# XMM-Newton Data Analysis Workshop

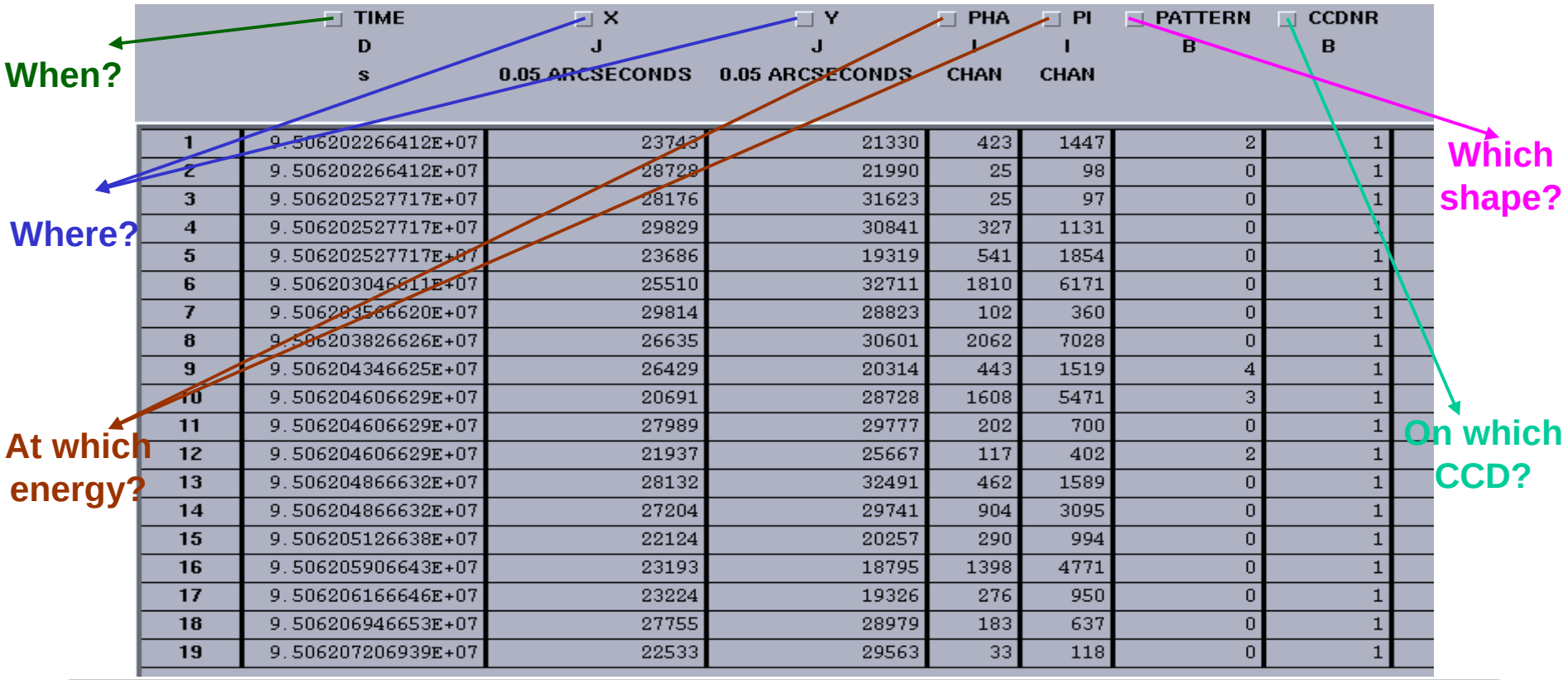
27th November 2001

## Event list manipulation and screening

Matteo Guainazzi
XMM-Newton Science Operation Center

# Event lists

**`e[pm]proc` and `e[pm]chain` produce calibrated and concatenated *event lists*.**
**Each event is individually time-tagged, and its spatial, energy … properties are registered**




| | TIME<br>D<br>s | X<br>J<br>0.05 ARCSECONDS | Y<br>J<br>0.05 ARCSECONDS | PHA<br>I<br>CHAN | PI<br>I<br>CHAN | PATTERN<br>B | CCDNR<br>B |
|---|---|---|---|---|---|---|---|
| 1 | 9.506202266412E+07 | 23743 | 21330 | 423 | 1447 | 2 | 1 |
| 2 | 9.506202266412E+07 | 28728 | 21990 | 25 | 98 | 0 | 1 |
| 3 | 9.506202527717E+07 | 28176 | 31623 | 25 | 97 | 0 | 1 |
| 4 | 9.506202527717E+07 | 29829 | 30841 | 327 | 1131 | 0 | 1 |
| 5 | 9.506202527717E+07 | 23686 | 19319 | 541 | 1854 | 0 | 1 |
| 6 | 9.506203046611E+07 | 25510 | 32711 | 1810 | 6171 | 0 | 1 |
| 7 | 9.506203586620E+07 | 29814 | 28823 | 102 | 360 | 0 | 1 |
| 8 | 9.506203826626E+07 | 26635 | 30601 | 2062 | 7028 | 0 | 1 |
| 9 | 9.506204346625E+07 | 26429 | 20314 | 443 | 1519 | 4 | 1 |
| 10 | 9.506204606629E+07 | 20691 | 28728 | 1608 | 5471 | 3 | 1 |
| 11 | 9.506204606629E+07 | 27989 | 29777 | 202 | 700 | 0 | 1 |
| 12 | 9.506204606629E+07 | 21937 | 25667 | 117 | 402 | 2 | 1 |
| 13 | 9.506204866632E+07 | 28132 | 32491 | 462 | 1589 | 0 | 1 |
| 14 | 9.506204866632E+07 | 27204 | 29741 | 904 | 3095 | 0 | 1 |
| 15 | 9.506205126638E+07 | 22124 | 20257 | 290 | 994 | 0 | 1 |
| 16 | 9.506205906643E+07 | 23193 | 18795 | 1398 | 4771 | 0 | 1 |
| 17 | 9.506206166646E+07 | 23224 | 19326 | 276 | 950 | 0 | 1 |
| 18 | 9.506206946653E+07 | 27755 | 28979 | 183 | 637 | 0 | 1 |
| 19 | 9.506207206939E+07 | 22533 | 29563 | 33 | 118 | 0 | 1 |

# Browsing an event list I: SAS and FTOOLS

**Event lists (as most of the XMM-Newton data) are FITS files, which can be manipulated with FTOOLS/LHEASOFT, alongside with specific SAS tasks:**

- **dump FITS files to ASCII:**

```
fdump infile=file.fits outfile=file.asc columns=- rows=-
```

- **visualise header keywords *(attributes):***

```
fkeyprint infile=file.fits keynam=KEYWORD outfile=STDOUT
```

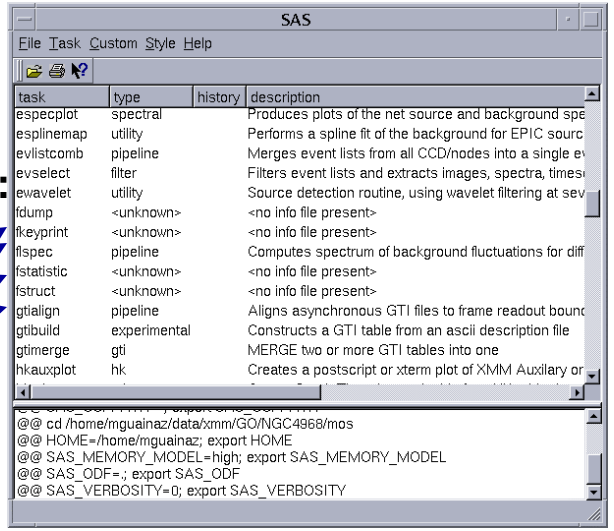
- **show the structure of a FITS file:**

```
fstruct infile=file.fits
```

- **calculate statistics on the table of a FITS file extension:**

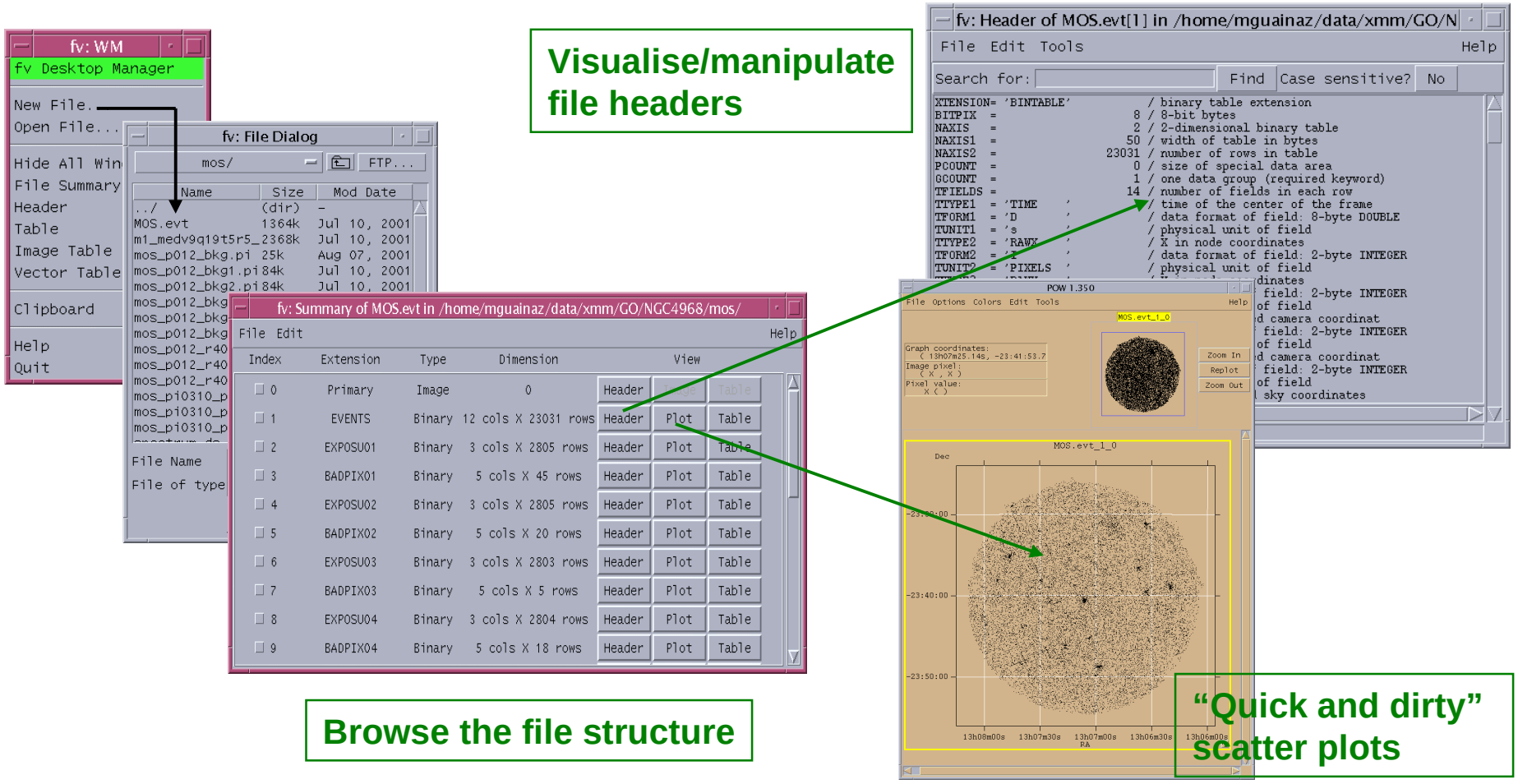
```
fstatistic infile=file.fits colname=COLUMN
```

**SAS provides a GUI interface to run these and other LHEASOFT tasks**



# Browsing and event list files II: “fv”

**Event files can be browsed with the GUI LHEASOFT interface fv as well:**



# SAS specific manipulation tasks (I)

**SAS provides also a set of specific tools to manipulate XMM-Newton FITS files, based on a specific library: the DAL (Data Access Layer).**

| | | FTOOLS | |
|---|---|---|---|
| addattribute | Add an attribute to a dataset | `fmodhead` | PostScript ChangeLog |
| dsaddarray | Add an array to a dataset | `fimgcreate` | PostScript ChangeLog |
| dsaddcolumn | Add a column to a table | | PostScript ChangeLog |
| dsaddcomment | Add a comment to an attributable object | | PostScript ChangeLog |
| dsaddhistory | Add a history record to an attributable object | | PostScript ChangeLog |
| dsaddrows | Add a range of rows to a table | | PostScript ChangeLog |
| dsaddtable | Add a table to a dataset | `fcreate` | PostScript ChangeLog |
| dsattr | Get attribute values | `fkeyprint` | PostScript ChangeLog |
| dsconv | Convert columns that contain time stamps or angles to real numbers | | PostScript ChangeLog |
| dscopyattr | Copy a list of attributes to an attributable | `ffilecat` | PostScript ChangeLog |
| dscopyblock | Copy a list of blocks to a dataset | `fextract` | PostScript ChangeLog |
| dscopycolumn | Copy a list of columns to a table | `faddcol` | PostScript ChangeLog |
| dscopyrows | Copy a range of rows in the given table | `flookup` | PostScript ChangeLog |
| dscp | Copy an object | | PostScript ChangeLog |
| dscreatedataset | Create a dataset | `fcreate` | PostScript ChangeLog |
| dsdeletenullvalue | Delete the null value from an array or column | | PostScript ChangeLog |
| dshead | ASCII dump of first part of an object | `fdump prdata=no` | PostScript ChangeLog |

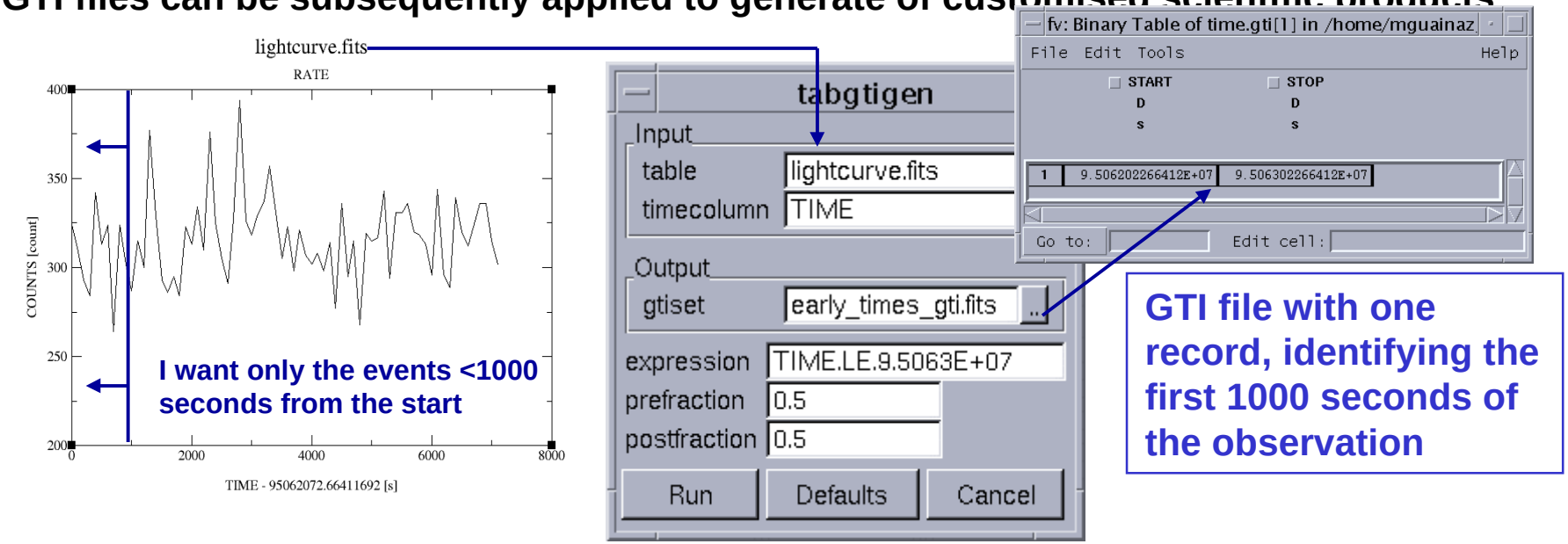
# SAS specific manipulation tasks (II)

| | | | |
|---|---|---|---|
| dsinfo | Retrieve information from object | FTOOLS | PostScript ChangeLog |
| dsinserttable | Insert one table (source) into another (destination) | fmerge | PostScript ChangeLog |
| dslatts | List the attributes in the given attributable objects | | PostScript ChangeLog |
| dslcols | List the columns in the given tables and/or or datasets | flcol | PostScript ChangeLog |
| dsls | List the datasets in the given directory | | PostScript ChangeLog |
| dsmv | Move an object | | PostScript ChangeLog |
| dsnullify | Set all data elements in an object to null | | PostScript ChangeLog |
| dspurify | Purify a dataset | | PostScript ChangeLog |
| dsrelabel | Relabel an object | | PostScript ChangeLog |
| dsrename | Rename an object | | PostScript ChangeLog |
| dsreplacenulls | Replace the null values in an object with a new value | | PostScript ChangeLog |
| dsreshape | Reshape the dimensions of an array or column | chimgtyp, fcollen | PostScript ChangeLog |
| dsrm | Delete a list of objects | fdel... | PostScript ChangeLog |
| dsrmattr | Delete a list of attributes from an attributable | fmodhead | PostScript ChangeLog |
| dsrmrows | Remove a range of rows from a table | fdelrow | PostScript ChangeLog |
| dssetarrayelement | Set the value of an array element | fparimg | PostScript ChangeLog |
| dssetattr | Set/Add an attribute | fmodhead | PostScript ChangeLog |
| dssetcolumnelement | Set the value of a column element | fpartab | PostScript ChangeLog |
| dssetdata | Copy an object's data to another object | | PostScript ChangeLog |
| dssetlabel | Set the label of an array, table, column or attribute | | PostScript ChangeLog |
| dssetnullvalue | Set the null value of an array or column | | PostScript ChangeLog |
| dssetunits | Set the units of an array, column or attribute | | PostScript ChangeLog |
| dsstats | Produce dataset statistics | fstatistics | PostScript ChangeLog |
| dsstruct | Get the structure of a list of datasets | | PostScript ChangeLog |
| dstail | ASCII dump of last part of an object | fdump | PostScript ChangeLog |
| dstranstype | Convert the datatype of a list of objects | | PostScript ChangeLog |
| dsvalidate | Check a dataset. | fverify | PostScript ChangeLog |
| dsverify | Check a dataset. | | PostScript ChangeLog |

# The concept of Good Time Intervals (GTI)

**One defines Good Time Interval file the set of time intervals where a given scientific product (*e.g.:* an event list) is accumulated. EPIC event lists have one GTI extension for each chip. They play a crucial role in the calculation of the exposure times, or to remove high particle background phases.**

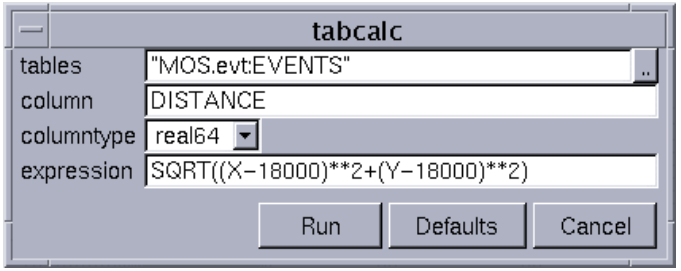
**GTIs can be generated with the SAS task `tabgtigen` from HK or scientific light curves. GTI files can be subsequently applied to generate of customised scientific products**



# Manipulating the event list columns

**Event list columns can be algebraically manipulated to produce new or to modify existing columns with the SAS task `tabcalc`. Examples:**

**1. Generation of a column containing the DISTANCE from a given pixel [in the example: (18000, 18000) in sky coordinates]**




**2. Generation of a new TIME column, where times are expressed as seconds from the observation start:**


| PATTERN B | CCDNR B | DISTANCE D | TIMESTAR D |
|---|---|---|---|
| 2 | 1 | 6.638595408669E+03 | 0.000000000000E+00 |
| 0 | 1 | 1.144596365537E+04 | 0.000000000000E+00 |
| 0 | 1 | 1.700403202185E+04 | 2.613056555390E+00 |
| 0 | 1 | 1.745899544647E+04 | 2.613056555390E+00 |
| 0 | 1 | 5.836981839958E+03 | 2.613056555390E+00 |
| 0 | 1 | 1.651707059378E+04 | 7.801989525557E+00 |
| 0 | 1 | 1.602210738324E+04 | 1.300208248198E+01 |
| 0 | 1 | 1.527574633201E+04 | 1.560213899612E+01 |
| 4 | 1 | 8.740860197944E+03 | 2.080213196576E+01 |
| 3 | 1 | 1.106035555486E+04 | 2.340216849744E+01 |
| 0 | 1 | 1.544272806210E+04 | 2.340216849744E+01 |
| 2 | 1 | 8.618750373459E+03 | 2.340216849744E+01 |

# `selectlib:` a selection/manipulation library

**All the operations to manipulate tables and columns in an EPIC event lists are driven by the `selectlib` library. Examples of the allowed operations:**

- **boolean: “==“, “>”, “<=“, “||”, “&&”, “!” ... E.g.: `(CCDNR==1)&&(PHA>=300)`**
- **arithmetic/trigonometric: “+”, “abs(x)”, “sin(x)”, “log(x)” … E.g.: `(log(PI)>0)`**
- **string manipulation: “upper/lower”, “=“, “>”, “+”, “ascii” … E.g.: `'W' + ' XMM'` ⇒ `'W XMM'`**
- **definition of a selection expression as a keyword. E.g.: `#DISTANCE < 128` if a keyword `DISTANCE == SQRT((X-18000)**2+(Y-18000)**2)` exists in a to-be-screened file**
- **bitwise (BW) operators: “BW AND/OR”, “left/right shift”**
- **built-in constants: “#PI”, “RAD”, “#E”, “TRUE/FALSE” … E.g.: `PATTERN>#PI`**

# Region filters function

**In order to facilitate the extraction of scientific products in spatial regions, a number of pre-defined selection regions are available in `selectlib`:**

- `point(x0,y0,Xcolumn,Ycolumn)`
- `line(x0,y0,x1,y1,Xcolumn,Ycolumn)`
- `circle(xCenter,yCenter,radius,Xcolumn,Ycolumn)`
- `sector(xCenter,yCenter,fromAngle,toAngle,Xcolumn,Ycolumn)` or `pie(xCenter,yCenter,fromAngle,toAngle,Xcolumn,Ycolumn)`
- `ring(xCenter,yCenter,radius1,radius2,Xcolumn,Ycolumn)` or `annulus(xCenter,yCenter,radius1,radius2,Xcolumn,Ycolumn)`
- `ellipse(xCenter,yCenter,xHalfWidth,yHalfWidth,rotation,Xcolumn,Ycolumn)`
- `elliptannulus(xCenter,yCenter,xHalfWidthInner,yHalfWidthInner xHalfWidthOuter,yHalfWidthOuter,rotationInner,rotationOuter,Xcolumn,Ycolumn)` or `elliptring(xCenter,yCenter,xHalfWidthInner,yHalfWidthInner xHalfWidthOuter,yHalfWidthOuter,rotationInner,rotationOuter,Xcolumn,Ycolumn)`
- `box(xCenter,yCenter,xHalfWidth,yHalfWidth,rotation,Xcolumn,Ycolumn)`
- `rectangle(xLoLeft,yLoLeft,xUpRight,yUpRight,rotation,Xcolumn,Ycolumn)`
- `rhombus(xCenter,yCenter,xHalfWidth,yHalfWidth,rotation,Xcolumn,Ycolumn)` or `diamond(xCenter,yCenter,xHalfWidth,yHalfWidth,rotation,Xcolumn,Ycolumn)`
- `polygon(x1,y1,x2,y2,x3,y3,x4,y4,...,Xcolumn,Ycolumn)`

**Example: to select all events within 128 pixels from the sky pixel (18000, 18000):**

```
circle (18000, 18000, 128, X, Y)
```

# File-based filters

**Three file-based filters exist within `selectlib`:**

- **GTI-filter**: `gti(gti.fits, TIME)` **selects all the events, whose TIME belongs to at least on of the GTIs defined in `gti.fits` (assuming that the event list time column is `TIME`)**

- **Mask filter**: `mask(mask.fits, X0, Y0, X, Y)` **selects all the events which fall on a position `[(X0-X),(Y0-Y)]`, whose corresponding mask value is non-zero. It can be applied to sky coordinates positions as well, if the mask contains WCS information**

- **Region filter**: `region(region.fits, X, Y)` **selects all the events whose position (in sky pixels in this case) belongs to `region.fits`**

# IN-operator

**A generic operator family exists, which allows expressions in the form:**

```
arith in (...)
```

- **IN-intervals:**

| interval specification | alternative expression | meaning |
|---|---|---|
| : or ⟨:] or [:) or ⟨:⟩ | true | $x = val$ |
| $val$ or [$val$] | $val == x$ | $val <= x$ |
| $val$: or [$val$:] or [$val$:⟩ | $val <= x$ | $val < x < +\infty$ |
| ⟨$val$:] or ⟨$val$:⟩ | $val < x$ | $-\infty < x <= val$ |
| :$val$ or [:$val$] or ⟨:$val$] | $val >= x$ | $-\infty < x < val$ |
| [:$val$⟩ or ⟨:$val$⟩ | $val > x$ | $lo <= x <= hi$ |
| $lo$:$hi$ or [$lo$:$hi$] | $lo <= x$ && $hi >= x$ | $lo < x <= hi$ |
| ⟨$lo$:$hi$] | $lo < x$ && $hi >= x$ | $lo <= x < hi$ |
| [$lo$:$hi$⟩ | $lo <= x$ && $hi > x$ | $lo < x < hi$ |
| ⟨$lo$:$hi$⟩ | $lo < x$ && $hi > x$ | >= |

**Example:**

```
PI in [100, 300)
```

**is the same as:**

```
(PI=>100)&&(PI<300)
```

- **IN-GTI:** `TIME IN gti(gti.fits)` **is the same as** `gti(gti.fits, TIME)`

- **IN-filter:** `(X, Y) in circle(18000, 18000, 128)` **is the same as**

```
circle (18000, 18000, 128, X, Y)
```

**If you are scared enough, you may ask: do I really need to learn all this stuff to extract my customised scientific products? The answer is no … as it will be shown in the next presentation.**