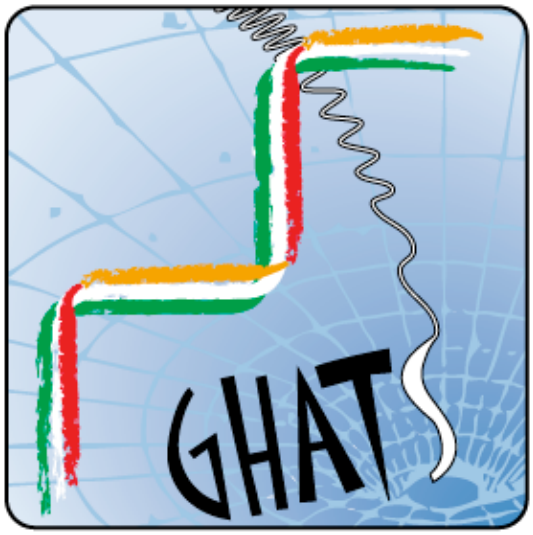
GHATS

USER'S MANUAL

Version 3.2.0 - 24 May 2022

General
High-Energy
Aperiodic
Timing
Software



Author: Tomaso M. Belloni
(INAF - Osservatorio Astronomico di Brera)

NOISE, n. A stench in the ear. Undomesticated music. The chief product and authenticating sign of civilization.

(A. Bierce "The Devil's Dictionary")

It is of a flawless white color, right from its nose to its magnificent tail.

(Mahabharata, The Story of Garuda)

Revision history

VERSION 1.0.1

Fixed uppercase/lowercase for *POWER/FFT* in gh_xte
Fixed crash in *GH_CROSS*
Added *GH_PLOT_LAG*
Added *GH_PLOT_COH*
Fixed wrong stretch time in *GHATS_ALL* output
Fixed wrong stretch time in *GH_XTE* output to terminal and log file

VERSION 1.0.2

GH_CROSS_RANGE added
GH_PLOT_CROSS removed (duplicate of *GH_PLOT_LAG*)
GH_XTE Fixed bug for sliding in turbo mode
GH_XTE Fixed bug for recalculation of number of PDS in non-turbo
GH_XTE Fixed bug: now it finds housekeeping files in turbo mode even if data are elsewhere
GH_XTE Removed calls to *FSC_FILE_BASENAME*, also removed from package for compatibility
GH_XTE Channel range now restricted to available, not claimed to be the same in input
GH_CROSS_RANGE Intercept case when zero or one bins are to be averaged
GH_CROSS/GH_CROSS_RANGE /TIMELAG keyword changed to TLAG for uniqueness
GH_CROSS/GH_CROSS_RANGE Fixed bug for time selection
GH_DYNA Changed default frequency rebinning from -100 to 1 (no rebinning)
GH_COLORS added
GHX/GH_DYN/GH_COLORS/GH_HK/GH_CROSS/GH_CROSS_RANGE fixed bug for times

VERSION 1.1.0

GH_XTE Added windowing option
GH_SWIFT added
GH_HK renamed *GH_HK_XTE*
GH_PLOT_HK renamed *GH_PLOT_HK_XTE*
GH_XTE Increased max number of files to 1000
GH_XTE/GH_SWIFT fixed but for empty data stretch
GH_CROSS Poissonian estimate was valid only for RXTE

Revision history (cont'd)

VERSION 1.1.1

GH_CROSS_RANGE Added output of average cross-spectrum value (complex)
GH_XMM Added
GH_NUSTAR Added

VERSION 2.0.0

GH_CROSS_RANGE	Fixed bug for time lag output
GH_XTE	Fixed bug that limited time rebinning
GH_SWIFT	Fixed bug that limited time rebinning
GH_XMM	Fixed bug that limited time rebinding
GH_NUSTAR	Fixed bug that limited time rebinning
GH_CROSS	Fixed for release of logical units
GH_CROSS	Fixed bug for Poissonian estimate for instruments different from PCA
GH_CROSS_RANGE	Fixed for release of logical units
GH_NUSTAR	Increased maximum number of GTI intervals
GH_PLOT_HK_XTE	Bug fixed
GHX	Bug with file logical unit fixed
GH_XTE/SWIFT/XMM/NUSTAR	Bug with input from command line fixed
GH_LAXPC	Added
GH_CZTI	Added
GH_CROSS_RANGE	Fixed bug for time lag computation
GHX	Indices for SEL option now start from 0 (IDL WHERE compatibility)
GH_OPLOT_NUPOWER	fixed crash
GH_*	Fixed bug for color accumulation in all modules but RXTE/PCA
GH_PLOT_LICU_NOGAP	Added

VERSION 3.2.0

GH_CROSS	Added RATE keyword
GH_CROSS	absolute value check for time difference for file compatibility
GH_CROSS	added /poivalue keyword for real part of Poissonian contribution to be subtracted
GH_CROSS	added keyword for cross-spectrum output
GH_*	added reading of TIMEZERO keyword from GTI files and extensions

Introduction

The need for a timing package

The long-living RossiXTE mission was aimed at observing time variable phenomena from X-ray sources. Its main instrument, a large proportional counter which, in addition to its large collecting area, provided high time resolution with good calibration and knowledge of dead time effects, was designed for the study of fast variability. While the XRONOS package distributed by NASA/HEASARC can provide tools for the timing analysis of periodic phenomena such as pulsars, unfortunately, no comprehensive and efficient tool was provided to the community for the aperiodic timing analysis of RXTE data and none existed before. Each group was left to their own devices and had to develop a custom package.

The background behind GHATS

At INAF-OAB, more than ten years ago, I started writing a package devoted to RXTE/PCA data (the HEXTE instrument, although called High-Energy X-ray Timing Experiment, is almost unusable when it comes to timing). The philosophy behind the software can be summarized in two points:

1. The program had to work directly from RXTE raw files, without forcing the user to produce light curves before the analysis. In other words, the program itself would accumulate its own light curves, which would be discarded at the end.

2. Timing analysis based on Fourier Transforms is essentially based upon two separate steps. The first is the production of Power Density Spectra (PDS) or Fast Fourier Transforms (FFT). The second is the analysis of the products. The first task is essentially an almost non-interactive process: one gives input parameters and the program delivers the final product (there is some interaction as the range of possible parameters must depend on the specific data types and a decision can only be made after reading the input files). The second task is highly interactive: read PDS files, manipulate them, fit them.

Given these starting points, the best solution was to sacrifice some speed and write the software in an integrated (graphical) environment. Although yours truly is not a fan of IDL, which I find extremely cumbersome and incredibly old-fashioned to the point of reflecting what nowadays can only be defined as bad programming, the choice fell upon IDL. The reasons are simple: it's a very widely used package and there are existing libraries which make life easier. The downside of it is that IDL is not a freely available program and must be bought from a commercial source, which effectively limits its availability. However, since years considerable effort is going into the development of the GNU Data Language (GDL), essentially a public-domain clone of IDL. Although it is by far not complete, GDL is at a sufficiently advanced stage to be used for my purposes.

A new package with limited distribution

From all this, a package called MU was born. Initially only used within my institute, it was moderately exported so that it is now used by very selected people in Groningen, Rome, Mumbai and Kolkata. It worked only on RXTE/PCA data and continues to do so. The upcoming ASTROSAT mission provided the incentive to extend it and adapt it to other instruments. Although the package was in a good shape, the data format was too RXTE specific and needed to be changed and made more general.

Introduction

From the old to the new

This is how GHATS was born. Release 1.0.0 had not been used by anyone other than me for testing. Then the package was released. The few users of MU will recognize essentially the same software, with only command names changed. This is because the changes in data format are of course invisible and the extension to other missions are additions to the existing code. In due time, specific extensions to ASTROSAT will be available.

The future

What GHATS needs now is testing by a larger number of users. In particular, the GDL version of MU has been used only by one person until now and certainly will profit from more users.

Installation

The package

At present, the GHATS package can be obtained from INAF-OAB and is not distributed in any way. When you receive it, you cannot distribute it further.

Unpacking

You can unpack the package in your preferred directory. Notice that it contains only ASCII files, as the IDL/GDL distribution is though source code (GDL does not yet allow binary distribution). The tar file will unpack two files and two directories:

- FFT: the directory with the files for FFT production
- ANALYSIS: the directory for the interactive analysis
- ghats: the executable for IDL
- gghats: the executable for GDL

Move the executable you are going to use to a directory in your path and make it executable if it is not already so.

In order to have a running version of GHATS you need to have the following installed on your computer:

Software dependencies

1. An updated and running version of IDL or GDL. Notice that if you find a precompiled GDL for your architecture, you still need the source files as many procedures are external.
2. An updated version of the IDL ASTRON library (from http://idlastro.gsfc.nasa.gov/).
3. An updated version of Craig Markwardt's IDL library (from http://www.physics.wisc.edu/~craigm/idl/idl.html). Not everything is needed, but it is mostly routines which are useful to have handy.
4. An updated version of the IDL_EXTRACT package (from http://idlastro.gsfc.nasa.gov/ftp/contrib/rxte/).

To have GHATS running, you need to customize your ghats/gghats startup file. The standard version that comes with the package contains:

1) export MU_PATH=/Users/belloni/Desktop/GHATS/
Replace with the directory containing your GHATS installation.
2) export IDL_STARTUP=$MU_PATH/ANALYSIS/ghats.pro
3) export IDL_DIR=/Applications/itt/idl/idl80
Replace with the location of your IDL/GDL directory
4) export IDL_PATH=\+$IDL_DIR/lib:\+$MU_PATH:\+/Users/belloni/gdl/src
Here you should export the locations of the IDL (or GDL) library. If you installed the additional libraries in different locations, add those as well.
5) $IDL_DIR/bin/idl
The path of the IDL/GDL executable, which is run from here.

You then invoke GHATS running the appropriate script (for IDL or GDL).

Walkthrough

The data

The starting data are directly products from the instruments. In the case of RXTE data, this means science array and event data (SA and SE). Supported data modes are:

- Binned
- Single Bit
- Event
- GoodXenon (preprocessed)

Binned data contain photons within a range of channels from 0 up, Single Bit within a limited range, Event from a starting channel (which can be 0) up to the maximum. GoodXenon data are unbinned data, so they cover the full channel range. The time resolution and channel resolution of the data depends on the precise data type.

The input to the PDS/FFT production program is either a file, a list of files, or a list of lists of files.

- **Single file.** It must be one of the science PCA files (starting with FS37, FS3b, FS3f or FS4f).
- **List of files (a.k.a. metafile)**. It is an ASCII file containing filenames (with pathnames if necessary), one per line. The files do not have to reside in the same directory or come from the same observation, but they must be homogeneous (same mode, same time resolution, same energy ranges and binning). They should not be simultaneous, but this is not usually possible. Time sorting is not needed. Metafiles are identified by a starting '@' character.
- **List of list of files.** It is an ASCII file containing a list of metafile names, one per line). The metafiles must themselves contain files which are homogeneous and do not overlap either in time or in channel space. The time resolutions do not have to be the same: the lowest one will be used by the program. Metametafiles are identified by a starting '@@'.

EXAMPLE: the observation contains the following files

Binned (B_2ms_4B_0_35_H)
FS37_473ed44-473fa98
FS37_4740390-4741118
FS37_4741b40-4742798

Event: (E_16us_16B_36_1s)
FS3b_473ed44-473fa9e
FS3b_4740390-474111e
FS3b_4741b40-474279e

Single bit: (SB_62us_0_35_500ms)
FS3f_473ed44-473fa9e
FS3f_47403a0-474111e
FS3f_4741b40-474279e

Not usable: (D_1us_0_249_128_1s_F)

The data

The first group of files contains data binned into 2 ms time bins and four channel bins between channels 0 and 35 (a check on the RXTE Appendix E, ftp://legacy.gsfc.nasa.gov/xte/nra/appendix_f/RXTE_tech_append.pdf, will show that the channel boundaries are 0-13, 14-18, 19-25, 26-35).

The second group contains event (photon by photon) data with a time resolution of 16 microseconds and binned in 16 channels starting from channel 36. Therefore, binned and even data together cover the full channel range, but at a non-homogeneous time resolution. It is not possible to sample frequencies higher than 256 Hz below channel 35.

This limitation is solved by the third group of files, that contains data binned into 62 microsecond bins and only one channel between 0 and 35. We lose energy resolution but can go above the kHz.

What we can do is feed the program a single file (you pick), or a file called for instance @sb.lis containing a list of the single-bit files, or a file called @event.lis containing a list of the event files, or a file called @@all.lis containing @sb.lis on the first line and @event.lis on the second. To be specific:

```
$ ls FS3f* > @sb.lis
$ ls FS3b* > @event.lis
$ ls @sb.lis @event.lis > @@all.lis
```

The production program

Before running the software, unfortunately we cannot deal with compressed files. If the FS files above are gzipped, you need to unzip them. Also, you need to unzip a few auxiliary files which are going to be used by the software:

```
$ gzip -d FS4a* FH5a*
```

Start GHATS. We are now ready to start the production program, called **GH_XTE**. It can be called in three separate ways: without arguments, with arguments and with a parameter file. Let us first start with the interactive way, running it without any arguments

```
Ghats> gh_xte
```

If you are running IDL, you can use the /TURBO keyword, which switches to a different FFT engine (IDL_EXTRACT by C. Markwardt) and is faster.

A 'file selection' will appear, which by default will only show relevant files. in Our case, let us choose @event.lis

We are shown the available channel ranges and asked whether we want all of them or a selection. Let us go for all (default).

The production program

We are asked whether we want to produce a PDS, i.e. a file containing power spectra, or an FFT, a file containing Fourier transforms. Unless we plan to work on time lags or coherence function, a PDS is what we want. Let us choose POWER (the default).

Output file name. The default is the same name of the input file (without '@' in this case), with the extension PDS. Such extension is not mandatory, but it helps further defaults. Let us stick with event.pds.

We can rebin our data in time if we are not interested in the highest reachable frequencies. In our case the time resolution of the dataset is 16 microseconds. Since the RXTE clock works in powers of two, this means that we have 16384 time ticks per second. This means we can reach a Nyquist frequency of 8192 Hz, too high for our purposes. Let us rebin by a factor of 4 and go up only to 2048 Hz, sufficient to detect high-frequency features if present.

Next we have to tell the program how many time bins we want for each data stretch. The program will accumulate data until it has reached the required number of consecutive time bins, produce the PDS and store it. If a gap is reached before the number of bins is reached, it will discard the current interval and start again accumulation at the end of the gap. This will ensure that no gaps fall into data stretches. We have two ways to tell the program how long each data stretch should be. The first is to input the exact number of data points per stretch, which requires a calculation (division of the required length by the time resolution). The other is to input the length in time. Only powers of two are accepted in either case. The program discriminates between the two cases by checking whether you input an integer (number of bins) or real (time length) number. The program informs you beforehand how many points are needed for 16-second stretches (16 seconds is the accumulation time for RXTE/PCA spectra data). In our case, 131072 points per 16.0 seconds. Since we want, say, 32 seconds per stretch, we can either input 262144, or opt for 32.0. I never use the first option.

The program starts running. It first presents an overview of what will be done, which is also written in a log file called in this case event.log:

```
------------------------------------------------------------
Program gh_xte     Tue Jan  3 15:26:00 2012    User: belloni
------------------------------------------------------------
Working directory                :
/Users/belloni/data/xte/u1636/realtime/96087-01-84-10/pca
Input filename                   :
/Users/belloni/data/xte/u1636/realtime/96087-01-84-10/pca/@event.lis
Instrument                       : XTE/PCA
Observation ID                   : 96087-01-84-10
Source name                      : 4U_1636-53
Start date of observation        : 2011-12-29T04:01:36

Channels for accumulation        : 0-249
Type of output                   : POWER
Output filename                  : event.pds
Time resolution (s)              : 1/8192
Nyquist frequency (Hz)           : 4096.0000
Number of points per FFT         : 262144
```

The production program

```
Corresponding to length (s)      : 32.000000
Total number of FFTs             : 64
Time step                        : 32.000000
------------------------------------------------------------
```

Overview of the results

We have now produced a file called event.pds. This file contains the series of power density spectra as a function of time, the corresponding count rate for each of them (light curve), the photons from the spectral files in three energy bands (defaults here) and some housekeeping information.

The first thing we can do is have an overview of the content of the file. We can use the command

Ghats> ghats_all,'event.pds'

```
GH_info ------------------------------------------------

PDS file name    :   event.pds

Observatory      :   XTE
Instrument       :   PCA
Target           :   4U_1636-53

N. of trafos     :   64
Trafo duration   :   32.000000
N. of powers     :   262144
Start-end chans.:   0-249
Starting mjd     :   55924.168
Time step        :   0.031250000
Barycentered     :   NO
Spectral bins    :   3
Background       :   NO
------------------------------------------------------------
```

The output includes some of the previous info, plus the last three lines which informs us that the input times were not barycentered, that the spectral bins used were three (default) and that there is no background associated to the PDS file. Well see that later.

What is important is that our file contains a sequence of 64 PDS, each covering 32 seconds of uninterrupted data. You can obtain the same text output with the command

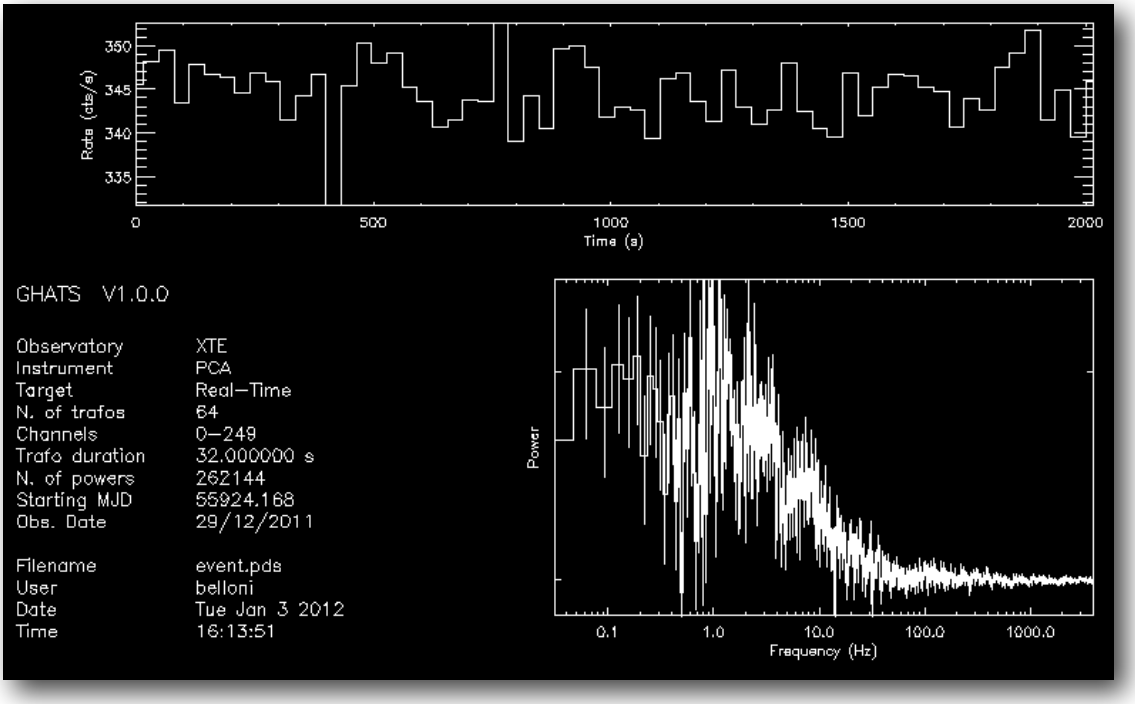
Ghats> gh_info,'event.pds'

An important (and sorry) point for GDL users. If you use GDL, you can do away with typing the input file name, as for all analysis programs. If you use a variable which contains a non-existent filename or even which does not exist, a file selection window will be opened and the filename will be used and stored in that variable for further use. It is very convenient. Unfortunately, this does not work with GDL. The variable might be empty or contain a non-existent filename, but it must exist. To overcome this problem, I suggest you add to the ghats.pro routine (see startup command) a list of empty strings which you can use later.

Walkthrough

ç

The ghats_all command, in addition to the text output, produces a summary plot which shows light curve (one count rate point per PDS) and the averaged PDS.



By default, the Poissonian level is not estimated and subtracted, the PDS rebinning is -100 (see below) and the output goes to the screen. You can have the Poissonian contribution subtracted with the keyword /POISSON, have a different rebinning factor with the keyword REBIN and produce a PostScript file (called ghats.ps) with the keyword /PS.

Light cuve production

We are not happy with just looking at the data summary. We can extract the light curve and plot it with:

```
Ghats> gh_licu,'event.pds',time,licu
Ghats> gh_plot_licu,time,licu
```

PDS extraction (with defaults)

Then we can extract the PDS ourselves, bin it and plot it with:

```
Ghats> ghx,'event.pds',nu,pow,pow_e, /poisson
   64 power spectra selected
```

This reads in the average PDS in three arrays: nu for frequency, pow for power and pow_e for error on the power. The /poisson keyword forces computation and subtraction of the Poissonian component.

```
Ghats> gh_reb,nu,pow,pow_e,-100,x,y,ye
```

This rebins the data with a rebin factor -100 and writes the output to the variables x,y,y_e. Of course the choice of the variable names is arbitrary.

Finally, we can plot it:

```
Ghats> gh_plot_power,x,y,ye
```

Custom PDS extraction

From the overview, we can see that the average count rate is around 340 cts/s, but we can use IDL/GDL to calculate it exactly from the light curve array we extracted:

```
Ghats> print,mean(licu)
    344.632
```

Let us say that we are interested only in averaging the PDS corresponding to count rates larger than the average. We can do it with:

```
Ghats> ghx,'event.pds',nu,pow,pow_e, /poisson,rate=[344.632,10000]
  33 power spectra selected
```

If we want to limit ourselves to the first 500 seconds of data, still selecting only count rates above the average:

```
Ghats> ghx,'event.pds',nu,pow,pow_e, /poisson,rate=[344.632,10000],
time=[0,500]
  10 power spectra selected
```

We can also select only a range of indices:

```
Ghats> ghx,'event.pds',nu,pow,pow_e, /poisson,index=[13,34]
  22 power spectra selected
```

Notice that GHATS commands assume indices start from 1 and not from 0.

Finally, we add yet another selection, only odd indices, making use of IDL/GDL:

```
Ghats> odd_indices = indgen(64)+findgen(32)+1
Ghats> ghx,'event.pds',nu,pow,pow_e, /poisson,rate=[344.632,10000],
time=[0,500], sel=odd_indices
  7 power spectra selected
```

One more thing. The workhorse of GHATS is gh_xte. You can also extract an average PDS which is not Leahy normalized, but rms normalized. For this you need the count rate, which is available in the data file, and the background count rate in the same energy band, which you must estimate by other means. Once you have the background rate, say 13.76 counts/s, you can extract a rms normalized PDS with:

```
Ghats> Ghats> ghx,'event.pds',nu,pow,pow_e, /poisson,rate=[344.632,10000],
time=[0,500], sel=odd_indices, rms=17.36
  7 power spectra selected
```

List of GHATS commands

`gh_xte`	PDS/FFT extraction from RXTE/PCA files.
`gh_swift`	PDS/FFT extraction from Swift/XRT files.
`gh_xmm`	PDS/FFT extraction from XMM (pn & MOS) files.
`gx_nustar`	PDS/FFT extraction from NuSTAR files.
`gx_nicer`	PDS/FFT extraction from NICER files.
`gx_laxpc_misra`	PDS/FFT extraction from AstroSat/laxpc files.
`gx_nustar_czti`	PDS/FFT extraction from AstroSat/CZTI files.
`gx_huiyan_le`	PDS/FFT extraction from Huiyan (HXMT) LE files.
`gx_huiyan_me`	PDS/FFT extraction from Huiyan (HXMT) ME files.
`gx_huiyan_he`	PDS/FFT extraction from Huiyan (HXMT) HE files.
`gahts/gghats`	Startup procedure under IDL/GDL. Not for interactive use
`gh_colors`	Extract X-ray colors from a PDS file
`gh_cross`	Production of time/phase-lag spectrum and lag spectrum from two FFT
`gh_cross_range`	Extract phase/time lag around a feature
`gh_dyn`	Extraction of time-frequency image from PDS
`gh_get_vle`	Extract VLE flag from PCA HK data
`gh_hk_xte`	Extraction of housekeeping information from PDS/FFT for RXTE/PCA
`gh_info`	Output to terminal of basic PDS/FFT info
`gh_licu`	Extraction of light curve from PDS/FFT
`gh_nspectra`	Extraction of number of spectra for PDS/FFT
`gh_oplot_nupower`	Overplot of PDS in νP_ν form
`gh_oplot_power`	Overplot of PDS in Pν form
`gh_pha`	Production of PHA file from PDS
`gh_plot_coh`	Plot coherence
`gh_plot_lag`	Plot phase/time lags
`gh_plot_hk_xte`	Plot of summary of XTE HK information. Does not need previous gh_hk
`gh_plot_licu`	Plot of light curve

List of GHATS commands

`gh_plot_licu_nogaps`	Plot of light curve vs. array index, with gaps shown
`gh_plot_nupower`	Plot of PDS in νP_ν form
`gh_plot_power`	Plot of PDS in $P\nu$ form
`gh_reb`	PDS rebinning
`gh_rmf`	production of (diagonal) response file to associate to PHA file for XSPEC
`gh_version`	Output of GHATS version
`gh_xspec`	Production of PHA/RSP file for XSPEC fitting. Incorporates gh_pha/rmf
`ghats_all`	Graphical overview of a PDS file
`ghats_getheader`	Extract PDS/FFT header. Not for interactive use
`ghats_openfft`	Open a FFT file. Not for interactive use
`ghats_openpds`	Open a PDS file. Not for interactive use
`ghrebin`	Low-level rebin procedure. Not for interactive use
`ghx`	Main PDS extraction procedure.
`poisson_estimate`	Calculate Poissonian contribution to PDS
`read_fft_line`	Read in a FFT from a FFT file
`read_pds_line`	Read in a PDS from a PDS file
`rebin_cross`	Rebin a lag spectrum

GH_XTE

```
gh_xte, [/ghx], [/turbo],[bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/
         bary][,window=window][,wpar=wpar]
gh_xte,infilename,outtype,channels,treb,npds,oufilename, [/ghx], [/turbo],
         [bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/bary]
         [,window=window][,wpar=wpar]
gh_xte,parameterfile
```

Program for the production of PDS/FFT files starting from RXTE/PCA science/event array files. It is the main extraction procedure of the software.

infilename	Name of file in input. It can be an RXTE/PCA data file, a metafile (starting with '@') or a metametafile (starting with '@@'). If the working directory is different from the data directory, full path names must appear in the (meta-)metafiles.
outtype	'POWER' for PDS production, 'FFT' for FFT production
channels	array with start-end values for PHA channels (example: [0,35])
treb	Rebin factor in time (integer)
npds	If int/long: number of light curve points to be FFTed in each interval. If float/double: length of the intervals in seconds. It must be a power of two.
oufilename	Name of file in output. The extension should be 'pds' for PDS and 'fft' for FFT files
parameterfile	Name of parameter file. It should start with '#'. The ASCII file must contain the previous information, one per line
GHX	If set, ghats_all is launched at the end of the program.
TURBO	If set, the program will use C. Markwardt's IDL_EXTRACT routine, running much faster. Not available under GDL.
BANDS	Optional channel bands for the production of three standard2 light curves. It is an array of six elements ([0,10,11,20,21,40]) means three curves in the PCA channel ranges 0-10, 11-20 and 21-40 respectively. They can then be used to produce X-ray hardnesses. Default values are [3,6,7,14,15,20].
GTI	optional user-supplied GTIs, to be intersected with the ones in the data files. If set, it must be a valid filename. Two possible input formats are possible. Either an ASCII file with lines containing start end time of each GTI (in RXTE time units), or a standard GTI FITS file. In case of a FITS file its TIMEZERO keyword (if present) is added to the GTI times, in case of an ASCII file obviously it is not.
SLIDING	optional parameter for sliding the time window. If not set, the time intervals for the FFT (data stretch) do not overlap. If set to N, they are shifted by an Nth of their total duration. As an example, if the data intervals are 16 seconds long, sliding =4 will shift two consecutive intervals by four seconds in start time. If this keyword is set, the averaged PDS obtained with GHX does not obviously make statistical sense.

GH_XTE (cont'd)

BARY	flag for barycentered data. If set, the program will not read the TIME column, but BARYTIME. It is the user's responsibility to make sure that the BARYTIME column exists.
WINDOW	string with the name of the window function to apply to each stretch of data. Possible choices are currently: 'Boxcar' : rectangular window (no windowing: the default) 'Bartlett' 'Hann' 'Welch' 'Cosine' 'Hanning' 'Hamming' 'Triplet' 'Gauss' 'Kaiser' (see appendix A for a description of the functional shape of the windows)
WPAR	additional parameter for windows which require one (see appendix A)

GH_SWIFT

```
gh_swift, [/ghx], [bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/bary]
          [,window=window][,wpar=wpar]
gh_swift,infilename,outtype,channels,treb,npds,oufilename, [/ghx],
          [bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/bary]
          [,window=window][,wpar=wpar]
gh_swift,parameterfile
```

Program for the production of PDS/FFT files starting from Swift/XRT photon files. Both WT and PO files are accepted, although it is not possible to combine them. For Swift, the turbo flag is not available

infilename	Name of file in input. It can be a Swift/XRT photon file, a metafile (starting with '@') or a metametafile (starting with '@@'). If the working directory is different from the data directory, full path names must appear in the (meta-)metafiles.
outtype	'POWER' for PDS production, 'FFT' for FFT production
channels	array with start-end values for PHA channels (example: [0,1023])
treb	Rebin factor in time (integer)
npds	If int/long: number of light curve points to be FFTed in each interval. If float/ double: length of the intervals in seconds. It must be a power of two.
oufilename	Name of file in output. The extension should be 'pds' for PDS and 'fft' for FFT files
parameterfile	Name of parameter file. It should start with '#'. The ASCII file must contain the previous information, one per line
GHX	If set, ghats_all is launched at the end of the program.
BANDS	Optional channel bands for the production of three light curves for colors. It is an array of six elements ([0,100,101,200,201,400]) means three curves in the XRT channel ranges 0-100, 101-200 and 201-400 respectively. They can then be used to produce X-ray hardnesses. Default values are [2,100,101,400,401,1023].
GTI	optional user-supplied GTIs, to be intersected with the ones in the data files. If set, it must be a valid filename. Two possible input formats are possible. Either an ASCII file with lines containing start end time of each GTI (in Swift time units), or a standard GTI FITS file. In case of a FITS file its TIMEZERO keyword (if present) is added to the GTI times, in case of an ASCII file obviously it is not.
SLIDING	optional parameter for sliding the time window. If not set, the time intervals for the FFT (data stretch) do not overlap. If set to N, they are shifted by an Nth of their total duration. As an example, if the data intervals are 16 seconds long, sliding =4 will shift two consecutive intervals by four seconds in start time. If this keyword is set, the averaged PDS obtained with GHX does not obviously make statistical sense.
BARY	flag for barycentered data. If set, the program will not read the TIME column, but BARYTIME. It is the user's responsibility to make sure that the BARYTIME column exists.

GH_SWIFT (cont'd)

WINDOW	string with the name of the window function to apply to each stretch of data. Possible choices are currently: 'Boxcar' : rectangular window (no windowing: the default) 'Bartlett' 'Hann' 'Welch' 'Cosine' 'Hanning' 'Hamming' 'Triplet' 'Gauss' 'Kaiser' (see appendix A for a description of the functional shape of the windows)
WPAR	additional parameter for windows which require one (see appendix A)

GH_XMM

```
gh_xmm, [/ghx], gti=usergtifile, [bands=[s1,e1,s2,e2,s3,e3]], [/sliding], [/bary]
          [,window=window][,wpar=wpar]
gh_xmm,infilename,outtype,channels,treb,npds,oufilename, gti=usergtifile, [/ghx],
          [bands=[s1,e1,s2,e2,s3,e3]], [/sliding], [/bary] [,window=window][,wpar=wpar]
gh_xmm,parameterfile
```

Program for the production of PDS/FFT files starting from XMM photon files. Both pn and MOS files are accepted.

infilename	Name of file in input. It can be an XMM MOS or pn photon file, a metafile (starting with '@') or a metametafile (starting with '@@'). If the working directory is different from the data directory, full path names must appear in the (meta-)metafiles.
outtype	'POWER' for PDS production, 'FFT' for FFT production
channels	array with start-end values for PHA channels (example: [0,1023])
treb	Rebin factor in time (integer)
npds	If int/long: number of light curve points to be FFTed in each interval. If float/ double: length of the intervals in seconds. It must be a power of two.
oufilename	Name of file in output. The extension should be 'pds' for PDS and 'fft' for FFT files
parameterfile	Name of parameter file. It should start with '#'. The ASCII file must contain the previous information, one per line
GHX	If set, ghats_all is launched at the end of the program.
BANDS	Optional channel bands for the production of three light curves for colors. It is an array of six elements ([0,100,101,200,201,400]) means three curves in the channel ranges 0-100, 101-200 and 201-400 respectively. They can then be used to produce X-ray hardnesses. Default values are [70,500,501,4000,4001,10000].
GTI	**MANDATORY** user-supplied GTIs, to be intersected with the ones in the data files. If set, it must be a valid filename. Two possible input formats are possible. Either an ASCII file with lines containing start end time of each GTI (in XMM time units), or a standard GTI FITS file. For XMM the GTI file is not optional, as it is the only way to obtain the GTI information. In case of a FITS file its TIMEZERO keyword (if present) is added to the GTI times, in case of an ASCII file obviously it is not.
SLIDING	optional parameter for sliding the time window. If not set, the time intervals for the FFT (data stretch) do not overlap. If set to N, they are shifted by an Nth of their total duration. As an example, if the data intervals are 16 seconds long, sliding =4 will shift two consecutive intervals by four seconds in start time. If this keyword is set, the averaged PDS obtained with GHX does not obviously make statistical sense.
BARY	flag for barycentered data. If set, the program will not read the TIME column, but BARYTIME. It is the user's responsibility to make sure that the BARYTIME column exists.

GH_XMM (cont'd)

WINDOW	string with the name of the window function to apply to each stretch of data. Possible choices are currently: 'Boxcar' : rectangular window (no windowing: the default) 'Bartlett' 'Hann' 'Welch' 'Cosine' 'Hanning' 'Hamming' 'Triplet' 'Gauss' 'Kaiser' (see appendix A for a description of the functional shape of the windows)
WPAR	additional parameter for windows which require one (see appendix A)

GH_LAXPC_MISRA

```
gh_laxpc_misra, [/ghx], gti=usergtifile, [bands=[s1,e1,s2,e2,s3,e3]], [/sliding], [/
         bary][,window=window][,wpar=wpar]
gh_laxpc_misra,infilename,outtype,channels,treb,npds,oufilename, gti=usergtifile, [/
         ghx],[bands=[s1,e1,s2,e2,s3,e3]], [/sliding], [/bary] [,window=window]
         [,wpar=wpar]
gh_laxpc_misra,parameterfile
```

Program for the production of PDS/FFT files starting from Astrosat/LAXPC photon files. LAXPC units are treated as separate instruments. The string MISRA refers to the fact that the photon files must be one produced with the procedures by Ranjeev Misra.

infilename	Name of file in input. It can be a LAXPC photon file, a metafile (starting with '@') or a metametafile (starting with '@@'). If the working directory is different from the data directory, full path names must appear in the (meta-)metafiles.
outtype	'POWER' for PDS production, 'FFT' for FFT production
channels	array with start-end values for PHA channels (example: [0,1023])
treb	Rebin factor in time (integer)
npds	If int/long: number of light curve points to be FFTed in each interval. If float/ double: length of the intervals in seconds. It must be a power of two.
oufilename	Name of file in output. The extension should be 'pds' for PDS and 'fft' for FFT files
parameterfile	Name of parameter file. It should start with '#'. The ASCII file must contain the previous information, one per line
GHX	If set, ghats_all is launched at the end of the program.
BANDS	Optional channel bands for the production of three light curves for colors. It is an array of six elements ([0,100,101,200,201,400]) means three curves in the channel ranges 0-100, 101-200 and 201-400 respectively. They can then be used to produce X-ray hardnesses. Default values are [16,43,44,113,114,257].
GTI	User-supplied GTIs, to be intersected with the ones in the data files. If set, it must be a valid filename. Two possible input formats are possible. Either an ASCII file with lines containing start end time of each GTI (in Astrosat time units), or a standard GTI FITS file. *** A GTI file must be given to have meaningful results. As default, the GTI file created by the standard software can be given. In case of a FITS file its TIMEZERO keyword (if present) is added to the GTI times, in case of an ASCII file obviously it is not.
SLIDING	optional parameter for sliding the time window. If not set, the time intervals for the FFT (data stretch) do not overlap. If set to N, they are shifted by an Nth of their total duration. As an example, if the data intervals are 16 seconds long, sliding =4 will shift two consecutive intervals by four seconds in start time. If this keyword is set, the averaged PDS obtained with GHX does not obviously make statistical sense.
BARY	flag for barycentered data. If set, the program will not read the TIME column, but BARYTIME. It is the user's responsibility to make sure that the BARYTIME column exists.

GH_LAXPC_MISRA (cont'd)

WINDOW	string with the name of the window function to apply to each stretch of data. Possible choices are currently: 'Boxcar' : rectangular window (no windowing: the default) 'Bartlett' 'Hann' 'Welch' 'Cosine' 'Hanning' 'Hamming' 'Triplet' 'Gauss' 'Kaiser' (see appendix A for a description of the functional shape of the windows)
WPAR	additional parameter for windows which require one (see appendix A)

GH_CZTI

```
gh_czti, [/ghx], gti=usergtifile, [bands=[s1,e1,s2,e2,s3,e3]], [/sliding], [/bary]
         [,window=window][,wpar=wpar][,quadrants=quadrants][,fraction]
gh_czti,infilename,outtype,channels,treb,npds,oufilename, gti=usergtifile, [/ghx],
         [bands=[s1,e1,s2,e2,s3,e3]], [/sliding], [/bary] [,window=window][,wpar=wpar]
gh_czti,parameterfile
```

Program for the production of PDS/FFT files starting from Astrosat/CZTI photon files. CZTI units are treated as separate instruments.

infilename	Name of file in input. It can be a CZTI photon file, a metafile (starting with '@') or a metametafile (starting with '@@'). If the working directory is different from the data directory, full path names must appear in the (meta-)metafiles.
outtype	'POWER' for PDS production, 'FFT' for FFT production
channels	array with start-end values for PHA channels (example: [0,500])
treb	Rebin factor in time (integer)
npds	If int/long: number of light curve points to be FFTed in each interval. If float/ double: length of the intervals in seconds. It must be a power of two.
oufilename	Name of file in output. The extension should be 'pds' for PDS and 'fft' for FFT files
parameterfile	Name of parameter file. It should start with '#'. The ASCII file must contain the previous information, one per line
GHX	If set, ghats_all is launched at the end of the program.
BANDS	Optional channel bands for the production of three light curves for colors. It is an array of six elements ([0,100,101,200,201,400]) means three curves in the channel ranges 0-100, 101-200 and 201-400 respectively. They can then be used to produce X-ray hardnesses. Default values are [20,80,81,256,257,500].
GTI	User-supplied GTIs, to be intersected with the ones in the data files. If set, it must be a valid filename. Two possible input formats are possible. Either an ASCII file with lines containing start end time of each GTI (in Astrosat time units), or a standard GTI FITS file. In case of a FITS file its TIMEZERO keyword (if present) is added to the GTI times, in case of an ASCII file obviously it is not.
SLIDING	optional parameter for sliding the time window. If not set, the time intervals for the FFT (data stretch) do not overlap. If set to N, they are shifted by an Nth of their total duration. As an example, if the data intervals are 16 seconds long, sliding =4 will shift two consecutive intervals by four seconds in start time. If this keyword is set, the averaged PDS obtained with GHX does not obviously make statistical sense.
BARY	flag for barycentered data. If set, the program will not read the TIME column, but BARYTIME. It is the user's responsibility to make sure that the BARYTIME column exists.

GH_CZTI (cont'd)

QUADRANTS	array with the CZTI quadrants to be used for the accumulation. Default is [0,1,2,3]
FRACTION	value for the minimum exposure for CZTI pixels to be accepted. Default is 0.0
WINDOW	string with the name of the window function to apply to each stretch of data. Possible choices are currently: 'Boxcar' : rectangular window (no windowing: the default) 'Bartlett' 'Hann' 'Welch' 'Cosine' 'Hanning' 'Hamming' 'Triplet' 'Gauss' 'Kaiser' (see appendix A for a description of the functional shape of the windows)
WPAR	additional parameter for windows which require one (see appendix A)

GH_NUSTAR

```
gh_nustar, [/ghx], [bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/bary]
          [,window=window][,wpar=wpar]
gh_nustar,infilename,outtype,channels,treb,npds,oufilename, [/ghx],
          [bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/bary]
          [,window=window][,wpar=wpar]
gh_nustar,parameterfile
```

Program for the production of PDS/FFT files starting from NuSTAR photon files. The two telescopes can be combined through the use of metafiles.

infilename	Name of file in input. It can be a NuSTAR photon file, a metafile (starting with '@') or a metametafile (starting with '@@'). If the working directory is different from the data directory, full path names must appear in the (meta-)metafiles.
outtype	'POWER' for PDS production, 'FFT' for FFT production
channels	array with start-end values for PHA channels (example: [0,1023])
treb	Rebin factor in time (integer)
npds	If int/long: number of light curve points to be FFTed in each interval. If float/ double: length of the intervals in seconds. It must be a power of two.
oufilename	Name of file in output. The extension should be 'pds' for PDS and 'fft' for FFT files
parameterfile	Name of parameter file. It should start with '#'. The ASCII file must contain the previous information, one per line
GHX	If set, ghats_all is launched at the end of the program.
BANDS	Optional channel bands for the production of three light curves for colors. It is an array of six elements ([0,100,101,200,201,400]) means three curves in the channel ranges 0-100, 101-200 and 201-400 respectively. They can then be used to produce X-ray hardnesses. Default values are [100,800,801,4095,100,4095].
GTI	optional user-supplied GTIs, to be intersected with the ones in the data files. If set, it must be a valid filename. Two possible input formats are possible. Either an ASCII file with lines containing start end time of each GTI (in NuSTAR time units), or a standard GTI FITS file. In case of a FITS file its TIMEZERO keyword (if present) is added to the GTI times, in case of an ASCII file obviously it is not.
SLIDING	optional parameter for sliding the time window. If not set, the time intervals for the FFT (data stretch) do not overlap. If set to N, they are shifted by an Nth of their total duration. As an example, if the data intervals are 16 seconds long, sliding =4 will shift two consecutive intervals by four seconds in start time. If this keyword is set, the averaged PDS obtained with GHX does not obviously make statistical sense.
BARY	flag for barycentered data. If set, the program will not read the TIME column, but BARYTIME. It is the user's responsibility to make sure that the BARYTIME column exists.

GH_NUSTAR (cont'd)

WINDOW	string with the name of the window function to apply to each stretch of data. Possible choices are currently: 'Boxcar' : rectangular window (no windowing: the default) 'Bartlett' 'Hann' 'Welch' 'Cosine' 'Hanning' 'Hamming' 'Triplet' 'Gauss' 'Kaiser' (see appendix A for a description of the functional shape of the windows)
WPAR	additional parameter for windows which require one (see appendix A)

GH_NICER

```
gh_nicer, [/ghx], [bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/bary]
          [,window=window][,wpar=wpar]
gh_ nicer,infilename,outtype,channels,treb,npds,oufilename, [/ghx],
          [bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/bary]
          [,window=window][,wpar=wpar]
gh_nicer,parameterfile
```

Program for the production of PDS/FFT files starting from NICER photon files.

infilename	Name of file in input. It can be a NICER photon file, a metafile (starting with '@') or a metametafile (starting with '@@'). If the working directory is different from the data directory, full path names must appear in the (meta-)metafiles.
outtype	'POWER' for PDS production, 'FFT' for FFT production
channels	array with start-end values for PHA channels (example: [0,1023])
treb	Rebin factor in time (integer)
npds	If int/long: number of light curve points to be FFTed in each interval. If float/ double: length of the intervals in seconds. It must be a power of two.
oufilename	Name of file in output. The extension should be 'pds' for PDS and 'fft' for FFT files
parameterfile	Name of parameter file. It should start with '#'. The ASCII file must contain the previous information, one per line
GHX	If set, ghats_all is launched at the end of the program.
BANDS	Optional channel bands for the production of three light curves for colors. It is an array of six elements ([0,100,101,200,201,400]) means three curves in the channel ranges 0-100, 101-200 and 201-400 respectively. They can then be used to produce X-ray hardnesses. Default values are [100,800,801,4095,100,4095].
GTI	optional user-supplied GTIs, to be intersected with the ones in the data files. If set, it must be a valid filename. Two possible input formats are possible. Either an ASCII file with lines containing start end time of each GTI (in NuSTAR time units), or a standard GTI FITS file. In case of a FITS file its TIMEZERO keyword (if present) is added to the GTI times, in case of an ASCII file obviously it is not.
SLIDING	optional parameter for sliding the time window. If not set, the time intervals for the FFT (data stretch) do not overlap. If set to N, they are shifted by an Nth of their total duration. As an example, if the data intervals are 16 seconds long, sliding =4 will shift two consecutive intervals by four seconds in start time. If this keyword is set, the averaged PDS obtained with GHX does not obviously make statistical sense.
BARY	flag for barycentered data. If set, the program will not read the TIME column, but BARYTIME. It is the user's responsibility to make sure that the BARYTIME column exists.

GH_NICER (cont'd)

WINDOW	string with the name of the window function to apply to each stretch of data. Possible choices are currently: 'Boxcar' : rectangular window (no windowing: the default) 'Bartlett' 'Hann' 'Welch' 'Cosine' 'Hanning' 'Hamming' 'Triplet' 'Gauss' 'Kaiser' (see appendix A for a description of the functional shape of the windows)
WPAR	additional parameter for windows which require one (see appendix A)

GH_HUIYAN_LE

```
gh_huiyan_le, [/ghx], [bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/
          bary][,window=window][,wpar=wpar]
gh_huiyan_le,infilename,outtype,channels,treb,npds,oufilename, [/ghx],
          [bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/bary]
          [,window=window][,wpar=wpar]
gh_huiyan_le,parameterfile
```

Program for the production of PDS/FFT files starting from Huiyan/LE photon files.

infilename	Name of file in input. It can be a Huiyan/LE photon file, a metafile (starting with '@') or a metametafile (starting with '@@'). If the working directory is different from the data directory, full path names must appear in the (meta-)metafiles.
outtype	'POWER' for PDS production, 'FFT' for FFT production
channels	array with start-end values for PHA channels (example: [0,1023])
treb	Rebin factor in time (integer)
npds	If int/long: number of light curve points to be FFTed in each interval. If float/ double: length of the intervals in seconds. It must be a power of two.
oufilename	Name of file in output. The extension should be 'pds' for PDS and 'fft' for FFT files
parameterfile	Name of parameter file. It should start with '#'. The ASCII file must contain the previous information, one per line
GHX	If set, ghats_all is launched at the end of the program.
BANDS	Optional channel bands for the production of three light curves for colors. It is an array of six elements ([0,100,101,200,201,400]) means three curves in the channel ranges 0-100, 101-200 and 201-400 respectively. They can then be used to produce X-ray hardnesses. Default values are [100,800,801,4095,100,4095].
GTI	optional user-supplied GTIs, to be intersected with the ones in the data files. If set, it must be a valid filename. Two possible input formats are possible. Either an ASCII file with lines containing start end time of each GTI (in NuSTAR time units), or a standard GTI FITS file. In case of a FITS file its TIMEZERO keyword (if present) is added to the GTI times, in case of an ASCII file obviously it is not.
SLIDING	optional parameter for sliding the time window. If not set, the time intervals for the FFT (data stretch) do not overlap. If set to N, they are shifted by an Nth of their total duration. As an example, if the data intervals are 16 seconds long, sliding =4 will shift two consecutive intervals by four seconds in start time. If this keyword is set, the averaged PDS obtained with GHX does not obviously make statistical sense.
BARY	flag for barycentered data. If set, the program will not read the TIME column, but BARYTIME. It is the user's responsibility to make sure that the BARYTIME column exists.

GH_HUIYAN_LE (cont'd)

WINDOW	string with the name of the window function to apply to each stretch of data. Possible choices are currently: 'Boxcar' : rectangular window (no windowing: the default) 'Bartlett' 'Hann' 'Welch' 'Cosine' 'Hanning' 'Hamming' 'Triplet' 'Gauss' 'Kaiser' (see appendix A for a description of the functional shape of the windows)
WPAR	additional parameter for windows which require one (see appendix A)

GH_HUIYAN_ME

```
gh_huiyan_me, [/ghx], [bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/
          bary][,window=window][,wpar=wpar]
gh_huiyan_me,infilename,outtype,channels,treb,npds,oufilename, [/ghx],
          [bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/bary]
          [,window=window][,wpar=wpar]
gh_huiyan_me,parameterfile
```

Program for the production of PDS/FFT files starting from Huiyan/ME photon files.

infilename	Name of file in input. It can be a Huiyan/ME photon file, a metafile (starting with '@') or a metametafile (starting with '@@'). If the working directory is different from the data directory, full path names must appear in the (meta-)metafiles.
outtype	'POWER' for PDS production, 'FFT' for FFT production
channels	array with start-end values for PHA channels (example: [0,1023])
treb	Rebin factor in time (integer)
npds	If int/long: number of light curve points to be FFTed in each interval. If float/ double: length of the intervals in seconds. It must be a power of two.
oufilename	Name of file in output. The extension should be 'pds' for PDS and 'fft' for FFT files
parameterfile	Name of parameter file. It should start with '#'. The ASCII file must contain the previous information, one per line
GHX	If set, ghats_all is launched at the end of the program.
BANDS	Optional channel bands for the production of three light curves for colors. It is an array of six elements ([0,100,101,200,201,400]) means three curves in the channel ranges 0-100, 101-200 and 201-400 respectively. They can then be used to produce X-ray hardnesses. Default values are [100,800,801,4095,100,4095].
GTI	optional user-supplied GTIs, to be intersected with the ones in the data files. If set, it must be a valid filename. Two possible input formats are possible. Either an ASCII file with lines containing start end time of each GTI (in NuSTAR time units), or a standard GTI FITS file. In case of a FITS file its TIMEZERO keyword (if present) is added to the GTI times, in case of an ASCII file obviously it is not.
SLIDING	optional parameter for sliding the time window. If not set, the time intervals for the FFT (data stretch) do not overlap. If set to N, they are shifted by an Nth of their total duration. As an example, if the data intervals are 16 seconds long, sliding =4 will shift two consecutive intervals by four seconds in start time. If this keyword is set, the averaged PDS obtained with GHX does not obviously make statistical sense.
BARY	flag for barycentered data. If set, the program will not read the TIME column, but BARYTIME. It is the user's responsibility to make sure that the BARYTIME column exists.

GH_HUIYAN_ME (cont'd)

WINDOW	string with the name of the window function to apply to each stretch of data. Possible choices are currently: 'Boxcar' : rectangular window (no windowing: the default) 'Bartlett' 'Hann' 'Welch' 'Cosine' 'Hanning' 'Hamming' 'Triplet' 'Gauss' 'Kaiser' (see appendix A for a description of the functional shape of the windows)
WPAR	additional parameter for windows which require one (see appendix A)

GH_HUIYAN_HE

```
gh_huiyan_he, [/ghx], [bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/
          bary][,window=window][,wpar=wpar]
gh_huiyan_he,infilename,outtype,channels,treb,npds,oufilename, [/ghx],
          [bands=[s1,e1,s2,e2,s3,e3]],[gti=usergtifile], [/sliding], [/bary]
          [,window=window][,wpar=wpar]
gh_huiyan_he,parameterfile
```

Program for the production of PDS/FFT files starting from Huiyan/HE photon files.

infilename	Name of file in input. It can be a Huiyan/HE photon file, a metafile (starting with '@') or a metametafile (starting with '@@'). If the working directory is different from the data directory, full path names must appear in the (meta-)metafiles.
outtype	'POWER' for PDS production, 'FFT' for FFT production
channels	array with start-end values for PHA channels (example: [0,1023])
treb	Rebin factor in time (integer)
npds	If int/long: number of light curve points to be FFTed in each interval. If float/double: length of the intervals in seconds. It must be a power of two.
oufilename	Name of file in output. The extension should be 'pds' for PDS and 'fft' for FFT files
parameterfile	Name of parameter file. It should start with '#'. The ASCII file must contain the previous information, one per line
GHX	If set, ghats_all is launched at the end of the program.
BANDS	Optional channel bands for the production of three light curves for colors. It is an array of six elements ([0,100,101,200,201,400]) means three curves in the channel ranges 0-100, 101-200 and 201-400 respectively. They can then be used to produce X-ray hardnesses. Default values are [100,800,801,4095,100,4095].
GTI	optional user-supplied GTIs, to be intersected with the ones in the data files. If set, it must be a valid filename. Two possible input formats are possible. Either an ASCII file with lines containing start end time of each GTI (in NuSTAR time units), or a standard GTI FITS file. In case of a FITS file its TIMEZERO keyword (if present) is added to the GTI times, in case of an ASCII file obviously it is not.
SLIDING	optional parameter for sliding the time window. If not set, the time intervals for the FFT (data stretch) do not overlap. If set to N, they are shifted by an Nth of their total duration. As an example, if the data intervals are 16 seconds long, sliding =4 will shift two consecutive intervals by four seconds in start time. If this keyword is set, the averaged PDS obtained with GHX does not obviously make statistical sense.
BARY	flag for barycentered data. If set, the program will not read the TIME column, but BARYTIME. It is the user's responsibility to make sure that the BARYTIME column exists.

GH_HUIYAN_HE (cont'd)

WINDOW	string with the name of the window function to apply to each stretch of data. Possible choices are currently: 'Boxcar' : rectangular window (no windowing: the default) 'Bartlett' 'Hann' 'Welch' 'Cosine' 'Hanning' 'Hamming' 'Triplet' 'Gauss' 'Kaiser' (see appendix A for a description of the functional shape of the windows)
WPAR	additional parameter for windows which require one (see appendix A)

GHATS/GGHATS

```
ghats
gghats
```

Program to start the package. It is usually not run interactively, but executed at startup of IDL/GDL

GH_COLORS

```
gh_colors,filename,time,hr1,hr2[,rate1,rate2,rate3]
```

Program to extract X-ray color information from a PDS file.

filename	Input file name of PDS file
time	Output array with time values
hr1	Output array with HR1 values (rate2/rate1)
hr1	Output array with HR2 values (rate3/rate1)
rate1	Output light curve in band 1 (optional)
rate2	Output light curve in band 2 (optional)
rate3	Output light curve in band 3 (optional)

GH_CROSS

```
gh_cross,filename1,filename2,frequency,lag,lag_err,coh,coh_err,irf, [/tlag],
         [index=iarray], [time=tarray], [sel=selarray], [/poivalue], [cross=cross]
```

Program to read in two FFT from compatible FFT files, selecting on a range of FFTs in time, range, or selection of indices. It produces phase lags (in radians) or time lags, and coherence.
WARNING! In coherence computation, for RXTE/PCA data the Poissonian levels are estimated using precise expressions, while for other missions the Poissonian level is assumed to be 2.0.
Positive lags indicate that the second time series lags the first.

filename1	Name of file for FFT1.
filename2	Name of file for FFT2. It must be compatible with FFT1 (same observatory, instrument, target, time resolution, frequency resolution, start time and number of FFT)
frequency	output array containing the frequencies of the lag/coherence spectrum
lag	output array containing the phase/time lag spectrum
lag_e	output array containing the errors for the phase/time lag spectrum
coh	output array containing the coherence spectrum
coh_e	output array containing the errors for the coherence spectrum
irf	Rebinning factor (integer). If positive, it will be used as a linear rebinning factor. If negative, the output will be rebinned logarithmically, i.e. each frequency bin will be larger than the previous one by a factor of exp(-1/irf). Typical negative value is -100.
TLAG	If set, the program will output time lags. If not, phase lags will be used.
INDEX	Optional index range for accumulated spectra. Example: index=[12,34] will average only spectra from #12 to #34 in the FFT file
TIME	Optional time range for accumulated spectra. Example: time=[0.0,500.0] will average only spectra corresponding to that time interval in the FFT file
SEL	Optional list of indices corresponding to spectra to accumulate. Example: sel=[1,3,5,7,9] will average only spectra corresponding to those indices in the FFT file. Remember that indices start from 1 in PDS/FFT files.
POIVALUE	Average value of the real part of the cross spectrum to be subtracted from the output value before computing the lag. It can be obtained with GH_CROSS_RANGE giving in input a broad frequency range dominated by Poissonian noise.
CROSS	2-olumn COMPLEX array. Column 1: cross spectrum; column 2: errors on the real and imaginary parts of the cross spectrum.

GH_CROSS_RANGE

```
gh_cross_range,filename1,filename2,centroid,width,frequency,lag,lag_err,irf,
               [/tlag], [index=iarray], [time=tarray], [sel=selarray], [/plot],
               [/poivalue]
```

Program to read in two FFT from compatible FFT files, selecting on a range of FFTs in time, range, or selection of indices. It produces phase lags (in radians) or time lags averaged over a feature. For the feature, the user inputs centroid and FWHM and the average will be over the frequency range [centroid-FWHM/2, centroid+FWHM/2]. It optionally plots the full lag spectrum with the relevant interval marked.
WARNING! In coherence computation, for RXTE/PCA data the Poissonian levels are estimated using precise expressions, while for other missions the Poissonian level is assumed to be 2.0.
Positive lags indicate that the second time series lags the first.

filename1	Name of file for FFT1.
filename2	Name of file for FFT2. It must be compatible with FFT1 (same observatory, instrument, target, time resolution, frequency resolution, start time and number of FFT)
centroid	Centroid frequency for the averaging
width	FWHM for the averaging
frequency	output array containing the frequencies of the lag/coherence spectrum
lag	output array containing the phase/time lag spectrum
lag_e	output array containing the errors for the phase/time lag spectrum
irf	Rebinning factor (integer) for the lag plot. If positive, it will be used as a linear rebinning factor. If negative, the output will be rebinned logarithmically, i.e. each frequency bin will be larger than the previous one by a factor of exp(-1/irf). Typical negative value is -100.
TLAG	If set, the program will output time lags. If not, phase lags will be used.
INDEX	Optional index range for accumulated spectra. Example: index=[12,34] will average only spectra from #12 to #34 in the FFT file
TIME	Optional time range for accumulated spectra. Example: time=[0.0,500.0] will average only spectra corresponding to that time interval in the FFT file
SEL	Optional list of indices corresponding to spectra to accumulate. Example: sel=[1,3,5,7,9] will average only spectra corresponding to those indices in the FFT file. Remember that indices start from 1 in PDS/FFT files.
PLOT	Optional keyword. If set, the full phase/time lag spectrum is plotted, rebinned according to irf, with vertical lines marking the averaging interval.
POIVALUE	Average value of the real part of the cross spectrum to be subtracted from the output value before computing the lag. It can be obtained with the same program giving in input a broad frequency range dominated by Poissonian noise.

GH_DYN

```
gh_dyn,filename,time,rate,frequency,dynima, [index=iarray], [frebin=fr], [trebin=tr]
```

Program to read in a PDS file and extract a light curve and a time-frequency image

filename	Name of input PDS file.
time	Output array for time values
rate	Output array for count rate values
frequency	Output array containing the frequencies
dynima	Output array containing the time-frequency image
INDEX	Optional index range for accumulated spectra. Example: index=[12,34] will average only spectra from #12 to #34 in the FFT file
FREBIN	Frequency rebinning factor (integer). If positive, it will be used as a linear rebinning factor. If negative, the output will be rebinned logarithmically, i.e. each frequency bin will be larger than the previous one by a factor of exp(-1/irf). Typical negative value is -100. Default is 1.
TREBIN	Time rebinning factor (integer). Only linear rebinning is allowed.

GH_GETVLE

```
gh_getvle,filename,i_vle,[/nodialog]
```

Program to read in the RXTE/PCA VLE flag from a FH5a* file. Not intended for interactive use.

filename	Name of input PDS file.
i_vle	Output VLE flag value
NODIALOG	If this keyword is not set, no dialog window will be opened. Kind of reverse logic here.

GH_HK_XTE

```
gh_hk_xte,filename,time,vle,ndet,poisson
```

Program to read in a XTE/PCA PDS file and extract a housekeeping information for each spectrum, such as VLE rate, number of active detectors and estimated Poissonian level

filename	Name of input PDS file.
time	Output array for time values, one per PDS in the input file
vle	Output array with VLE (Very Large Events) rate values
ndet	Output array with number of active detectors
poisson	Output array with estimated Poisson value (using the expression from Zhang et al., 1995, ApJ, 449, 930.

GH_INFO

```
gh_info,filename
```

Program to display basic information for a PDS/FFT file

filename	Name of input PDS or FFT file.

GH_LICU

```
gh_licu,filename,time,rate, [delta=t], [mjd=rmjd0]
```

Program to read in a PDS file and extract a light curve

filename	Name of input PDS file.
time	Output array for time values
rate	Output array for count rate values
DELTA	Optional output keyword containing the time bin size for the light curve.
RMJD	Optional output keyword containing the MJD corresponding to time 0.

GH_NSPECTRA

```
gh_nspectra,filename,n
```

Program to read in the number of PDS in a PDS file

filename	Name of input PDS file.
n	Output number of PDS

GH_OPLOT_NUPOWER

```
gh_oplot_nupower,frequency,power,power_err
```

Program to overplot a PDS in νP_ν form

frequency	Array with input frequencies
power	Array with input powers
power_err	Array with input errors on powers

GH_OPLOT_POWER

```
gh_oplot_power,frequency,power,power_err
```

Program to overplot a PDS in P_v form

frequency	Array with input frequencies
power	Array with input powers
power_err	Array with input errors on powers

GH_PHA

```
gh_pha,power,power_err,phaname,rmfname,telescope,instrument
```

Program to write a PDS to a PHA file for XSPEC fitting. Not intended for interactive use (use GH_XSPEC)

power	Array with input powers
power_err	Array with input errors on powers
phaname	Filename for output PHA file
rmfname	Filename for associated RSP file, to be written into a keyword for automatic load together with the PHA
telescope	Input string with telescope name, for header (8 characters)
instrument	Input string with instrument name, for header (8 characters)

GH_PLOT_COH

```
gh_plot_coh,frequency,coh,coh_err,[xlow,xhigh],[ylow,yhigh], [\ps]
```

Program to plot a light curve

frequency	Array with input frequencies
coh	Array with input coherence
coh_err	Array with input coherence errors
xlow	Optional low X value for plot (if specified, also xhigh must be given)
xhigh	Optional high X value for plot
ylow	Optional low Y value for plot (if specified, also yhigh must be given)
yhigh	Optional high Y value for plot
PS	Keyword for PostScript output. If set, the plot will go to a PS file called ‘gh_coh.ps’

GH_PLOT_LAG

```
gh_plot_lag,frequency,lag,lag_err,[xlow,xhigh],[ylow,yhigh], [/ps], [/time], [/lin], [/
          hyp]
```

Program to plot a light curve

frequency	Array with input frequencies
lag	Array with input phase lags
lag_err	Array with input phase-lag errors
xlow	Optional low X value for plot (if specified, also xhigh must be given)
xhigh	Optional high X value for plot
ylow	Optional low Y value for plot (if specified, also yhigh must be given)
yhigh	Optional high Y value for plot
PS	Keyword for PostScript output. If set, the plot will go to a PS file called 'gh_lag.ps'
TIME	Keyword: if set, input is time lags. The program changes only the Y label
LIN	Keyword: if set, plot lags on linear scale
HYP	Keyword: if set, the hyperbolic sine of the lags is plotted. This allows a log plot also for negative lags

GH_PLOT_HK_XTE

```
gh_plot_hk_xte,filename, [\ps]
```

Program to plot XTE housekeeping information. It does not need gh_hk as it reads HK info directly from a PDS file

filename	Input filename for PDS file
PS	Keyword for PostScript output. If set, the plot will go to a PS file called ‘gh_hk.ps’

GH_PLOT_LICU

```
gh_plot_licu,time,rate,[xlow,xhigh],[ylow,yhigh], [\ps]
```

Program to plot a light curve

time	Array with input times
rate	Array with input count rates
xlow	Optional low X value for plot (if specified, also xhigh must be given)
xhigh	Optional high X value for plot
ylow	Optional low Y value for plot (if specified, also yhigh must be given)
yhigh	Optional high Y value for plot
PS	Keyword for PostScript output. If set, the plot will go to a PS file called ‘gh_licu.ps’

GH_PLOT_LICU_NOGAPS

```
gh_plot_licu_nogaps,time,rate,[xlow,xhigh],[ylow,yhigh], [\ps]
```

Program to plot a light curve where the time axis represents the index of the point in the array and the gaps are shown by vertical dotted lines

time	Array with input times
rate	Array with input count rates
xlow	Optional low X value for plot (if specified, also xhigh must be given)
xhigh	Optional high X value for plot
ylow	Optional low Y value for plot (if specified, also yhigh must be given)
yhigh	Optional high Y value for plot
PS	Keyword for PostScript output. If set, the plot will go to a PS file called 'gh_licu.ps'

GH_PLOT_NUPOWER

```
gh_plot_nupower,frequency,power,power_err,[xlow,xhigh],[ylow,yhigh], [\ps]
```

Program to plot a PDS in νP_ν form

frequency	Array with input frequencies
power	Array with input powers
power_err	Array with input errors on powers
xlow	Optional low X value for plot (if specified, also xhigh must be given)
xhigh	Optional high X value for plot
ylow	Optional low Y value for plot (if specified, also yhigh must be given)
yhigh	Optional high Y value for plot
PS	Keyword for PostScript output. If set, the plot will go to a PS file called 'gh_nupower.ps'

GH_PLOT_POWER

```
gh_plot_power,frequency,power,power_err,[xlow,xhigh],[ylow,yhigh], [\ps]
```

Program to plot a PDS in P_v form

frequency	Array with input frequencies
power	Array with input powers
power_err	Array with input errors on powers
xlow	Optional low X value for plot (if specified, also xhigh must be given)
xhigh	Optional high X value for plot
ylow	Optional low Y value for plot (if specified, also yhigh must be given)
yhigh	Optional high Y value for plot
PS	Keyword for PostScript output. If set, the plot will go to a PS file called ‘gh_power.ps’

GH_REB

```
gh_reb,frequency,power,power_err,irf,nu,pow,pow_e
```

Program to rebin PDS linearly or logairthmically

frequency	Array with input frequencies
power	Array with input powers
power_err	Array with input errors on powers
irf	Frequency rebinning factor (integer). If positive, it will be used as a linear rebinning factor. If negative, the output will be rebinned logarithmically, i.e. each frequency bin will be larger than the previous one by a factor of exp(-1/irf). Typical negative value is -100.
nu	Array with output rebinned frequencies
pow	Array with output rebinned powers
pow_e	Array with output errors on rebinned powers

GH_RMF

```
gh_rmf,nulow,nuhigh,rmfname,telescope,instrument
```

Program to write a (diagonal) RSP file associated to a PHS file for XSPEC fitting. Not intended for interactive use (use GH_XSPEC)

nulow	Array with low boundaries for frequency bins
nuhigh	Array with high boundaries for frequency bins
rmfname	Filename for output RSP file.
telescope	Input string with telescope name, for header (8 characters)
instrument	Input string with instrument name, for header (8 characters)

GH_VERSION

```
gh_version
```

Program to display in an external window the version number for GHATS.

GH_XSPEC

```
gh_xspec,frequency,power,power_err,outputname, [\xspec], [telescope=tel],
         [instrument=inst]
```

Program to output a PDS in PHA form (with associated diagonal response in an RSP file) for XSPEC (or ISIS) fitting

frequency	Array with input frequencies
power	Array with input powers
power_err	Array with input errors on powers
outputname	File name for the output PHA and RSP files. No extension is required.
XSPEC	If set, this keyword will have XSPEC started automatically (if in the users's path) with the outputname file in input
TELESCOPE	Optional input telescope name, for header (8 characters)
INSTRUMENT	Optional input instrument name, for header (8 characters)

GHATS_ALL

```
ghats_all,filename, [/ps], [/poisson], [rebin=reb]
```

Program to produce a summary plot from a PDS file. The plot includes text information, light curve and average power spectrum

filename	Input filename for PDS file
PS	Keyword for PostScript output. If set, the plot will go to a PS file called 'gh_power.ps'
POISSON	If set, the Poisson values computed with the formula of Zhang et al. (1995, ApJ, 449, 930) are subtracted
REBIN	Optional frequency rebinning factor (integer). If positive, it will be used as a linear rebinning factor. If negative, the output will be rebinned logarithmically, i.e. each frequency bin will be larger than the previous one by a factor of exp(-1/irf). Default value if keyword is not set is -100.

GHATS_GETHEADER

```
ghats_getheader,unit,gh_version_string,observatory,instrument,source,rmjd,nft,T,
        ntotal_ffts,e,sliding,baryflag,n_spectral_bins,background_flag,dummy
```

Program to extract header information from a PDS file. The file must have been already opened with ghats_openpds. Once opened, the file must be closed manually. Not intended for interactive use.

unit	Input file unit, received from ghats_openpds
gh_version_string	Output string with GHATS version information
observatory	Output string with observatory name
instrument	Output string with instrument name
source	Output string with source name
rmjd	Output MJD value for start time
nft	Output number of time bins per interval (twice the number of frequencies)
ntotal_ffts	Output number of PDS intervals contained in the file
e	Output 2*I array with start channel and end channel
sliding	Output value for the proliferation factor. In case sliding windows were chosen in gh_xte, this is the sliding factor used (see page for gh_xte).
baryflag	Output flag: 1=input times were barycentered
n_spectral_bins	Output number of spectral bins associated to each PDS. For RXTE this is fixed to three
background_flag	Output flag to inform whether additional background information is available in the file. For RXTE, this is always 0
dummy	Output array containing additional information.

GHATS_OPENFFT

```
ghats_openfft,filename,unit, [/dialog]
```

Program to open an FFT file. It gives in output the unit associated to the file. The file must be close by the user. Not intended for interactive use.

filename	Input file name for FFT file
unit	Output file unit
DIALOG	If set, a dialog window for the file selection will be opened.

GHATS_OPENPDS

```
ghats_openpds,filename,unit, [/dialog]
```

Program to open a PDS file. It gives in output the unit associated to the file. The file must be close by the user. Not intended for interactive use.

filename	Input file name for PDS file
unit	Output file unit
DIALOG	If set, a dialog window for the file selection will be opened.

GHREBIN

```
ghrebin,x,sx,y,sy,irf,xr,sxr,yr,syr,nrd
```

Program to rebin a PDS. Low level routine, not intended for interactive use. Use gh_reb to rebin.

x	Input array of frequency values
sx	Input array on frequency errors. If no errors are available, this variable is dummy (use x)
y	Input array of power values
sy	Input array on power errors. If no errors are available, this variable is dummy (use y)
irf	Input frequency rebinning factor (integer). If positive, it will be used as a linear rebinning factor. If negative, the output will be rebinned logarithmically, i.e. each frequency bin will be larger than the previous one by a factor of exp(-1/irf). Typical negative value is -100.
xr	Output array of rebinned frequency values
sxr	Output array of errors for xr
yr	Output array of rebinned power values
syr	Output array of errors for yr
nrd	Output number of elements of xr and yr

GHX

```
ghx,filename,frequency,power,power_err, [index=in], [time=ti], [rate=ra], [sel=se],
        [/poisson], [rms=r]
```

Program to extract a PDS from a PDS file. The backbone of the analysis part of GHATS.

filename	Input file name of PDS file
frequency	Output frequency array
power	Output power array
power_err	Output array with errors on power
INDEX	Optional index range for accumulated spectra. Example: index=[12,34] will average only spectra from #12 to #34 in the FFT file
TIME	Optional time range for accumulated spectra. Example: time=[0.0,500.0] will average only spectra corresponding to that time interval in the FFT file
RATE	Optional rate range for accumulated spectra. Example: rate=[125.0,347.0] will average only spectra where the total count rate is between 125 and 347 cts/s.
SEL	Optional list of indices corresponding to spectra to accumulate. Example: sel=[1,3,5,7,9] will average only spectra corresponding to those indices in the FFT file. Remember that indices start from 0 in PDS/FFT files.
POISSON	If set, the Poisson values computed with the formula of Zhang et al. (1995, ApJ, 449, 930) are subtracted
RMS	Optional value for the total average background rate in the considered channel band. When it is specified, the program will convert the powers from Leahy normalization to squared rms normalization.

POISSON_ESTIMATE

```
poisson_estimate,poisson_level,cnts,time_per_spectrum,current_vle_rate,nfreq,ndet,
          i_vle,tdead, [/differential]
```

Program to obtain the frequency-dependent RXTE/PCA Poissonian level after Zhang et al. (1995, ApJ, 449, 930). Not intended for interactive use.

poisson_level	Output array with frequency-dependent Poissonian level
cnts	Input count rate
time_per_spectrum	Input time length for the data interval
current_vle_rate	Input RXTE/PCA VLE rate
nfreq	Input number of frequencies in the PDS
ndet	Input number of active PCA detectors
i_vle	Input VLE flag
tdead	Input dead time value

DIFFERENTIAL If this keyword is set, the program returns the Poisson array for the current data stretch. If it is not set, the program adds the Poissonian array to the sum of the previous ones. The final division by the number of PDS in that case is left to the user.

READ_FFT_LINE

```
read_fft_line,unit,muflag,rmjd,cnts,poisson,current_vle_rate,ndetectors,values
```

Program to read in the next FFT from an already opened FFT file. Not intended for interactive use.

unit	Input unit number for FFT file
muflag	not used
rmjd	Output MJD value corresponding to the start time of the current interval
cnts	Output count rate
poisson	Output Poissonian level (average value)
current_vle_rate	Output VLE rate
ndetectors	Output number of active detectors
values	Output array of FFT values (complex numbers)

READ_PDS_LINE

```
read_pds_line,unit,muflag,rmjd,cnts,poisson,current_vle_rate,ndetectors,variance,values
```

Program to read in the next PDS from an already opened PDS file. Not intended for interactive use.

unit	Input unit number for PDS file
muflag	not used
rmjd	Output MJD value corresponding to the start time of the current interval
cnts	Output count rate
poisson	Output Poissonian level (average value)
current_vle_rate	Output VLE rate
ndetectors	Output number of active detectors
variance	Output zero-frequency power
values	Output array of PDS values

REBINCROSS

```
rebincross,frequency,cross,irf,nu,rcross,nout
```

Program to rebin a cross spectrum linearly or logairthmically. Not intended for interactive use.

frequency	Array with input frequencies
cross	Array with cross powers
irf	Frequency rebinning factor (integer). If positive, it will be used as a linear rebinning factor. If negative, the output will be rebinned logarithmically, i.e. each frequency bin will be larger than the previous one by a factor of exp(-1/irf). Typical negative value is -100.
nu	Output array with output rebinned frequencies
rcross	Output array with output rebinned cross powers
nout	Output number of rebinned frequencies

APPENDIX A

Window functions

Here are listed the functional forms of the available window functions. The functions here refer to the the time interval [-T/2 , T/2] and are zero outside it. α, β, σ and γ are parameters which need to be supplied by the user.

Boxcar: $$f(t) = 1$$

Bartlett: $$f(t) = 1 - \left| \frac{t - T/2}{T/2} \right|$$

Hann: $$f(t) = \frac{1}{2}\left[1 - \cos\left(\frac{2\pi t}{T/2}\right)\right]$$

Welch: $$f(t) = 1 - \left(\frac{t - T/2}{T/2}\right)^2$$

Cosine: $$f(t) = \cos(\pi t)$$

Hanning: $$f(t) = \cos^2(\pi t)$$

Hamming: $$f(t) = \alpha + (1 - \alpha)\cos^2(\pi t)$$

Triplet: $$f(t) = e^{\frac{-\beta}{|t|}} \cos^2(\pi t)$$

Gauss: $$f(t) = \exp^{\frac{-1}{2}\frac{t^2}{\sigma^2}}$$

Kaiser: $$f(t) = \gamma\sqrt{1 - (2t)^2}$$