mangopy

P. Venkatraman, L. Lamarche, A. Bhatt

Jul 09, 2021

CONTENTS:

CHAPTER

ONE

MANGOPY

This repository contains code for processing all-sky images generated by the Mid-latitude All-sky-imager Network for Geospace Observations (MANGO), a project funded by the US National Science Foundation (NSF) (Grant # AGS-1452357 and AGS-1933013). MANGO has a network of 7 all-sky (wide-angle) cameras across the continental United States to observe the ionospheric airglow occurring at the heights of 250-300km and 9 proposed cameras observing the airglow at 90-100km. More information at the MANGO website.

The mangopy project is a python package for accessing and visualizing MANGO data. The project has several authors as mentioned in each code header. This code can be applied to image processing of other all-sky data (i.e. systems with wide-angle cameras pointed vertically towards the sky). The code was developed under three NSF funded projects, MANGO (Grant # AGS-1452357 and AGS-1933013) and InGeO (Grant # OAC-1835573).

1.1 Installing cartopy

Cartopy is used for visualizing MANGO data and must be installed before mangopy. Please refer to the cartopy installation instructions and make sure cartopy is successfully installed on your system before attempting to install mangopy.

1.2 Installing mangopy

Mangopy can be installed with pip:

```
pip install mangopy
```

The mangopy developers encourage community contributions! Please create a pull request if you've fixed a problem or created a new feature that would be useful to the community!

For basic usage examples, please refer to the mangopy tutorial jupyter notebook, mangopy tutorial.

Full mangopy documentation

CHAPTER

TWO

INSTALLATION

2.1 Installing cartopy

Cartopy is used for visualizing MANGO data and must be installed before mangopy. Please refer to the cartopy installation instructions and make sure cartopy is successfully installed on your system before attempting to install mangopy.

2.2 Installing mangopy

Mangopy can be installed with pip:

```
pip install mangopy
```

Alternatively, mangopy can be installed directly from the GitHub repository:

```
pip install git+https://github.com/mangonetwork/mangopy.git
```

2.3 Installing for development

If you would would like to clone the mangopy repo so you can modify the source code, follow the following instructions:

Clone the mangopy git repo:

```
git clone https://github.com/mangonetwork/mangopy.git
```

Change directories into the mangopy directory:

```
cd mangopy
```

To install in development mode, run the following command:

```
pip install -e .
```

CHAPTER

THREE

USAGE

3.1 Mango

First, import the Mango class:

```
from mangopy import Mango
%pylab inline
```

To instantiate a Mango object:

```
mango_object = Mango()
```

You can also specify the directory where the data can be found using the 'datadir' keyword.

Specify what site you would like to look at. For example, for Capitol Reef Field Station:

```
site = mango_object.get_site_info('Capitol Reef Field Station')
```

To specify a particular time and view an image, import datetime, and set a datetime object. You can view the image for the specified site at this time:

```
import datetime as dt

time_obj = dt.datetime(2016, 4, 10, 5, 30)
mango_object.plot(site, time_obj)
```

To show the site on a map:

```
mango_object.map(site, time_obj)
```

3.2 Mosaic

Now, import the Mosaic class:

```
from mangopy import Mosaic
```

Here's how you can instantiate the Mosaic class. By default, all sites will appear on the Mosaic, but you can specify a list of sites using the sites keyword. For all sites:

```
mosaic_all_sites = Mosaic()
```

To visualize a mosaic of Capitol Reef Field Station and Bridger alone:

```
mosaic_specific = Mosaic(sites = ['Capitol Reef Field Station', 'Bridger'])
```

To plot the mosaic for all sites at the time specifed:

```
mosaic_all_sites.plot_mosaic(time_obj)
```

If you prefer working with Jupyter Notebooks, here is the same tutorial, with an additional 'Accessing Data' example available on Jupyter Notebooks.

CHAPTER

FOUR

REFERENCE

4.1 Mango class

class mangopy.**Mango**(*datadir=None*, *download_data=False*)

Bases: object

Object for accessing and ploting data from a single MANGO camera.

Parameters

- **datadir** (*str, optional*) – Path to exisiting directory containing MANGO data.
- **download_data** (*bool, optional*) – If True, downloads data from ftp server.

fetch_datafile(*site*, *date*, *save_directory=None*)

Fetches mango data from online repository. Curtesy of AReimer's url_fetcher() function.

Parameters

- **site** (*str*) – Camera site name.
- **date** (*datetime object*) – Date image was taken.
- **save_directory** (*str, optional*) – Directory where files will be saved.

get_data(*site*, *targtime*)

Accesses the images and position of a site, given the site name and time.

Parameters

- **site** (*str*) – Camera site name
- **targtime** (*datetime object*) – Time of image as requested by user.

Returns

- **img_array** (*array*) – Image array
- **lat** (*float*) – Latitude array
- **lon** (*float*) – Longitude array
- **truetime** (*datetime object*) – Time at which image was taken.

get_site_info(*sites*)

Obtains information about sites given as user input.

Parameters **sites** (*list*) – List of sites.

Returns **site_list** – List of dictionaries with information about sites.

Return type list

map(*site*, *targtime*)
Plots a single MANGO image on the map.

Parameters

- **site** (*str*) – Camera site name
- **targtime** (*datetime object*) – Time of image as requested by user.

plot(*site*, *targtime*)
Plots a single MANGO image.

Parameters

- **site** (*str*) – Camera site name
- **targtime** (*datetime object*) – Time of image as requested by user.

read_datafile(*filename*, *targtime*)
Helper function for getting data; reads data in from hdf5 file.

Parameters

- **filename** (*str*) – hdf5 filename
- **targtime** (*datetime object*) – Time of image as requested by user

Returns

- **img_array** (*array*) – Image array
- **lat** (*float*) – Latitude array
- **lon** (*float*) – Longitude array
- **truetime** (*datetime object*) – Time image was taken

4.2 Mosaic class

class mangopy.**Mosaic**(*sites='all'*, *datadir=None*)
Bases: *mangopy.mango.Mango*

Object for creating and visualizing mosaics of all cameras in the MANGO network.

Parameters

- **sites** (*list, optional*) – Sites to be plotted as mosaic on map.
- **datadir** (*str, optional*) – Path to exisiting directory containing MANGO data.

create_all_mosaic(*date*, *saveFig=False*)
Creates all mosaic images for a particular date. Images should be approximately 5 minutes apart.

Parameters **date** (*datetime object*) – Date for which mosaic is created.

create_mosaic(*time*, *cell_edges=False*)
Creates the background grid for images at specifed time.

Parameters

- **time** (*datetime object*) – User requested time.
- **cell_edges** (*boolean, optional*) – Draws cell edges if set to True.

Returns

- **combined_grid** (*array*) – Background grid image.
- **grid_lat_values** (*array*) – Grid latitude values.
- **grid_lon_values** (*array*) – Grid longitude values.

`create_mosaic_movie`(*date*)
Creates a movie of all mosaic images for particular date. Requires ffmpeg to be installed.

Parameters `date` (*datetime object*) – Date for which mosaic movie is created.

`generate_grid`()
Create base background grid. Original images have the following approximate resolution:

lat_res ~ 0.025 degrees

lon_res ~ 0.035 degrees

Returns

- **grid_array** (*array*) – Array of grid latitude and longitude values.
- **edge_array** (*array*) – Array of edge latitude and longitude values.

`get_nearest_index`(*site*, *background_grid*, *time*)
Gets nearest neighbor interpolation indices for the specifed site.

Parameters

- `site` (*str*) – Site for which you need indices.
- `background_grid` (*array*) – Base background grid.
- `time` (*datetime object*) – Time of image as requested by user.

Returns **nearest_idx** – Nearest index of each image cell closest to grid cell.

Return type array

`grid_mosaic`(*time*, *grid*, *hierarchy*)
Creates combined grid based on hierarchy.

Parameters

- `time` (*datetime object*) – Time of images on mosaic as requested by user.
- `grid` (*array*) – Base background grid.
- `hierarchy` (*array*) – Hierarchy of sites to be plotted.

Returns

- **combined_grid** (*array*) – Combined grid.
- **truetime** (*datetime object*) – Time images were taken.

`haversine`(*lat0*, *lon0*, *lat*, *lon*)
Calculates distance (in km) between two points on Earth, assuming spherical Earth.

Parameters

- `lat0` (*float*) – Latitude of site.
- `lon0` (*float*) – Longitude of site.
- `lat` (*float*) – Latitude of grid.
- `lon` (*float*) – Longitude of grid.

Returns **km** – Haversine distance in kilometers.

Return type float

`plot_mosaic`(*time*, *dpi=300*, *saveFig=False*)

Plots images of sites closest to requested time on map with grid.

Parameters

- **time** (*datetime object*) – Time of images on mosaic as requested by user.
- **dpi** (*int, optional*) – Defaults to 300.
- **saveFig** (*boolean, optional*) – Saves figure of mosaic if set to True.

`site_hierarchy`(*grid_points*)

Calculates site hierarchy for common grid based on the distance of each point from each site. Site hierarchy is used to determine which camera to plot in each cell of the mosaic.

Parameters **grid_points** (*array*) – Coordinate points of base background grid.

Returns **hierarchy** – Array containing hierarchy of sites.

Return type array

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

INDEX