# slconv

January 12, 2017

**Abstract**

Converts a FITS source list to gaia or ds9 format.

## 1 Instruments/Modes

| Instrument | Mode |
|---|---|
| Not XMM-specific. | |

## 2 Use

| | |
|---|---|
| pipeline processing | yes |
| interactive analysis | yes |

## 3 Description

### 3.1 Introduction.

Task **slconv** converts a FITS file containing a list of source positions to a format that can be read by one or other of the image-analysis applications gaia and ds9. The choice between these two is made via the parameter `outputstyle`. (This parameter is case-insensitive.) Ds9 is described in reference [3] and gaia in [2]. The format of gaia region files is described in reference [1].

The input FITS file must contain at least two columns, of data type either real32 or real64, which are assumed to contain the RA and dec values of the sources. The names of these columns should be passed to **slconv** via the `racolumn` and `deccolumn` parameters. The units of the columns can be either decimal degrees or radians, this unit being supplied to the task via the `radecunits` parameter. (String-valued, hh:mm:ss-type columns may be permitted in a later version.) Both columns must have the same unit.

Sources are plotted with shape and colour defined by the parameters `shape` and `colour`. A subset of sources can be selected for plotting; the radii of the circles can be specified in several ways; and labels can be added to the circles. These facilities are described in the following subsections.

The output file name is composed of a prefix + period + suffix. The user can choose the prefix via the outfileprefix parameter, but the suffix is always 'reg' for outputstyle='ds9' and 'gaia' for outputstyle='gaia'.

## 3.2 Source selection.

This is accomplished via the parameter expression. The user can supply a boolean selection expression to this parameter, involving (for example) names of columns in the source table. The full grammar of such expressions is described in the **selectlib** library.

An example of a situation in which some selection may be desirable is in the case in which the source list is the standard XMM **emldetect** pipeline product. These lists contain many rows corresponding to the same source position but with different values of the ID_BAND column. Inclusion of all the rows in the table will lead to plots which have many concentric source circles at each position. It is better to use a selection expression such as

```
  expression='(ID_BAND==0)'
```

to limit the sources plotted to those with ID_BAND = 0 (or whatever other band number is desired).

## 3.3 Maximum number of sources.

It may be desirable to apply (in addition to the source selection described above) some definite upper limit to the number of sources to be plotted - say, to plot only the first 50 brightest sources. Such a limit may be difficult or impossible to achieve by use of the expression parameter. For this reason, an ncut parameter has also been provided. However, this parameter alone cannot provide full versatility. It is not sufficient just to specify the number of sources to keep: it is also necessary to determine which sources to keep. This means in practice that the sources must be sorted into a series before the series can be truncated after the ncutth member. The parameter ncutsortexpression has been provided to allow the sources to be sorted. This works as follows: the task looks in parameter ncutsortexpression for an expression involving column names; this expression is evaluated and the sources are then ordered in decreasing order of the resulting value.

For example,

```
  slconv srclisttab=bert.ds:SRCLIST radiusexpression='RATE'
  withncut=yes ncutsortexpression='RATE' ncut=10
```

produces a ds9 region file containing data for the first 10 brightest sources from the SRCLIST table in file 'bert.ds' (that is, those 10 sources in the table which have the highest values of RATE).

## 3.4 Source circle radii.

The output file (in either format) directs the plotting package (gaia or ds9) to draw a circle of a certain radius at a certain sky position. The radius can be made an arbitrary function of column values in the input source table. The task first calculates the value of the column expression in parameter radiusexpression. What happens next depends on the value of the radiusstyle parameter. If this is 'raw' (the default), the task does no further processing: the result of evaluating radiusexpression is used directly as the radius of the circle to be plotted. However this procedure is clearly somewhat vulnerable to error by the user. One can envisage a situation in which several iterations might be necessary before the sizes of the plotted circles attained a pleasing appearance. This is unavoidable if precise control over the source circle radii is desired; however there may be other occasions on which the user just wants the source circle radii to display qualitative information. It is for this purpose that the other setting, 'auto', of radiusstyle was designed. Under this setting, **slconv** does its best to scale the circle radii already evaluated from radiusexpression such that the maximum and minimum radii will be sensible fractions of the image size, and so that the circles are distributed fairly evenly between these extrema.

### 3.5 Source region shapes.

Several region shapes can be selected through parameter shape: circle, box, diamond, ellipse and boxcircle. The size of these regions is determined once the radii is calculated as explained in previous section:

- - box side = 2× circle_radius
- - ellipse_minor_axis= circle_radius
- - ellipse_major_axis= 2× circle_radius

### 3.6 Source labels.

Labels can be added to the sources by setting withlabels='yes'. The source of the label information depends on the value of the parameter labelstyle. This has two possible values:

1. labelstyle='sortedint': The labels comprise a sequence of positive-valued integers. These are obtained by sorting the sources according to some criterion and then numbering them in this order. The sorting criterion is specified by the parameter labelsortstyle, described as follows:
    - labelsortstyle='radius': The sorting is according to the same value that was used to scale the radii. Hence the sequence of integer labels starts at the largest circle and increases in order of their decreasing size.
    - labelsortstyle='expr': In this case the sorting is done by evaluating the expression contained in parameter sortexpression. The label numbering is in decreasing order of the resulting value.
    - labelsortstyle='rownumber': This is equivalent to no sorting - the labels are just taken from row numbers in the source list (note this is the expression-FILTERED source list not the ORIGINAL one).
2. labelstyle='expr': This specifies that the labels are alphanumeric strings, obtained by evaluating the expression contained in parameter labelexpression. Eventually the task will be able to accept in this parameter an expression involving several numeric-valued columns. The task would then evaluate the expression, and set source labels from the result. However at present it cannot do this: you can only supply names of single columns from the source list. String-valued columns are also allowed in this case.

NOTE that, whereas ds9 allows one to offset the labels from the source circles, no such facility appears to be available in gaia.

The colour of the source circles and labels may be specified via the parameter colour.

## 3.7 Extra columns for GAIA output.

Gaia region files are organised as a table of data, with additional information in the header which both specifies the source symbols and directs how their size and orientation should be calculated from the information in the table. Details of this arrangement can be found in reference [1]. Task **slconv** permits only circles as source symbols, but the type of symbol and its specification expression can easily be altered by editing the output region file after running **slconv**. To allow for greater flexibility however **slconv** allows the user to copy additional columns from the input FITS source list to the gaia region file. (Columns entitled 'Id', 'ra' and 'dec' are always generated in the output.) These column names should be listed in the parameter `extragaiacolumns`. Note that the default setting of this parameter is to copy two columns from the source list, namely `RADEC_ERR` and `RATE`. If columns of these names are not present in the input source list, or not desired in the output, they should be explicitly removed from `extragaiacolumns` at the time of invocation of **slconv**.

## 3.8 Examples.

The following minimal command

```
slconv srclisttab=srclist.ds radiusexpression='RATE/10.0'
```

does the following:

- Reads sources from the first table in the dataset srclist.ds;
- Reads RA and dec information, in decimal degrees, from columns named `RA` and `DEC`;
- Uses all the sources (no filtering or truncation);
- Determines the radii of the ds9 or gaia circles in arcseconds, directly using the number found in a column called `RATE`, divided by 10;
- Specifies green circles for the source symbols;
- Writes the output to a ds9-style region file called region.reg.

If data from some other table in srclist.ds is desired, say one called `SOURCES`, it must be specified in the value of `srclisttab` as follows:

```
slconv srclisttab=srclist.ds:SOURCES radiusexpression='RATE/10.0'
```

If the RA and dec are contained in columns of names different from the defaults, these also must be specified directly, eg:

```
slconv srclisttab=srclist.ds radiusexpression='RATE/10.0'
racolumn=RA_CORR deccolumn=DEC_CORR
```

Similarly with the unit of these data:

```
slconv srclisttab=srclist.ds radiusexpression='RATE/10.0' radecunits=radians
```

An example of some filtering of the input source list:

```
  slconv srclisttab=srclist.ds radiusexpression='RATE/10.0'
  expression='(ID_BAND==0)&&(!(isnull(EXT)))&&((ID_INST!=1)||(RAWY>15))'
```

Suppose you wanted to display only the 10 most extended sources in an XMM-product source list. You could do:

```
  slconv srclisttab=srclist.ds radiusexpression='RATE/10.0'
  expression='!(isnull(EXT))' withncut=yes
  ncutsortexpression='EXT' ncut=10
```

If you wanted the source circles in the previous example to scale with the extent `EXT` instead of `RATE`/10, but were not very concerned to maintain an exact quantitative arrangement, the following would produce an acceptable qualitative result:

```
  slconv srclisttab=srclist.ds radiusexpression='EXT' radiusstyle=auto
  expression='!(isnull(EXT))' withncut=yes
  ncutsortexpression='EXT' ncut=10
```

Still maintaining this example, suppose you also wanted the sources to be numbered in decreasing order of the extent, you should do:

```
  slconv srclisttab=srclist.ds radiusexpression='EXT' radiusstyle=auto
  expression='!(isnull(EXT))' withncut=yes
  ncutsortexpression='EXT' ncut=10
  withlabels=yes labelstyle=sortedint labelsortstyle=radius
```

This locks the source numbering into the calculation of radii. If you changed `radiusexpression` in the previous command line to '`DET_ML`' for example, but nothing else, then the radii would then decrease with a decrease in the detection likelihood `DET_ML`, but the labels would still be integers in an ordered sequence following the decrease in radius. If you wanted to tie the label sequence definitely to, say, `EXT`, regardless of which rule the radius followed, you would rather do

```
  slconv srclisttab=srclist.ds radiusexpression='EXT' radiusstyle=auto
  expression='!(isnull(EXT))' withncut=yes
  ncutsortexpression='EXT' ncut=10
  withlabels=yes labelstyle=sortedint labelsortstyle=expr sortexpression='EXT'
```

To use the actual value of, say, column `COUNTS` as the label, do:

```
  slconv srclisttab=srclist.ds radiusexpression='RATE/10.0'
  withlabels=yes labelstyle=expr labelexpression='COUNTS'
```

So far all these examples have produced an output file in ds9 format, since this reflects the default setting of the parameter `outputstyle`. To obtain gaia-style output, this must be set to 'gaia' explicitly:

```
slconv srclisttab=srclist.ds radiusexpression='RATE/10.0' outputstyle=gaia
```

With this nearly all-default command line, the processing is identical to that in the first example in this subsection, except that the task in addition looks for two more columns in the input source table: `RADEC_ERR` and `RATE`, because the default value of `extragaiacolumns` lists these two columns. If these are found, their contents are copied to columns with identical names in the output table (an error results if either is not found). If you want a different set of columns, these must of course be explicitly specified, for example:

```
slconv srclisttab=srclist.ds radiusexpression='RATE/10.0' outputstyle=gaia
extragaiacolumns='RADEC_ERR COUNTS EXT EXT_ML'
```

## 4 Parameters

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

| **Parameter** | Mand | Type | Default | Constraints |
|---|---|---|---|---|

| **srclisttab** | yes | table | srclist.ds:SRCLIST | |
|---|---|---|---|---|

The name of the FITS dataset+table which contains the input source list. If the colon + table name is not given, the first table in the dataset is used.

| **racolumn** | no | string | RA | |
|---|---|---|---|---|

The name of the column from which to read the Right Ascensions of the sources.

| **deccolumn** | no | string | DEC | |
|---|---|---|---|---|

The name of the column from which to read the Declinations of the sources.

| **radecunits** | no | string | decimaldeg | decimaldeg—hdms—radians |
|---|---|---|---|---|

The unit to employ when reading `racolumn` and `deccolumn`. NOTE that hdms (ie hh:mm:ss etc) is not yet supported.

| **expression** | no | string | ‘’ | |
|---|---|---|---|---|

Source selection expression, possibly involving column names, keyword values or constants. See **selectlib** for the grammar. Only those sources for which the expression evaluates to TRUE are plotted. The default (an empty string) is TRUE for all sources.

| **withncut** | no | boolean | no | |
|---|---|---|---|---|

Switch which specifies whether the source list (after filtering with `expression`) should be truncated at some maximum number, specified by parameter `ncut`.

| **ncutsortexpression** | yes | string | RATE | |
|---|---|---|---|---|

If a truncation of the size of the source list is desired (ie if `withncut`=yes), the list must first be sorted according to some criterion. The task evaluates `ncutsortexpression` and sorts the sources in decreasing order of the result. The parameter `ncut` is then read and only the first `ncut` sources (in their sorted order) are plotted.

| **ncut** | no | integer | 30 | 0 $\leq$`ncut` |
|---|---|---|---|---|

Maximum number of sources to plot. This parameter has no effect when `withncut`=no.

| | | | | |
|---|---|---|---|---|
| **radiusexpression** | yes | string | RATE | |

This parameter specifies an arithmetical expression, possibly involving column names, keyword values or constants. The result is used to scale the radii of the plotted circles.

| | | | | |
|---|---|---|---|---|
| **radiusstyle** | no | string | raw | auto—raw |

The radii of the source circles are taken from the result of the expression supplied to parameter `radiusexpression`; but this result can either be used directly (`radiusstyle`='raw', the default) or the value may be processed in an 'intelligent' manner by the task (`radiusstyle`='auto'). The 'auto' routine attempts to choose the extrema and distribution of the circle radii so as to give a visually pleasing result.

| | | | | |
|---|---|---|---|---|
| **withlabels** | no | boolean | no | |

Switch which specifies whether labels should be added to the output.

| | | | | |
|---|---|---|---|---|
| **labelstyle** | no | string | expr | sortedint—expr |

This parameter is read by the task if `withlabels`='yes' and controls how source labels are assigned. See the description of this parameter in subsection 3.6 but, briefly: if `labelstyle`='expr', the task uses the values generated by evaluation of the column expression supplied to `labelexpression` directly as label text; for `labelstyle`='sortedint', the labels are constrained to be a sequence of positive integers, assigned according to the setting of the parameter `labelsortstyle`.

| | | | | |
|---|---|---|---|---|
| **labelsortstyle** | no | string | expr | radius—expr—rownumber |

This parameter controls how integer-sequence source labels are assigned - that is, it is read by the task only if `labelstyle`='sortedint'. See the description of this parameter in subsection 3.6 but, briefly: if `labelsortstyle` = 'expr', the task evaluates `sortexpression` and assigns the (integer-valued) labels in decreasing order of the result; if `labelsortstyle` = 'radius', the integers increase with decreasing source circle radius; whereas if `labelsortstyle` = 'rownumber', the sequence of integers merely reflects the row number of the source entry in the (filtered, not original) source list.

| | | | | |
|---|---|---|---|---|
| **sortexpression** | yes | string | ML_ID_SRC | |

This parameter is active only if `labelstyle` = 'sortedint' and `labelsortstyle` = 'expr'. In this case the task evaluates `sortexpression` and assigns the (integer-valued) labels in decreasing order of the result.

| | | | | |
|---|---|---|---|---|
| **labelexpression** | yes | string | ML_ID_SRC | |

This parameter is active if `labelstyle` = 'expr'. Eventually the task will be able to accept in this parameter an expression involving several numeric-valued columns, evaluate the expression, and set source labels from the result; however at present it cannot do this: you can only supply names of single columns from the source list. String-valued columns are also allowed in this case.

| | | | | |
|---|---|---|---|---|
| **colour** | no | string | green | |

Colour of the plotted circle + label. Available colours are black—white—red—green—blue—cyan—purple—yellow.

| | | | | |
|---|---|---|---|---|
| **shape** | no | string | circle | |

Shape of the plotted region. Available shapes are circle—box—diamond—ellipse—boxcircle.

| | | | | |
|---|---|---|---|---|
| **outputstyle** | no | string | ds9 | ds9—gaia |

This parameter controls the output format. At present only ds9 or gaia formats are available. Note that this parameter is not case-sensitive.

| | | | | |
|---|---|---|---|---|
| **extragaiacolumns** | no | string list | RADEC_ERR RATE | |

This contains a list of extra column names to be copied to the gaia output file. It is of course only read if `outputstyle`='gaia'.

| | | | | |
|---|---|---|---|---|
| **outfilestyle** | no | string | prefix | prefix—whole |

The user has the facility to supply just a file prefix (`outfilestyle`='prefix'), in which case the post-period suffix is set to 'reg' for ds9 files and 'gaia' for gaia files, or to supply the full file name (`outfilestyle`='whole').

| | | | | |
|---|---|---|---|---|
| **outfileprefix** | no | string | region | |

Prefix of the output file name. This is followed by a period and the suffix 'reg' or 'gaia' for `outputstyle`='ds9' or 'gaia' respectively. This parameter is read if `outfilestyle`='prefix'.

| | | | | |
|---|---|---|---|---|
| **outfile** | no | string | region.txt | |

The complete output file name. This parameter is read if `outfilestyle`='whole'.

## 5 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.

**badOutputStyle** *(error)*
The value of the `outputstyle` parameter was not recognized.

**badLabelSortStyle** *(error)*
The value of the `labelsortstyle` parameter was not recognized.

**badLabelStyle** *(error)*
The value of the `labelstyle` parameter was not recognized.

**noSourcesSelected** *(error)*
The selection expression passed zero sources.

**tResultColAlreadyExists** *(error)*
Signals that the source list already contains a column `T_RESULT`.

**indexColAlreadyExists** *(error)*
Signals that the source list already contains a column `INDEX`.

**badRaDecUnit** *(error)*
The value of the `radecunits` parameter was not recognized.

**rResultColAlreadyExists** *(error)*
Signals that the source list already contains a column `R_RESULT`.

**allRadiiAreNulls** *(error)*
All the rows specified by `radiusexpression` are null-valued.

**scaleBalanceError** *(error)*
The subroutine scaleBalance returned a non-zero error status.

**lResultColAlreadyExists** *(error)*
Signals that the source list already contains a column `L_RESULT`.

**badLabelExpression** *(error)*
The task has assumed that `labelexpression` contains just one column name. It has not found a column of that name in the source list.

**badRegionShape** *(error)*
The region shape is not allowed

**badLabelColDataType** *(error)*
The data type of the column specified in parameter `labelexpression` is not recognized.

**badExtraGaiaColName** *(error)*
One of the columns listed in `extragaiacolumns` was not found in the source list.

**badExtraGaiaColDataType** *(error)*
One of the columns listed in `extragaiacolumns` had an unrecognized data type.

**badOutFileStyle** *(error)*
The value of the `outfilestyle` parameter was not recognized.

**someRadiiAreNegative** *(warning)*
Some of the source radii were found to be less than zero. You may need to rethink `radiusexpression`.
*corrective action:* These values will be replaced by zero.

**someRadiiAreNulls** *(warning)*
Some of the source radii were found to be null-valued (NaNs).
*corrective action:* The offending sources are not plotted.

**allIntegerRadiiAreZero** *(warning)*
After conversion to integer, all of the radii were found to be equal to zero. You will need to make them bigger in `radiusexpression`.
*corrective action:* No action.

**someIntegerRadiiAreZero** *(warning)*
After conversion to integer, some of the radii were found to be equal to zero. You may want to make them bigger in `radiusexpression`.
*corrective action:* No action.

**noSources** *(warning)*
No sources have passed the filtering expression.
*corrective action:* Program exits gracefully.

**radiiAllEqual** *(warning)*
The maximum and minimum source radii are the same. This can happen in a number of ways. Maybe there is only one source; or maybe you have a constant expression in `radiusexpression`. In the latter case there is no point in asking for `radiusstyle`='auto'.
*corrective action:* Task proceeds anyway.

## 6 Input Files

1. A FITS-format binary table containing at least two columns, which are assumed to contain RA and Dec values for the sources. These may have either real32 or real64 data types (their data types need not be the same). The names of these columns are not constrained but if they are not called `RA` and `DEC` then their names should be supplied to the parameters

`racolumn` and `deccolumn`. Units can be either decimal degrees (`radecunits`='decimaldeg') or radians (`radecunits`='radians'). String-type columns with HH:MM:SS-style data may be accepted by a future version of the task.

Additional columns as listed in parameters `expression`, `ncutsortexpression`, `radiusexpression`, `sortexpression`, `labelexpression` and `extragaiacolumns` may be required.

## 7 Output Files

The output file is an ascii file that conforms to one of two formats, ds9 or gaia. The desired format is given by the parameter `outputstyle`. These are described below:

- DS9 file: This consists of a header line defining the symbol colour among other things, followed by a lines describing the source symbol type location and size. There is one line per source. If labels are specified there will be in addition one more line per source, specifying the label text and position. The structure of the ds9 region file is specified in reference [3].
- GAIA file: Gaia region files consist of several header lines followed by a table. The header lines define (among other things) the source symbol types and colours and specify how their occurrence, sizes and orientations should be calculated. The table has one row per source and should have at least columns entitled 'Id', 'ra' and 'dec'. The structure of the gaia region file is specified in reference [1].

## 8 Algorithm

```
Pass input source table through selection expression;

if (usecorr) {
  ra  = readColumn(RA_CORR);
  dec = readColumn(DEC_CORR);
} else {
  ra  = readColumn(RA);
  dec = readColumn(DEC);
}

Convert ra, dec to pixel coordinates xima and yima;

# Calculate radii:
foreach i (1 .. numOfRows(sourceTable)) {
  protoRadius(i) = evaluateRadiusExpression();
}

protoMedian = medianValue(protoRadius);
scaledMedian = (protoMedian - minval(protoRadius))
 / (maxval(protoRadius) - minval(protoRadius));

sag = (scaledMedian - radiusbias)
 / (radiusbias * (2 * scaledMedian - 1) - scaledMedian);

foreach i (1 .. numOfRows(sourceTable)) {
  scaledRadius(i) = (protoRadius(i) - minval(protoRadius))
```

```
   / (maxval(protoRadius) - minval(protoRadius));

  radius(i) = scaledRadius(i) * (1 + sag)
   / (1 + sag * (2 * scaledRadius(i) - 1));
}

# Sort the sources:
foreach i (1 .. numOfRows(sourceTable)) {
  index(i) = i
}
Sort index in the style requested by --ncutsortstyle;

if (--ncutsortstyle eq 'none') {
  numLastSource = numOfRows(sourceTable);
} else {
  numLastSource = min(numOfRows(sourceTable), --ncut);
}

# Allocate source labels;

# Output:
foreach i (1 .. numLastSource) {
  writeOutputCircle(xima(index(i)), yima(index(i)), radius(index(i)));
}
```

## 9 Comments

-

## References

[1] A. C. Davenhall. *Writing Catalogue and Image Servers for GAIA and CURSA*. CCLRC/Rutherford Appleton Laboratory, 1 edition, July 2000. Found at the URL: `http://www.starlink.rl.ac.uk/star/docs/ssn75.htx/ssn75.html`.

[2] N. Gray P. W. Draper and D. S. Berry. *GAIA - Graphical Astronomy and Image Analysis Tool*. CCLRC/Rutherford Appleton Laboratory, 2.6 edition, September 2001. Found at the URL: `http://www.starlink.rl.ac.uk/star/docs/sun214.htx/sun214.html`.

[3] SAO/HEAD. *DS9 Reference Manual*, 2.1 edition, April 2002. Found at the URL: `http://hea-www.harvard.edu/RD/ds9/doc/v2.1/ref/ref.html`.