import matplotlib

from matplotlib import pyplot as plt

import numpy as np

import datetime as dt

from spacepy import pycdf

data\_folder = 'C:\\Users\\david\\Desktop\\spacedata\\PSP\\'

ephem\_file = 'spp\_fld\_l1\_ephem\_spp\_vso\_20191226\_v00.cdf'

psp\_mag\_VSO\_fname1 = 'psp\_fld\_l2\_mag\_VSO\_2019122612\_v01.cdf';

psp\_mag\_VSO\_fname2 = 'psp\_fld\_l2\_mag\_VSO\_2019122618\_v01.cdf';

psp\_mag\_VSO\_fname3 = 'psp\_fld\_l2\_mag\_VSO\_2019122618\_v01.cdf';

ephem\_cdf = pycdf.CDF(data\_folder+ephem\_file)

# note how by asking for all elements explicitly, we get an ndarray, not a pycdf Var

psp\_vfb2\_ephem\_ttags = ephem\_cdf['epoch'][:]

print('Data type of time tags array: ',type(psp\_vfb2\_ephem\_ttags))

print('Data type of each time tag: ',type(psp\_vfb2\_ephem\_ttags[10]))

psp\_vfb2\_VSO\_position = ephem\_cdf['position'][:]

psp\_vfb2\_VSO\_velocity = ephem\_cdf['velocity'][:]

psp\_vfb2\_VSO\_position\_Rv = (1.0/6051.8)\*psp\_vfb2\_VSO\_position[:,:]

#print(psp\_vfb2\_VSO\_position\_Rv[100])

# use linalg.norm to compute magnitude of vector

# note use of axis, so that we get back an array of distances to Venus

psp\_vfb2\_venus\_range\_Rv = np.linalg.norm(psp\_vfb2\_VSO\_position\_Rv,axis=1)

#print(psp\_vfb2\_venus\_range\_Rv[1000])

# set time range for plots

timerange = [dt.datetime(2019, 12, 26,18,00,0),dt.datetime(2019, 12, 26,18,30,0) ]

# arrange panels, and share the x (ie time) axis so all panels have same time range

fig, axes = plt.subplots(nrows=3, ncols=1,sharex=True)

fig.tight\_layout()

# limit to a given time range by using logical indexing for arrays

#ephem\_ttag\_mask = (psp\_vfb2\_ephem\_ttags > timerange[0]) & (psp\_vfb2\_ephem\_ttags <= timerange[1])

#plt.plot(psp\_vfb2\_ephem\_ttags[ttag\_mask],psp\_vfb2\_venus\_range\_Rv[ttag\_mask])

#alternative

# Zoom in by setting axis limits

axes[0].plot(psp\_vfb2\_ephem\_ttags,psp\_vfb2\_venus\_range\_Rv)

axes[0].set\_ylim(0.0, 12.0)

axes[0].set\_xlim(timerange[0],timerange[1])

#axes[0].set\_title('PSP VFB2')

axes[0].set\_ylabel('Venus range ($R\_V$)')

ephem\_cdf.close()

psp\_mag\_122618\_cdf = pycdf.CDF(data\_folder+psp\_mag\_VSO\_fname2)

print(psp\_mag\_122618\_cdf)

psp\_vfb2\_mag\_ttags = psp\_mag\_122618\_cdf['epoch\_mag\_VSO'][:]

psp\_vfb2\_mag\_VSO = psp\_mag\_122618\_cdf['psp\_fld\_l2\_mag\_VSO'][:,:]

print('size/shape of mag data: ',psp\_vfb2\_mag\_VSO.shape )

# construct a logical mask for timetags in time range

mag\_ttag\_mask = (psp\_vfb2\_mag\_ttags > timerange[0]) & (psp\_vfb2\_mag\_ttags <= timerange[1])

print('done mask')

mag\_ttags = psp\_vfb2\_mag\_ttags[mag\_ttag\_mask]

magVSO\_Bx = psp\_vfb2\_mag\_VSO[mag\_ttag\_mask,0] # Bx component

magVSO\_By = psp\_vfb2\_mag\_VSO[mag\_ttag\_mask,1] # By

magVSO\_Bz = psp\_vfb2\_mag\_VSO[mag\_ttag\_mask,2] # Bz

magVSO\_Btot = np.linalg.norm(psp\_vfb2\_mag\_VSO[mag\_ttag\_mask,:],axis=1)

print(magVSO\_Btot.shape)

axes[1].plot(mag\_ttags, magVSO\_Btot )

axes[2].plot(mag\_ttags, magVSO\_Bx )

axes[2].plot(mag\_ttags, magVSO\_By )

axes[2].plot(mag\_ttags, magVSO\_Bz )

axes[1].set\_ylabel('B total (nT)')

axes[2].set\_ylabel('B xyz (nT)')

plt.show()