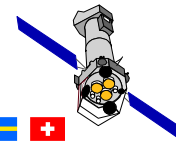

Event lists manipulation and screening

9th ESAC SAS Workshop

June 29 – July 3, 2009

Ignacio de la Calle on behalf of Matthias Ehle
XMM-Newton Science Operations Centre



XMM-Newton

EPIC event lists

e[m/p]proc produces **calibrated & concatenated event lists** (also in pipeline products). Each event is individually time-tagged, and its spatial, energy ... properties are registered.

	TIME D s	X J 0.05 arcsec	Y J 0.05 arcsec	PHA I channel	PI I eV	PATTERN B	CCDNR B
1	7.939931837937E+07	37851	33832	32	710	225	1
2	7.939931837937E+07	32875	26997	72	375	0	1
3	7.939931997292E+07	39505	31888	175	1575	2	1
4	7.939932037132E+07	28673	23282	673	4235	3	1
5	7.939932176569E+07	39469	32907	39	525	78	1
6	7.939932176569E+07	39294	32728	26	330	78	1
7	7.939932176569E+07	32099	27429	1578	8050	0	1
8	7.939932355842E+07	29135	24923	642	3525	2	1
9	7.939932435520E+07	39095	31376	54	265	0	1
10	7.939932435520E+07	37947	31409	76	595	1	1
11	7.939932475360E+07	29632	21073	807	4440	0	1
12	7.939932495279E+07	40686	30086	1924	9880	0	1
13	7.939932614795E+07	39385	31144	135	675	0	1
14	7.939932614795E+07	30534	23569	613	3800	2	1
15	7.939932754233E+07	32366	25941	122	980	3	1
16	7.939932813992E+07	37047	29179	1075	6010	2	1
17	7.939933053024E+07	36413	30314	124	620	0	1
18	7.939933112783E+07	37099	28445	291	1510	0	1
19	7.939933132702E+07	31470	23443	211	1155	0	1

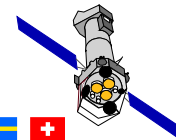
When? (points to TIME column)

Where? (points to X and Y columns)

At which energy? (points to PHA and PI columns)

Which shape? (points to PATTERN column)

On which CCD? (points to CCDNR column)



XMM-Newton

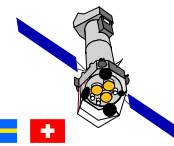
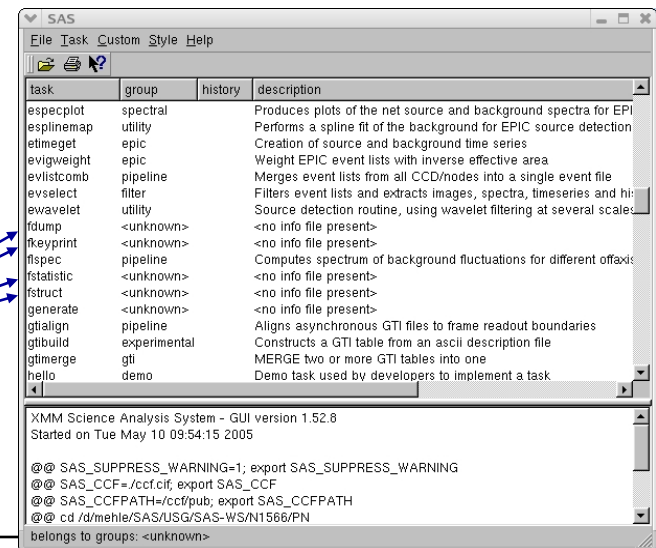
Matthias Ehle

Browsing an event list: 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 a column of a FITS file:
`fstatistic infile=file.fits colname=COLUMN`

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



XMM-Newton

Matthias Ehle

Browsing an event list: "fv"

Event files can also be browsed with an ftools Graphical User Interface (GUI): **fv**

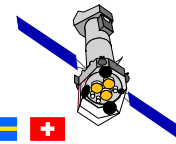
The image shows a composite screenshot of the fv GUI. On the left is the main menu with options like 'New File...', 'Open File...', 'SkyView...', 'Catalogs...', 'VizieR...', 'Run Ftool...', 'Display Device', 'Hide All Windows', 'File Summary', 'Header', 'Table', 'Image Table', 'Vector Table', 'Preference', 'Clipboard', 'Help', and 'Quit'. In the center is a 'fv: File Dialog' window showing a list of files with columns for Name, Size, and Mod Date. Below it is a 'fv: Summary of 0106_0110980401_EMOS1_S001_ImagingEvs.ds' window with a table of file headers. On the right is a 'fv: Histogram' window showing a 2D histogram plot of DEC (deg) vs RA (deg) with various parameters like TLMin, TLMax, Data Min, Data Max, Min, Max, Bin Size, and Row Range. A 1D histogram is also visible at the bottom of the histogram window.

Visualise/manipulate file headers

Browse the file structure

"Quick and dirty" scatter plots

Create/visualise 1D or 2D Histograms



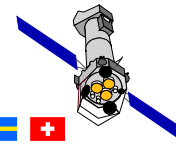
XMM-Newton

Matthias Ehle

SAS specific manipulation tasks (I)

SAS provides 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	<code>fmodhead</code>	PostScript ChangeLog
dsaddarray	Add an array to a dataset	<code>fimgcreate</code>	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	<code>fcreate</code>	PostScript ChangeLog
dsattr	Get attribute values	<code>fkeyprint</code>	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	<code>ffilecat</code>	PostScript ChangeLog
dscopyblock	Copy a list of blocks to a dataset	<code>fextract</code>	PostScript ChangeLog
dscopycolumn	Copy a list of columns to a table	<code>faddcol</code>	PostScript ChangeLog
dscopyrows	Copy a range of rows in the given table	<code>flookup</code>	PostScript ChangeLog
dscp	Copy an object		PostScript ChangeLog
dscreatedataset	Create a dataset	<code>fcreate</code>	PostScript ChangeLog
dsdeletenullvalue	Delete the null value from an array or column		PostScript ChangeLog
dshead	ASCII dump of first part of an object	<code>fdump prdata=no</code>	PostScript ChangeLog



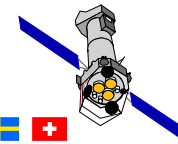
XMM-Newton

Matthias Ehle

SAS specific manipulation tasks (II)

FTOOLS

dsinfo	Retrieve information from object		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.		PostScript ChangeLog
dsverify	Check a dataset.	fverify	PostScript ChangeLog



XMM-Newton

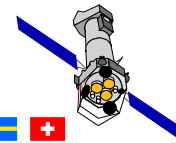
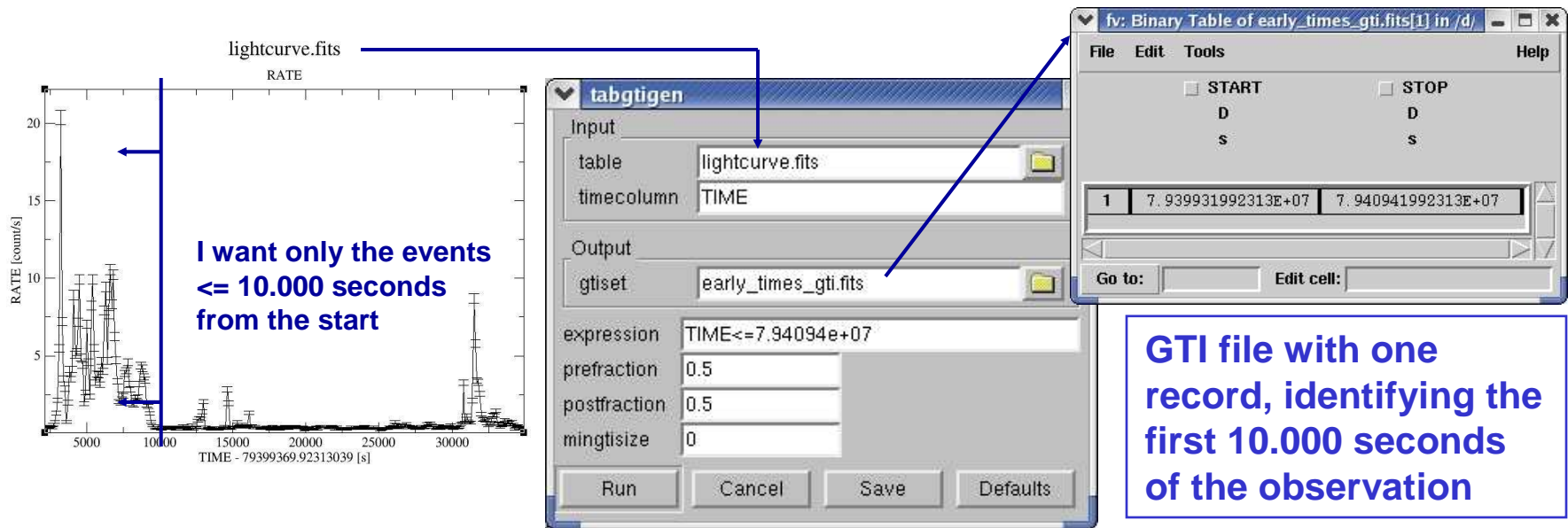
Matthias Ehle

The concept of Good Time Intervals (GTI)

Good Time Intervals: a set of time intervals where a given scientific product (e.g.: an event list) is accumulated.

EPIC event lists: one GTI extension for each chip. Important in calculation of **exposure times**, or to **remove high particle background phases**.

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



XMM-Newton

Matthias Ehle

Manipulating 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:

The screenshot shows the `tabcalc` dialog box with the following configuration:

- tables: MOS1_Evts.ds:EVENTS
- column: DISTANCE
- columntype: real64
- columnunit: unit
- columnlabel: label
- expression: $\text{SQRT}((X-18000)**2+(Y-18000)**2)$

Below the dialog box, a table displays the results for the DISTANCE and TIMESTAR columns. The DISTANCE column is checked, and the TIMESTAR column is also checked. The table shows the following data:

	PATTERN	CCDNR	DISTANCE	TIMESTAR
	B	B	D	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	

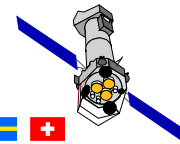
The screenshot shows the `tabcalc` dialog box with the following configuration:

- tables: MOS1_Evts.ds:EVENTS
- column: TIMESTAR
- columntype: real64
- columnunit: unit
- columnlabel: label
- expression: $\text{TIME}-9.506202266411692\text{E}+07$

selectlib: a selection/manipulation library

All operations to manipulate tables and columns in an XMM-Newton event list are driven by the **selectlib** library. Examples of 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`



XMM-Newton

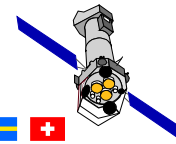
Matthias Ehle

Functions for spatial filters

In order to facilitate 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)`



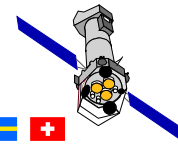
XMM-Newton

Matthias Ehle

File-based filters

Three file-based filters exist within `selectlib`:

- **GTI-filter:** `gti(gtifits, TIME)`: selects all events, whose `TIME` belongs to at least one of the GTIs defined in `gtifits` (assuming that `TIME` is the event list time column)
- **Mask filter:** `mask(maskfits, X0, Y0, X, Y)`: selects all 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, if the mask contains WCS information
- **Region filter:** `region(regionfits, X, Y)`: selects all events whose position (in sky pixels in this case) belongs to `regionfits` (cf. McDowell & Rots: FITS Region Binary Table Design)



XMM-Newton

Matthias Ehle

IN-operator

A generic operator family exists, allowing expressions of form `arith in (...)`

interval specification	alternative expression	meaning
<code>:</code> or <code>(:)</code> or <code>[:)</code> or <code>(:)</code>	<code>true</code>	$-\infty < x < +\infty$
<code>val</code> or <code>[val]</code>	<code>val == x</code>	$x = val$
<code>val:</code> or <code>[val:]</code> or <code>[val:]</code>	<code>val <= x</code>	$val \leq x < +\infty$
<code>(val:]</code> or <code>(val:]</code>	<code>val < x</code>	$val < x < +\infty$
<code>: val</code> or <code>[: val]</code> or <code>(: val]</code>	<code>val >= x</code>	$-\infty < x \leq val$
<code>[: val)</code> or <code>(: val)</code>	<code>val > x</code>	$-\infty < x < val$
<code>lo: hi</code> or <code>[lo: hi]</code>	<code>lo <= x && hi >= x</code>	$lo \leq x \leq hi$
<code>(lo: hi]</code>	<code>lo < x && hi >= x</code>	$lo < x \leq hi$
<code>[lo: hi)</code>	<code>lo <= x && hi > x</code>	$lo \leq x < hi$
<code>(lo: hi)</code>	<code>lo < x && hi > x</code>	$lo < x < hi$

Example:

`PI in [100, 300)`

is the same as:

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

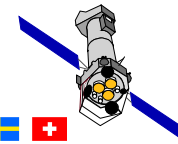
- IN-intervals:**

- 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.



XMM-Newton

Matthias Ehle