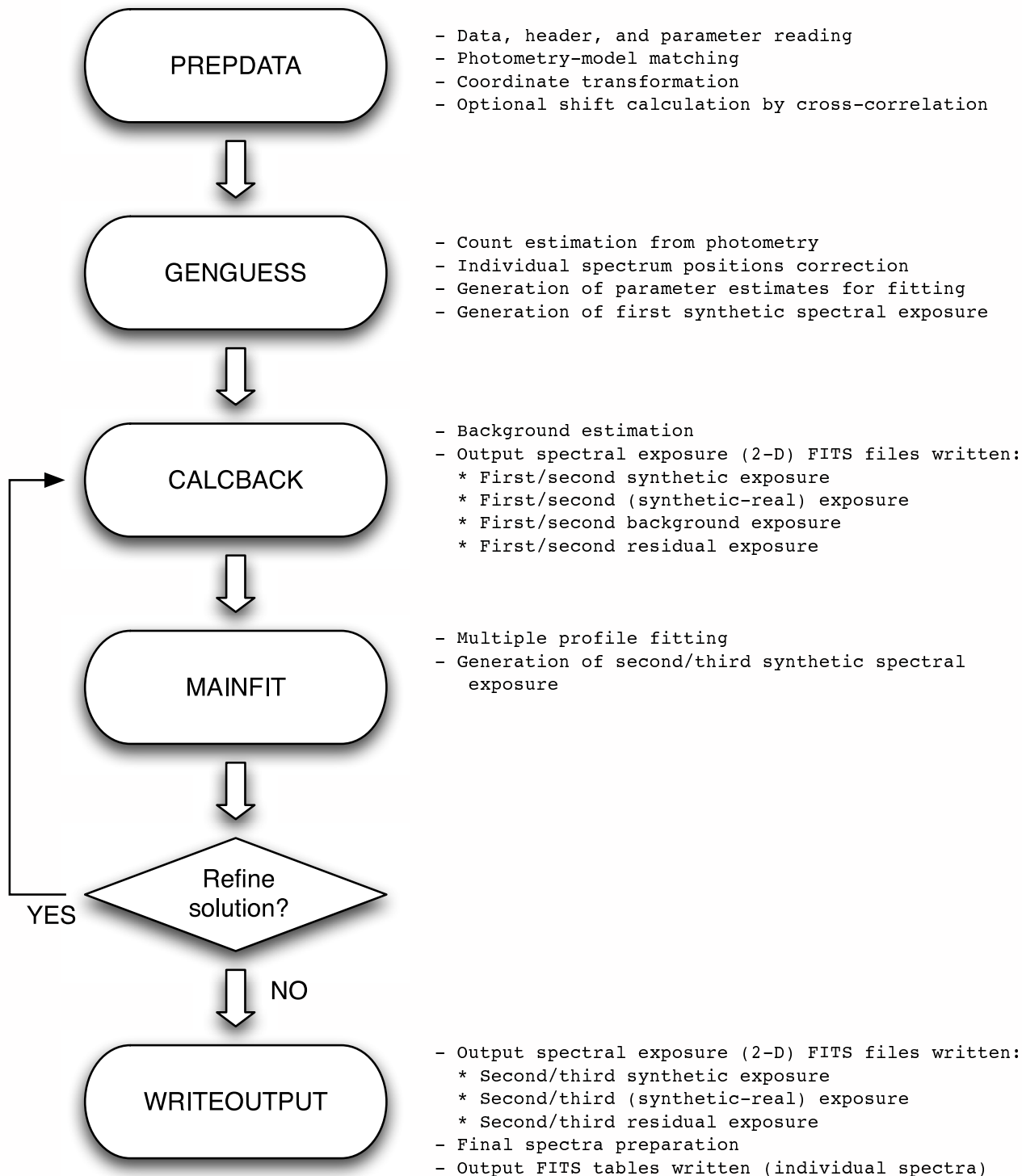


Figure 2: Basic flow diagram and description of the tasks executed by the MULTISPEC modules.

MULTISPEC PARAMETERS AND KEYWORDS

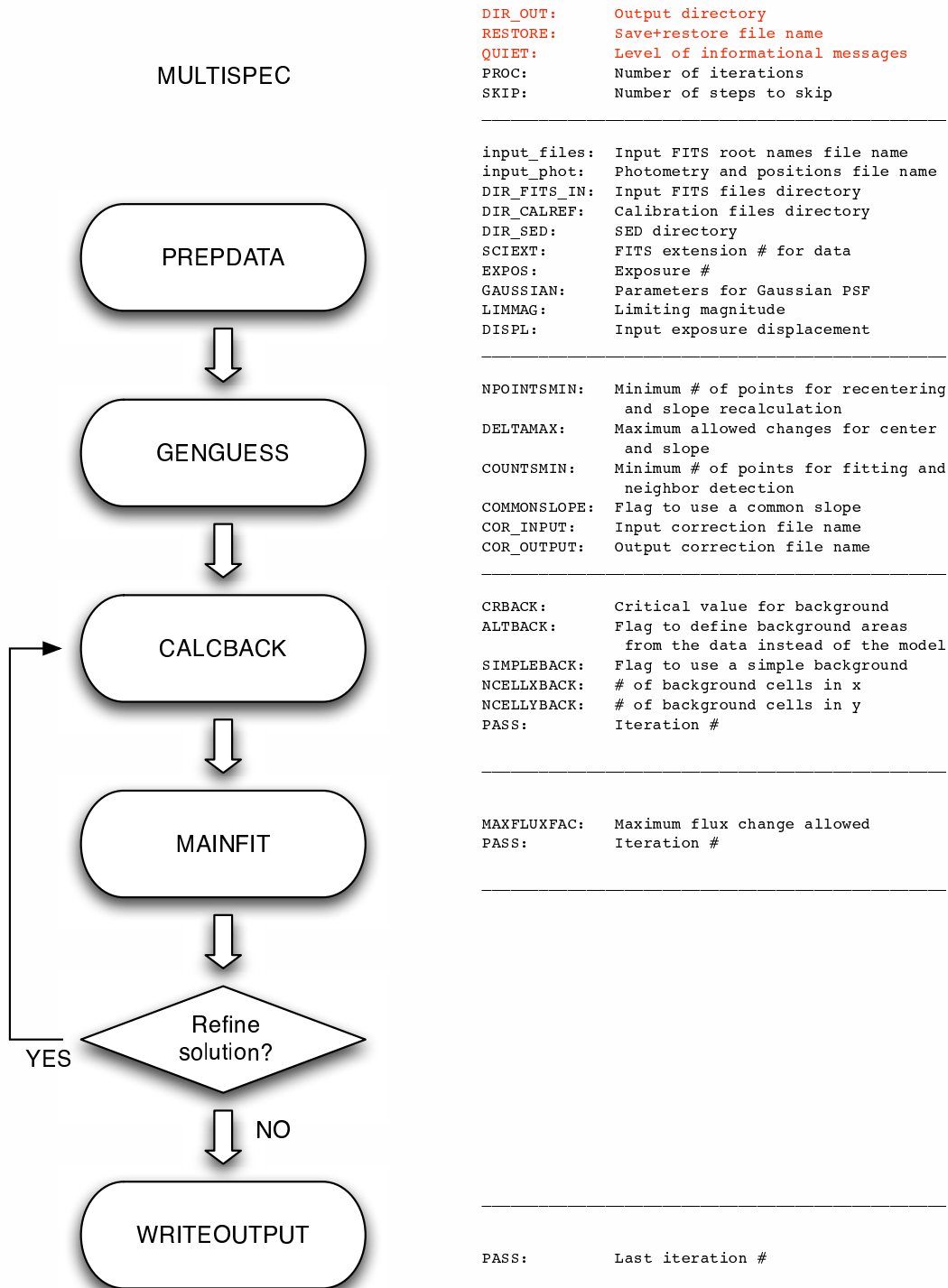


Figure 3: Parameters (lowercase, mandatory) and keywords (uppercase, optional) available for the MULTISPEC wrapper and the individual modules. The three wrapper keywords in red can be used with any module.

rest of the entries for all modules are keyword parameters (or, simply, keywords) and must be specified explicitly. Keywords marked in red can be used in any of the modules. Conversely, any keyword module must be used in the wrapper if one wants it to be passed along.

A typical MULTISPEC execution can be the following:

```
unix shell > idl
IDL Version 6.3, Mac OS X (darwin i386 m32). (c) 2006, Research Systems, Inc.
Installation number: 501126.
Licensed for use by: Astrophysics and Rock'n'Roll Institute of Matalascabras del Valle
IDL> MULTISPECINIT ; Required before any MULTISPEC session!
IDL> MULTISPEC, 'multispec\_files.dat', 'multispec\_phot.dat', EXPOS=1, LIMMAG=22.0, $
DIR\_OUT='multispec\_tabular', DIR\_FITS\_IN='data', NPOINTSMIN=[3,5], $
COUNTSMIN=[1.,200.,200.], /COMMONSLOPE, /ALTBACK, $
NCELLXBACK=8, NCELLYBACK=8, PROC=2
```

The MULTISPEC command is equivalent to sequentially executing the following modules:

```
IDL> PREPDATA, 'multispec_files.dat', 'multispec_phot.dat', EXPOS=1, LIMMAG=22.0, $
DIR_OUT='multispec_tabular', RESTORE=restore, DIR_FITS_IN='data'
IDL> GENGUESS, DIR_OUT='multispec_tabular', RESTORE=restore, $
NPOINTSMIN=[3,5], COUNTSMIN=[1.,200.,200.], /COMMONSLOPE
IDL> CALCBACK, DIR_OUT='multispec_tabular', RESTORE=restore, PASS=1, $
/ALTBACK, NCELLXBACK=8, NCELLYBACK=8
IDL> MAINFIT, DIR_OUT='multispec_tabular', RESTORE=restore, PASS=1
IDL> CALCBACK, DIR_OUT='multispec_tabular', RESTORE=restore, PASS=2, $
/ALTBACK, NCELLXBACK=8, NCELLYBACK=8
IDL> MAINFIT, DIR_OUT='multispec_tabular', RESTORE=restore, PASS=2
IDL> WRITEOUTPUT, DIR_OUT='multispec_tabular', RESTORE=restore, PASS=2
```

The contents of the two files (`multispec_files.dat` and `multispec_phot.dat`) referenced in the positional parameters above are:

`multispec_files.dat:`

```
o8oq51010 o8oq51v4q
o8oq51020 o8oq51v4q
```

`multispec_phot.dat:`

ID	x	y	RA	DEC	sed	e(4405-5495)	stisnuv_f25cn270
01	633.113	108.355	0	0	kurucz_20000_ms_z0.fits	0.100	18.997
02	893.161	133.428	0	0	kurucz_20000_ms_z0.fits	0.100	19.348
03	958.081	152.871	0	0	kurucz_20000_ms_z0.fits	0.100	19.280
04	267.624	198.855	0	0	kurucz_20000_ms_z0.fits	0.100	20.420
05	663.703	250.857	0	0	kurucz_20000_ms_z0.fits	0.100	18.697
06	243.274	291.270	0	0	kurucz_20000_ms_z0.fits	0.100	18.445
07	250.131	293.810	0	0	kurucz_20000_ms_z0.fits	0.100	17.363
08	209.589	400.362	0	0	kurucz_20000_ms_z0.fits	0.100	18.587
09	531.092	510.935	0	0	kurucz_20000_ms_z0.fits	0.100	17.414
10	891.600	647.002	0	0	kurucz_20000_ms_z0.fits	0.100	18.695

11	441.156	751.724	0	0	kurucz_20000_ms_z0.fits	0.100	17.664
12	174.953	838.880	0	0	kurucz_20000_ms_z0.fits	0.100	20.188
13	31.852	875.237	0	0	kurucz_20000_ms_z0.fits	0.100	19.304
14	929.493	928.874	0	0	kurucz_20000_ms_z0.fits	0.100	19.432
15	691.118	866.638	0	0	kurucz_20000_ms_z0.fits	0.100	19.878

Here is a description of each of the wrapper parameters/keywords:

- **DIR_OUT** (default: ' . / '): Directory where output files are written.
- **RESTORE** (default: dataset name): MULTISPEC uses a series of IDL .sav files to transfer data between its different modules and RESTORE is used as the root for the name of the file. By default, it is set by PREPDATA (the first module) to be the HST dataset name of the spectrum (e.g. 'o8oq51010' in the example above). Note that if the wrapper is not used (see the module-by-module example above), then the keyword must be used in order to let each module know which data to process. In such a case, the RESTORE keyword can be used as an output for PREPDATA and as an input for the rest of the modules.
- **QUIET** (default: 0): Level of informational messages. The available options are:
 - -1: All informational messages (a lot!).
 - 0: Normal level.
 - +1: Only warnings and the beginning of each module.
 - +2: You are left completely in the dark about what is going on.
- **PROC** (default: 1): This keyword determines the number of iterations for the calculation of the background and the profile fitting. The available options are:
 - 1: Sequence: initial guess → background calculation → profile fitting.
 - 2: Sequence: initial guess → background calculation → profile fitting → background calculation → profile fitting.

See the PASS keyword in CALCBACK, MAINFIT, and WRITEOUTPUT if you prefer to use module-by-module execution.
- **SKIP** (default: 0): Number of modules to skip before starting the execution. This option is useful to restart the execution from a certain point, possibly changing some keywords. More specifically, the starting points for the possible values are:
 - 0: PREPDATA.
 - 1: GENGUESS.
 - 2: CALCBACK (first pass).
 - 3: MAINFIT (first pass).
 - 4: WRITEOUTPUT if PROC = 1, CALCBACK (second pass) if PROC = 2.
 - 5: MAINFIT (second pass) if PROC = 2.
 - 6: WRITEOUTPUT if PROC = 2.

5.2 PREPDATA

PREPDATA executes the preliminary steps required in MULTISPEC. It reads the different text and FITS files, selects the corresponding reference values from the information in the headers, calculates the best SED model for each star, obtains the transformation between image and spectral spatial coordinates and, optionally, recenters the data. The following parameters/keywords are available with PREPDATA:

- `input_files` (default: `'multispec_files.dat'`): This parameter gives the name of the file that contains the root names of the datasets to be processed. The file should be located in the same directory where MULTISPEC is being executed and can contain either one or two columns. The first column contains the root name for the spectral exposures while the second (optional) one contains the root name for the image in which the photometry has been measured. The purpose of the second column is to rotate and displace the coordinates of the photometry in order to put them in the reference system of the spectral exposure. If the spectral exposure and the image were taken at the same position and orientation (or if only a small dithering shift was applied), the second column is unnecessary. Note that MULTISPEC is designed to work with non-drizzled HST data. Therefore, it will always try to read the spectral exposure files in the following sequence: [a] `rootname + _sciX.cor.fits` (where X refers to the scientific extension, see below), [b] `rootname + _flt.fits`, [c] `rootname + .fits`. For the image files only [b] and [c] are attempted.
- `input_phot` (default: `'multispec_phot.dat'`): This parameter gives the name of the file that contains the data for each star (see previous subsection for an example). The first column should contain an ID number and the second and third the x and y positions of the stars in the image. The fourth and fifth columns should contain the right ascension and declination of the star in degrees (these values are not actually used by MULTISPEC but are simply passed on to the output, so a dummy value of 0 may be used if the user does not know or is not interested in the celestial coordinates of the stars). The sixth column contains the SED to be used as a model for the star, the seventh column the value of $E(4405 - 5495)$ (monochromatic $E(B - V)$) to be applied to the SED to extinguish it (using a Cardelli et al. (1989) law with $R_{5495} = 3.1$), and the eighth column the magnitude for normalization (after extinction has been applied). All the filters available in the standard CHORIZOS distribution can be used to normalize the SED⁴. Several SEDs are available with the MULTISPEC distribution (see below for the file location) and the user can add more creating FITS tables with a wavelength and a flux table (wavelength should be in units of Å and flux should be expressed as f_{λ} in units of $\text{erg s}^{-1} \text{cm}^{-2} \text{Å}^{-1}$).
- `DIR_FITS_IN` (default: `'./'`): Directory where input FITS files are located.
- `DIR_CALREF` (default: `MULTISPEC_DIR + '/calref/' + mode name + '/'`): Directory where reference files are located. The current choices for the mode name are `nuv-mama_prism`, `fuv-mama_g1401`, and `none`.
- `DIR_SED` (default: `MULTISPEC_DIR + '/calref/sed/'`): Directory where SEDs are located.
- `SCIEXT` (default: 1): Extension in the FITS file that contains the data. If the file is of the `_flt.fits` type (see above), PREPDATA will attempt to use the following extension for the data weights. Otherwise, the data weights are built from the data itself.
- `EXPOS` (default: 1): Number of the image/spectrum exposure pair in `input_files` to process.
- `GAUSSIAN` (default: `[0, 0]`): By default, MULTISPEC uses a tabular PSF which for the current version is provided for the supported STIS modes. Activating this keyword uses a Gaussian analytic PSF, which should be done always if the tabular PSF is not present. `GAUSSIAN` should be a 2-element floating-point vector, with the first component giving the width of the PSF in pixels and the second one the angle that the trace forms with the horizontal (in CCW degrees).
- `LIMMAG` (default: `80.0`): Maximum magnitude limit to include in the extraction.
- `DISPL` (default: 0): By default, MULTISPEC applies a cross-correlation between the real spectral exposure and a synthetic spectral exposure (more on that below) to calculate the displacement between the image coordinates and the spectral ones (after accounting for geometric distortions and, possibly, header displacements and rotation). If this keyword is set to a two-element floating-point vector, then the cross-correlation is not calculated and that displacement (in x, y pixels) is applied instead.

PREPDATA requires several calibration files which are described in the following section.

⁴The reader is referred to the CHORIZOS handbook, available from the same website as MULTISPEC (<http://www.iaa.es/~jmaiz>) for more details on synthetic photometry.

5.3 GENGUESS

GENGUESS uses the SED derived for each star by PREPDATA to generate the following initial estimates: [1] the number of counts as a function of wavelength for each star, $C'_k(i) = \sum_j C_k(i, j)$; [2] the corresponding synthetic spectral exposure obtained by placing each star at its corresponding location and adding the result for all the stars convolved by the corresponding PSF profile, $C(i, j)$; and [3] the additional parameters $y_k(i)$ and $\text{discretize}_j[P_\lambda(y - y_k(i))]$ for the multiple-profile fitting at each column. The initial estimates in [1] and [2] will be used to calculate the background and as initial guesses for the multiple-profile-fitting algorithm in the next two modules. From now on we will call the result of [2] the *synthetic exposure* and the slitless spectral exposure the *real exposure*.

GENGUESS can also calculate small corrections to the positions of the individual stars in the spectral exposure, $y_k(i)$, if the geometric distortion is not known to a high degree of precision. Those corrections are in the form of a linear transformation of the type $\Delta y_k(i) = a_k + b_k(i - i_{0,k})$, where $i_{0,k}$ is the horizontal index of the pixel where the center of the spectrum for star k lies, and are calculated from a fit to the position in the real exposure using $C'_k(i)$ for the number of counts. The following keywords are available with GENGUESS:

- **NPOINTS** (default: [5 , 10]): Minimum number of points to have the small position correction calculated for an individual star. The first value corresponds to the minimum number of points required to do a vertical shift ($a_k \neq 0$, $b_k = 0$) and the second one to the equivalent for a full correction ($a_k \neq 0$, $b_k \neq 0$). In order to calculate the correction, GENGUESS first collapses the real exposure into 10-pixel horizontal bins and calculates for each star how many of those bins have a large number of counts (see COUNTSMIN) and are located at least five pixels away (in y) from the nearest bright neighbor, making them good candidates for recentering. The user can decrease the value if the number of stars in the field is low.
- **DELTAMAX** (default: [1.5 , 0.005]): Maximum allowed absolute values for a_k and b_k . In those cases where the fitted values are outside the allowed range, no individual correction is applied.
- **COUNTSMIN** (default: [10 , 200 , 1000]): Minimum number of synthetic counts $C'_k(i)$: [a] in a single column needed for MULTISPEC to extract the spectrum (if non-zero, the final spectrum will be truncated at its low-S/N extremes, note that this is not used by GENGUESS itself but later on by MAINFIT), [b] in a 10-pixel horizontal bin for GENGUESS to calculate a_k and b_k , and [c] in a 10-pixel horizontal bin for GENGUESS to consider a neighbor as bright enough to affect the calculation of a_k and b_k (see NPOINTS).
- **COMMONSLOPE** (default: 0): Flag that, when activated (by adding /COMMONSLOPE or COMMONSLOPE=1 to the command line), forces b_k to be the same for all stars. The chosen value is calculated from a robust mean of b_k and then the individual values of a_k are recalculated subject to that slope value.
- **COR_INPUT** (default: ' '): If this keyword is used, it points to a file (in DIR_OUT) with the values for a_k and b_k . Such a file is written by setting COR_OUTPUT and is useful, for example, when one has a high S/N and a low S/N exposure and one desires to use the first one to determine the values for a_k and b_k for the second one.
- **COR_OUTPUT** (default: ' '): If this keyword is used, it writes the file described in COR_INPUT.

Typically, the user will want to fine-tune the above parameters with different runs in order to select which stellar positions to correct and which ones not. If the geometric distortion correction is accurate, a_k and b_k can be forced to be zero simply by setting NPOINTS to very large values. In the more typical situation when the user will want only some bright isolated stars to have their positions corrected based on their measured traces, the rest of the objects will also be corrected based on the values of a_k and b_k of the bright stars in the same region of the detector.

GENGUESS produces as output the positions of all the individual spectra. The output is in the three forms of: [a] a human-readable text file, [b] a PS file with the positions drawn on top of the real exposure (see Fig. 4), [c] a file that can be read as a region list by SAOimage DS9 in order to overplot the positions on e.g. the FITS file of the real exposure. In the graphical output, red is used for stars where the small position correction has been applied individually and green for those where an average of the nearby stars has been used instead. All of the output files are written in DIR_OUT and are easily distinguished by their suffixes (.dat, .ps, and .ds9 for [a], [b], and [c] above, respectively).

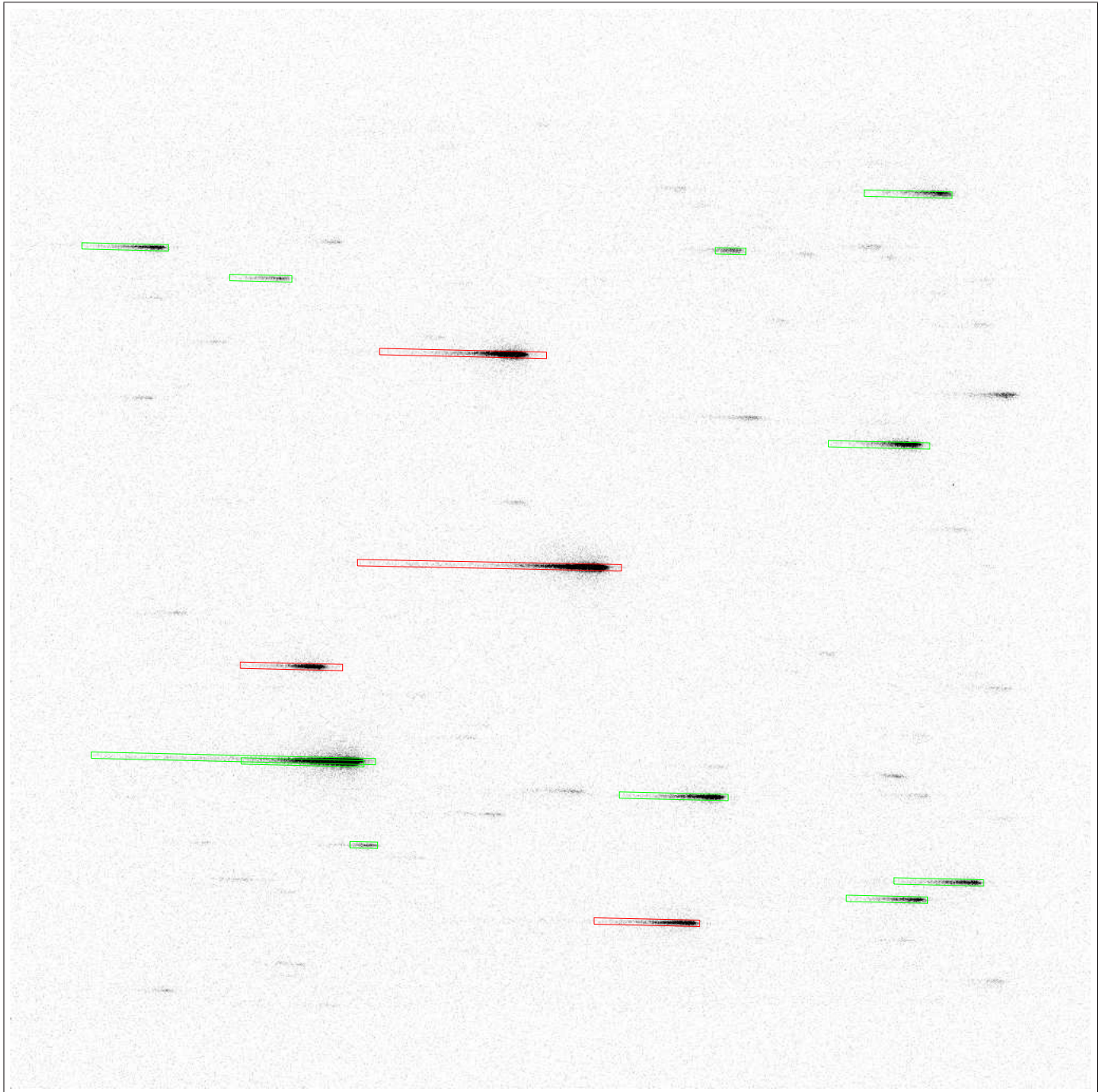


Figure 4: Sample PS file output from GENGUESS. Note that the bright star in the lower left quadrant is a binary, so it appears as a double green rectangle (two stars, no individual recentering) instead of a single red one (one star, recentering). The stars without a rectangle have not been selected for fitting because they are fainter than MLIM. The STIS objective-prism field shown is the same one as in the middle panel of Fig. 1.

5.4 CALCBACK

CALCBACK calculates the background of the slitless spectral exposure. The following keywords are available:

- CRBACK (default: calculated by CALCBACK): Critical value (in counts) used to define the background region. If not set, it is calculated from the image statistics.
- ALTBACK (default: 0): By default, the background region is defined from the synthetic exposure (generated by GENGUESS in the first iteration and by MAINFIT in the second one). If this flag is set (by adding /ALTBACK or ALTBACK=1 to the command line), then the real exposure is used to define the background region.
- SIMPLEBACK (default: 0): By default, CALCBACK calculates a local background, which is useful for the case where the background is a rapidly varying function of position (e.g. when there is nebulosity or a strong contribution from unresolved stars). Setting this flag defines the background simply by interpolating between the central values at the background cells.
- NCELLXBACK (default: 16): Number of background cells in the x direction.
- NCELLYBACK (default: 16): Number of background cells in the y direction.
- PASS (default: 1): Iteration number. The allowed values are 1 and 2.

CALCBACK uses as its first background estimate the subtraction of the synthetic exposure from the real exposure. It then calculates a (broad) second background estimate by a weighted mean of the first estimate in each of the NCELLXBACK \times NCELLYBACK cells, and robustly smoothing the result. If SIMPLEBACK is set, that second estimate is what is used as background. Otherwise, CALCBACK defines the background and source regions, either from the real or the synthetic exposure (depending on whether ALTBACK is set or not), by (a) using CRBACK to provide a first estimate of the source region, (b) expanding it to include the neighboring pixels, and (c) defining the background region as the complementary of the source region. A new background estimate is calculated by using (a) the first estimate in the background region and (b) the second estimate in the source region. The new estimate is then robustly smoothed once more to produce the final background.

As previously mentioned, the background calculation in a slitless spectral exposure is far from trivial. It is expected that the user will have to try different values of the keywords until he/she is satisfied with the results. In order to simplify the process, every time CALCBACK is executed it produces four 2-D FITS files:

- The synthetic exposure (suffix `synt_X.fits`; see Fig. 5, upper right panel).
- The result of subtracting the real exposure from the synthetic one (suffix `synt-real_X.fits`; see Fig. 5, lower left panel, for an example of the inverse of this file).
- The final background (suffix `back_X.fits`; see Fig. 5, lower center panel). This file contains three extensions, which correspond to the background itself, its uncertainty, and a weight map.
- The full residual (suffix `res_X.fits`; see Fig. 5, lower right panel).

The X in the file names refers to the iteration number. Note that the first, second, and fourth of the files above are generated one last time by WRITEOUTPUT. E.g., the command previously used as an example generates three synthetic exposures in the subdirectory `multispec_tabular:080q51010_ext001_synt_1.fits` in the first CALCBACK execution, corresponding to the initial estimate produced by GENGUESS; `080q51010_ext001_synt_2.fits` in the second CALCBACK execution, corresponding to the spectra measured in the first MAINFIT execution; and `080q51010_ext001_synt_3.fits` in the WRITEOUTPUT execution, corresponding to the spectra measured in the second (final) MAINFIT execution. The `ext001` added indicates that the data analyzed correspond to the first FITS extension of `080q51010_flt.fits`.

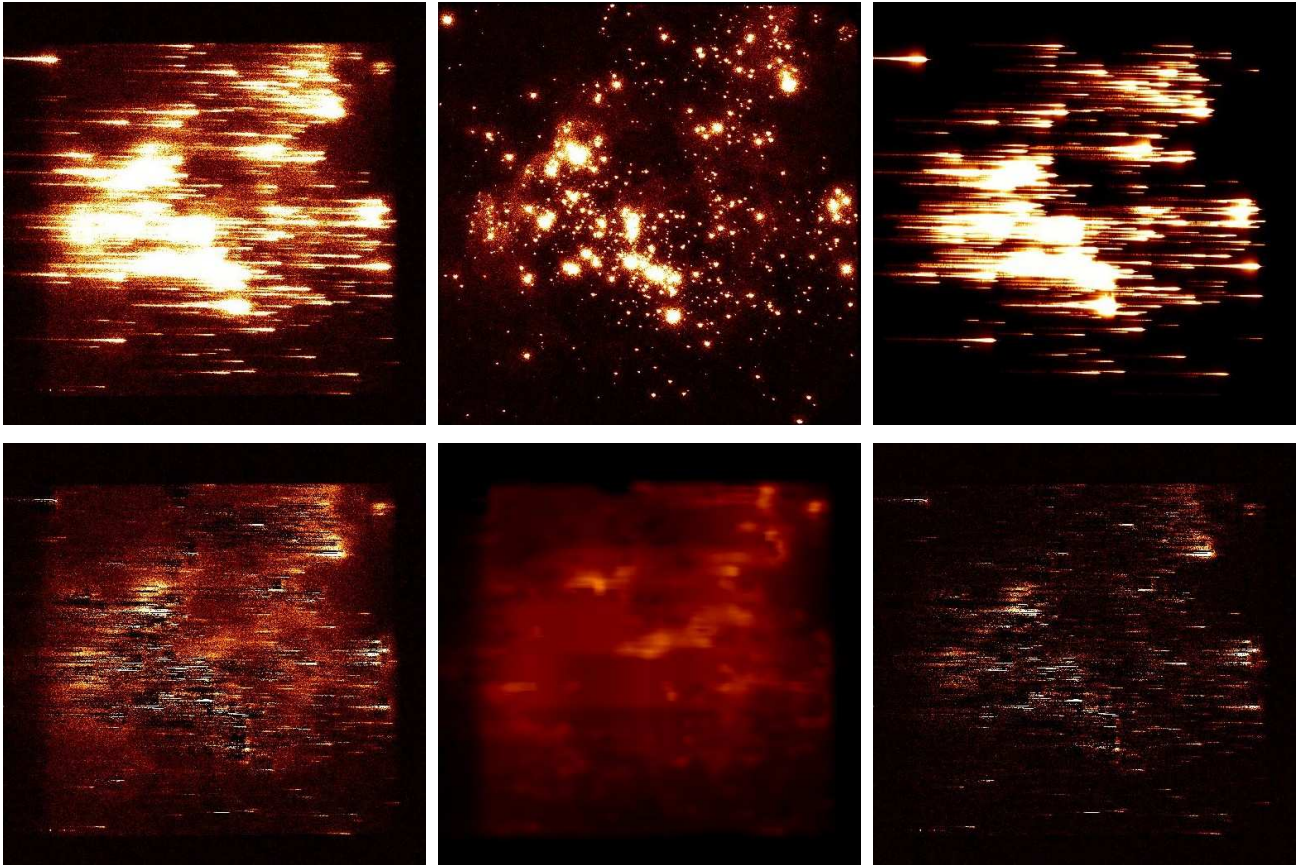


Figure 5: Different 2-D data used or generated by MULTISPEC for the crowded field case shown in Fig. 1. [upper left] Spectral (real) exposure. [upper center] Associated exposure in imaging mode. [upper right] Spectral (synthetic) exposure. [lower left] Partial residual (real minus synthetic exposures). [lower center] Final background. [lower right] Full residual (real exposure minus background and synthetic exposure). All panels use the same color scale and count limits. Note that for the STIS NUV-MAMA the plate scale (in arcseconds/pixel) for an image is 83.4% that of an objective-prism spectral exposure (compare the above left and above center panels).

5.5 MAINFIT

MAINFIT is the core of MULTISPEC, since it is where the spectrum of each star is extracted. This is done by scanning the detector column by column and simultaneously fitting the contribution of each one of the stars present in that column, as described in the previous section. The fitting algorithm uses as initial guess the values calculated in GENGUESS and places two constraints on each of the individual values of $F_k(\lambda_k(i))$: a (negative) minimum based on the read noise of the detector, the background uncertainty, and the PSF width; and a maximum of MAXFLUXFAC times the initial guess. The first one is included to avoid unphysical large negative fluxes (e.g. when a dim spectrum is hidden in the wings of a very bright one or immersed in a bright background). The second one is included to avoid “runaway” fits when two spectra of very different intensity have separations of a small fraction of a pixel. Also, a short and a long wavelength limit are imposed on the fit for every star by calculating the values where the estimate of $C'_k(i) < \text{COUNTSMIN}[0]$ counts (see GENGUESS for the description of the three COUNTMIN components). This restriction is equivalent to the imposition of a minimum S/N for the detection of a star by crowded-field photometry codes.

MAINFIT has two keywords available:

- MAXFLUXFAC (default: 3.0): Maximum multiplicative factor with respect to the initial guess allowed for the measured

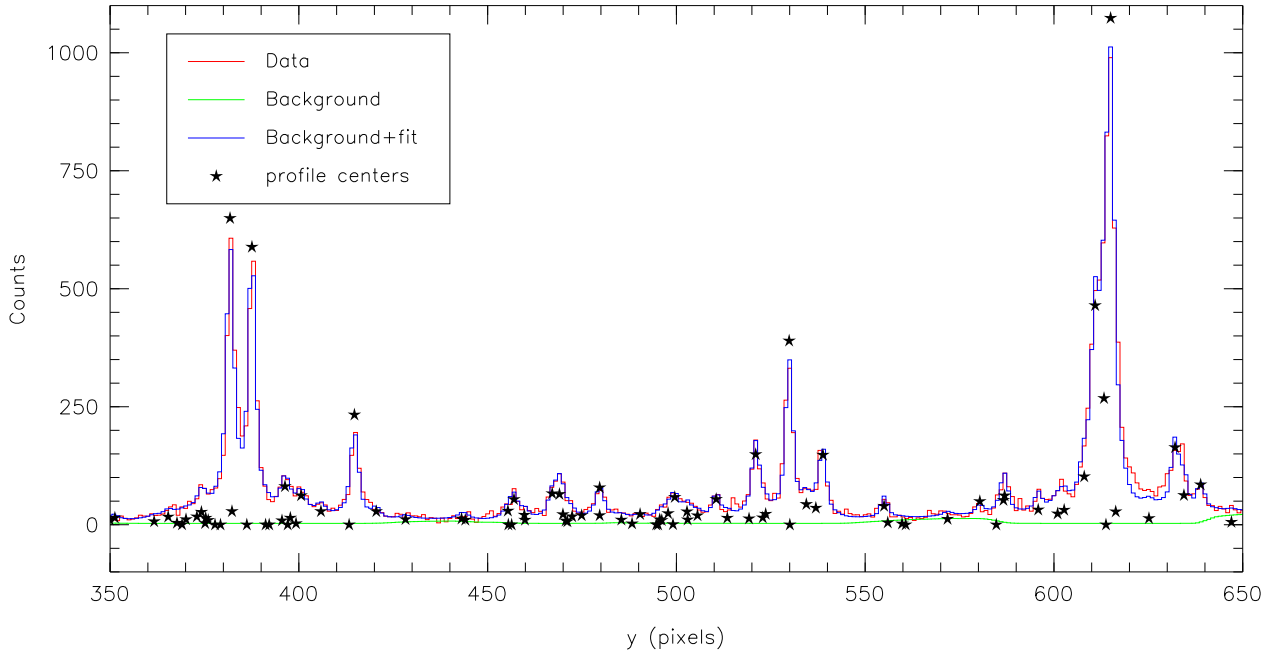


Figure 6: Section of a column with results from MAINFIT.

number of counts.

- PASS (default: 1): Iteration number. The allowed values are 1 and 2.

MAINFIT updates the values generated by GENGUESS in case a second iteration is used and prepares the data for their use by WRITEOUTPUT.

A sample fit for a section of a column is shown in Fig. 6.

5.6 WRITEOUTPUT

WRITEOUTPUT organizes and writes the final data. It generates three 2-D FITS files (with suffixes `synt_X.fits`, `synt-real_X.fits`, and `res_X.fits`, X being 2 or 3 depending on the number of iterations performed, see the CALCBACK subsection) and a FITS table for each of the extracted stars (for the example above, where 15 stars are present, the file names are `o8oq51010_ext001_star0001.fits ... o8oq51010_ext001_star0015.fits`). The columns in the FITS table are:

- `lambda`: Wavelength in \AA .
- `flux`: Spectral flux, $F_k(\lambda)$, in $\text{erg s}^{-1} \text{cm}^{-2} \text{\AA}^{-1}$.
- `sflux`: Uncertainty in the spectral flux in $\text{erg s}^{-1} \text{cm}^{-2} \text{\AA}^{-1}$.
- `cts`: Counts, $C'_k(i)$.
- `scts`: Uncertainty in counts.
- `cts0`: Initial estimate used for the counts.

- res11: Residual counts integrated over 11 pixels centered around $y_k(i)$.
- res61: Residual counts integrated over 61 pixels centered around $y_k(i)$.
- fact: Sensitivity function, $s(\lambda)$, in $(\text{erg s}^{-1} \text{ cm}^{-2} \text{ \AA}^{-1})/(\text{cts s}^{-1} \text{ pixel}_{\lambda}^{-1})$.
- back: Background value in counts.
- backs: Background uncertainty in counts.
- flag: Flag with the possible values of 0 (good fit), 1 (fit at lower edge of possible range), 2 (fit at upper edge of possible range), and 3 (not fitted).
- distany: y distance (in pixels) to the nearest fitted spectrum.
- distm1: y distance (in pixels) to the nearest fitted spectrum with at least $1/10 \times C'_k(i)$.
- distp0: y distance (in pixels) to the nearest fitted spectrum with at least $1 \times C'_k(i)$.
- distp1: y distance (in pixels) to the nearest fitted spectrum with at least $10 \times C'_k(i)$.
- distp2: y distance (in pixels) to the nearest fitted spectrum with at least $100 \times C'_k(i)$.
- chitest: Reduced χ^2 for the seven pixels in the vertical direction centered around y .
- x: x coordinate in IDL style (center of pixel at lower left corner has $x = 0$) for this wavelength.
- y: y coordinate in IDL style (center of pixel at lower left corner has $y = 0$) for the center of the cross-dispersion spatial profile.

WRITEOUTPUT also populates the header of the FITS table with information about the datasets used, the star (coordinates, magnitudes, identification), and the MULTISPEC run. One keyword is available in WRITEOUTPUT:

- PASS (default: 1): Last iteration number used in MAINFIT. The allowed values are 1 and 2.

5.7 Spectra post-processing

The FITS table that WRITEOUTPUT generates for each star is the end product for a single slitless spectral exposure. However, the user may want to improve the result by combining different exposures with the same orientation to increase the S/N ratio or from different orientations in order to eliminate superpositions between stars. Indeed, the reason why WRITEOUTPUT includes columns such as back, flag, distany, distm1, distp0, distp1, distp2, and chitest is that the user may use them as criteria for such combinations. The choice of criterion (or combination thereof) to combine spectra has only been partially explored with the available data at the current time: more work is planned for the future. It is also possible to use the res11 and res61 columns to check for possible residuals due to e.g. an incorrect assumption of the cross-dispersion spatial profile.

6 Using MULTISPEC: how to work with the calibration files

MULTISPEC was originally written for with the HST/STIS NUV-MAMA objective prism and it was first used by Maíz Apellániz and Bohlin (2005) to calibrate that STIS mode using spectral exposures of the standard star HS 2027+6051. An example of those exposures is shown in the left panel of Fig. 1. The current version can be applied to other modes but the STIS heritage is still reflected in the choice of calibration (or reference) files needed: MULTISPEC uses [a] four files with the same format as the STIS ones and [b] three files with a different format if they have not been produced that way. MULTISPEC processes the calibration files in a procedure called READTABS which is included as an appendix in this document for reference.

If `DIR_CALREF` is not specified, `READTABS` looks for the calibration files in `MULTISPEC_DIR + /calref/ + mode name`, where *mode name* is built by `MULTISPEC` by searching in the input FITS file for the keywords `DETECTOR` and `OPT_ELEM` (if the latter is not found, it alternatively searches for `GRATING`). *mode name* is then built as the concatenation of the two strings joined with `_`, stripped of leading or trailing blank spaces, and transformed into lowercase characters. For example, a STIS NUV-MAMA objective prism has the following (partial) header:

```
[...]
SUBARRAY=                F / data from a subarray (T) or full frame (F)
DETECTOR= 'NUV-MAMA      ' / detector in use: NUV-MAMA, FUV-MAMA, or CCD
OPT_ELEM= 'PRISM        ' / optical element in use
APERTURE= 'F25SRF2      ' / aperture name
[...]
```

leads to the calibration files for that mode being located in `MULTISPEC_DIR + /calref/nuv-mama_prism/`. If the keywords are not found, `MULTISPEC` will use `MULTISPEC_DIR + /calref/none/` instead. The code will also look for several other keywords in the input FITS file: `APERTURE`, `FILTER`, `CENWAVE`, `NAXIS`, `NAXIS1`, `NAXIS2`, `APERTAB`, `PHOTTAB`, `TDSTAB`, and `DISPTAB`. The last four keywords contain the names of the four STIS-like calibration files:

- `APERTAB`: Relative aperture throughput table.
- `PHOTTAB`: Photometric throughput table.
- `TDSTAB`: Time-dependent sensitivity algorithm used
- `DISPTAB`: Dispersion coefficient table.

Refer to http://www.stsci.edu/hst/stis/calibration/reference_files for more information on those STIS reference files and to the appendix here to see how `READTABS` extracts the information from them.

The three non-STIS reference files are:

- `disptab.dat`: An ASCII file with two or three lines. The first one indicates the dispersion type, the second one the telescope area in cm^2 , and the optional third one whether one desires the dispersion solution to be expanded at its long-wavelength end. See the appendix and the provided examples in the distribution for more details.
- `geomtab.dat`: An ASCII file with information relevant to the geometric distortion. In particular it should include four columns corresponding to the geometric distortion coefficients (c_x , c_y , d_x , d_y) for the transformation between the image and the slitless spectral exposure. The coefficients for the NUV-MAMA objective prism were defined in Maíz Apellániz and Úbeda (2004). A third, fourth, fifth, or sixth order polynomial must be used and the coefficients listed in the order: (0,1), (1,1), (0,2), (1,2), (2,2) ... The normalization is such that $c_{x,1,1} \approx c_{y,0,1} \approx d_{x,1,1} \approx d_{y,0,1} \approx 1.0$ and $c_{x,0,1} \approx c_{y,1,1} \approx d_{x,0,1} \approx d_{y,1,1} \approx 0.0$.
- An optional PSF FITS file that must be named according to the optical element or grating used with `_psf.fits` appended (e.g. `prism_psf.fits` or `g140l_psf.fits`). The file is not needed if one uses analytical (Gaussian) PSFs. Future versions of `MULTISPEC` will include the option of generating PSFs for other modes.

I would like to thank Linda Dressel and Paul Goudfrooij for their comments reviewing the original version of this document and Knox S. Long for his help with the coding for the G140L mode.

Bibliography

- Anderson, J. and King, I. R. 2000, *PASP* **112**, 1360
- Cardelli, J. A., Clayton, G. C., and Mathis, J. S. 1989, *ApJ* **345**, 245
- Maíz Apellániz, J. 2004, *PASP* **116**, 859
- Maíz Apellániz, J. 2005, *STIS Instrument Science Report 2005-02 (STScI: Baltimore)*
- Maíz Apellániz, J. and Bohlin, R. C. 2005, *STIS Instrument Science Report 2005-01 (STScI: Baltimore)*
- Maíz Apellániz, J. and Úbeda, L. 2004, *STIS Instrument Science Report 2004-01 (STScI: Baltimore)*
- Porter, J. M., Oudmaijer, R. D., and Baines, D. 2004, *A&A* **428**, 327

Appendix: The READTABS procedure

```

PRO READTABS, hdrst,                $ ; Input
      DIR_CALREF=dir_calref, $ ; Input/Output
      GAUSSIAN=gaussian,        $ ; Input
      QUIET=quiet                ; Flag
COMMON params, cx, cy, dx, dy, xref, yref, xscale, yscale, isscale, ipscale, spscale
COMMON params2, detector, grating, filter, cenwave, nx, ny, ixref, iyref, $ ; Output (general values)
      lambdart, rthroughput, lambdapt, pthroughput,                $ ; Output (throughput vectors)
      lambdatds, mjdtds, facttds,                                $ ; Output (throughput vectors)
      disptype, area, a2cent_disp, a_disp, righttext,            $ ; Output (dispersion information)
      ref_files                                                  ; Output (reference file names)
COMMON psf,      psf, sampl, sizepsf, sizepsf2, irelpsf, psf0, psf00, angle0
;
; Detector/grating identification and calibration directory assignment
;
detector      = STRTRIM(SXPAR(hdrst,'DETECTOR'),2)
grating1      = STRTRIM(SXPAR(hdrst,'OPT_ELEM'),2)
grating2      = STRTRIM(SXPAR(hdrst,'GRATING' ),2)
IF grating1 NE '0' THEN grating = grating1 ELSE grating = grating2
IF detector EQ '0' OR grating EQ '0' THEN BEGIN
  detector    = 'NONE'
  grating     = 'NONE'
  IF quiet LT 2 THEN PRINT, 'WARNING:          NONE used as detector/grating'
ENDIF
COMMONMULTISPEC, '', '',                $ ; Variables
      DIR_CALREF=dir_calref,            $ ; Directories
      DETECTOR=detector, GRATING=grating
;
; We extract additional parameters from the header
;
filter1       = STRTRIM(SXPAR(hdrst,'APERTURE'),2)
filter2       = STRTRIM(SXPAR(hdrst,'FILTER' ),2)
IF filter1 NE '0' THEN filter = filter1 ELSE filter = filter2
IF filter EQ '0' THEN filter = 'NONE'
cenwave       = STRTRIM(SXPAR(hdrst,'CENWAVE' ),2)
naxes         =          SXPAR(hdrst,'NAXIS' )
nx            =          SXPAR(hdrst,'NAXIS1' )
ny            =          SXPAR(hdrst,'NAXIS2' )
IF naxes NE 2 OR nx EQ 0 OR ny EQ 0 THEN STOP, 'FITS data is not 2-D'
;
; APERTAB: relative aperture throughput table
;
apertab0      = SXPAR(hdrst,'APERTAB')
IF apertab0 NE '0' THEN $
  IF STRMID(apertab0 ,0,5) EQ 'oref$' THEN apertab = STRMID(apertab0,5,999) $
                                          ELSE apertab = apertab0                $
ELSE
  apertab     = 'apertab.fits'

```

```

vapertab      = MRDFITS(dir_calref + apertab,1,hdrapertab,STATUS=status,/SILENT)
IF status LT 0 THEN STOP, 'Error reading APERTAB file: ' + dir_calref + apertab
filtlist     = STRTRIM(vapertab.aperture,2)
pos          = WHERE(filtlist EQ filter,npos)
IF npos NE 1 THEN BEGIN
  pos        = WHERE(filtlist EQ 'ANY',npos)
  IF npos NE 1 THEN STOP, 'APERTAB error'
ENDIF
npoints      = vapertab[pos].nelem
lambdart     = vapertab[pos].wavelength
rthroughput  = vapertab[pos].throughput
lambdart     = lambdart[0:npoints-1]
rthroughput  = rthroughput[0:npoints-1]
IF rthroughput[0] NE 0.0 THEN BEGIN
  bleft      = (rthroughput[1]-rthroughput[0])/(lambdart[1]-lambdart[0])
  aleft      = rthroughput[0]-lambdart[0]*bleft
  IF aleft GE 0.0 THEN BEGIN
    lambdart  = [ 0.0, lambdart]
    rthroughput = [aleft,rthroughput]
    npoints   = npoints + 1
  ENDIF ELSE BEGIN
    lambdart  = [0.0,-aleft/bleft, lambdart]
    rthroughput = [0.0, 0.0,rthroughput]
    npoints   = npoints + 2
  ENDELSE
ENDIF ELSE BEGIN
  lambdart    = [0.0, lambdart]
  rthroughput = [0.0,rthroughput]
  npoints     = npoints + 1
ENDELSE
IF rthroughput[npoints -1] NE 0.0 THEN BEGIN
  bleft      = (rthroughput[npoints -1]-rthroughput[npoints -2])/(lambdart[npoints -1]-lambdart[npoints -2])
  aleft      = rthroughput[npoints -1]-lambdart[npoints -1]*bleft
  IF bleft GE 0.0 THEN BEGIN
    lambdart  = [lambdart , 1.e06]
    rthroughput = [rthroughput,rthroughput[npoints -1]]
    npoints   = npoints + 1
  ENDIF ELSE BEGIN
    lambdart  = [ lambdart,-aleft/bleft,1.e06]
    rthroughput = [rthroughput, 0.0, 0.0]
    npoints   = npoints + 2
  ENDELSE
ENDIF ELSE BEGIN
  lambdart    = [ lambdart,1.e06]
  rthroughput = [rthroughput, 0.0]
  npoints     = npoints + 1
ENDELSE
;
; PHOTTAB: photometric throughput table
;
phottab0     = SXP(hdrst,'PHOTTAB')
IF phottab0 NE '0' THEN $
  IF STRMID(phottab0 ,0,5) EQ 'oref$' THEN phottab = STRMID(phottab0,5,999) $
  ELSE phottab = phottab0 $
  ELSE
    phottab = 'phottab.fits'
vphottab     = MRDFITS(dir_calref + phottab,1,hdrphottab,STATUS=status,/SILENT)
IF status LT 0 THEN STOP, 'Error reading PHOTTAB file: ' + dir_calref + phottab
gratlist     = STRTRIM(vphottab.opt_elem,2)
cenwlist     = STRTRIM(vphottab.cenwave ,2)
pos          = WHERE(gratlist EQ grating AND cenwlist EQ cenwave,npos)
IF npos NE 1 THEN BEGIN
  pos        = WHERE(gratlist EQ 'ANY' AND cenwlist EQ cenwave,npos)
  IF npos NE 1 THEN STOP, 'PHOTTAB error'
ENDIF
npoints2     = vphottab[pos].nelem

```

```

lambdapt      = vphottab[pos].wavelength
pthroughput  = vphottab[pos].throughput
lambdapt      =   lambdapt[0:npoints2-1]
pthroughput  = pthroughput[0:npoints2-1]
IF pthroughput[0] NE 0.0 THEN BEGIN
  bleft = (pthroughput[1]-pthroughput[0])/(lambdapt[1]-lambdapt[0])
  aleft = pthroughput[0]-lambdapt[0]*bleft
  IF aleft GE 0.0 THEN BEGIN
    lambdapt      = [ 0.0,   lambdapt]
    pthroughput  = [aleft, pthroughput]
    npoints2     = npoints2 + 1
  ENDIF ELSE BEGIN
    lambdapt      = [0.0, -aleft/bleft,   lambdapt]
    pthroughput  = [0.0,           0.0, pthroughput]
    npoints2     = npoints2 + 2
  ENDELSE
ENDIF ELSE BEGIN
  lambdapt      = [0.0,   lambdapt]
  pthroughput  = [0.0, pthroughput]
  npoints2     = npoints2 + 1
ENDELSE
IF pthroughput[npoints2-1] NE 0.0 THEN BEGIN
  bleft = (pthroughput[npoints2-1]-pthroughput[npoints2-2])/(lambdapt[npoints2-1]-lambdapt[npoints2-2])
  aleft = pthroughput[npoints2-1]-lambdapt[npoints2-1]*bleft
  IF bleft GE 0.0 THEN BEGIN
    lambdapt      = [lambdapt      ,           1.e06]
    pthroughput  = [pthroughput, pthroughput[npoints2-1]]
    npoints2     = npoints2 + 1
  ENDIF ELSE BEGIN
    lambdapt      = [   lambdapt, -aleft/bleft, 1.e06]
    pthroughput  = [pthroughput,           0.0, 0.0]
    npoints2     = npoints2 + 2
  ENDELSE
ENDIF ELSE BEGIN
  lambdapt      = [   lambdapt, 1.e06]
  pthroughput  = [pthroughput, 0.0]
  npoints2     = npoints2 + 1
ENDELSE
;
; TDSTAB: Time-dependent sensitivity algorithm used
;
tdstab0       = SXPARG(hdrst, 'TDSTAB')
IF tdstab0 NE '0' THEN $
  IF STRMID(tdstab0 ,0,5) EQ 'oref$' THEN tdstab = STRMID(tdstab0 ,5,999) $
                                     ELSE tdstab = tdstab0 $
  ELSE
    tdstab = 'tdstab.fits'
vdstab       = MRDFITS(dir_calref + tdstab ,1,hdrtdstab ,STATUS=status,/SILENT)
IF status LT 0 THEN STOP, 'Error reading TDSTAB file: ' + dir_calref + tdstab
gratlist     = STRTRIM(vdstab.opt_elem ,2)
pos          = WHERE(gratlist EQ grating, npos)
IF npos NE 1 THEN BEGIN
  pos        = WHERE(gratlist EQ 'ANY' , npos)
  IF npos NE 1 THEN STOP, 'TDSTAB error'
ENDIF
mjdtds0     = vdstab[pos].time
lambdatds0  = vdstab[pos].wavelength
slopetds0   = vdstab[pos].slope
pos         = WHERE(mjdtds0 NE 0.0, nmjtdts0)
IF nmjtdts0 LT 2 THEN STOP, 'TDSTAB error'
mjdtds      = [0.0, mjdtds0[pos], 1.e5]
pos2        = WHERE(lambdatds0 NE 0.0, nlambdatds0)
lambdatds   = [0.0, lambdatds0[pos2], 1.e10]
slopetds    = FLTARR(nlambdatds0+2, nmjtdts0+1)
posm        = (1+INTARR(nlambdatds0)) # pos
pos2m       = pos2 # (1+INTARR( nmjtdts0))

```

```

slopetds[1:nlambdatds0,1:nmjdtlds0] = slopetds0[[pos2m],[posm]]
slopetds[          0,1:nmjdtlds0] = slopetds[          1,1:nmjdtlds0]
slopetds[nlambdatds0+1,1:nmjdtlds0] = slopetds[nlambdatds0,1:nmjdtlds0]
facttlds          = FLTARR(nlambdatds0+2,nmjdtlds0+2)
facttlds[* ,0] = 1.0
FOR i=1,nmjdtlds0+1 DO facttlds[* ,i] = facttlds[* ,i-1] + (slopetds[* ,i-1]/365.25/100.)*(mjdtlds[i]-mjdtlds[i-1])
;
; DISPTAB2: dispersion type and telescope area
;
disptab2          = 'disptab.dat'
READDATA, dir_calref + disptab2, disptype0, /ONESTRING, ERR=err, /QUIET
IF err NE 0 THEN STOP, 'Error reading DISPTAB dat file: ' + dir_calref + disptab2
STRSUB, disptype0[0], disptype, ' ', ''
IF disptype NE 'lambda=a_1+a_2/(dx-a_0)+a_3/(dx-a_0)^2+a_4/(dx-a_0)^3+a_5/(dx-a_0)^4' AND $
    disptype NE 'dx=a_1+a_2*lambda+a_3*lambda^2' AND $
    disptype NE 'dx=lambda' THEN STOP, 'Unrecognized dispersion type'
area              = DOUBLE(disptype0[1])
righttext         = 0
IF N_ELEMENTS(disptype0) GT 2 THEN BEGIN
    IF STRTRIM(disptype0[2],2) EQ '1' THEN righttext = 1 ; Extension to longer wavelengths
ENDIF
;
; DISPTAB: dispersion coefficient table
;
;
; PRISM:
;
;
; lambda = a_1 + ----- + ----- + ----- + -----
;              (dx-a_0)      (dx-a_0)^2   (dx-a_0)^3   (dx-a_0)^4
;
; GRATING:
;
; dx = a_1 + a_2*lambda + a_3*lambda^2
;
; NONE:
;
; dx = lambda
;
IF disptype NE 'dx=lambda' THEN BEGIN
    disptab0       = SXPARR(hdrst,'DISPTAB')
    IF disptab0 NE '0' THEN $
        IF STRMID(disptab0 ,0,5) EQ 'oref$' THEN disptab = STRMID(disptab0,5,999) $
        ELSE disptab = disptab0 $
        ELSE
            disptab = 'disptab.fits'
    vdisptab       = MRDFITS(dir_calref + disptab,1,hdrdisptab,STATUS=status,/SILENT)
    IF status LT 0 THEN STOP, 'Error reading DISPTAB fits file: ' + dir_calref + disptab
    gratlist       = STRTRIM(vdisptab.opt_elem,2)
    cenwlist       = STRTRIM(vdisptab.cenwave,2)
    pos            = WHERE(gratlist EQ grating AND cenwlist EQ cenwave,npos)
    IF npos EQ 0 THEN BEGIN
        pos        = WHERE(gratlist EQ 'ANY' AND cenwlist EQ cenwave,npos)
        IF npos EQ 0 THEN STOP, 'DISPTAB error'
    ENDIF
    apert          = STRTRIM(vdisptab[pos[0]].ref_aper,2)
    npoints        =          vdisptab[pos[0]].ncoeff
    a2cent_disp    =          vdisptab[pos].a2center
    a              = DOUBLE( vdisptab[pos].coeff)
    a_disp         =          a[0:npoints-1,*]
ENDIF ELSE BEGIN
    disptab        = 'NONE'
ENDELSE
;
; GEOMTAB: image-to-spectral exp. geometric distortion table
;

```

```

geomtab0      = SXP(hdrst,'GEOMTAB')
IF geomtab0 NE '0' THEN $
  IF STRMID(geomtab0 ,0,5) EQ 'oref$' THEN geomtab = STRMID(geomtab0,5,999) $
                                     ELSE geomtab = geomtab0 $
  ELSE
    geomtab = 'geomtab.dat'
READDATA, dir_calref + geomtab, cx, cy, dx, dy, ERR=err, /QUIET
IF err NE 0 THEN STOP, 'Error reading GEOMTAB file: ' + dir_calref + geomtab
ncx           = N_ELEMENTS(cx)
IF ncx NE 11 AND ncx NE 16 AND ncx NE 22 AND ncx NE 29 THEN $
  STOP, 'Incorrect GEOMTAB file: ' + geomtab
xref          = cx[0]
yref          = cy[0]
ixref        = xref ; Originally, xref, yref referred to the geometric distortion solution and
iyref        = yref ; ixref, iyref to the dispersion solution: now they are the same
xscale       = dx[0]
yscale       = dy[0]
isscale      = cx[1]
ipscale      = cy[1]
spscale      = ipscale/isscale
cx           = cx[2:ncx-1]
cy           = cy[2:ncx-1]
dx           = dx[2:ncx-1]
dy           = dy[2:ncx-1]
;
; PSFTAB: PSF data
;
IF KEYWORD_SET(GAUSSIAN) EQ 0 THEN gaussian = [0., 0.]
IF N_ELEMENTS(GAUSSIAN) NE 2 THEN STOP, 'Incorrect format for GAUSSIAN'
IF TOTAL(ABS(gaussian)) EQ 0 THEN BEGIN
  psftab      = STRLOWCASE(grating) + '_psf.fits'
  psf0        = MRDFITS(dir_calref + psftab,0,hdrpsf,STATUS=status1,/SILENT)
  psf00       = MRDFITS(dir_calref + psftab,2,          STATUS=status2,/SILENT)
  IF status1 LT 0 OR status2 LT 0 THEN STOP, 'Error reading PSF fits file: ' + dir_calref + psftab
  angle0      = SXP(hdrpsf,'ANGLE0')
  sampl       = SXP(hdrpsf,'SAMPL')
  sizepsf     = SXP(hdrpsf,'SIZEPSF')
  sizepsf2    = sampl*sizepsf
ENDIF ELSE BEGIN
  sigma       = gaussian[0]
  IF sigma GT 10.0 THEN STOP, 'Gaussian sigma too large: ' + STRTRIM(sigma,2)
  psftab      = 'Gaussian with sigma = ' + STRTRIM(sigma,2)
  angle0      = gaussian[1]
  sampl       = 5
  sizepsf     = 101
  sizepsf2    = sampl*sizepsf
  yy          = (FINDGEN(sizepsf2)-0.5*(sizepsf2-1))/sampl
  psf0        = EXP(-0.5*(yy/sigma)^2)
  FOR i=0,sampl-1 DO psf0[sampl*INDGEN(sizepsf)+i] = $
    psf0[sampl*INDGEN(sizepsf)+i]/TOTAL(psf0[sampl*INDGEN(sizepsf)+i])
  psf0        = TRANSPOSE([[psf0],[psf0]])
  psf00       = {lambda:[1.,1.e9]}
ENDELSE
;
; We collect all file names
;
ref_files     = [apertab,phottab,tdstab,disptab,disptab2,geomtab,psftab]
END

```