Brief tutorial for ERG part_products

T. Hori, Y. Miyoshi, C.-W. Jun, S. Nakamura, M. Kitahara
(ERG-SC, ISEE)

Goal of this tutorial

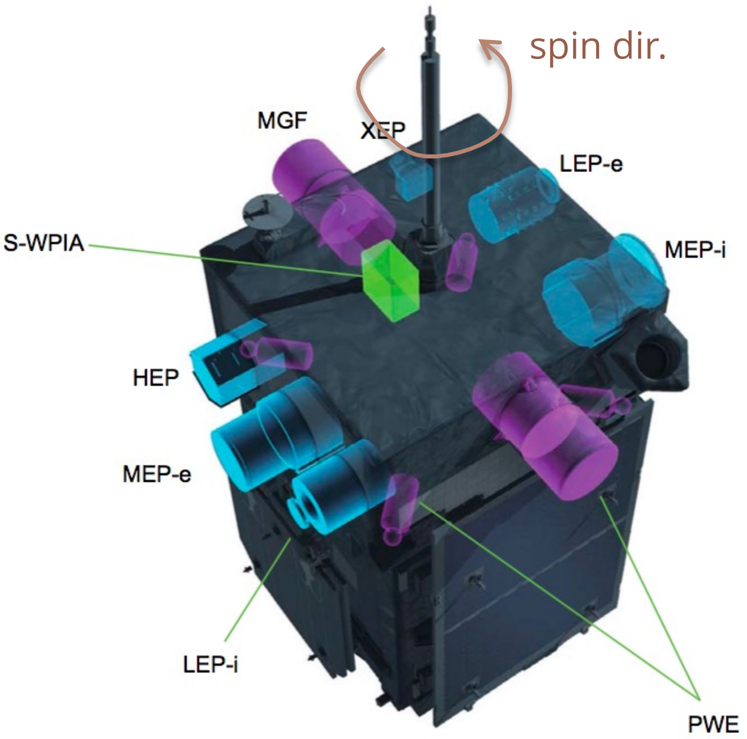
- To get familiar with how to load, plot, and manipulate the particle data of the ERG satellite.

  - Load and plot particle flux data of LEP-e, LEP-i, MEP-e, MEP-i, HEP, and XEP.
  - Use the "part_products" library to make a plot of:
    - Energy-time spectrogram
    - Phi-/Theta-angle spectrogram
    - Pitch angle spectrogram
    - Gyro-phase spectrogram
    - Velocity moments

Brief introduction of Arase's particle data

About particle data obtained by Arase

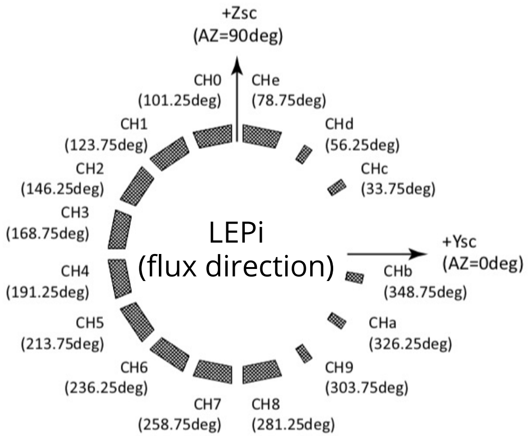
- LEP-e (PI: S.-Y. Wang, ASIAA)
  - https://earth-planets-space.springeropen.com/articles/10.1186/s40623-017-0748-6
  - Lv.2 3-D electron flux data (DOI: 10.34515/DATA.ERG-04001)
- LEP-i (PI: K. Asamura, JAXA/ISAS)
  - https://earth-planets-space.springeropen.com/articles/10.1186/s40623-018-0846-0
  - Lv.2 3-D ion flux data (DOI: 10.34515/DATA.ERG-05000)
- MEP-e (PI: S. Kasahara, Univ. of Tokyo)
  - https://earth-planets-space.springeropen.com/articles/10.1186/s40623-018-0847-z
  - Lv.2 3-D electron flux data (DOI: 10.34515/DATA.ERG-02000)
- MEP-i (PI: S. Yokota, Osaka Univ.)
  - https://earth-planets-space.springeropen.com/articles/10.1186/s40623-017-0754-8
  - Lv.2 3-D ion flux data (DOI: 10.34515/DATA.ERG-03000)
- HEP (PI: T. Mitani, JAXA/ISAS)
  - https://earth-planets-space.springeropen.com/articles/10.1186/s40623-018-0853-1
  - Lv.2 3-D electron flux data (DOI: 10.34515/DATA.ERG-01000)
- XEP (PI: N. Higashio, JAXA)
  - https://earth-planets-space.springeropen.com/articles/10.1186/s40623-018-0901-x
  - Lv.2 2-D electron flux data (DOI: 10.34515/DATA.ERG-00000)

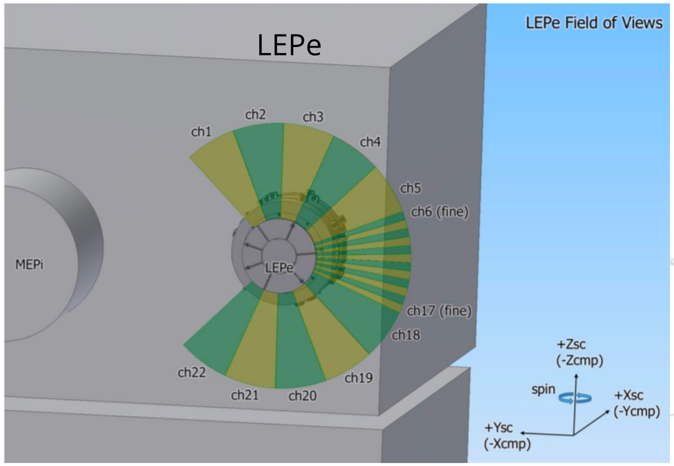


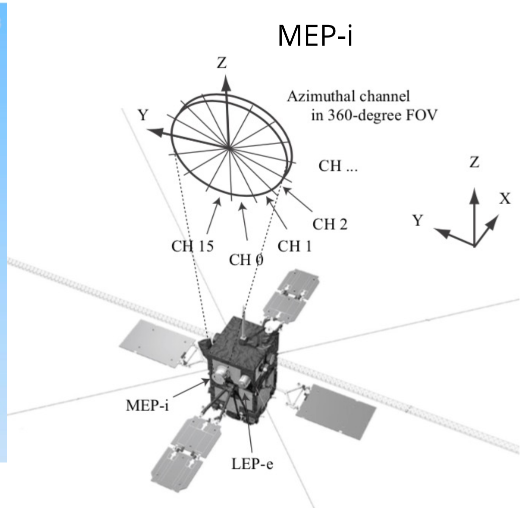
[Miyoshi+2018]

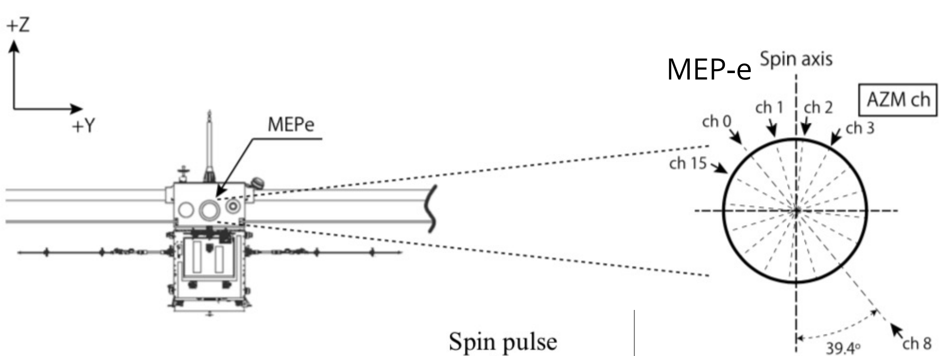
Quick reference for the field-of-views of the particle instruments

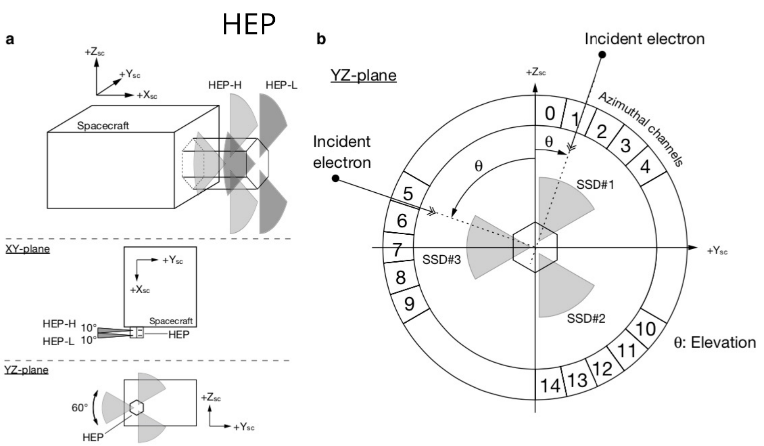
Please see the instrument papers in the prev. page for the details



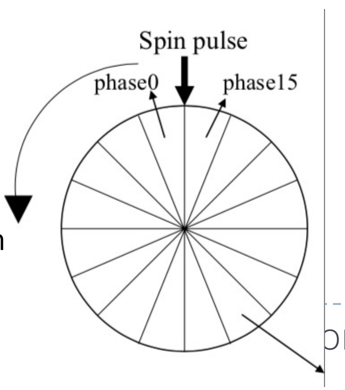








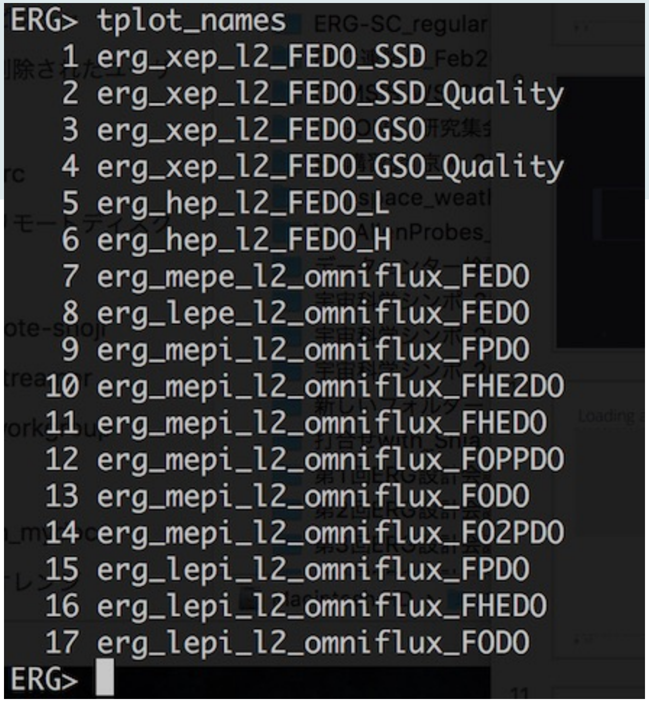
XEP
Directional binning by spin motion



Omni-flux data

Loading and plotting omni-flux data (1)

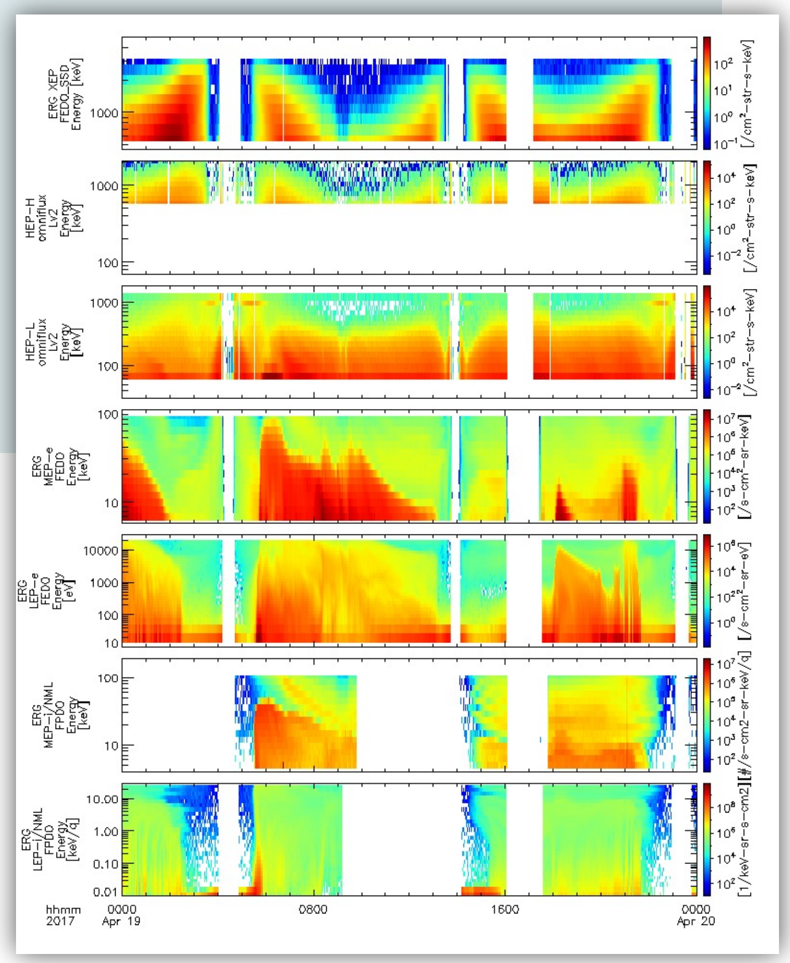
```
;; Set the time span
timespan, '2017-04-19'
;; Load data
erg_load_xep, datatype='omniflux'
erg_load_hep, datatype='omniflux'
erg_load_mepe, datatype='omniflux'
erg_load_lepe, datatype='omniflux'
erg_load_mepi_nml, datatype='omniflux'
erg_load_lepi_nml, datatype='omniflux'
tplot_names
```



```
ERG> tplot_names
   1 erg_xep_l2_FEDO_SSD
   2 erg_xep_l2_FEDO_SSD_Quality
   3 erg_xep_l2_FEDO_GSO
   4 erg_xep_l2_FEDO_GSO_Quality
   5 erg_hep_l2_FEDO_L
   6 erg_hep_l2_FEDO_H
   7 erg_mepe_l2_omniflux_FEDO
   8 erg_lepe_l2_omniflux_FEDO
   9 erg_mepi_l2_omniflux_FPDO
  10 erg_mepi_l2_omniflux_FHE2DO
  11 erg_mepi_l2_omniflux_FHEDO
  12 erg_mepi_l2_omniflux_FOPPDO
  13 erg_mepi_l2_omniflux_FODO
  14 erg_mepi_l2_omniflux_FO2PDO
  15 erg_lepi_l2_omniflux_FPDO
  16 erg_lepi_l2_omniflux_FHEDO
  17 erg_lepi_l2_omniflux_FODO
ERG>
```

Loading and plotting omni-flux data (2)

```
Loadct2, 33  ;; color table: Blue-Red

tplot,[ 'erg_xep_l2_FEDO_SSD', $
  'erg_hep_l2_FEDO_H','erg_hep_l2_FEDO_L', $
  'erg_mepe_l2_omniflux_FEDO', $
  'erg_lepe_l2_omniflux_FEDO', $
  'erg_mepi_l2_omniflux_FPDO', $
  'erg_lepi_l2_omniflux_FPDO'    ]
```



!!! CAUTION !!!

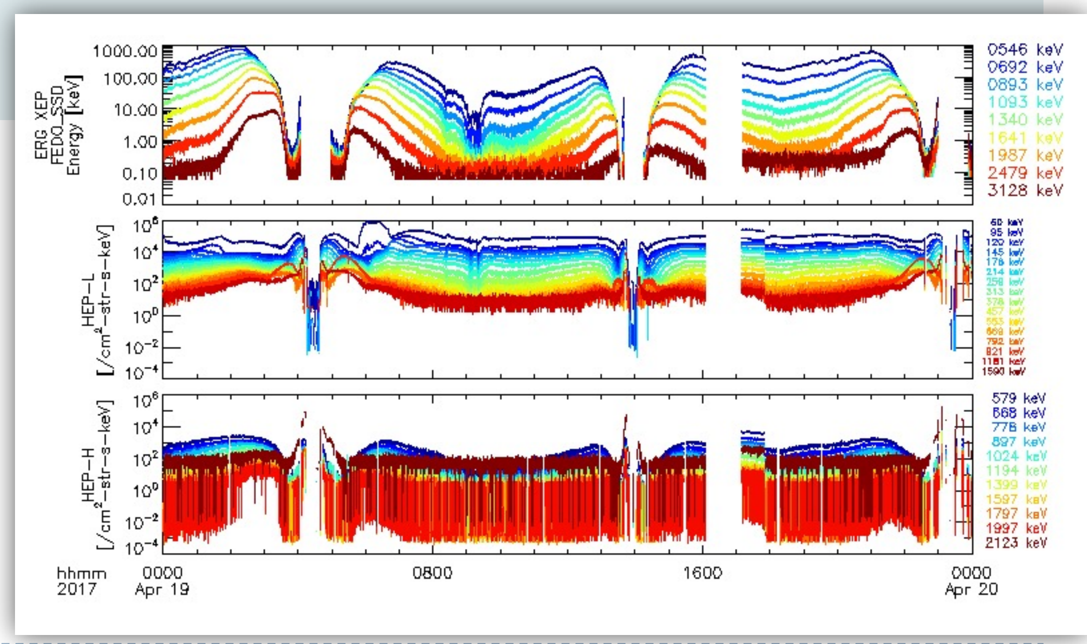
The omni-flux of ERG's particle data is just a simple, arithmetic average of fluxes over all directional bins,

$$omniflux(E) = \frac{1}{N}\sum_{dir.ch.}^{N} flux(dir, E) \neq \frac{1}{\pi}\int_0^{\pi} d\alpha \cdot flux(\alpha, E),$$

NOT equal to a pitch-angle-averaged flux.

Loading and plotting omni-flux data (3)

```
vn = 'erg_xep_l2_FEDO_SSD'
options, vn, spec=0 & ylim, vn, 0, 0, 1

tplot ;; Replot the previously plotted variables

erg_load_hep, datatype='omniflux', /lineplot
tplot, ['erg_xep_l2_FEDO_SSD', 'erg_hep_l2*line']
```



Loading and plotting 3-D flux data (1)

```
;; ERG working group ID/password if the WG-level access control is applied.
uname = '???????????'  &  pass = '????????'

timespan, '2017-04-19'
erg_load_xep, datatype='2dflux'
erg_load_hep, datatype='3dflux', uname=uname, pass=pass
erg_load_mepe, datatype='3dflux'
erg_load_lepe, datatype='3dflux', uname=uname, pass=pass
erg_load_mepi_nml, datatype='3dflux'
erg_load_lepi_nml, datatype='3dflux', uname=uname, pass=pass
```

Setting the keyword, `datatype='3dflux'`, makes erg_load_??? load 3-D flux data, except for XEP whose full resolution data are `'2dflux'`.

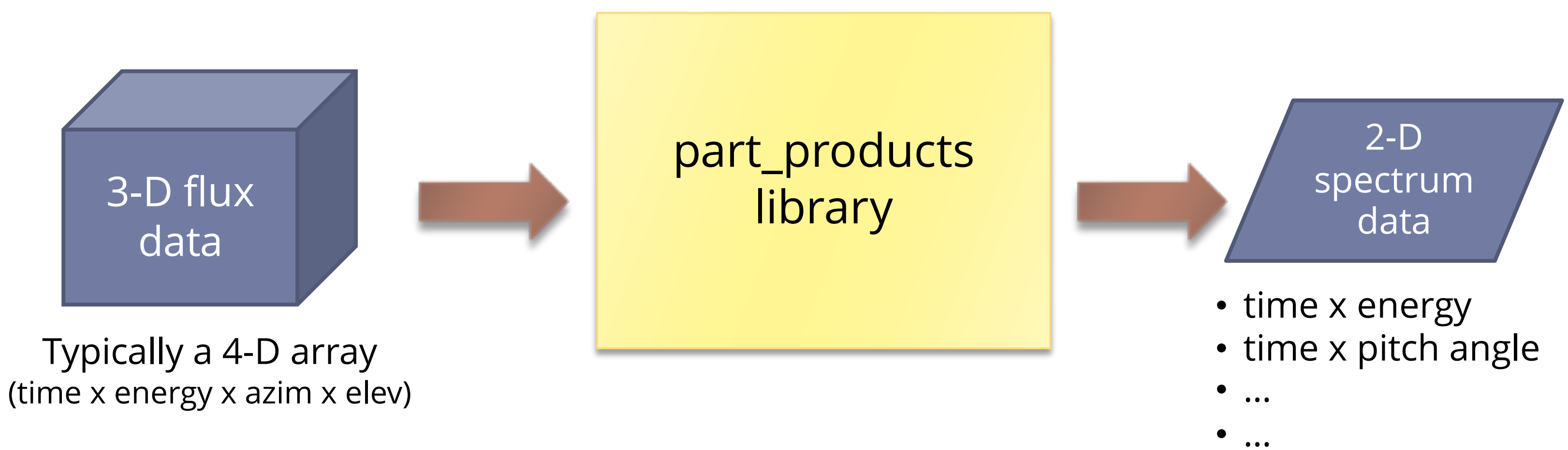
!! CAUTION !!

Generally, a 3-D / 2-D flux data variable itself cannot be plotted with the "tplot" command, because they contains a 3-D or higher dimension array.

Deriving various time-series of spectrum data with part_products

What's "part_products"?

- A set of generic routines bundled to SPEDAS to make tplot variables for various types of spectrum plot.



What's "part_products"?

- This library is available from ERG-SC for LEP-e, LEP-i, MEP-e, MEP-i, HEP, and XEP data.
  - `erg_xep_part_products`: XEP
  - `erg_hep_part_products`: HEP
  - `erg_mep_part_products`: MEP-e and MEP-i
  - `erg_lep_part_products`: LEP-e and LEP-i

part_product"s" (plural!)

Typically a 4-D array
(time x energy x azim x elev)

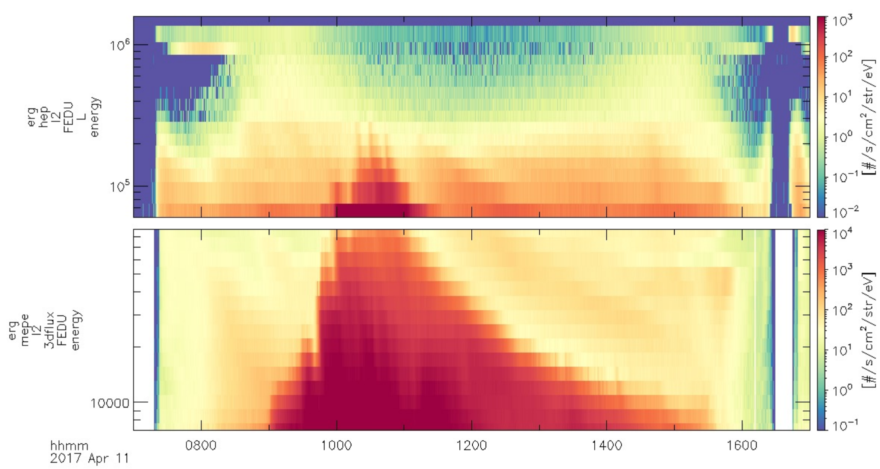
- time x energy
- time x phi/theta angle
- time x pitch angle
- ...

Energy-time spectrogram

```
Loadct2, 74, /reverse   ;; color table: reversed CB-Spectral

del_data, '*'
timespan, '2017-04-11/07:00',10, /hour
erg_load_hep, datatype='3dflux', uname=uname, pass=pass
erg_load_mepe, datatype='3dflux'

erg_hep_part_products, 'erg_hep_l2_FEDU_L'
erg_mep_part_products, 'erg_mepe_l2_3dflux_FEDU'

tplot, ['erg_hep_l2_FEDU_L_energy','erg_mepe_l2_3dflux_FEDU_energy']
```

Running part_produtcts without any option gives omni-directional fluxes by averaging over all directions.

The default unit of energy and differential flux becomes eV and #/s/cm2/str/eV when flux data are processed with part_products.



Energy-time spectrogram in different units

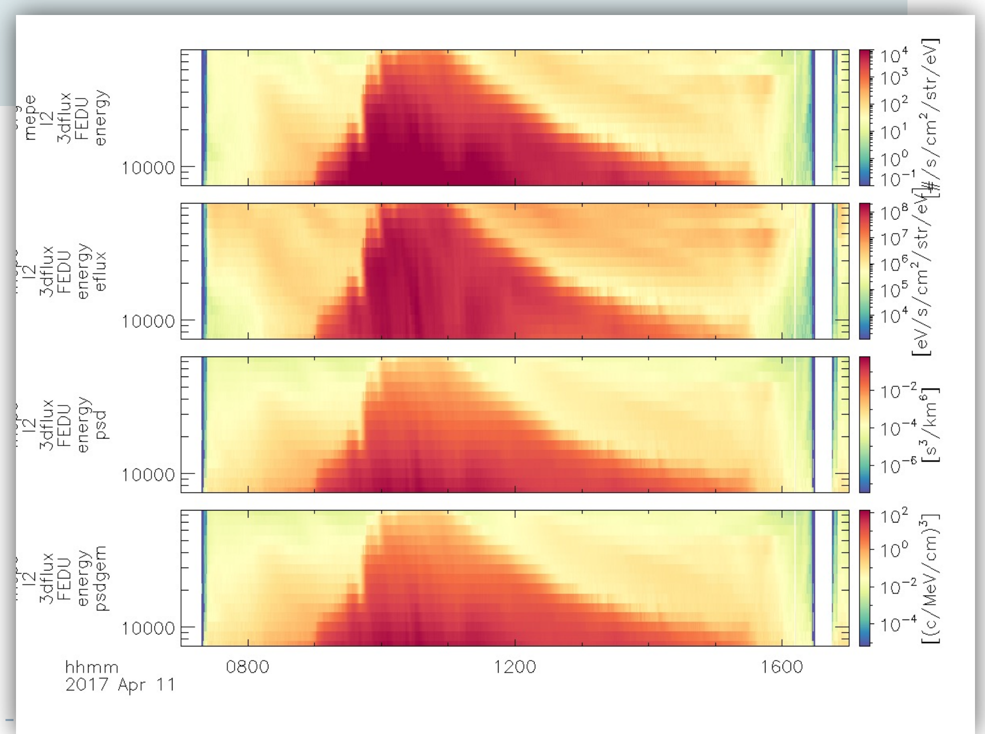
```
erg_mep_part_products, 'erg_mepe_l2_3dflux_FEDU', units='flux' ;; Default
erg_mep_part_products, 'erg_mepe_l2_3dflux_FEDU', units='eflux', suffix='_eflux'
erg_mep_part_products, 'erg_mepe_l2_3dflux_FEDU', units='df', suffix='_psd'
erg_mep_part_products, 'erg_mepe_l2_3dflux_FEDU', units='df', suffix='_psdgem', /rela

tplot, ['erg_mepe_l2_3dflux_FEDU_energy*']
```

Differential number flux
[/s/cm²/sr/eV]

Differential energy flux
[eV/s/cm²/sr/eV]

Phase space density
[s³/km⁶]

Phase space density
[(c/MeV/cm)³]

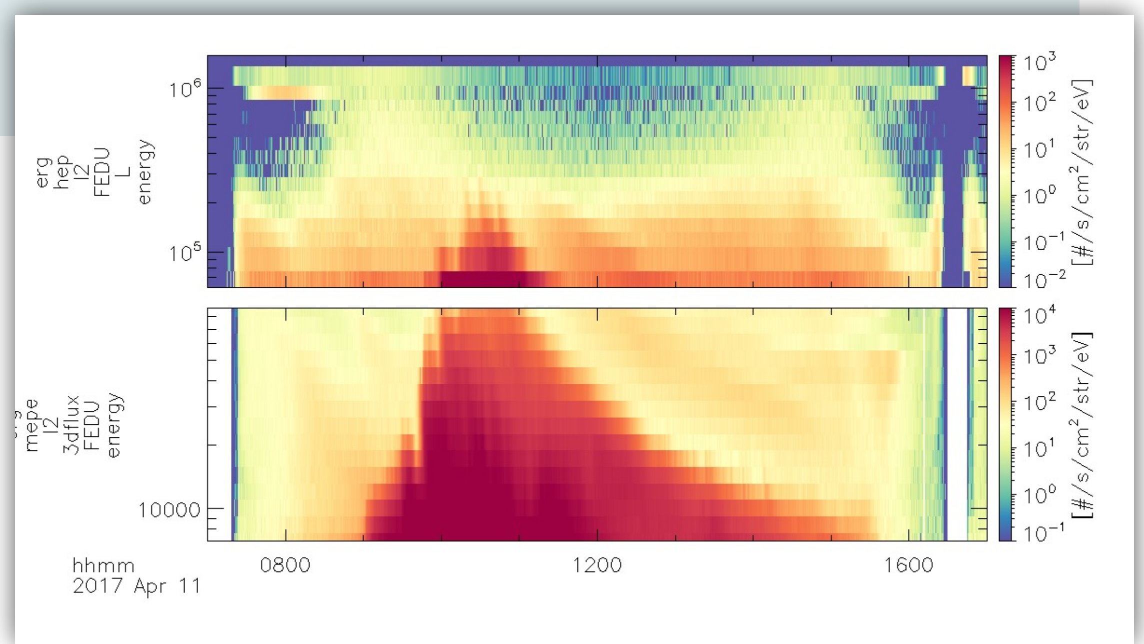


Energy-time spectrogram for fluxes in a limited range of direction

```
get_timespan, tr
erg_hep_part_products, 'erg_hep_l2_FEDU_L', theta=[-30.,30], trange=tr
erg_mep_part_products, 'erg_mepe_l2_3dflux_FEDU', phi=[0.,90], trange=tr

tplot, ['erg_hep_l2_FEDU_L_energy','erg_mepe_l2_3dflux_FEDU_energy']
```

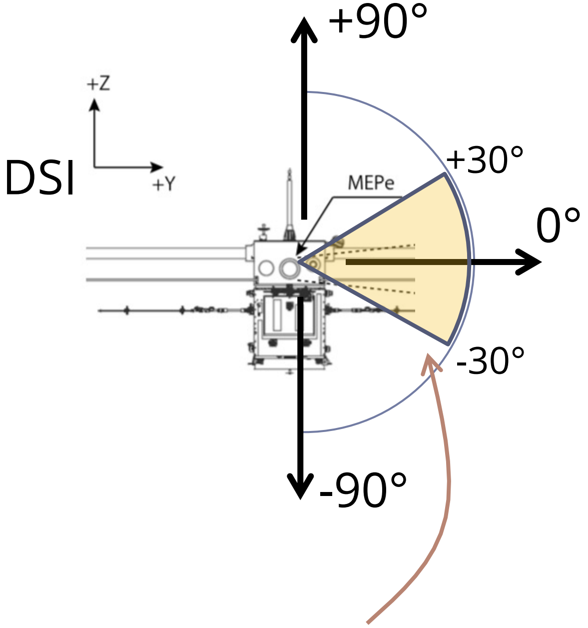
phi and **theta** should be given as a range of angle in the DSI coordinates.

Keywords **theta** and **phi** can be set together to specify a limited area of solid angle in DSI.



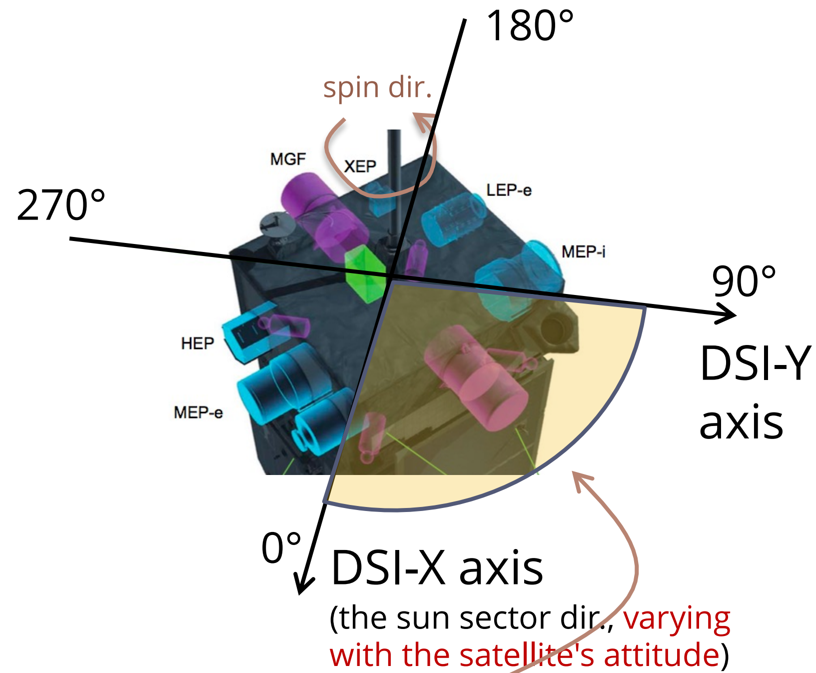
How to express a directional range by keywords "phi" and "theta"

"theta" angle
= **elevation** angle in DSI coord.
of particles' flux direction




```
theta = [ -30, 30 ]
```

"phi" angle
= **azimuthal** angle in DSI coord.
of particles' flux direction



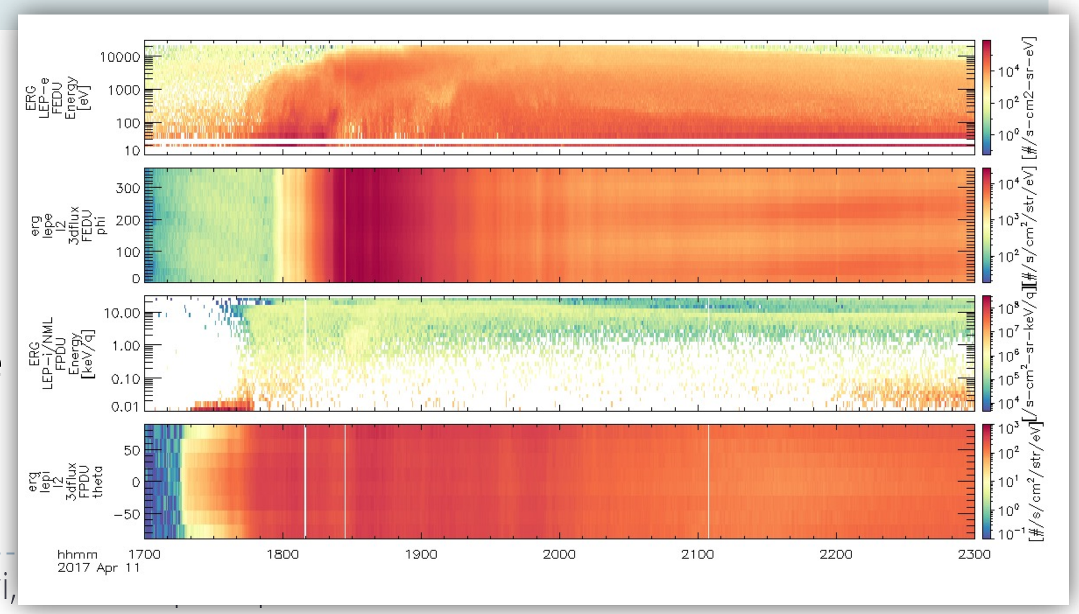

```
phi = [ 0, 90 ]
```

Phi-/Theta-angle spectrogram

```
del_data, '*'
timespan, '2017-04-11/17:00', 6, /hour
erg_load_lepe, datatype='3dflux', varformat='FEDU'
erg_load_lepi_nml, datatype='3dflux', varformat='FPDU'

get_timespan, tr
erg_lep_part_products, 'erg_lepe_l2_3dflux_FEDU', outputs='phi', trange=tr, energy=[3000.,10000.]
erg_lep_part_products, 'erg_lepi_l2_3dflux_FPDU', outputs='theta', trange=tr, energy=[8000.,20000.]
zlim, 'erg_lepi_l2_3dflux_FPDU_theta', 1e-1, 1e+3, 1

tplot, ['erg_lepe_l2_3dflux_FEDU*', 'erg_lepi_l2_3dflux_FPDU*']
```

The flux distribution over the entire "phi" or "theta" angle in DSI coord. is obtained.

Keyword 'energy' specifies an energy range in eV for which particle flux data are averaged to deduce a phi-/theta-spectrogram.



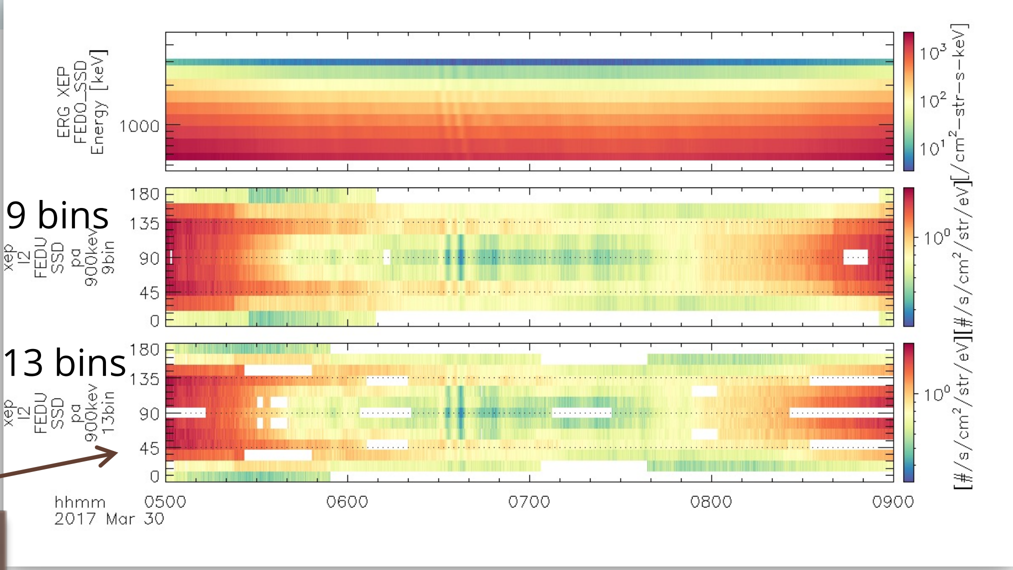
Pitch-angle spectrogram (1)

```
del_data, '*'
timespan, '2017-03-30/05:00', 4, /hour & get_timespan, tr
erg_load_xep, datatype='omniflux' & erg_load_xep, datatype='2dflux'
erg_load_mgf & erg_load_orb

erg_xep_part_products, 'erg_xep_l2_FEDU_SSD', output='pa', pos='erg_orb_l2_pos_gse', mag='erg_mgf_l2_mag_8sec_dsi', trange=tr, energy=[800000., 1000000.], /no_ang_weighting, regrid=[16, 9], suffix='_893kev_9pabin'
erg_xep_part_products, 'erg_xep_l2_FEDU_SSD', output='pa', pos='erg_orb_l2_pos_gse', mag='erg_mgf_l2_mag_8sec_dsi', trange=tr, energy=[800000., 1000000.], /no_ang_weighting, regrid=[16, 13], suffix='_893kev_13pabin'

tplot, 'erg_xep_l2_' + [ 'FEDO_SSD', 'FEDU_SSD_*pabin' ]
```

output='pa' gives a pitch angle (PA) sorted spectrogram.

You should set an energy range with energy keyword, otherwise fluxes are averaged over all the energy range.

The 2nd element of keyword regrid sets the number of PA bins.



Some of the 13.8° (180/13) bins miss the data, because XEP, onboard the spinning satellite, gives e- flux at every 22.5°, which is larger than the PA bin width (also see page 24).

The flux modulations discussed by Teramoto+, 2019

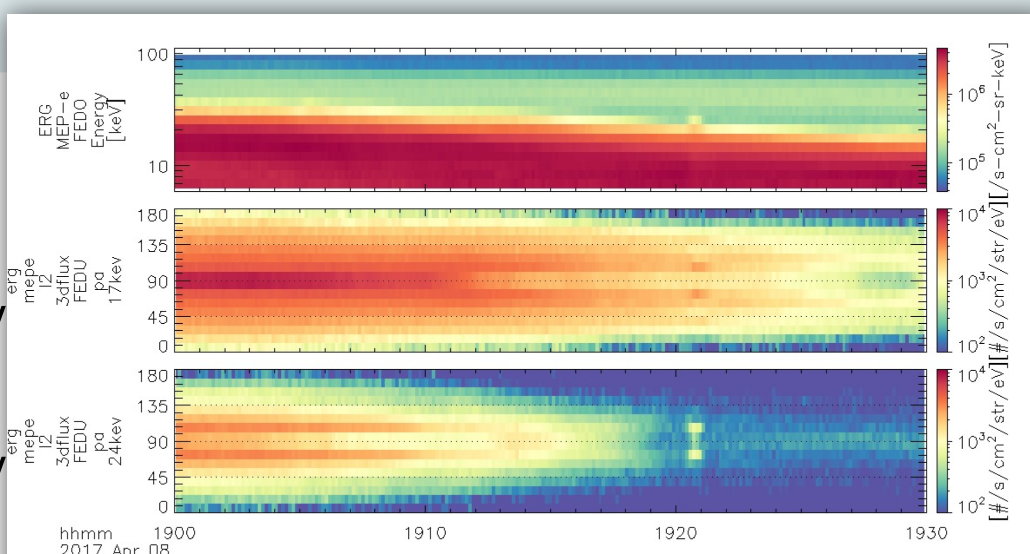
Pitch-angle spectrogram (2)

```
del_data, '*'
timespan, '2017-04-08/19:00', 30, /min & get_timespan, tr
erg_load_mepe, datatype='3dflux', varformat='FEDU'
erg_load_mgf & erg_load_orb

erg_mep_part_products, 'erg_mepe_l2_3dflux_FEDU', output='pa', pos='erg_orb_l2_pos_gse',
mag='erg_mgf_l2_mag_8sec_dsi', energy=[16000., 18000.], suffix='_17kev', trange=tr, /no_ang_weighting
erg_mep_part_products, 'erg_mepe_l2_3dflux_FEDU', output='pa', pos='erg_orb_l2_pos_gse',
mag='erg_mgf_l2_mag_8sec_dsi', energy=[23000., 25000.], suffix='_24kev', trange=tr, /no_ang_weighting

zlim, 'erg_mepe_l2_3dflux_FEDU_pa*', 1e+2, 1e+4, 1
tplot, 'erg_mepe_l2_3dflux_'+['FEDO', 'FEDU_pa*']
```

MEP-e omniflux

PA spec. for 17 keV

PA spec. for 24 keV



The pitch-angle spectra discussed by Kurita+, GRL, 2018

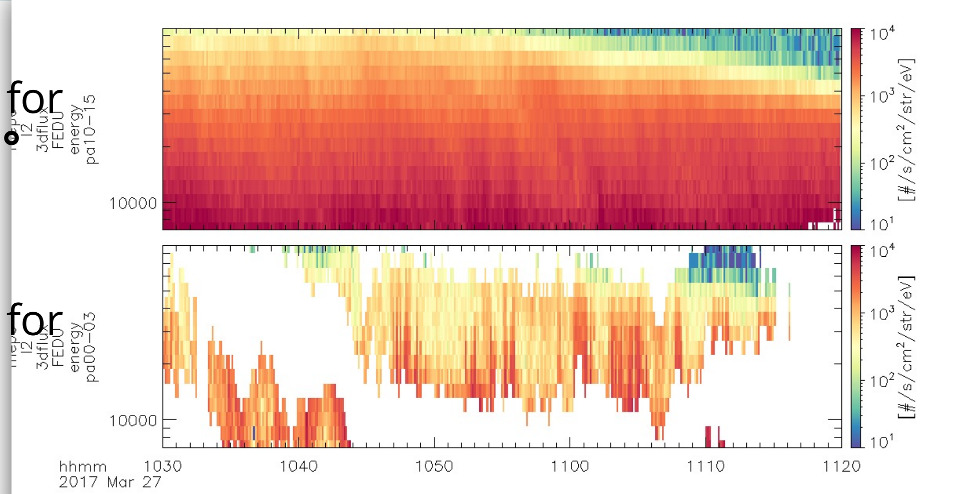
Energy-time spectrogram for a limited pitch-angle range

```
del_data, '*'
timespan, '2017-03-27/10:30', 50, /min & get_timespan, tr
erg_load_mepe, datatype='3dflux'
erg_load_mgf & erg_load_orb

erg_mep_part_products, 'erg_mepe_l2_3dflux_FEDU', pos='erg_orb_l2_pos_gse',
mag='erg_mgf_l2_mag_8sec_dsi', output='energy', trange=tr, pitch=[0.,3.],
suffix='_pa00-03', /no_ang_weighting
erg_mep_part_products, 'erg_mepe_l2_3dflux_FEDU', pos='erg_orb_l2_pos_gse',
mag='erg_mgf_l2_mag_8sec_dsi', output='energy', trange=tr, pitch=[10.,15.],
suffix='_pa10-15', /no_ang_weighting

tplot, 'erg_mepe_l2_3dflux_FEDU_energy_pa'+['10-15', '00-03']
```

`output='energy', pitch=[min, max]` generates an energy-time spectrogram for particles with pitch angles of min < PA < max.

E-t diagram for PA = 10°–15°

E-t diagram for PA = 0°–3°



The electron flux variations discussed by Kasahara+, Nature, 2018

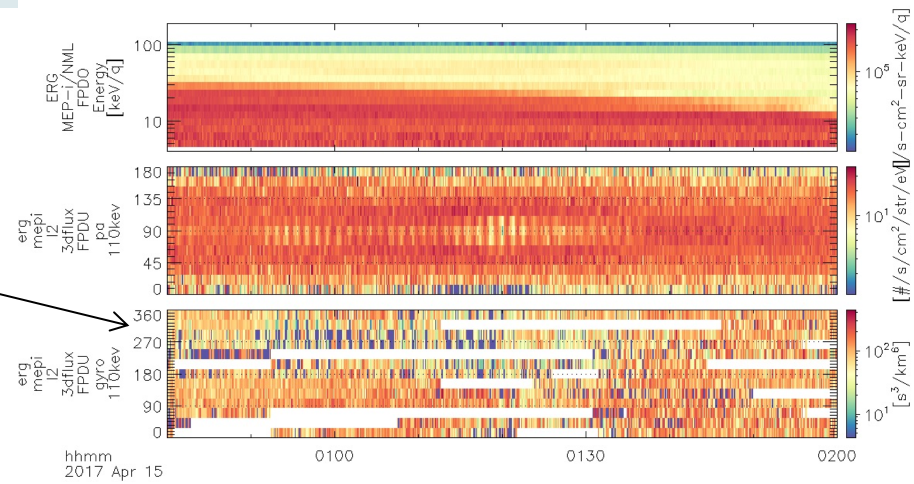
Gyro-phase spectrogram

```
del_data, '*'
timespan, ['2017-04-15/00:40', '2017-04-15/02:00'] & get_timespan, tr
erg_load_mepi_nml, datatype='3dflux', varformat='FPDU'
erg_load_mgf & erg_load_orb


erg_mep_part_products, 'erg_mepi_l2_3dflux_FPDU', output='pa', pos='erg_orb_l2_pos_gse',
mag='erg_mgf_l2_mag_8sec_dsi', energy=[108000., 112000.], regrid=[16,13], trange=tr, suffix='_110kev',
fac_type='mphism', /no_ang_weighting
erg_mep_part_products, 'erg_mepi_l2_3dflux_FPDU', output='gyro', pos='erg_orb_l2_pos_gse',
mag='erg_mgf_l2_mag_8sec_dsi', energy=[108000., 112000.], pitch=[85.,95.], regrid=[13,16], trange=tr,
suffix='_110kev', fac_type='mphism', units='df', /no_ang_weighting


tplot, 'erg_mepi_l2_3dflux_'+['FPDO', 'FPDU_pa*', 'FPDU_gyro*']
```

Field-aligned (FA) coord. is given by keyword `fac_type`. See the appendix for the definition details.

`output='gyro', pitch=[min, max]` gives a so-called gyro-phase spectrogram for particles of min < PA < max.

0°: a flux going in the x direction
90°: a flux going in the y direction
180°: a flux going in the -x direction
270°: a flux going in the -y direction
on the X-Y plane in a designated FA coordinates



The flux modulations discussed by Yamamoto+, GRL, 2018

`no_ang_weighting` keyword?

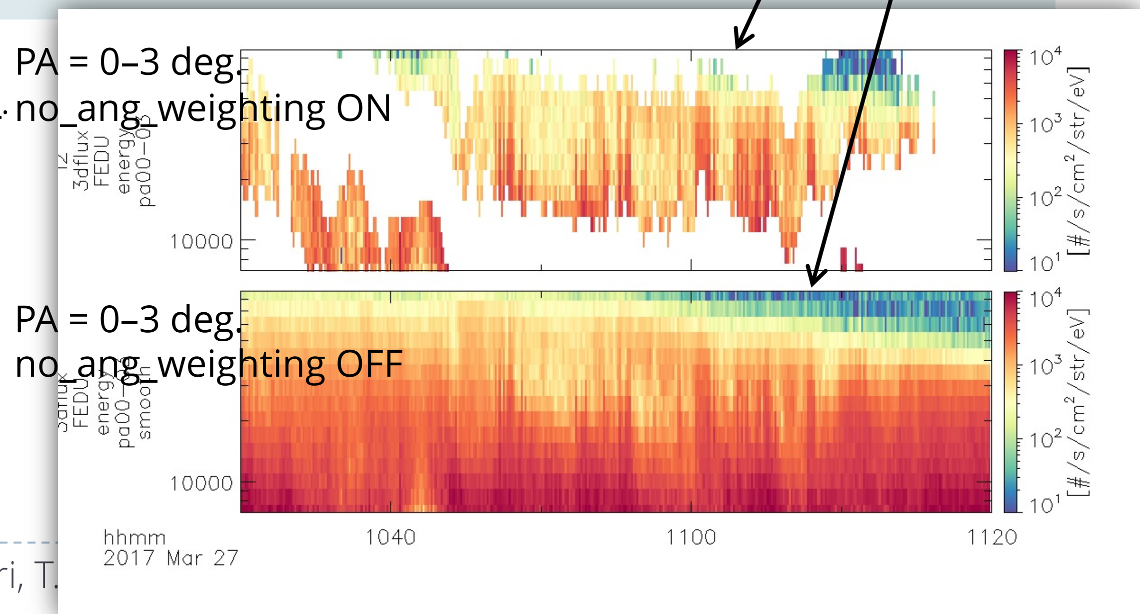
```
del_data, '*'
timespan, '2017-03-27/10:30', 50, /min & get_timespan, tr
erg_load_mepe, datatype='3dflux'
erg_load_mgf & erg_load_orb

erg_mep_part_products, 'erg_mepe_l2_3dflux_FEDU', pos='erg_orb_l2_pos_gse',
mag='erg_mgf_l2_mag_8sec_dsi', output='energy', trange=tr, pitch=[0.,3.],
suffix='_pa00-03', /no_ang_weighting
erg_mep_part_products, 'erg_mepe_l2_3dflux_FEDU', pos='erg_orb_l2_pos_gse',
mag='erg_mgf_l2_mag_8sec_dsi', output='energy', trange=tr, pitch=[0.,3.],
suffix='_pa00-03_smooth'

tplot, 'erg_mepe_l2_3dflux_FEDU_energy_pa*'
```

The two spectra are totally different, despite that the same PA range is set??

If `no_ang_weighting` is set, <u>any smoothing over direction</u> in the velocity space is <u>suppressed</u>.

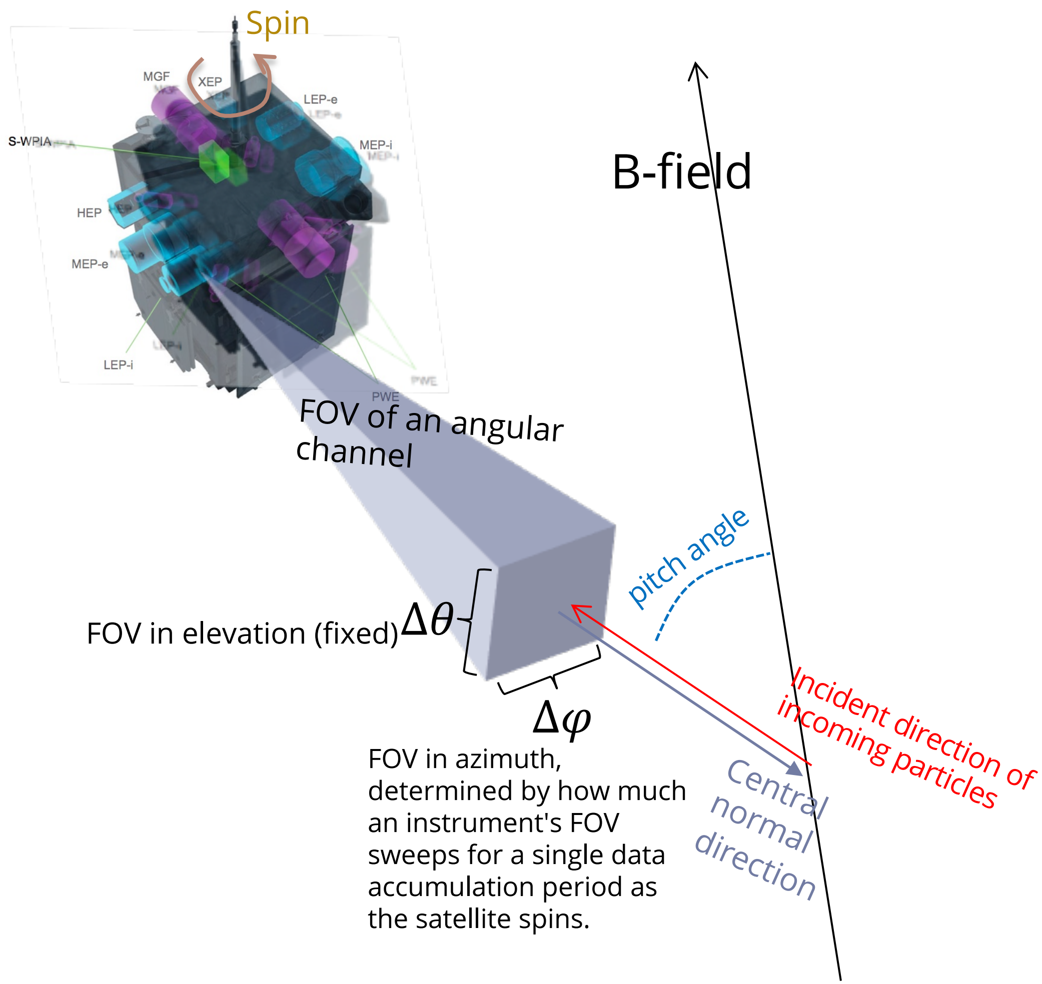
<u>You must first check a non-smoothed result with this option in your data alnalysis.</u>

By default, part_products interpolate over neighboring directional bins to obtain a smoothed distribution. The result thus could contain finite flux values even in directions that were not measured by the instrument, leading to a misleading interpretation.



Resolution of pitch-angle (PA) spectra

ATTENTION!!



You **must set a right angular width of PA bin/range** in deriving PA spec. or E-t spec. for a limited PA range, considering the angular resolution of measurement and your scientific purpose. **Unrealistically small PA bin may give a meaningless or misleading result**.

Reason:
Although an instrument scans a finite angular range as a single directional channel, part_products uses only the central normal direction of the directional channel to evaluate a pitch angle.

The effective resolution of PA spec. is determined by the effective field-of-view (FOV) $(\Delta\theta, \Delta\varphi)$ of a single angular bin for which an instrument accumulates particles' counts. The FOV range varies with instrument.

For example, $\Delta\theta$ *and* $\Delta\varphi$ are 12° and 22.5°, respectively for HEP 3-D flux data. Thus, it is nonsense to use 5° PA bins to derive PA spec. and discuss the resultant PA distribution with 5° accuracy: the instrument cannot revolve such a fine angle range.

Velocity moments of 3-D distribution functions

Moment calculation by part_products

- A part_products calls moments_3d(), an internal routine, to calculate various velocity moments from a 3-D distribution function.

- So far the part_products for LEP-e, LEP-i, MEP-e and MEP-i support the moment calculation.

Moment calculation (1)

```
del_data, '*'
timespan, '2017-03-27/10:00', 1, /hour  & get_timespan, tr
erg_load_mepe, datatype='3dflux', varformat='FEDU'
erg_load_mepi_nml, datatype='3dflux', varformat='FPDU'
erg_load_mgf & erg_load_orb
erg_mep_part_products, 'erg_mepi_l2_3dflux_FPDU', pos='erg_orb_l2_pos_gse',
mag='erg_mgf_l2_mag_8sec_dsi', output='moments', trange=tr
erg_mep_part_products, 'erg_mepe_l2_3dflux_FEDU', pos='erg_orb_l2_pos_gse',
mag='erg_mgf_l2_mag_8sec_dsi', output='moments', trange=tr
```

```
ERG> tplot_names, 'erg_mepi_l2_3dflux_FPDU_*'
  47 erg_mepi_l2_3dflux_FPDU_avgtemp
  48 erg_mepi_l2_3dflux_FPDU_density
  49 erg_mepi_l2_3dflux_FPDU_eflux
  50 erg_mepi_l2_3dflux_FPDU_flux
  51 erg_mepi_l2_3dflux_FPDU_mftens
  52 erg_mepi_l2_3dflux_FPDU_ptens
  53 erg_mepi_l2_3dflux_FPDU_sc_current
  54 erg_mepi_l2_3dflux_FPDU_velocity
  55 erg_mepi_l2_3dflux_FPDU_vthermal
  56 erg_mepi_l2_3dflux_FPDU_magf
  57 erg_mepi_l2_3dflux_FPDU_magt3
  58 erg_mepi_l2_3dflux_FPDU_t3
  59 erg_mepi_l2_3dflux_FPDU_sc_pot
  60 erg_mepi_l2_3dflux_FPDU_symm
  61 erg_mepi_l2_3dflux_FPDU_symm_theta
  62 erg_mepi_l2_3dflux_FPDU_symm_phi
  63 erg_mepi_l2_3dflux_FPDU_symm_ang
ERG>
```

Primary parameters calculated with the part_products:

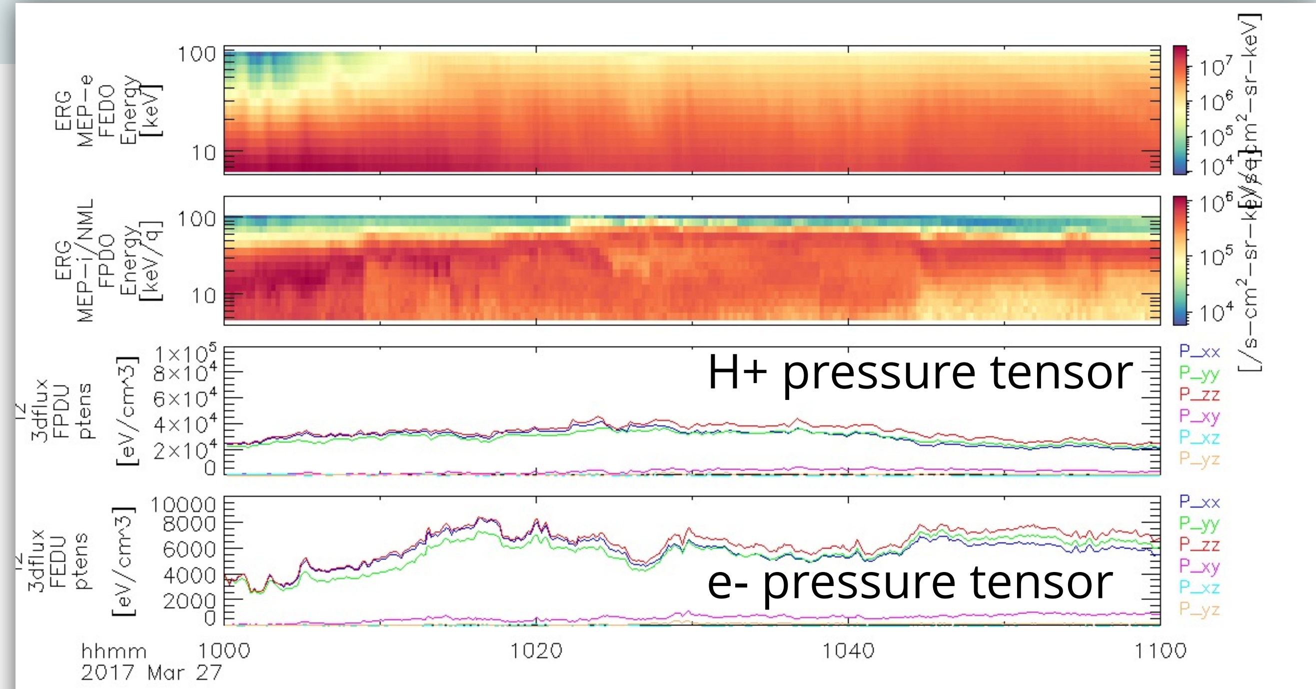
- density: number density
- avgtemp: scalar temperature (!)
- velocity: bulk velocity
- vthermal: thermal velocity
- mtens: momentum flux density tensor
- ptens: pressure tensor
- t3: temperature tensor (!)
- magt3: perpendicular/parallel temperature (!)
- flux: number flux
- eflux: energy flux

All vector and tensor quantities in DSI coordinates.

(!) Note that these are NOT a temperature defined as a width of Maxwellian distribution.

Moment calculation (2)

```
ylim, '*FPDU*ptens', 0, 1e+5, 0   ;; set y-ranges with linear scale
ylim, '*FEDU*ptens', 0, 1e+4, 0
tplot, ['erg_mep?_l2_3dflux_'+['F?DO','F?DU_ptens']]
```



Particle flux and pressure variations discussed by Hori+, GRL, 2018

Caveats of velocity moment data

ATTENTION!!

- **Many potential pitfalls** in deriving and interpreting velocity moment data:
  - We cannot integrate flux: flux values are measured at discrete energies and directions and thus are just "summed up" in the velocity space to yield a velocity moment value.
    - No fitting on a velocity distribution is performed in the velocity moment calculation by the part_products.
  - Velocity moments are always based on a limited energy / angular range of particle data: we cannot sum up over the entire velocity space.
  - Temperature is calculated essentially as **the partial pressure** divided by **the partial density** and, as a result, can be QUITE DIFFERENT from a temperature defined for the Maxwellian distribution.
- If you examine velocity moment data in your study, we **strongly recommend contacting the instrument's PI team in the early stage of your research** and thereby working with them, to avoid misuse and misinterpretation of the data.

It is also highly recommended to consult a good tutorial of the velocity moment calculation from particle data provided by Yokota-san:
https://ergsc.isee.nagoya-u.ac.jp/data/website/archives/documents_old/science201809/pdf/20180920_ERG_SWG_Tutorial_Yokota.pdf

How to use internal routines of part_products

An alternative way to deduce pitch-angle spectra rather manually

erg_???_get_dist(): Put 3-D flux data in a 3-D data structure

```
timespan, '2017-05-28'
erg_load_lepi_nml, datatype='3dflux', varformat='FPDU'

dist = erg_lepi_get_dist( 'erg_lepi_l2_3dflux_FPDU', /structure )

help, dist
help, dist[0]
```

Each `get_dist()` should be used for the 3-D flux data (2-D flux data for XEP) of each instrument, by providing a tplot variable of 3-D flux data as the argument.

- For LEP-i: `erg_lepi_get_dist()`
- For LEP-e: `erg_lepe_get_dist()`
- For MEP-e: `erg_mepe_gest_dist()`
- For MEP-i: `erg_mepi_get_dist()`
- For HEP: `erg_hep_get_dist()`
- For XEP: `erg_xep_get_dist()`

3-D data structure common to particle data that SPEDAS can handle

```
ERG> help, dist[100]
   PROJECT_NAME    STRING     'ERG'
   SPACECRAFT      LONG                  1
   DATA_NAME       STRING     'LEP-i Proton 3dflux'
   UNITS_NAME      STRING     'flux'
   UNITS_PROCEDURE STRING     'erg_convert_flux_units'
   SPECIES         STRING     'proton'
   VALID           BYTE          1
   CHARGE          FLOAT            1.00000
   MASS            FLOAT          0.0104535
   TIME            DOUBLE        1.4959304e+09
   END_TIME        DOUBLE        1.4959304e+09
   DATA            FLOAT      Array[30, 16, 8]
   BINS            FLOAT      Array[30, 16, 8]
   ENERGY          FLOAT      Array[30, 16, 8]
   DENERGY         FLOAT      Array[30, 16, 8]
   NENERGY         LONG                30
   NBINS           LONG               128
   PHI             FLOAT      Array[30, 16, 8]
   DPHI            FLOAT      Array[30, 16, 8]
   THETA           FLOAT      Array[30, 16, 8]
   DTHETA          FLOAT      Array[30, 16, 8]
ERG>
```

An example for LEP-i 3-D flux data:

`dist` is an array of structures each of which contains a set of data for each spin.

"DATA" holds the flux data as a 3-D array of 30 ene. ch x 16 spin sector x 8 sensors.

ENERGY and DENERGY are the central energies and energy ranges of the energy channels.

PHI, DPHI, THETA, and DTHETA have phi/theta angles of particle-going directions and angular widths measured by directional channels of a particle instrument in the DSI coordinate system.

By default, phi/theta angles in DSI coordinates are stored by `erg_get_???_dist()`.

`erg_pgs_make_fac, spd_pgs_do_fac`: Transformation to the field-aligned coordinates (FAC)

```
;; Prepare the MGF and orbit data
erg_load_mgf & set_erg_var_label
;; Make transformation matrices for the FAC
erg_pgs_make_fac, dist.time, 'erg_mgf_l2_mag_8sec_dsi', 'erg_orb_l2_pos_gse', $
                  fac_output=fac_mat, fac_type='mphism'

dist_fac = dist ;; Make a copy of the dist structure

;; Transform phi/theta values in the dist structure to those in FAC for each time frame
for i = 0L, n_elements(dist.time)-1 do begin & $
  spd_pgs_do_fac, dist[i], reform( fac_mat[i, *, *], [3,3] ), $
                  output=dist_tmp,   error=error & $
  dist_fac[i] = dist_tmp & $
endfor
```

Note that:

- Both the magnetic field data in DSI and the orbit data in GSE should be given to `erg_pgs_make_fac`. They are automatically interpolated in time to match time frames of particle data given as a 1-D array in SPEDAS time unit (`dist.time` in the above case).
- The transformation matrix is made for the particle time frames, as a 3-D array of time x 3 x 3 (`fac_mat` in the above case).
- `spd_pgs_do_fac` changes only phi and theta arrays in a particle data structure.

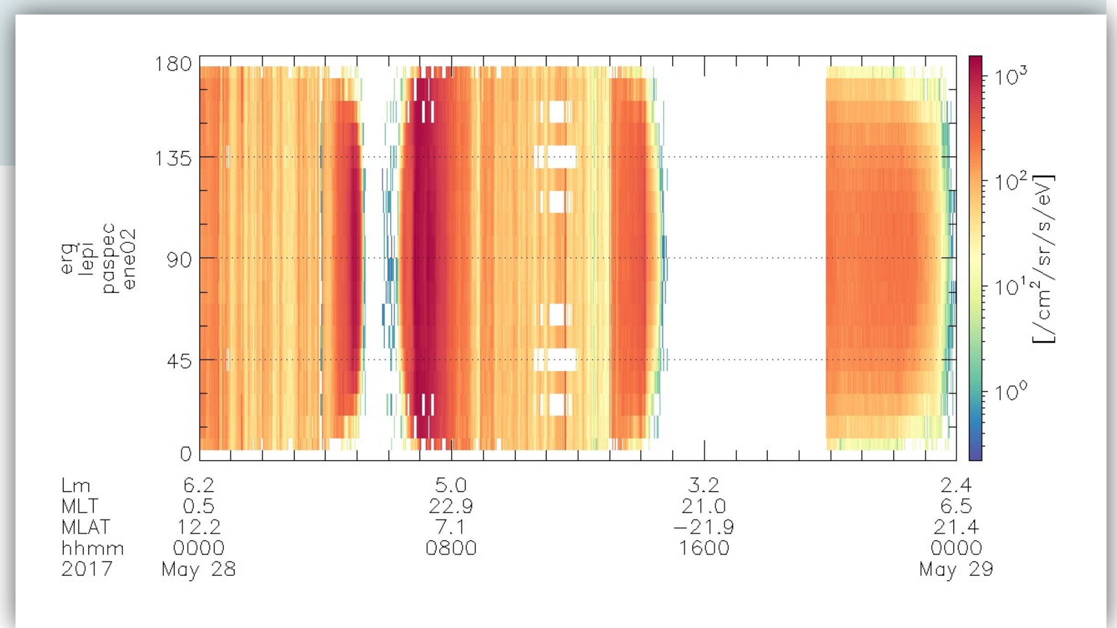
Binning and averaging flux data in FAC to deduce pitch-angle spectra

```
dist_fac.theta = 90. - dist_fac.theta  ;; colat. in FAC = pitch angle

;; Prepare data arrays for a selected energy channel
enech = 2 ;; ch02 --> 19.2 keV
dat_arr = reform( dist_fac.data[ enech, *, * ] )
pa_arr = reform( dist_fac.theta[ enech, *, * ] )
ntimes = n_elements(dist_fac.time)
dim = dimen( dist_fac[0].data )
t_arr =  rebin( reform( dist_fac.time, [1,1,ntimes] ), [dim[1:*],ntimes] )

;; Use a generic routine "bin2d" to calculate average fluxes for the time x pitch-angle bins
id = where( finite(dat_arr) and finite(pa_arr) ) ;;To exclude NaN and Inf from the averaging with bin2d
bin2d, t_arr[id], pa_arr[id], dat_arr[id], xrange=minmax(t_arr), yrange=[0.,180.], binum=[ntimes,18], $
       xc=time_c, yc=pa_c, ave=aveflux, binhist=binnm, /double
```

- `Bin2d` calculates an average flux for each time x pitch-angle bin. No smoothing or interpolation is applied, unlike how data are averaged by part_products.
- The original version of `bin2d.pro` does not accept `/double` keyword. Please download and **use `bin2d.pro` and `bin1d.pro` of the SPEDAS-j tools**, which are available from the SPEDAS-j website at: `https://github.com/spedas-j/member_contrib/tree/master/misc/bin12d`

Binning and averaging flux data in FAC to deduce pitch-angle spectra (cont'd)

```
;; Put the resultant arrays in a tplot varirable
vname = 'erg_lepi_paspec_ene02'
store_data, vname, data={ x:time_c, y:aveflux, v:pa_c }
;; Set some plot properties
options, vname, spec=1, constant=[45,90,135], ytickinterval=45., yminor=3
Options, vname, ztitle='[/cm!U2!N/sr/s/eV]', zticklen=-0.4, ztickunits='scientific'
ylim, vname, 0, 180, 0
zlim, vname, 0, 0, 1 ;; auto-scale in log

;; Plot!
tplot, vname
```



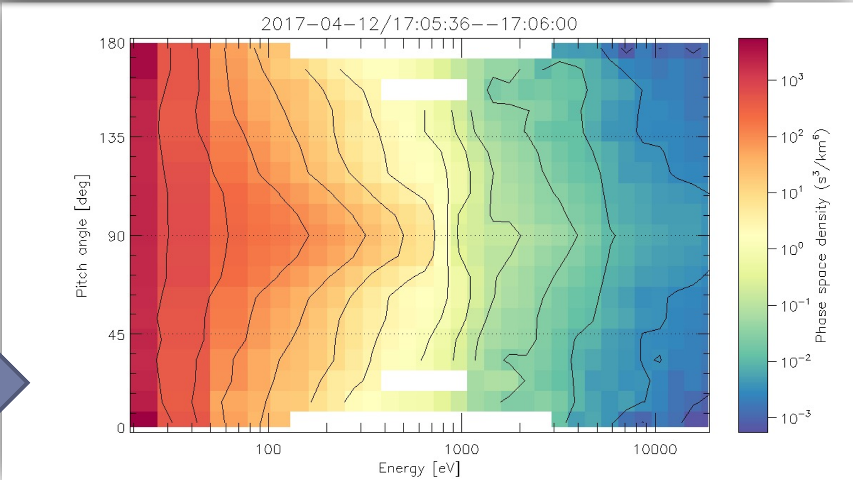
One of the pitch-angle spectra shown in Asamura+, EPS, 2018 is reproduced!

An easy energy-pitch-angle spectrogram plotter
erg_part_en_pa_spec_plot

```
erg_part_en_pa_spec_plot, dist $
   , time=time $       ; a time or time range for plotting
   , units=units $     ; physical unit 'flux','eflux','df_km','df_cm'
   , with_contour=with_contour $  ; to overlay contour lines
   , zrange=zrange $   ; explicitly set the range for the color scale
   , npabin=npabin $   ; number of pitch angle bins (default: 19)
   , rslt=rslt $       ; to obtain data arrays which have been plotted
   , noplot=noplot $   ; set to suppress replotting
   , transpose=transpose  : set to make an energy v.s. PA plot
```

You can use this routine for a 3D data structure of all particle data.

```
timespan, '2017-04-12/16:00', 2, /hour
get_timespan, tr
erg_load_lepe, datatype='3dflux'
erg_load_mgf & erg_load_pos

dists = erg_lepe_get_dist( $
         'erg_lepe_l2_3dflux_FEDU', trange=tr)
erg_part_en_pa_spec_plot, dists, $
      time=['2017-04-12/17:05:35','2017-04-12/17:06:08'], $
      /with_contour, units='df_km'
```

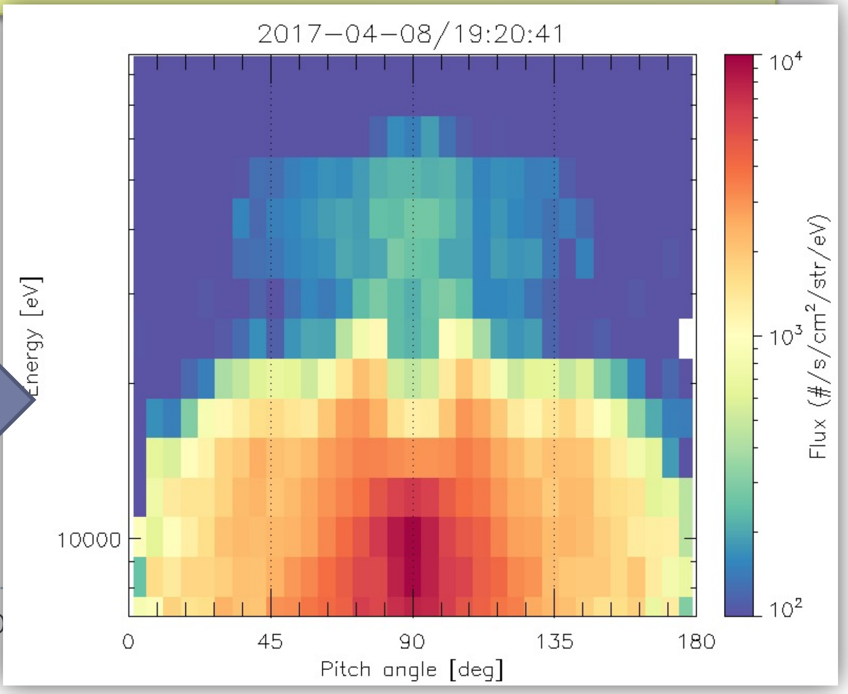


An easy energy-pitch-angle spectrogram plotter
erg_part_en_pa_spec_plot

```
erg_part_en_pa_spec_plot, dist $
   , time=time $        ; a time or time range for plotting
   , units=units $      ; physical unit 'flux','eflux','df_km','df_cm'
   , with_contour=with_contour $  ; to overlay contour lines
   , zrange=zrange $   ; explicitly set the range for the color scale
   , npabin=npabin $   ; number of pitch angle bins (default: 19)
   , rslt=rslt $        ; to obtain data arrays which have been plotted
   , noplot=noplot $   ; set to suppress replotting
   , transpose=transpose  : set to make an energy v.s. PA plot
```

Keyword **transpose** makes an enrgy v.s. PA plot.

```
timespan, '2017-04-08/19:00', 30
get_timespan, tr
erg_load_mepe, datatype='3dflux'
erg_load_mgf & erg_load_pos

dists = erg_mepe_get_dist( $
        'erg_mepe_l2_3dflux_FEDU', trange=tr)
erg_part_en_pa_spec_plot, dists, /transpose,
time='2017-04-08/19:20:41', zrange=[1e+2, 1e+4] ,
npabin=33
```



Appendix

Definition of the FA coordinate system"s" used by part_products

In the field-aligned (FA) coordinate systems, the Z-axis is always in the local magnetic field direction. An X-axis or Y-axis should be defined separately to form a right-handed system. The following options for the Y-axis are available in the part_products library, which is usually given by keyword "`fac_type`" to the `erg_pgs_make_fac` routine.

`'xgse'`

- Y-axis: the vector product of Z-axis and the Xgse direction ($e_y = e_z \times e_{x\_gse}$)
- X-axis: $e_y \times e_z$

`'(m)phigeo'`

- Y-axis: the vector product of Z-axis and the phi direction (roughly eastward) in the geographical (GEO) coordinate system at a satellite location. `mphigeo` uses the negative phi direction (roughly westward) instead.
- X-axis: $e_y \times e_z$ (roughly radially outward for `phigeo` and radially inward for `mphigeo`)

`'(m)phism'`

- Y-axis: the vector product of Z-axis and the phi direction (roughly eastward) in the solar-magnetic (SM) coordinate system at a satellite location. `mphism` uses the negative phi direction (roughly westward) instead.
- X-axis: $e_y \times e_z$ (roughly radially outward for `phism` and radially inward for `mphism`)

`'xdsi'`

- Y-axis: the vector product of Z_{SGI} axis and X_{DSI} axis. Can be calculated with MGF 8-s data during eclipse periods.