

spi_obs_fit

User Manual

Version 3.1.0
19 June 2006

Jürgen Knödseder
Centre d'Etude Spatiale des Rayonnements
knodseder@cesr.fr
<http://www.cesr.fr/~jurgen/index.html>

Note to the user

This software has been written to analyse data of the SPI telescope onboard INTEGRAL. Particular care has been taken in making the software user friendly and well documented. If you appreciated this effort, and if this software and User Manual were useful for your scientific work, the author would appreciate a corresponding acknowledgment in your published work.

Contents

1	Introduction	1
2	Getting started	1
3	Parameter file	3
4	Working examples	14
4.1	Fitting the Crab flux	14
4.2	Spectral analysis of the 511 keV bulge	14
5	Interface definition	15
6	Algorithms	15
7	Error codes	15

1 Introduction

The executable `spi_obs_fit` is part of the SPI Scientific Analysis Tools software component (SAT). It allows for fitting of sky models on top of a model of the instrumental background to the data.

A sky model may be any representation of the gamma-ray intensity distribution, such as an intensity map (e.g. a tracer map of diffuse emission) or a set of point or point-like sources. `spi_obs_fit` accepts intensity distributions and point sources as input parameter (point sources may be specified in celestial or galactic coordinates). `spi_obs_fit` allows the simultaneous fitting of up to 20 sky models.

The sky models are convolved with the instrumental response function to predict the number of expected events in each data-space cell (the SPI data-space is spanned by the pointing number, the pseudo-detector identifier, and the energy). `spi_obs_fit` then determines scaling factors for all sky models, which may then be used to determine gamma-ray fluxes. `spi_obs_fit` also determines the significance of the respective models.

In principle, the architecture of `spi_obs_fit` allows fitting of individual model scaling parameters for each data-space bin and model component. In practice, data-space bins are combined in so-called *parameter groups*, i.e. a specific model parameter affects in general a group of data-space bins. Parameter groups can be defined for all of the three SPI data-space dimensions. In this way it is possible to fit

- time variations
- detector-to-detector variations
- energy variations (i.e. spectra)

for both the source and the background models. The fit of all parameters is performed in one shot to allow to trace correctly the parameter correlations. This leads to large memory requirements if more than a few 1000 parameters are fit simultaneously.

`spi_obs_fit` is written in the ANSI C++ language and has been developed under ISDC support platform 6.3. It requires `spi_toolslib` version 4.10.6 or higher and `spi_fitlib` version 1.2.0 or higher.

2 Getting started

Before installing `spi_obs_fit`, make sure that the ISDC support platform 6.3 or higher is installed on your system, and that you have installed the libraries `spi_toolslib` version 4.10.6 or higher and `spi_fitlib` version 1.2.0 or higher.

After downloading the `spi_obs_fit.tar.gz` file, step into a directory that should hold the distribution, move the `spi_obs_fit.tar.gz` file into this directory and type after the UNIX prompt `$` (don't type this prompt):

```
$ gunzip spi_obs_fit.tar.gz
$ tar xvf spi_obs_fit.tar
```

The first command uncompresses the distribution file, the second unpacks the files.

Before configuration, the distribution needs to be reset to a clean state. To do this, type

```
$ make distclean
```

Then, configure the distribution. It is assumed here that you have previously installed the ISDC support platform, thus you should type

```
$ ~/bin/ac_stuff/configure
```

Finally, build the distribution by typing

```
$ make global_install
```

To perform a unit test, type

```
$ make test
```

3 Parameter file

```
#####
#
#           Centre d'Etude Spatiale des Rayonnements
#           (in collaboration with ISDC)
#
#           SPI Observation Fitting
#
# -----#
#
# File:      spi_obs_fit.par
# Version:   3.0.0
# Component: IA
#
# Author:    Juergen Knoedlseder
#            knodlseder@cesr.fr
#            CESR
#
# Purpose:   Parameter file for the SPI Observation Fitting task
#
#####
#
# The input DOLs/filenames
#=====
ingrpDOL, s,q,"og_spi.fits",,,"Input Observation Group DOL or filename"
inebdsDOL,s,q,      "",,,"Energy boundary DOL or filename"
inpntDOL, s,q,      "",,,"Pointing DOL or filename"
ingtiDOL, s,q,      "",,,"Good Time Interval DOL or filename"
indtiDOL, s,q,      "",,,"Deadtime DOL or filename"
indspDOL, s,q,      "",,,"Detector Spectrum DOL or filename"
inpefDOL, s,q,      "",,,"PSD Efficiency DOL or filename"
inprfDOL, s,q,      "",,,"PSD Response DOL or filename"
inidxDOL, s,q,      "",,,"Background Model Index DOL or filename"
inirfDOL, s,q,      "irf.fits",,,"Instrument Response Index DOL or filename"
#
# The output DOLs/filenames
#=====
outgrpDOL,  s,q,      "og_spi.fits",,,"Output Observation Group DOL or filename"
outresDOL,  s,q,      "results.fits",,,"Result DOL or filename"
outimaidxDOL,s,q,      "image_idx.fits",,,"Result Image Index DOL or filename"
outimaDOL,  s,q,      "image.fits",,,"Result Image filename"
outebdsDOL, s,q,"energy_boundaries.fits",,,"Result Energy boundary DOL or filename"
outdspDOL,  s,q,      "evts_det_spec.fits",,,"Result Detector event spectra DOL or filename"
outspeidxDOL,s,q,      "sky_model_idx.fits",,,"Result Sky Model Index DOL or filename"
outspeDOL,  s,q,      "sky_model.fits",,,"Result Sky Model"
outbgmidxDOL,s,q,      "back_model_idx.fits",,,"Result Background Model Index DOL or filename"
outbgmDOL,  s,q,      "back_model.fits",,,"Result Background Model"
outdmpFile, s,h,      "",,,"Obsolete"
#
# Task parameters
#=====
dataSpaceEmin,r,q,"507.5",,,"Data-space energy band minimum (keV)"
```

```

dataSpaceEmax,r,q,"514.5",,, "Data-space energy band maximum (keV)"
dataSpaceEbin,r,q, "7.0",,, "Data-space energy binsize (keV)"
sumDataSpace, b,q,  yes,, "Sum data-space energy bins ?"
filterThres, r,q,  "0.0",,, "Exclude pointings that are worse than threshold"
warnThres, r,q,  "3.0",,, "Signal pointings that are worse than threshold"
#
# Sky model parameters
#=====
nmodel, i,q, 2,0,20,"Number of model components"
#
model01,s,q, "G0+0",,, "Model 1"
eng01, s,q, "511.0",,, "Model 1 energy range (keV)"
par01, s,q, "date 10 days",,, "Model 1 parameters"
#
model02,s,q, "bulge.fits[0]",,, "Model 2"
eng02, s,q, "511.0",,, "Model 2 energy range (keV)"
par02, s,q, "fit",,, "Model 2 parameters"
#
model03,s,q, "",,, "Model 3"
eng03, s,q, "",,, "Model 3 energy range (keV)"
par03, s,q, "",,, "Model 3 parameters"
#
model04,s,q, "",,, "Model 4"
eng04, s,q, "",,, "Model 4 energy range (keV)"
par04, s,q, "",,, "Model 4 parameters"
#
model05,s,q, "",,, "Model 5"
eng05, s,q, "",,, "Model 5 energy range (keV)"
par05, s,q, "",,, "Model 5 parameters"
#
model06,s,q, "",,, "Model 6"
eng06, s,q, "",,, "Model 6 energy range (keV)"
par06, s,q, "",,, "Model 6 parameters"
#
model07,s,q, "",,, "Model 7"
eng07, s,q, "",,, "Model 7 energy range (keV)"
par07, s,q, "",,, "Model 7 parameters"
#
model08,s,q, "",,, "Model 8"
eng08, s,q, "",,, "Model 8 energy range (keV)"
par08, s,q, "",,, "Model 8 parameters"
#
model09,s,q, "",,, "Model 9"
eng09, s,q, "",,, "Model 9 energy range (keV)"
par09, s,q, "",,, "Model 9 parameters"
#
model10,s,q, "",,, "Model 10"
eng10, s,q, "",,, "Model 10 energy range (keV)"
par10, s,q, "",,, "Model 10 parameters"
#
model11,s,q, "",,, "Model 11"
eng11, s,q, "",,, "Model 11 energy range (keV)"
par11, s,q, "",,, "Model 11 parameters"

```

```

#
model12,s,q,          "",,,"Model 12"
eng12, s,q,          "",,,"Model 12 energy range (keV)"
par12, s,q,          "",,,"Model 12 parameters"
#
model13,s,q,          "",,,"Model 13"
eng13, s,q,          "",,,"Model 13 energy range (keV)"
par13, s,q,          "",,,"Model 13 parameters"
#
model14,s,q,          "",,,"Model 14"
eng14, s,q,          "",,,"Model 14 energy range (keV)"
par14, s,q,          "",,,"Model 14 parameters"
#
model15,s,q,          "",,,"Model 15"
eng15, s,q,          "",,,"Model 15 energy range (keV)"
par15, s,q,          "",,,"Model 15 parameters"
#
model16,s,q,          "",,,"Model 16"
eng16, s,q,          "",,,"Model 16 energy range (keV)"
par16, s,q,          "",,,"Model 16 parameters"
#
model17,s,q,          "",,,"Model 17"
eng17, s,q,          "",,,"Model 17 energy range (keV)"
par17, s,q,          "",,,"Model 17 parameters"
#
model18,s,q,          "",,,"Model 18"
eng18, s,q,          "",,,"Model 18 energy range (keV)"
par18, s,q,          "",,,"Model 18 parameters"
#
model19,s,q,          "",,,"Model 19"
eng19, s,q,          "",,,"Model 19 energy range (keV)"
par19, s,q,          "",,,"Model 19 parameters"
#
model20,s,q,          "",,,"Model 20"
eng20, s,q,          "",,,"Model 20 energy range (keV)"
par20, s,q,          "",,,"Model 20 parameters"
#
# Background model parameters
#=====
bgmcomp01,s,q,  "fix",,,"Background model component 1 fit method"
bgmcomp02,s,q,  "dete",,,"Background model component 2 fit method"
bgmcomp03,s,q,  "dete",,,"Background model component 3 fit method"
bgmcomp04,s,q,  "dete",,,"Background model component 4 fit method"
bgmcomp05,s,q,  "",,,"Background model component 5 fit method"
bgmcomp06,s,q,  "",,,"Background model component 6 fit method"
bgmcomp07,s,q,  "",,,"Background model component 7 fit method"
bgmcomp08,s,q,  "",,,"Background model component 8 fit method"
bgmcomp09,s,q,  "",,,"Background model component 9 fit method"
bgmcomp10,s,q,  "",,,"Background model component 10 fit method"
bgmcomp11,s,q,  "",,,"Background model component 11 fit method"
bgmcomp12,s,q,  "",,,"Background model component 12 fit method"
bgmcomp13,s,q,  "",,,"Background model component 13 fit method"
bgmcomp14,s,q,  "",,,"Background model component 14 fit method"

```

```

bgmcomp15,s,q,      "",,,"Background model component 15 fit method"
bgmcomp16,s,q,      "",,,"Background model component 16 fit method"
bgmcomp17,s,q,      "",,,"Background model component 17 fit method"
bgmcomp18,s,q,      "",,,"Background model component 18 fit method"
bgmcomp19,s,q,      "",,,"Background model component 19 fit method"
bgmcomp20,s,q,      "",,,"Background model component 20 fit method"
#
# Response parameters
#=====
srcGamma,  r,h, "2.0",,,"Power-law E-gamma exponent gamma"
rspIntdlogE,r,h,"0.03",,,"Response interpolation log10 step size"
#
# Standard parameters
#=====
clobber,b,h,yes,,,"Overwrite existing output data ?"
verbose,i,h,2,0,4,"Information logging level"

```

Instead of specifying complete DOLs (Data Object Locations), which are composed of a filename plus the data structure extension (HDU), `spi_obs_fit` accepts also simple filenames and adds the appropriate data structure extensions. This means that **specified data structure extensions are ignored**. The only exception are the `model n` parameters which need an extension in case that a skymap has been specified (see below).

The parameters have the following meaning:

- `ingrpDOL` (optional) specifies the DOL or filename of the input Observation Group (HDU [GROUPING]) for which model fitting should be performed. The input group has to be of level BIN_I.
If an output Observation Group has been specified (parameter `outgrpDOL`), the specification of this parameter is optional. If the parameter is left blank, the output Observation Group will then be used as input Observation Group. Otherwise, the input Observation Group will be copied into the output Observation Group.
- `inebdsDOL` (optional) specifies the DOL or filename of an energy boundary definition (HDU [SPI.-EBDS-SET]). This data structure specifies the energy boundaries of the binned data.
If a [SPI.-EBDS-SET] element exists already in the input Observation Group, this element will be replaced by the specified DOL in the output Observation Group. Otherwise, the specified DOL will be attached to the output Observation Group. If left blank, it is assumed that a [SPI.-EBDS-SET] element exists already in the input Observation Group. If no such element is found, however, the task execution is aborted with an error message.
- `inpntDOL` (optional) specifies the DOL or filename of a Pointing definition (HDU [SPI.-OBS.-PNT]). This data structure specifies the SPI pointings during data taking.
If a [SPI.-OBS.-PNT] element exists already in the input Observation Group, this element will be replaced by the specified DOL in the output Observation Group. Otherwise, the specified DOL will be attached to the output Observation Group. If left blank, it is assumed that a [SPI.-OBS.-PNT] element exists already in the input Observation Group. If no such element is found, however, the task execution is aborted with an error message.
- `ingtiDOL` (optional) specifies the DOL or filename of a Good Time Interval definition (HDU [SPI.-OBS.-GTI]). This data structure specifies the time intervals that have been used for data taking.
If a [SPI.-OBS.-GTI] element exists already in the input Observation Group, this element will be replaced by the specified DOL in the output Observation Group. Otherwise, the specified DOL will

be attached to the output Observation Group. If left blank, it is assumed that a [SPI.-OBS.-GTI] element exists already in the input Observation Group. If no such element is found, however, the task execution is aborted with an error message.

- **indtiDOL** (optional) specifies the DOL or filename of a Deadtime data structure (HDU [SPI.-OBS.-DTI]). This data structure contains the livetime and the deadtime ratio for the binned data.

If a [SPI.-OBS.-DTI] element exists already in the input Observation Group, this element will be replaced by the specified DOL in the output Observation Group. Otherwise, the specified DOL will be attached to the output Observation Group. If left blank, it is assumed that a [SPI.-OBS.-DTI] element exists already in the input Observation Group. If no such element is found, however, the task execution is aborted with an error message.

- **indspDOL** (optional) specifies the DOL or filename of a Detector Spectra data structure (HDU [SPI.-OBS.-DSP]). This data structure contains the detector spectra for all pseudo-detectors and pointings.

If a [SPI.-OBS.-DSP] element exists already in the input Observation Group, this element will be replaced by the specified DOL in the output Observation Group. Otherwise, the specified DOL will be attached to the output Observation Group. If left blank, it is assumed that a [SPI.-OBS.-DSP] element exists already in the input Observation Group. If no such element is found, however, the task execution is aborted with an error message.

- **inpefDOL** (optional) specifies the DOL or filename of a PSD efficiency data structure (HDU [SPI.-OBS.-PEF]). This data structure contains the PSD efficiencies for the binned data.

If a [SPI.-OBS.-PEF] element exists already in the input Observation Group, this element will be replaced by the specified DOL in the output Observation Group. Otherwise, the specified DOL will be attached to the output Observation Group. If left blank, it is assumed that a [SPI.-OBS.-PEF] element exists already in the input Observation Group. If no such element is found, however, the task execution is aborted with an error message.

- **inprfDOL** (optional) specifies the DOL or filename of a PSD response data structure (HDU [SPI.-OBS.-PRF]). This data structure contains the PSD response for the binned data.

If a [SPI.-OBS.-PRF] element exists already in the input Observation Group, this element will be replaced by the specified DOL in the output Observation Group. Otherwise, the specified DOL will be attached to the output Observation Group. If left blank, it is assumed that a [SPI.-OBS.-PRF] element exists already in the input Observation Group. If no such element is found, however, the task execution is aborted with an error message.

- **inidxDOL** (optional) specifies the DOL or filename of a Background Model index data structure (HDU [SPI.-BMOD-DSP-IDX]). This data structure contains pointers to the components of the instrumental background model.

If a [SPI.-BMOD-DSP-IDX] element exists already in the input Observation Group, this element will be replaced by the specified DOL in the output Observation Group. Otherwise, the specified DOL will be attached to the output Observation Group. If left blank, it is assumed that a [SPI.-BMOD-DSP-IDX] element exists already in the input Observation Group. If no such element is found, however, the task execution is aborted with an error message.

- **inirfDOL** (optional) specifies the DOL or filename of a Instrument Response Function (IRF) index (HDU [SPI.-IRF.-RSP-IDX]). This data structure contains an index to the response functions that should be used for response calculation. It is only needed if intensity skymaps or individual sources have been specified as sky models.

If a [SPI.-IRF.-RSP-IDX] element exists already in the input Observation Group, this element will be replaced by the specified DOL in the output Observation Group. Otherwise, the specified DOL will be

attached to the output Observation Group. If left blank, it is assumed that a [SPI.-IRF.-RSP-IDX] element exists already in the input Observation Group. If no such element is found, however, the task execution is aborted with an error message.

- `outgrpDOL` (optional) specifies the DOL or filename of the output Observation Group (HDU [GROUPING]). The output Observation Group will be a copy of the input Observation Group plus the resulting sky image index, sky model index and background model index attached.

If an input Observation Group has been specified (parameter `ingrpDOL`), the specification of this parameter is optional. If the parameter is left blank, the input Observation Group will then be used as output Observation Group.

- `outresDOL` specifies the DOL or filename of the fitting results (HDU [SPI.-SRCL-RES]). After execution of the task, this file will contain the model fitting results.

In the actual version of `spi_obs_fit`, no results are written into this data structure.

- `outimaidxDOL` specifies the DOL or filename of the output sky image index (HDU [SPI.-SKY.-IMA-IDX]). After execution of the task, this index will contain pointers to the sky images that have been provided as models and that have been scaled by the best fitting scaling factors.

This index will be attached to the output Observation Group. Any index of the same type that may already exist in the output Observation Group will be detached before. If the specified DOL is identical to an already existing DOL, this DOL will be deleted if the `clobber` parameter is `yes` (otherwise the task will abort with an error).

- `outimaDOL` specifies the filename of the sky image data structure (HDU [SPI.-SKY.-IMA]). **Note that this filename is relative to the sky image index (parameter `outimaidxDOL`), and that no HDU extension should be provided for this parameter.** After execution of the task, this file will contain the sky images that have been provided as models and that have been scaled by the best fitting scaling factors.

- `outebdsDOL` specifies the DOL or filename of the resulting energy boundary definition (HDU [SPI.-EBDS-SET]). After execution of the task, this file will contain the energy boundaries of the data that have been selected for fitting.

- `outdspDOL` specifies the DOL or filename of the resulting event spectra (HDU [SPI.-OBS.-DSP]). After execution of the task, this file will contain the event spectra of the data that have been selected for fitting.

- `outspeidxDOL` specifies the DOL or filename of the sky model index (HDU [SPI.-SDET-SPE-IDX]). After execution of the task, this index will contain pointers to sky models that have been provided on input and that have been scaled by the best fitting scaling factors.

This index will be attached to the output Observation Group. Any index of the same type that may already exist in the output Observation Group will be detached before. If the specified DOL is identical to an already existing DOL, this DOL will be deleted if the `clobber` parameter is `yes` (otherwise the task will abort with an error).

- `outspeDOL` specifies the filename of the sky model data structure (HDU [SPI.-SDET-SPE]). **Note that this filename is relative to the sky model index (parameter `outidxDOL`), and that no HDU extension should be provided for this parameter.** After execution of the task, this file will contain the sky models that have been provided on input and that have been scaled by the best fitting scaling factors.

- `outbgmidxDOL` specifies the DOL or filename of the background model index (HDU [SPI.-BMOD-DSP-IDX]). After execution of the task, this index will contain pointers to the background model components that have been provided on input and that have been scaled by the best fitting scaling factors.

This index will be attached to the output Observation Group. Any index of the same type that may already exist in the output Observation Group will be detached before. If the specified DOL is identical to an already existing DOL, this DOL will be deleted if the `clobber` parameter is `yes` (otherwise the task will abort with an error).

- `outbgmDOL` specifies the filename of the background model data structure (HDU [SPI.-BMOD-DSP]). **Note that this filename is relative to the background model index (parameter `outbgmidxDOL`), and that no HDU extension should be provided for this parameter.** After execution of the task, this file will contain the background model components that have been provided on input and that have been scaled by the best fitting scaling factors.
- `outdmpFile` this parameter is obsolete since version 2.0.0.
- `dataSpaceEmin` specifies the lower boundary of the range of the data-space energy bins that should be considered for model fitting. All data-space energy bins that overlap with the interval [`dataSpaceEmin`,`dataSpaceEmax`] will be used for model fitting.
- `dataSpaceEmax` specifies the upper boundary of the range of the data-space energy bins that should be considered for model fitting. All data-space energy bins that overlap with the interval [`dataSpaceEmin`,`dataSpaceEmax`] will be used for model fitting.
- `dataSpaceEbin` specifies the binsize of the bins that should be used for fitting. This parameter is only valid if `sumDataSpace = yes`. In this case there are 3 options.
 If `dataSpaceEbin` is positive, the energy band specified by `dataSpaceEmin` and `dataSpaceEmax` is split into bins of width `dataSpaceEbin`.
 If `dataSpaceEbin` is negative, the energy band specified by `dataSpaceEmin` and `dataSpaceEmax` is split into `-dataSpaceEbin` logarithmic energy bins. For example, for `dataSpaceEbin = -100`, 100 logarithmic energy bins are used.
 If `dataSpaceEbin = 0.0` a single energy bin will be used with binwidth `dataSpaceEmax - dataSpaceEmin`.
- `sumDataSpace` specifies if selected data-space energy bins should be combined. If `yes` is specified, the energy bins of the input data-space will be resampled into the energy bin definition specified by `dataSpaceEmin`, `dataSpaceEmax`, and `dataSpaceEbin` (see above). If `no` is specified, the original energy bin width definition will be used (i.e. the energy bins are not resampled).
- `filterThres` specifies a threshold (in units of standard deviation) for data filtering. All pointings (summed over pseudo-detectors) for which the model differs by more than the specified number of standard deviations from the observed counts will be excluded from the analysis (they will also be excluded in the result datasets). The model fit is then repeated without these bins. If `filterThres = 0.0`, no data filtering will be applied.
- `warnThres` specifies a threshold (in units of standard deviation) for emission of a warning message in the log file that indicates model-data mismatch. The logic is identical than for the `filterThres` parameter, yet instead of excluding the pointing only a warning will be issued.
- `nmodel` specifies the number of sky model components that should be fitted. This number has to be comprised between 1 and 20.
- `modelnn` defines the sky model component `nn`, where `nn` is comprised between 01 and 20.

There are three options: (1) specify a DOL of a sky intensity distribution, (2) specify a DOL of a sky data-space model, and (3) specify the coordinates of a point source.

If a DOL of a sky intensity distribution (in form of an image) is specified (option 1), the corresponding image is loaded and convolved with the instrumental response function in order provide a data-space model of the sky intensity distribution. `spi_obs_fit` is rather tolerant in reading different kinds of FITS images and requires only a minimum amount of information in the FITS header. Therefore,

the source photon energy range covered by the image has to be specified by the parameter `engnn` (see below).

The image can be presented in either celestial or galactical coordinate systems. `spi_obs_fit` automatically recognises this by checking the `CTYPE1` keyword of the image: if it contains `GLON` or `GALACTIC` the image will be assumed in galactic coordinates, otherwise it will be loaded in celestial coordinates. In case of doubts about how the input image was loaded by `spi_obs_fit` one may set the parameter `verbose = 3`. The log file will then contain information about the images that have been loaded.

The units of the image pixels are assumed to be of the general form $\text{photons cm}^{-2} \text{s}^{-1} (\text{sr}^{-1}) (\text{keV}^{-1})$, where sr^{-1} and keV^{-1} are optional. `spi_obs_fit` will check the `BUNIT` keyword of the image to determine which type of pixel unit is present. If no `BUNIT` keyword is present the image will be assumed to be in units of $\text{photons cm}^{-2} \text{s}^{-1} \text{sr}^{-1}$.

In order to be tolerant to different image file types, no HDU extension type is a priori assumed for the image, hence the user has to specify the extension number or name in the `DOL`. For example, to load the image found in extension number 2 from the file `myimage.fits` one has to specify `myimage.fits[2]` (note that extensions are counted from 0 in FITS!)

If the input FITS file contains multiple images in a 3rd dimension (i.e. the `NAXIS` keyword is ≥ 3) `spi_obs_fit` is able to load a subset of those and to combine these *image slices* into a single image. In this case one has to append the requested range of image slices to the `DOL`, separated by a comma. The range of allowed indices starts from 1 and runs up to the number of available image slices (given by the `NAXIS3` keyword). For example, to load and add the 2nd and 3rd image from image extension number 2 in the file `myimage.fits` one has to specify `myimage.fits[2],2-3`. To extract only layer 7 one should specify `myimage.fits[2],7`. If nothing is specified, the 1st image layer will be loaded.

If a `DOL` of a sky data-space model is specified (option 2), `spi_obs_fit` will take this data-space model directly without passing by the convolution step. This may accelerate the fitting process considerably! The `DOL` has to be of HDU type `SPI.-SDET-SPE` and it has to have the same number of pointings and the same number of pseudo-detectors as the specified observation group. Concerning the energy bins, the model needs either to have the same energy binning definition as the input observation group, or the same energy binning that is requested for model fitting (this means that one can use this option to refit models from a previous run of `spi_obs_fit`).

If the coordinates of a point source are specified (option 3), `spi_obs_fit` will build the instrumental response for a point source at the specified position for this sky model. If 2 arguments are given it is assumed that the coordinate is given in degrees. For example, `102.0+13.0` signifies a point source at Right Ascension 102.0 and declination 13.0 degrees. If the arguments are preceded by the letter `G` the coordinates are assumed as galactic coordinates instead of celestial coordinates. For example, the Crab nebula may be specified using `G184.56-5.78`. If six arguments are given, the coordinate is assumed to be given in sexagesimal format (e.g. hours minutes seconds degrees minutes seconds). For example, Cyg X-1 may be specified as `19 58 21.68 +35 12 05.8`. All other coordinate formats will be rejected by `spi_obs_fit`.

- `engnn` defines the source photon energy range that is covered by the sky model component `nn`, where `nn` is comprised between 01 and 20. This energy range specifies the spectral properties of the model that should be fitted. 3 cases are to be distinguished:

Gamma-ray line source analysis: For a gamma-ray line source a single energy value should be specified which gives the gamma-ray line energy in keV. The input photon spectrum is then assumed of Dirac-shape at the given energy. If the selected data-space has multiple energy bins, the source will only lead to an instrumental response in the single energy bin into which the line energy falls. Thus for gamma-ray line source analysis usually only a single data-space energy bin is set up (by making sure that `dataSpaceEbin = dataSpaceEmax - dataSpaceEmin` and by selecting `sumDataSpace = yes`).

Continuum source analysis: For a continuum source an energy interval in the form `engnn = min-max` should be specified, where `min` and `max` are the minimum and maximum photon energy

(in keV) of the source, respectively. Since `spi_obs_fit` does not consider off-diagonal elements of the instrumental response matrix, it is sufficient that the specified energy range covers the data-space energy range (given by the interval `[dataSpaceEmin,dataSpaceEmax]`). Within this energy interval, the source is assumed to have a powerlaw with an index that is specified by the parameter `srcGamma`.

Spectral (gamma-ray line) analysis: The spectral gamma-ray line analysis consists of model fitting in a large number of data-space energy bins, defined by the parameters `dataSpaceEmin`, `dataSpaceEmax`, `dataSpaceEbin`, and `sumDataSpace = yes`. For this type of analysis the parameter `engnn` has to be left blank, and the fit method parameter `parnn` should contain `EBIN` (see below). In this case the source photon energy range is assumed to be identical to the data-space energy range. Within this range the source is assumed to have a constant flux.

Alternatively, one may specify - similar to the continuum analysis - an energy interval `engnn = min-max` that covers the energy range covered by `dataSpaceEmin-dataSpaceEmax`; at the same time, the spectral index `srcGamma` should be set to `0.0` (**this alternative method should lead to faster execution than the first method.**)

The actual source flux for each energy bin is then determined by the fitting procedure, providing a source spectrum. Note that this analysis is very similar to the analysis described for continuum sources, yet here no powerlaw slope has been assumed for the source spectrum (the reader may verify that for sufficiently fine energy binning the result of both analyses converge).

- `parnn` specifies the fitting method for sky model component `nn`, where `nn` is comprised between 01 and 20. These methods define the parameters that should be used to fit the model to the data (the string is interpreted case insensitive).

The standard case is a single scaling parameter for the sky model, and in this case `FIT` should be specified. Note that by default, a point source is scaled to the flux of 10^{-5} photons $\text{cm}^{-2} \text{s}^{-1}$ (keV^{-1}). Gamma-ray line sources are given in units of total line flux (photons $\text{cm}^{-2} \text{s}^{-1}$) while continuum sources are given in intensity units (photons $\text{cm}^{-2} \text{s}^{-1} \text{keV}^{-1}$). If the sky model is an image, the scaling factor is simply a multiplying factor for the image (knowing the flux in the input image and multiplying by the fitted scaling factor gives the fitted SPI flux).

If the flux (or scaling) of the source model should be kept fixed (instead of being fitted) one has to specify `FIX`. Note that point sources are internally scaled by `spi_obs_fit` to a flux of 10^{-5} photons $\text{cm}^{-2} \text{s}^{-1}$ (keV^{-1}), thus specifying simply `FIX` will keep the point source at this level. Using `FIX=x` the level at which the sky model should be kept fixed can be specified. For point sources, `x` is the flux in units of 10^{-5} photons $\text{cm}^{-2} \text{s}^{-1}$ (keV^{-1}). For images, `x` is the image scaling factor.

For continuum source analyses or spectral (gamma-ray) line analyses the source flux should in general be fitted in a number of energy bins. This can be done by setting the fitting method to `EBIN`. In this case the sky model has a free flux/scaling for each of the data-space energy bins (which are specified by the `dataSpaceEmin`, `dataSpaceEmax`, `dataSpaceEbin`, and `sumDataSpace` parameters). For example, with the setting

```
dataSpaceEmin = 500.0
dataSpaceEmax = 520.0
dataSpaceEbin = 1.0
sumDataSpace = yes
```

the `EBIN` method will provide 20 flux values for 1 keV wide energy bins in the interval 500-520 keV.

There are further fitting options that may be combined with the precedently mentioned methods, and that concern the time variation of the sky model. So far, only a single scaling factor could be obtained for the sky model over the entire observing period. However, many gamma-ray sources are variable, and `spi_obs_fit` allows to determine the lightcurve of the objects by fitting individual scaling factors for specific time intervals. The following methods are implemented:

`DATE`: intervals of constant duration (standard lightcurve method)

`ORBIT`: one scaling factor per satellite orbit (3 days)

`POINT`: one scaling factor per telescope pointing

The DATE method takes as additional parameters the length of the time interval, expressed either in seconds, minutes, hours, days, or years. For example, DATE 3 HOURS will split the observation interval into equal time intervals of 3 hour duration. If no unit is specified, seconds will be assumed.

Constrained fitting may be performed by adding the MIN=xx or MAX=xx options, where xx specifies a numerical extreme value that the corresponding model parameter may take. If a boundary is reached, spi_obs_fit will fix the corresponding value to this boundary and optimise the remaining parameters. Consequently, the fixed parameter has no associated statistical uncertainty (it is set to 0.0).

Last but not least, individual source scaling factors may be assumed for each SPI detector. Scientifically this is not very meaningful, except for source polarisation studied using double detector events. However, this option may be useful to cross check the analysis result by verifying that the source flux determined with each of the telescope detectors is identical. This option is enabled by adding the DETE method to the parameter string.

Thus, overall, the variety of possible source model parametrisations is very large, and the following examples should guide the user on what he may need to specify for his own analysis:

FIT: determine the source flux

FIT, EBIN: make a source spectrum

FIT, EBIN, MIN=0.0: make a non-negative source spectrum

EBIN: this is equivalent to FIT, EBIN

DATE 10 days: make a source lightcurve on 10 days intervals

FIT, DATE 10 days: this is again equivalent to the preceding method

DATE 30 days, EBIN: make a source lightcurve for each energy bin on 30 days intervals

FIT, DETE: determine the source flux on each detector

DETE: equivalent to FIT, DETE

ORBIT, DETE: determine source flux in each orbit and on each detector

As probably recognised, the FIT keyword can be omitted as soon as a time variation or detector variation method has been specified (since the purpose of FIT is only to distinguish between source fitting and keeping the source at a fixed flux level; conversely, combining FIX with any of the other methods does not make sense since nothing is fitted in any case).

- **bgmcompnn** specifies parameters for the background model component *nn*, where *nn* is comprised between 01 and 20. These methods define the parameters that should be used to fit the background model to the data (the string is interpreted case insensitive). Note that a SPI background model may contain various components (grouped by an index table) and that *nn* refers here to the component number (starting from 1 for the first component).

The methods that are available for background model fitting are basically equivalent to those available for source model fitting, which are explained above for the parameters **parnn**. Internally, source models and background models are equivalent for spi_obs_fit, so there is no reason not to make available the same methods for both type of models (that is basically the reason why POINT, ORBIT, and DETE are also available for sky models).

An additional option exists to accommodate for changes in the instrumental background count rates after the failure of Germanium detectors. If the keyword GEDFAIL is appended to the parameter, separate background model scaling factors are fitted for each combination of failed Germanium detectors. For example, if the observation group covers data from the beginning of the INTEGRAL (all detectors enabled) to revolution 260 (at which detector 2 and 17 are not more available), the option GEDFAIL leads to a splitting of the observation group into 3 intervals, the first covering the data for which all detectors were available, the second covering data where detector 2 failed, and the third covering data where both detector 2 and 17 failed. GEDFAIL may be combined with any of the available options (e.g. ORBIT, DATE, etc.). In the case of ORBIT, GEDFAIL, for example, orbits that are covering the moment of detector failure are split into two intervals, one covering the interval before the failure, another covering the interval after the failure.

Fixing a background model to a specific level can be done by using the method **FIX=x** (similar to the handling of sky models), where *x* specifies the scaling of the background model component. If **=x** is

omitted, the background model component will be kept at its original level.

The methods available to specify the model parameters in the pointing dimension of the SPI data-space are: FIT (or GLOBAL), POINT, ORBIT, and DATE. For compatibility reasons with former versions of `spi_obs_fit`, the option GLOBAL was kept, yet for symmetry with the sky models we encourage the usage of FIT. A special feature of the `bgmcompn` parameter is that if it is left blank, FIT is assumed automatically.

Typically, the most crucial part of SPI data analysis is the modelling of the instrumental background, and in particular the prediction of the time variability of the background. In many cases, the rate of saturated events in the germanium detectors (GEDSAT) is a good first order predictor, and a background model based on this tracer is a good starting point. On second order, however, GEDSAT is not sufficient and the background model needs refinement. Such refinement can be done in `spi_obs_fit` by adjusting the time variation of the background on specific time scales, using one of the methods POINT, ORBIT, and DATE. In general, this implies a loss of detection sensitivity, and the trade-off between systematical errors (introduced by uncertainties in the background model) and statistical errors (introduced by the counting statistics) has to be evaluated. Yet at least for consistency checking, runs using the time variability methods for the background model are encouraged. Note that the POINT method leads to a large number of model parameters, which will in general lead to large memory requirements for `spi_obs_fit`, and which will also substantially slow down the execution time of the task. ORBIT seems to provide satisfactorily results in many cases.

Fitting background model parameters for each of the pseudo-detectors can be done by adding DETE to the parameter. For example, ORBIT, DETE fits one background model scaling factor for each orbit and each detector. Specifying DETE alone (which is equivalent to FIT, DETE) fits one background model scaling factor for each detector. In general, adding the DETE method provides basically no sensitivity penalty but reduces the systematic uncertainties of the background model. Hence the usage of DETE is largely recommended (yet, in combination with time variation methods the number of model parameters may become very large, leading to huge memory requirements and slowing down the fitting process).

Fitting background model parameters for each of the data-space energy bins can be done by adding the EBIN keyword. This is the recommended standard method for spectral analysis. In this case the spectral shape of the instrumental background will be determined (or better adjusted) by `spi_obs_fit`. Without the EBIN method, the background model as stored in the Observation Group would need to predict the spectral background shape precisely. So again, to reduce systematic uncertainties in spectral studies, the usage of EBIN is recommended.

Constrained fitting may be performed by adding the MIN=xx or MAX=xx options, where xx specifies a numerical extreme value that the corresponding model parameter may take. If a boundary is reached, `spi_obs_fit` will fix the corresponding value to this boundary and optimise the remaining parameters. Consequently, the fixed parameter has no associated statistical uncertainty (it is set to 0.0).

Note that a former versions of `spi_obs_fit` used the methods GLOBAL-E, DETE-E, POINT-E, ORBIT-E, and DATE-E to specify energy dependent fitting. These methods are now obsolete.

- `srcGamma` specifies the gamma-ray source power-law $E^{-\gamma}$ slope γ for sky image convolution in the case that the sky image describes a continuum source. The default value is `srcGamma = 2.0`.
- `rspIntdlogE` specifies the number of logarithmic energy integration steps for sky image convolution in the case that the sky image describes a continuum source. The default value is `rspIntdlogE = 0.03`.
- `clobber` specifies if existing output data structures should be overwritten or not. If `yes` is specified, the executable will notify the user about the deletion of any file. If `no` is specified and the executable attempts to overwrite existing data, the task will exit with an error message.
- `verbose` specifies the level of information that `spi_obs_fit` will dump into the log file. The following levels exist (recommended level is 2):


```
#
bgmcomp01,    s,q,          "fix",,, "Background model component 1 fit method"
bgmcomp02,    s,q,    "dete,ebin",,, "Background model component 2 fit method"
bgmcomp03,    s,q,    "dete,ebin",,, "Background model component 3 fit method"
bgmcomp04,    s,q,    "dete,ebin",,, "Background model component 4 fit method"
```

Note that here a spectrum from 500-520 keV is produced, with 1 keV energy bin size.

5 Interface definition

TBW

6 Algorithms

TBW

7 Error codes

The executable `spi_obs_fit` may stop with the following error codes:

<code>SPI_OBS_FIT_ERROR_MEM_ALLOC</code>	-233500
<code>SPI_OBS_FIT_ERROR_BAD_PARAMETER</code>	-233501
<code>SPI_OBS_FIT_ERROR_BAD_EBOUNDS</code>	-233502
<code>SPI_OBS_FIT_ERROR_NO_MINIMUM</code>	-233503
<code>SPI_OBS_FIT_ERROR_NO_MAXIMUM</code>	-233504
<code>SPI_OBS_FIT_ERROR_FOPEN</code>	-233505
<code>SPI_OBS_FIT_ERROR_UNKNOWN_DOL_TYPE</code>	-233506
<code>SPI_OBS_FIT_ERROR_INVALID_COORD</code>	-233507
<code>SPI_OBS_FIT_ERROR_BAD_NUM_PNT</code>	-233508
<code>SPI_OBS_FIT_ERROR_SOFTWARE_BUG</code>	-233509

They have the following meaning:

- `SPI_OBS_FIT_ERROR_MEM_ALLOC` : the allocation of dynamical memory has failed. Probable your system resources are too limited to run this task.
- `SPI_OBS_FIT_ERROR_BAD_PARAMETER` : invalid task parameter found.
- `SPI_OBS_FIT_ERROR_BAD_EBOUNDS` : invalid energy boundaries found.
- `SPI_OBS_FIT_ERROR_NO_MINIMUM` : no minimum found in energy range.
- `SPI_OBS_FIT_ERROR_NO_MAXIMUM` : no maximum found in energy range.
- `SPI_OBS_FIT_ERROR_FOPEN` : unable to open/create ASCII file.
- `SPI_OBS_FIT_ERROR_UNKNOWN_DOL_TYPE` : specified image DOL contains no image.
- `SPI_OBS_FIT_ERROR_INVALID_COORD` : specified coordinate string is not valid.
- `SPI_OBS_FIT_ERROR_BAD_NUM_PNT` : invalid number of pointings found in pointing file.

- `SPI_OBS_FIT_ERROR_SOFTWARE_BUG` : `spi_obs_fit` detected an inconsistency that should never occur.

In addition, all errors that may occur in calls to ISDC support functions (such as for example DAL, RIL or PIL) are forwarded. Please consult the ISDC web pages for getting information about these error codes.