rgssources

August 16, 2005

Abstract

The task constructs a list of sources that are to be processed by RGS pipeline.

1 Instruments/Modes

Instrument	Mode
RGS	SPECTROSCOPY

2 Use

pipeline processing	yes
interactive analysis	yes

3 Description

3.1 Introduction.

This task creates the RGS source list. This source list is used as the main reference for the definition of selection regions by **rgsregion** and for the creation of the response matrix by **rgsrmfgen**. For the latter, knowledge of the source position is required in order to correctly determine the wavelength scale. In pipeline use, the RGS source list is used to indicate those sources for which products should be extracted.

3.2 Environment variables.

In nearly all cases, **rgssources** requires information from the CCF. (The exceptions are trivial cases such as running with `filemode`='modify' but with both `addusersource` and `withepicset` = 'no'.) Because the exceptions are insignificant and involve complicated logic to disentangle, it has been made mandatory to supply a CCF to the task. The location of the CCF is pointed to in the usual way, ie via the environment variable SAS_CCF.

The CCF interface needs to know whether the instrument is RGS1 or RGS2 in order to provide the correct boresight etc. Although the `instexpid` parameter has not been made mandatory, the task will

fail with an error if the information is not ultimately available. See the algorithm section 7 for details of the way the task attempts to deduce the instrument.

The task may also require information from the ODF summary file, in which case the variable SAS_ODF must be set before running the task. This information may be necessary for one of two reasons:

- The attitude must be calculated.
- It is desired to write observation-specific keywords to the source list header.

Every attempt has been made to avoid this requirement where possible and thus cater to the user who does not have the entire ODF at their fingertips.

3.3 Where the data comes from.

The source data can come from several sources:

- A source list from a previous run of **rgssources** (note that from version 5.1, **rgssources** is now compatible with all earlier source list formats).
- The proposed target source.
- The attitude of the spacecraft.
- A source list output by either **emldetect** or **eboxdetect**.
- A source position supplied on the command line by the user.

These are described individually below.

3.3.1 Previous RGS source list.

The task **rgssources** can either modify a previously-created source list or create a new one, depending on the value of the parameter `filemode`. If `filemode`='create', the name of the newly created file is read from parameter `srclist`. If on the other hand `filemode`='modify', the name of the existing file to be edited must instead be supplied to `srclist`. Note that if `filemode`='create' and `srclist` gives the name of an existing file, this existing file may (subject to the SAS_CLOBBER setting) be completely overwritten.

Although sources are never deleted from previous lists, some of the column entries may be altered. Older-format source tables are automatically converted to conform to the current format. (Note however that NO other extensions to the source list file are upgraded in format by **rgssources**.)

If `filemode`='create', the name of the RGS instrument must be supplied via the parameter `instexpid`. If on the other hand `filemode`='modify', the instrument designation is read from header of the old file.

Note that all sources must originally have come from one of the following three categories. Their labels (see description of column `LABEL`, section 5) will reflect that fact.

3.3.2 Proposal and Attitude sources.

These are now added automatically by the task and there are no longer any command-line parameters that relate to them directly. If a previous RGS list is supplied (ie `filemode`='modify'), the proposal source from that file is used; otherwise this information is read from the ODF. If a previous RGS list is supplied, and `changeattitude`=no, the attitude source from that file is used; otherwise the entries from this source are calculated from those parameters relating to the attitude (see subsection 3.5).

The respective values of the `LABEL` column for the proposal and attitude sources are 'PROPOSAL' and 'ONAXIS'.

3.3.3 EPIC source list.

A source list compiled by either **emldetect** or **eboxdetect** can be added to the RGS source list. This is accomplished by setting `withepicset`=yes and giving the name of the file via `epicset`.

These EPIC source lists include positions and count rates for each source within several different energy bands. However the RGS cameras only have significant sensitivity in band 2 (500-2000 eV). Therefore only EPIC sources which have `ID_BAND`=2 are nowadays copied to the RGS list. Older-format lists may contain just band 1 sources, or epic sources from all bands.

More than one EPIC source list can be added to the RGS list by running **rgssources** several times in 'modify' mode on the same RGS set but with different EPIC sets as input.

The values of the `LABEL` column of the RGS source list must be unique. For EPIC sources this is partially achieved by including the value of the EPIC source list column `ML_ID_SRC` or `BOX_ID_SRC` in the label for that source. These column values are unique within the EPIC source list itself. However this number becomes non-unique when several EPIC lists are included in the one RGS file. To retain uniqueness of the `LABEL` value, another parameter is provided: `epiclabelprefix`. The `LABEL` value is constructed from the string supplied via this parameter (default value is 'EPIC'), followed by the digits of the `ML_ID_SRC` or `BOX_ID_SRC` value. Note that all `LABEL` values are converted to upper case.

3.3.4 User-supplied source.

The user can supply the position, brightness and label of a source. The position can be specified in either of two styles, controlled via the parameter `positionstyle`: if `positionstyle`='radec' (the default), the source J2000 right ascension and declination can be entered (in decimal degrees) via the parameters `ra` and `dec`; if `positionstyle`='wrtatt', the source position with respect to the spacecraft pointing direction, in the cartesian dispersion/cross-dispersion basis, can be entered via the parameters `deltadisp` and `deltaxdsp`. The units of the latter two are arcminutes.

The estimated brightness in counts per second can be entered via the parameter `rate`; the user source label via `label`.

More than one user source can be added to the RGS list by running **rgssources** several times in 'modify' mode on the same RGS set, with `addusersource`=yes each time. Note that a different value of `label` must be chosen each time, in fact `label` must be different to any of the values in column `LABEL` of the old list.

Note that all `LABEL` values are converted to upper case, hence a source with `label` given as 'Fred' will conflict with a previous source supplied with `label`='fred' since they each get translated to 'FRED'.

3.4 Prime source.

The prime source is used in the pipeline by **rgsangles** as a reference to calculate approximate corrections for attitude variation errors in the BETA values of events. **rgssources** always selects one member of its output list to be the prime source and stores the INDEX column value of this source in the source-table keyword PRIMESRC.

rgssources also calculates a confusion value between every other source and the prime source (see subsection 5.2); these values are stored in the column CONFUSION.

The prime source can be designated in five ways, via a number of different parameters:

- The previous value can be retained.
- The user-supplied source can be designated as the prime source.
- The brightest source can be designated as the prime source.
- The prime source can be designated via one the parameters primelabel, primeindex or primeexpression, given the appropriate setting of primestyle.
- The task can be allowed to choose the best candidate.

To see how these interact in detail, see the algorithm in section 7, also the examples in section 6; however the order of precedence is roughly as the entries appear in the list above.

3.5 Attitude parameters.

The attitude is defined as the pointing direction plus the position- or roll-angle of the spacecraft. It is necessary to define the attitude for **rgssources** because the task must calculate values for the columns DELTA_DISP, DELTA_XDSP, FOV_PHI and FOV_R, which record the source positions with respect to the attitude.

In the case that filemode='modify', the attitude is left by default at the value in the previous file. To alter the attitude stored in the file while in this filemode, the parameter changeattitude must be specifically set by the user to 'yes'. Assuming that this is done, the attitude can then either be calculated (attstyle='start', 'mean', 'median' or 'expmedian') or input directly by the user (attstyle='user', then supply attra, attdec and attapos). The four types of calculation are briefly described as follows:

- attstyle='start': the attitude at the start of the exposure is used.
- attstyle='mean': the mean value of the attitude (as sampled in the attitude history file) is used. This value is read from the output file from **atthkgen**.
- attstyle='median': the median value of the attitude (as sampled in the attitude history file) is used.This value is read from the output file from **atthkgen**.
- attstyle='expmedian': the task **attfilter** filters the attitude data such that only time values within the exposure duration remain. This task also calculates the usual set of means, medians etc for thsi restricted set of attitude data. With attstyle='expmedian', **rgssources** reads the median value of the attitude during the exposure from the output file from **attfilter**.

3.6 Upgrading the format.

rgssources is now backward-compatible, ie it can read source lists which were created in the pre-sas-5.1 format. The task attempts to upgrade the format of these older files and in fact can be used to do this exclusively, without the necessity to add to or otherwise modify the source list. The upgrading is performed by a public subroutine called convertOldFiles located in the module rgssources_update.f90. The interface is as follows:

```
subroutine convertOldFiles(setName, writeObsKwds, writeExpKwds&
, instrumentId, expId)

  implicit none

  character(*),   intent(in)           :: setName
  logical,        intent(in), optional :: writeObsKwds, writeExpKwds
  integer(int32), intent(in), optional :: instrumentId, expId
end subroutine convertOldFiles
```

The algorithm is described in section 7.

The upgrade consists of the the following steps:

- The source list table is renamed from SOURCES to SRCLIST.
- The attitude keywords RA_REF, DEC_REF and APOS_REF are moved from the source table to the primary header, converted from radians to decimal degrees, and renamed RA_PNT, DEC_PNT and PA_PNT.
- The INSTRUME keyword is written.
- Standard values are written to the TELESCOP, FILTER, ORIGIN, RADECSYS and EQUINOX keywords.
- (Optional) values are written to the OBS_MODE, OBS_ID, OBJECT and OBSERVER keywords. These require SAS_ODF to be set.
- (Optional) values are written to the DATAMODE and EXPIDSTR keywords. These require SAS_ODF to be set.
- The column names are converted.
- The column values are converted. See notes below.
- If there were sources of epic origin in the old-format list, the appropriate sets of E_EXPRnn, E_CONTnn, E_BANDnn and E_FILTnn keywords are written. These all have value 'UNKNOWN' except E_BANDnn.

Notes on the column value conversion:

- INDEX and RATE values are retained.
- Sources can be divided into those for which the coordinates are fixed with respect to the celestial sphere and those which are fixed with respect to the reference attitude. (Since the latter may take on slightly different values, depending on how it is calculated, the two are

not quite the same thing.) In older-format files, the proposal and all epic and user sources fell into the former category and the onaxis and offaxis sources into the latter. The former sources have FIXED_ON_SKY set to true, vice versa for the latter. RA and DEC values are retained for all the FIXED_ON_SKY sources, and DELTA_DISP, DELTA_XDSP, FOV_PHI and FOV_R are recalculated; for the non-FIXED_ON_SKY sources it is the other way around.

- The LABEL values of the proposal and onaxis sources are uppercase-converted respectively to 'PROPOSAL' and 'ONAXIS'.
- For epic sources, ID_BAND and ML_ID_SRC are used to construct (hopefully) unique LABEL values, of the form 'OLDEPICbbnnnnn', where bb and nnnnn are respectively the ID_BAND and ML_ID_SRC numbers. A tally is kept of the number of different ID_BAND numbers encountered and these are used to assign values to the EPIC_FILE column of these sources.
- Offaxis sources are given LABEL values of 'OFFAXISnnnnn' where nnnnn is the number of offaxis sources so far detected in the file.
- Sources which have old LABEL values which are neither 'proposal', 'onaxis', 'epic' or 'offaxis' are presumed to be user-added sources. Their LABEL values are just converted to upper-case. Note however that **rgssources** will fail with an error if non-unique labels are detected.
- A non-zero value of SRC_SELECT is translated into a true value of PROCESS.
- A non-zero value of BACK_SELECT is translated into a true value of BKG_EXCLUDE.
- CONFUSION is calculated from scratch.

The prime source is left untouched (although **rgssources** will fail if its INDEX value does not exist in the file). No other extensions of the source list file are altered or deleted during this upgrade process.

4 Input Files

1. EPIC sources set with a binary extension table named SRCLIST (required only if withepicset = 'yes').

   The following columns need to be present in this table:

   - RA: this value is copied into the RGS column of the same name.
   - DEC: this value is copied into the RGS column of the same name.
   - ID_BAND: this value is tested, only those EPIC sources with ID_BAND=2 being transcribed to the RGS list.
   - ML_ID_SRC (if the source list was made by **emldetect**) or BOX_ID_SRC (if the source list was made by **eboxdetect**): this number is included in the LABEL value of the source in the RGS list.
   - RATE: this value is copied into the RGS column of the same name.
2. RGS sources set as described in the 'Output files' section (required only if filemode = 'modify').
3. The attitude history file created by **atthkgen** (required only if ((filemode = 'modify' and changeattitude = 'yes') or filemode = 'create') and attstyle = 'mean' or 'median'.).
4. The filtered attitude history file created by **attfilter** (required only if ((filemode = 'modify' and changeattitude = 'yes') or filemode = 'create') and attstyle = 'expmedian'.).

5 Output Files

1. RGS sources set with a binary extension table named SRCLIST. The header has all the keywords mandatory for PPS products, in particular

   - RA_PNT: The right ascension of the attitude in decimal degrees.
   - DEC_PNT: The declination of the attitude in decimal degrees.
   - PA_PNT: The position angle of the attitude in decimal degrees.

   The SRCLIST table has the following keywords:

   - PRIMESRC: The INDEX value (see column description below) of the prime source.
   - E_EXPRn: There are n occurrences of this keyword, one for each EPIC source list added to the RGS list. The numbers n are consecutive, starting at 1. The values of these keywords are the exposure IDs of the respective EPIC source files.
   - E_CONTn: Similar to the E_EXPRn keyword, but this records the value of the CONTENT keyword in the EPIC file header.
   - E_BANDn: Similar to the E_EXPRn keyword, but this records the value of ID_BAND used to select the EPIC sources. This is presently equal to 2 but could change in the future.
   - E_FILTn: Similar to the E_EXPRn keyword, but this records the value of the FILTER keyword in the EPIC file header.

   The SRCLIST table has the following columns:

Column name:	Data type:	Description:
INDEX	int16	Source index number. Each source has a unique value, which **rgssources** never alters.
LABEL	string	Label for the source. These values are also unique to each source. Only upper case is used. At present, label values can only be 20 characters or less in length. Trailing spaces are not allowed.
RA	real32	J2000 right ascension in decimal degrees.
DEC	real32	J2000 declination in decimal degrees.
RATE	real32	Counts per second.
DELTA_DISP	real32	Offset on the sky, in the dispersion direction, of the source with respect to the pointing direction. Given in arcminutes.
DELTA_XDSP	real32	Offset on the sky, in the cross-dispersion direction, of the source with respect to the pointing direction. Given in arcminutes.
FOV_PHI	real32	This and the next column give the polar coordinates of DELTA_DISP and FOV_PHI. Units for both are decimal degrees. FOV_PHI is the angle of the source position from the -ve dispersion axis towards the +ve cross-dispersion axis.
FOV_R	real32	
CONFUSION	real32	This is a measure of how confused the source is with respect to the prime source. See subsection 5.2 for a description of how it is calculated. It is a dimensionless number.
PROCESS	bool	This column is used by **rgsregions** to flag those sources for which spectrum extraction regions should be calculated. This column is no longer set by **rgssources**, so all values are written as false.
BKG_EXCLUDE	bool	This column is used by **rgsregions** to flag those sources which should be excluded from the background spectrum extraction region. This column is no longer set by **rgssources**, so all values are written as false.
FIXED_ON_SKY	bool	This column flags those sources for which the positional information was derived from right ascension and declination. The only sources for which FIXED_ON_SKY is false are the attitude source and any user source supplied with userstyle='wtatt'.
EPIC_FILE	int16	This gives the number of the E_EXPRn, E_CONTn, E_BANDn and E_FILTn keywords appropriate to the source if it has been derived from an EPIC source list. Eg, for EPIC_FILE=3, the details of the original list from which this source came can be found from the keywords E_EXPR3, E_CONT3, E_BAND3 and E_FILT3.

Note that the RGS source list set is also used to store the spectrum extraction regions created by **rgsregions**. These become invalidated if the attitude is altered; in this case **rgssources** deletes them. See the algorithm (section 7) for details of the circumstances under which this occurs.

The RGS source list table is required to have 1 source whose position is taken from the observation proposal, and 1 source whose position is equal to the RGS attitude (stored in the dataset header keywords RA_PNT, DEC_PNT and PA_PNT). The LABEL values of these two sources are PROPOSAL and ONAXIS respectively.

5.1 Uniqueness of LABEL and INDEX values.

These column values are maintained unique in order to provide the user with convenient ways to select either the prime source or (via the selection expression parameters of **rgsregions**) a subset of sources for which spectrum-extraction regions are desired. When a previously-made RGS source list is modified, the task checks to make sure the labels and index values in this list are unique. (They are left so by any previous invocation of **rgssources**, but values might be changed in the interim by use of the appropriate ftool, for example.)

Further added sources from either a submitted EPIC list or the user are checked one by one against all previous LABEL values in the list before themselves being added to it. An INDEX value equal to the next highest unused value is given to each added source.

If a source is found to have the same LABEL value as an existing member of the source list, the action taken by the task depends on the value of the parameter clobberonlabel. If clobberonlabel=no (the default), the task fails with an error. Otherwise it discards the newer source, with a warning.

5.2 CONFUSION values.

It is intended to flag in the RGS pipeline sources which are confused with the prime source, so as to allow the end user to assess whether these have a deleterious effect on the spectra of the prime source. For this purpose **rgssources** calculates, for each source, a value which is stored in a column called CONFUSION. This value is calculated from the following formula:

$$C = I_{\rm prime}^{-1}\Big\{p_1 + p_2\, I\, \exp\Big[-\frac{(|\alpha-\alpha_{\rm prime}|+a)^2}{2b^2} - \frac{(|\chi-\chi_{\rm prime}|+c)^2}{2d^2}\Big]\Big\}.$$

Here C is the confusion value, I is the source brightness (ie value of the RATE column) and α and χ are respectively the dispersion and cross-dispersion angles in arcseconds. The constants p_1, p_2, a, b, c and d will eventually be stored in CCF MiscData, but at present are 'hard-wired' into the task code, with values as follows:

$$\begin{aligned} p_1 &= -0.00653\ \text{cts/sec} \\ p_2 &= 1.01 \\ a &= 36.9\ \text{arcsec} \\ b &= 593.4\ \text{arcsec} \\ c &= 2.29\ \text{arcsec} \\ d &= 23.3\ \text{arcsec} \end{aligned}$$

6 Examples.

Before running **rgssources** you will need to set SAS_CCF. SAS_ODF may also be required (see below for details).

There are essentially three reasons for invoking **rgssources**:

- To create a new source list (eg when invoked by either the Pipeline Control and Monitoring System (PCMS) or by **rgsproc**).

- To update the format of a pre-sas-5.1 source list.
- To add further sources to an existing source list.

Examples of each are detailed below.

6.1 Pipeline useage.

rgssources is usually first invoked within the PCMS or within **rgsproc**. In PCMS usage the command should be

```
  rgssources filemode=create srclist=rgssrclist.ds<or whatever>
atthkset=<att hist file> enablefilter=yes withepicset=yes
epicset=<epic source list> instrument=<rgs1 or 2> exposure=<exposure number>
primestyle=auto
```

`enablefilter` ensures that only epic sources within the RGA filed of view are selected; `primestyle`=auto forces **rgssources** to choose a prime source within the field of view. At present, given `primestyle`=auto, the proposal source is chosen if it is within the field of view; the brightest in-field epic source if not; with the pointing position as a final fallback.

6.2 Upgrading the format.

rgssources is now backward-compatible: it is able to add sources to source lists which were created in the pre-sas-5.1 format. See section 3.6 for a detailed description of the process. The task attempts to upgrade the format of these older files and in fact can be used to do this exclusively, without the necessity to add to or otherwise modify the source list. To fully upgrade an older source list, **rgssources** needs the following information:

- The instrument name.
- Information from the CCF.
- The exposure number.
- Information from the ODF.

The first two items are mandatory; the second two optional. Details of this are as follows:

6.2.1 Instrument name.

If the file name is in either the PPS or the **rgsproc** formats, then **rgssources** is able to deduce the instrument name from the file name. The PPS format is PooooooooooiiteeeSRCLI_0000.FIT; the **rgsproc** format is rrrroooooooooiiteee00_sources.ds. In these templates, rrrr is the rev number, ooooooooo is the observation identifier, ii the instrument (either R1 or R2), eee the exposure number and t the exposure type (either S or U)). However forcing **rgssources** to do this is not the preferred option, which is rather to give the instrument name directly via the parameter `instexpid`.

6.2.2 CCF.

To access this, the user should set SAS_CCF in the usual way.

6.2.3 Exposure number.

rgssources can also deduce the exposure number from the standard file name formats described above. However this is not the preferred option, which is to give the exposure number directly via the parameter `instexpid`. However, the information is optional. If the file name is not in a standard format, and the exposure number is not otherwise known, its use can be turned off by setting `writeexpkwds` to 'no' or 'false'. In this case the only effect is that some exposure-specific keywords are not written to the file header. (Note that the exposure number is required if it is desired to recalculate the attitude.)

6.2.4 ODF.

To access this, the user should set SAS_ODF to point to the ODF in question. However, the information is optional. If the user does not have access to the ODF, it can be rendered unnecessary by setting `writeobskwds` to 'no' or 'false'. In this case the only effect is that some observation-specific keywords are not written to the file header. (Note that ODF access is required if it is desired to recalculate the attitude.)

6.2.5 Example calls.

Full upgrades:

```
  setenv SAS_CCF <the .cif file>
  setenv SAS_ODF <the ODF directory>

  rgssources srclist=PooooooooooiiteeeSRCLI_0000.FIT
```

or

```
  setenv SAS_CCF <the .cif file>
  setenv SAS_ODF <the ODF directory>

  rgssources srclist=rrrrooooooooooiiteeee00_sources.ds
```

or

```
  setenv SAS_CCF <the .cif file>
  setenv SAS_ODF <the ODF directory>

  rgssources srclist=<non-standard name> instrument=<rgs1 or rgs2>
exposure=<exp num>
```

Just format upgrade without exposure keywords:

```
  setenv SAS_CCF <the .cif file>
  setenv SAS_ODF <the ODF directory>

  rgssources srclist=PooooooooooiiteeeSRCLI_0000.FIT writeexpkwds=no
```

or

```
  setenv SAS_CCF <the .cif file>
  setenv SAS_ODF <the ODF directory>

  rgssources srclist=rrrrooooooooooiiteeee00_sources.ds writeexpkwds=no
```

or

```
  setenv SAS_CCF <the .cif file>
  setenv SAS_ODF <the ODF directory>

  rgssources srclist=<non-standard name> instrument=<rgs1 or rgs2>
writeexpkwds=no
```

Just format upgrade with neither exposure nor observation keywords:

```
  setenv SAS_CCF <the .cif file>

  rgssources srclist=PooooooooooiiteeeSRCLI_0000.FIT writeexpkwds=no
writeobskwds=no
```

or

```
  setenv SAS_CCF <the .cif file>

  rgssources srclist=rrrrooooooooooiiteeee00_sources.ds writeexpkwds=no
writeobskwds=no
```

or

```
  setenv SAS_CCF <the .cif file>

  rgssources srclist=<non-standard name> instrument=<rgs1 or rgs2>
writeexpkwds=no writeobskwds=no
```

6.3 Adding further sources.

6.3.1 Simplest case.

After the initial source list has been made, the user will generally want to add some source positions of his/her own. The simplest call to do this is:

```
  setenv SAS_CCF <the .cif file> [if it hasn't already been done, of course.
From now on I'll assume that SAS_CCF has been set - it is always necessary.]

  rgssources srclist=<previous rgs source list> addusersource=yes
ra=<user src ra> dec=<user src dec>
```

Notes:

1. It is unnecessary to set the parameter `filemode`=modify since this is the default value.
2. It is unnecessary with this call to supply an attitude history file.
3. It is unnecessary with this call to set the environment variable SAS_ODF.
4. If you `addusersource` but forget the `ra` and/or `dec` parameters, the task will issue a warning.
5. The pointing and prime source remain unchanged by default.
6. It is a good idea to specify a label as well for your source via the parameter `label`. Reason: the default value is 'USER'. If you try to add another source using again the default `label` value, the task will not allow this because `LABEL` values must be unique.
7. Parameter `enablefilter` if set will not at present prevent you adding an out-of-FOV source to the source list; however it will enable the task to generate a warning after the fact.

6.3.2 Old-format source list.

The considerations of subsection 6.2 apply. Note however that **rgssources** does its best to upgrade the format of any old files it encounters. It will incorporate the instrument, exposure etc into the file header if you supply them just one time on the command line; you won't then need to do so for subsequently added sources.

6.3.3 Elaborations.

If you know the position of your source with respect to the instrument pointing direction but not its ra and dec, you may want to specify the position instead via the `deltadisp` and `deltaxdsp` parameters, viz:

```
  rgssources srclist=rgssrclist.ds addusersource=yes positionstyle=wrtatt
deltadisp=<displacement of user src in dispersion direction from attitude>
deltaxdsp=<ditto cross-dispersion direction>
```

Both the `delta*` parameters have units of arcminutes. Such entries in the source list have their column `FIXED_ON_SKY` values set false. This means that their values for `RA` and `DEC` may be altered by subsequent invocations of **rgssources**, but columns `DELTA_DISP`, `DELTA_XDSP`, `FOV_PHI`, and `FOV_R` will never be altered. The opposite holds for sources which have been entered with a defined ra and dec; these have `FIXED_ON_SKY` equal to true.

Note that you need to explicitly set the parameter `positionstyle` to 'wrtatt' for this style of user-source input - its default value is 'radec'.

If you want to change the attitude you must explicitly set the parameter `changeattitude`. Just providing values for `att*` parameters for example won't have any effect. A change of attitude to a user-specified value can be achieved by:

```
  rgssources srclist=rgssrclist.ds changeattitude=yes attapos=<position
angle of instrument pointing> attdec=<etc> attra=<etc>
```

Note that in this case the atthkfile is not needed. Other ways to respecify the attitude might however require it. In these cases the task will fail with an error if `atthkset` is not filled. Note also that a recalculation of the attitude can only be done if the exposure number is known and if the user has access to the ODF. The exposure number may not be present in the source list header if the source list originated in an older format. In this case, if **rgssources** is not able to deduce the exposure number from the file name (which has to be in either PPS or **rgsproc** standard format for this to be possible), then you must supply this value to parameter `instexpid`, whether you have `writeexpkwds` set or not.

Be warned that changing the attitude will have two effects: firstly all the `DELTA_DISP`, `DELTA_XDSP`, `FOV_PHI`, and `FOV_R` values will be recalculated for those sources for which `FIXED_ON_SKY` is true, whereas `RA` and `DEC` are recalculated for those for which `FIXED_ON_SKY` is false; secondly, the region extensions created by **rgsregions** (if any) are falsified and are therefore deleted.

Changing the prime source is similar, you need to set the parameter `changeprime`. To add a user source and reset the prime source to this source, you should do something like

```
  rgssources srclist=rgssrclist.ds addusersource=yes changeprime=yes
userasprime=yes (followed by position and optionally the label of the
user source as described above)
```

Having to set both `changeprime` and `userasprime` is a bit irritating and we hope to amend the logic of the task in a future version so that this is no longer necessary. `changeprime` itself is necessary to prevent the task reverting to a prime source value defined by the default values for the tree of prime source selection parameters. Future modifications to the parameter interface may allow the task to detect whether a parameter has appeared on the command line or not and much of these extra parameters will vanish.

Note that after changing the prime source, the section of the proc/pipeline from **rgsangles** onwards should be re-run. This is because the prime source is used to calculate attitude-drift corrections to the rgs events.

7 Algorithm

```
  read parameters;

################## start format upgrade section.

  if (filemode eq 'modify') {
    # convert old-style source list if necessary:
    if (! instrument_supplied) {
      if (hasAttribute(set, 'INSTRUME')) {
        instrument = stringAttribute(oldListSet, 'INSTRUME');
      } else {
```

```
        # attempt to get it from file name:
if (! instrument_in_filename) {call error;}
      }
    }

    call CAL_setState(instrument=instrumentId);

    rename table to 'SRCLIST';
    get attitude from *_REF keywords;
    write_generic_keywords;
    if (writeobskwds) {write_observation_keywords;}
    if (writeexpkwds) {
      if (! exposure_supplied) {
        if (hasAttribute(set, 'EXP_ID')) {
          exposure = stringAttribute(oldListSet, 'EXP_ID');
        } else {
          # attempt to get it from file name:
if (! exposure_in_filename) {call error;}
        }
      }
    }

    # Convert columns:
    n_offaxis_sources = 0;
    foreach (source) {
      fixed_on_sky = true; # default
      process      = false; # default
      bkg_exclude  = false; # default
      if (old_label eq 'proposal') {
        new_label = 'PROPOSAL';
        new_ra    = old_ra;
        new_dec   = old_dec;

      } elsif (label eq 'onaxis') {
        new_label    = 'ON_AXIS';
        new_ra       = attitude_ra;
        new_dec      = attitude_dec;
        delta_disp   = 0;
        delta_xdsp   = 0;
        new_fov_phi  = 0;
        new_fov_r    = 0;
        fixed_on_sky = false;

      } elsif (label eq 'epic') {
        new_label = 'OLDEPIC'.id_band.ml_id_src;
        new_ra    = old_ra;
        new_dec   = old_dec;
        epic_file = number of bands found;

      } elsif (label eq 'offaxis') {
        n_offaxis_sources = n_offaxis_sources + 1;

        new_label    = 'OFFAXIS'.n_offaxis_sources;
        delta_disp   = rgs_disp;
        delta_xdsp   = rgs_xdsp;
```

```
        calculate new_fov_phi, new_fov_r;
        fixed_on_sky = false;

      } else { # assume is user-added source.
        new_label = upper_case(old_label);
        new_ra    = old_ra;
        new_dec   = old_dec;
      }
      new_rate = old_rate;
      if (src_select  > 0) {process     = true;}
      if (back_select > 0) {bkg_exclude = true;}
      if (fixed_on_sky) {
        calculate delta_disp;
        calculate delta_xdsp;
        calculate new_fov_phi;
        calculate new_fov_r;

      } else {
        calculate new_ra;
        calculate new_dec;
      }
    }

  calculate_confusion(src_data, prime_index);
  delete old columns;
  write column data to new columns;
  write epic keywords;
  }

################## end format upgrade section.

  # get the instrument:
  if (filemode eq 'modify') {
    instrument = stringAttribute(oldListSet, 'INSTRUME');
  } else {
    # get instrument from parameter;
if (instrument parameter at default value) {call error;}
  }
  # get the exposure number:
  expid_is_valid = true;
  if (filemode eq 'modify') {
    exposure = int32Attribute(oldListSet, 'EXP_ID');
  } else {
    # get exposure from parameter;
    if (exposure parameter at default value) {
      expid_is_valid = false;
    }
  }

  call CAL_setState(instrument=instrumentId);

  # get the pointing:
  if (filemode .eq. 'modify' && ! changeAttitude) {
    # read attitude from old source list;
  } elsif (attitudeStyle .eq. 'user') {
```

```
    # read attitude from attra, attdec and attapos parameters;
  } else {
if (!expid_is_valid) {call error;}
    call OAL_setState(instrument=instrument, exposureNr=exposure);
    if (attitudeStyle .eq. 'start') {
      # get attitude from attitude at start of observation;
    } elsif (attitudeStyle .eq. 'mean') {
      # get attitude from mean of attitude history file values;
    } elsif (attitudeStyle .eq. 'median') {
      # get attitude from median of attitude history file values;
    }
  }

  allocate(src_data(max_number_sources));

  if (filemode eq 'create') {
    proposal position -> src_data(1);
    attitude position -> src_data(2);

  } else { # filemode eq 'modify'
    open(old rgs set, 'READ');
    call check(old rgs list); # This checks that all label names are unique,
    # and that there is a proposal and attitude source.

    old rgs sources -> src_data;
    release(old rgs set);

    if (changeattitude) {attitude position -> src_data(index of onaxis src);}
  }

  if (withepicset) {
    clone(old epic set);
    if (enablefilter) {spatial filtering of cloned epic set;}
    foreach(epic source) {
      label = 'epiclabelprefix'.column('*_ID_SRC');
      foreach(source already in src_data) {
        if (label eq src_data(LABEL)) {
          no_label_clash = F;
          if (clobberonlabel) {
      last;
          } else {
call error;
          }
        }
      }
      if (no_label_clash) {
        epic source -> src_data;
      }
    }
    release(old epic set);
  }

  if (addusersource) {
    foreach(source already in src_data) {
      if (label eq src_data(LABEL)) {
```

```
        no_label_clash = F;
        if (clobberonlabel) {
    last;
        } else {
call error;
        }
       }
    }
    if (no_label_clash) {
      user source -> src_data;
    }
  }

  # Possible routine in this place that filters out (or at least identifies)
  # all pairs of too-close sources.

  if (changeattitude or null in column) {
    foreach(source in src_data) {
      # Calculate ra/dec or delta_disp/delta_xdsp/fov_phi/fov_r, as appropriate;
    }
  }

  # Write the data to file:
  if (filemode eq 'create') {
    open(new rgs set, 'CREATE');
    # write keywords to header:
    write_generic_keywords;
    if (writeobskwds) {write_observation_keywords;}
    if (writeexpkwds && expid_is_valid) {write_exposure_keywords;}

  } else { # filemode eq 'modify'
    open(old rgs set, 'MODIFY');
    if (changeattitude) {
      rewrite attitude attributes (#RA_PNT etc);
      delete all region extensions;
      set PROCESS and BKG_EXCLUDE to false;
    }
    delete all table entries;
  }

  src_data -> new rgs set;
  if (withepicset) {
    write_epic_specific_keywords;
  }

  if (enablefilter) {check that all members of the src list are within the
  fov, warn if not;}

  # Now calculate prime source index value:
  if (filemode eq 'modify' && !changeprime) {
    prime_index = prime_index in old file;
  } else {
    if (addusersource && userasprime) {
      prime_index = index of (last) user-added source;
    } else {
```

```
    if (primestyle eq 'label') {
      prime_index = index of source with label=primesourcelabel;
    } elsif(primestyle eq 'index') {
      prime_index = primesourceindex;
    } elsif(primestyle eq 'brightest') {
      prime_index = index of brightest source;
    } elsif(primestyle eq 'auto') {
      if (proposal source is within FOV) {
        prime_index = index of proposal source;
      } elsif (there are some epic sources within FOV) {
        prime_index = index of brightest within-FOV epic source;
      } else {
        prime_index = index of pointing (ONAXIS) source;
      }
    }
  }
} else {
  prime_index = previous prime_index;
}

if (filemode eq 'create' || changeprime) {
  prime_index -> #PRIMESRC;
}

call calculate_confusion(src_data, prime_index);
# Write the confusion data to file:
# release the data set.
```

8 Parameters

This section documents the parameters recognized by this task (if any).

Parameter	Mand	Type	Default	Constraints
filemode	no	string	modify	modify—create

Controls whether the task opens a previous source list for editing or creates a new one.

changeprime	no	boolean	no	yes—no

Only active in `filemode`='modify'. Unless this parameter is set, the previous prime source index number is retained.

changeattitude	no	boolean	no	yes—no

Only active in `filemode`='modify'. Unless this parameter is set, the previous attitude (stored in the header) is retained.

srclist	yes	dataset	rgsset.ds	

The name of the rgs source list. If `filemode`='create', the output is written to this file. If there is an existing file of this name, it will be overwritten unless SAS_CLOBBER is unset. If `filemode`='modify', the task looks for an existing source list of this name and modifies it.

instexpid	no	string		

This parameter contains information about both the instrument (that is, RGS1 or 2) and the exposure

identifier (a letter S or U, indicating scheduled or unscheduled, followed by a three-digit numeric identifier. The `instexpid` string can be supplied in a number of different forms, but the two most useful are (i) as a six-character string comprising either R1 or R2 followed by the exposure identifier (an example: 'R2S003'); (ii) the name of any of RGS-specific files in the ODF can also be used. This parameter is mandatory if `filemode`='create', or in cases where the instrument and/or exposure can neither be read from the file header or deduced from its name.

writeobskwds	no	boolean	yes	yes—no

If this is set, the task attempts to write observation-specific keywords to the file header. The user must point the environment variable SAS_ODF to the ODF directory for this to succeed.

writeexpkwds	no	boolean	yes	yes—no

If this is set, the task attempts to write exposure-specific keywords to the file header. For this to succeed, the user must point the environment variable SAS_ODF to the ODF directory, and the task must also be able to determine the exposure number, either via the `instexpid` parameter, or from the `EXPIDSTR` keyword in the file header, or (if neither are present) from the file name.

clobberonlabel	no	boolean	no	yes—no

Labels in RGS source lists are required to be unique. Where a clash is detected between a source already in the list and a new candidate source, the task takes one of two actions, depending on the value of this parameter: if 'yes', the candidate is discarded; if 'no', the task halts with an error.

primestyle	no	string	label	label—index—expr—brightest—auto

Only active if `changeprime`=yes and either `addusersource` or `userasprime`=no. It controls the way in which the prime source is specified. See the parameters `primelabel` and `primeindex`. (An additional possible value of 'expression' is planned.)

primelabel	no	string	PROPOSAL	

If `primestyle` is active and set to 'label', this parameter gives the value of the `LABEL` column of the source that it is desired the `PRIMESRC` keyword should point to.

primeindex	no	integer	1	0 <primeindex

If `primestyle` is active and set to 'index', the `PRIMESRC` keyword is set to this value.

primeexpression	no	string		

This mode is not yet supported.

attstyle	no	string	expmedian	mean—median—start—user—expmedian

Controls the way the attitude is calculated. If 'mean', the attitude is calculated from the mean of the values in the attitude history file. If 'median', the median of these values is used. If the value is 'start', the task uses the attitude at the start of the exposure as the reference attitude. A value of 'expmedian' tells the task to use the median of the attitude during the exposure only, as calculated by **attfilter**. The final value, 'user', allows the user to input the numbers him/herself via the next three parameters.

meanset	no	dataset	atthk.dat	

The name of the attitude history file. This file is a necessary input in the case that `attstyle` is 'mean'.

medianset	no	dataset	atthk.dat	

The name of the attitude history file. This file is a necessary input in the case that `attstyle` is 'median'.

attra	yes	angle	0	$0 \leq$ `attra` ≤ 360

Only active if `attstyle`='user'. The right ascension of the attitude, in decimal degrees.

attdec	yes	angle	0	$-90 \leq$`attdec`≤ 90

Only active if `attstyle`='user'. The declination of the attitude, in decimal degrees.

attapos	yes	angle	0	$0 \leq$`attapos`≤ 360

Only active if `attstyle`='user'. The position angle of the attitude, in decimal degrees.

expmediantable	no	table	attgti.ds:STDGTI	

The name of the table in the filtered attitude history file in which the exposure-median keywords can be found. This file is a necessary input in the case that `attstyle` is 'expmedian'.

addusersource	no	boolean	no	yes—no

This should be set if the user wishes to add a source to the list with a position specified on the command line.

label	no	string	USER	

Only active if `addusersource`=yes. This is written directly to the `LABEL` column of the output source list. The empty string is not permitted.

rate	no	real	0.0	0.0 <rate

Only active if `addusersource`=yes. The brightness of the source in counts per second. It is anticipated that this parameter won't be used much, since this is not a quantity that is likely to be known in most circumstances. The default value of 0.0 is harmless.

userasprime	no	boolean	no	yes—no

Only active if `addusersource`=yes. If `changeprime`=yes and `userasprime`=yes, then the attribute `PRIMESRC` is set to the index number of the user source.

process	no	boolean	no	yes—no

Only active if `addusersource`=yes. This causes the value in the `PROCESS` column to be set to true for the user-added source.

bkgexclude	no	boolean	yes	yes—no

Only active if `addusersource`=yes. This causes the value in the `BKG_EXCLUDE` column to be set to true for the user-added source.

positionstyle	no	string	radec	radec—wrtatt

Only active if `addusersource`=yes. If `positionstyle`='radec', then the position of the user-added source is expected via the parameters `ra` and `dec`. If on the other hand `positionstyle`='wrtatt' (With Respect To ATTitude), then the position of the user-added source is expected via the parameters `deltadisp` and `deltaxdsp`.

ra	yes	angle	0	$0 \leq$`ra`≤ 360

Only active if `addusersource`=yes and `positionstyle`='radec'. The right ascension of the user-added source, in decimal degrees.

dec	yes	angle	0	$-90 \leq$`dec`≤ 90

Only active if `addusersource`=yes and `positionstyle`='radec'. The declination of the user-added source, in decimal degrees.

deltaxdsp	yes	real	0.0	

Only active if `addusersource`=yes and `positionstyle`='wrtatt'. The displacement in arcminutes of the

user-added source from the pointing direction, in the dispersion direction.

deltadisp	yes	real	0.0	

Only active if `addusersource`=yes and `positionstyle`='wrtatt'. The displacement in arcminutes of the user-added source from the pointing direction, in the cross-dispersion direction.

withepicset	no	boolean	no	yes—no

If this is set, the task looks for the parameter `epicset`, giving the name of an EPIC source list.

epicset	no	dataset		

The name of a set containing a list of sources. Formats output by the tasks **emldetect** and **eboxdetect** are accepted.

epiclabelprefix	no	string	EPIC	

This parameter gives the string which is used by the task as a prefix when constructing `LABEL` values for EPIC-derived sources. The other part of the `LABEL` is the number `ML_ID_SRC` or `BOX_ID_SRC`. The main purpose of this parameter is to allow several EPIC-derived source lists to be included in the one RGS list if desired, while retaining unique labels.

enablefilter	no	boolean	no	yes—no

Active only if `withepicset`=true. In this case the task adds to the output list only those epic sources which correspond to cross-dispersion angles on the RGS camera between -3 and +3 arcminutes from camera centre. Regardless of whether epic sources are added or not, the task checks the positions of all sources if `enablefilter` is set and warns about any that fall outside the FOV.

doconfusion	no	boolean	no	yes—no

Active only if `withepicset`=true. This parameter causes the task to check the epic sources + proposal position for confusion in the EPIC field of view. It is mainly designed for use in the PCMS, to prevent automatic extraction of too many spectra for what is essentially the same object.

9 Errors

This section documents warnings and errors generated by this task (if any). Note that warnings and errors can also be generated in the SAS infrastructure libraries, in which case they would not be documented here. Refer to the index of all errors and warnings available in the HTML version of the SAS documentation.

noAttitude *(error)*
No attitude information is available from the **oal** for the start time of the exposure.

badAttitudeStyle *(error)*
The value of `attstyle` was not recognized.

badFileMode *(error)*
The value of `filemode` was not recognized.

nonUniqueEpicId *(error)*
There is more than one source in the EPIC source list with a given `ML_ID_SRC` or `BOX_ID_SRC` number.

tooManyProposals *(error)*
The RGS source list to be modified has more than one source with the label 'PROPOSAL'.

noProposal *(error)*
The RGS source list to be modified has no source with the label 'PROPOSAL'.

tooManyAttSrc *(error)*
The RGS source list to be modified has more than one source with the label 'ONAXIS'.

noAttSrc *(error)*
The RGS source list to be modified has no source with the label 'ONAXIS'.

badEpicIdColumn *(error)*
The supposed EPIC source list has neither a `ML_ID_SRC` or `BOX_ID_SRC` column.

emptyUserLabel *(error)*
The `label` parameter, which is the desired label of the user-added source, is the empty string.

duplicateLabel *(error)*
A candidate additional source has been found to have the same label as one of the existing sources. This error can only occur if the parameter `clobberonlabel`=no.

notSupported *(error)*
A mode has been requested that is not yet implemented.

badPrimeStyle *(error)*
The value of `primestyle` was not recognized.

unableToSetNull *(error)*
For some reason it wasn't possible to set an integer 32 null value for the column `EPIC_FILE`.

badSrcFileType *(error)*
The source list supplied to parameter `srclist` is not in a recognised format.

unknownInstrument *(error)*
No valid value was provided for the `instexpid` parameter, and **rgssources** was unable to deduce the instrument from the old-format file header or name.

notStandardFileName *(error)*
The task has attempted to deduce either the instrument or the exposure number from the file name, but the name was not found to adhere to either the PPS or **rgsproc** formats.

instIdClash *(warning)*
The instrument recorded in the source list header keyword INSTRUME is not the same as that derived from the value of the `instexpid` parameter.
corrective action: The task uses the value from the source list header.

expIdStrClash *(warning)*
The 4-character exposure identifier recorded in the source list header keyword EXPIDSTR is not the same as that derived from the value of the `instexpid` parameter.
corrective action: The task uses the value from the source list header.

noValidEpicSources *(warning)*
None of the epic sources have met all the conditions for inclusion.
corrective action: No epic sources are added to the file.

noEpicSourcesInFov *(warning)*
None of the epic sources were found to lie within the field of view of the RGS camera.
corrective action: No epic sources are added to the file.

noEpicSources *(warning)*
There are no epic sources in the correct energy band.
corrective action: No epic sources are added to the file.

duplicateLabel *(warning)*
A candidate additional source has been found to have the same label as one of the existing sources. If `clobberonlabel`=no, this condition causes the task to fail with an error.
corrective action: New source is not appended to the source list.

outsideFOV *(warning)*
A source has been found to lie outside the field of view of the RGS camera.
corrective action: No action.

confSetAllZeroRate *(warning)*
A set of confused sources has none with a positive value of RATE.
corrective action: All sources flagged as confused.

primeLabelNotFound *(warning)*
The user has set `changeprime`=yes and `primestyle`='label', but no source with a label equal to `primelabel` has been found in the file.
corrective action: The prime source is set to point to the proposal source.

primeIndexNotFound *(warning)*
The user has requested that the prime source be set to a source of a particular index value, but no source in the list has this index value.
corrective action: The prime source is set to point to the proposal source.

noBrightPrime *(warning)*
The user has requested that the prime source be set to the index of the brightest source, but brightest source has a rate less than or equal to zero.
corrective action: The prime source is set to point to the proposal source.

noChangePrime *(warning)*
The user has set `userasprime`, but not `changeprime`.
corrective action: Prime source is not changed.

primeAtLabelDefault *(warning)*
The user has set `changeprime`=yes and `primestyle`='label', but `primelabel` still has its default value. This *may* indicate that the user forgot to set it.
corrective action: No action.

primeAtIndexDefault *(warning)*
The user has set `changeprime`=yes and `primestyle`='index', but `primeindex` still has its default value. This *may* indicate that the user forgot to set it.
corrective action: No action.

primeAtDefault *(warning)*
A new RGS source list is being created, but `primeindex` still has its default value. This *may* indicate that the user forgot to set it.
corrective action: No action.

primeOutsideFOV *(warning)*
The designated prime source is outside the field of view of the RGS camera.
corrective action: No action.

instrumentClash *(warning)*
The instrument deduced from the file name does not agree with that deduced from the file header.
corrective action: Header instrument will be used.

uncertainInstrument *(warning)*
The instrument has been deduced from the file name.
corrective action: Continues.

uncertainExposure *(warning)*
No `instexpid` value was supplied, and there is no `EXPIDSTR` keyword in the file.
corrective action: Task attempts to deduce exposure number from the file name.

badPrimeRate *(warning)*
The `RATE` value of the designated prime source has been found to be less than or equal to zero. Confusion values can therefore not be calculated.
corrective action: Confusion values are set to zero.

References