Computation, Analysis, and Visualization of In Situ and Remote Sensing Data Using Python

Jared Rennie, Andrew Buddenberg, Kelly Gassert, Ronald Leeper, Laura Stevens, Scott Stevens

Inspire. Advance. Engage.

Motivation

The demand for weather, water, and climate information has been high, with an expectation of long, serially complete observational records spanning from minute to century timescales. The completeness of these data is essential to the assessment of the Earth's climate in reports such as the Bulletin of the American Meteorological Society's (BAMS) **State of the Climate** report and the U.S. Global Change Research Program's (USGCRP) **National Climate Assessment**.

The amount of data archived and preserved at NOAA's National Climatic Data Center (NCDC) span nearly ***18 petabytes***, and it is expected to increase at an exponential rate. In addition, there has been a need for openness and transparency to ensure the integrity and reproducibility of data from original observation to final product.

Scientists at both the Cooperative Institute for Climate and Satellites–North Carolina (CICS-NC) and NCDC have begun to address these concerns using the Python open source language. Here we will present advancements and challenges of utilizing weather and climate data to meet public, scientific, sector, and archival needs.

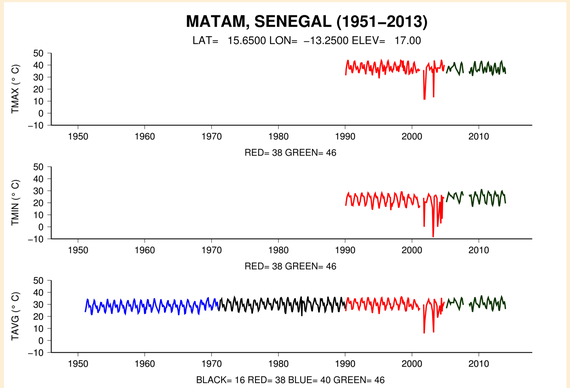
Resources

Data archived at NCDC come in many different formats, including ASCII, netCDF and GRIB. Some data are gridded, while others are individual point data. Almost all have some form of missing values.

Members of CICS-NC have access to a computing cluster with UV-CDAT version 2.0.0 (Williams et al. 2013a), that utilizes the Maui Cluster Scheduler, along with the Terascale Open-source Resource and QUEue Manager (TORQUE).

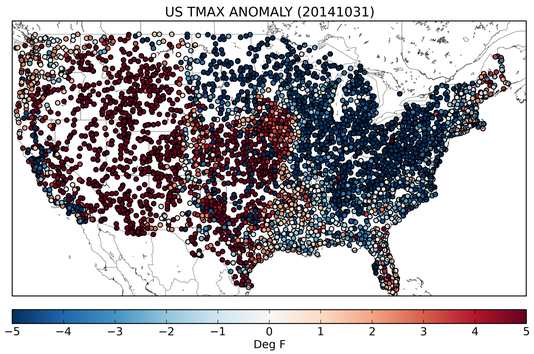
Python packages used here include Numpy, Scipy, Matplotlib, netCDF4, pygrib and Basemap.

TIME SERIES ANALYSIS

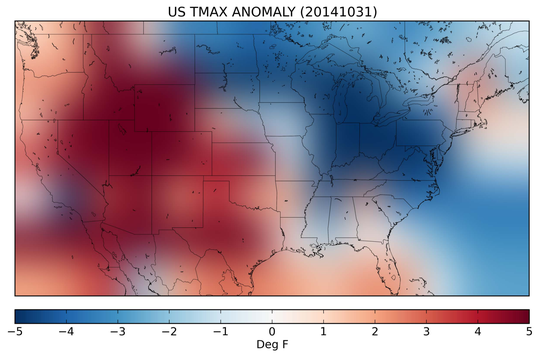


The International Surface Temperature Initiative's Global Databank (Rennie et al. 2014), pieces together long temperature time series from numerous sources. The above station failed Bartlett's test for equal variance.

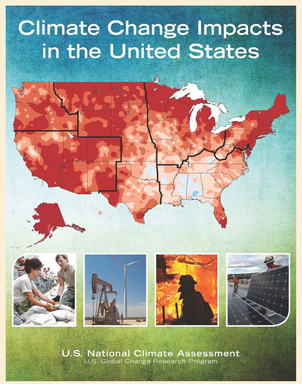
IN SITU DATA



Maximum temperature anomaly data for October 31, 2014, derived from the Global Historical Climatology Network–Daily Dataset. (GHCN-D: Menne et al. 2012). Anomalies were calculated using the 1981–2010 Normals (Arguez et al. 2012).



Spatial interpolation of maximum temperature anomaly for October 31, 2014. Point data was gridded and then interpolated using a bicubic scheme. Here, the "scipy" package was used for both the gridding and interpolation.



METADATA TRACKING

Many of the figures used in the Third National Climate Assessment Report (USGCRP 2014) were generated in Python using data from CICS-NC and NCDC.

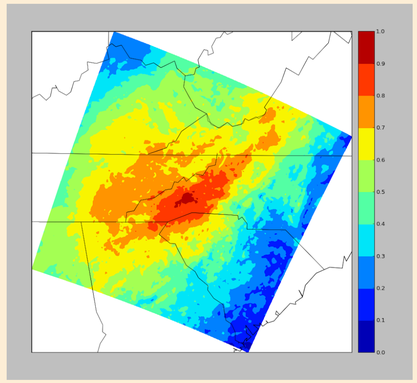
In addition, USGCRP has established the Global Change Information System (GCIS) to better track the provenance of the report using a web-based resource for traceable, sound global change data and products.

Link: **http://data.globalchange.gov**
Github: **https://github.com/USGCRP/gcis-py-client**

REMOTE SENSING DATA

Using the National Center for Environment Predictions' (NCEP's) 4 km gridded Stage IV Radar dataset, spatial correlations were made between an individual point on the map and the center of the box (for this plot, Asheville, NC). Data were read in using pygrib, Pearson's correlation was calculated, and results were contoured using the "contourf" function.

This plot explains how far away Stage IV data can be reliable from the center point. This proves very useful for assessing areas with little or no rain gauge data.



Discussion

The PyAOS resource is great for any scientist, whether an expert, or just starting in Python. It can easily interface with other languages and packages, and perhaps most important, it's open source and free! One issue is performance on large datasets. The numerical computation and memory handling in programs such as C or Fortran still perform better than utilizing Python. One example is NCDC's Pairwise Homogeneity Algorithm (PHA; Menne and Williams 2009). The PHA, written in C and Fortran, requires 48 GB of RAM.

Future work includes connecting NCDC's data through its Historical Observing Metadata Repository (HOMR). This will be done using the requests and JSON packages. We also plan to work with Google to integrate our data within their Earth Engine. Finally, we hope to increase awareness of Python to the CICS-NC and NCDC community and encourage future projects to be coded in Python.

More Information

This study was supported by NOAA through the Cooperative Institute for Climate and Satellites–North Carolina (CICS-NC) under Cooperative Agreement NA09NES4400006.

This Poster: **http://goo.gl/bTzXNv**

E-Mail: **jared@cicsnc.org** | Twitter: **@jjrennie**