digitalearth Documentation

Mostafa Farrag

Feb 12, 2023

CONTENTS

CHAPTER

ONE

DIGITALEARTH - GEOSPATIAL VISUALIZATION PACKAGE

digitalearth is a Python package providing different plots for rasters and vector data

1.1 Main Features

-

1.1.1 Installation

Stable release

Please install `digitalearth` in a Virtual environment so that its requirements don't tamper with your system's python.

conda

the easiest way to install `digitalearth` is using `conda` package manager. `digitalearth` is available in the conda-forge channel. To install you can use the following command:

- `conda install -c conda-forge digitalearth`

If this works it will install Hapi with all dependencies including Python and gdal, and you skip the rest of the installation instructions.

Installing Python and gdal dependencies

The main dependencies for digitalearth are an installation of Python 3.9+, and gdal

Installing Python

For Python we recommend using the Anaconda Distribution for Python 3, which is available for download from https://www.anaconda.com/download/. The installer gives the option to add `python` to your `PATH` environment variable. We will assume in the instructions below that it is available in the path, such that `python`, `pip`, and `conda` are all available from the command line.

Note that there is no hard requirement specifically for Anaconda's Python, but often it makes installation of required dependencies easier using the conda package manager.

Install as a conda environment

The easiest and most robust way to install Hapi is by installing it in a separate conda environment. In the root repository directory there is an `environment.yml` file. This file lists all dependencies. Either use the `environment.yml` file from the master branch (please note that the master branch can change rapidly and break functionality without warning), or from one of the releases {release}.

Run this command to start installing all Hapi dependencies:

- `conda env create -f environment.yml`

This creates a new environment with the name `digitalearth`. To activate this environment in a session, run:

- `conda activate digitalearth`

For the installation of Hapi there are two options (from the Python Package Index (PyPI) or from Github). To install a release of Hapi from the PyPI (available from release 2018.1):

- `pip install digitalearth=={release}`

From sources

The sources for HapiSM can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/MAfarrag/digitalearth
```

Or download the tarball:

```
$ curl -OJL https://github.com/MAfarrag/digitalearth/tarball/main
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

To install directly from GitHub (from the HEAD of the master branch):

- `pip install git+https://github.com/MAfarrag/digitalearth.git`

or from Github from a specific release:

- `pip install git+https://github.com/MAfarrag/digitalearth.git@{release}`

Now you should be able to start this environment's Python with `python`, try `import digitalearth` to see if the package is installed.

More details on how to work with conda environments can be found here: https://conda.io/docs/user-guide/tasks/manage-environments.html

If you are planning to make changes and contribute to the development of Hapi, it is best to make a git clone of the repository, and do a editable install in the location of you clone. This will not move a copy to your Python installation directory, but instead create a link in your Python installation pointing to the folder you installed it from, such that any changes you make there are directly reflected in your install.

- `git clone https://github.com/MAfarrag/digitalearth.git`
- `cd digitalearth`
- `activate digitalearth`
- `pip install -e .`

Alternatively, if you want to avoid using `git` and simply want to test the latest version from the `master` branch, you can replace the first line with downloading a zip archive from GitHub: https://github.com/MAfarrag/digitalearth/archive/master.zip libraries.io.

Install using pip

Besides the recommended conda environment setup described above, you can also install Hapi with `pip`. For the more difficult to install Python dependencies, it is best to use the conda package manager:

- `conda install numpy scipy gdal netcdf4 pyproj`

you can check libraries.io. to check versions of the libraries

Then install a release {release} of digitalearth (available from release 2018.1) with pip:

- `pip install digitalearth=={release}`

Check if the installation is successful

To check it the install is successful, go to the examples directory and run the following command:

- `python -m digitalearth.*******`

This should run without errors.

Note: This documentation was generated on Feb 12, 2023

Documentation for the development version: https://digitalearth.readthedocs.org/en/latest/

Documentation for the stable version: https://digitalearth.readthedocs.org/en/stable/

1.1.2 Plot raster/array

Library

- import the libraries

```
from Hapi.visualizer import Visualize as vis
import gdal
import pandas as pd
```

Paths

- define paths to the raster file.

```
RasterAPath = "data/GIS/Hapi_GIS_Data/acc4000.tif"
```

Read the raster

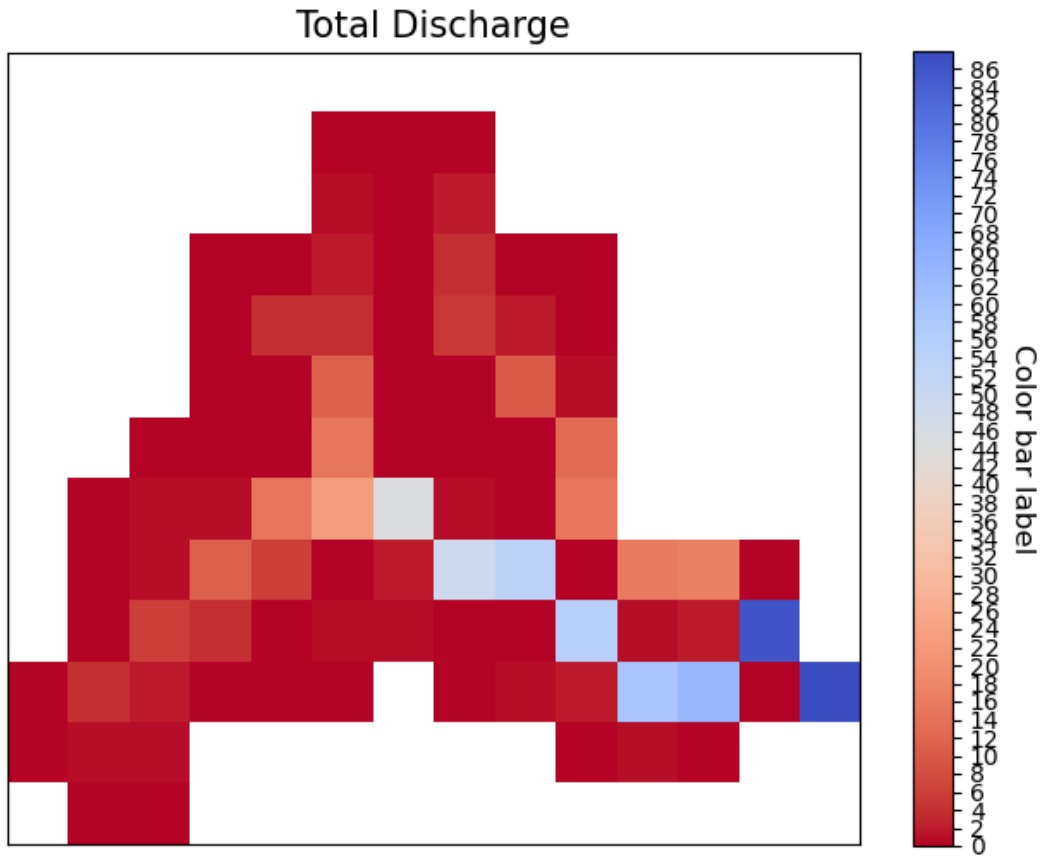
-To plot the array you need to read the raster using gdal.

```
src = gdal.Open(RasterAPath)
```

Default Plot

- Then using all the default parameters in the PlotArray method you can directly plot the gdal.Dataset.

```
vis.PlotArray(src)

(<Figure size 576x576 with 2 Axes>,
 <AxesSubplot:title={'center':'Total Discharge'}>)
```

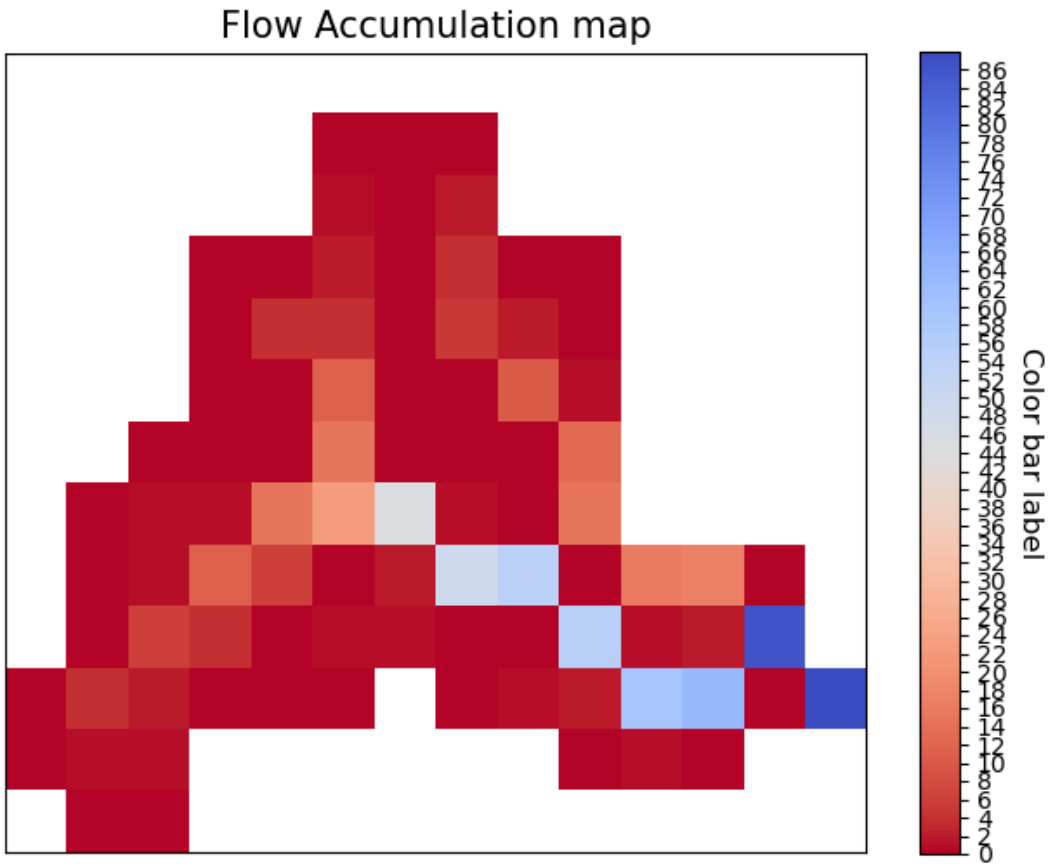


- **However as you see in the plot you might need to adjust the color to different color scheme or the** display of the colorbar, colored label. you might don't need to display the labels showing the values of each cell, and for all of these decisions there are a lot of customizable parameters.

Basic Figure features

- First for the size of the figure you have to pass a tuple with the width and height.
- **Figsize**
  [[tuple], optional] figure size. The default is (8,8).
- **Title**
  [[str], optional] title of the plot. The default is 'Total Discharge'.
- **titlesize**
  [[integer], optional] title size. The default is 15.

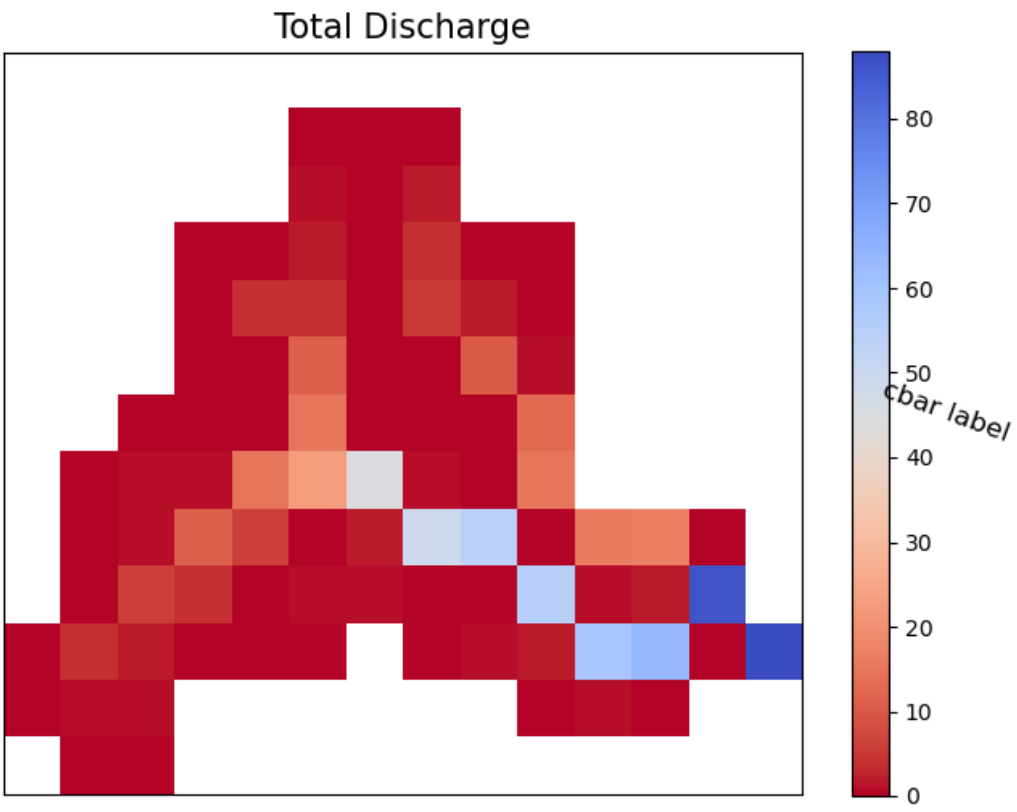
```
Figsize=(8, 8)
Title='Flow Accumulation map'
titlesize=15

vis.PlotArray(src, Figsize=Figsize, Title=Title, titlesize=titlesize)


(<Figure size 576x576 with 2 Axes>,
<AxesSubplot:title={'center':'Flow Accumulation map'}>)
```



Color Bar

- **Cbarlength**
  [[float], optional] ratio to control the height of the colorbar. The default is 0.75.
- **orientation**
  [[string], optional] orintation of the colorbar horizontal/vertical. The default is 'vertical'.
- **cbarlabelsize**
  [integer, optional] size of the color bar label. The default is 12.
- **cbarlabel**
  [str, optional] label of the color bar. The default is 'Discharge m3/s'.
- **rotation**
  [[number], optional] rotation of the colorbar label. The default is -90.
- **TicksSpacing**
  [[integer], optional] Spacing in the colorbar ticks. The default is 2.

```
Cbarlength=0.75
orientation='vertical'
cbarlabelsize=12
cbarlabel= 'cbar label'
rotation=-20
TicksSpacing=10

vis.PlotArray(src, Cbarlength=Cbarlength, orientation=orientation,
              cbarlabelsize=cbarlabelsize, cbarlabel=cbarlabel, rotation=rotation,
              TicksSpacing=TicksSpacing)


(<Figure size 576x576 with 2 Axes>,
<AxesSubplot:title={'center':'Total Discharge'}>)
```



Color Schame

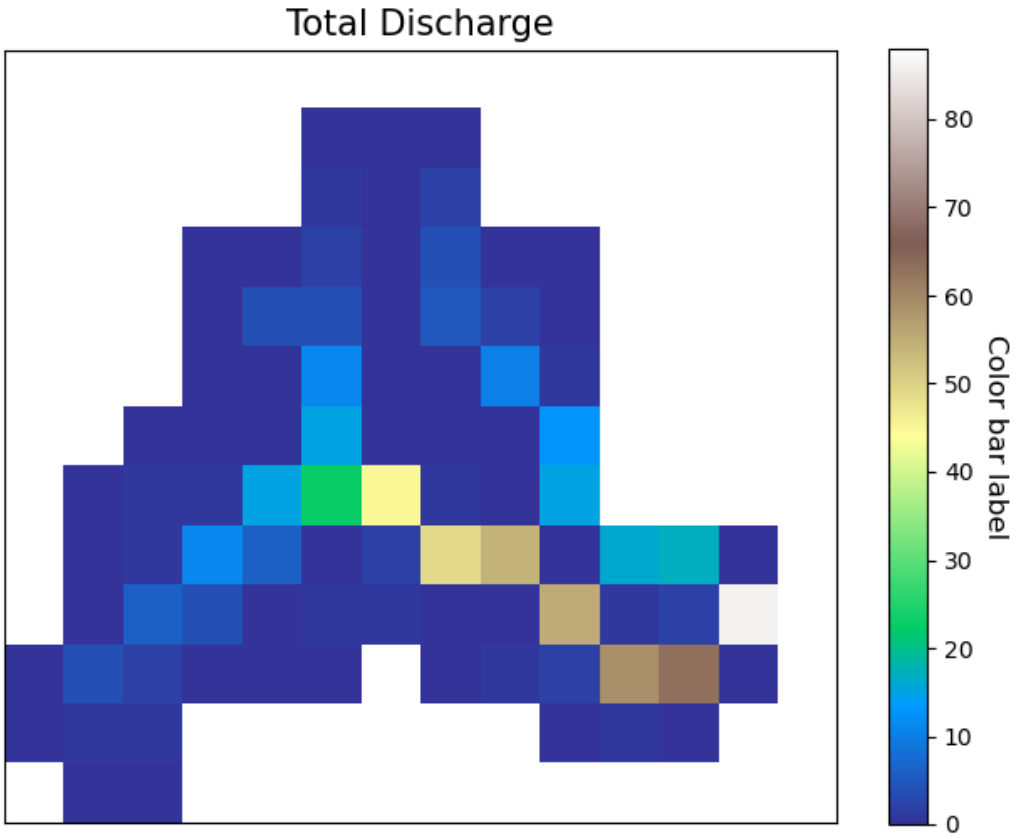
- **ColorScale**
  [integer, optional] there are 5 options to change the scale of the colors. The default is 1. 1- ColorScale 1 is the normal scale 2- ColorScale 2 is the power scale 3- ColorScale 3 is the SymLogNorm scale 4- ColorScale 4 is the PowerNorm scale 5- ColorScale 5 is the BoundaryNorm scale
- **gamma**
  [[float], optional] value needed for option 2 . The default is 1./2..
- **linthresh**
  [[float], optional] value needed for option 3. The default is 0.0001.
- **linscale**
  [[float], optional] value needed for option 3. The default is 0.001.
- **midpoint**
  [[float], optional] value needed for option 5. The default is 0.
- **cmap**
  [[str], optional] color style. The default is 'coolwarm_r'.

```
# for normal linear scale
ColorScale = 1
cmap='terrain'
vis.PlotArray(src, ColorScale=ColorScale, cmap=cmap, TicksSpacing=TicksSpacing)

```

(continues on next page)

(continued from previous page)

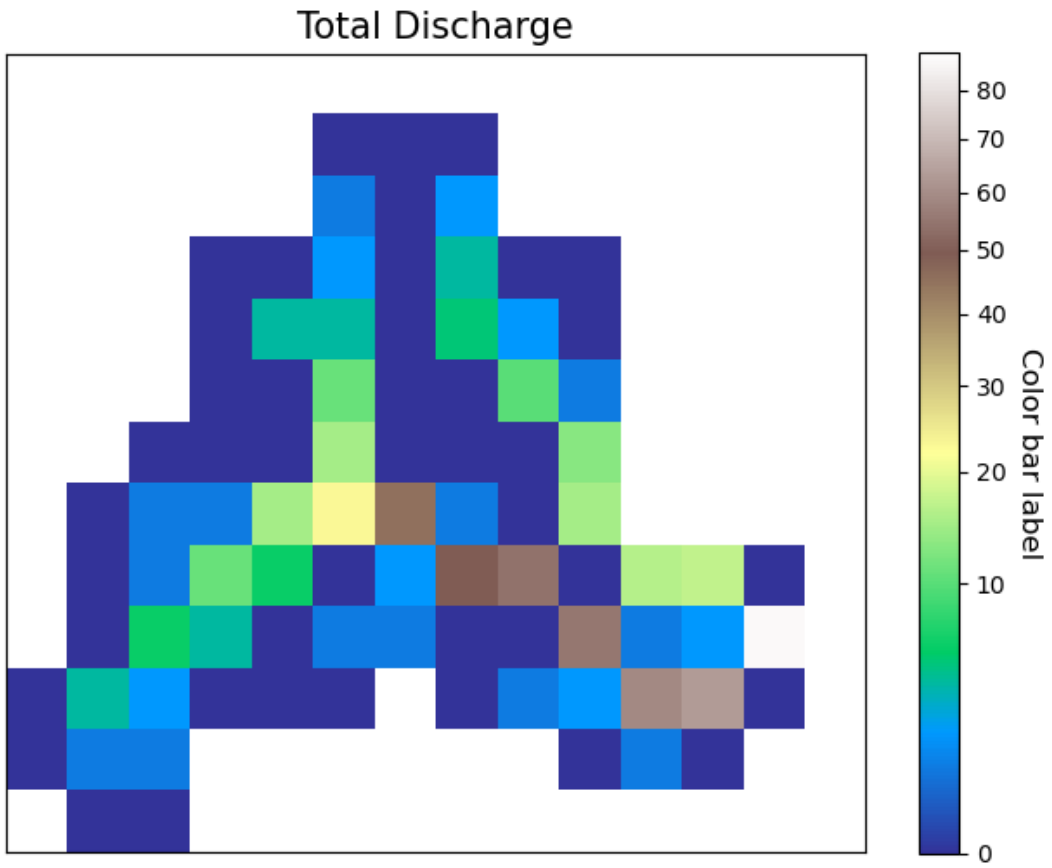
```
(<Figure size 576x576 with 2 Axes>,
<AxesSubplot:title={'center':'Total Discharge'}>)
```

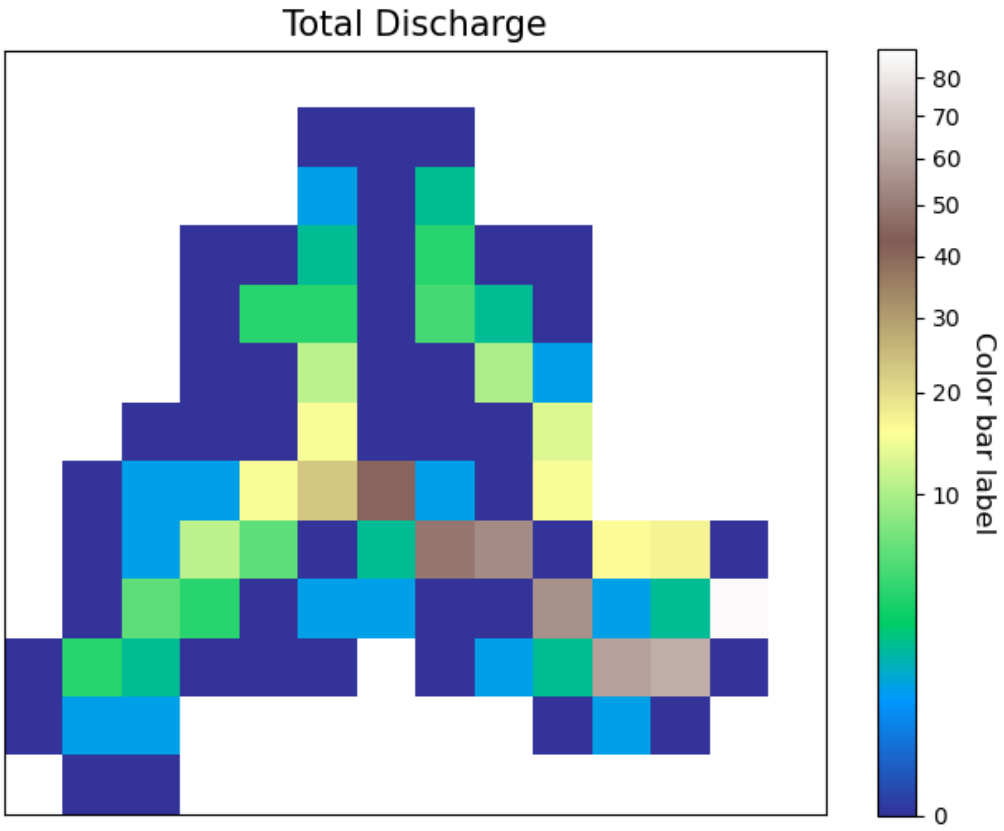


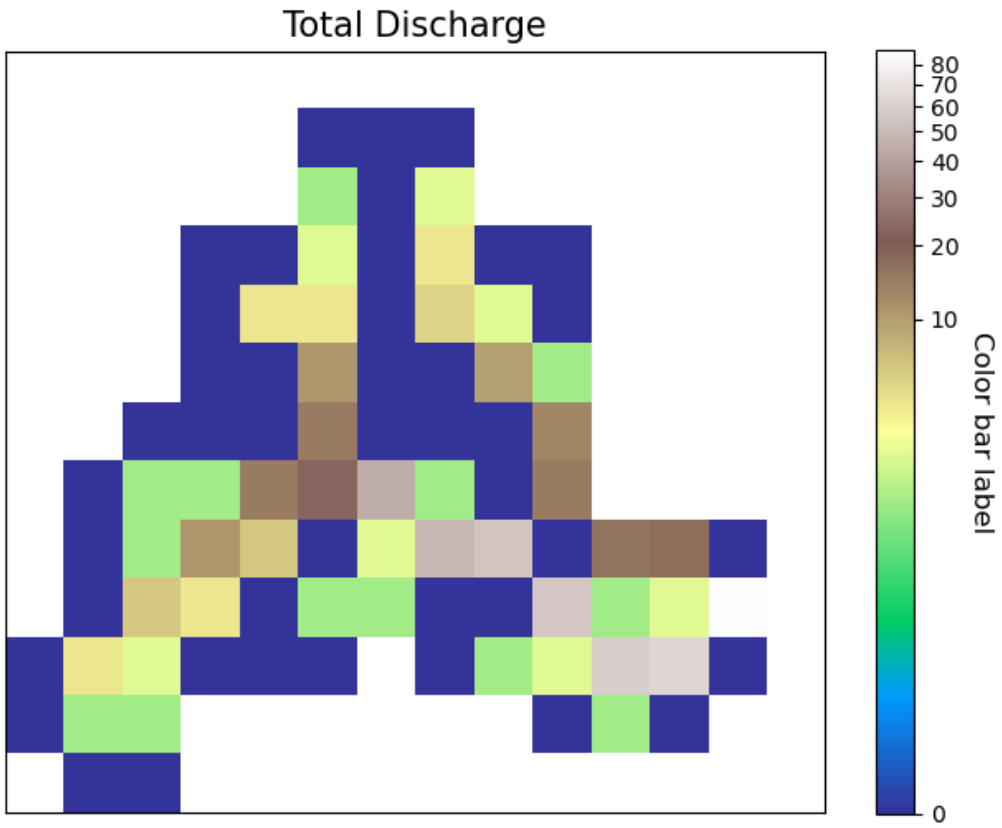
Power Scale

- The more you lower the value of gamma the more of the color bar you give to the lower value range

```
ColorScale = 2
gamma=0.5

vis.PlotArray(src, ColorScale=ColorScale, cmap=cmap, gamma=gamma,
              TicksSpacing=TicksSpacing)

vis.PlotArray(src, ColorScale=ColorScale, cmap=cmap, gamma=0.4,
          TicksSpacing=TicksSpacing)

vis.PlotArray(src, ColorScale=ColorScale, cmap=cmap, gamma=0.2,
          TicksSpacing=TicksSpacing)

(<Figure size 576x576 with 2 Axes>,
<AxesSubplot:title={'center':'Total Discharge'}>)
```

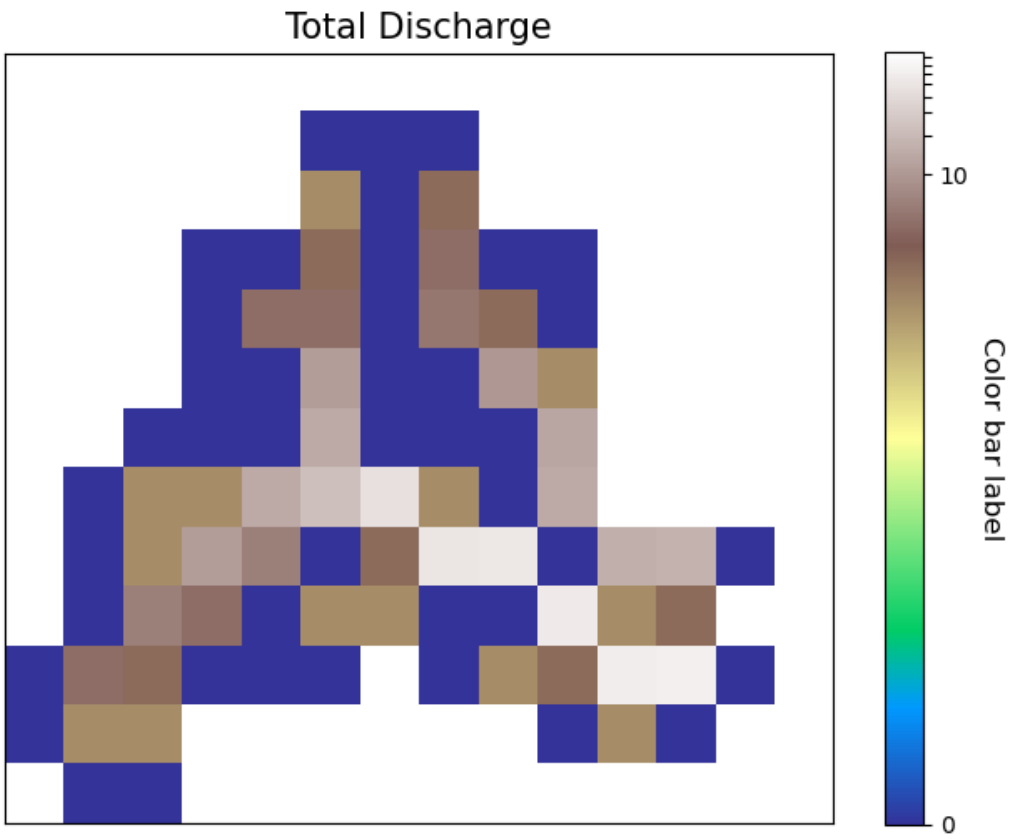

Total Discharge
80
70
60
50
40
30
20
10
0
Color bar label


Total Discharge
80
70
60
50
40
30
20
10
0
Color bar label



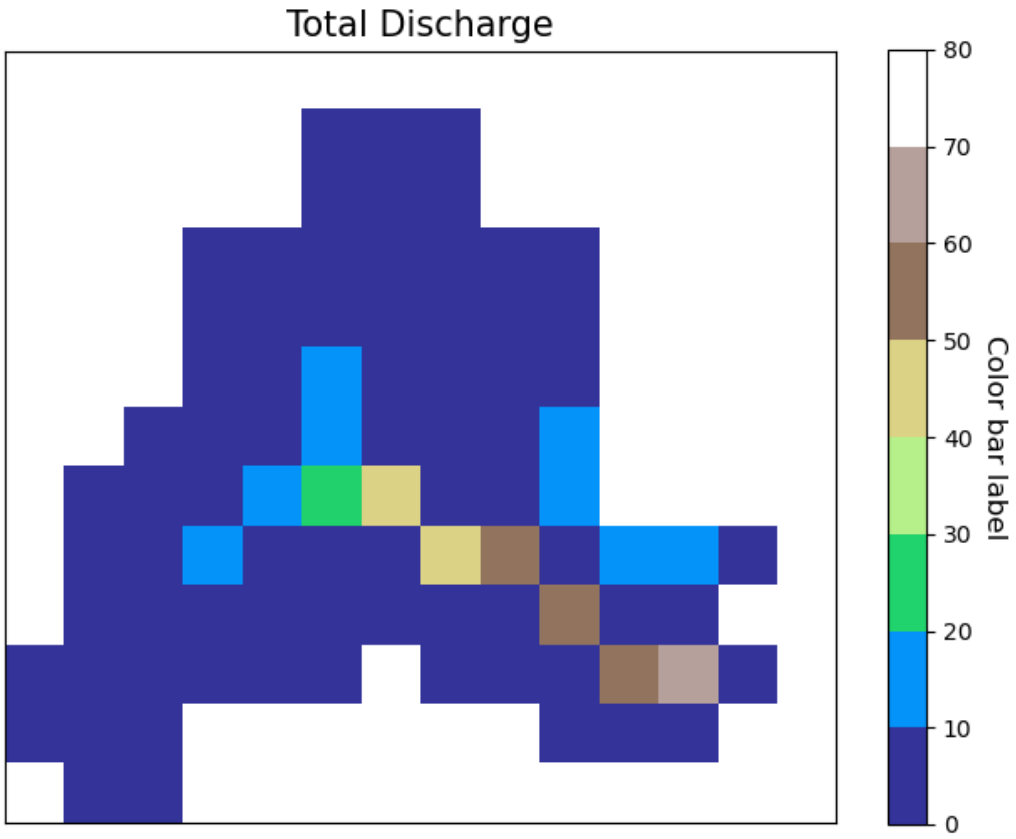
SymLogNorm scale

```
ColorScale = 3
linscale=0.001
linthresh=0.0001
vis.PlotArray(src, ColorScale=ColorScale, linscale=linscale, linthresh=linthresh,
              cmap=cmap, TicksSpacing=TicksSpacing)


(<Figure size 576x576 with 2 Axes>,
<AxesSubplot:title={'center':'Total Discharge'}>)
```



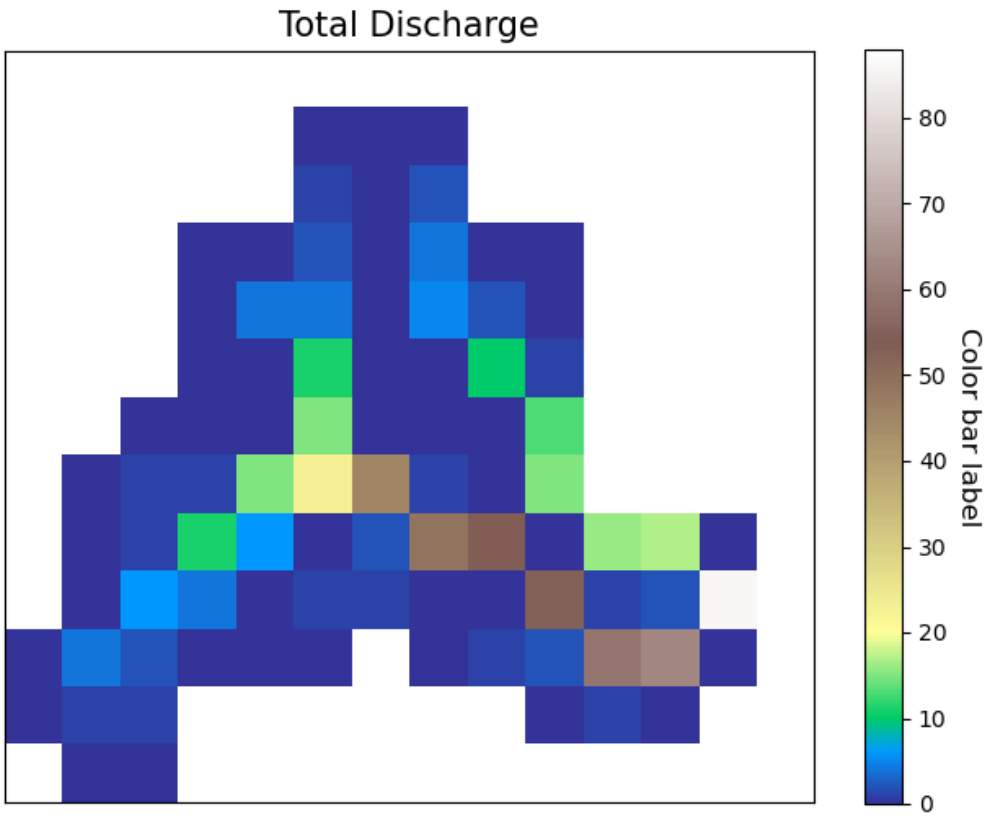
PowerNorm scale

```
ColorScale = 4
vis.PlotArray(src, ColorScale=ColorScale,
          cmap=cmap, TicksSpacing=TicksSpacing)

(<Figure size 576x576 with 2 Axes>,
<AxesSubplot:title={'center':'Total Discharge'}>)
```



Color scale 5

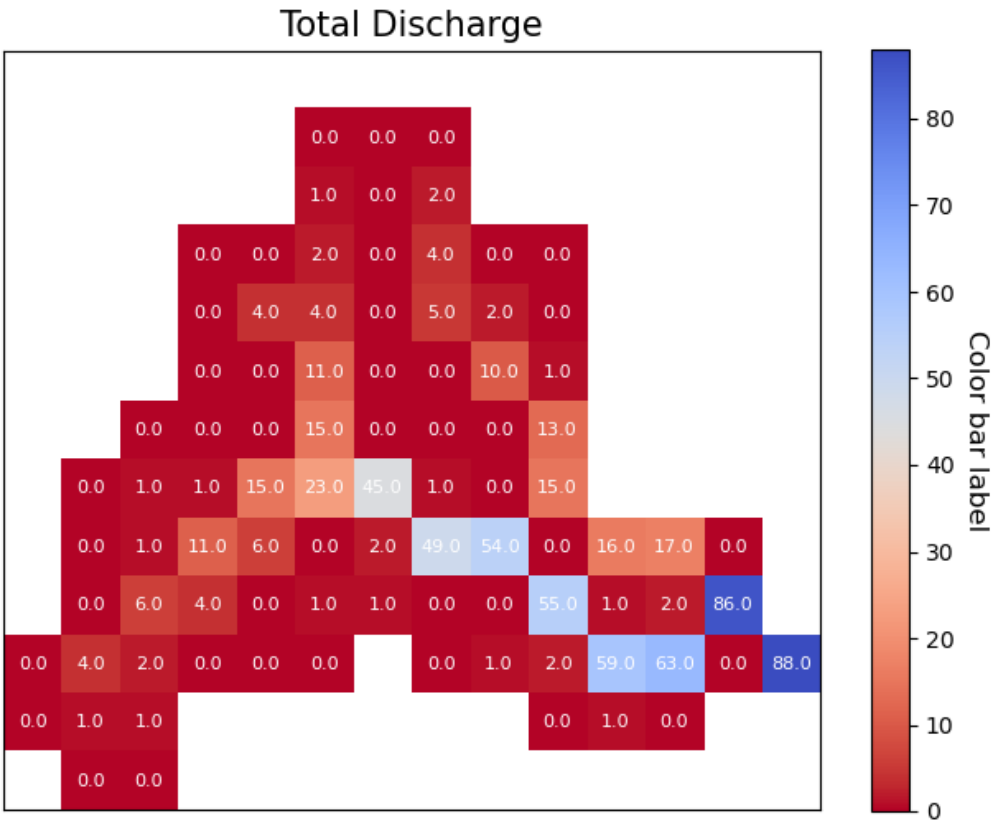
```
ColorScale = 5
midpoint=20
vis.PlotArray(src, ColorScale=ColorScale, midpoint=midpoint,
              cmap=cmap, TicksSpacing=TicksSpacing)


(<Figure size 576x576 with 2 Axes>,
<AxesSubplot:title={'center':'Total Discharge'}>)
```



Cell value label

- **display_cellvalue**
  [[bool]] True if you want to display the values of the cells as a text
- **NumSize**
  [integer, optional] size of the numbers plotted intop of each cells. The default is 8.
- **Backgroundcolorthreshold**
  [[float/integer], optional] threshold value if the value of the cell is greater, the plotted numbers will be black and if smaller the plotted number will be white if None given the maxvalue/2 will be considered. The default is None.

```
display_cellvalue = True
NumSize=8
Backgroundcolorthreshold=None

vis.PlotArray(src, display_cellvalue=display_cellvalue, NumSize=NumSize,
              Backgroundcolorthreshold=Backgroundcolorthreshold,
              TicksSpacing=TicksSpacing)

(<Figure size 576x576 with 2 Axes>,
<AxesSubplot:title={'center':'Total Discharge'}>)
```



Plot Points

if you have points that you want to display in the map you can read it into a dataframe in condition that it has two columns “x”, “y” which are the coordinates of the points of theand they have to be in the same coordinate system as the raster.

- read the points

```
pointsPath = "data/GIS/Hapi_GIS_Data/points.csv"
points = pd.read_csv(pointsPath)
```

- plot the points

```
Gaugecolor='blue'
Gaugesize=100
IDcolor="green"
IDsize=20
vis.PlotArray(src, Gaugecolor=Gaugecolor, Gaugesize=Gaugesize,
              IDcolor=IDcolor, IDsize=IDsize, points=points,
              display_cellvalue=display_cellvalue, NumSize=NumSize,
              Backgroundcolorthreshold=Backgroundcolorthreshold,
              TicksSpacing=TicksSpacing)

(<Figure size 576x576 with 2 Axes>,
```

(continues on next page)

(continued from previous page)

```
<AxesSubplot:title={'center':'Total Discharge'}>)
```