Visualization of Diverse Geospace Data in Python

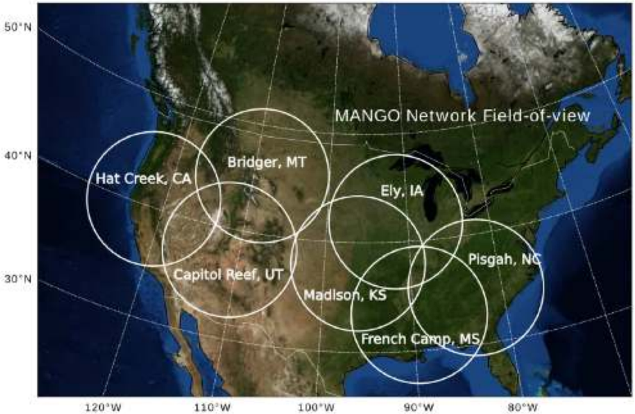
Leslie Lamarche, Asti Bhatt, Ashton Reimer,
Pablo Reyes, Todd Valentic

SRI International, Menlo Park, CA, USA

2019 CEDAR Workshop
Santa Fe, New Mexico, USA
June 16 – June 21, 2019

MANGOPY

Mid-latitude All-sky-imager Network for Geophysical Observations (MANGO)



The network of 630nm airglow imagers is supported through an NSF-CEDAR grant to study TID generation and evolution from lower atmospheric/auroral sources

- First-look movies available at `mango.sri.com/data`
- Data files available at `ftp://isr.sri.com/pub/earthcube/provider/asti/MANGOProcessed`
- Data available as png files of individual images and composite hdf5 files for entire nights

What is mangopy?

mangopy is a python package for accessing and working with MANGO imager data

Capabilities:

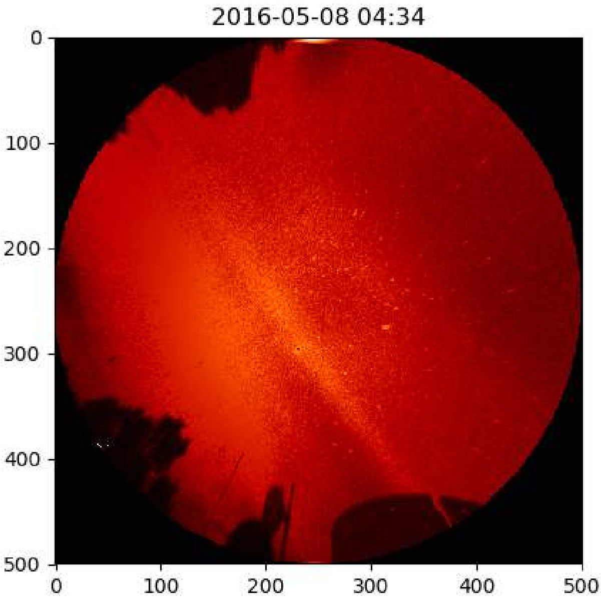
- Fetch MANGO data files from the FTP server
- Read composite hdf5 files and retrieve data arrays
- View imager data at a particular time either as individual images or on a map
- Create combined mosaic images from all cameras in the network
- python 2 and python 3 compatible

mangopy can be cloned and installed from GitHub:
https://github.com/astib/MANGO

Mango Demo

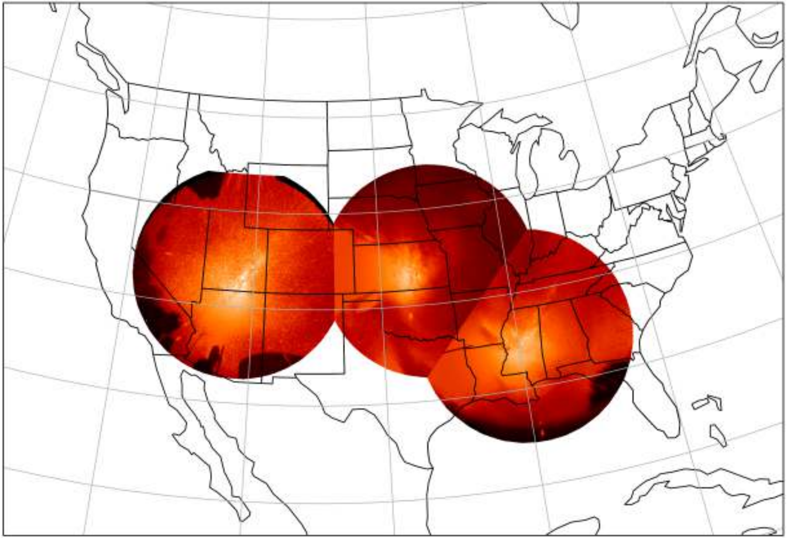
```
from mangopy import Mango, Mosaic
import datetime as dt

# First, view a single image
t0 = dt.datetime(2016,5,8,4,34)
mango = Mango()
site = mango.get_site_info(’Capitol Reef Field Station’)
mango.plot(site,t0)

# Then, view a compiled mosaic
t0 = dt.datetime(2016,8,3,6,15)
mosaic = Mosaic()
mosaic.plot_mosaic(t0)
```

Single MANGO Image



Mosaic Image

2016-08-03 06:15



Capitol Reef Field Station - 06:16:24
Rainwater Observatory - 06:16:00
Madison - 06:15:50

Future Work

- Expand the ability of mangopy to produce other data products
  - keograms
  - difference images
  - movies of both single imagers and mosaics
- Investigate different and more efficient ways of creating mosaic images, including different gridding schemes
- Add functionality to work with png image files instead of just hdf5 data sets

We welcome suggestions of other useful features! Please contact the InGeO team or submit a GitHub issue with ideas!

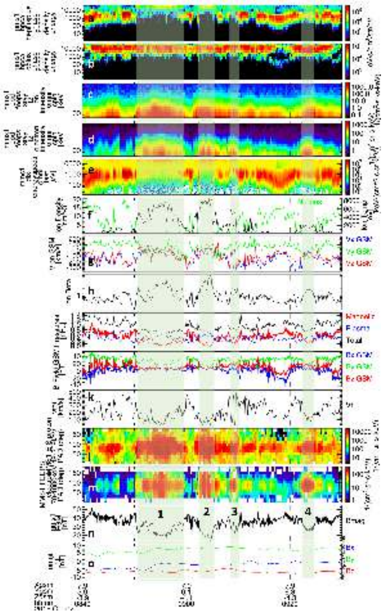
MIVIT

Multi-Instrument Visualization Toolkit (MIVIT)

Goal: Make it easier to compare diverse geospatial data sets

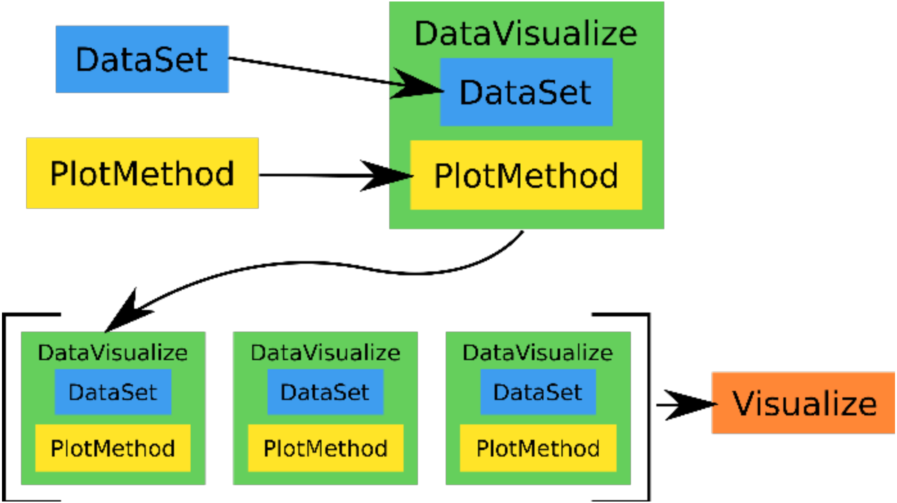
- Time series from different instruments are relatively straightforward to compare
- Data sets with 1 or more spatial dimensions are much more challenging to visualize simultaneously
- MIVIT is a package that allows users to compare multiple spatial data sets in a general way

`https://github.com/EarthCubeInGeo/MIVIT.git`



[image from Nykyri et al., 2019]

Architecture



DataSet

PlotMethod

DataVisualize

DataSet

PlotMethod

DataVisualize

DataSet

PlotMethod

DataVisualize

DataSet

PlotMethod

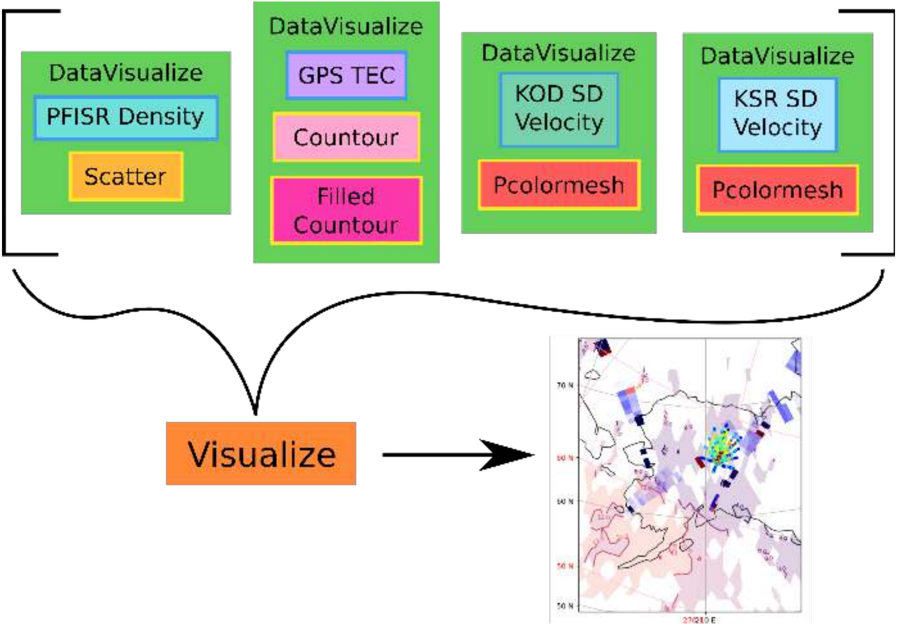
DataVisualize

DataSet

PlotMethod

Visualize

Architecture Example



DataVisualize

PFISR Density

Scatter

DataVisualize

GPS TEC

Countour

Filled Countour

DataVisualize

KOD SD Velocity

Pcolormesh

DataVisualize

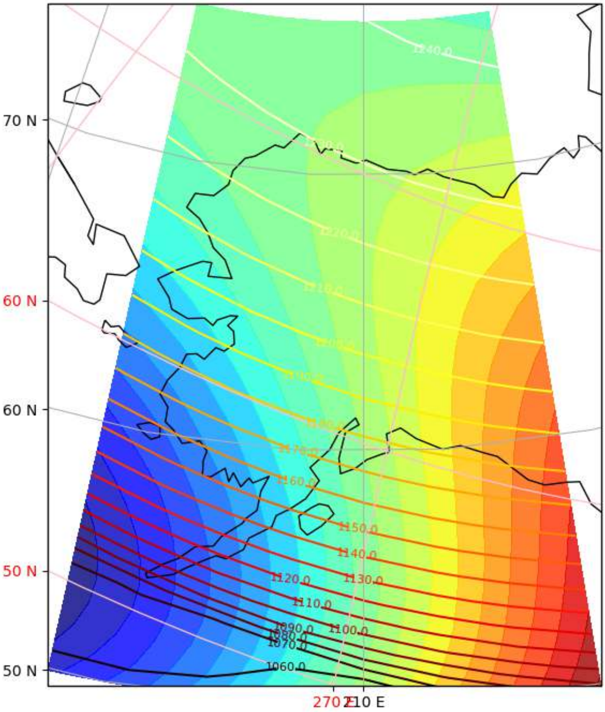
KSR SD Velocity

Pcolormesh

Visualize

Code Example

```
import mivit
import numpy as np
import datetime as dt

alt, lat, lon, dens, Tn, Ti = np.loadtxt(’iri.txt’).T

time_range = [dt.datetime(2019,6,18,1,30),dt.datetime(2019,6,18,1,31)]

# set density data visualization object
ds = mivit.DataSet(latitude=lat.reshape(8,26), longitude=lon.reshape(8,26)
pm = mivit.PlotMethod(cmap=’jet’, plot_type=’contourf’, label=’Electron De
dens = mivit.DataVisualization(ds, pm)

# set temperature data visualization object
ds = mivit.DataSet(latitude=lat.reshape(8,26), longitude=lon.reshape(8,26)
pm = mivit.PlotMethod(cmap=’hot’, plot_type=’contour’, label=’Ion Temperat
temp = mivit.DataVisualization(ds, pm)

# create plot
plot = mivit.Visualize([dens,temp], map_features=[’gridlines’,’coastlines’
plot.one_map()
```


1240.0
1230.0
1220.0
1210.0
1200.0
1190.0
1180.0
1170.0
1160.0
1150.0
1140.0
1130.0
1120.0
1110.0
1100.0
1090.0
1080.0
1070.0
1060.0
70 N
60 N
60 N
50 N
50 N
270 E
210 E

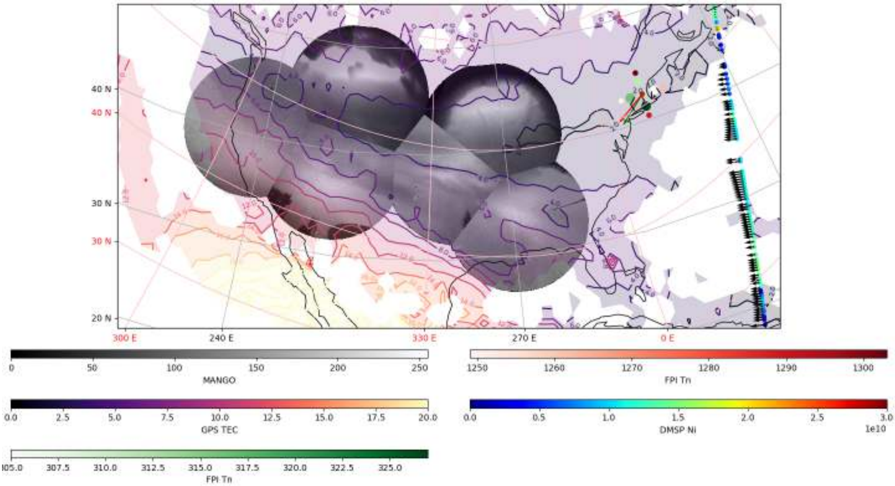
Helper scripts

- DataSet objects are useful to work with, but still have to read the data file or work with an API to get the coordinates and values to initialize
- Accessing data is a very hard step to abstract away in a completely general way

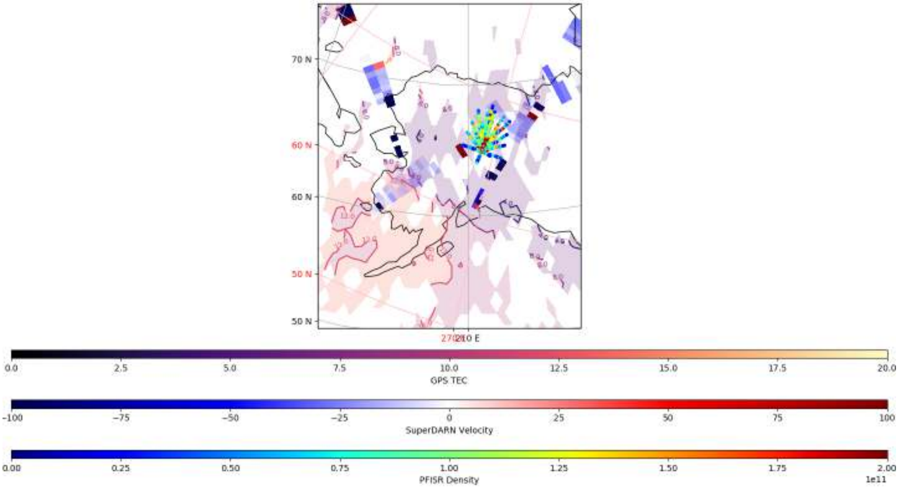
Solution: Create a set of helper scripts that create a DataSet object for a particular instrument or DataSet

- This makes it easier for users to visualize a wide variety of data sets without learning how to read new data file or use new software
- These helper scripts use software packages developed by the data providers - these packages are optional requirements for MIVIT
- Writing helper scripts for every single possible CEDAR instrument and data set is VERY time consuming
- If you write code to populate a DataSet object with data you work with, please consider contributing!

Memorial Day Storm



Alaska - November 7, 2015



Future Work

This is a new project that is actively being improved and developed!

- Improve labeling/color bar options
- Expand the available helper scripts - currently, MIVIT includes helper scripts for AMISR, SuperDARN, MANGO, and some Madrigal data sets
- Incorporate common models such as IRI, MSIS-E, and IGRF
- Introduce options for 3D visualization and interactive output
- Data gridding and interpolation
- Optimization - some functionality of MIVIT, especially things that involve querying APIs for data availability, can be quite slow

We welcome any ideas, suggestions, or recommendations that will make this software more useful!

Useful Links

InGeO Website: `https://ingeo.datatransport.org/home/`

MANGOPY GitHub Page: `https://github.com/astib/MANGO`

MIVIT GitHub Page: `https://github.com/EarthCubeInGeo/MIVIT`

Resen Docs: `https://resen.readthedocs.io/en/latest/`

This software was developed under the InGeO project, currently supported by the National Science Foundation's Cyberinfrastructure for Sustained Scientific Innovation (CSSI) program (Grant #1835573).