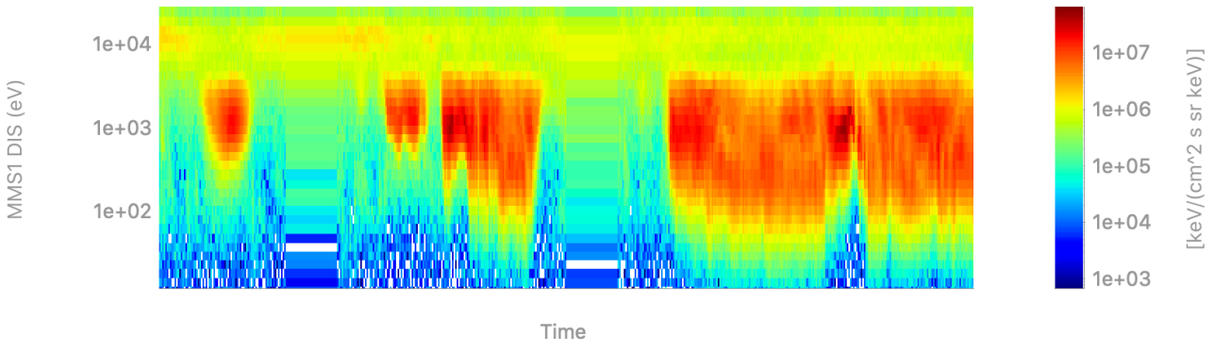
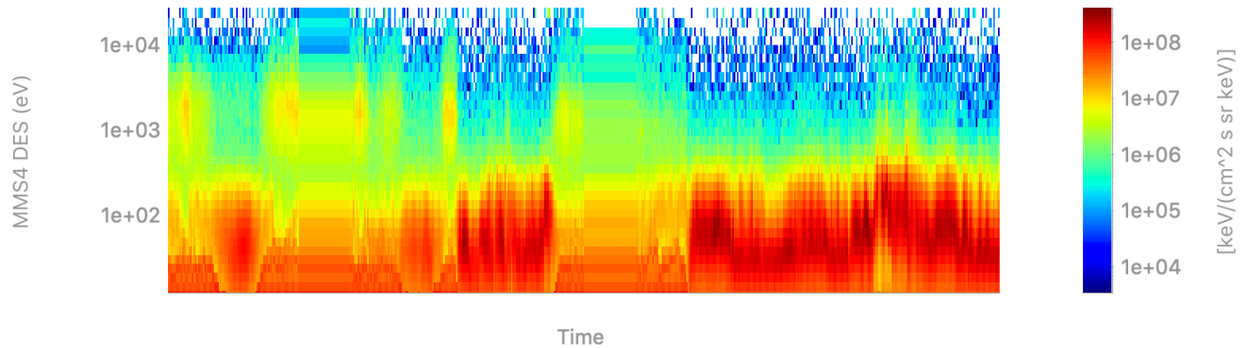
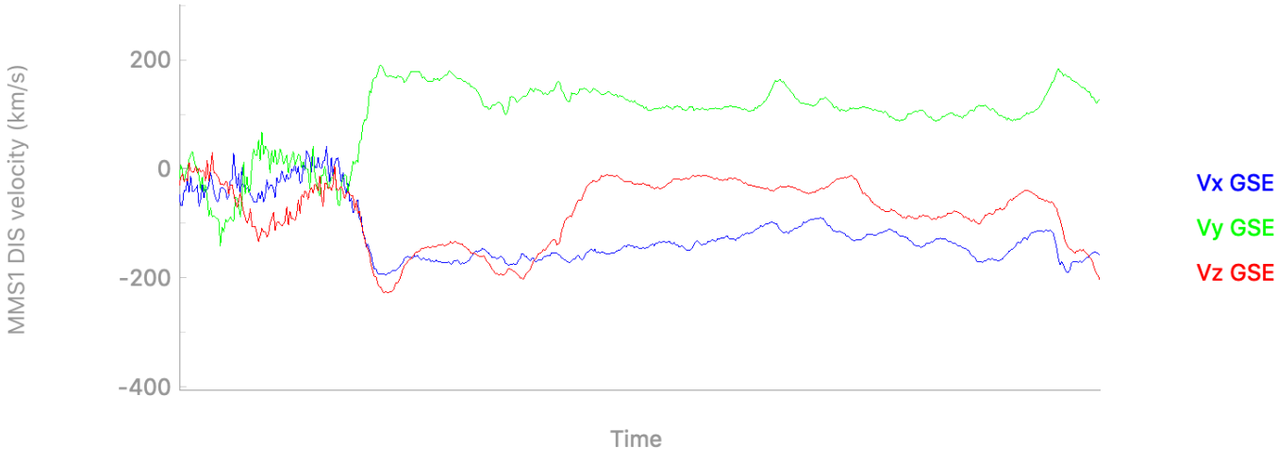




pySPEDAS: Space Physics Environment Data Analysis Software in Python

Eric Grimes
egrimes@igpp.ucla.edu



MMS plug-in for IDL SPEDAS

- 12 load routines
- ~40k lines of code
- >500 unit/regression tests
- 74 crib sheets
- Very powerful, but limited to those of us with IDL licenses

Why pySPEDAS?

- Python is free, easy to learn, and is supported by a very large community
- Python has a robust library ecosystem
- Python allows multiple namespaces
- Increasingly being taught as an “Intro to Programming” course

Allows for the MMS data to be used in new ways!

Other ways to access MMS data in Python

- Matthew Argall's pymms; provides access to team-only data as well as L2
- ai.cdas (unofficial CDAWeb library)
- David Stansby's heliopy
- Bob Weigel's HAPI client (via CDAWeb)
- manually via pyCDF, pysatCDF, cdflib, etc
- import via the IDL-Python bridge

...

Again, why pySPEDAS?

- The goal is to provide the same **science quality data products** found in the IDL MMS plug-in, e.g.,
    - De-flagged FGM data by default (i.e., data flagged by the team are NaN'd)
    - Omni-directional FEEPS/EIS/HPCA data products
        - For FEEPS, this includes sun-light contamination removal, flat-field corrections, bad eye and energy channel removal, etc
        - FEEPS/EIS pitch angle distributions
        - FPI lossy compression, error bars
        - etc.
- and make these easy to access (i.e., using the same reasonable defaults we use in the IDL plug-in, and a similar interface to load the data)
- and provide the same high level of quality assurance (automated tests, including tests that verify the data products match those loaded in IDL SPEDAS)
- and provide crib sheets, tutorials, webinars to spread the knowledge

Note: we're using a permissive license (MIT), so these features can be modified and included in the projects on the previous slide

Getting Started

https://github.com/spedas/pyspedas

Python 3.5+ is required.

I suggest downloading the ZIP rather than using PyPI for the time being

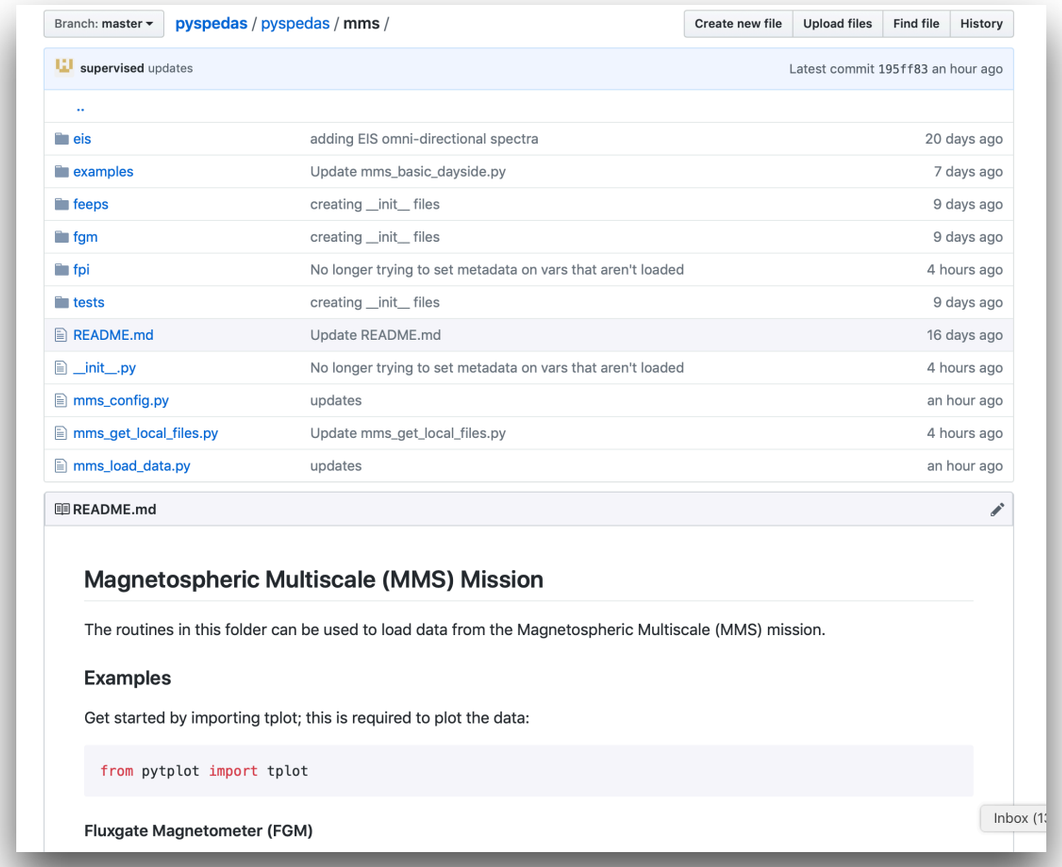
Config settings (e.g., local_data_dir) are set in the hash table stored in mms_config.py

Supported MMS load routines for L2:

FGM, SCM, EDP, DSP, EDI, FPI
HPCA, FEEPS, EIS, ASPOC, MEC

Supported keywords:

probe, trange, data_rate
datatype, suffix, get_support_data



Currently only supports L2 data

Getting Started

Required Python Packages

pytplot
cdflib
pydivide
pyqtgraph
numpy
requests
dateutil

“pip install -r requirements.txt”
in the pyspedas folder should install everything needed.

Loading MMS Data

Text editor/IDE

```
from pyspedas.mms import mms_load_fgm, mms_load_eis

mms_load_fgm(trange=['2015-10-16', '2015-10-17'])

# the load routines accept all of the standard keywords
mms_load_eis(probe='4', data_rate='brst', datatype=['extof', 'phxtof'], trange=['2015-10-16/13:05', '2015-10-16/13:10'])
```

Python 3 interpreter

```
>>> from pyspedas.mms import mms_load_fgm, mms_load_eis
>>>
>>> mms_load_fgm(trange=['2015-10-16', '2015-10-17'])
20-Feb-19 15:53:44: Loading pydata/mms/mms1/fgm/srvy/l2/2015/10/mms1_fgm_srvy_l2_20151016_v4.18.0.cdf
Cannot find x axis.
No attribute named DEPEND_TIME or DEPEND_0 in                    variable Epoch
Cannot find x axis.
No attribute named DEPEND_TIME or DEPEND_0 in                    variable Epoch_state
The lengths of x and y do not match!
20-Feb-19 15:57:40: Loaded variables:
mms1_fgm_b_gse_srvy_l2
mms1_fgm_b_gsm_srvy_l2
mms1_fgm_b_dmpa_srvy_l2
mms1_fgm_b_bcs_srvy_l2
mms1_fgm_flag_srvy_l2
mms1_fgm_r_gse_srvy_l2
mms1_fgm_r_gsm_srvy_l2
mms1_fgm_hirange_srvy_l2
mms1_fgm_bdeltahalf_srvy_l2
```

Note: load routine defaults are the same as in the IDL plug-in

Listing Loaded MMS Data

```
from pytplot import tplot_names

tplot_names()
```

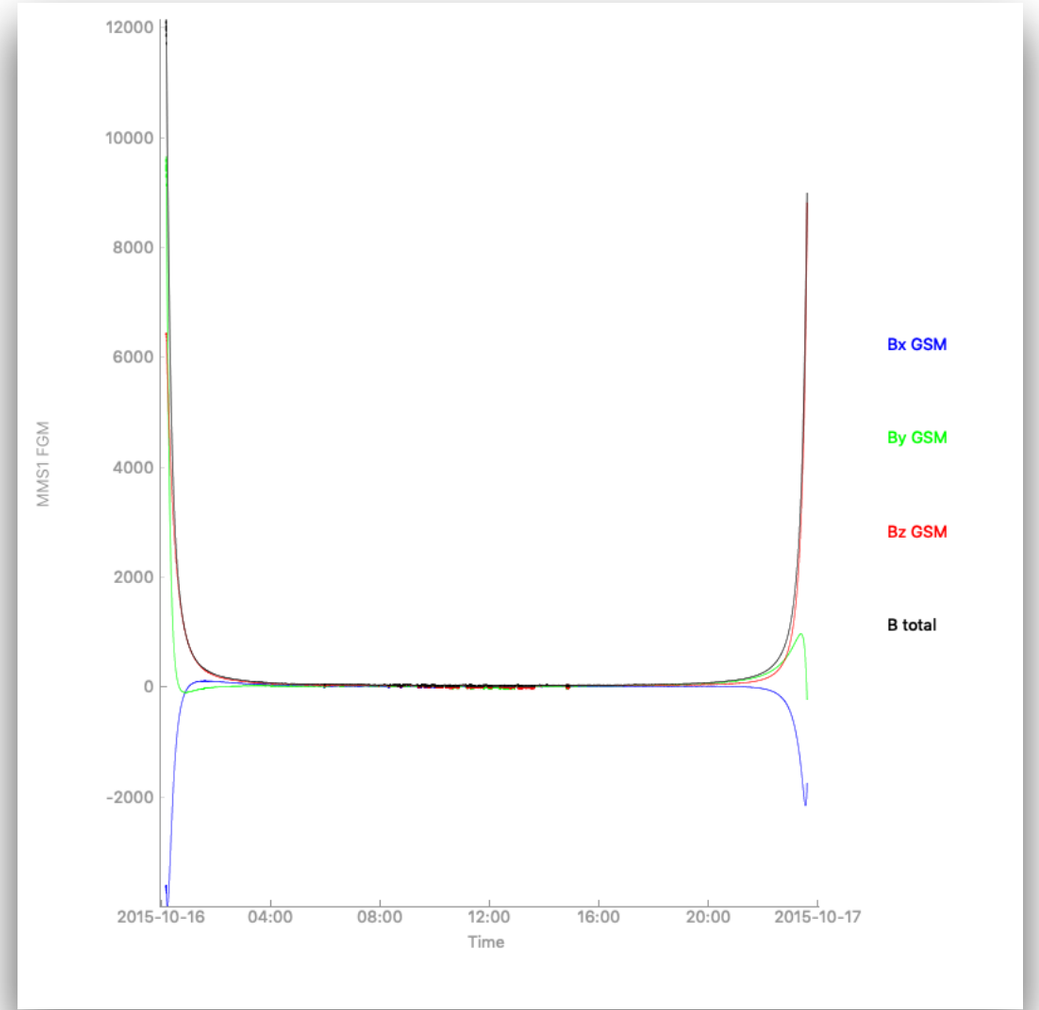
```
>>> from pytplot import tplot_names
>>>
>>> tplot_names()
0 : mms1_fgm_b_gse_srvy_l2
1 : mms1_fgm_b_gsm_srvy_l2
2 : mms1_fgm_b_dmpa_srvy_l2
3 : mms1_fgm_b_bcs_srvy_l2
4 : mms1_fgm_flag_srvy_l2
5 : mms1_fgm_r_gse_srvy_l2
6 : mms1_fgm_r_gsm_srvy_l2
7 : mms1_fgm_hirange_srvy_l2
8 : mms1_fgm_bdeltahalf_srvy_l2
9 : mms1_fgm_stemp_srvy_l2
10 : mms1_fgm_etemp_srvy_l2
11 : mms1_fgm_mode_srvy_l2
12 : mms4_epd_eis_brst_extof_proton_P3_counts_t0
13 : mms4_epd_eis_brst_extof_proton_P3_counts_t1
14 : mms4_epd_eis_brst_extof_proton_P3_counts_t2
15 : mms4_epd_eis_brst_extof_proton_P3_counts_t3
16 : mms4_epd_eis_brst_extof_proton_P3_counts_t4
17 : mms4_epd_eis_brst_extof_proton_P3_counts_t5
18 : mms4_epd_eis_brst_extof_proton_P3_cps_t0
19 : mms4_epd_eis_brst_extof_proton_P3_cps_t1
20 : mms4_epd_eis_brst_extof_proton_P3_cps_t2
21 : mms4_epd_eis_brst_extof_proton_P3_cps_t3
22 : mms4_epd_eis_brst_extof_proton_P3_cps_t4
23 : mms4_epd_eis_brst_extof_proton_P3_cps_t5
24 : mms4_epd_eis_brst_extof_proton_P3_flux_t0
25 : mms4_epd_eis_brst_extof_proton_P3_flux_t1
26 : mms4_epd_eis_brst_extof_proton_P3_flux_t2
27 : mms4_epd_eis_brst_extof_proton_P3_flux_t3
28 : mms4_epd_eis_brst_extof_proton_P3_flux_t4
29 : mms4_epd_eis_brst_extof_proton_P3_flux_t5
30 : mms4_epd_eis_brst_extof_alpha_P3_counts_t0
31 : mms4_epd_eis_brst_extof_alpha_P3_counts_t1
32 : mms4_epd_eis_brst_extof_alpha_P3_counts_t2
```

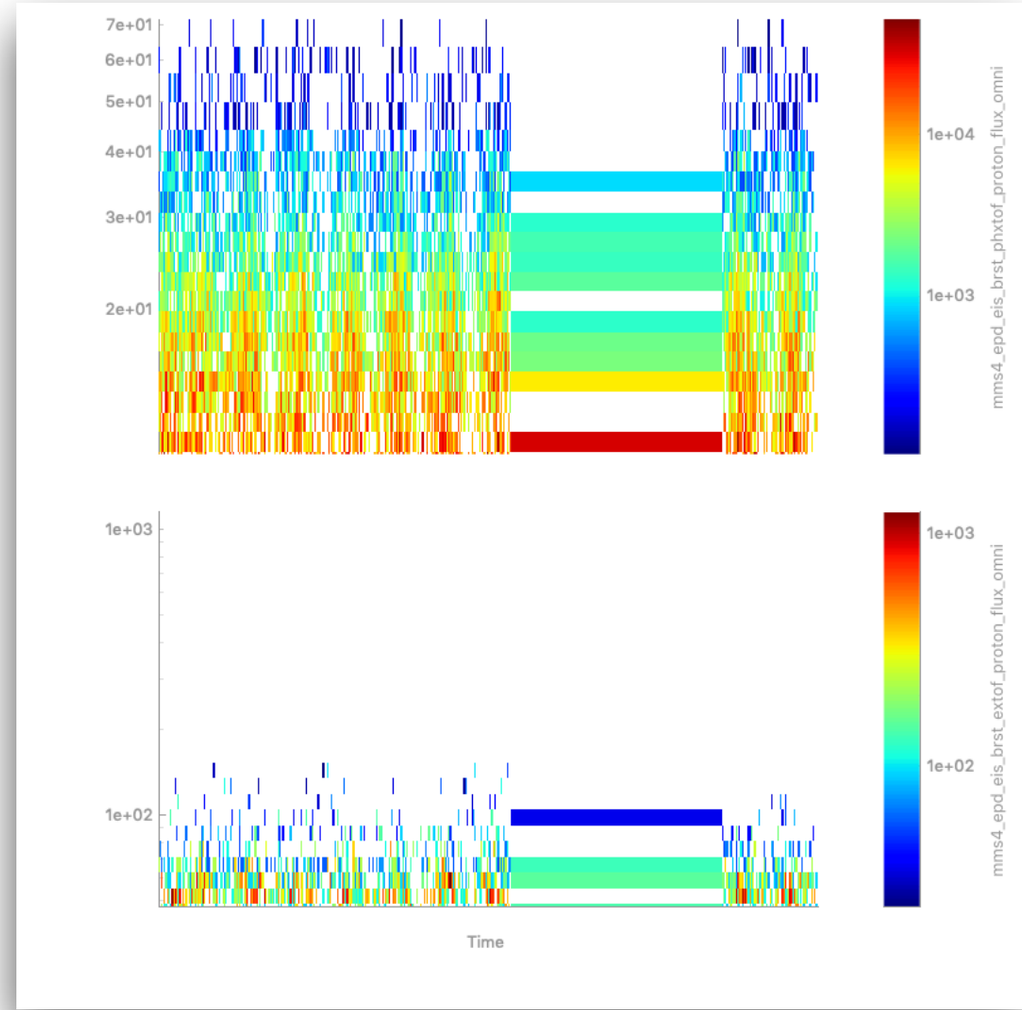
Plotting MMS Data

```
from pytplot import tplot

tplot('mms1_fgm_b_gsm_srvy_l2')

tplot(['mms4_epd_eis_brst_phxtof_proton_flux_omni', 'mms4_epd_eis_brst_extof_proton_flux_omni'])
```



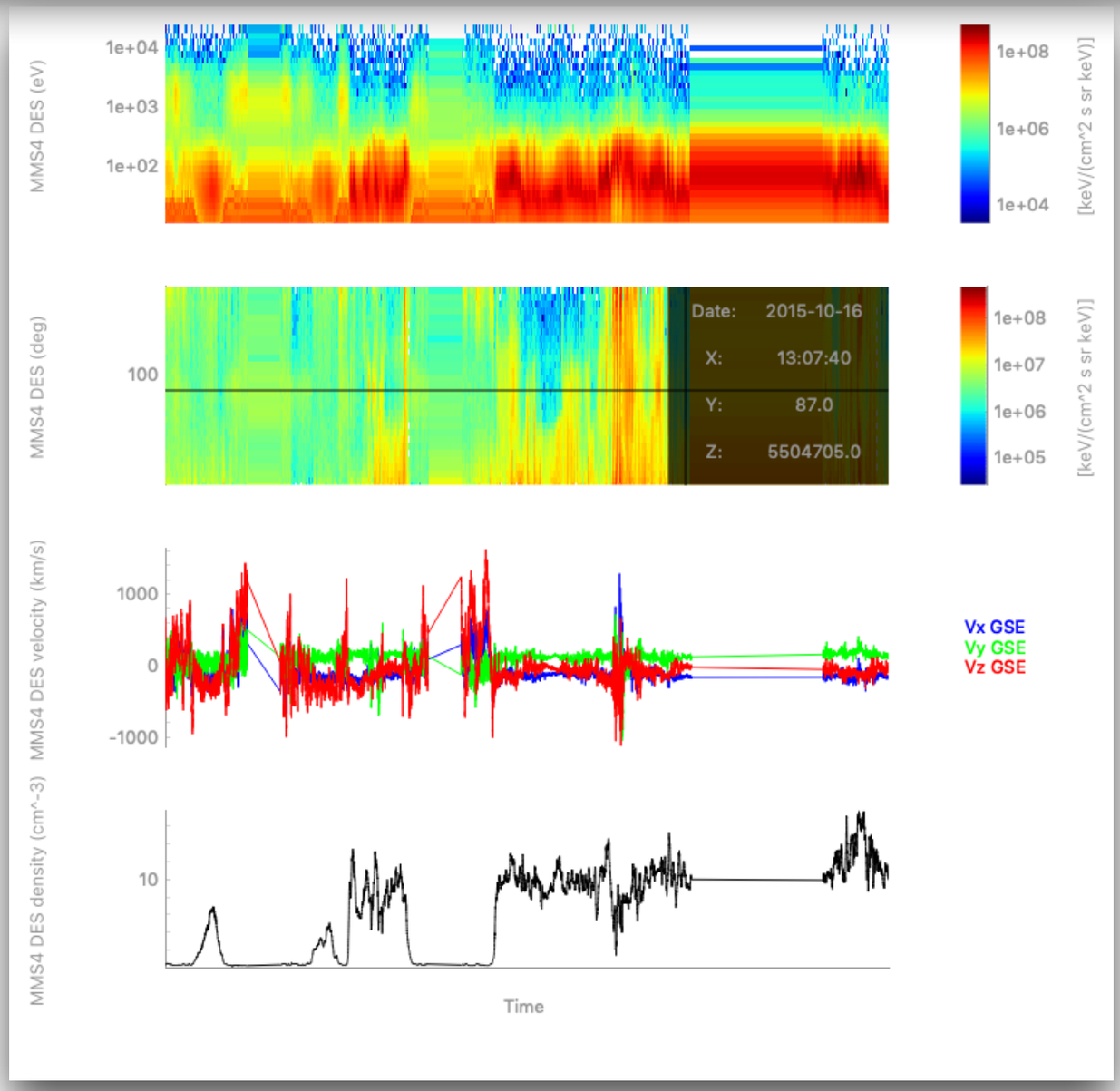


Plotting MMS Data

```
from pyspedas.mms import mms_load_fpi

mms_load_fpi(probe='4', data_rate='brst', datatype='des-moms', trange=['2015-10-16/13:00', '2015-10-16/13:10'])

tplot(['mms4_des_energyspectr_omni_brst', 'mms4_des_pitchangdist_miden_brst', 'mms4_des_bulkv_gse_brst', 'mms4_des_numberdensity_brst'])
```



Get to the Data

```python
from pytplot import get_data, store_data

# extract the data from a tplot variable
times, data = get_data('mms1_fgm_b_gsm_srvy_l2')
```

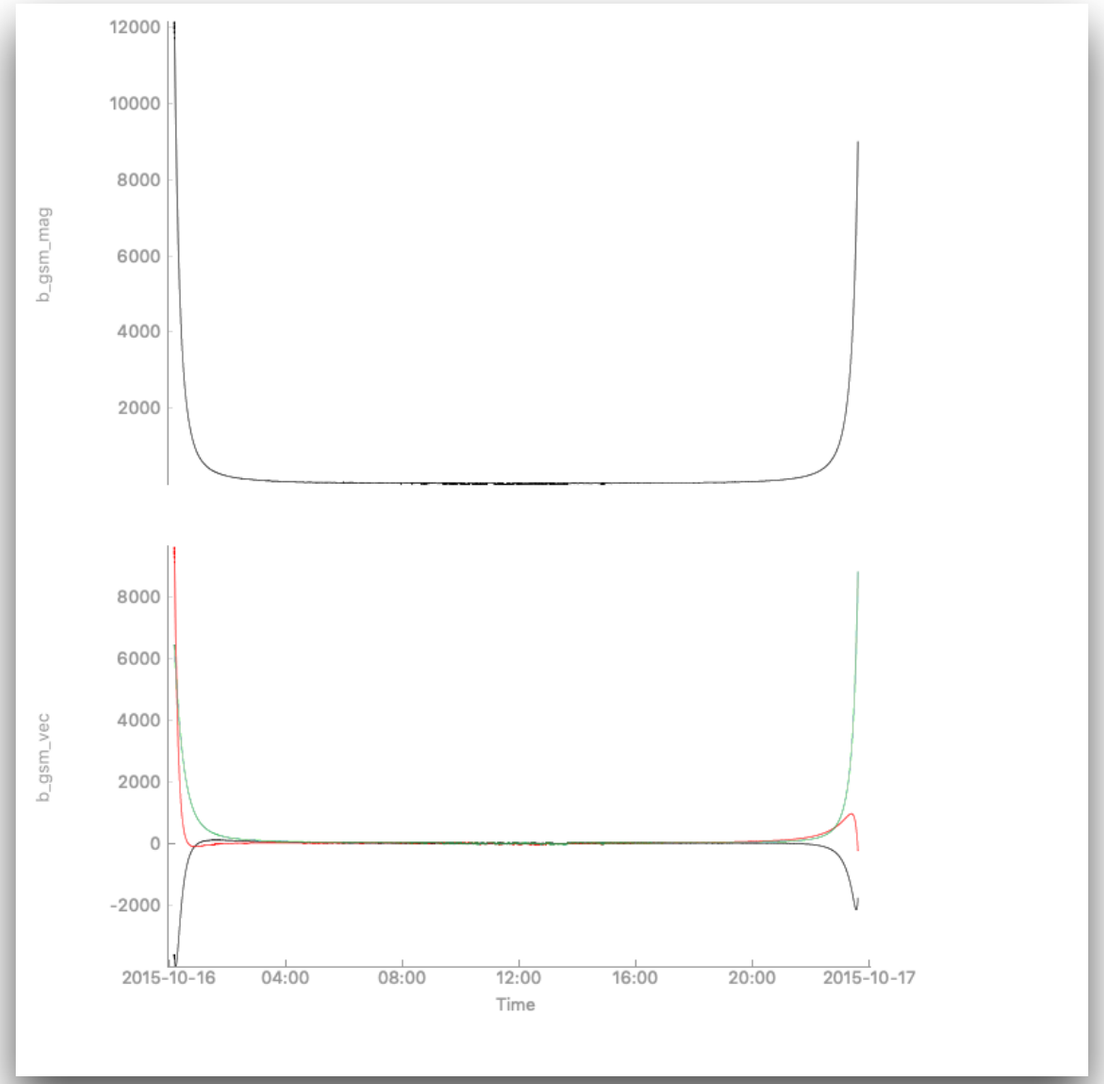
```
>>> from pytplot import get_data, store_data
>>>
>>> times, data = get_data('mms1_fgm_b_gsm_srvy_l2')
>>>
>>> data[10000]
array([-3246.3035,  3114.4595,  4377.299 ,  6276.869 ], dtype=float32)
>>>
>>> times[10000]
1444954863.34657
```

Create tplot Variables

```
store_data('b_gsm_vec', data={'x': times, 'y': data[:,0:3]}) # B-field vector

store_data('b_gsm_mag', data={'x': times, 'y': data[:,3]}) # B-field magnitude

tplot(['b_gsm_mag', 'b_gsm_vec'])
```

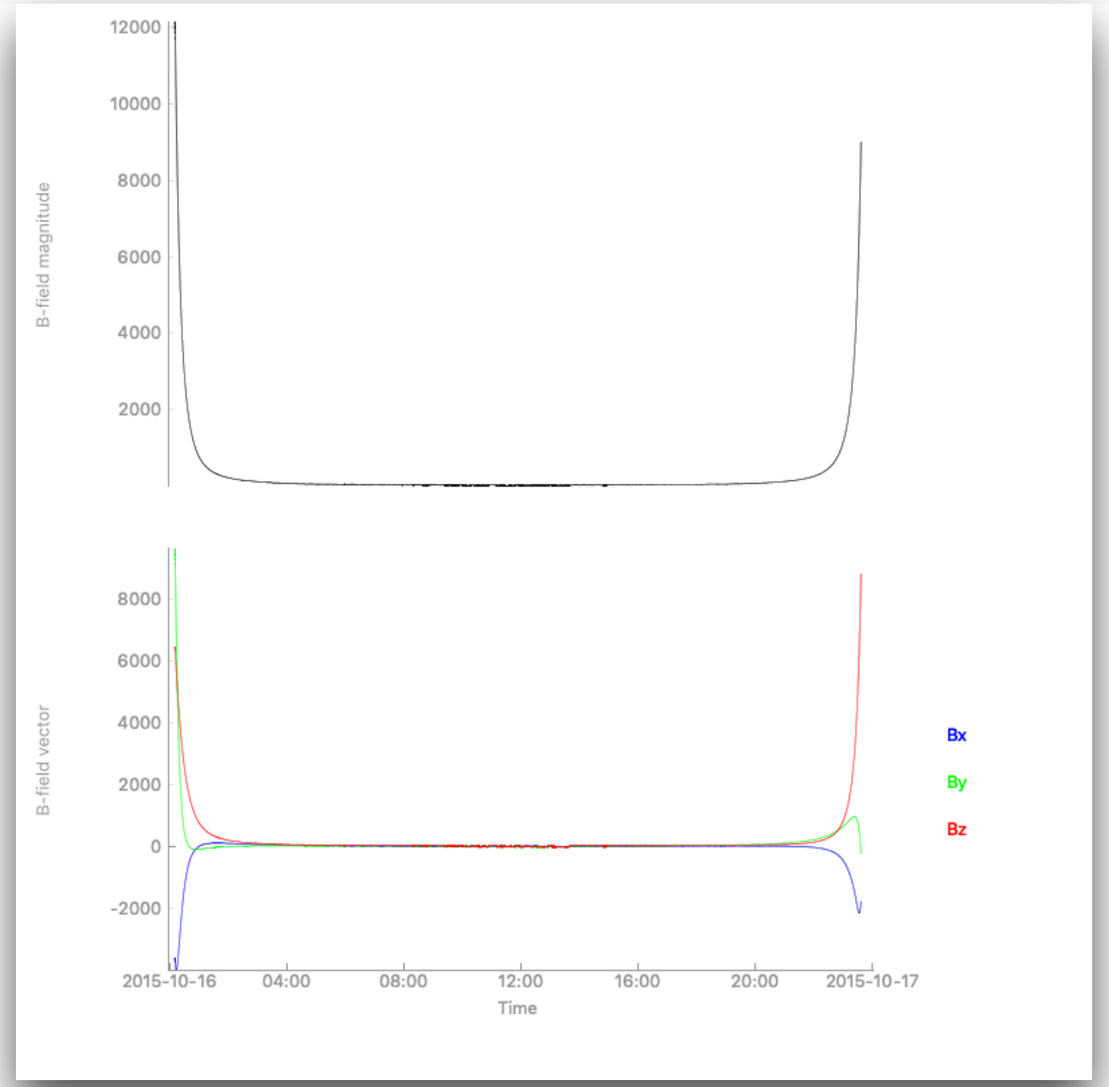


Modify Variable Metadata

```
from pytplot import options

options('b_gsm_mag', 'ytitle', 'B-field magnitude')
options('b_gsm_vec', 'ytitle', 'B-field vector')
options('b_gsm_vec', 'color', ['b', 'g', 'r'])
options('b_gsm_vec', 'legend_names', ['Bx', 'By', 'Bz'])

tplot(['b_gsm_mag', 'b_gsm_vec'])
```



Time Conversions

```
from pyspedas.utilities.time_string import time_string

print(time_string(1444954863.34657))

from pyspedas.utilities.time_double import time_double

print(time_double('2015-10-16 00:21:03.346570'))
```

```
>>> from pyspedas.utilities.time_string import time_string
>>>
>>> print(time_string(1444954863.34657))
2015-10-16 00:21:03.346570
>>>
>>> from pyspedas.utilities.time_double import time_double
>>>
>>> print(time_double('2015-10-16 00:21:03.346570'))
1444954863.34657
```

Saving the Data

```
from pyspedas.utilities.tplot2ascii import tplot2ascii

tplot2ascii(['b_gsm_mag', 'b_gsm_vec'])
```

```
>>> from pyspedas.utilities.tplot2ascii import tplot2ascii
>>>
>>> tplot2ascii(['b_gsm_mag', 'b_gsm_vec'])
>>>
```

b_gsm_vec.txt

```
1444953613.330852,8580.49,7339.21,6250.034
1444953613.455853,8861.722,7441.703,6211.8975
1444953613.580855,9138.241,7578.622,6165.077
1444953613.705856,9406.668,7749.714,6109.84
1444953613.830858,9665.108,7954.0244,6046.553
1444953613.955859,9903.166,8192.649,5975.494
1444953614.080861,10139.0,8459.444,5897.341
1444953614.205863,10274.0205,8770.482,5813.036
1444953614.330864,10026.204,9148.692,5731.218
1444953614.455866,9709.858,9523.264,5654.0244
1444953614.580867,9388.68,9884.538,5580.668
1444953614.705869,9048.596,10233.249,5511.6763
1444953614.830871,8695.527,10569.052,5446.823
1444953614.955872,8329.004,10889.934,5386.6587
1444953615.080874,7949.6226,11195.827,5331.161
1444953615.205875,7559.18,11480.314,5281.9287
1444953615.330877,7232.053,11415.872,5321.9487
1444953615.455878,6963.2007,11188.5,5401.8604
1444953615.58088,6720.637,10968.896,5478.021
1444953615.705882,6514.079,10726.67,5557.9785
1444953615.830883,6341.362,10475.321,5638.278
1444953615.955885,6203.628,10216.211,5718.509
1444953616.080886,6101.144,9952.583,5797.8105
1444953616.205888,6033.7954,9687.856,5875.298
1444953616.330889,6001.3047,9425.165,5950.172
1444953616.455891,6003.0757,9167.679,6021.643
1444953616.580893,6038.1587,8918.608,6088.9336
1444953616.705894,6105.329,8681.051,6151.3105
1444953616.830896,6203.0513,8458.026,6208.08
1444953616.955897,6329.996,8251.896,6258.7114
1444953617.080899,6484.0366,8065.5503,6302.5796
```

b_gsm_mag.txt

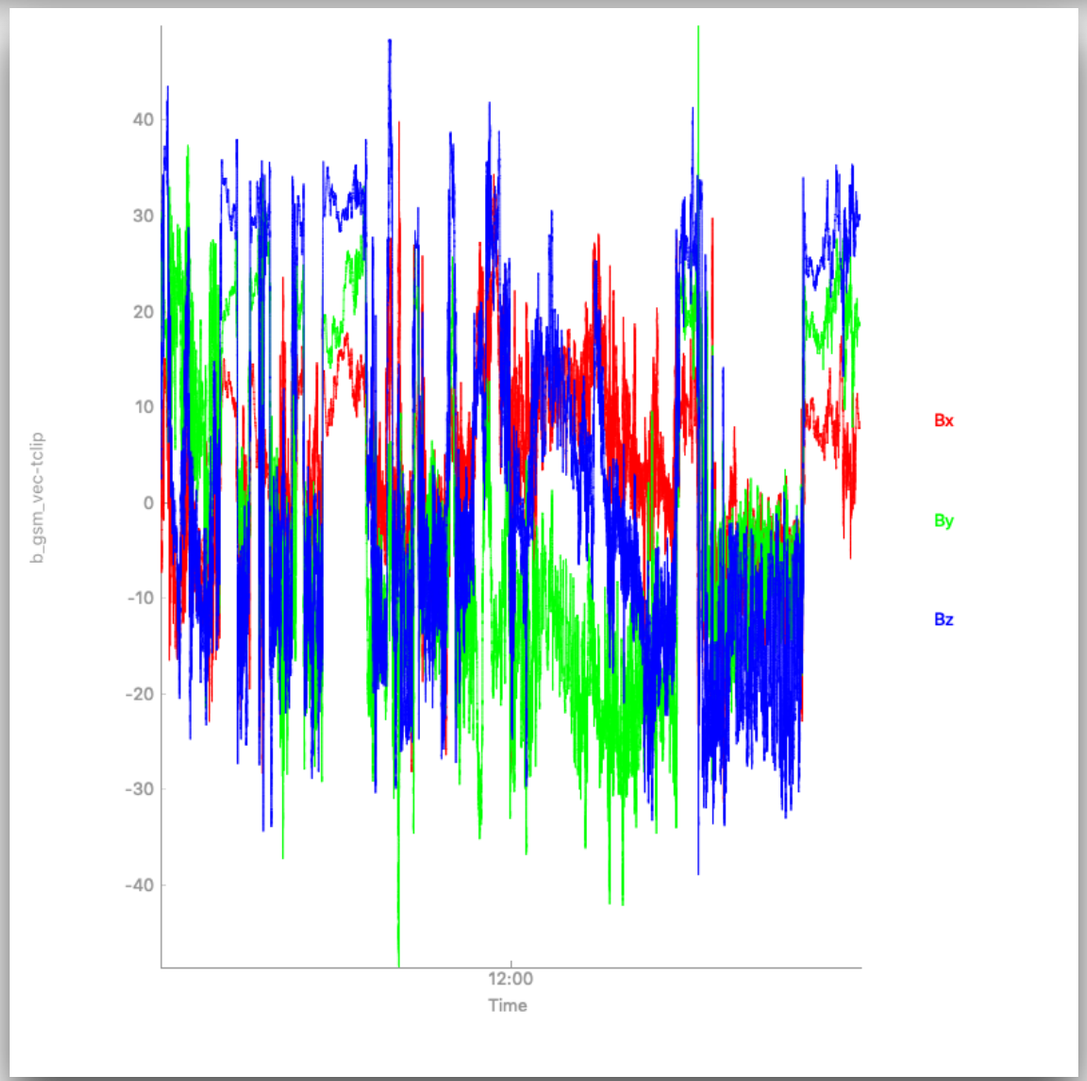
```
1444953613.330852,12905.493
1444953613.455853,13133.802
1444953613.580855,13377.262
1444953613.705856,13633.547
1444953613.830858,13901.138
1444953613.955859,14173.875
1444953614.080861,14461.679
1444953614.205863,14706.062
1444953614.330864,14733.303
1444953614.455866,14728.948
1444953614.580867,14730.758
1444953614.705869,14730.072
1444953614.830871,14730.409
1444953614.955872,14730.208
1444953615.080874,14729.709
1444953615.205875,14725.405
1444953615.330877,14524.045
1444953615.455878,14242.5
1444953615.58088,13981.857
1444953615.705882,13725.37
1444953615.830883,13480.928
1444953615.955885,13249.805
1444953616.080886,13034.281
1444953616.205888,12836.68
1444953616.330889,12659.145
1444953616.455891,12503.736
1444953616.580893,12372.391
1444953616.705894,12266.796
1444953616.830896,12188.368
1444953616.955897,12138.126
1444953617.080899,12116.861
1444953617.2059,12124.912
```

Time-clip the Data

```
from pyspedas.analysis.time_clip import time_clip

time_clip('b_gsm_vec', '2015-10-16/10:00', '2015-10-16/14:00')

options('b_gsm_vec-tclip', 'color', ['r', 'g', 'b'])
options('b_gsm_vec-tclip', 'legend_names', ['Bx', 'By', 'Bz'])
tplot('b_gsm_vec-tclip')
```

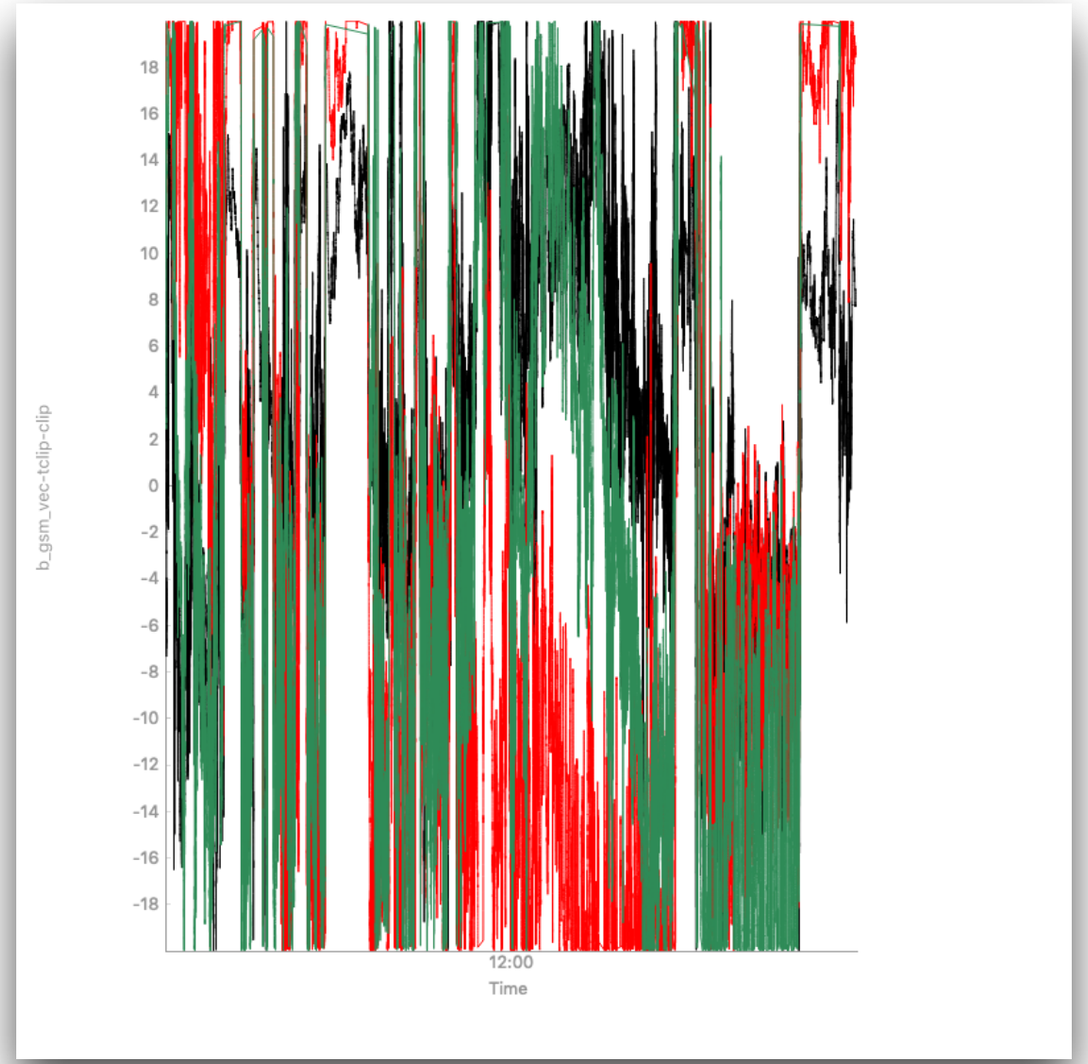


Clip the Y-axis Data

```
from pyspedas.analysis.tclip import tclip

tclip('b_gsm_vec-tclip', -20, 20)

tplot('b_gsm_vec-tclip-clip')
```



Current Status: MMS Data Products

	IDL	Python
All L2 data from CDFs		
EIS omni-directional spectra		
Deflagged FGM data		
Corrected FEEPS omni-directional spectra		
Omni-directional HPCA spectra		
Spin-averaged EIS, FEEPS spectra		
EIS, FEEPS pitch angle distributions		
Combined EIS PHxTOF and ExTOF proton PADs		
EIS PADs with multiple probes		
QL/L1 data from CDFs		

Thank You!

Contributions welcomed/encouraged!

Questions, bugs, comments, concerns: egrimes@igpp.ucla.edu