

Help on module operational_sep_quantities:

NAME

operational_sep_quantities

AUTHOR

Katie Whitman

FILE

operational-sep/operational_sep_quantities.py

DATA

```
__email__ = 'kathryn.whitman@nasa.gov'  
__maintainer__ = 'Katie Whitman'
```

VERSION

V1.1

FUNCTIONS

all_program_info()

: This program will calculate various useful pieces of operational information about SEP events from GOES-08, -10, -11, -12, -13, -14, -15 data and the SEP-EM (RSDv2) dataset.

SEP event values are always calculated for threshold definitions:

- >10 MeV exceeds 10 pfu
- >100 MeV exceed 1 pfu

The user may add multiple additional thresholds through the command line. This program will check if data is already present in a 'data' directory. If not, GOES or EPHIN data will be automatically downloaded from the web. SEP-EM (RSDv2) data must be downloaded by the user and unzipped inside the 'data' directory. Because the SEP-EM data set is so large (every 5 minutes from 1974 to 2015), the program will break up the data into yearly files for faster reading.

The values calculated here are important for space radiation operations:

- Onset time, i.e. time to cross thresholds
- Onset peak intensity
- Onset peak time
- Maximum intensity
- Time of maximum intensity
- Rise time (onset to peak)
- End time, i.e. fall below $0.85 \times \text{threshold}$ for 3 points (15 mins for GOES)

Duration

Event-integrated fluences

Proton fluxes at various times after threshold crossing (UMASEP option)

User may choose differential proton fluxes (e.g. $[\text{MeV s sr cm}^2]^{-1}$) or integral fluxes (e.g. $[\text{s sr cm}^2]^{-1}$ or pfu). The program has no internal checks or requirements on units - EXCEPT FOR THE THRESHOLD DEFINITIONS OF >10, 10 and >100, 1. If you convert those thresholds in the main program to your units, you should be able to generate consistent results.

Also, all of the plots and messages refer to MeV, pfu, and cm. Change those labels everywhere if you choose other units. Currently no features to change units automatically.

If a previous event is ongoing and the specified time period starts with a threshold already crossed, you may try to set the --DetectPreviousEvent flag. If the flux drops below threshold before the next event starts, the program will identify the second event. This will only work if the threshold is already crossed for the very first time in your specified time period, and if the flux drops below threshold before the next event starts.

If the event has an initial increase above threshold for a few points, falls below threshold, then continues to increase above threshold again, you may try to use the --TwoPeak feature to capture the full duration of the event. The initial increase above threshold must be less than a day. An example of this scenario can be seen in >100 MeV for 2011-08-04.

A flag was added with the "UMASEP" option. When this flag is used (--UMASEP), the code finds all information for four energy channels and thresholds used by UMASEP: >10 MeV, 10 pfu; >100 MeV, 1 pfu; >30 MeV, 1 pfu; >50 MeV, 1 pfu. The proton flux in each of these channels is reported for multiple times after threshold crossing (Ts). The applied time delays are as follows:

- >10 MeV - Ts + 3, 4, 5, 6, 7 hours
- >30 MeV - Ts + 3, 4, 5, 6, 7 hours
- >50 MeV - Ts + 3, 4, 5, 6, 7 hours
- >100 MeV - Ts + 3, 4, 5, 6, 7 hours

RUN CODE FROM COMMAND LINE (put on one line), e.g.:

```
python3 operational_sep_quantities.py --StartDate 2012-05-17
--EndDate '2012-05-19 12:00:00' --Experiment GOES-13
--FluxType integral --showplot --saveplot
```

RUN CODE FROM COMMAND FOR USER DATA SET (put on one line), e.g.:

```
python3 operational_sep_quantities.py --StartDate 2012-05-17
--EndDate '2012-05-19 12:00:00' --Experiment user --ModelName MyModel
--UserFile MyFluxes.txt --FluxType integral --showplot
```

RUN CODE IMPORTED INTO ANOTHER PYTHON PROGRAM, e.g.:

```
import operational_sep_quantities as sep
start_date = '2012-05-17'
end_date = '2012-05-19 12:00:00'
experiment = 'GOES-13'
flux_type = 'integral'
model_name = " #if experiment is user, set model_name to describe data set
user_file = " #if experiment is user, specify filename containing fluxes
showplot = True #Turn to False if don't want to see plots
saveplot = False #turn to true if you want to save plots to file
detect_prev_event = True #Helps if previous event causes high intensities
two_peaks = False #Helps if two increases above threshold in one event
umasep = False #Set to true if want UMASEP values (see explanation above)
threshold = '100,1' #default; modify to add a threshold to 10,10 and 100,1
FirstStart, LastEnd, ShortEvent, LateHundred, sep_year, sep_month, sep_day =
sep.run_all(start_date, end_date, experiment, flux_type, model_name, user_file, showplot,
saveplot, detect_prev_event, two_peaks, umasep, threshold)
```

Set the desired directory locations for the data and output at the beginning of the program in datapath and outpath. Defaults are 'data' and 'output'.

In order to calculate the fluence, the program determines time_resolution (seconds) from two (fairly random) data points at the start of the SEP event. GOES and SEPTEM data sets have a time resolution of 5 minutes. If the user wishes to use a data set with measurements at irregular times, then the subroutine calculate_fluence should be modified.

OUTPUT: This program outputs 3 to 4 files, 1 per defined threshold plus a summary file containing all of the values calculated for each threshold. A file named as e.g. fluence_GOES-13_differential_gt10_2012_3_7.csv contains the event-integrated fluence for each energy channel using the specified threshold (gt10) to determine start and stop times. A file named as e.g. sep_values_GOES-13_differential_2012_3_7.csv contains start time, peak flux, etc, for each of the defined thresholds.

Added functionality to output the >10 MeV and >100 MeV time series for the date range input by the user. If the original data were integral fluxes, then the output files simply contain the >10 and >100 MeV time series from the input files. If the original data were differential fluxes, then the estimated >10 and >100 MeV fluxes are output as time series.

USER INPUT DATA SETS: Users may input their own data set. For example, if an SEP modeler would like to feed their own intensity time series into this code and calculate all values in exactly the same way they were calculated for data, it is possible to do that. Fluxes should be in units of $1/[\text{MeV cm}^2 \text{ s sr}]$ or $1/[\text{cm}^2 \text{ s sr}]$ and energy channels in MeV for the plot labels to be correct. You can use any units, as long as you are consistent with energy units in energy channel/bin definition and in fluxes and you **MODIFY THE THRESHOLD VALUES TO REFLECT YOUR UNITS.** You may then want to modify plot labels accordingly if not using MeV and cm.

NOTE: The first column in your flux file is assumed to be time in format YYYY-MM-DD HH:MM:SS. IMPORTANT FORMATTING!!

NOTE: The flux file may contain header lines that start with a hash #, including blank lines.

NOTE: Any bad or missing fluxes must be indicated by a negative value.

NOTE: Put your flux file into the "datapath" directory. Filenames will be relative to this path.

NOTE: Please use only differential or integral channels. Please do not mix them. You may have one integral channel in the last bin, as this is the way HEPAD works and the code has been written to include that HEPAD >700 MeV bin along with lower differential channels.

USER VARIABLES: The user must modify the following variables at the very top of the code (around line 30):

user_col - identify columns in your file containing fluxes to analyze; even if your delimiter is white space, consider the date-time column as one single column. SET AT TOP OF CODE.

user_delim - delimiter between columns, e.g. " " or "," Use " " for any amount of whitespace. SET AT TOP OF CODE.

user_energy_bins - define your energy bins at the top of the code in the variable user_energy_bins. Follow the format in the subroutine define_energy_bins. SET AT TOP OF CODE.

user_fname - specify the name of the file containing the fluxes through an argument in the command line. --UserFile The user_fname variable will be updated with that filename. ARGUMENT

time_resolution - will be calculated using two time points in your file; if you have irregular time measurements, calculate_fluence() must be modified/rewritten. **AUTOMATICALLY DETERMINED.**

calculate_event_info(energy_thresholds, flux_thresholds, dates, integral_fluxes, detect_prev_event, two_peaks, is_diff_thresh)

: Uses the integral fluxes (either input or estimated from differential channels) and all the energy and flux thresholds set in the main program to calculate SEP event quantities.

Threshold crossing time (onset)

Peak Flux in date range specified by user

Time of Peak Flux

Rise Time (onset to peak flux)

Event End Time (below $0.85 \times \text{threshold}$ for 3 data points, e.g 15 min)

Duration (onset to end)

If the `detect_prev_event` flag is set to true and the threshold is crossed on the first time in the specified date range, this indicates that fluxes were already high due to a previous event. The code will look for the flux to drop below threshold and then increase above threshold again during the specified time period. If this flag is not set, then the code simply take the first threshold crossing as the start of the SEP event.

If the `two_peaks` flag is set to true, the code will use the first identified threshold crossing as the start time. A second threshold crossing will be allowed. The event end will be determined as the drop below threshold after the second threshold crossing. The duration will be the end time - first threshold crossing time. The peak will be determined as the largest flux value found in both threshold crossings. The peak and rise times will be calculated from the first threshold crossing time.

calculate_fluence(dates, flux)

: This subroutine sums up all of the flux in the 1D array "flux". The "dates" and "flux" arrays input here should reflect only the intensities between the SEP start and stop times, determined by the subroutine `integral_threshold_crossing`. "flux" should contain the intensity time series for a single energy bin or single integral channel (1D array). The subroutine does not differentiate between differential or integral flux. dates contains the 1D array of datetimes that correspond to the flux measurements.

The `extract_date_range` subroutine is used prior to calling this one to make the dates and fluxes arrays covering only the SEP time period. The flux will be multiplied by `time_resolution` and summed for all of the data between the start and end times. Data gaps are taken into account. Fluence units will be $1/[\text{MeV cm}^2 \text{ sr}]$ for GOES or SEP-EM differential fluxes or $1/[\text{cm}^2 \text{ sr}]$ for integral fluxes.

For SRAG operational purposes, the event start and end is determined by the $>100 \text{ MeV}$ fluxes, as they are most pertinent to astronaut health inside of the space station.

calculate_onset_peak(experiment, energy_thresholds, dates, integral_fluxes, crossing_time, event_end_time, showplot)

- : Calculate the peak associated with the initial SEP onset. This subroutine searches for the rollover that typically occurs after the SEP onset. The peak value will be specified as the flux value at the rollover location.
- The onset peak may provide a more physically appropriate comparison with models.
- If code cannot identify onset peak, it will return a value of -1 on the date 1970-01-01.

calculate_umasep_info(energy_thresholds, flux_thresholds, dates, integral_fluxes, crossing_time)

- : Uses the integral fluxes (either input or estimated from differential channels) and all the energy and flux thresholds set in the main program to calculate SEP event quantities specific to the UMASEP model.
- Flux at threshold crossing time + 3, 4, 5, 6, 7 hours

check_bin_exists(threshold, energy_bins)

- : If a user specifies a threshold for a differential energy bin, check to see that the energy bin is in the requested data set.
- threshold is a list of strings.
- Expect threshold[0] = lowedge-highedge, threshold[1] = flux value

check_data(startdate, enddate, experiment, flux_type)

- : Check that the files containing the data are in the data directory. If the files for the requested dates aren't present, they will be downloaded from the NOAA website. For SEPv2 (RSDv2) data, if missing, the program prints the URL from which the data can be downloaded and unzipped manually.
- The RSDv2 data set is very large and takes a long time to read as a single file. This program will generate files containing fluxes for each year for faster reading.

check_ephin_data(startdate, enddate, experiment, flux_type)

- : Check for SOHO/COSTEP/EPHIN data on your computer. If not there, download from <http://ulysses.physik.uni-kiel.de/costep/level3/l3i/> 30 minute data will be downloaded. Intensities are in units of $(\text{cm}^2 \text{ s sr mev/nuc})^{-1}$
- First available date is 1995 12 8 (DOY = 342).

The files are available in daily or yearly format.

check_for_bad_data(dates, fluxes, energy_bins)

: Search the data for bad values (flux < 0) and fill the missing data with an estimate flux found by performing a linear interpolation with time, using the good flux values immediately surrounding the data gap.

check_goes_data(startdate, enddate, experiment, flux_type)

: Check that GOES data is on your computer or download it from the NOAA website. Return the filenames associated with the correct GOES data.

check_paths()

: Check that the paths that hold the data and output exist. If not, create.

check_sepem_data(startdate, enddate, experiment, flux_type)

: Check if SEPEM data is present on the computer. Break into yearly files if needed. Return SEPEM filenames for analysis.

define_energy_bins(experiment, flux_type)

: Define the energy bins for the selected spacecraft or data set.
If the user inputs their own file, they must set the user_energy_bins variable at the top of the code.

do_interpolation(i, dates, flux)

: If bad fluxes (flux < 0) are found in the data, find the first prior data point and the first following data point that have good flux values. Perform linear interpolation in time:
$$F(t) = F1 + (t - t1) * (F2 - F1) / (t2 - t1)$$

This subroutine does the calculation for a single instance of bad data that corresponds to array index i.

extract_date_range(startdate, enddate, all_dates, all_fluxes)

: Extract fluxes only for the dates in the range specified by the user.

extract_integral_fluxes(fluxes, experiment, flux_type, flux_thresholds, energy_thresholds, energy_bins)

: Select or create the integral fluxes that correspond to the desired energy thresholds.

If the user selected differential fluxes, then the differential fluxes will be converted to integral fluxes with the minimum energy defined by the set energy thresholds.

If the user selected integral fluxes, then the channels corresponding to the desired energy thresholds will be identified.

find_goes_data_dimensions(filename)

: Input open csv file of GOES data. Identifies the start of the data by searching for the string 'data:', then returns the number of header rows and data rows present in the file.

from_differential_to_integral_flux(experiment, min_energy, energy_bins, fluxes)

: If user selected differential fluxes, convert to integral fluxes to calculate operational threshold crossings (>10 MeV protons exceed 10 pfu, >100 MeV protons exceed 1 pfu).

Assume that the measured fluxes correspond to the center of the energy bin and use power law interpolation to extrapolate integral fluxes above user input min_energy.

The intent is to calculate >10 MeV and >100 MeV fluxes, but leaving flexibility for user to define the minimum energy for the integral flux.

An integral flux will be provided for each timestamp (e.g. every 5 mins).

get_fluence_spectrum(experiment, flux_type, model_name, energy_threshold, flux_threshold, sep_dates, sep_fluxes, energy_bins, is_diff_thresh, save_file)

: Calculate the fluence spectrum for each of the energy channels in the user selected data set. If the user selected differential fluxes, then the fluence values correspond to each energy bin. If the user selected integral fluxes, then the fluence values correspond to each integral bin.

Writes fluence values to file according to boolean save_file.

If the user input a threshold for a differential energy bin, is_diff_thresh will be true and the filename will not contain "gt".

get_west_detector(filename, dates)

: For GOES-13+, identify which detector is facing west from the orientation flag files. Get an orientation for each data point.

EPEAD orientation flag. 0: A/W faces East and B/E faces West.

1: A/W faces West and B/E faces East. 2: yaw-flip in progress.

integral_threshold_crossing(energy_threshold, flux_threshold, dates, fluxes)

: Calculate the time that a threshold is crossed.

Operational thresholds used by the NASA JSC Space Radiation Analysis Group to determine actions that should be taken during an SEP event are:

>10 MeV proton flux exceeds 10 pfu ($1/[\text{cm}^2 \text{ s sr}]$)

>100 MeV proton flux exceeds 1 pfu ($1/[\text{cm}^2 \text{ s sr}]$)

An SEP event is considered to start if 3 consecutive points are above threshold. The start time is set to the first point that crossed threshold.

If the user input differential flux, then the program has converted it to integral fluxes to calculate the integral threshold crossings.

If the user selected fluxes that were already integral fluxes, then the correct channel was identified to determine these crossings.

GOES and SEP-EM data have a 5 minute time resolution, so initial crossings are accurate to 5 minutes.

This program also calculates peak flux and time of peak flux for time period specified by user. It then calculates rise time by comparing:

rise time = time of peak flux - threshold crossing time

If no thresholds are crossed during specified time period, peak time and rise time will be 0.

The event end time is much more subjective. For this program, I follow the code that officially calls the end of an event when SRAG supports ISS operations. When the flux has three consecutive points (15 minutes of GOES data) below $0.85 \times \text{threshold}$, then the event is ended. The end time is corrected back to the first of the three data points. This end criteria has no physics basis. It simply ensures that a new event will not erroneously begin within the SRAG SEP alarm code due to flux fluctuations around threshold as the SEP flux decays away slowly.

If the time period specified by the user ends before the event falls below $0.85 \times \text{threshold}$, then the event_end_time is set to the last time in the specified time range. The program will give a message indicating that the user may want to extend the requested time frame in order to better-capture the end of the event.

make_yearly_files(filename)

: Convert a large data set into yearly files.

print_values_to_file(experiment, flux_type, model_name, energy_thresholds, flux_thresholds, crossing_time, onset_peak, onset_date, peak_flux, peak_time, rise_time, event_end_time, duration, integral_fluxes, is_diff_thresh, umasep, umasep_times, umasep_fluxes)

: Write all calculated values to file for all thresholds. Event-integrated fluences for >10, >100 MeV (and user-defined threshold) will also be included. Writes out file with name e.g.
output/sep_values_experiment_fluxtype_YYYY_M_D.csv
If the UMASEP option was selected, add on the proton values calculated at the UMASEP Ts + Xhr time points.
is_diff_thresh indicates whether the user input a differential threshold. If so, the user threshold bin(s) will refer to differential fluxes.

read_in_ephin(experiment, flux_type, filenames1)

: Read in EPHIN files from your computer.

read_in_files(experiment, flux_type, filenames1, filenames2, filenames_orien)

: Read in the appropriate data files with the correct format. Return an array with dates and fluxes. Bad flux values (any negative flux) are set to -1. Format is defined to work with the files downloaded directly from NOAA or the RSDv2 (SEP-EM) website as is.

The fluxes output for the GOES-13+ satellites are always from the westward-facing detector (A or B) by referring to the orientation flags provided in the associated orientation file. Data taken during a yaw flip (orientation flag = 2) are excluded and fluxes are set to -1.

Note that the EPS detectors on GOES-08 and -12 face westward. The EPS detector on GOES-10 faces eastward. GOES-11 is a spinning satellite.

read_in_goes(experiment, flux_type, filenames1, filenames2, filenames_orien)

: Read in GOES data from your computer.

read_in_sepem(experiment, flux_type, filenames1)

: Read in SEP-EM data files from the computer.

read_in_user_files(filenames1)

: Read in file containing flux time profile information that was specified by the user.

The first column MUST contain the date in YYYY-MM-DD HH:MM:SS format. The remaining flux columns to be read in are specified by the user in the variable user_col at the very beginning of this program.

The date column should always be considered column 0, even if you used whitespace as your delimiter. The code will consider the date format

YYYY-MM-DD HH:MM:SS as one column even though it contains whitespace.
Any number of header lines are allowed, but they must be indicated by #
at the very beginning, including empty lines.
Be sure to add the energy bins associated with your flux columns in the
subroutine `define_energy_bins` under the "user" is statement.

**`report_threshold_fluxes(experiment, flux_type, model_name, energy_thresholds,
energy_bins, sep_dates, sep_fluxes)`**

- : Report fluences for specified thresholds, typically >10, >100 MeV.
These values are interesting to use for comparison with literature and
for quantifying event severity.
If the user has input integral channels, then this information has also
been saved in the output fluence file. If the user selected differential
channels, then these values come from the estimated integral fluxes.

**`run_all(str_startdate, str_enddate, experiment, flux_type, model_name, user_file, showplot,
saveplot, detect_prev_event, two_peaks, umasep, str_thresh)`**

- : Runs all subroutines and gets all needed values. Takes the command line
arguments as input. Written here to allow code to be imported into
other python scripts.
`str_startdate`, `str_enddate`, `experiment`, `flux_type` are strings.
`model_name` is a string. If model is "user", set `model_name` to describe
your model (e.g. MyModel), otherwise set to "".
`user_file` is a string. Default is ". If user is selected for experiment,
then name of flux file is specified in `user_file`.
`showplot`, `detect_prev_event`, `two_peaks`, and `umasep` are booleans.
Set `str_thresh` to be '100,1' for default value or modify to add your own
threshold.
This routine will generate boolean flags that indicate if the event
starts at the very first time point, ends on the very last time point,
has a duration less than 12 hours, or has a >100 MeV onset more than
24 hours after the >10 MeV onset. These flags intend to help the user
running the program in batch mode if a certain event might have
incorrect timing.

**`save_integral_fluxes_to_file(experiment, flux_type, model_name, energy_thresholds,
crossing_time, dates, integral_fluxes)`**

- : Output the time series of integral fluxes to a file. If the input
data set was in integral channels, then this file will contain exactly
the same values in the time series.
If the input data set was in differential energy bins, then this file

contains the estimated integral fluxes calculated in this program.