pySPEDAS: Space Physics Environment Data Analysis Software in Python

Eric W. Grimes[1], Nikolaos Hatzigeorgiu[2], James W. Lewis[2], Cynthia Russell[1], James M McTiernan[2], Alexander Drozdov[1] and Vassilis Angelopoulos[1]

(1) University of California, Los Angeles, Department of Earth, Planetary, and Space Sciences and Institute of Geophysics and Planetary Physics, Los Angeles, CA

(2) University of California Berkeley, Space Sciences Laboratory, Berkeley, CA

Contact: egrimes@igpp.ucla.edu

Getting Started

https://github.com/spedas/pyspedas
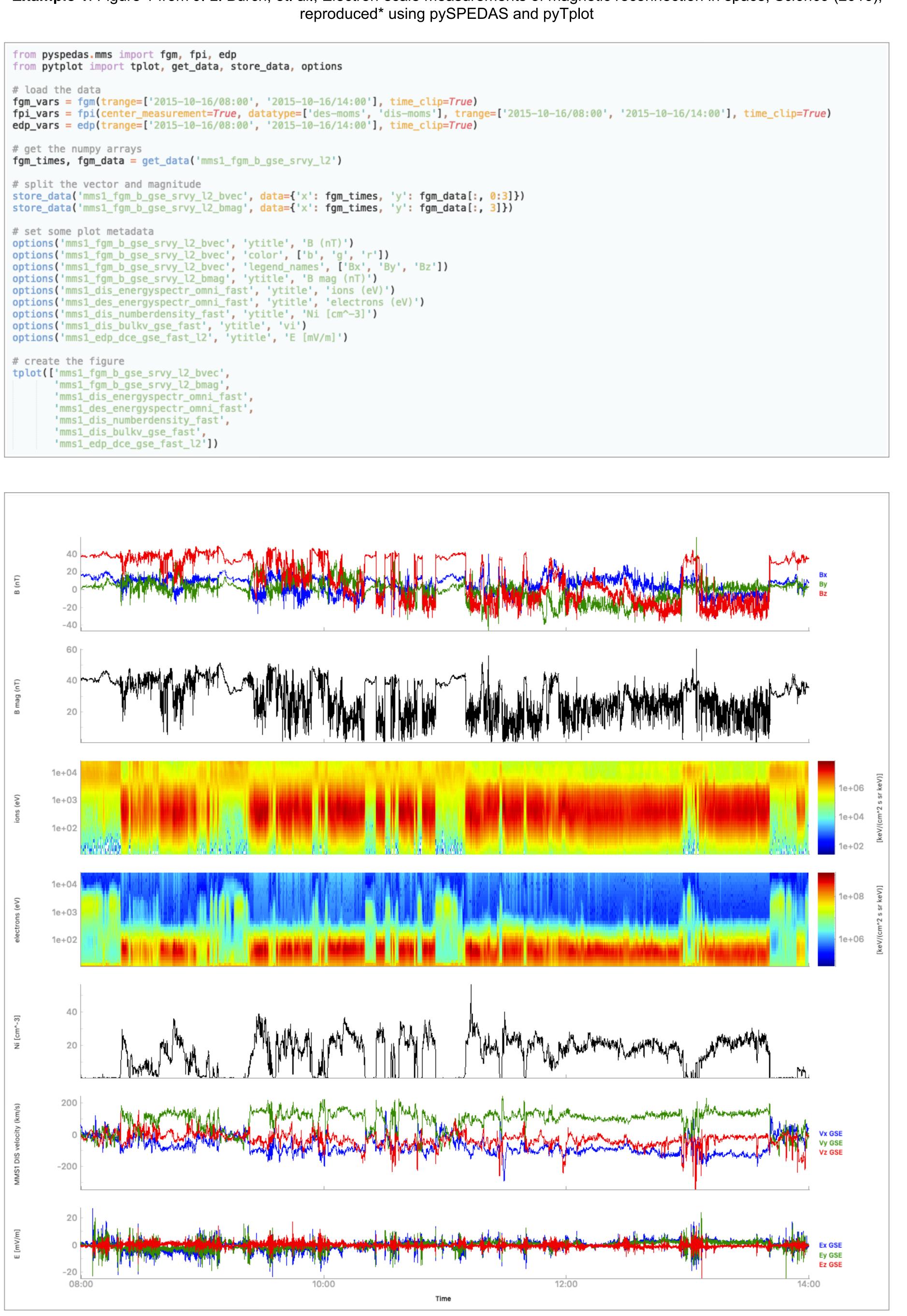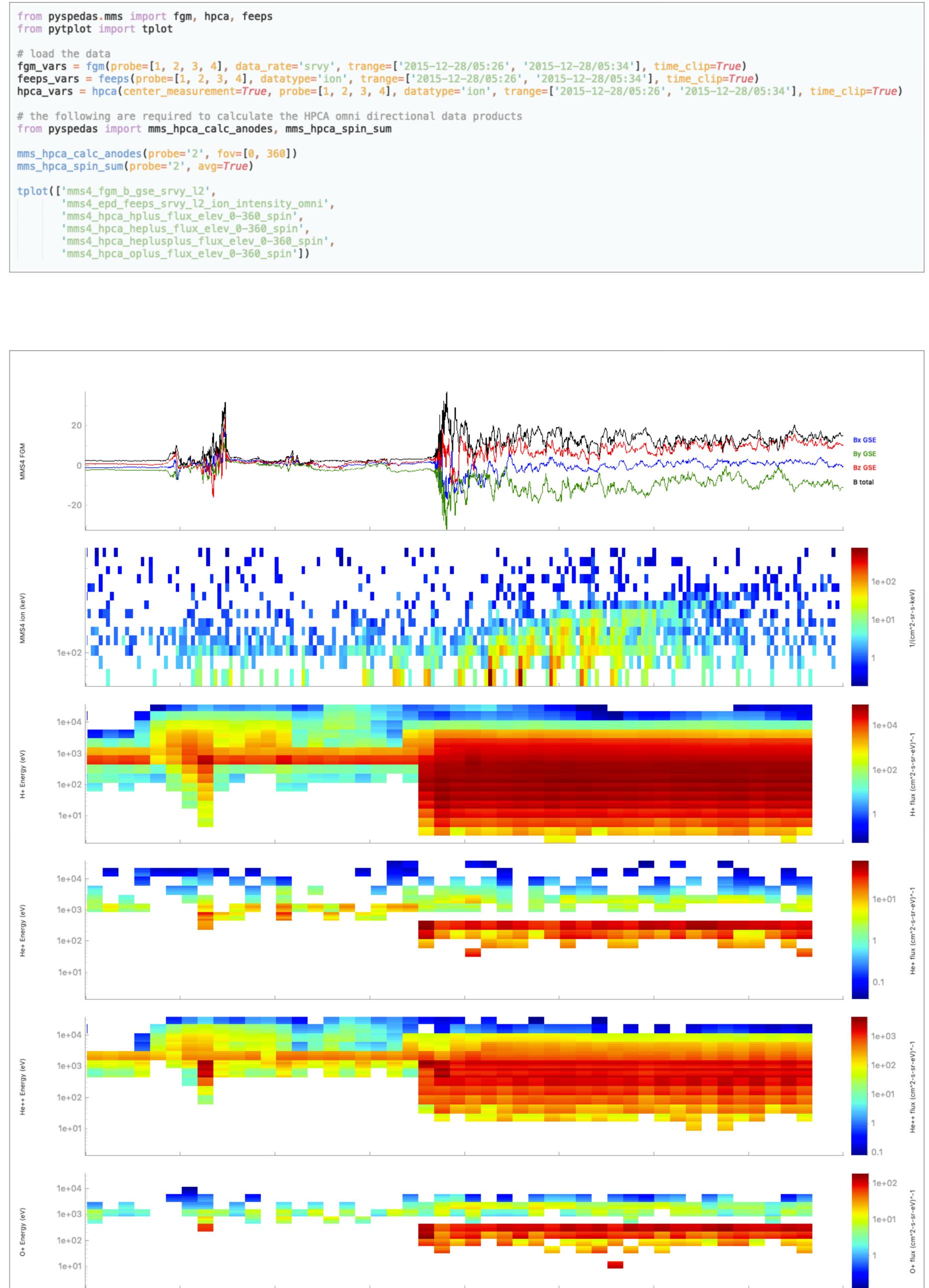
Python 3.5+ is required

To install pySPEDAS:

pip install pyspedas
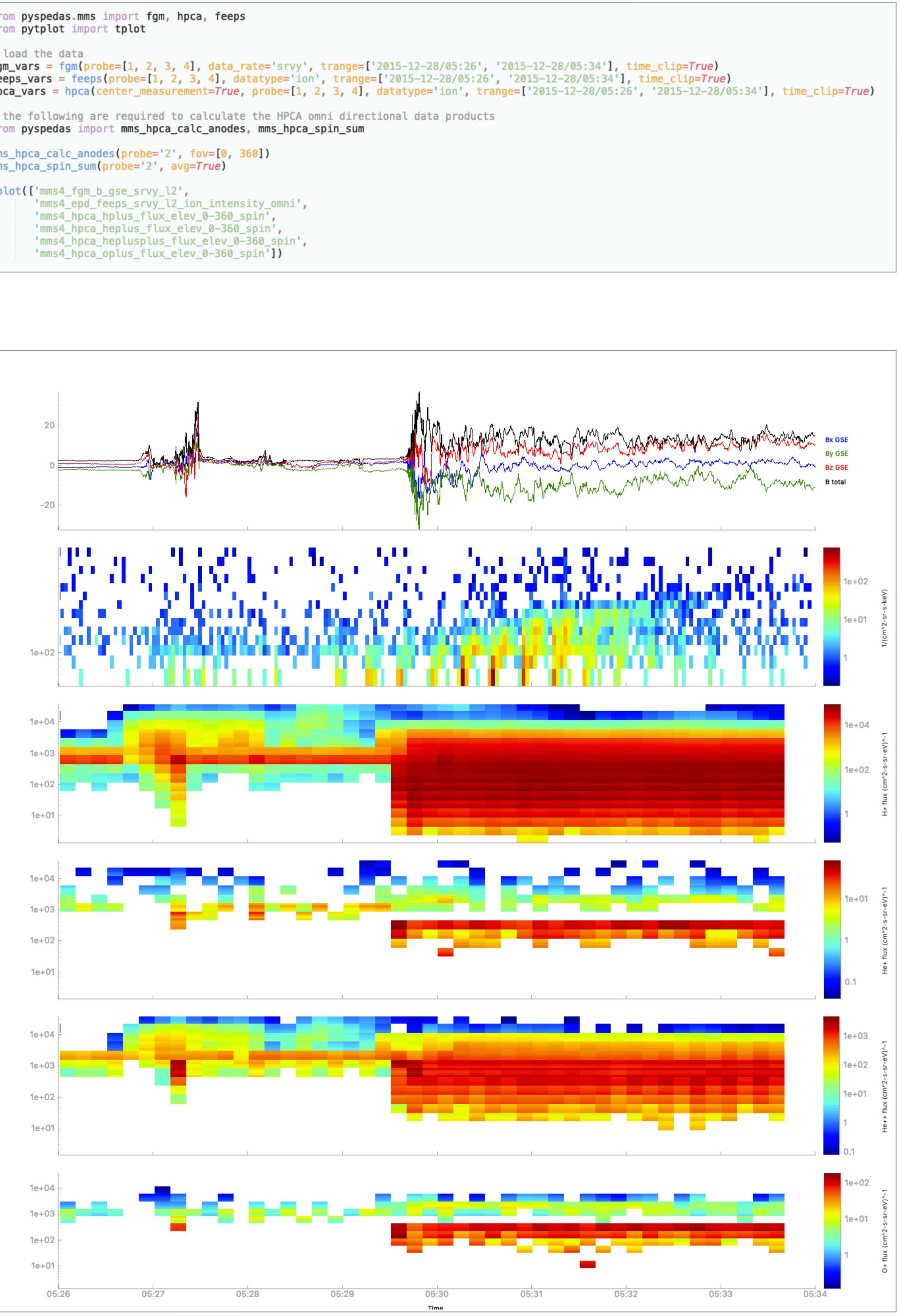
or

conda install -c spedas pyspedas

Abstract

SPEDAS (Space Physics Environment Data Analysis Software) is a framework, written in IDL, to support loading, plotting, analysis, and integration of data from a number of space- and ground-based observatories, including THEMIS, MMS, GOES, ERG, IUGONET, and most data sets archived at CDAWeb. While powerful, IDL has numerous limitations, including the high cost of licensing, limited support and issues created by a single namespace. Due to these limitations and the increasing popularity of the Python programming language, we are collaborating with several missions to bring their data products into the Python environment through a project called pySPEDAS. pySPEDAS currently supports loading data from the MMS, THEMIS, MAVEN and GOES missions, as well as several ground-based observatories and includes some basic tools for analyzing these data. For quality assurance, we're validating that the data products match those loaded with IDL and building a suite of unit and regression tests, which are automatically ran on every commit. In this poster, we'll discuss getting started with pySPEDAS, show several examples and discuss future plans.
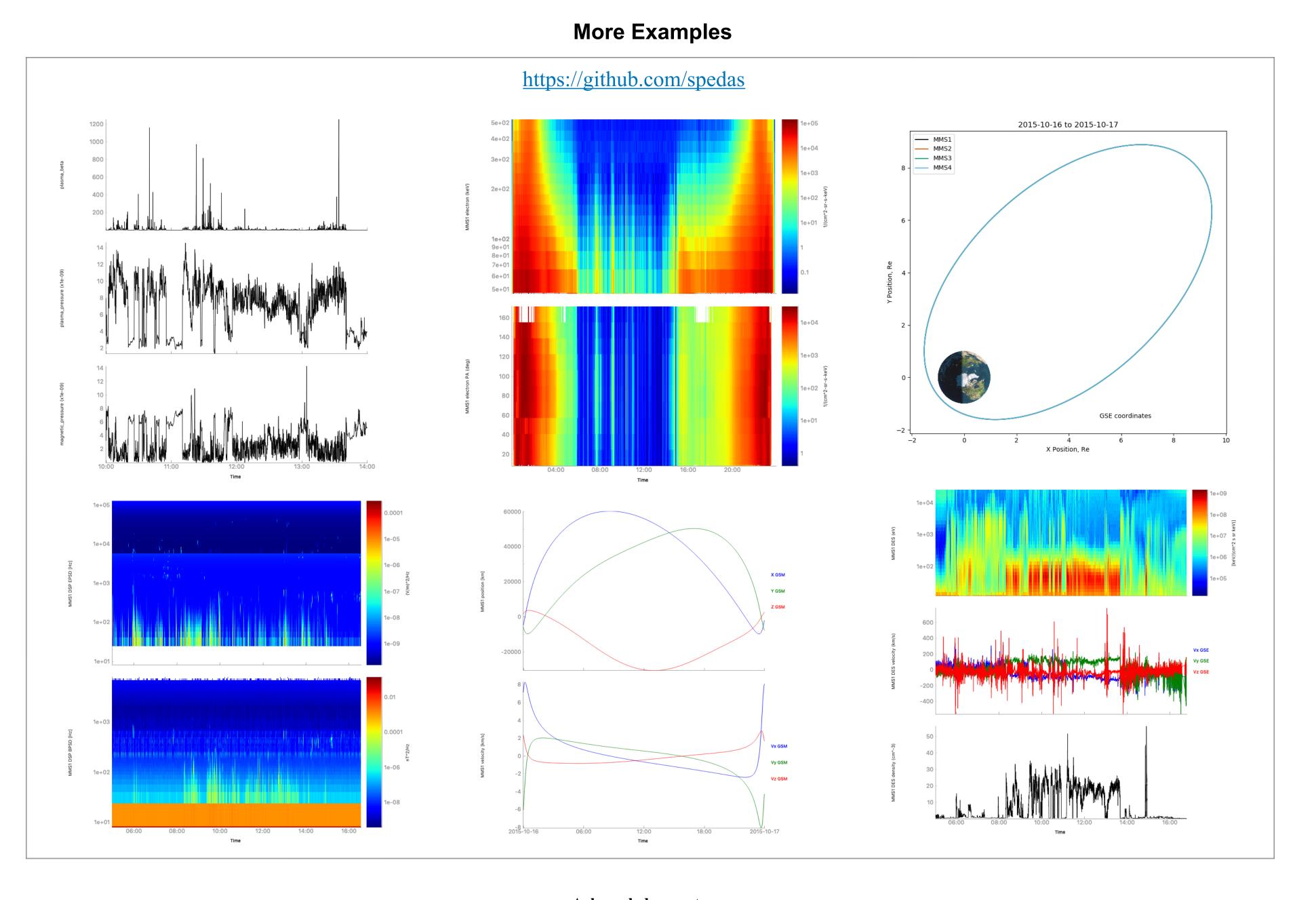
Future Plans

Load routine improvements and templates
New load routines (e.g., Parker Solar Probe, ARASE, etc)
Support for coordinate system transformations
Visualization of plasma distribution functions
Improved quality assurance
Improved documentation
Many more examples

Examples

Example 1: Figure 1 from J. L. Burch, et. al., Electron-scale measurements of magnetic reconnection in space, *Science* (2016), reproduced* using pySPEDAS and PyTplot

```python
from pyspedas.mms import fgm, fpi, edp
from pytplot import tplot, get_data, store_data, options

# load the data
fgm_vars = fgm(trange=['2015-10-16/08:00', '2015-10-16/14:00'], time_clip=True)
fpi_vars = fpi(center_measurements=True, datatype=['des-moms', 'dis-moms'], trange=['2015-10-16/08:00', '2015-10-16/14:00'], time_clip=True)
edp_vars = edp(trange=['2015-10-16/08:00', '2015-10-16/14:00'], time_clip=True)

# get the numpy arrays
fgm_times, fgm_data = get_data('mms1_fgm_b_gse_srvy_l2')

# split the vector and magnitude
store_data('mms1_fgm_b_gse_srvy_l2_bvec', data={'x': fgm_times, 'y': fgm_data[:, 0:3]})
store_data('mms1_fgm_b_gse_srvy_l2_bmag', data={'x': fgm_times, 'y': fgm_data[:, 3]})

# set some plot metadata
options('mms1_fgm_b_gse_srvy_l2_bvec', 'ytitle', 'B (nT)')
options('mms1_fgm_b_gse_srvy_l2_bvec', 'color', ['b', 'g', 'r'])
options('mms1_fgm_b_gse_srvy_l2_bvec', 'legend_names', ['Bx', 'By', 'Bz'])
options('mms1_fgm_b_gse_srvy_l2_bmag', 'ytitle', 'B mag (nT)')
options('mms1_dis_energyspectr_omni_fast', 'ytitle', 'ions (eV)')
options('mms1_des_energyspectr_omni_fast', 'ytitle', 'electrons (eV)')
options('mms1_dis_numberdensity_fast', 'ytitle', 'Ni (cm^-3)')
options('mms1_dis_bulkv_gse_fast', 'ytitle', 'vi')
options('mms1_edp_dce_gse_fast_l2', 'ytitle', 'E (mV/m)')

# create the figure
tplot(['mms1_fgm_b_gse_srvy_l2_bvec',
       'mms1_fgm_b_gse_srvy_l2_bmag',
       'mms1_dis_energyspectr_omni_fast',
       'mms1_des_energyspectr_omni_fast',
       'mms1_dis_numberdensity_fast',
       'mms1_dis_bulkv_gse_fast',
       'mms1_edp_dce_gse_fast_l2'])
```


Example 2: Extended Data Figure 1 from D. L. Turner, et. al., Autogenous and efficient acceleration of energetic ions upstream of Earth's bow shock, *Nature* (2018), reproduced* using pySPEDAS and PyTplot

```python
from pyspedas.mms import fgm, hpca, feeps
from pytplot import tplot

# load the data
fgm_vars = fgm(probe=[1, 2, 3, 4], data_rate='srvy', trange=['2015-12-28/05:26', '2015-12-28/05:34'], time_clip=True)
feeps_vars = feeps(probe=[1, 2, 3, 4], datatype='ion', trange=['2015-12-28/05:26', '2015-12-28/05:34'], time_clip=True)
hpca_vars = hpca(center_measurements=True, probe=[1, 2, 3, 4], datatype='ion', trange=['2015-12-28/05:26', '2015-12-28/05:34'], time_clip=True)

# the following are required to calculate the HPCA omni directional data products
from pyspedas import mms_hpca_calc_anodes, mms_hpca_spin_sum

mms_hpca_calc_anodes(probe='2', fov=[0, 360])
mms_hpca_spin_sum(probe='2', avg=True)

tplot(['mms4_fgm_b_gse_srvy_l2',
       'mms4_epd_feeps_srvy_l2_ion_intensity_omni',
       'mms4_hpca_hplus_flux_elev_0-360_spin',
       'mms4_hpca_heplus_flux_elev_0-360_spin',
       'mms4_hpca_heplusplus_flux_elev_0-360_spin',
       'mms4_hpca_oplus_flux_elev_0-360_spin'])
```


Example 3: MMS curlometer calculations with pySPEDAS and pyTplot

```python
from pyspedas.mms import fgm, curlometer
from pytplot import tplot

# load the FGM and position data in GSE coordinates for all 4 spacecraft
fgm_vars = fgm(probe=[1, 2, 3, 4], trange=['2015-10-30/05:15:45', '2015-10-30/05:15:48'], data_rate='brst', time_clip=True, varformat='*_gse_*')

positions = ['mms1_fgm_r_gse_brst_l2', 'mms2_fgm_r_gse_brst_l2', 'mms3_fgm_r_gse_brst_l2', 'mms4_fgm_r_gse_brst_l2']
fields = ['mms1_fgm_b_gse_brst_l2', 'mms2_fgm_b_gse_brst_l2', 'mms3_fgm_b_gse_brst_l2', 'mms4_fgm_b_gse_brst_l2']

# call pyspedas.mms.curlometer to do the calculations
curlometer_vars = curlometer(fields=fields, positions=positions)

tplot(['jtotal', 'jperp', 'jpar'])
```


More Examples

https://github.com/spedas

Notes

* To keep the examples short, there are some minor differences between Example 1 and Example 2 and the figures in the papers; e.g., in Example 1, the FPI units are different and the spacecraft position labels are missing; in Example 2, the energy spectra are only plotted for 1 spacecraft instead of combining the data from all spacecraft

** For more information on PyTplot, please see:

https://github.com/MAVENSDC/PyTplot