colsmooth

April 16, 2023

Abstract

This task smoothes a column of a fits file.

1 Instruments/Modes

Instrument	Mode
Any	Any

2 Use

pipeline processing	no
interactive analysis	yes

3 Description

3.1 Introduction.

The task convolves a specified column of a fits file by a user-specified convolving function. This can be used to smooth the values in the column. Specifically, for input vector x defined from $i = 1$ to $i = N$ and convolver c defined from $-M$ to M, **colsmooth** performs the following transformation:

$$y_i = \frac{\sum_{j=-M}^{M} F_{i,j}}{\sum_{j=-M}^{M} G_{i,j}} \tag{1}$$

where

$$F_{i,j} = \begin{cases} c_j \; x_{i-j} & \text{if } i - j \in [1, N] \\ 0 & \text{else} \end{cases} \tag{2}$$

and the weight term $G_{i,j}$ is given by

$$G_{i,j} = \begin{cases} c_j & \text{if } i-j \in [1,N] \\ 0 & \text{else} \end{cases}$$

These formulae automatically normalise the convolver c. If this is not desired (indicated by setting the parameter `normalise` to 'no'), the y values are multiplied by $\sum_{j=-M}^{M} c_j$ in addition to the above.

The user may also specify an uncertainty column. The values of this column, which we denote by u, are transformed as follows:

$$v_i = \frac{\sqrt{\sum_{j=-M}^{M} H_{i,j}}}{\sum_{j=-M}^{M} G_{i,j}} \tag{3}$$

The vector v here represents the transformed uncertainty values. The quantity H is given by

$$H_{i,j} = \begin{cases} c_j^2 \; u_{i-j}^2 & \text{if } i-j \in [1,N] \\ 0 & \text{else} \end{cases}$$

and G is as given above. The v values can be 'unnormalised' in the same way as y above.

It is probably as well to remind users that the benefits of smoothing are almost entirely cosmetic. The fundamental amount of information in the column cannot of course be increased by such processing. In fact use of a low-pass filter may be deceptive: the amount of noise may not have decreased very much although the graph looks much better because of the suppression of the more visible high-frequency end of the noise spectrum.

This task is not XMM-specific.

3.2 Convolver.

There are several ways in which the user can specify the convolving function, governed by the parameter `convolvertype`. These are:

- `convolvertype` = 'set': the task then expects the convolver to be contained in the column of a fits table described by parameters `convolvertable` and `convolvercolumn`. The data type of the column must be real32.
- `convolvertype` = 'tophat': this is a convolver which is equal to $1/W$ within W contiguous channels and 0 elsewhere. The task requires W to be specified via the parameter `width`.
- `convolvertype` = 'gaussian': the task reads the sigma value from the parameter of the same name. The convolver is truncated at 5 times sigma either side of the centre of the gaussian.
- `convolvertype` = 'user': in this mode, the user can supply the convolver directly as a list of real numbers via the parameter `convolver`.

The user should pay careful attention to the phase of the convolver. Equation 1 (and all others that include a sum over the convolver values) implies that the number of convolver channels, $2M + 1$, is an odd number, and that the centre of the convolver is located at the centre channel. However, convolvers with an even number of channels are also permitted. In this case the lowest channel of the upper half of the channels is taken to be the centre channel. In other words, the sums in this case extend from $1 - M$ to M for a total number of convolver channels $2M$. To clarify with a couple of examples, calls of the form

```
colsmooth convolvertype=user convolver='0 0 1 0 0'
```

or

```
colsmooth convolvertype=user convolver='0 0 0 1 0 0'
```

will leave unchanged the input values in inset.ds:SPECTRUM column `RATE`, but

```
colsmooth convolvertype=user convolver='1 0 0 0'
```

will left-shift all the input values by 2 channels. (Note that in this last example, the right-most two channels of the output will contain **dal**-type null values. Zero values would probably be better and the task may be changed in future to write zeros in this case.)

The elements of the convolver may not sum to zero. Negative and zero-valued elements should be avoided if filling of holes is desired (see section 3.3.1 below).

Note that, under default settings, no convolution is performed.

3.3 Masking.

It may be desirable to avoid smoothing some parts of the input column x, for example if some values of x are nulls. The mask can be supplied via two different methods, governed by the parameter `masktype`, namely:

- `masktype` = 'set': the task then expects the mask to be contained in the column of a fits table described by parameters `masktable` and `maskcolumn`. The data type of the column must be boolean, and it must have the same number of rows as `incolumn`. The dataset/table can be the same ones as contain `incolumn`.
- `masktype` = 'expression': the task expects then a selection expression to be supplied via the parameter `maskexpression`. The expression should be limited to functions or tests of columns in the input table given by the parameter `intable`.

Note that the default setting of `masktype` is 'none', with obvious effects.

The effect of the mask is to enable smoothing only where the mask is true. Specifically, equation 1 should be re-written as

$$y_i = \frac{\sum_{j=-M}^{M} F_{i,j}}{\sum_{j=-M}^{M} G_{i,j}} \text{ where mask value } m_i \text{ is true, } = \textit{null} \text{ elsewhere.} \quad (4)$$

Equation 3 needs to be similarly altered. The value *null* is a null value written by use of (and therefore recognized by) the appropriate **dal** calls.

3.3.1 Avoiding 'droop' and filling in holes.

The weighting factor G in equations 1 and 3 prevents ugly distortions of the smoothed output y near the ends of the range of y at $i = 1$ and $i = N$. This treatment is also desirable near holes in the mask. To prevent this 'droop' in the output close to holes, equation 2 is modified to the following:

$$F_{i,j} = \begin{cases} c_j \ x_{i-j} & \text{if } i-j \in [1,N] \text{ and mask } m_{i-j} \text{ is true} \\ 0 & \text{else} \end{cases}$$

(The specifications for G and H should be amended in analogous fashion.) It should be pointed out that this cosmetic improvement is achieved at the cost of a little less noise suppression in these channels.

Similar costs are incurred by another appearence-enhancing feature of **colsmooth**, namely the ability to 'paper over' holes in the mask which are significantly smaller than the width M of the convolver. The size of holes to be interpolated is provided via the integer parameter `fillholewidth`. The interpolation is implemented by employing two different masks: let's us call them m and m'. The one (m) that appears in equation 3.3.1 and the analogous equations for G and H is left unchanged. The other (m'), which appears in equation 4 and the analogous equation for uncertainties v, is for the most part set equal to m; the exception being within stretches of contiguous $m = \mathit{false}$ channels of length=`fillholewidth`: m' within such stretches is set to 'true'. The result is that y (and v) within these stretches or gaps is calculated as a weighted sum of only those values of x that fall outside the gap. The results are acceptable provided the sum of those parts of the convolver that fall outside the gap onto the valid parts of x is not too small a fraction of the total sum of the convolver.

Note that the default setting of `fillholewidth`=0 disables any hole-filling.

3.4 Output.

If `withouttable` = 'yes', the task attempts to write the smoothed values to columns `outcolumn` and `outerrorscolumn` in the set `outtable`. Otherwise the smoothed values are written over the top of the input column values. Values (and uncertainties) for which the mask is false (but only if the gap is larger than `fillholewidth`) are written as nulls.

4 Parameters

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

Parameter	Mand	Type	Default	Constraints

intable	yes	dataset	inset.ds:SPECTRUM	

The name of the fits dataset and table that contains the values to be smoothed.

incolumn	yes	string	RATE	

The name of the column in `intable` that contains the values to be smoothed.

withouttable	no	boolean	no	yes—no

If 'yes', the task attempts to write the smoothed values to column `outcolumn` (and, optionally, `outerrorscolumn`) of dataset `outtable`; if 'no', the task overwrites the values in `incolumn`.

outtable	no	dataset	outset.ds:SPECTRUM	

The name of the fits dataset and table which will contain the smoothed values.

outcolumn	no	string	RATE	

The name of the column in `outtable` which will contain the smoothed values.

witherrors	no	boolean	no	yes—no

If 'yes', the task looks for uncertainties in `incolumn` in the column designated by `inerrorscolumn`.

inerrorscolumn	no	string	STAT_ERR	

The name of the column in `intable` that contains the uncertainties to `incolumn`. Only read if `witherrors`='yes'.

outerrorscolumn	no	string	STAT_ERR	

The name of the column in `outtable` which will contain the uncertainties to the smoothed values in `outcolumn`. Only written if `witherrors`='yes'.

convolverstyle	no	string	tophat	set—tophat—gaussian—user

Mode of entering the convolving function.

convolvertable	no	dataset	convolver.ds:CONV	

If `convolvertype` = 'set', the names of the dataset+table which contain the convolver are read from this parameter.

convolvercolumn	no	string	CONV	

If `convolvertype` = 'set', the name of the column which contains the convolver is read from this parameter.

width	no	integer	1	1 <=`width`

If `convolvertype` = 'tophat', this parameter gives the width in channels of the top hat.

sigma	no	real	1.0	0 <`sigma`

If `convolvertype` = 'gaussian', this parameter gives the characteristic width of the gaussian.

convolver	no	real list	'1.0'	

If `convolvertype` = 'user', the task reads the convolver directly from the list of real numbers contained in this parameter.

normalize	no	boolean	yes	yes—no

If 'yes', the convolving function is normalised (ie, divided by the sum of all its values).

maskstyle	no	string	none	set—expression—none

Mode of entering the mask vector.

masktable	no	dataset	mask.ds:MASK	

If `masktype` = 'set', the names of the dataset+table which contain the mask are read from this parameter.

maskcolumn	no	string	MASK	

If `masktype` = 'set', the name of the column which contains the mask is read from this parameter.

maskexpression	no	string		

If `masktype` = 'expression', the task reads a selection expression from this parameter. Those rows of `intable` for which the expression evaluates to true cause the corresponding elements of the mask vector to be also set to true.

fillholewidth	no	integer	0	0 <=`fillholewidth`

Gaps in the mask of this width or less are interpolated.

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.

badConvolverType *(error)*
The value of the `convolvertype` is not recognised.

badMaskType *(error)*
The value of the `masktype` parameter is not recognised.

convolverTooSmall *(error)*
The convolver size is less than 1 channel.

zeroConvolverNorm *(error)*
The convolver elements sum to zero.

zerosInConvolver *(error)*
Bad status return from convolution algorithm, indicating the convolver contains some negative- or zero-valued channels. This error should only occur if `fillholewidth` has been set to some value greater than zero.

mustLimitInterpolation *(warning)*
The user has requested via the parameter `fillholewidth` that holes in the mask up to this size be interpolated. However the task has found that the convolver is not wide enough to permit this size hole to be interpolated.
corrective action: The value of the maximum hole to be filled is truncated accordingly.

6 Input Files

1. (Mandatory) a FITS file containing a binary table with a column containing the values to be smoothed. This may optionally contain uncertainties in another column. The data type of either column may be int8, int16, int32, real32 or real64.
2. (Optional) a FITS file containing a binary table with a column containing the convolver values. The data type of the column may be int8, int16, int32, real32 or real64. The number of rows of this table need not be the same as the number of rows of the input table.
3. (Optional) a FITS file containing a binary table with a boolean column containing mask values. This must have the same number of rows as the input table, indeed the mask column may be a column in that table.

7 Output Files

1. (Optional) If `withouttable` is 'yes', the smoothed output is written to this file. It will be a FITS file containing a binary table with a real32 column containing the smoothed values, and (optionally) another real32 column containing the uncertainties to the smoothed values. If `withouttable` = 'no', the input columns will be overwritten with smoothed values. In this case the original data type and all other contents of the input file are retained.

8 Algorithm

```
&readParameters;
convolver = &getConvolver;
(values, uncerts) = &getInSet;
mask = &getMask;

noHoleMask = mask;
foreach (hole in mask) {
  if (hole width <= fillholewidth) {
    noHoleMask = true;
    values = 0.0;
  }
}

if (normalise) {
  norm = 1.0;
} else {
  norm = sum(convolver);
}

smoothedValues = 0.0;
for (n = 1 to vectorSize) {
next if (!noHoleMask(n));
  summ = 0.0;
  weight = 0.0;
```

```
    for (i = 1 to convolverSize) {
      j = n - (i - 1 - convolverSize / 2)
    next if (j < 1 || j > vectorSize || !mask(j));
      summ = summ + convolver(i) * values(j);
      weight = weight + convolver(i);
    }
    smoothedValues(n) = summ * norm / weight;
  }

  if (witherrors) {
    smoothedUncerts = 0.0;
    for (n = 1 to vectorSize) {
    next if (!noHoleMask(n));
      summ = 0.0;
      weight = 0.0;
      for (i = 1 to convolverSize) {
        j = n - (i - 1 - convolverSize / 2)
      next if (j < 1 || j > vectorSize || !mask(j));
        summ = summ + convolver(i) * convolver(i) * values(j) * values(j);
        weight = weight + convolver(i);
      }
      smoothedUncerts(n) = sqrt(summ) * norm / weight;
    }
  }

  &writeFitsOutput(smoothedValues, smoothedUncerts, noHoleMask);
  if (withPlotOutput) {
    &writePlotOutput(smoothedValues, smoothedUncerts, noHoleMask);
  }
```

9 Comments

- Note that the ftool fsumrows can be used to perform a top-hat smoothing.

References