spi_psd_efficiency

User Manual

Version 1.4.3
8 September 2003

Jürgen Knödlseder
Centre d'Etude Spatiale des Rayonnements
knodlseder@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

The executable `spi_psd_efficiency` derives the PSD efficiency from an input group (science window group, observation group, or index group) for a user defined time interval. The PSD efficiency is defined as the fraction of all PSD events (i.e. all events within the PSD energy range) that is detected by the PSD sub-assembly. **The PSD efficiency should not be confused with the PSD background rejection efficiency, which is derived using the task `spi_psd_si`**. PSD efficiencies are derived for all 19 detection channels separately.

There are two PSD efficiencies that have to be distinguished:

- the total PSD efficiency `PSD_EFF` that is defined as the fraction of PSD events that are indeed detected by the PSD sub-assembly.
- the analysis error-free PSD efficiency `PSD_EFF_NOERR` that is defined as the fraction of PSD events that are indeed detected by the PSD sub-assembly and for which no pulse shape analysis errors occured.

Pulse shape analysis errors may occur in case of saturated events or noise triggers (see the `spi_psdlib` User Manual for a complete description of pulse shape analysis errors). It is the PSD efficiency `PSD_EFF_NOERR` that should be used for scientific analysis, since all PSD events with pulse shape analysis errors have no associated valid single/multiple flag information, hence they should be classified as single detector events (SE).

In addition to the efficiencies, `spi_psd_efficiency` derives also the PSD event lower and upper energy thresholds and threshold widths for all channels. Only within the lower and upper energy threshold, the PSD efficiency can be considered as constant.

`spi_psd_efficiency` has been designed to execute in any kind of pipeline, such as the science window pipeline or the revolution pipeline. However, it can also be applied to an observation group for **scientific analysis, where the efficiency parameters will be required for PSD response calculation**. Since a sufficient number of PSD events is required for threshold and efficiency determination, the **revolution pipeline is the priviledged location for using `spi_psd_efficiency` in a routinely manner**. `spi_psd_efficiency` has been designed to append regularily parameters to a file, and hence to build a time history of the PSD efficiencies and thresolds. To make the logic work, `spi_psd_efficiency` needs an index group on input, which it will search automatically for sufficient **new data** (i.e. data that are dated after the last entry in result file) for efficiency and threshold computation. If not enough new data is available, `spi_psd_efficiency` will do nothing.

`spi_psd_efficiency` is written in the ANSI C++ language. The task has been developped under ISDC support platform 5.2 and requires `spi_psdlib` version 2.0.2 and `spi_toolslib` version 3.4.3 or higher.

2 Getting started

Before installing `spi_psd_efficiency`, make sure that the ISDC support platform 5.2 or higher is installed on your system, and that you have installed the libraries `spi_psdlib` version 2.0.2 and `spi_toolslib` version 3.4.3 or higher.

After downloading the `spi_psd_efficiency.tar.gz` file, step into a directory that should hold the distribution, move the `spi_psd_efficiency.tar.gz` file into this directory and type:

```
$ gunzip spi_psd_efficiency.tar.gz
$ tar xvf spi_psd_efficiency.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
```

Make sure that the test data `spi_test_data-1.0.tar.gz` are available at your site (they should reside in a directory whose name is defined by the `ISDC_TEST_DATA_DIR` environment variable).

3 Parameter file

```
#############################################################################
#                                                                           #
#               Centre d'Etude Spatiale des Rayonnements                    #
#                     (in collaboration with ISDC)                          #
#                                                                           #
#                     SPI PSD efficiency calculation                        #
#                                                                           #
# ------------------------------------------------------------------------- #
#                                                                           #
#  File:      spi_psd_efficiency.par                                        #
#  Version:   1.4.2                                                         #
#  Component: osm                                                           #
#                                                                           #
#  Author:    Juergen Knoedlseder                                           #
#             knodlseder@cesr.fr                                            #
#             CESR                                                          #
#                                                                           #
#  Purpose:   Parameter file of the SPI PSD efficiency calculation          #
#             executable                                                    #
#                                                                           #
#  History:   1.4.2   5-Feb-2003  First ISDC delivery (Rev. 4)              #
#                                                                           #
#############################################################################
#
# Input DOLs
#===========
inDOL,    s, ql, "swg_prp_idx.fits[1]",,,"Input Group DOL (SWG/OG/IDX)"
coeffDOL, s, ql, "spi_gain_coeff_001.fits[1]",,,"Gain correction DOL (File/IDX)"
alertDOL, s, ql, "psd_limits_idx.fits[1]",,,"Alert Limit DOL (File/IDX)"
#
# Output DOL
#===========
outDOL, s, ql, "efficiency.fits",,,"Output DOL (HDU optionally)"
#
# OBT limits
#===========
minOBT, s, ql,     "",,,"Event usage minimum OBT"
maxOBT, s, ql,     "",,,"Event usage maximum OBT"
append, b, ql,    yes,,,"Append minimum OBT to last results ?"
slice,  b, ql,    yes,,,"Split time interval in constant ONTIME intervals ?"
nopart, b, ql,    yes,,,"Skip partial time intervals ?"
ontime, r, ql, 3600.0,,,"Constant ONTIME slice (seconds)"
#
# PSD discrimination
#===================
onground,   b, h,  no,,,"Use onground discrimination ?"
thresnoerr, b, h, yes,,,"Derive threshold from PE w/o errors ?"
#
# Efficiency fit parameters
#==========================
engmin, r, h,   50.0, 0.0, 8000.0, "Fit energy range minimum (keV)"
```

```
engmax, r, h, 8000.0, 0.0, 8000.0, "Fit energy range maximum (keV)"
engbin, r, h,   10.0, 0.0,  500.0, "Spectral bin size for fitting (keV)"
ignore, b, h,    yes,,,            "Ignore fit alerts ?"
#
# Diagnostic parameters
#======================
saveSpec,  b, h,          no,,, "Save PE/(SE+PE) in OBS spectra ?"
saveDOL,   s, h, "spec.fits",,, "DOL for spectra saving (HDU optionally)"
reportfit, b, h,          no,,, "Report efficiency fitting details ?"
#
# Limit checking definitions
#===========================
limcheck, b, h,   yes,,, "Perform limit checking ?"
alert0,   b, h,   yes,,, "Generate level 0 alerts ?"
alert1,   b, h,   yes,,, "Generate level 1 alerts ?"
alert2,   b, h,   yes,,, "Generate level 2 alerts ?"
alert3,   b, h,   yes,,, "Generate level 3 alerts ?"
minPE,    i, h, 10000,,, "Minimum number of PSD events for limit checking"
#
# ISDC Standard Parameters
#=========================
clobber, b, h, no,,,   "Overwrite existing data structures ?"
mode,    s, h, "ql",,, "Execution mode"
```

The following parameters have to be specified:

- `inDOL` specifies the input DOL (science window, observation group, or index file) for which the PSD efficiencies and thresholds should be derived. A ISDC level of `PRP` is sufficient if a calibration coefficient file or index file is specified in the `coeffDOL` parameter. Energy correction is in this case performed *on-the-fly*. However, if the input group is of level `COR`, the calibrated energy information from the corresponding data structures is taken, irrespectively of the presence of a calibration DOL. Note that if `onground` pulse shape analysis errors should be considered, the `COR` level is needed in any case!
- `coeffDOL` specifies a calibration file `[SPI.-COEF-CAL]` or calibration index file `[SPI.-COEF-CAL-IDX]` that is used for energy calibration **if the ISDC level of the input group is** `PRP` (otherwise the energy calibration is extracted from the `COR` data).
- `alertDOL` (optional) if alert limit checking is requested (`limcheck = yes`), this parameter specifies the DOL of the alert limit file `[SPI.-ALRT-LIM]` or the alert limit index `[SPI.-ALRT-LIM-IDX]` (including the HDU).
- `outDOL` specifies the output **filename** into which the efficiency and threshold information is written. The specification of the HDU `[SPI.-EFFI-PSD]` is optional, but not required by the task. If the data structure exists already, `spi_psd_efficiency` appends rows to the existing table. If the data structure or the file does not exist, `spi_psd_efficiency` creates a new file/HDU.
- `minOBT` specifies the minimum OBT limit of the events that should be used for efficiency and threshold determination. The OBT format is a character string. Leading `0` may be omitted. If the character string is empty, or if any non-number character is specified (such as "no" for example), no mimimum OBT limit is applied (and data accumulation starts with the first event in the input group).
- `maxOBT` specifies the maximum OBT limit of the events that should be used for efficiency and threshold determination. The OBT format is a character string. Leading `0` may be omitted. If the character string is empty, or if any non-number character is specified (such as "no" for example), no maximum OBT limit is applied (and data accumulation stops with the last event in the input group).

- `append` specifies if the minimum OBT limit should be set to the last OBT that occurs in the output file (specified by `outDOL`) in order to produce a continuous set of PSD efficiencies and thresholds. The last OBT will be extracted from the keyword `OBTLAST` in the output file. **This parameter is only active if `minOBT` has not been set by the user**, i.e. `minOBT` has precedence and will not be overwritten.

- `slice` specifies if the input group should be "sliced" into time frames of constant `ONTIME` (the `ONTIME` is the time, specified in seconds, during which SPI science data were accumulated and made available to the observer).

- `nopart` (optional) if `slice = yes`, specifies if partial time slices, i.e. time slices with durations that are shorter than the requested `ONTIME`, should be skipped. Partial time slices may occur at the end of a data stream, and to assure a uniform quality of the PSD efficiency parameters it is recommended to set this parameter to `yes`. Together with `append = yes`, re-execution of `spi_psd_efficiency` at a later time will append new time slices that start with the OBT of the last appended time slice.

- `ontime` (optional) if `slice = yes`, specifies the `ONTIME` duration of each time slice in seconds. Note that the last time slice has generally an effective `ONTIME` that is shorter than the specified value, since in general, the available `ONTIME` is not an integer multiple of the value specified by `ontime`.

- `onground` specifies if onground or onboard analysis errors should be considered for PSD efficiency (and optionally threshold) determination. The result parameter `PSD_EFF_NOERR` gives the PSD efficiency excluding all events for which a pulse shape analysis error occured. Pulse shape analysis errors may occur onboard, hence these events are excluded in any case for efficiency calculation. However, PSD onground discrimination may add additional analysis errors, for example in the case that the PSD configuration is not available. **If onground PSD discrimination is used for scientific analysis – the default selection – also onground analysis errors should be considered for PSD efficiency calculation.**

- `thresnoerr` specifies whether the PSD threshold determination should be based on all PSD events (select `no`) or only on PSD events without pulse shape analysis error (select `yes`). As default, use `thresnoerr = yes` (`no` may be used for PSD performance analysis if for some reason many – or all – PSD events have analysis errors).

- `engmin` specifies the minimum energy limit (in units of keV) for PSD efficiency and threshold determination. This limit should be set well below the PSD lower energy threshold (a few 100 keV), but a value of `0` should be avoided to exclude possible invalid low energy events that may pollute the data (e.g. events for which no corrected energy is available). In principle, there is no need to alter the default setting of 50 keV.

- `engmax` specifies the maximum energy limit (in units of keV) for PSD efficiency and threshold determination. This limit should be set well above the PSD upper energy threshold (about 2.3 MeV), and there is in principle no need to change the default setting of 8000 keV.

- `engbin` specifies the energy binning used for PSD efficiency and threshold determination. As default, a value of 10 keV is proposed.

- `ignore` specifies if PSD efficiency fit errors should be ignored. In this case, corresponding entries in the `SPI.-EFFI-PSD` table will be set to `0`, yet the program execution is not stopped. As default, specify `yes`, if using `spi_psd_efficiency` in a pipeline (this avoids task errors). In case of a fit error, **optional alert limit checking will also be suppressed if `ignore=yes`**.

- `saveSpec` specifies if the event spectra that are used for PSD efficiency and threshold determination should be stored in an ISDC `OBS` data structure for diagnostics (see the `spi_toolslib` User Manual for more informations on how the spectral information is stored in the `OBS` data structure). $4 \times 19 \times n$ spectra are generated in this case (with n being the number of time slices) with the SPI pointing identifiers `PTID_SPI` `SE_`*xxx* (single detector event spectra for time slice *xxx*, `PE_`*xxx* (PSE event spectra

for time slice *xxx*, `EFF_xxx` (efficiency spectra multiplied by 1000 for time slice *xxx*, and `EFF_NE_xxx` (efficiency spectra derived without PSD errors, multiplied by 1000 for time slice *xxx*. As default, specify `no`.

- `saveDOL` (optional) specifies the **filename** of the `OBS` data structure if `saveSpec = yes`.
- `reportfit` specifies if detailed fit results should be reported into the log file. As default, `reportfit = no`, yet in case of trouble with fitting the PSD efficiency and threshold function one may switch to `yes` for diagnostics.
- `limcheck` specifies if alert limit checking should be performed by `spi_psd_efficiency`. If set to `yes`, `spi_psd_efficiency` compares the thresholds and efficiencies to the limits that are specified in the alert limit file (see `alertDOL`) and (optionally) creates ISDC alerts (see parameters `alert0` to `alert3`). Alert limit checking will be only performed for time slices that show sufficient event statistics. The event statistics limit is defined by the parameter `minPE`.
- `alert0` (optional) if alert limit checking is enabled (`limcheck = yes`), generates level 0 ISDC alerts.
- `alert1` (optional) if alert limit checking is enabled (`limcheck = yes`), generates level 1 ISDC alerts.
- `alert2` (optional) if alert limit checking is enabled (`limcheck = yes`), generates level 2 ISDC alerts.
- `alert3` (optional) if alert limit checking is enabled (`limcheck = yes`), generates level 3 ISDC alerts.
- `minPE` (optional) if alert limit checking is enabled (`limcheck = yes`), specifies the minimum required number of PSD event (PE) per detector to initiate alert limit checking. This parameter avoids alert limit checking in case of insufficient event statistics. To achieve reasonably good PSD thresholds and efficiencies, a minimum number of $\sim$ 10000 PSD events (**TBC**) per detector should be requested.
- `clobber` ISDC standard parameter (not used so far).
- `mode` ISDC standard parameter (not used so far).

4 Interface definition

`spi_psd_efficiency` derives PSD efficiencies and thresholds from a stream of photon data. This photon data may be either grouped in a science window group, an observation group, or an index file. `spi_psd_efficiency` needs at least an ISDC level of `PRP`. In this case, a calibration file (`SPI.-COEF-CAL`) or calibration index file (`SPI.-COEF-CAL-IDX`) has to be specified, and energy calibration is performed *on-the-fly*. However, if the input group is of level `COR`, the calibrated energy information from the corresponding data structures is taken, irrespectively of the presence of a calibration DOL.

If onground analysis errors should be considered correctly for efficiency and threshold calculation (`onground = yes`), the `PSD_CORFLAG` column of PSD events (`PE`) has also to be filled, and an ISDC level of `COR` is now mandatory. In this case, make also sure that `spi_psd_postprocess` has been executed before calling `spi_psd_efficiency` to fill the `PSD_CORFLAG` column with meaningful values.

`spi_psd_efficiency` scans the input group for all single event (`SE`) and PSD event (`PE`) data it can find and builds efficiency spectra `PE / (SE + PE)`. An OBT interval may be specified that selects a particular sub-interval for which PSD efficiencies and thresholds should be determined. In addition, one may slice the specified OBT interval in time intervals of constant `ONTIME`. Optionally, time slices that are shorter than 95% of the requested `ONTIME` may be skipped, thus assuring an uniform quality (in terms of data statistics) of the efficiency results.

For each time-interval, `spi_psd_efficiency` adds one row to the `SPI.-EFFI-PSD` result data structure. `spi_psd_efficiency` fills all columns of this data structure, hence it can be considered as complete after `spi_psd_efficiency` has finished. The `OBTFIRST` and `OBTLAST` keywords are also updated, so that

index group generating tools may be used to assess the validity interval of the data structure. In particular, `spi_psd_efficiency` may access the `OBTLAST` keyword set by a previous run if continuous time slices should be added to the output file (parameter `append = yes`). The following columns are filled by `spi_psd_efficiency`:

- `OBT_START` : OBT start of the time interval of the actual row.
- `OBT_STOP` : OBT stop (or end) of the time interval of the actual row.
- `ONTIME` : ontime for this row in seconds.
- `PSD_LW_THRES` : lower energy threshold values in keV for all 19 PSD channels. A value of `0` indicates that the corresponding fit failed.
- `PSD_LW_WIDTH` : lower energy threshold widths in keV for all 19 PSD channels. A value of `0` indicates that the corresponding fit failed.
- `PSD_UP_THRES` : upper energy threshold values in keV for all 19 PSD channels. A value of `0` indicates that the corresponding fit failed.
- `PSD_UP_WIDTH` : upper energy threshold widths in keV for all 19 PSD channels. A value of `0` indicates that the corresponding fit failed.
- `PSD_EFF` : PSD efficiency for all detected PSD events (including pulse shape analysis errors) for all 19 PSD channels. A value of `0` may either indicate that no PSD events were detected during the actual time interval, or that the fit failed. To resolve this ambiguity, the fit failed if also all threshold values and widths are zero. Also, the column `PE_NUM` indicates the number of PSD events that have been found for the actual time interval.
- `PSD_EFF_NOERR` : PSD efficiency for all detected PSD events (excluding pulse shape analysis errors) for all 19 PSD channels. A value of `0` may either indicate that no PSD events were detected during the actual time interval, or that the fit failed. To resolve this ambiguity, the fit failed if also all threshold values and widths are zero. Also, the column `PE_NUM` indicates the number of PSD events that have been found for the actual time interval.
- `SE_NUM` : number of single detector events (`SE`) that have been collected for efficiency and threshold determination.
- `PE_NUM` : number of PSD events (`PE`; including pulse shape analysis errors) that have been collected for efficiency and threshold determination.
- `PE_NUM_NOERR` : number of PSD events without pulse shape analysis errors that have been collected for efficiency and threshold determination.
- `PE_NUM_SGLE` : number of PSD events without pulse shape analysis errors that have a single-site flag (i.e. background events).
- `PE_NUM_MULT` : number of PSD events without pulse shape analysis errors that have a multiple-site flag (i.e. photon events).

Optionally, `spi_psd_efficiency` is able to generate ISDC alerts if some of the efficiency parameters fall out of the defined limits. The alert limits are defined by an `SPI.-ALRT-LIM` structure from which the following columns are used by `spi_psd_efficiency` for alert generation:

- `PAR_NAME` specifies the parameter for which the limits apply. Valid parameter names are `PSD_LW_THRES`, `PSD_LW_WIDTH`, `PSD_UP_THRES`, `PSD_UP_WIDTH`, `PSD_EFF`, and `PSD_EFF_NOERR`. If one of those parameter names is defined, the specified limits apply to **all** 19 PSD channels. By adding `_L`*n* to the parameter

name (where n runs from 0 to 18), limits may be specified for a given PSD channel (for example `PSD_LW_THRES_L3` specifies the limits for the lower PSD threshold of channel 3). Channel specific parameters have precedence over common parameters (i.e. those without the `_Ln` extension), hence one may define common limits for all 19 PSD channels and overwrite a few limits for specific channels by specifying explicitely the channel number.

- `MIN_VAL` specifies the lower limits (inclusive) for the four ISDC alert levels (DAL table columns 1-3 corresponds to alert levels 0-3).
- `MAX_VAL` specifies the upper limits (inclusive) for the four ISDC alert levels (DAL table columns 1-3 corresponds to alert levels 0-3).
- `SUB_ASSEMBLY` specifies the SPI PSD sub-assembly and must contain the entry `SPI_PSD`.

All other columns (`OBT_START`, `OBT_END`, `CHECK_MODE`, `ALERT_DELAY`) of the `SPI.-ALRT-LIM` structure are ignored. The validity of the alert limit file is defined by the two keywords `VSTART` and `VSTOP`. If an alert limit index is used, these keywords are used to selected the alert limit file that is appropriate for the PSD efficiency validity time interval. If the PSD efficiency validity time interval stops before the validity of the earliest alert limits, the earliest alert limits are used by `spi_psd_efficiency` (a warning is issued by `spi_psd_efficiency` in this case). If the validity time interval starts after the validity of the last alert limits, the last alert limits are used by `spi_psd_efficiency` (a warning is issued by `spi_psd_efficiency` in this case). If the PSD efficiency validity time interval overlaps with the transition of two (or more) alert limits files, those alert limits are applied that have the longest time overlap with the PSD efficiency validity time interval.

5 Algorithm

The PSD efficiency and threshold is determined by fitting the function

$$f(E) = \begin{cases} p_0 \exp\left(-\frac{1}{2}\frac{(E-p_1)^2}{p_2^2}\right) & : \quad E < p_1 \\ p_0 & : \quad p_1 \le E \le p_3 \\ p_0 \exp\left(-\frac{1}{2}\frac{(E-p_3)^2}{p_4^2}\right) & : \quad E > p_3 \end{cases} \tag{1}$$

to the spectrum that is defined by

```
EFF = PE / (SE + PE)
```

where `PE` is the number of PSD events in a spectral bin, `SE` is the number of single detector events in the same spectral bin, and `EFF` is the resulting **efficiency spectrum** value for this bin. p_0 - p_4 are the five parameters of the fit with the meaning

- p_0: PSD efficiency `PSD_EFF` or `PSD_EFF_NOERR` (depending on how the `PE` were selected).
- p_1: lower energy threshold value `PSD_LW_THRES`
- p_2: lower energy threshold width `PSD_LW_WIDTH`
- p_3: upper energy threshold value `PSD_UP_THRES`
- p_4: upper energy threshold width `PSD_UP_WIDTH`

Note, that PSD_LW_THRES and PSD_UP_THRES are not really threshold values, but merely the values between which the PSD efficiency stays constant. To convert these values into 50% thresholds, use

$$t_{\mathrm{low}}(0.5) = \texttt{PSD\_LW\_THRES} - \sqrt{2\ln 2} \times \texttt{PSD\_LW\_WIDTH} \tag{2}$$

and

$$t_{\mathrm{up}}(0.5) = \texttt{PSD\_UP\_THRES} + \sqrt{2\ln 2} \times \texttt{PSD\_UP\_WIDTH} \tag{3}$$

for the lower and upper threshold, respectively.

Optional alert limit checks are performed with higher alert levels preceeding lower alert levels, i.e. an alert of the highest possible level is generated. Minimum parameter limits are checked before maximum parameter limits, and minimum limits have precedence over maximum limits (this is not really of relevance since a parameter can not both violate the minimum and maximum limit **unless the alert limit file has not been set up correctly**, i.e. the minimum limit is always smaller or equal to the maximum limit). Alert limits are inclusive, i.e. a minimum limit violation alert is generated if

$$\texttt{PARAMETER} < \texttt{MIN\_VAL} \tag{4}$$

is fulfilled, and a maximum limit violation alert is generated if

$$\texttt{PARAMETER} > \texttt{MAX\_VAL} \tag{5}$$

is fulfilled.

Limit violation alerts are only generated if

$$\texttt{PE\_NUM} \geq \texttt{minPE} \tag{6}$$

is fulfilled, where PE_NUM is the number of PSD events for one detector in a time slice.

6 Alerts

spi_psd_efficiency may optionally generate alerts that signal possible PSD/SPI misfunctions. The following list provides the alert parameters and the actions that should be taken in case of occurence of the alerts. The alert parameter in the alert message is followed by the extension _L*n* where *n=0-18* specifies the PSD detection channel for which the alert occured.

If the action **standard** is specified in the table, the standard alert action should be performed (**TBD**).

Parameter	Level	Action
PSD_LW_THRES	0-3	standard
PSD_LW_WIDTH	0-3	standard
PSD_UP_THRES	0-3	standard
PSD_UP_WIDTH	0-3	standard
PSD_EFF	0-3	standard
PSD_EFF_NOERR	0-3	standard

7 Error codes

The following error codes are defined:

```
SPI_PSD_EFFICIENCY_ERROR_MEM_ALLOC              -230100
SPI_PSD_EFFICIENCY_ERROR_INVALID_MODE           -230101
```

```
SPI_PSD_EFFICIENCY_ERROR_NEG_SE                 -230102
SPI_PSD_EFFICIENCY_ERROR_NEG_PE                 -230103
SPI_PSD_EFFICIENCY_ERROR_NEG_PE_NOERR           -230104
SPI_PSD_EFFICIENCY_ERROR_NEG_PE_SGLE            -230105
SPI_PSD_EFFICIENCY_ERROR_NEG_PE_MULT            -230106
SPI_PSD_EFFICIENCY_ERROR_FIT                    -230107
SPI_PSD_EFFICIENCY_ERROR_INDEX_SELECT           -230108
```

They have the following meaning:

- `SPI_PSD_EFFICIENCY_ERROR_MEM_ALLOC` : a memory allocation error occured while allocating memory dynamically. Probably, your system resources are not sufficient to execute this task.
- `SPI_PSD_EFFICIENCY_ERROR_INVALID_MODE` : an invalid parameter has been specified for **`spiMode`**. Only `0` or `3` are allowed values. Correct the task parameter file and restart task.
- `SPI_PSD_EFFICIENCY_ERROR_NEG_SE` : the single event spectrum has a negative content. This error should never occur. It may point towards a **`spi_toolslib`** error.
- `SPI_PSD_EFFICIENCY_ERROR_NEG_PE` : the PSD event spectrum has a negative content. This error should never occur. It may point towards a **`spi_toolslib`** error.
- `SPI_PSD_EFFICIENCY_ERROR_NEG_PE_NOERR` : the PSD event without analysis errors spectrum has a negative content. This error should never occur. It may point towards a **`spi_toolslib`** error.
- `SPI_PSD_EFFICIENCY_ERROR_NEG_PE_SGLE` : the PSD single-site event spectrum has a negative content. This error should never occur. It may point towards a **`spi_toolslib`** error.
- `SPI_PSD_EFFICIENCY_ERROR_NEG_PE_MULT` : the PSD multiple-site event spectrum has a negative content. This error should never occur. It may point towards a **`spi_toolslib`** error.
- `SPI_PSD_EFFICIENCY_ERROR_FIT` : the task stopped since an efficiency fitting error occured and hence the results may be invalid. To avoid task interruption, specify **`ignore = yes`**.
- `SPI_PSD_EFFICIENCY_ERROR_INDEX_SELECT` : while searching a single member in an index group, DAL3GEN returned more than one member. This should never happen. If this error occurs, it is likely that the alert limit index you specified on input is somehow corrupted.