Using Python to Plot OPeNDAP Files in the NCAR/EOL Field Data Archive

Updated: 08 September 2022

Authorship: NCAR Earth Observing Laboratory Data Management & Services Group
Contact: eol-archive@ucar.edu

Acknowledgment: This work was sponsored by the National Science Foundation. This material is based upon work supported by the National Center for Atmospheric Research, a major facility sponsored by the National Science Foundation and managed by the University Corporation for Atmospheric Research. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the National Science Foundation.

General Description

This document serves as a guide on how python scripts can be used with OPeNDAP within the EOL Field Data Archive to plot variables found in files with OPeNDAP access without needing to download the files. Two examples of python plotting scripts can be found in the files "python_xy.py" and "python_xyz.py". The scripts were created by NCAR EOL and create 2D and 3D plots in python, respectively. Note that these scripts are provided "as is" and are not supported. For questions about Python scripts and other features, refer to the Python help page at:

https://www.python.org/about/help/

OPeNDAP within the EOL Field Data Archive

The EOL Field Data Archive (FDA) contains datasets that have OPeNDAP compatibility. An example dataset in the FDA would be GCIP/ ESOP-95 Surface: Hourly Surface Meteorological Composite. This dataset was used to generate the plots in the sections below.

For this dataset, 2D plots were created for a one site (Ashton) for Temperature and Dew Point using script **python_xy.py** (See Figure 8). A 3D plot was also created using **python_xyz.py** that shows Temperature specific time over the entire area of interest in the file (See figure 14).

How to use the Python Plot tools with the OPeNDAP Capabilities in the EOL Field Data Archive

Download Necessary Packages

In order to successfully run these scripts, you must have the following packages downloaded and use Python3:

- Pydap
- Matplotlib
- Numpy

Choose a dataset in the EOL FDA that has OPeNDAP Capabilities

Figure 1 shows the top portion of the EOL FDA dataset description page for the GCIP/ESOP-95 Surface: Hourly Surface Meteorological Composite dataset. The OPeNDAP feature is located in the “Data access” section of the dataset page. Clicking on the “OPeNDAP access” link will show the OPeNDAP page for the dataset. This page will show a list of all the files in the dataset and their corresponding OPeNDAP links. If a dataset in the EOL FDA does not have an OPeNDAP link in the “Data access” section, then the dataset does not have OPeNDAP capabilities.

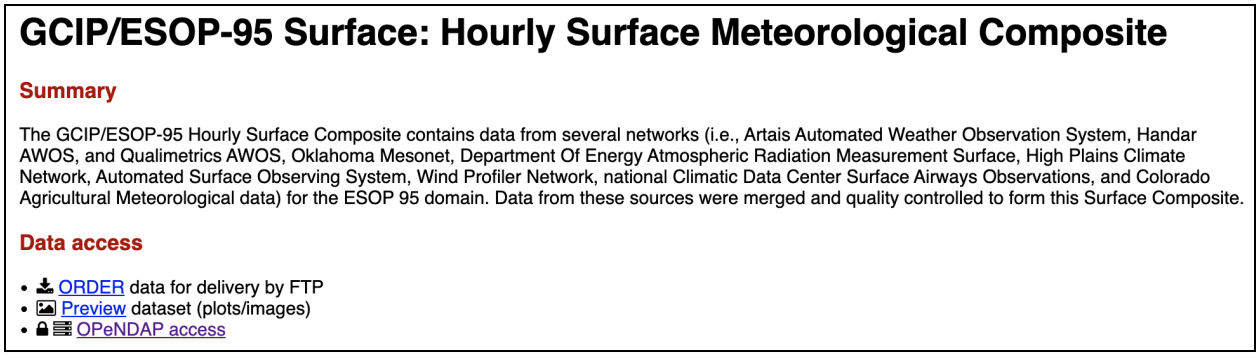

Figure 1: Top portion of the dataset description page for the EOL GCIP/ESOP-95 Surface: Hourly Surface Meteorological Composite dataset. The OPeNDAP feature for datasets (if applicable) is located in the “Data access” section of the dataset description page.

Get the Link to the File for Plotting

Figure 2 shows the “OPeNDAP access” page for the EOL GCIP/ESOP-95 Surface: Hourly Surface Meteorological Composite dataset. This page lists all files in the dataset. From here the OPeNDAP link files can be found. Clicking the “scissors” symbol will show the OPeNDAP link for a specific file. This link can then be fed into the Python scripts for plotting. For the plots shown in this document, file “ES95HRLY_950715.qcf” was used and the OPeNDAP link for that file is:

https://data.eol.ucar.edu/opendap/data/esop_95/hrly_sfc/ES95HRLY_950715.qcf

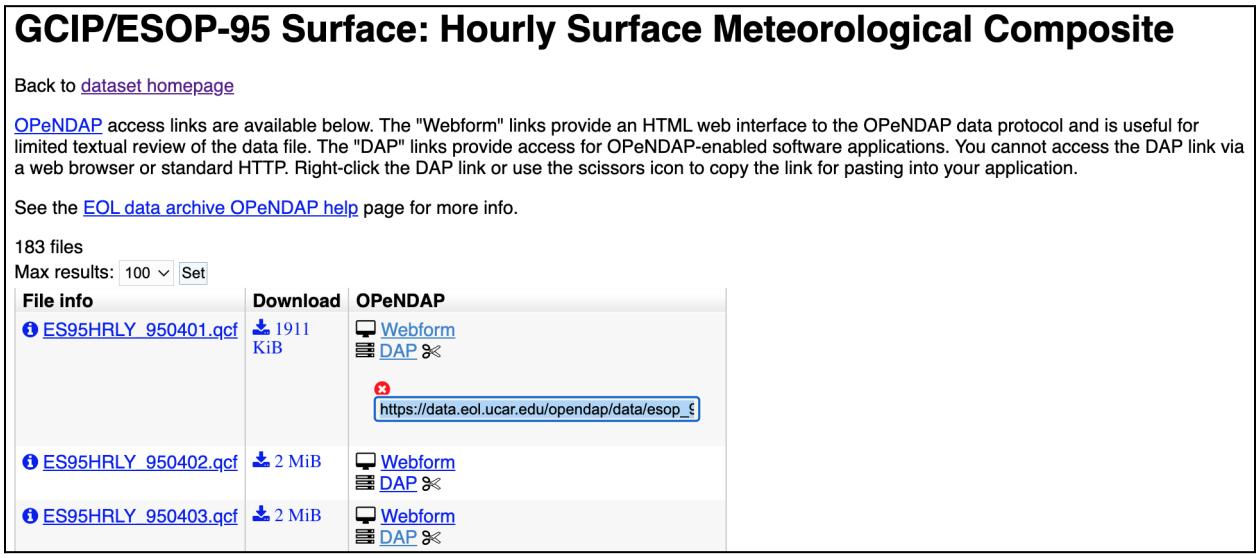

Figure 2: The OPeNDAP page for the EOL GCIP/ESOP-95 Surface: Hourly Surface Meteorological Composite dataset. Clicking on the “scissors” symbol for a file will show the OPeNDAP link for that specific file.

Creating 2D plots

Changing hard-coded values

The script uses the following hard-coded entities that may need to be changed. Note that to locate any of these values, simply search **"HARD-CODED"** in this script.

- *missing_value:* This is a string containing the value used for missing data points
    - For our example, the missing value is “-999.99”
- *var_index:* This is the number of variables preceding the variables available to plot against time. This includes any variables containing station information, time, and location. Count the number of variables that include date, time, or location information.

- For our example, the *var_index* is 9 because there are nine date, time, and location variables (date_nominal, time_nominal, date, time, network_name, platform_name, latitude, longitude, station_occurrence)
- Figure 3 shows how to view and count these variables using the OPeNDAP webform page.

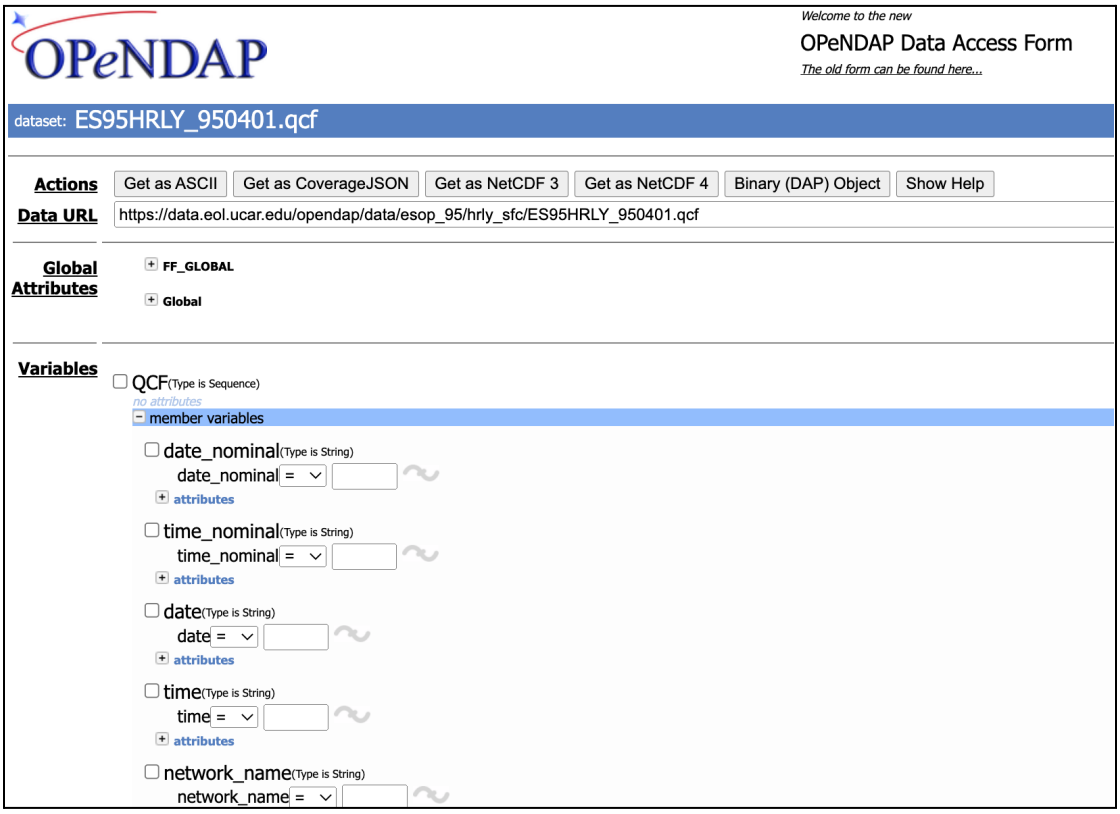

Figure 3: OPeNDAP webform page where variables are listed.

- *formatData*: This is a string containing format information to convert strings to the python object datetime.
  - For our example, the formatData = “%Y/%m/%d%H%:%M:%S” where Y is the year, m is the month, d is the day, H is the hour, M is the minute and S is the seconds value.
  - For more information on the datetime object and format strings, consult: https://docs.python.org/3/library/datetime.html
- *variable names*: This script assumes that the variable names from the OPeNDAP file are "date", "time", "network_name", "platform_name".
- *Graph elements*:
  - *graph size*: This is hard-coded into the figure declaration and declares the size of the output graph.

- *y-ticks*: The graphs are formatted so that the y-ticks are in steps of 3 from the minimum value to the maximum value.
- *x-labe*l: The graphs will be produced with the label "Date and Time" on the x-axis.
- *title*: The plot title is set to be the variable(s), date range, and location name.
- *file name*: Similarly to the plot title, the file name to be saved includes the start date, station name, and variable names.

To change any of the values listed above, simply search "HARD-CODED" in this script.

Calling the script from the command line:
To call this script, use the following command:

python3 python_xy.py *OPeNDAP_url*

where *OPeNDAP_url* is the link to the dataset acquired earlier in this section. See Figure 2 above. Figure 4 shows the command to call this script using the GCIP/ESOP-95 Surface: Hourly Surface Meteorological Composite Dataset.

```
python3 python_xy.py https://data.eol.ucar.edu/opendap/data/
esop_95/hrly_sfc/ES95HRLY_950715.qcf
```

Figure 4: Calling the python plot script from the command line

Answering the prompts:
Once you call the script, you should see the prompt displayed in Figure 5 asking which variable(s) you would like to plot. If you are unsure of the variables that are available to plot, enter the word “list”. Otherwise, enter the variable(s) to plot separated by commas as seen in Figure 5. Please note that you must enter the variables exactly as they appear in the OPeNDAP webform or variable list to avoid errors.

```
Which variables would you like to plot against time? To see a list of possible variables, ent
er 'list'. Please note that not entering the variable name as it appears within the file or O
PeNDAP webform will result in errors.

list

This file has the following variables available to plot against time:


['surface_altitude', 'surface_air_pressure', 'surface_air_pressure_flag', 'air_pressure_at_me
an_sea_level', 'air_pressure_at_mean_sea_level_flag', 'air_pressure_at_mean_sea_level_compute
d', 'air_pressure_at_mean_sea_level_computed_flag', 'air_temperature', 'air_temperature_flag'
, 'dew_point_temperature', 'dew_point_temperature_flag', 'wind_speed', 'wind_speed_flag', 'wi
nd_from_direction', 'wind_from_direction_flag', 'precipitation_amount', 'precipitation_amount
_flag', 'gust_indicator', 'wind_speed_of_gust', 'wind_speed_of_gust_flag', 'present_weather',
 'present_weather_flag', 'visibility_in_air', 'visibility_in_air_flag', 'cloud_layer_first_ba
se_altitude', 'cloud_layer_first_base_indicator', 'cloud_layer_first_base_indicator_flag', 'c
loud_layer_first_area_fraction', 'cloud_layer_first_area_fraction_flag', 'cloud_layer_second_
base_altitude', 'cloud_layer_second_base_indicator', 'cloud_layer_second_base_indicator_flag'
, 'cloud_layer_second_area_fraction', 'cloud_layer_second_area_fraction_flag', 'cloud_layer_t
hird_base_altitude', 'cloud_layer_third_base_indicator', 'cloud_layer_third_base_indicator_fl
ag', 'cloud_layer_third_area_fraction', 'cloud_layer_third_area_fraction_flag']


Enter variable(s) to plot separated by commas:
air_temperature, dew_point_temperature
```

Figure 5: Answering the prompt to select variables to plot.

After successfully entering the variables to plot, you will be prompted to enter the station(s) for which you want to create plots as seen in Figure 6. Again, if you are unsure of the stations available to plot, enter the word “list”. Otherwise, enter the station(s) to plot separated by commas. For our example, we will be creating plots for the ARMSFC1-Ashton station.

```
To see a list of possible stations, enter 'list'. Please note that not entering the station n
ame as it appears within the file will result in errors.

ARMSFC1-Ashton
```

Figure 6: Answering the prompt to select stations to plot.

Lastly, the script will prompt you for the label for the y-axis. For our example, since the variables to plot are air temperature and dew point temperature, we will use “Degrees (C)” as seen in Figure 7.

```
What would you like the label to be on the y-axis?
Degrees (C)

Creating plots...


Plot saved in the current directory with the name ARMSFC1-Ashton_air_temperature_dew_point_te
mperature_199507142355.png
```

Figure 7: Entering the label for the y-axis, and the output of the program.

As seen in Figure 8, after entering the label for the y-axis, the script will create and save the plot(s) to the current directory. Figure 8 shows the generated 2D plot.

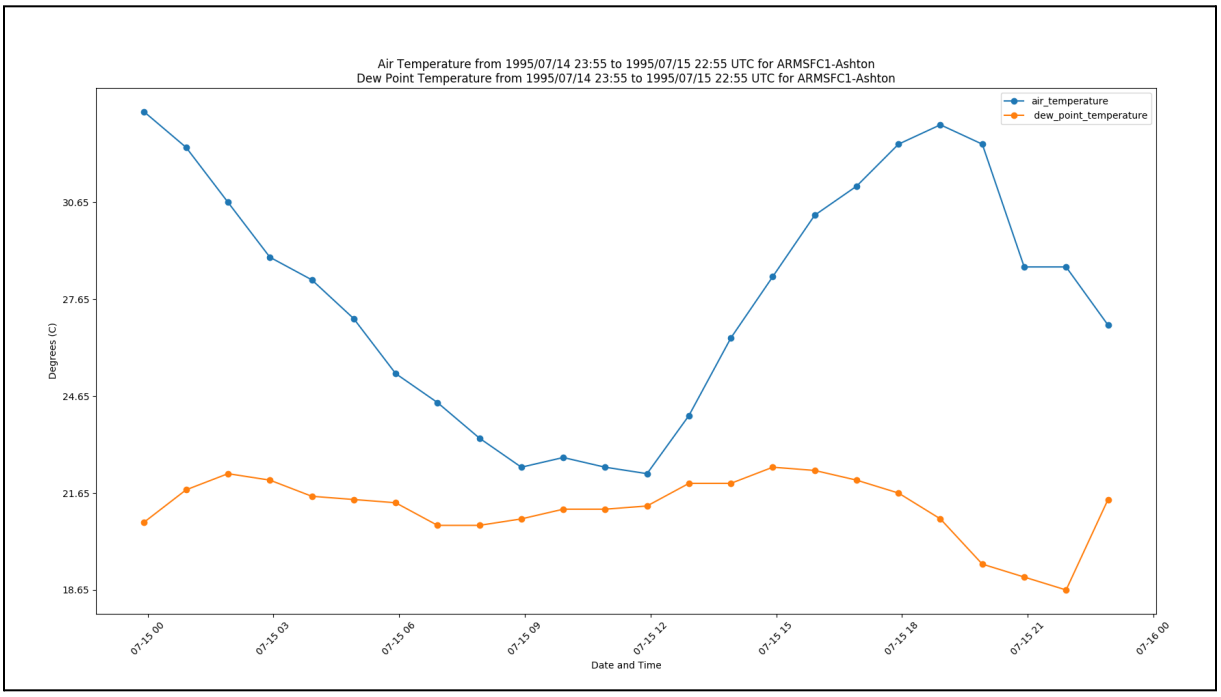

Figure 8: Example 2D plot of Air Temperature and Dew Point Temperature for ARMSFC1-Ashton

Creating 3D Plots

Changing hard-coded values

The 3-D plot script uses the following hard-coded entities that may need to be changed. Note that to locate any of these values, simply search **"HARD-CODED"** in this script.

- *missing_value*: This is a string containing the value used for missing data points
    - For our example, the missing value is “-999.99”
- *var_index*: This is the number of variables preceding the variables available to plot against time. This includes any variables containing station information, time, and location. Count the number of variables that include date, time, or location information.
    - For our example, the var_index is 9 because there are nine date, time, and location variables (date_nominal, time_nominal, date, time, network_name, platform_name, latitude, longitude, station_occurrence)
    - Figure 3 shows how to view and count these using the OPeNDAP webform page.
- *formatData*: This is a string containing format information to convert strings to the python object datetime.

    - For our example, the formatData = “%Y/%m/%d%H%:%M:%S” where Y is the year, m is the month, d is the day, H is the hour, M is the minute and S is the seconds value.
    - For more information on the datetime object and format strings, consult: https://docs.python.org/3/library/datetime.html
- *variable names*: This script assumes that the variable names from the OPeNDAP file are "date" or “date_nominal”, "time" or “time_nominal”, "latitude", "longitude".
- *Lines*: This is a boolean variable that determines whether or not the graph will have lines from the data point to the xy-axis. See Figure 13 for an example with lines.
- *Project name*: This is used for the plot title.
- *Graph elements*:
    - *graph size*: This is hard-coded into the figure declaration and declares the size of the output graph.
    - *X-label*: Set to longitude.
    - *Y-labe*l: Set to latitude.
    - *Initial view*: This is the view the plot will be produced at. For more information, see: https://matplotlib.org/2.0.2/mpl_toolkits/mplot3d/api.html
    - *Plot title*: This is set to include the project name, date, and variable.
    - *file name*: Similarly to the plot title, the file name to be saved includes the date and variable.

To change any of these values, simply search "HARD-CODED" in this script.

Calling the script from the command line:
To call this script, use the following command:

python3 python_xyz.py *OPeNDAP_url*

where *OPeNDAP_url* is the link to the dataset acquired earlier in this section. See Figure 2. Figure 9 shows the command to call this script using the GCIP/ESOP-95 Surface: Hourly Surface Meteorological Composite Dataset.

```
python3 python_xyz.py https://data.eol.ucar.edu/opendap/data/
esop_95/hrly_sfc/ES95HRLY_950715.qcf
```

Figure 9: Calling the python plot script from the command line.

Answering the prompts:
Once you call the script, you should see the prompt displayed in Figure 10 asking which variable(s) you would like to plot. If you are unsure of the variables that are available to plot, enter the word “list”. Otherwise, enter the variable(s) to plot separated by commas

as seen in Figure 10. Please note that you must enter the variables exactly as they appear in the OPeNDAP webform or variable list to avoid errors.

```
Which variable would you like to plot against space? To see a list of possible variables, ent
er 'list'. Please note that not entering the variable name as it appears within the file or O
PeNDAP webform will result in errors.

list

This file has the following variables available to plot against time:


['surface_altitude', 'surface_air_pressure', 'surface_air_pressure_flag', 'air_pressure_at_me
an_sea_level', 'air_pressure_at_mean_sea_level_flag', 'air_pressure_at_mean_sea_level_compute
d', 'air_pressure_at_mean_sea_level_computed_flag', 'air_temperature', 'air_temperature_flag'
, 'dew_point_temperature', 'dew_point_temperature_flag', 'wind_speed', 'wind_speed_flag', 'wi
nd_from_direction', 'wind_from_direction_flag', 'precipitation_amount', 'precipitation_amount
_flag', 'gust_indicator', 'wind_speed_of_gust', 'wind_speed_of_gust_flag', 'present_weather',
 'present_weather_flag', 'visibility_in_air', 'visibility_in_air_flag', 'cloud_layer_first_ba
se_altitude', 'cloud_layer_first_base_indicator', 'cloud_layer_first_base_indicator_flag', 'c
loud_layer_first_area_fraction', 'cloud_layer_first_area_fraction_flag', 'cloud_layer_second_
base_altitude', 'cloud_layer_second_base_indicator', 'cloud_layer_second_base_indicator_flag'
, 'cloud_layer_second_area_fraction', 'cloud_layer_second_area_fraction_flag', 'cloud_layer_t
hird_base_altitude', 'cloud_layer_third_base_indicator', 'cloud_layer_third_base_indicator_fl
ag', 'cloud_layer_third_area_fraction', 'cloud_layer_third_area_fraction_flag']


Enter variable to plot:
air_temperature
```

Figure 10: Answering the prompt to select variables to plot.

After successfully entering the variables to plot, you will be prompted to enter whether you would like to use actual time or nominal time for the plot as seen in Figure 11. Using nominal time will result in more data points in the plot. The nominal time is the nearest top of the hour time for the observation as compared to the actual time of the observation in most data files. Most networks actually take the observation about 5 minutes before the hour, but this varies from network to network and station to station in a network.

```
Would you like to use nominal time or actual time for the plots? Enter nominal or actual.
Please note that using nominal time will result in more data points at each time for the plot.

nominal
```

Figure 11: Answering the prompt to select either actual time or nominal time for the plot.

Next, you will be prompted to enter the date(s) and time(s) for which you want to create plots as seen in Figure 12. Again, if you are unsure of the stations available to plot, enter the word “list”. Otherwise, enter the date(s) and time(s) to plot separated by commas. For our example, we will be creating plots for July 15, 1995 at 00:00 UTC.

```
Which time(s) would you like to create 3D plots for? Enter UTC datetimes separated by commas.
 Please enter the dates and times (UTC) with the following convention:

YYYY/mm/dd-HH:MM  where YYYY = year, mm = month, dd = day, HH = hour, MM = minutes.

To see a list of possible times, enter 'list'. Please note that not entering the date and tim
es as they appear within the file will result in errors.

1995/07/15-00:00
```

Figure 12: Answering the prompt to select datetimes to plot.

Lastly, the script will prompt you for the label for the z-axis. For our example, since the variable to plot is air temperature, we will use “Degrees (C)” as seen in Figure 13.

```
What would you like the label to be on the z-axis?
Degrees (C)

Creating plots...


Plot saved in the current directory with the name 199507150000_air_temperature.png
```

Figure 13: Entering the label for the z-axis, and the output of the program.

After entering the label for the z-axis, the script will create and save the plot(s) to the current working directory. The generated 3D plot can be seen in Figure 14.

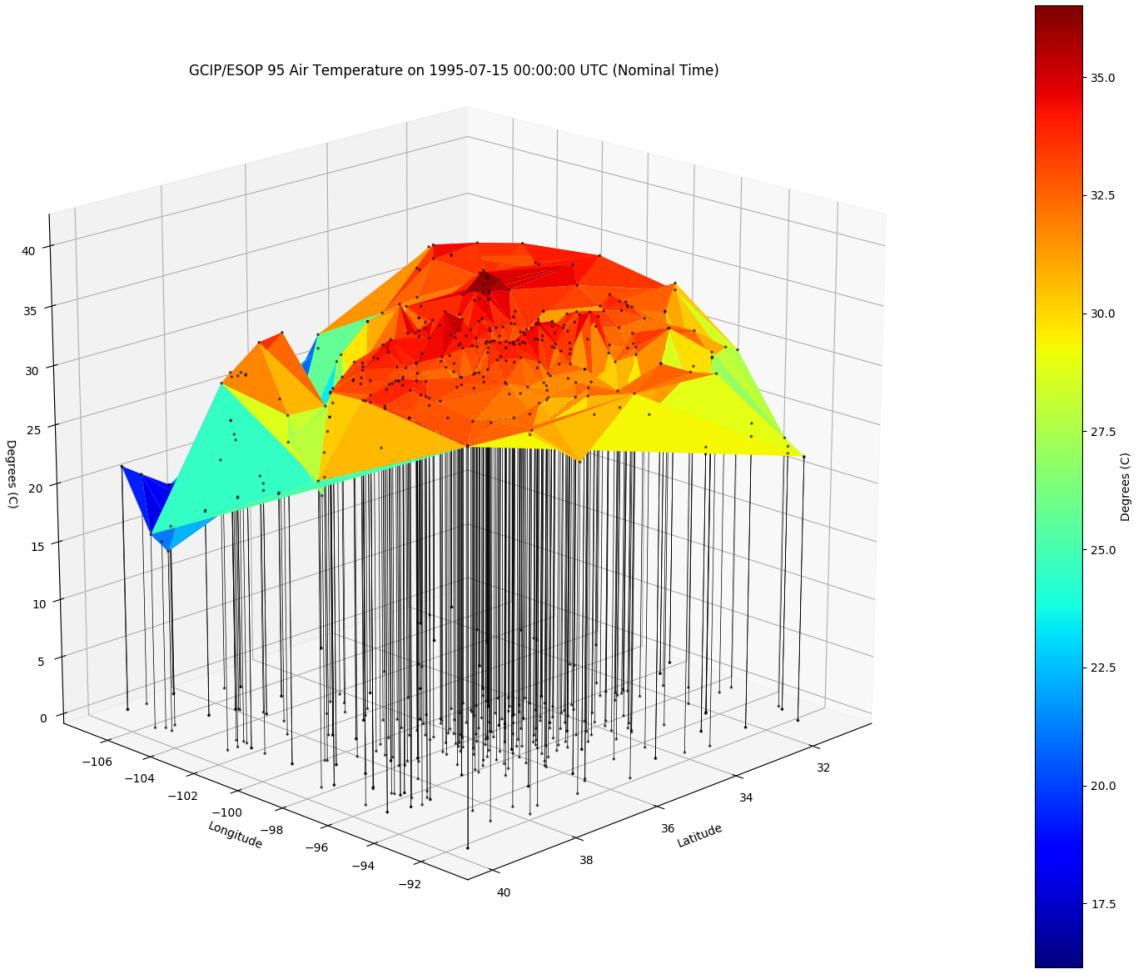

Figure 14: 3D plot of Air Temperature on July 15, 1995 at 00:00 UTC over the area of interest.