8 Attachments

The following are CXC Data Systems help pages. They are maintained by CXCDS and will be updated as required.

1. `specdetect.hlp`
2. `specidentify.hlp`
3. `specmeasure.hlp`
4. `tg_create_mask.hlp`
5. `tg_resolve_events.hlp`
6. `tgdetect.hlp`
7. `tgextract.hlp`

   The following is an unmaintained document. It gives a graphical presentation of the data flow and gives graphical presentation of simulations at different stages of Level 1.5 and Level 2 processing.

8. "AXAF Dispersive Spectroscopy Pipeline Overview", David Huenemoerder, (HEAD-CC-3, 11 December 1998)

9 specdetect.hlp

```
NAME

    specdetect -- Detect 1D features in a grating spectrum(a).

USAGE

    specdetect infile outfile thresh_emission
    [flux_corrected] [input_continuum_file] [thresh_absoption]
    [continuum_method] [median_n_bins] [continuum_offset]
    [head_tail_cont_method] [head_value] [tail_value]
    [low_tmedian] [high_tmedian] [clobber] [verbose]

DESCRIPTION

    This program finds local maxima in one-dimensional grating data.
    A peak is defined as any bin where the counts are higher than at
    both neighboring bins.  Detection of absorption features and
    edges are not currently implemented.

    This program runs on input Type I or Type II PHA grating data.
    If Type I is used, an @file can hold a list of these files;
    Type II's should be done one at a time (all the spectra in
    a Type II file are processes in one execution of specdetect).

    If the input file is Type II or if there is a list of input
    Type I files, the output will broken up into extension.  Each
    extension contains peak results for one input spectrum.  The
    input TG_SRCID, TG_PART, TG_M and SPEC_NUM are maintained in
    the output table as keywords, in order to help trace the
    order of the results back to the intput data.

    If the input is one Type I file, then there is only one
    output extension containing peak information (since there
    is only one input spectrum).

    The user has the option to input their own continuum file(s)
    for the data.  Continuum files are in ascii format (see below for
    file description).  The number of the input continuum file(s)
    must match the number of spectra in the input file(s) (ie: the
    number of rows in infile must be the number of files in the
    @continuum_file_list).  If there is only one spectrum in
    the infile (one row or one Type I file), then the continuum
    file can be specified as the file name itself.

    If no continuum is supplied by the user, the program uses one
    of three methods to estimate the continuum
    level in the input spectrum.  The program then searches for
    maxima which exceed the input noise threshold above the
    continuum.  Detected maxima are subtracted from the
    spectrum, and the continuum calculation is repeated
    to refine the continuum estimate.  Once the continuum level is
    determined, the scan for maxima is repeated
    to get the final list of detected features and approximate
    widths and amplitudes for use as initial guesses in subsequent
```

```
    line profile fitting.

    The output file from this tool is root_peaks2.fits.  This file
    contains a LINELIST extension(s) which contains the list of
    features found with the initial guesses for the fitting
    routines per spectrum; the SPECTRUM extenstion(s) contains the
    continuum spectra for each input spectrum.  If the user had
    specified their own input continuum spectrum/a to be used w/
    the processing of the input dataset, then, this same spectrum/a
    is written to the output SPECTRUM extension(s).  For every
    input spectrum, there are two extensions written to the
    output file:  LINELISTi and SPECTRUMi, where 'i' increments
    by one, following number of input spectra.

    The output file may be used as input to specmeasure.

    Specdetect works on unit units of: angstrom, mm, degrees,
    pixel, keV and eV.  All these units are available in
    tgextract and the rest of the spec tools (specmeasure
    and specidentify).

PARAMETERS

    infile [filename]  (filename | root_pha2.fits | @file)

        Input FITS grating spectrum(a) data file (Type I or Type II
        pha file).  This file can be generated by the tool tgextract.
        An @list of Type I files may also be used as input.

    outfile = "." [filename] (root_peaks2.fits)

Output file name for peaks and continuum.

"." creates an output file name based on the input data file
name, e.g., if infile = "root_pha2.fits", then outfile =
"root_peaks2.fits".

    thresh_emission = 4.0 [real]

Minimum emission peak amplitude threshold.  The
        emission line detect threshold = (thresh_emission *
        uncertainty).  The threshold value is
        not from the minimum scale of 0 counts; the threshold is
        set from the continuum, as the minimum point.

    flux_corrected = no  [boolean]

        Is the input file flux corrected?  This parameter is currently
        not being used by the tool processing.

    (input_continuum_file = NONE) [@list_filename, filename or NONE]
                                  (root_cont.dat)

        Optional input continuum file name. The
        file must have all 4 columns:  col_1=position, col_2=bin width,
```

```
        col_3=counts or flux, col_4=counts_err or flux_err.  The number
        of bins must be the same as the input spectrum(a) data bins.
        If a file is specified, then the continuum will not be
        calculated internally.
        @list_filename syntax will be used if mutiple input continuum files
        are used. This means to open the ascii file called "list_filename"
        and look inside for other file names which contain the continuum
        information.  There must be a one-to-one match in the continuum
        file list to the row numbers of the infile Type II spectra or
        @list of Type I files.

        Ex: input_continuum_file=@continuum_file_list

        > cat continuun_file_list
          file1.dat
          file2.dat
          file3.dat

          Input PHA Type II file contains 3 rows (spectra) for
          processing; or, there were three Type I PHA files
          entered in an input @list.

    (thresh_absoption = 0.0) [real]

Minimum absorption peak amplitude.  The absorption line
        detect threshold = (thresh_absorption * uncertainty).
        This parameter is currently not being used by the tool,
        since there is no algorithm in place for the absorption
        line detection.

    (continuum_method = "median") [spline3|median|tmedian]

Find the continuum using any of these methods: smoothing
        'spline3', running 'median', truncated running 'tmedian'.
        Descriptions of all three are listed below.

        1) The smoothing spline (spline3) uses:

        ALGORITHM 642 COLLECTED ALGORITHMS FROM ACM.
        ALGORITHM APPEARED IN ACM-TRANS. MATH. SOFTWARE, VOL.12, NO. 2,
        JUN., 1986, P. 150.
        SUBROUTINE NAME     - CUBGCV

        ---------------------------------------------------------------

        COMPUTER            - VAX/DOUBLE

        AUTHOR              - M.F.HUTCHINSON
                              CSIRO DIVISION OF MATHEMATICS AND STATISTICS
                              P.O. BOX 1965
                              CANBERRA, ACT 2601
                              AUSTRALIA

        2) The running median method involves calculating the median
        within a window of width "median_n_bins" (a tool parameter).
```

```
    The calculated median value is used as the continuum level
    at location + median_n_bins/2, and the window is shifted one bin.

    1 25 50 75 100 125 150 175
    |----------------| calculates median value for point 25
     |----------------| calculates median value for point 26
      |----------------| calculates median value for point 27
       |----------------| calculates median value for point 28
        |----------------| calculates median value for point ...
         |----------------|

    Once the running median is calculated, the
    head bins median_n_bins/2 and tail bins median_n_bins/2 of the
    continuum data remain blank.  There are 4 methods of filling in
    these sections of data;  see head_tail_cont_method method below.

    3) The truncated running median is similar to the
    running median, but with the option to exclude bins with
    counts outside a given range.  The tool parameters
    "low_tmedian" and "high_tmedian" define this range.
    For example, computing the truncated median with
    high_tmedian=0.80 excludes bins ranked in the top 20 percent
    in a given window.  If low_tmedian and high_tmedian
    are both 0, this is equivalent to the running
    median method without truncation.  The sum (high_tmedian +
    low_tmedian) must be in the range (0,1), otherwise all the data
    will be excluded.

    If the median or tmedian option is used, the parameters
    median_n_bins and head_tail_cont_method must be set (see
    below).  If the tmedian method is chosen, the low_tmedian
    and high_tmedian parameters must also be set (see below).

(median_n_bins = 50) [even integer > 4]

    Number of bins for running or truncated median for continuum
    calculations.

(continuum_offset = 1) [integer 1 -> 3]

    Offset from line center to sample continuum [Line Spread
    Function Width]. The continuum slope calculations are done using
    formulae derived from the resolving power data plotted in the
    proposer's guide.

    Line Spread Function Width:
    NOTE: these values are taken from an email from John Houck.
    NOTE: all LSF_WIDTH calculations are approximations;
          we should switch to use the PSFLIB once it is finalised

    width[angstrom] = a * lam_angstrom**b     (no m factor)
    heg a=    0.0169500   b=    -0.112000
    meg a=    0.0312400   b=    -0.168900
    leg a=    0.0529900   b=   -0.0918000
    width[mm]  = m * a * d_mm**b
```

```
        heg a=    0.0120275   b=    -0.112000
        meg a=   0.00443006   b=    -0.168900
        leg a=    0.0105108   b=   -0.0918000
        width[pixel]  = m * a * d_pixel**b
        heg a=     0.501146   b=    -0.112000
        meg a=     0.184586   b=    -0.168900
        leg a=      1.63466   b=   -0.0918000
        width[deg]  = m * a * d_deg**b
        heg a=  7.98154e-05   b=    -0.112000
        meg a=  2.93982e-05   b=    -0.168900
        leg a=  6.97506e-05   b=   -0.0918000
        width[kev]  = a * e_kev**b       (no m factor)
        heg a=   0.00103120   b=      2.11200
        meg a=   0.00164692   b=      2.16890
        leg a=   0.00339198   b=      2.09180
        width[ev]  = a * e_ev**b         (no m factor)
        heg a=  4.75711e-07   b=      2.11200
        meg a=  5.12828e-07   b=      2.16890
        leg a=  1.79909e-06   b=      2.09180

        lam_angstrom = photon wavelength in angstrom

           d_mm = dispersion distance in mm
        d_pixel = dispersion distance in pixel
          d_deg = dispersion distance in degrees

          e_kev = photon energy in keV
           e_ev = photon energy in eV

              m = grating dispersion order

        There are no factors of m = grating dispersion order for
        the wavelength and energy units cases because the dispersion
        order has already been taken into account.  For example,
        if a spectral line has width 0.1 angstrom, it will have
        that width in every order -- its just that in the higher
        orders, the line is spread out over more of the detector.

        This is why the "m" factors appear only in the detector-related
        units mm, pixel, deg.

To estimate the continuum slope for a peak
detected at pos_peak:

cont_guess_left  = data( pos_peak - cont_offset * LSF_WIDTH )
cont_guess_right = data( pos_peak + cont_offset * LSF_WIDTH )
cont_slope = (cont_guess_right - cont_guess_left) /
                 ( 2 * cont_offset * LSF_WIDTH )

The equations are solved for input spectrum "data" which is
given as a function of energy units. Cont_offset is an adjustment
parameter.

    (low_tmedian = 0.20) [real 0 -> 1]
```

```
        For truncated median, reject bin values <= (low_tmedian * median).
        The sum (high_tmedian + low_tmedian) must be less than 1.

    (high_tmedian = 0) [real 0 -> 1]

        For truncated median, reject bins values >= (high_tmedian * median).
        The sum (high_tmedian + low_tmedian) must be less than 1.

    (head_tail_cont_method = default) [default|values|extrapolate|copy]

        To estimate continuum values in the few bins near the ends of the
        spectrum, the running median estimator pads the ends of
        the observed spectrum, slightly extending its wavelength range.
        These padding values must be specified by the user.
        The available options are:
            default:      pad with zeros
            values:       pad with user specified non-zero values
                          (head_value and tail_value refer to either end)
            extrapolate:  pad by linearly extrapolating the last few data values
            copy:         pad using a copy of the last few data values

            An example of the 'copy' method:

          copy of this
            ^       ^
            |       |
        |-------|-------|
        -25 25 50
        |----------------| calculates median value for point 1
         |----------------| calculates median value for point 2
          |----------------| calculates median value for point 3
           |----------------| calculates median value for point 4
            |----------------| calculates median value for point ...
             |----------------|

    (head_value = 0) [real]

        If head_tail_cont_method=values, set short wavelength (left) end
        here.  The left median_n_bins/2 will all be set to this value.

    (tail_value = 0) [real]

        If head_tail_cont_method=values, set long wavelength (right) end
        here.  The right median_n_bins/2 will all be set to this value.

    (clobber = no) [boolean]

If set to yes, existing output files will be overwritten.

    (verbose = 1) [integer 0:5]

Program verbosity.  0 means display no messages to the screen
        while running.  5 means report maximum on what the program is
doing.
```

```
    (mode = "ql") [string]

Parameter interface mode.  "ql" stands for "query and learn",
        which prompts you for input and updates
     "specdetect.par" with your responses.
```

EXAMPLES

```
    1. Find peaks with amplitudes of at least 4 time above the
       threshold, using the median method:

> pset specdetect
        Input data file; Type I or II pha file or @list of Type I files: root_pha2.fits
        Output peaks and continuum file name [root_peaks2.fits]:
        emission line detect threshold = (thresh_emission * uncertainty) (0) (4):
        Is the input file flux corected? (no):
        optional input continuum file name [root_cont.dat] (NONE):
        absorption line detect threshold = (thresh_absorption * uncertainty) (0) (2):
        Continuum calculation method (median|spline3|tmedian) (median):
        Number of bins for median of continuum calc - EVEN number (4) (50):
        Offset from line center to sample continuum [LSF width] (1:20) (1):
        Head and tail t/median continuum calc
              (default|values|extrapolate|copy) (default):
        If head_tail_cont_method=values, set short wavelength (left) end (0):
        If head_tail_cont_method=values, set long wavelength (right) end (0):
        For truncated median, reject bin values <= (low_tmedian * median) (0:1) (0.2):
        For truncated median, reject bins values >= (high_tmedian * median) (0:1) (0):
        OK to overwrite existing output files? (yes):
        Verbosity level (0 = no display) (0:5) (0):
        mode (ql):
```

DATE

```
    05/14/98
    03/19/99
    04/15/99
    06/04/99
```

10 specidentify.hlp

```
NAME

    specidentify -- identify features in grating spectrum using
                    linelist data.

USAGE

    specidentify infile linelist_file ASC_or_Spex outfile
        [doppler_shift_unit] [source_doppler_shift]
        [tolerance] [temp_unit] [source_temp] [n_max_matches]
        [source_density] [filter_string] [clobber] [verbose]
        [mode]

DESCRIPTION

    This program uses the information from the input fitted
    peaks file (created by specmeasure) to correlate the
    list of observed lines with the database line list. The
    results show the number of line matches within the allowed
    wavelength error tolerance.  The user can apply a number of
    element/ion filters on the line list, in order to match
    only a certain group.  The user can also specify the
    source temperature and density, which helps narrow down
    the matching criteria for the lines.

    The input data can have units of: angstrom, mm, degrees,
    pixel, keV or eV.  Since the line lists are all wavelength
    based, the matching is also done in wavelength units;  we
    therefore have to convert non-angstrom input data
    into angstroms.  The results are all written in
    wavelength scale, in order to be matched to the line lists.

PARAMETERS

    infile [filename]   (root_fit2.fits)

Input data file name which contains fit line features & stats;
        this file is the output of the specmeasure tool.

    linelist_file = [$ASCDS_CALIB/Spex_linelist.fits |
                     $ASCDS_CALIB/ASC_linelist.fits  |
                     $ASCDS_CALIB/ASC_linelist_all.fits]

        Database line list for identifications.  There are currently
        three available line lists in the $ASCDS_CALIB directory:

        Spex_linelist.txt   -  Spex ascii version
        ASC_linelist.fits   -  ASC fits version created by R. Smith
                               and N. Brickhouse; this file is a subset
                               of the entire line list file.  It contains
                               the brightest 8000+ line id's with
                               wavelength range of 1 to 80 angstroms.
        ASC_linelist_all.fits   -  ASC fits version created by R. Smith
                               and N. Brickhouse; this file a complete
```

```
                           version of the line list.  It contains
                           59000+ line id's.  If you want to use
                           this file when running specidentify, you
                           MUST filter the file to narrow down the
                           number of lines to under 10 thousand.
                           ASC_linelist.fits has already been
                           filtered and should be used instead of
                           the entire list, unless you want to use
                           your own filter.

        Since the ASC line list has not been officially released, the
        current default linelist_file is set to the Spex file.
        The current format of this file is:

             id number in master list.
             element id = Fe, Si, S, etc.
             ion id = by roman numeral
             transition type = defined in e.g. Mewe and Gronenschild, 1981,
                      A&A Supp., 45, 11.
             wavelength = Angstrom
             energy = keV
             list of line strength values at 40 temperatures. (log Kelvin)

        Example 2 lines, from the current list:

        Nr  ion      Transit.   E\(eV\) Lambda\(A\)  4.0  4.1  4.2
        4.3  4.4  4.5  4.6  4.7  4.8  4.9  5.0  5.1  5.2  5.3  5.4
        5.5  5.6  5.7  5.8  5.9  6.0  6.1  6.2  6.3  6.4  6.5  6.6
        6.7  6.8  6.9  7.0  7.1  7.2  7.3  7.4  7.5  7.6  7.7  7.8
        7.9  8.0

        1658 Fe IX     A8B         6.327 1959.600   0.00     0.00
        0.00     0.00     0.00     0.00     0.00     0.00     0.00
        0.00     0.00     0.00     0.00   8.97 7.42 6.41 5.83 5.54
        5.43 5.49 5.80 6.50 7.63 9.21   0.00     0.00     0.00
        0.00     0.00     0.00     0.00     0.00     0.00     0.00
        0.00     0.00     0.00     0.00     0.00     0.00     0.00


        If using the ASC linelist, the file format can be found in
        the L2 ARD ICD.
           http://hea-www.harvard.edu/asclocal/sds/development/ICD/

    ASC_or_Spex = "Spex"  [string:  ASC|asc|Spex|SPEX|spex]

        Input line list specified must be either ASC or Spex.

    outfile = "." [filename or '.']  (root_ident2.fits)

Output identified peaks file.

        "." creates an output file name based on the input data file
        name, e.g., if features_file = "root_fit2.fits", then outfile =
        "root_ident2.fits".
```

```
        Since the Spex line list and the ASC linelist contain some
        different information, the output matching resultant columns
        are not the same, when using one vs the other.  Please be
        aware that the two line lists contain different data and
        different columns of information regarding the line.  The
        output is geared toward the input linelists' structure.
        The Grating L2 ICD specifies which columns are created
        when using the Spex vs the ASC line lists.  The results
        of running the same input data on the two different line
        lists, cannot be easily compared.

    (doppler_shift_unit = "redshift") [redshift|km/s]

The units of source_doppler_shift.

    (source_doppler_shift = 0)  [real]

The Doppler shift of the source.  The value should be set
        according to the units set in the 'doppler_shift_unit'
        parameter.  The Doppler shift is used to shift the wavelengths
        of the observed spectral peaks to the 'rest' wavelength; the
        rest wavelength is used to match the lines in the data base.
        The relativistic Doppler shift formula is used:

                delta_lam = lam_observed - lam_rest

                z = delta_lam/lam_rest =
                    (sqrt((1+v/c)/(1-v/c))) - 1)

                C = 2.99792458E5 km/s

    (tolerance = 0)

        Maximum allowed fractional error in line position. For example,
        suppose the measured confidence limits on the line are
        (c1,c2) with the best position at "c" and suppose the line
        database lists confidence limits (d1,d2) on the line position
        with the tabulated position at "d".  A line is accepted as
        a candidate identification if

                   1)  (d1,d2) overlaps (c1,c2)   OR
                   2)  if abs(c-d) < tolerance * (c+d)/2

    (temp_unit = "Kelvin") [Kelvin|keV]

The units of source_temp.

    (source_temp = unknown)  [string: 'unknown' or values:
                              1E4 -> 1E8 Kelvins, 1E11  -> 1E15 keV]

Estimated source temperature in units of temp_units.  The
        temperature is used to filter the list of candidate identifications
        for spectral features.  Only lines present in collisional
        ionization equilibrium at this temperature are retained in
        the identification list.  If the input temperature is not
```

```
        an exact match to the linelist data, interpolation is performed,
        using the closest neighbor values. If the temperature is "unknown",
        the brightest collisional ionization equilibrium line will be
        matched, for each input line.  The pipeline default is "unknown".
        The output file contains a temperature column which lists
        the temperature at which the brightest ion was found, for
        that wavelength.  If the temperature was given as input,
        it is repeated in this output column;  only lines at that
        temperature are matched.

    (source_density = 0)  [cm^-3]

The ion number density.  If density is != 0, then the
        line list is filtered with this value.  Please NOTE:
        only the ASC line lists can be filtered on density,
        since the Spex line list does not have a density feature.

    (n_max_matches = 5)     [integer 1 -> N]

        List only the N matches found per input emission line.
        Multiple line matches can occure for any of the input
        emission lines (especially with the tolerance != 0.0);
        this parameter limits the number of allowed output matches.
        The N wavelengths closest to the input emission line
        will be output, instead of all the matches found.

    (filter_string = "")  [comma separated list or blank string]

        This is a list of elements and/or ions to include or exclude
        from the line identification search.  Elements are specified
        by their 2 letter chemical symbol ("Fe", "Si"), ions are
        specified by the chemical symbol and ion charge ("Fe 0" is
        neutral iron, "Fe 25" is hydrogenic, "Fe 24" is He-like,
        etc.).  If the filter string is empty or blank, all elements
        and ions in the database are included in the search.  If specific
        elements and ions are listed, only those listed are included
        in the search.  Specific elements or ions can be excluded from
        the search by prefixing the name with a tilde character ("~Fe"
        would excluded all iron ions from the search, "~Fe 17" would
        exclude only Fe XVIII).   If inclusive and exclusive filtering
        is combined, all excluded elements/ions must appear before
        the inclusive names:  if including all iron ions except Fe +15,
        the string should read "~Fe 15, Fe")

        EXAMPLE 1:
         include Mg 7
         include Mg 14
         exclude Fe 23
         include all the rest of the ions of Fe

         filter_string = "Mg 7, Mg 14, ~Fe 23, Fe"

        EXAMPLE 2:
         exclude Mg
         exclude Si
```

```
         exclude Fe 23

                filter_string = "~Mg, ~Si, ~Fe 23"
                includes all elements and ions in the
                line list, except Magnesium, Silicon and Fe +23.

    (clobber = no) [boolean = yes or no]

Overwrite outfile if it exists.

    (verbose = 0) [0 = no display info, 5 = max display info]

Controls the amount of output on your screen while
specidentify is running.  0 means work silently.

    (mode = "ql") [string]

Parameter interface mode.  "ql" stands for "query and learn"
which prompts you for input and updates
"specidentify.par" with your responses.

EXAMPLE

    Find all the matched lines and ions to the fitted peaks found in
    "root_fit2.fits".  This file was generated by specmeasure
    which fit integrated lorentzian peaks to the output from the
    specdetect.

    > pset specidentify
Input data file with fit line features & stats [root_fit2.fits] ():
Database line list for identifications))echo
            $ASCDS_CALIB/Spex_linelist.fits ->
            /home/ascds/DS.release/data/Spex_linelist.fits):
DB line list type (ASC|asc|Spex|SPEX|spex) (Spex):
Output identified peaks file [root_ident2.fits] (.):
Source Doppler shift units (redshift|km/s) (redshift):
Source redshift (0):
Maximum allowed fractional error in line position (0):
Source temperature units (for below) (Kelvin|keV) (Kelvin):
Source temperature or 'unknown' (will find all temp matches) (unknown):
List only the N matches found per input emission line. (5) (5):
Ion number density [cm^-3]; 0 => no density filtering (0):
List of the element/ions to filter in the linelist matching ():
OK to overwrite existing output file? (no):
Verbosity level (0 = no display) (0:5) (0):
mode (ql):

    > specidentify

    (output file is root_ident2.fits)

DATE

    11/18/97
    03/19/99
```

```
04/15/99
06/04/99
```

11 specmeasure.hlp

```
NAME

    specmeasure -- fit line profiles to peaks in
                one-dimensional grating data using ascfit.

USAGE

    specmeasure infile outfile model [fit_region]
[max_fwhm] [min_fwhm] [max_continuum_slope]
        [ascfit_method] [ascfit_stats] [ascfit_confidence]
        [flux_corrected] [clobber] [verbose] [mode]

DESCRIPTION

    This program fits line profiles to detected peaks
    using the information in the peaks file created by the
    tool specdetect, as initial guesses for fitting parameters.
    Fitting is done using the ascfit library.

    There are currently 2 integrated (and 2 non-integrated)
    fitting models: lorentzInt, xsgauss (and gauss, lorentz).
    Integrated fitting should be used since the fits should be
    integrated over the bins while fitting the peaks on top
    of the continuum level.  The continuum fitting is always
    done using an integrated polynomial (polyInt).  Results
    of the fits, along with min/max fit ranges are written
    to the outfile.  Fitting can be performed on input data
    data in the following units:  angstrom, mm, degrees, pixel,
    keV and eV.

PARAMETERS

    input_data_file [filename]  (root_peaks2.fits)

        Input data LINELIST/SPECTRUM file, created by specdetect.
        The file contians at least one LINELIST and one SPECTRUM
        extension(s) which contian the peaks and the continuum
        for a sample spectrum(a).

    outfile = "." [filename or '.']  (root_fit2.fits)

Output file name, for the fit results of the data and
        continuum.  If ".", then the root is used to create
        the outfile:  infile=root_peaks2.fits ->
        outfile=root_fit2.fits

    model = "lorentzInt" [lorentzInt|xsgauss|gauss|lorentz]

Type of line profile to fit.  Currently there are
        2 integrated model choices (lorentzInt, xsgauss) and two
        non integrated model choices (gauss, lorentz).  The
        continuum is always fit with an integrated model: polyInt
        (integrated polynomial).  There is currently no way
        to change the continuum function fitting model.
```

```
    (fit_region = 2.0) [floating point]

For each line, limit the fit to data points between initial
position (angstrom) +/- fwhm times this factor.  Fit region
        (line_center +/- (fit_region * fwhm_max))

    (max_fwhm = 4.0) [floating point]

        Max allowed fwhm = (line_spread_function * max_fwhm).
        This parameter sets the limits of the best fit values
        determined by ascfit. The LSF is derived from the
        resolving power data plotted in the proposer's guide.

        Line Spread Function Width:
        NOTE: these values are taken from an email from John Houck.
        NOTE: all LSF_WIDTH calculations are approximations;
              we should switch to use the PSFLIB once it is finalised

        width[angstrom] = a * lam_angstrom**b     (no m factor)
        heg a=    0.0169500   b=    -0.112000
        meg a=    0.0312400   b=    -0.168900
        leg a=    0.0529900   b=   -0.0918000
        width[mm]  = m * a * d_mm**b
        heg a=    0.0120275   b=    -0.112000
        meg a=   0.00443006   b=    -0.168900
        leg a=    0.0105108   b=   -0.0918000
        width[pixel]  = m * a * d_pixel**b
        heg a=     0.501146   b=    -0.112000
        meg a=     0.184586   b=    -0.168900
        leg a=      1.63466   b=   -0.0918000
        width[deg]  = m * a * d_deg**b
        heg a=  7.98154e-05   b=    -0.112000
        meg a=  2.93982e-05   b=    -0.168900
        leg a=  6.97506e-05   b=   -0.0918000
        width[kev]  = a * e_kev**b       (no m factor)
        heg a=   0.00103120   b=      2.11200
        meg a=   0.00164692   b=      2.16890
        leg a=   0.00339198   b=      2.09180
        width[ev]  = a * e_ev**b         (no m factor)
        heg a=  4.75711e-07   b=      2.11200
        meg a=  5.12828e-07   b=      2.16890
        leg a=  1.79909e-06   b=      2.09180

        lam_angstrom = photon wavelength in angstrom

           d_mm = dispersion distance in mm
        d_pixel = dispersion distance in pixel
          d_deg = dispersion distance in degrees

          e_kev = photon energy in keV
           e_ev = photon energy in eV

              m = grating dispersion order
```

```
        There are no factors of m = grating dispersion order for
        the wavelength and energy units cases because the dispersion
        order has already been taken into account.  For example,
        if a spectral line has width 0.1 angstrom, it will have
        that width in every order -- its just that in the higher
        orders, the line is spread out over more of the detector.

        This is why the "m" factors appear only in the detector-related
        units mm, pixel, deg.

    (min_fwhm = 0.25) [floating point]

        Min allowed fwhm = (line_spread_function * min_fwhm).
This parameter sets the limits of the best fit values
        determined by ascfit. The LSF computation details are
        listed in the min_fwhm parameter.

    (max_continuum_slope = 0.01) [floating point]

        Max allowed fractional change in the continuum across a given
        fit region.  If the fit region lies between lambda_lo and
        lambda_hi and the continuum function evaluated at the center of
        the fit region is C(lambda_mid).  The best fit continuum slope
        satisfies:

        fabs(slope) <=  max_continuum_slope * C(lambda_mid) /
                         (lambda_hi - lambda_mid)

        Steeper slopes are not allowed.  If max_continuum_slope = 0.1,
        then the linear continuum function should be flat enough so
        that the continuum at either edge of the fit region should be
        within 10% of the value at the center.

    (ascfit_method = "simplex") [string: grid|grid-powell|levenberg-
                                 marquardt|montecarlo|monte-powell|
                                 powell|simplex|simul-ann-1|simul-ann-2|
                                 simul-pow-1|simul-pow-2]

Use this ascfit fitting method.  See ascfit
        documentation for further details.  Please note that some
        methods are better for particular input units.  We have
        found that grid and grid-powell are better methods for
        fitting data in energy units (eV and keV).

    (ascfit_stats = "chi parent") [string: chi gaussian|chi parent|
                                   chi gehrel|cash|likelihood|userstat]

Use this ascfit fitting statistical method.  Since the data
        contains zero and negative counts, some of the statistical
        methods may fail.  Please be careful when setting
        this parameter; 'chi parent' works with data which may have
        0 and negative counts.  See ascfit documentation
        for further details.

    (ascfit_confidence = 68.3) [real: 68.3|90.0|99.0]
```

```
        Confidence level for delta chi squared fitting.

    (flux_corrected = no) [boolean: yes or no]

        Is the input file flux corrected?  This parameter is currently
        unused.

    (clobber = no) [boolean: yes or no]

Overwrite outfile if it exists.

    (verbose = 0) [integer (0:5)]

Verbosity level (0 = no display)

    (mode = "ql") [string]

Parameter interface mode.  "ql" stands for "query and learn"
which prompts you for input and updates
"specmeasure.par" with your responses.

EXAMPLE

    Fit an integrated lorentzian line profile to each of the peaks found
    in "root_peaks2.fits".

    > pset specmeasure
Input data LINELIST/SPECTRUM file [root_peaks2.fits] ():
Output datafile name with best fit params [root_fit2.fits] ():
Line profile type (gaussInt|lorentzInt|gauss|xsgauss|lorentz) (lorentzInt):
Fit region (line_center +/- (fit_region * fwhm_max)) (0) (2.0):
Max allowed fwhm = (line_spread_function * max_fwhm) (0) (4.0):
Min allowed fwhm = (line_spread_function * min_fwhm) (0) (0.25):
Max allowed fractional change in the continuum across a given
                                                 fit region (0:1) (0.1):
Ascfit method: powell, simplex, grid ... (grid|grid-powell|
                       levenberg-marquardt|montecarlo|monte-powell|
                       powell|simplex|simul-ann-1|simul-ann-2|simul-pow-1|
                       simul-pow-2) (simplex):
Ascfit stats: chi gaussian, chi gehrel, chi parent, cash,
                      likelihood ... (chi gaussian|chi parent|chi gehrel|
                      cash|likelihood|userstat) (chi parent):
Confidence level for delta chi squared (68.3|90.0|99.0) (68.3):
Is the input file flux corrected? (no):
OK to overwrite existing output file? (no):
Verbosity level (0 = no display) (0:5) (0):
mode (ql):

    > specmeasure

DATE

    05/14/98
    03/19/99
```

```
04/15/99
06/04/99
```

12 tg_create_mask.hlp

```
NAME
----
    tg_create_mask -- Calculates the mask regions of the grating arms
    for AXAF flight L1 grating data files.  The output is a region
    file(s) in sky coordinates.

USAGE
-----
    tg_create_mask infile outfile input_pos_tab grating_obs
[input_psf_tab] [grating_par] [detector]
[radius_factor_zero] [width_factor_arms]
        [r_astig_max_hetg] [r_astig_max_letg] [r_mask_max_hetg]
        [r_mask_max_letg] [verbose] [clobber] [use_user_pars]
        [last_source_toread] [sA_id] [sA_zero_x] [sA_zero_y] [sA_zero_rad]
        [sA_width_heg] [sA_width_meg] [sA_width_leg] [sB_zero_x]
        ... [sC...] ... [sD...] ... [sE...] ... [sF...] ...
        [sJ_id] [sJ_zero_x] [sJ_zero_y] [sJ_zero_rad]
        [sJ_width_heg] [sJ_width_meg] [sJ_width_leg]

DESCRIPTION
-----------
    This program can be run in a variety of ways.  The three main
    options are:

(1) Run on a single input event file to get a single output
    resultant region file.   This method does not use user specified
    mask sizes and calculates the grating mask from calibration data
    and error-budget approximations.  A zero order source position
    table is also reguired as input (for details on table, see below);
    as well as a calibration data file which has the fwhm of an AXAF
    source versus to off-axis angle.  There is a default table pointer
    used by the tool, but it can be reset by the user.

(2) Run on a stack of input event files and position files
    to get a stack of resultant region files.  This method is the
    same as the the above, with the exception of working on
    multiple input/output files.

(3) Run only on a SINGLE input event file and specify the
    regions explicitly (this method does not run on stacks).  The
    difference from methods (1) and (2) above is that the
    the mask sizes are completely user specific.  This means that the
    region mask sizes are not calculated using the calibration table;
    instead, the user parameters are used as input to the mask
    sizes.  The tool does simple calculations of mask lengths to
    make sure they fit on the detector space, as well as write the
    resultant region file in the correct format for the next tool
    to use as input.

    Noting the above methods of running the program, the user must
    decide whether they want the mask regions to be calculated by
    the tool or whether they want to set the information themselves.
    Read the appropriate section below of how to run the tool.
    The defaults are set up for automated calculations of the masks
```

```
(methods (1) and (2) from above).


Automatic Calculations
----------------------
Required inputs are an event parameter/file/stack, and
source position table/stack.  The tool uses the source positions
in your table and calculates all the mask characteristics using
the defocus, astigmatism, off axis angle, the telescope roll and
it's tolerance, along with calibration grating data characteristics.
The calculations of the mask involve:

The routine tg_calc_zero_mask() is called by tg_create_mask() to
calculation the zero order masks in r/d coordinates.  Zero order
radius is calculated using the defocus and interpolated fwhm of
the data.  A scaling width factor (which is set by the user) is
used to increase the zero order calculated radius value.
The equation is:

defocus  = sim_x header keyword
roll     = roll_nom header keyword
roll_tol = set to 0.075 degrees by default; this value may need
           to get updated during flight

NOTE: Positive roll angle is clockwise on the sky x/y plane.
Since this is opposite the counterclockwise positive angle
logic, the input roll is flipped when it is read.  Flipping
the roll takes out the clockwise/counterclockwise logic
differences.

d_defocus=(2*gratng_ring_radius/flight_rowland_diam^2)*(|defocus|)
F_BOX=2*sqrt(3)
F_GAUSS=2*sqrt(2*log(2))
radius=(sqrt( (interpolated_fwhm/F_GAUSS)^2 + (d_defocus/F_BOX)^2) )
      * width_factor_zero_order

The routine tg_calc_arm_mask() is called by tg_create_mask() to
calculation grating arm mask in r/d coordinates.  Arm mask width
are calculated using the defocus, interpolated fwhm, max r and
roll tolerance.  A scaling width factor (which is set by the user)
is used to increase the arm mask calculated width value.
The equation is:

d_astig_defocus=(2*grating_ring_radius/grtptr->rowland)
                *(r_max^2) + |(defocus/flight_rowland_diam)|
d_roll=roll_tollerance*r_max
F_BOX=2*sqrt(3)
F_GAUSS=2*sqrt(2*log(2))
width =  2*sigma*width_factor + roll'
width=2*(sqrt( (interpolated_fwhm/F_GAUSS)^2) +
             ((d_astig_defocus/F_BOX)^2) )*(width_factor_arms) + d_roll

Once mask widths are calculated, mask lengths are also calculated
using grating characteristics.  Both the mask lengths and widths are
then converted to sky coordinates and written to the region description
```

```
    file/stacks.


    Explicit specification of the Mask
    ----------------------------------
    Required inputs are an event file parameter file/stack.
    The user must set the following mask parameters in order for the
    tool to work properly:
# --------------------------------------------------------------------------
# The parameters below are to be set ONLY if the user wants to use their
# own grating mask sizes instead of having the masks automatically generated.
# Only ONE input file, with up to 10 soures, can be processed using the user
# params.  @ lists of multiple files can only be done with automated mask
# processing, or by running each file individually with hand set mask sizes.
# To start, you MUST set the following parameters:
#
# > pset tg_create_mask use_user_pars=yes last_source_toread=[letter A -> J]
#
# The parameter last_source_toread should be set to the last source letter
# for which you will enter parameters.  If you want to input 2 sources
# (regardless of their source id's), the last_source_toread=B. Sections
# A -> J are for (upto) 10 user specified sources.  In each sections,
# each source must have an ID, a zero order center position specified,
# as well as the grating mask width(s).  An example with 2 HETG sources,
# with src_id's 6 and 3:
#
# > pset tg_create_mask use_user_pars=yes max_source_num=B
# > pset tg_create_mask sA_id=6 sA_zero_x=4762.34 sA_zero_y=2344.29
# > pset tg_create_mask sA_zero_rad=35 sA_width_heg=25 sA_width_meg=28
# > pset tg_create_mask sB_id=3 sB_zero_x=4063.54 sB_zero_y=6346.62
# > pset tg_create_mask sB_zero_rad=45 sB_width_heg=50 sB_width_meg=75
#            (units are all in sky pixels)
#
# NOTE: for Continuous Clocking data (CC mode), the HETG mask does not
# require the s#_width_heg, since the meg mask will encompase the entire
# data set.  HEG event processing in CC mode is done using the next
# tool tg_resolve_events.
# --------------------------------------------------------------------------

    You must set all the grating specific mask parameters for each of
    the sources you are working with.  Each source has the following
    possible group of params to be set:

sA_id,i,h,,,,"Source A - source id number"
sA_zero_x,r,h,,,,"Source A - x position of zero order"
sA_zero_y,r,h,,,,"Source A - y position of zero order"
sA_zero_rad,i,h,,,,"Source A - radius of zero order mask"
sA_width_heg,i,h,,,,"Source A - width of heg mask in sky pixels"
sA_width_meg,i,h,,,,"Source A - width of meg mask in sky pixels"
sA_width_leg,i,h,,,,"Source A - width of leg mask in sky pixels"

    The zero order id, x, y, and rad MUST be set (even if the zero order
    is not imaged on the detector).  Setting the grating width
    depends on which grating you are using.  If you have the HETG, then
    set the width_heg AND width_meg params; if you have the LETG, then
```

```
    set the width_leg.
    These groups of params should be set for each of the sources in
    your field of view of your event file.

    Once mask widths are set and the tool is run, mask lengths are
    calculated using grating characteristics.  These mask lengths
    are then converted to sky coordinates and written to the region
    description file/stacks.  The rest of the mask characteristics
    are used from the parameter file settings of the mask placement.


PARAMETERS
----------
    The parameters for this tool are complicated if the user is specifying
    all the mask characteristics on their own.  However, the parameter
    setting get far more simple of the user does not specify the mask
    directly and lets the tool do all the calculations.  I will try to
    distinguish on method from the other, in the parameter descriptions.


    infile = [filename | @stack.lis]

        Input event file or stack.  Each input file could
        be an event file or just the ascii header of the event file.
If using an event file, there must be an EVENTS extension
        in order for the tool to open an extension with a full header.
        If using an ascii header of the event file, this file can be
        created using the tool "dmmakepar", which creates header files.
        Please be sure to use the EVENTS extension when creating the
        header file:  dmmakepar infile="file.fits[EVENTS]".
        A stack can contain a list of event files or event header files.
        If using the user specific mask characteristics, you can only
        have one input header file, stack lists are not permitted.

        An ascii event parameter file can be created using the task
        dmmakepar; this task takes an event file and writes it
        to an ascii parameter file.

        Sample input event ascii parameter stack file list:
              > cat header.lis
                   sample1_evt1a.par
                   sample2_evt1a.par
                   sample3_evt1a.par

        Sample input event file stack list:
              > cat header.lis
                   sample1_evt1a.fits
                   sample2_evt1a.fits
                   sample3_evt1a.fits


    outfile = [filename | @stack.lis]

Output region file or stack of region files.  If using the
        user specific mask characteristics, you can only have one
```

```
        output region file name, stack lists are not permitted.

        Sample output region stack list file:
              > cat outfile.lis
                   sample1_evt1a.reg
                   sample2_evt1a.reg
                   sample3_evt1a.reg

    input_pos_tab = [filename | @stack.lis]

Input table with zero order positions or stack of tables
        with zero order positions.  This file/stack is only required
        if the user is NOT specifying the mask sizes on their own.

        The table must be in fits format; this table is the output of
        the tool tgdetect (gratings detect: celldetect output
        along with source id information).  For each source, there must
        be a row of information containing:

           x    --- the x sky pixel position of the zero order center
           y    --- the y sky pixel position of the zero order center
           tg_srcid   --- unique source id you want to assign to each
                       source (available numbers range from 1-N)

         IMPORTANT: The table EXTNAME/HDUNAME name must be: "SRCLIST"
         NOTE: This tool processes up to 10 sources;  if there are
               more than 10 sources in this position table, only the
               first 10 will be processed.  The source ID does not
               have to match the row number in the table;  each ID can
       be an arbitraty number, and they can be in any order
               inside the table.  However, all ID's in a table must
               be unique;  the program will exit if any source id is
               duplicated.

        The user can also put a filter on the position table file,
        which filteres out some of the rows in the table.  EX:
        process only the source id = 1:

        postable.fits[SRCLIST][tg_srcid=1]

        If there is no filter specified, the entire table is used
        for processing.  If there are more than 10 sources and a
        filter has not been applied, only the first 10 listed are
        used for mask calcualtions.

        The off axis angle is calculated using the X,Y position in
        the position table along with the center position of the
        detector pixel plane:

        separation=sqrt(pow(fabs(position_X - pix_image_center_X),2) +
                       pow(fabs(position_Y - pix_image_center_Y),2));
        off axis angle = (separation)*(scale_pix_to_asec)*(SEC2MIN);

        Sample table data:
```

```
      XTENSION= 'BINTABLE'           / binary table extension
      EXTNAME = 'SRCLIST '           / name of this binary table extension
      HDUNAME = 'SRCLIST '           / ASCDM block name

        RA              DEC             RA_ERR          DEC_ERR
        deg             deg             deg             deg
      1   1.3209751E-04   1.1934853E-04   3.0824431E-06   3.1925649E-06

        X               Y               X_ERR           Y_ERR
        pixel           pixel           pixel           pixel
      1   4.0955295E+03   4.0973770E+03   2.2646520E-02   2.3455579E-02

        DETSIZE     BKGSIZE     NPIXSOU     NPIXBKG     NET_COUNTS
        pixel       pixel       pixel       pixel       count
      1           6           8          36          28   1.1500000E+03

        NET_COUNTS_ERR  BKG_COUNTS      BKG_COUNTS_ERR  NET_RATE
        count           count           count           count/s
      1   3.5133537E+01   2.5714285E+00   3.4178300E+00   0.0000000E+00

        NET_RATE_ERR    BKG_RATE        BKG_RATE_ERR    EXPTIME
        count/s         count/s         count/s/pixel   s
      1   0.0000000E+00   0.0000000E+00   0.0000000E+00   0.0000000E+00

        SNR             SIG_MINOR       SIG_MAJOR       INCL_ANGLE
                        pixel           pixel           deg
      1   3.2744457E+01   7.6528513E-01   8.0000687E-01   1.8847097E+01

        PSFRATIO        BLOCK       TG_SRCID

      1   1.6192012E+00           1      1


    (input_psf_tab = $ASCDS_CALIB/hrmaD1996-11-01wpsfN0001.fits) [filename]

Calibration file with mirror psf vs off-axis angle.
        Only required if no user mask parameters
        are specified. The current default can be reset to any other
        table with the same general information - the table type must
        be same. The $ASCDS_CALIB area contains the default
        psf_fwhm.fits table.

        IMPORTANT:: The table EXTNAME/HDUNAME must be: "AXAF_WPSF"

        Sample table:

> fdump hrmaD1996-11-01wpsfN0001.fits
Name of optional output file[STDOUT]
Names of columns[]
Lists of rows[-]
SIMPLE  =                    T / file does conform to FITS standard
BITPIX  =                   16 / number of bits per data pixel
NAXIS   =                    0 / number of data axes
EXTEND  =                    T / FITS dataset may contain extensions
COMMENT   FITS (Flexible Image Transport System) format defined in Astronomy and
```

```
COMMENT   Astrophysics Supplement Series v44/p363, v44/p371, v73/p359, v73/p365.
COMMENT   Contact the NASA Science Office of Standards and Technology for the
COMMENT   FITS Definition document #100 and other FITS information.
ENDSIMPLE  =                    T / file does conform to FITS standard
BITPIX  =                   16 / number of bits per data pixel
NAXIS   =                    0 / number of data axes
EXTEND  =                    T / FITS dataset may contain extensions
COMMENT   FITS (Flexible Image Transport System) format defined in Astronomy and
COMMENT   Astrophysics Supplement Series v44/p363, v44/p371, v73/p359, v73/p365.
COMMENT   Contact the NASA Science Office of Standards and Technology for the
COMMENT   FITS Definition document #100 and other FITS information.
END

XTENSION= 'BINTABLE'           /Binary table written by MWRFITS
BITPIX  =                    8 /Required value
NAXIS   =                    2 /Required value
NAXIS1  =                    8 /Number of bytes per row
NAXIS2  =                   13 /Number of rows
PCOUNT  =                    0 /Normally 0 (no varying arrays)
GCOUNT  =                    1 /Required value
TFIELDS =                    2 /Number of columns in table
COMMENT
COMMENT  *** Column formats ***
COMMENT
TFORM1  = 'E       '           /
TFORM2  = 'E       '           /
COMMENT
COMMENT  *** End of required fields ***
COMMENT
EXTNAME = AXAF_WPSF           /FWHM of zero-order PSF
COMMENT                                         +------------------+
COMMENT                                         | AXAF FITS   File |
COMMENT                                         +------------------+
COMMENT             *********************************************************
COMMENT             >    This file is written following certain AXAF-ASC    <
COMMENT             >    conventions which are documented in ASC-FITS-1.1   <
COMMENT             *********************************************************
COMMENT                       / Configuration control-------------------------
ORIGIN  = 'ASC     '
CREATOR = 'xxx - Version 0.0'
REVISION=                    0 / Processing system revision number
COMMENT $Id: ascfits.tex,v 1.4 1998/05/12 14:50:42 arots Exp arots $
CHECKSUM= '0000000000000000'   / ASCII encoded HDU checksum
DATASUM = '         0'         / Data unit checksum written in ASCII
CONTENT = 'CDB_HRMA_WPSF    '  / FWHM of zero-order PSF
HDUNAME = 'AXAF_WPSF    '      / Zero-order PSF FWHM
HDUSPEC = 'ARD Exposure Maps, Inst.Maps, TG L1.5-2 ICD' / ICD reference
HDUDOC  = 'ASC-FITS-1.1: McDowell, Rots: ASC FITS File Designers Guide'
HDUVERS = '1.0.0   '
HDUCLASS= 'ASC     '
HDUCLAS1= 'RESPONSE     '
HDUCLAS2= 'PSF     '
HDUCLAS3= 'WPSF    '
LONGSTRN= 'OGIP 1.0'           / The OGIP long string convention may be used.
CCLS0001= 'CPF     '
```

```
CDTP0001= 'DATA    '
CBD10001= 'THETA(0.0-30.0)arcmin'
CCNM0001= 'WPSF    '
CVSD0001= '1996-11-01T00:00:00'
CDES0001= 'Radial point spread function widths'
LONGSTRN= 'OGIP 1.0'           / The OGIP long string convention may be used.
COMMENT   This FITS file may contain long string keyword values that are
COMMENT   continued over multiple keywords.  This convention uses the  '&'
COMMENT   character at the end of a string which is then continued
COMMENT   on subsequent keywords whose name = 'CONTINUE'.
HISTORY   Input event file: hrcx123456789N000_evt0.fits
DATE    = '1998-0803T00:00:39' / Date and time of file creation (UTC)
MISSION = 'AXAF    '           / Mission is AXAF
INSTRUME= 'HRMA    '           / HRC, ACIS, EPHIN, S/C subsystems
DETNAM  = '        '           / Detector name
GRATING = 'NONE    '           / Grating
DATACLAS= 'SIMULATED'          / If fake data,
                               / include this (default is DATACLAS='OBSERVED')
COMMENT
COMMENT  *** Column Names ***
COMMENT
TTYPE1  = 'OFFAXIS '           /
TTYPE2  = 'FWHM    '           /
TUNIT1  = 'arcmin  '           /
TUNIT2  = 'arcsec  '           /
TLMIN1  =              0.00000 /
TLMIN2  =              0.00000 /
TLMAX1  =              60.0000 /
HISTORY Testing dph 980622; first bintable extension; principal hdu
END

        OFFAXIS         FWHM
        arcmin          arcsec
      1   0.0000000E+00   7.1070480E-01
      2   8.3300002E-02   7.2481644E-01
      3   1.6670001E-01   7.7298743E-01
      4   3.3329999E-01   8.4168440E-01
      5   5.0000000E-01   7.2425443E-01
      6   1.0000000E+00   8.5217273E-01
      7   2.0000000E+00   8.2401264E-01
      8   5.0000000E+00   2.7854469E+00
      9   1.0000000E+01   1.0696798E+01
     10   1.5000000E+01   1.7453949E+01
     11   2.0000000E+01   3.1037743E+01
     12   2.5000000E+01   3.8922913E+01
     13   3.0000000E+01   4.7982021E+01
```

```
    (grating_par = grating) [filename]

Name of grating constants parameter file.  The default is
        grating.par

    grating_obs = header_value [HETG|HEG|MEG|LETG|header_value]
```

```
Observed grating type: HETG or HEG or MEG or LETG.  The string
        "header_value" can be used; this allowes the tool to
        use the header keyword GRATING from the input header file,
        to set this parameter.

        The Drake Flat is currently not available.

        NOTE: ACIS CC HETG mode data is processed using the MEG,
              grating since data is only located in one strip/
              arm.  There is a note written within comments for
              output masks in CC mode, reminding the user that
              the mask region is over a CC mode dataset, not
              a grating 'arm'.

    (detector = header_value) [ACIS|HRC-I|HRC-S|header_value]

Detector type used in the observation: ACIS | HRC-I | HRC-S .
        The string "header_value" can be used; this allowes the tool
        to use the header keyword DETNAME from the input header file,
        to set this parameter.

    (radius_factor_zero = 50) [real]

A scale factor which multiplies the approximate calculation
        of the one-sigma zero order radius.

    (width_factor_arms = 20) [real]

A scale factor which multiplies the approximate one-sigma
        width of the region in the cross-dispersion direction

    (r_astig_max_hetg = 0.56) [float]

Max grating r coord (degrees, along the dispersion) for HETG
        astigmatism calculation.

    (r_astig_max_letg = 1.10) [float]

Max grating r coord (degrees, along the dispersion) for LETG
        astigmatism calculation.

    (r_mask_max_hetg = 0.992) [float]

Max grating r coordinate (degrees) for HETG mask (to support
        offset pointing).

    (r_mask_max_letg = 2.1) [float]

Max grating r coordinate (degrees) for LETG mask (to support
        offset pointing).

    (verbose = 0) [integer (0:5)]

Verbosity level: 0 - no output, 5 - max display.
```

```
    (clobber = no) [boolean (YES|yes|NO|no)]

Clobber existing outfile?


    THE FOLLOWING PARAMETERS ARE ONLY TO BE SET IF THE USER WANTS TO
    SPECIFY THE MASK SIZES EXPLICITLY.  YOU MUST SET THE APPROPRIATE
    SOURCE ID PARAMETERS WHICH CORRESPOND TO THE SOURCES YOU WANT TO
    PROCESS.

    (use_user_pars = no) [bool]

Use the mask parameters below: yes or no?

    (last_source_toread = A) [char A|B|C|D|E|F|G|H|I|J]

Last source name to be read, char A->J.

    # ----------------------------------------------------------------------
    #  Source A -> J parameters x 10
    # ----------------------------------------------------------------------
    # NOTE: for Continuous Clocking data (CC mode), the HETG mask does not
    # require the s#_width_heg, since the meg mask will encompase the entire
    # data set.  HEG processing is done using the next tool tg_resolve_events.
    # ----------------------------------------------------------------------

    (s?_id = ) [integer]

Source ? - source id number.

    (s?_zero_x = ) [float]

Source ? - x position of zero order in sky x pixels.

    (s?_zero_y = ) [float]

Source ? - y position of zero order in sky y pixels.

    (s?_zero_rad = ) [float]

Source ? - radius of zero order mask in sky pixels.

    (s?_width_heg = ) [float]

Source ? - width of heg mask in sky pixels in sky pixels.

    (s?_width_meg = ) [float]

Source ? - width of meg mask in sky pixels in sky pixels.

    (s?_width_leg = ) [float]

Source ? - width of leg mask in sky pixels in sky pixels.
```

```
    (mode = "ql") [string]

Parameter interface mode.  "ql" stands for "query and learn"
which prompts you for input and updates
"tg_create_mask.par" with your responses.

EXAMPLE
-------
    Three examples will be listed, one for each of the different
    methods of using the tool.

    (1) Single input and output files;  changing the width factors
        to be bigger than the defaults.
% tg_create_mask
          infile=sample1_evt1a.fits outfile=sample1_evt1a.reg
          input_pos_tab=table1_src1a.tab grating_obs=HETG detector=ACIS
          radius_factor_zero=20 radius_factor_arms=30

        The result should be a region sample1_evt1a.reg with the mask size
        information.

    (2) Stacks of input event file parameters and output region files.
% tg_create_mask
          infile=@input_par.lis outfile=@outfile.lis
          input_pos_tab=@input_table.lis grating_obs=LETG detector=HRC

        The result should be region files which were listed in the
        outfile.lis; with the mask size information.
        % cat input_par.lis
                   sample1_evt1a.par
                   sample2_evt1a.par
                   sample3_evt1a.par
        % cat input_table.lis
                   sample1_src1a.fits
                   sample2_src1a.fits
                   sample3_src1a.fits
        % cat outfile.lis
                   sample1_evt1a.reg
                   sample2_evt1a.reg
                   sample3_evt1a.reg

    (3) User specific mask sizes for a single data set.  There are
        2 HETG sources of interest with ID 3 and 5.
% tg_create_mask
          infile=sample1_evt1a.par outfile=sample1_evt1a.reg
          grating_obs=header_value detector=header_value
          use_user_pars=yes last_source_toread=B sA_id=3
          sA_zero_x=4832.93 sA_zero_y=6327.47 sA_zero_rad=76
          sA_width_heg=47 sA_width_meg=59 sB_id=5
          sB_zero_x=3218.735 sB_zero_y=5382.63 sB_zero_rad=93
          sB_width_heg=62 sB_width_meg=78

        The result should be a region sample1_evt1a.reg with the mask
        size information.  A sample output file is:
```

```
#.GRATING hetg
#.SOURCE 1
#.COMPONENT 0    # TG_PART = 0 = gzo
physical; circle 4097.54 4097.50 26
#.COMPONENT 1    # TG_PART = 1 = heg
physical; rotbox 4097 4097 8182 41 5.18d
#.COMPONENT 2    # TG_PART = 2 = meg
physical; rotbox 4097 4097 8182 56 -4.75d
```

The region file format places grating specific information as comments in the region file. The "#." signifies special grating comments which describe the grating (#.GRATING), source id (#.SOURCE) and grating part (#.COMPONENT). The grating describes the grating name for the entire region file; the source number is for the next lines of grating regions and is assumed to be the same until another source id is listed; the component preceeds each grating region.

ACIS only Continuous Clocking (CC) mode:
If an ACIS dataset is in CC mode, then the mask contains a note reminding the user that the HETG is processed using the MEG mask (with grating angle equal to zero). In the case of LETG, there is only one mask, as in non-CC mode dataset, but the CC mode message is still displayed. Here are two example of CC mode region files; one using the HETG and one using the LETG.

```
#.GRATING hetg
#.SOURCE 1
#.COMPONENT 0    # TG_PART = 0 = gzo
physical; circle 4097.54 4097.50 26
#.COMPONENT 1    # TG_PART = 2 = meg
# NOTE: This is CC data; an MEG mask is used for processing.
# NOTE: HEG events (not listed in mask) will be found using tg_resolve_events.
physical; rotbox 4097 4097 8182 59 0.00d


#.GRATING letg
#.SOURCE 1
#.COMPONENT 0    # TG_PART = 0 = gzo
physical; circle 4095.00 4097.00 44
#.COMPONENT 1    # TG_PART = 3 = leg
# NOTE: This is CC data; an LEG mask is used for processing.
physical; rotbox 4095 4097 8422 77 340.00d
```

The tool "dmrega2fits" can convert this output ascii file to a standard FITS region file. This tool can also append the FITS region to another fits file; resulting in a REGION extension on the output.

```
DATE
----
    01/11/98
    04/23/98
    06/15/98
```

```
09/16/98
12/07/98
02/23/99
03/19/99
04/15/99
04/29/99
06/03/99
```

13 tg_resolve_events.hlp

```
===============================================================================
                       HELP FILE for TG_RESOLVE_EVENTS
===============================================================================

NAME
----
     tg_resolve_events -- computes grating diffraction coordinates and adds
                          columns to the event file.


USAGE
-----
     tg_resolve_events infile outfile regionfile [acaofffile]
        [alignmentfile] [logfile] [rmfile] [grating_obs] [detector]
        [energy_lo_adj] [energy_hi_adj] [time_offset] [random] [rand_seed]
        [eventdef] [qp_internals] [qp_pagesize] [qp_bucketlen] [verbose]
        [clobber] [stdlev1] [mode]


DESCRIPTION
-----------
     The tg_resolve_events tool is utilized to perform level 1.5 processing
     on gratings data. This involves taking an input event list, a gratings
     region mask, and a response matrix table and computing coordinates
     (dispersion/cross-dispersion, grating diffraction pixel coords) and
     other gratings related data- ie. wavelength, order (acis only), and
     part (HEG, MEG, LEG) ids. Unless the eventdef parameter is modified,
     the following columns are added by default: tg_r, tg_d (dispersion and
     cross dispersion angles), tg_part, tg_mlam, tg_m, tg_lam, tg_srcid,
     gdpx, gdpy and tg_smap.


PARAMETERS
----------
     Tg_resolve_events requires a fits (or qpoe) input file containing event
     data, a region file containing one or more grating region discriptors,
     a response matrix table, and a parameter file containing the
     following parameters:

     infile     - (input event file- either qpoe or fits bin table)
                - this is an auto parameter that is used to specify the input
                  event file from which tg_resolve_events will process data.

     outfile    - (output event file- either qpoe or fits bin table)
                - this is an auto parameter that is used to specify where the
                  processed events are to be written. The type of the output
                  file is determined by the extension (.fits or .qp) of the
                  file name provided. If no extension is provided, the file
                  type defaults to fits bin-table.

     regionfile - (input region file or stack of region files)
                - this is an auto parameter that specifies the grating masks

     acaofffile - (input aspect offsets files- fits bin table format)
```

```
           - this is a hidden parameter that is used to specify the
             aspect offsets file for gratings coordinate
             transformations; this file is created in L1 event
             processing; NONE can be used when there is no aspect.

alignmentfile - (input alignment files- fits bin table format)
           - this is a hidden parameter that is used to specify the
             sim/fam alignment needed for coordinate
             transformations; this file is created in L1 event
             processing; NONE can be used when there is no alignment
             information for the data.

logfile    - (output text file or 'NONE' or 'stdout')
           - This value specifies the name of the ascii text file
             which this tool will generate if the verbose parameter (see
             below) is set to a value other than 0. If the value is set to
             "NONE" or "stdout", the output will be redirected to
             standard output (typically the screen).

rmfile     - (input table file- fits bin table format)
           - This is the name of the energy limit table, created from a
             response matrix, that is used to resolve orders (acis only).

grating_obs - (HETG, HEG, MEG, LETG, header_value)
           - This parameter specifies the grating type so that the
             appropriate values (such as grating period) can be read in
             from the appropriate grating parameter file.  If
             header_value is specified, the tool will use the
             GRATING keyword in the event header.  If the user
             specifies the grating name, it is checked against
             the header; a warning is printed if the header does
             not match the user input.

detector   - (acis, hrc-s, hrc-i, header_value)
           - The detector on which this data was collected. This
             parameter specifies the instrument parameter file to read
             in order to configure the instrument specific information
             used by tg_resolve_events.  If
             header_value is specified, the tool will use the
             DETECTOR keyword in the event header.  If the user
             specifies the detector name, it is checked against
             the header; a warning is printed if the header does
             not match the user input.

energy_lo_adj - (real number)
           - This parameter specifies the lowerbound tolerance factor to
             apply to the PI limit given in the response table for
             calculating orders. The lo-limit in the rmfile table is
             increased by this factor. (The default value is 1.0.)

energy_hi_adj - (real number)
           - This parameter specifies the upperbound tolerance factor to
             apply to the PI limit given in the response table for
             calculating orders. The hi-limit in the rmfile table is
             increased by this factor. (The default value is 1.0.)
```

```
time_offset - (real number)
            - Offset to add to event time to synch w/ alignment data.

random      - (yes, no)
            - This parameter controls whether a randomization factor is
              added to detector pixel coordinates to anti-alias the result.
              (The default is no.)

rand_seed   - (1 positive integer)
            - Random seed (for pixlib), 0 = use time dependent seed
             (The default is 1.)

eventdef    - (output format string)
            - This field allows the user to specify the contents of
              the output file. The value may either be set to a desired
              list of columns and data types by the user, or a redirect to
              predefined event definitions may be utilized via the redirect
              command.  The tool set_tg_resolve_eventdef can be sed
              to set up the eventdef parameter, depending on whether
              the instrument used was ACIS or HRC.

qp_internals - (yes, no, redirect to qpoe.par file)
            - This boolean parameter instructs tg_resolve_events whether
              or not to use the the page and bucket length values specified
              in the input file or to use the default values.

qp_pagesize - (integer or redirect to qpoe.par file)
            - allows the user to specify the qpoe page size

qp_bucketlen - (integer or redirect to qpoe.par file)
            - allows the user to specify the qpoe bucket length

verbose     - (0..5)
            - Option which allows the user to request a varying level of
              textual output based upon the program execution. Levels range
              from 0 to 5 with 0 representing no information and 5
              representing as detailed a log as possible. the log is
              written out to the value specified in the logfile
              parameter (see above).

clobber     - (no, yes, overwrite existing outfile)
            - Option which allows the user to overwrite the output
              event list, if it already exists.

stdlev1     - (output event definition string)
            - This event definition specifies the default output columns
              that will be written to the output file if the eventdef
              variable is redirected here.  See eventdef parameter
              (above) for more information on setting this parameter.
```

EXAMPLES

```
--------
     Tg_resolve_events may be executed utilizing either a parameter file or
```

```
command line arguments. When using command line arguments, parameters not
explicitly specified will default to the values contained in the parameter
file. Use "plist tg_resolve_events" to view the current parameter values.
This will cause all parameters and their respective values to be echoed to
the screen. The data will resemble the following:

     infile =        Input event file or stack
    outfile =        Output event file or stack
 regionfile =        Region file stack
(acaofffile = NONE)  aspect offset file ( NONE | none | <filename>)
(alignmentfile = NONE)  sim/fam alignment file ( NONE | none | <filename>)
   (logfile = tg_resolve_events.log) debug log file
    (rmfile = ))echo $ASCDS_CALIB/acisD1996-11-01irmfN0001.fits  -> /home/ascds/DS.daily/data/acisD1996-11-01irmfN0001.fits) RM lookup table for order resolving (acis only)
(grating_obs = )      Grating type (HETG|HEG|MEG|LETG|TOGA)
 (detector = )      Detector type: ACIS|HRC-I/HRC-S|HSI
(energy_lo_adj = 1.0)   Lower PI limit factor
(energy_hi_adj = 1.0)   Upper PI limit factor
(time_offset = 0)     Offset to add to event time to synch w/ alignment data
    (random = no)    allow randomized digitation of floating point numbers
 (rand_seed = 1)     random seed (for pixlib), 0 = use time dependend seed
  (eventdef = )stdlev1 -> {d:time,i:expno,l:gdp,d:rd,s:chip,s:tdet,d:det,d:sky,s:ccd_id,l:pha,s:pi,f:energy,s:grade,s:fltgrade,s:ccdnode,s:tg_m,d:tg_lam,d:tg_mlam,s:tg_srcid,s:tg_part,s:tg_smap}) output format definition
(qp_internals = )qpoe.qp_internals -> yes) prompt for qpoe internals?
(qp_pagesize = )qpoe.qp_pagesize -> 8192) page size for qpoe file
(qp_bucketlen = )qpoe.qp_bucketlen -> 16384) bucket length for qpoe file
   (verbose = )file.display -> 1) level of debug detail (0=none, 5=most)
 (clobber = yes)             Clobber outfile if it already exists?
   (stdlev1 = {d:time,i:expno,l:gdp,d:rd,s:chip,s:tdet,d:det,d:sky,s:ccd_id,l:pha,s:pi,f:energy,s:grade,s:fltgrade,s:ccdnode,s:tg_m,d:tg_lam,d:tg_mlam,s:tg_srcid,s:tg_part,s:tg_smap}) event format definition string
    (mode = ql)


Values enclosed in parentheses, such as random, must be changed prior to
execution or on the command line if the specified value is not desired.
Values not enclosed in parentheses, such as outfile, are auto parameters
which the user will be prompted to either input or hit <return> to accept
the current value. To change the value of a parameter type
"pset file parameter=value", ie. to change grating to HETG type:

      pset tg_resolve_events.par grating=HETG

Alternatively the parameter file editor pedit or any text editor may
be used.

Once all of the parameters have been set to the desired values, the
routine may be invoked by simply typing "tg_resolve_events". The routine
will prompt the user for the input (infile) event file, the output
(outfile) event file, and the region file (regionfile) names.
The user may either accept the suggested value by pressing <return>
or may enter a new value. The routine will then execute. Upon completion
the output event file will be written out, as well as a logfile if the
verbose parameter is nonzero.

To run with command line arguments, simply type the routine name followed
by parameter assignments. The format is
"function_name par1=val1 par2=val2 ... parn=valn", ie.
```

```
          tg_resolve_events infile=in.fits outfile=out.fits regionfile=mask.reg

     If the auto parameters are specified on the command line the routine will
     not prompt the user to accept or modify the values of these parameters.

     Following are a list of examples illustrating some of the various
     processing options available with tg_resolve_events.

     A) Process the hetg file grating1.fits using the region mask 2_src.reg and
        the response matrix ccdsigma.fits. Randomize the output coords to
        reduce aliasing, and print out a detailed log to the file run.log

        1) set up the hidden parameters

           pset tg_resolve_events.par grating_obs=HETG
           pset tg_resolve_events.par rmfile=acisD1996-11-01irmfN0001.fits
           pset tg_resolve_events.par detector=acis
           pset tg_resolve_events.par random=yes
           pset tg_resolve_events.par debug=5
           pset tg_resolve_events.par logfile=run.log

        2) invoke the tool

           tg_resolve_events

        3) When prompted for the "Input event file stack",
           enter grating1.fits

        4) When prompted for the "Region file stack",
           enter 2_src.reg

        5) When prompted for the "Output event file stack",
           enter test_out.fits

        6) examine results


NOTES
-----
     General-
        It is the user's responsibility to ensure that the parameters are
        correctly set.

        The requirements/algorithms for dealing with the drake flat are
        TBD are are not currently supported in tg_resolve_events

        If event files are in fits bin-table format, the event data is
        read from or written to the "EVENTS" extension. Necessary header
        keywords (ie. SIM_Z) are read from the principal header (also
        the EVENTS extension).

        WCS information is added to output coordinate columns if they are
        specified as vectored output coordinate pairs in the output
        eventdef (ie. "s:det" instead of "s;detx,s;dety" or "s:sky instead
        os "s:x,s:y").
```

```
SEE ALSO
--------
     For information regarding the data contained in the input/output files
     used by this task, please see:

        Acis Data Products:    Level 1 to ASC Archive ICD   (rev 1.9 04/07/99)
        Hrc Data Products:     Level 1 to ASC Archive ICD   (rev 1.5  11-03-98)
        Grating Data Products: Level 1.5 to ASC Archive ICD (rev 1.6  05-31-99)
        Analysis Ref data for Exp Maps & Gratings L1.5/2    (rev 1.2  12-15-98)


DATE
----
     03-22-99
     04-29-99
     06-03-99
```

14 tgdetect.hlp

```
NAME

    tgdetect -- Run celldetect on central region of a grating
                event list; filter the file on snr threshold
                and snr ratio tolerance to weed out zero order
                detections.  Compare resultant file with reference
                source list to match source lists and append new
                ones found.

USAGE

    tgdetect infile OBI_srclist_file outfile
    [zo_pos_x] [zo_pos_y] [zo_sz_filt_x] [zo_sz_filt_y]
    [snr_thresh] [thresh] [findpeaks] [celldetect_log]
    [psftable] [fixedcell] [fixedcell_cc_mode] [expfile]
    [bkgfile] [bkgvalue] [bkgerrvalue] [eband] [eenergy]
    [snrfile] [convolve] [xoffset] [yoffset] [cellfile]
    [centroid] [kernel] [snr_ratio_limit] [setsrcid]
    [max_separation] [clobber] [verbose]

DESCRIPTION

    Tgdetect is a wrapper script which sets up and executes
    several DM tools, celldetect, and two grating specific
    tgdetect sub-tools.  The functionality can be broken down
    into the following sections:

    TG_FILT
    -------
    A box region about the center pointing position is created.
    This filter string is based on the input parameters:
    zo_pos_x, zo_pos_y, zo_sz_filt_x, zo_sz_filt_y.  When all
    four parameters are set to "default", the header of the
    event list is used to figure out the location of the
    pointing and the default size of the box, depending on the
    name of the instrument used.  The parameters zo_pos_x and
    zo_pos_y are the RA_NOM and DEC_NOM of the observation,
    translated into pixel coordinates.  The parameters
    zo_sz_filt_x and zo_sz_filt_y specify the dimentions of the
    box size, in the x and y directions.  The box size parameters
    specify the entire side of the bix, not it's radius values.
    The defaults for the box dimentions are set to 200
    pixels for ACIS and 1800 pixels for HRC (except if ACIS is in
    CC mode; the size of the box is 400x400 pixels).  This box region
    is fed into celldetect in order to detect at least the target
    zero order position.
    The user can choose to enter their own center position and
    box dimensions using these same 4 parameters.  If any of
    these parameters are not set to 'default', then the number
    values are interpreted and default values are not used.

      EX:  instrument = ACIS,  RA_NOM = # deg,  DEC_NOM = # deg
                     pix ra_nom = 4096.5, pix dec_nom = 4096.5
```

```
  zo_pos_x=default, zo_pos_y=default
  zo_sz_filt_x=default, zo_sz_filt_y=default
  creates filter:  file[EVENTS][bin x=3800:4200,y=3820:4180]

  EX:  if user specifies all the values:

  zo_pos_x=4000, zo_pos_y=4000
  zo_sz_filt_x=400, zo_sz_filt_y=360
  creates filter:  file[EVENTS][bin x=3800:4200,y=3820:4180]


CELLDETECT
----------
The 2D detection tool celldetect is executed on the section
of the event list designated in the filter above.  Celldetect
outputs all the center positions of all source detections in
that region.  For more information on celldetect algorithms
and parameters, see the celldetect.hlp file.

If no sources are found by celldetect, a blank table is
output (table contains header/columns but no rows).  Since
there are three possible outcomes to running tgdetect:
sources found, no sources found, error in processing;
the exit status of tgdetect is output into a file called
'tgdetect.exit'.  This file is used in pipeline processing
in order to find out how the tool exited.  If no sources
are found, the tool is still successful in running, but
further processing with a blank table is impossible; the
pipeline stops execution with a success status.  In the
other two cases, the pipeline continues to process the
sources found (or, ends with failure since tgdetect
failed to run successfully).

Since celldetect finds all 'sources' in the region of interest,
these sources must be filtered for zeroth order detections.
Most of the tgdetect processing steps below are filtering
steps to find zeroth orders sources by weeding out any
possible contaminants.


SOURCE SELECT SNR
-----------------
The output from celldetect is filtered by signal to
noise threshold.  The parameter snr_thresh is used as the
filter on the sources;  all sources less than snr_thresh are
filtered out of the source list.  If the snr filter was too
high and filtered out all of the sources, the filter is
cut in half and rerun on the celldetect output file.  This
is repeated until at least one source is found to be equal
to or greater than the current setting for the snr_thresh
level.  If no sources are found when snr_thresh = 10, then
the program exits.  It is assumed that no zeroth order
sources have been found when the snr off all detects is
that low.
```

```
If one or more sources are found, they are sorted in order of
highest to lowest signal to noise.


TGIDSELECTSRC
-------------
The snr selected and sorted source list is filtered once more
on snr_ratio_limit.  It this step uses the highest snr as the
reference snr for the creation of ratios.  A ratio of the
(source snr / highest snr) is calculated for each source;
the parameter snr_ratio_limit is the cut off limit as to
whether the source is left in the source list or filtered
out.  A snr_ratio_limit=0, allows for all source to match
the limit criteria; a snr_ratio_limit=1 allows only the
highest snr to match the criteria and filters out the rest
of the sources in the list.

EX:   snr_ratio_limit=0.5

      source # snr ratio include or exclude
      --------     -----    -----          ------------------
         1         100      1                  include
         2          80     0.8                 include
         3          50     0.5                 include
         4          49     0.49                exclude
         5          10     0.10                exclude

A source ID column gets added to the resultant ratio filtered
source list.  The highest snr gets id=1;  the id's increment
by 1, for each source in the list.

For more details on this section of the processing, see the
tgidselectsrc.hlp file.


TGMATCHSRC
----------
The detected and filtered sources can be matched to a
previous OBI's source list.  This allows the source ID's
to be matched to previous results, in case the SNR's were
different for same source detections.  If a reference OBI
source list is not provided, the output source list from
this tool is simply a copy of the input source list.

If sources are matched to another source list, the output
source list is a combination of the current source list
and the reference source list.  All the source in the
reference list are matched within the max_separation
tolerance.  All matched source are output; all non-matched
sources are appended to the end of the list.

If the reference OBI source list is empty (no rows), then
the current source list is copied as the output resultant
list.  No matching is performed since there is no
reference to match to.
```

```
    The matching algorithm and tool details can be found in
    the tgmatchsrc.hlp file.


PARAMETERS

    Since tgdetect uses sub-tools to do a lot of it's functionality,
    it's parameters are broken up into sections, depending on which
    tool that parameter belongs to.  Please see the sub-tool help
    file for more information on some of the parameters listed
    below.  The subtools are: celldetect, tgidselectsrc, tgmatchsrc.

    infile      [filename (root_evt1.fits)]

Input L1 event file or @list (root_evt1.fits or @file.lis).
        Please see "NOTE" section below if using @lists.

    OBI_srclist_file      [filename or "NONE"]

Input source position(s) file from previous OBI or
        @list or NONE.  This file is used as a reference source
        list, in matching the detected sources from the 'infile'.
        When this file is not provided, the input source list is
        not matched against a reference source list; the results
        pertain strictly to the infile.
        Please see "NOTE" section below if using @lists.

    outfile      [filename (root_src1a.fits)]

Output source position(s) file name.

    (zo_pos_x = default)     [string 'default' or number]

        Center X of grating zero order filter position in sky pixel
        coordinates.  If set to 'default', the nominal RA position
        is used from the header and converted to detector pixel
        location.

    (zo_pos_y = default)     [string 'default' or number]

        Center Y of grating zero order filter position in sky pixel
        coordinates.  If set to 'default', the nominal DEC position
        is used from the header and converted to detector pixel
        location.

    (zo_sz_filt_x = default)     [string 'default' or number]

        Size X of grating zero order filter box in sky pixels.
        If set to 'default', the instrument determines the size,
        currently, the two known default are:
ACIS => 200 pixels
ACIS with CC mode => 400 pixels
HRC  => 1800 pixels
        Please note that this parameter is the entire length
```

```
        of the filter, not the radius;  if set to 400 pixels,
        the size of the filter will be 400 pixels in the x
        direction.

    (zo_sz_filt_y = default)     [string 'default' or number]

        Size Y of grating zero order filter box in sky pixels.
        If set to 'default', the instrument determines the size,
        currently, the two known default are:
ACIS => 200 pixels
ACIS with CC mode => 400 pixels
HRC  => 1800 pixels
        Please note that this parameter is the entire length
        of the filter, not the radius;  if set to 400 pixels,
        the size of the filter will be 400 pixels in the y
        direction.

    (snr_thresh = 40)      [real]

        Signal to noise threshold level for zero order filtering.
        All sources detect below the snr_thresh will not be
        counted as zero order source detections and will be dropped
        from the source list.

    (clobber = no)      [boolean]

Overwrite existing output file or not?  If file is write
        protected and clobber=yes; program will exit.  If file
        exists and clobber=no, program will also exit.

    (verbose = 0)       [integer 0->5]

Verbose setting.  Setting to zero is the least amount of
        output to the screen while setting of 5 is the most.

     #
     #   The following parameters are from celldetect; please
     #   see the celldetect.hlp file for more information.
     #
       (thresh = )celldetect.thresh -> 4) celldetect source threshold
    (findpeaks = yes)            find local peaks for celldetect
(celldetect_log = )celldetect.log -> no) Make a celldetect log file?
     (psftable = )celldetect.psftable -> /vobs/ASC_BUILD/data/psfsize.fits) table of PSF size data, for celldetect
    (fixedcell = )celldetect.fixedcell -> 0) celldetect fixed cell size to use
    (fixedcell_cc_mode = 99)      celldetect fixed cell size to use for ACIS CC mode
      (expfile = none)           Exposure map file name, for celldetect
      (bkgfile = none)           Background file, for celldetect
     (bkgvalue = )celldetect.bkgvalue -> 0) Background count/pixel, for celldetect
  (bkgerrvalue = )celldetect.bkgerrvalue -> 0) Background error, for celldetect
        (eband = )celldetect.eband -> 1.4967) energy band, for celldetect
      (eenergy = )celldetect.eenergy -> 0.928) encircled energy of PSF, for celldetect
      (snrfile = none)           celldetect snr output file (for convolution only)
     (convolve = )celldetect.convolve -> no) use convolutions for celldetect
      (xoffset = INDEF)          Celldetect offset of x axis from optical axis
      (yoffset = INDEF)          Celldetect offset of y axis from optical axis
```

```
     (cellfile = none)            output cell size image file
     (centroid = yes)             compute source centroids in celldetection?
       (kernel = fits)            Output file format for celldetect

     #
     #   The following parameters are from tgidselectsrc; please
     #   see the tgidselectsrc.hlp file for more information.
     #

(snr_ratio_limit = )tgidselectsrc.snr_ratio_limit -> 0) Value of SNR ratio to use as lower limit
     (setsrcid = )tgidselectsrc.setsrcid -> yes) Set src ids in output file?

     #
     #   The following parameters are from tgmatchsrc; please
     #   see the tgmatchsrc.hlp file for more information.
     #

(max_separation = )tgmatchsrc.max_separation -> 1) Maximum allowed separation (arcsec) for sources to match


EXAMPLES

    The regression test for this tool can be executed in the
    tgdetect directory, by typing:

    > make test

    The input and output data in echo'ed to the screen.


    Examples of non regression data testing:

    using reference file as NONE
    > tgdetect infile=root_evt1.fits OBI_srclist_file=NONE
      outfile=root_src1a.fits

    using source list file as reference
    > tgdetect infile=root_evt1.fits OBI_srclist_file=root2_src1a.fits
      outfile=root_src1a.fits

NOTES
    The use of @files may be misleading;  only the first file
    in the @list is used in processing.  @list were added in for
    pipeline use of @lists.  There are logic problems in using
    more than one input file in the list;  therefore, only
    the first file is used in any of the input @lists.

DATE

    11/04/98
    03/19/99
    04/15/99
    06/03/99
```

15 tgextract.hlp

NAME

```
    tgextract -- creates 1D spectrum(a) table file(s) from the
                 L1.5 output event list.
```

USAGE

```
    tgextract  infile outfile_type outfile
    tg_srcid_list tg_part_list tg_order_list ancrfile
    respfile [backfile] [rowid] [grating_par] [bin_units]
    [min_bin] [max_bin] [bin_size] [num_bins] [tg_d_units]
    [min_tg_d] [max_tg_d] [extract_background] [min_bkg_tg_d]
    [max_bkg_tg_d] [clobber] [verbose] [mode]
```

DESCRIPTION

```
   Bin Level 1.5 events into counts histograms, one per each
   source, grating part, and order, as specified by input
   parameters.  Calculate an uncertainty array for each
   spectrum. Optionally bin a background region, and also
   calculate its uncertainty array.  Output an AXAF FITS
   Type II PHA spectrum file, or N number of Type I PHA
   files.  From the binning parameters, create and write
   region extensions for the source (and optional
   background) region(s).

   Detailed Description:

   This tool should serve both the pipeline and interactive needs.

   For pipeline usage, certain parameter defaults will be imposed;
   for which some external calculations may be needed to specify
   the parameters.

   Primary function is to bin the event list into one or more
   1-dimensional spectra.

   Each spectrum is for a given set of photon attributes:

        - source_id (if there are more than one in the file),
        - part of the spectrum (such as HEG, MEG, LEG, or Drake 4-7), and
        - diffraction orders (if ACIS is the detector).

   The preferred output is a Type II PHA file, in which there is a row
   for each attribute set, and the spectrum is an array column.  For
   simplicity, each array quantity (the spectra) stored in a column
   will have the same width (i.e., number of bins per spectrum).  This
   means that the grids are different for different orders, and also,
   may be truncated differently for different source positions.  No
   checking will be done in the pipeline to avoid truncation of
   off-axis sources.  User interactive processing will be required to
   recover those parts of the spectra from the Level 1.5 event list).

   NOTE:  Alternatively, a Type I PHA file may be written, in which each
```

```
          attribute set specifies a separate file, since the Type I
          specification only allows a single spectrum (plus associated
          uncertainty and grid arrays).

   The cross-dispersion region is specified by limits constant with
   wavelength (a rectangle),
   TBR: or via a table.  (A table input will allow variable
        width with energy and order, and the advantage is for keeping
        approximately a constant effective area for each bin (table
        creation is TBD). )
   Limits must be consistent with the pre-filtering mask applied in
   Level 1.5 processing (i.e., there won't be any photons with tg_d
   greater than some pre-specified amount).

   Background regions can be specified parallel to the spectrum or
   (TBD:  via a table).  For pipeline use, the backgrounds
   will be symmetric and have top-bottom regions combined (i.e., same
   absolute valued region in the cross-dispersion direction, and
   binned together, such as -1.0 to -0.1 mm in tg_d, and 0.1 to 1.0,
   binned into one background spectrum instead of two).

   Each extracted region is specified as a FITS Region table in the
   REGION extension of the output pha file.  In the case where the
   background is extracted, there are a total of three region sections
   per spectrum: the spectrum region (SOURCE), the background region
   above the spectrum (BAKCGROUND_UP) and the background below the
   spectrum region (BACKGROUND_DOWN).


PARAMETERS

    infile [filename]  (root_evt1a.fits)

        Input event file (output event file from L1.5 processing).
        File should contains an EVENTS extension and N REGION
        extension(s) which describes the grating events extration
        region, per OBI.

    outfile_type [pha_typeII]  (pha_typeI|pha_typeII)

        Ouput file type: pha_typeI (single spectrum of OGIP PHA TYPE I)
        or pha_typeII (multiple spectra of OGIP PHA Type II).

    outfile [filename or . or root]  (root_pha2.fits or root or '.')

        Output file name.  If outfile_type=pha_typeII, the user can enter
        the full output file name or '.'.  If the '.' is used, the input
        file name is used to create the output file;  if infile =
        root_extension.fits, then outfile will be root_pha2.fits.  If
        outfile_type=pha_typeI, the user can enter the rootname of the
        output file or '.'.  If root name is used, the output file will
        be named as:  root_<srcid><part><order>_pha2.fits;  if '.'
        is used, the input file name is used to create the output file:
        if infile = root_extension.fits, then outfile will be
        root_<srcid><part><order>_pha2.fits.
```

```
tg_srcid_list [all]  ('all'|string|@filename)

    The list of source orders to process.
    "all" will process all the sources id's listed in the REGION
    extension(s).  A comma separated string list can be used instead of
    the REGION:   "1,2,5,7".  Another option is to use an @file
    which contains a comma separated list or source id's to process.

tg_part_list [header_value]  (HETG|HEG|MEG|LETG|HESF|header_value)

    Grating parts to process: HETG, HEG, MEG, LETG, HESF,
    header_value.  'header_value' will take the input event file
    header value of the keyword GRATING.

tg_order_list [default]  ('default'|comma list|range list|@file)

    Grating diffraction orders to process: 'default', comma list,
    range list, @file.
     - "default" is set to process the following:
       if ACIS:  1, 2, 3, -1, -2, -3
       if HRC:   -1, 1
     - a comma list is a comma separated string list of the orders
       the user wants to process, ie:
       "-5, -1, 1, 3"
     - a range list sets the min and max of the orders to process;
       all the orders in between, will be processed, ie:
       "-1..5"  will do orders from -1 to +5th order (excluding 0)
       a range list can be mixed with comma separated list
     - @file is a pointer to an ascii file which contains a comma
       separated list and/or range list of the orders to process

ancrfile [none]  (filename or 'none')

    Input ancillary response file name.  The filename or the word
    'none' gets written to output file in the ANCRFILE keyword.
    If file is provided, it is not applied to the data.

respfile [none]  (filename or 'none')

    Input redistribution file name.  The filename or the word
    'none' gets written to output file in the RESPFILE keyword.
    If file is provided, it is not applied to the data.

(backfile = "") [filename]

    If BACKFILE keyword is to be filled in, enter name here;
    otherwise, the keyword is not present in the output file.
    If file is provided, it is not applied to the data;
    however, background extraction of is not done, in this case.

(rowid = "") [string of length 64]

    If ROWID keyword is to be filled in, enter name here;
    otherwise, the keyword is not present in the output file.
```

```
(grating_par = grating) [param name]

    Name of the grating parameter file with grating characteristics.

(bin_units = angstrom) [angstrom|eV|KeV|degrees|pixel|mm]

    Bin units (for bin parameters below): angstrom, KeV, eV, degrees,
    pixel, mm.  Specifies the units of min_bin, max_bin, bin_size and
    of the output histogram.

    If units are Energy, then the tg_lam values in the event file must
    be converted to energy via E = hc/lambda, where hc=12398.54
    (hc=12.3985 for KeV) for lambda in angstroms and E in eV.

    If other bin parameters are left to defaults (angstroms), and
    bin_units is not angstrom, then binning parameters are converted
    to the specified unit and events are binned accordingly.

NOTE: Binning Parameters
------------------------
If the grating type is HETG, the binning parameters should be
given for MEG +1 order binning.  The bin characteristics
will be updated for all other orders and grating parts, using
the the following calculations:

        m * lambda = P * sin(theta)

    order * wavelength = period * sin(diffraction angle)

 Diffraction angle == tg_r is small, so sin(theta) ~ theta, so

        m * lambda = P * tg_r

 Calculating a bin size is done by taking the derivative:

        m * delta_lambda = P * delta_tg_r

        or equivalently

        delta_lambda  = (P/m) * delta_tg_r.

 If bins are specified for first order MEG and, the rest are
 scaled to:
            (Period_part/Period_MEG) / order

 If the grating is LETG (and detector is ACIS), the user can
 specify the binning for LETG +1, and the rest of the binning
 will be scaled by order only, as apposed to using the Period
 ratio's above.

(min_bin = compute) ['compute'|real number]

    Minimum dispersion coordinate on which to start the binning.
    A real valued number can be specified in any other diffraction
```

```
      coordinate: max energy [keV or eV], tg_r [degrees], gdp [pixel],
      or linear [mm].  The default is:  1.239854 angstrom (10.0 keV).

      It is assumed that the bin coordinate refers to the left edge
      of the bin, and the bin value is incremented if the data value
      is GE the bin coordinate, and LT the bin coordinate plus the
      bin_size.

      Min_bin could be overspecified:
         min_bin = max_bin - num_bins * bin_size
         A value of 'compute' means calculate using a combination of
         the bin values entered and the default values.

  (max_bin = compute) ['compute'|real number]

      Maximum dispersion coordinate on which to stop the binning.
      A real valued number can specified in any other diffraction
      coordinate: max energy [keV or eV], tg_r [degrees], gdp [pixel],
      or linear [mm].

      Default: Determined by grating type and part:
         HETGS/HEG: 20 A
 HETGS/MEG: 40 A
 LETGS:    170 A
         LETG/ACIS-S: 100 A
         HETG/HEG/HRC-S:  30 A
         HETG/MEG/HRC-S:  60 A

      It is assumed that the max_bin coordinate refers
      to the right edge of the bin, and the bin value is incremented if
      the data value is GE the left edge (the bin coordinate) and LT
      the left edge plust the bin_size (right edge).

      Max_bin could be overspecified:
         max_bin = min_bin + num_bins * bin_size
         A value of 'compute' means calculate using a combination of
         the bin values entered and the default values.

  (bin_size = compute) ['compute'|real number]

      This is the size of the bins for storing the counts
      histogram.  It is assumed that all bins are the same size, and
      that the bin coordinate refers to the left edge of the bin.

      Default:  Determined by grating type, part, and order, for
                0.012 mm linear scale (one-half spatial resolution).

              Wavelength bin is equal to:
                   TG_Period * linear_scale / Rowland_Spacing / tg_m

              First-order nominal size for 0.012 mm (acis):
                HETGS/HEG: 2.78e-03 A
                     /MEG: 5.56e-03 A
                LETGS    : 1.38e-03 A
```

```
            Bins may be specified in other diffraction
            coordinates:  energy [keV or eV], tg_r [degrees],
            gdp [pixel], or linear [mm].

      Bin_size could be overspecified:
         bin_size = (max_bin - min-bin) / num_bins
         A value of 'compute' means calculate using a combination of
         the bin values entered and the default values.

   (num_bins = compute) ['compute'|integer]

      The number of bins for the resultant spectra.  For now, the
      default values are (max_bin-min_bin)/bin_size.

      Default:  Determined by grating type and detector.  Nominal
            configurations:

          HETGS:  6972 (default bin_min, bin_max, bin_size)
          LETGS: 12268 (default bin_min, bin_max, bin_size)

      The number of bins is possibly overspecified. Min_bin, max_bin,
      and bin_size completely specify the number of bins.  The
      'compute' option is for for calculating the number of bins:
         num_bins = (max_bin - min_bin) / bin_size

   (tg_d_units = degrees) [degrees|arcsec|mm|pixel]

      Tg_d bin units (for tg_d parameters below): degrees, arcsec, mm,
      pixel.  Specifies the units of min_tg_d, max_tg_d,
      min_bkg_tg_d and max_bkg_tg_d.

      Only degrees are currently implemented in the tool, since tg_d
      is in degrees in the input file.

      NOTE: the following is a representation of the tg_d parameters:

        -----------------------------------  (+1.0)*(|max_bkg_tg_d|)
                                                       =~+1.0mm
          the high-side background region
  tg_d
   ^
   |      -----------------------------------  (+1.0)*(|min_bkg_tg_d|)
    +0.1  -----------------------------------    max_tg_d
     0.0       the "source" region  (the "prime" area) +----> tg_lam
    -0.1  -----------------------------------    min_tg_d
          -----------------------------------   (-1.0)*(|min_bkg_tg_d|)


         the low-side background region

        ------------------------------------   (-1.0)*(|max_bkg_tg_d|)
                                                       =~-1.0mm

        The region between min_tg_d and max_tg_d is the area when
        the data spectrum is extracted.  The area between
```

```
          (|min_bkg_tg_d| and |max_bkg_tg_d|) && (-1.0*min_bkg_tg_d
          and -1.0*|max_bkg_tg_d|) is the background spectrum.  Note
          that the top and bottom background sections get added together
          to create one background spectrum.

    (min_tg_d = compute) ['compute'|real number]

      Specifies the minimum tg_d range to include in the histogram.
      Units are specified by tg_d_units.
      Default:  Determined by configuration:

            HETGS: -0.1 mm, == -2.39 arcsec, == -6.64E-04 degrees
            LETGS: -0.2 mm, == -4.78 arcsec, == -1.33E-03 degrees

    (max_tg_d = compute) ['compute'|real number]

      Specifies the maximum tg_d range to include in the histogram.
      Units are specified by tg_d_units.
      Default:  Determined by configuration:

            HETGS: +0.1 mm, == 2.39 arcsec, == 6.64E-04 degrees
            LETGS: +0.2 mm, == 4.78 arcsec, == 1.33E-03 degrees

    (extract_background = yes) [boolean]

        Extract the local background spectrum?  If set to yes, the
        background spectrum will be in the output file(s).  The
        parameters min_bkg_tg_d and max_bkg_tg_d are used to
        determine the size of the background spectrum.  If set to
        NO, then the output column BACKGROUND is not present in
        the output file(s).

    (min_bkg_tg_d = compute) ['compute'|real number]

       Specifies the minimum absolute value of tg_d for a background
       region.  The default starts where the spectral region ends
       (at max(|min_tg_d| || |max_tg_d|)).
       Events with cross-dispersion from the background region will
       be binned into a background spectrum.  The units are to be
       specified by tg_d_units.
       Default: Depends on configuration:

           min_bkg_tg_d = max(|min_tg_d| || |max_tg_d|))

    (max_bkg_tg_d = compute) ['compute'|real number]

       Specifies the maximum absolute value of tg_d for a background
       region.  The default values are listed below;  however, if the
       set defaults are still within the non-background tg_d range,
       then, the max_bkg_tg_d is set to 10 times
       max(|min_tg_d| || |max_tg_d|).
       Events with cross-dispersion from the background region
       will be binned into a background spectrum.  The units are to be
       specified by tg_d_units.
       Default: Depends on configuration:
```

```
           HETGS: +/-1.0 mm == +/-23.9 arcsec, == 6.64E-03 degrees
           LETGS: +/-2.0 mm == +/-47.8 arcsec, == 1.33E-02 degrees

    (clobber = no) [boolean]

If set to yes, existing output files will be overwritten.

    (verbose = 0) [integer 0:5]

Program verbosity.  0 means display no messages to the screen
        while running.  5 means report maximum on what the program is
doing.

    (mode = "ql") [string]

Parameter interface mode.  "ql" stands for "query and learn",
        which prompts you for input and updates
     "tgextract.par" with your responses.


EXAMPLES

> tgextract infile=AS_004_evt1a.fits outfile_type=pha_typeII
            outfile=AS_004_pha2.fits tg_srcid_list=1
            tg_part_list=header_value tg_order_list=1 ancrfile=none
            respfile=none backfile="" rowid="" grating_par=grating
            bin_units=Angstrom min_bin=compute max_bin=compute
            bin_size=compute num_bins=compute tg_d_units=deg
            min_tg_d=default max_tg_d=default extract_background=yes
            min_bkg_tg_d=default max_bkg_tg_d=default  clobber=yes

  The output Type II pha file is called AS_004_pha2.fits, and can be
  viewed by any ftools and dmtools.  Here's a small section of the
  output file:

   EXTNAME=SPECTRUM
   ----------------
EXTNAME = 'SPECTRUM'           / name of this binary table extension
HDUNAME = 'SPECTRUM'           / ASCDM block name
CONTENT = 'SPECTRUM'           / File contains GTIs and events
HDUCLASS= 'OGIP    '
HDUCLAS1= 'SPECTRUM'
HDUCLAS2= 'TOTAL   '
HDUCLAS3= 'COUNTS  '
HDUCLAS4= 'TYPE:II '
HDUCLAS5= 'TG      '
HDUVERS = '1.0.0   '
HDUVERS1= '1.0.0   '
BACKFILE= 'none    '
CORRFILE= 'none    '
RESPFILE= 'none    '
ANCRFILE= 'none    '
AREASCAL=        1.0000000E+00
BACKSCAL=        1.0000000E+00
```

```
CORRSCAL=        1.0000000E+00
CHANTYPE= 'PI      '
POISSERR= 'T       '
QUALITY =                    0
GROUPING=                    0

        SPEC_NUM TG_M   TG_PART TG_SRC_ID

      1      1        1      1       1
      2      2        1      2       1

        CHANNEL COUNTS STAT_ERR        BACKGROUND BIN_LO
                counts counts          counts     Angstrom
      1      1       0   1.8660254E+00      0       6.199579855050732E-01
             2       0   1.8660254E+00      0       6.227381244564402E-01
             3       0   1.8660254E+00      0       6.255182634078071E-01
      ...
      2      1       0   1.8660254E+00      0       1.239854000000000E+00
             2       0   1.8660254E+00      0       1.245414000000000E+00
             3       0   1.8660254E+00      0       1.250974000000000E+00
      ...

        BIN_HI
        Angstrom
      1   6.227381244564402E-01
          6.255182634078071E-01
          6.282984023591741E-01
      ...
      2   1.245414000000000E+00
          1.250974000000000E+00
          1.256534000000000E+00
      ...

   EXTNAME=REGION
   --------------

       SPEC_NUM

      1      1
      2      1
      3      1
      4      2
      5      2
      6      2

        ROWID

      1 SOURCE
      2 BACKGROUND_UP
      3 BACKGROUND_DOWN
      4 SOURCE
      5 BACKGROUND_UP
      6 BACKGROUND_DOWN

        SHAPE            TG_LAM          TG_D            R
```

```
                        angstrom        degrees         (angstrom , degrees)
      1 box               1.0310479E+01   0.0000000E+00   1.9381042E+01
                                                          1.3277777E-03
      2 box               1.0310479E+01   3.6513889E-03   1.9381042E+01
                                                          5.9750001E-03
      3 box               1.0310479E+01   3.6513889E-03   1.9381042E+01
                                                          5.9750001E-03
      4 box               2.0619926E+01   0.0000000E+00   3.8760147E+01
                                                          1.3277777E-03
      5 box               2.0619926E+01   3.6513889E-03   3.8760147E+01
                                                          5.9750001E-03
      6 box               2.0619926E+01   3.6513889E-03   3.8760147E+01
                                                          5.9750001E-03

        ROTANG          COMPONENT INCLUDE TG_SRCID TG_M
        degrees
      1   0.0000000E+00      1         1       1        1
      2   0.0000000E+00      1         1       1        1
      3   0.0000000E+00      1         1       1        1
      4   0.0000000E+00      2         1       1        1
      5   0.0000000E+00      2         1       1        1
      6   0.0000000E+00      2         1       1        1
```

DATE

08/10/98
09/16/98
03/19/99
04/01/99
06/04/99