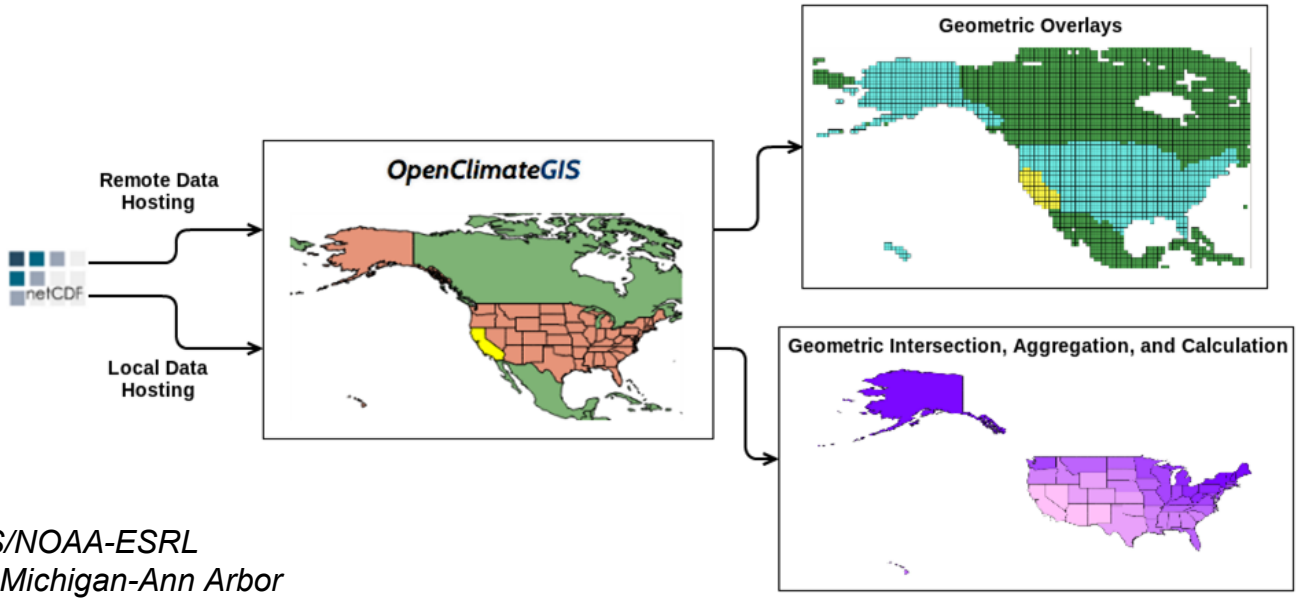
OpenClimateGIS: A Python Library for Geospatial Manipulations of CF Climate Datasets

Ben Koziol[1], Ryan O'Kuinghttons[1], Richard Rood[2], Cecelia DeLuca[1], Sylvia Murphy[1]

December, 2015



1. NESII/CIRES/NOAA-ESRL
2. University of Michigan-Ann Arbor

Presentation Outline

1. Overview of OpenClimateGIS
2. Subsetting
3. UV-CDAT & ESMPy Integration
4. Next Steps

What is OpenClimateGIS?

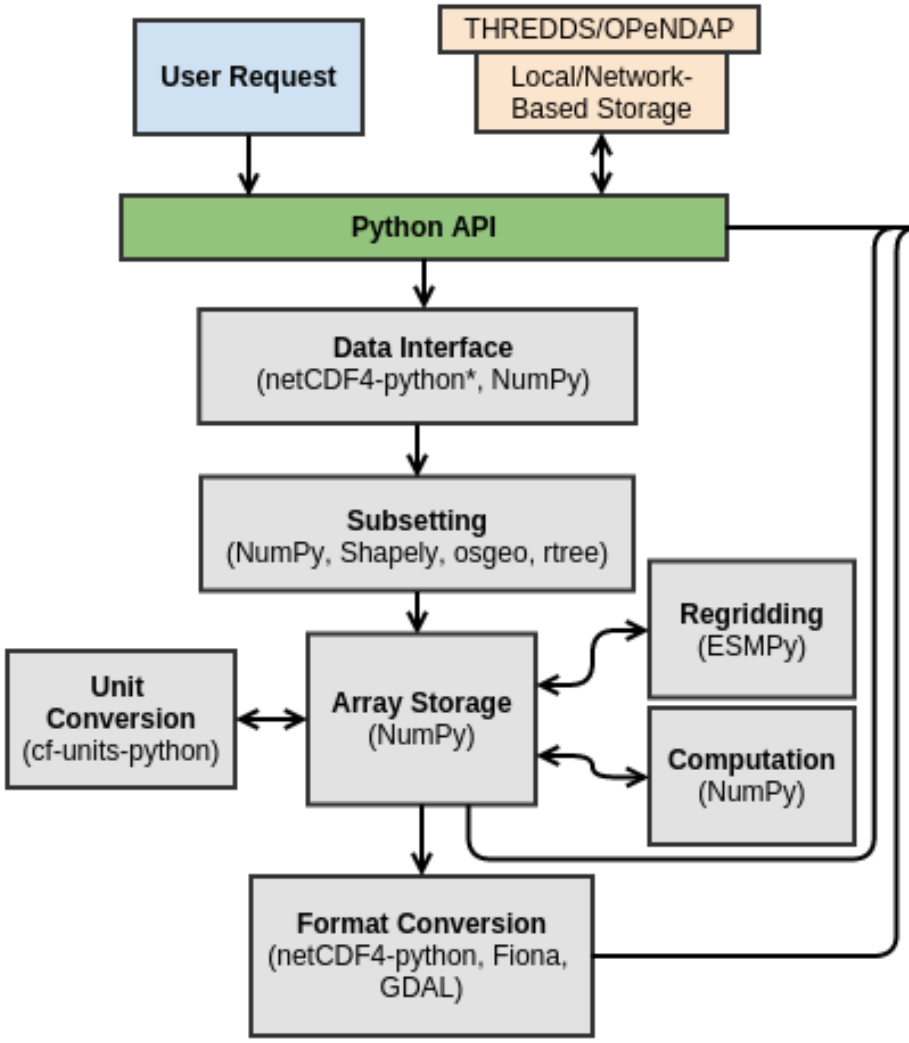
- OpenClimateGIS (OCGIS) is a standalone, Python-based, open source software library enabling dynamic access to and manipulation of climate data
- Software goal is to overcome barriers of usability of climate projections in adaptation planning and resource management
  - Translate out of climate data formats
  - Select geographical regions of interest
  - Select times/levels of interest
  - Compute application-relevant indices
  - Convert to end-user and analysis-ready formats
  - Provide comprehensive metadata
- Builds on numerous open source software libraries:

Required	Optional
netCDF4 numpy shapely fiona osgeo	rtree cfunits ESMF

Status

- Current Release: 1.01
- Project is fully open source under the University of Illinios-NCSA License (http://opensource.org/licenses/NCSA)
- Hosted on GitHub: https://github.com/NCPP/ocgis
  - Documentation as well: http://ncpp.github.io/ocgis/

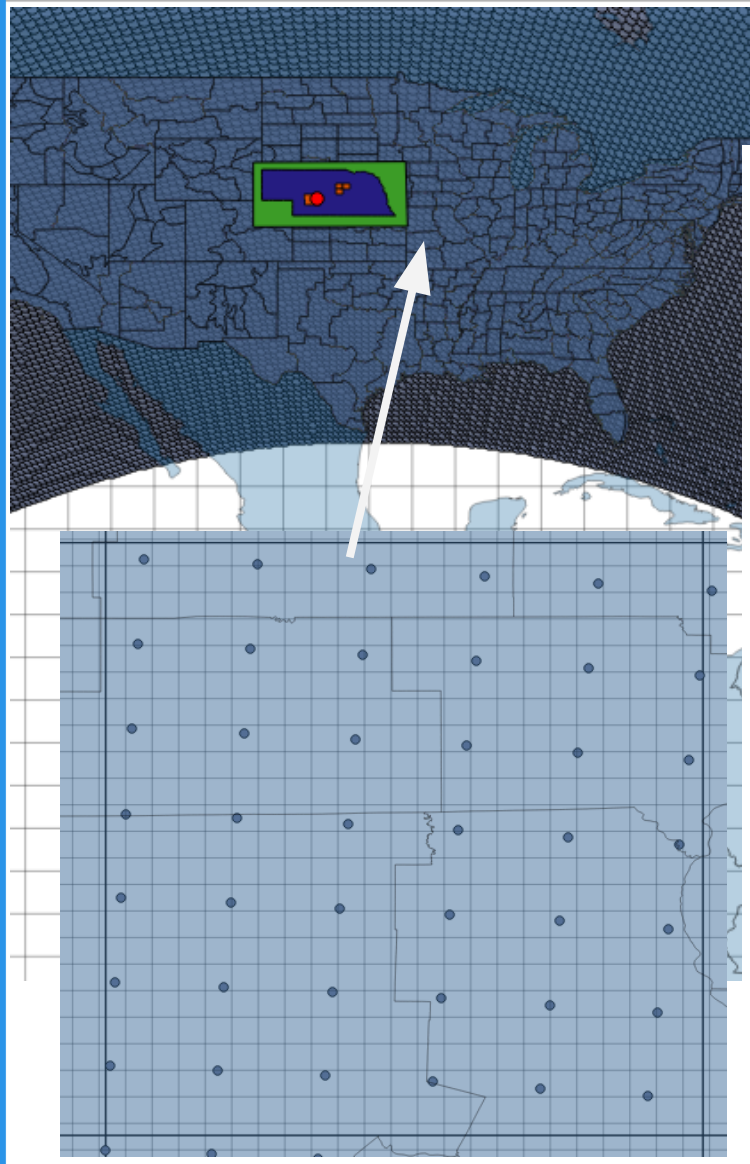
Software Architecture



* This is the library used for reading NetCDF. Depending on the input data format, other data APIs may be desirable (i.e. OGR for vector formats).

- Written in pure Python (2.7.x currently supported)
- Modular design for data interface, format conversion, and computations
- To the extent possible, operations use metadata to limit the amount of “value data” requested
- Built with generator functions at the operations API
- Implemented in serial - tiling functionality available for large array operations and large OPeNDAP requests

Subsetting



- Handles many types of geospatial subsetting:
  - Points
  - Arbitrary Polygons
  - Bounding Boxes
  - Collections of Points and Polygons
- Reads geometries directly from ESRI Shapefiles, point/bounding box sequences, Shapely geometry objects
- Temporal subsetting - time ranges or “regions” (i.e. arbitrary month and year combinations)
- Level subsetting - lower and upper bounds
- Reads and writes CF and PROJ.4 coordinate reference systems
- Wrapping and unwrapping for 360 geographic coordinate systems

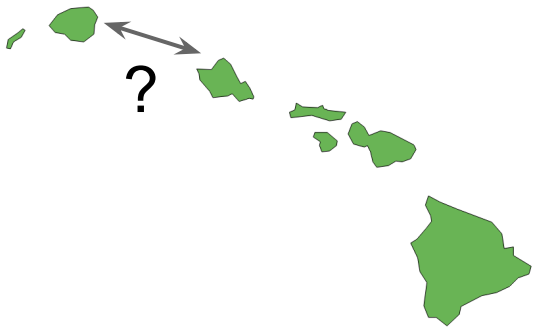
Quick Subset Example

```
import ocgis


ops = ocgis.OcgOperations(dataset={'uri':'/data/tas_kelvin.nc'},
                          time_region={'month': [6, 7, 8]},
                          geom=[-121, 38, -122, 40],
                          conform_units_to='celsius',
                          output_format='nc')
path = ops.execute()
```

ESMF/ESMPy Integration

- Currently support first-order bilinear and conservative regridding for rectilinear grids
- Moving towards full interoperability with ESMPy fields:
  - Add support for n-sided meshes (ESMPy 7.x)
  - Bring spatial operations into ESMPy - ESMPy (7.x) has support for n-dimensional arrays
  - Have preliminary implementation in feature branch allowing ESMPy field to be read and written similar to netCDF-CF datasets
- Coordinate systems are tricky:
  - Need to develop classification scheme to ensure appropriate determination of spherical v. planar
- Topology is also tricky:
  - How to handle Hawaii?
  - Connect vector GIS with netCDF



UV-CDAT Integration

- A logical entry point to access GIS operations in UV-CDAT is through ESMPy - existing interfaces may be reused?
- ESMPy-OCGIS is not finalized
- Important features to target for ESMPy API:
  - ESRI Shapefile I/O
  - Subsetting
  - Coordinate system transformations/remapping

Next Steps

- ESMPy → UV-CDAT
- ClimatePipes
- IS-ENES climate4impact & ICCLIM
- “PyData” → GeoPandas, xray
- Python 3.x, pip install

Contacts & Links

- Questions, comments, suggestions, or “hidden features”:
  - ocgis_support@list.woc.noaa.gov
- Mailing lists and releases:
  - ocgis_info@list.woc.noaa.gov
- Software links:
  - http://www.earthsystemcog.org/projects/openclimategis/
  - http://www.earthsystemcog.org/projects/downscaling-2013/climatetranslator
  - http://www.earthsystemcog.org/projects/esmpy/
  - http://www.esrl.noaa.gov/nesii/

Backup Slides

Computation

- Framework designed to accommodate a variety of climate indices and metrics:
  - **Temporally grouped functions** → monthly means, annual maximums, durations
  - **String-based functions** → 'diff=tasmax-tasmin'
  - **Simple transforms** → natural logarithm
  - **Multivariate functions** → heat indices
- Goal is to provide a simplified method for introducing new indices and a straightforward, timely method for documentation (currently works with the Sphinx Python documentation system)

class ocgis.calc.library.index.heat_index.**HeatIndex**(**args*, ***kwds*) [source]

Bases: ocgis.calc.base.AbstractMultivariateFunction, ocgis.calc.base.AbstractParameterizedFunction

calculate(*tas=None*, *rhs=None*, *units=None*) [source]

```
dtype = np.float32
parms_definition = {'units':str}
required_variables = ['tas','rhs']
key = 'heat_index'

def calculate(self,tas=None,rhs=None,units=None):
    if units == 'k':
        tas = 1.8*(tas - 273.15) + 32
    else:
        raise(NotImplementedError)

    c1 = -42.379
    c2 = 2.04901523
    c3 = 10.14333127
    c4 = -0.22475541
    c5 = -6.83783e-3
    c6 = -5.481717e-2
    c7 = 1.22874e-3
    c8 = 8.5282e-4
    c9 = -1.99e-6

    idx = tas < 80
    tas.mask = np.logical_or(idx,tas.mask)
    idx = rhs < 40
    rhs.mask = np.logical_or(idx,rhs.mask)

    tas_sq = np.square(tas)
    rhs_sq = np.square(rhs)

    hi = c1 + c2*tas + c3*rhs + c4*tas*rhs + c5*tas_sq + c6*rhs_sq + \
         c7*tas_sq*rhs + c8*tas*rhs_sq + c9*tas_sq*rhs_sq

    return(hi)
```

Dataset Bundling

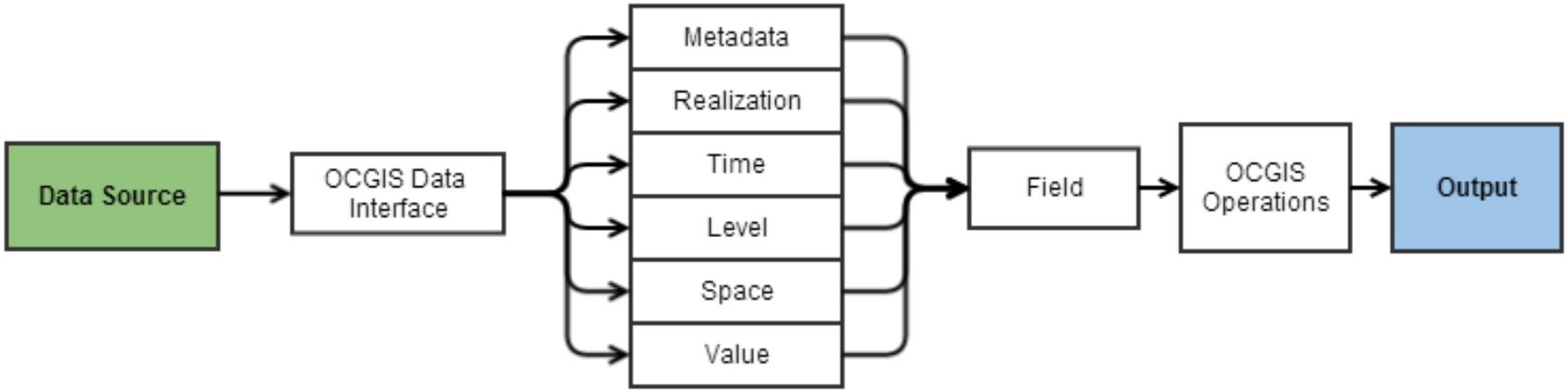
- Bundles or packages are groups of data over which to apply a common set of operations → idea is to extend ensembles
- OCGIS consolidates coordinate systems for the datasets and subset geometry(s) and applies selected operations to each in sequence
- The example data displayed below is from a CSV output from three datasets:
  a. CMIP5 Decadal Simulation (3 degrees, 360 lat/lon)
  b. NARCCAP CRCM-CGCM3 (50 km, Polar Stereographic)
  c. Maurer Gridded Observational (⅛ degrees, 180 lat/lon)
- Example description:
  a. Pull out all January dates
  b. Spatially subset and area-weight the values for grid cells intersecting the Nebraska state boundary
  c. Calculate the monthly mean and standard deviation
  d. Write data to CSV

VARIABLE	ALIAS	CALC_KEY	CALC_ALIAS	TIME	YEAR	MONTH	DAY	LEVEL	VALUE
tas	cmip5_tas	mean	mean_cmip5_tas	2001-01-16 00:00:00	2001	1	16		271.785
tas	cmip5_tas	std	stdev_cmip5_tas	2001-01-16 00:00:00	2001	1	16		5.59953
pr	narccap_pr	mean	mean_narccap_pr	1981-01-16 00:00:00	1981	1	16		1.0143e-05
pr	narccap_pr	mean	mean_narccap_pr	1981-02-16 00:00:00	1981	2	16		4.5889e-06
pr	narccap_pr	std	stdev_narccap_pr	1981-01-16 00:00:00	1981	1	16		2.38535e-05
pr	narccap_pr	std	stdev_narccap_pr	1981-02-16 00:00:00	1981	2	16		3.57291e-06
tasmax	maurer_tasmax	mean	mean_maurer_tasmax	1971-01-16 00:00:00	1971	1	16		1.37631
tasmax	maurer_tasmax	std	stdev_maurer_tasmax	1971-01-16 00:00:00	1971	1	16		7.20884

Format Conversion

- A general framework for data conversion allows data to be streamed to multiple formats (it is not terribly difficult to add output formats)
- There is a common set of headers for output files that may be adjusted to suit a user's needs (i.e. a user may only be interested in a timestamp and associated data value)
- OCGIS takes advantage of Fiona to write to OGC-compliant vector formats (i.e. ESRI Shapefile)
- In addition to the data and dimensional values, metadata is also maintained
- Currently supported formats: CSV, Key-CSV Shapefile (CSV+), GeoJSON, netCDF, ESRI Shapefile, array-based

Extensibility



Example Calculation Subclassing

```
class Min(base.AbstractUnivariateSetFunction):
    description = 'Min value for the series.'
    key = 'min'
    dtype = constants.np_float

    def calculate(self,values):
        return(np.ma.min(values,axis=0))

class Max(base.AbstractUnivariateSetFunction):
    description = 'Max value for the series.'
    key = 'max'
    dtype = constants.np_float

    def calculate(self,values):
        return(np.ma.max(values,axis=0))
```

Example NetCDF Data Reading

```
class NcVectorDimension(VectorDimension):

    def _set_value_from_source_(self):
        ## open the connection to the real dataset connect
        ds = self._data._open_()
        try:
            ## get the variable
            try:
                var = ds.variables[self.meta['name']]
            except KeyError as e:
                ## for the realization/projection axis, th
                ## value associated with it. in it's place
                ## array.
                if self._axis == 'R':
                    var = self._src_idx + 1
                else:
                    ocgis_lh(logger='interface.nc',exc=e)
            # format the slice
```

NCPP / NESII Group Overview

NESII

- **NCPP Mission:** To advance the development of standards, tools, and information that support the choice, interpretation, and use of climate change data in adaptation planning and resource management.
- NESII builds software infrastructure for Earth system modeling, data analysis, and scientific collaboration using open source, community development approaches
- NESII has been at ESRL / CIRES since November, 2009 - formerly the Earth System Modeling Infrastructure section at the National Center for Atmospheric Research
- Partners and customers are from research and operational centers, weather and climate, across U.S. agencies and international organizations

ESMF
ESMPy
NUOPC
Cupid
earth system es-doc documentation
OpenClimateGIS
ClimateTranslator

ClimateTranslator Web Interface

- NCPP has release an initial version of the ClimateTranslator which OCGIS as the processing backend
- The workflow-based web interface exposes much of the OCGIS functionality for data manipulations

NCPP (National Climate Predictions and Projections Platform)

ClimateTranslator Workflow: Step 1 of 3

Disclaimer: the ClimateTranslator and underlying OpenClimateGIS software are *beta* versions.

Data Selection

Please select a *dataset category*, a *dataset*, and a *variable*
(when selecting a *dataset package*, a *variable* selection is not necessary).

Dataset Category -- Please Select --

Dataset -- Please Select --

Variable -- Please Select --

Geo-Spatial Selection

Optionally, you may select either a *shape* geometry, a *bounding box*, or a *point*.
If no selection is made,the full dataset geo-spatial extent will be processed.

Shape: Type: -- Please Select -- Geometry: -- Please Select --

Bounding Box: Latitude min: max: Longitude min: max:

Point: Latitude: Longitude:

Temporal Selection

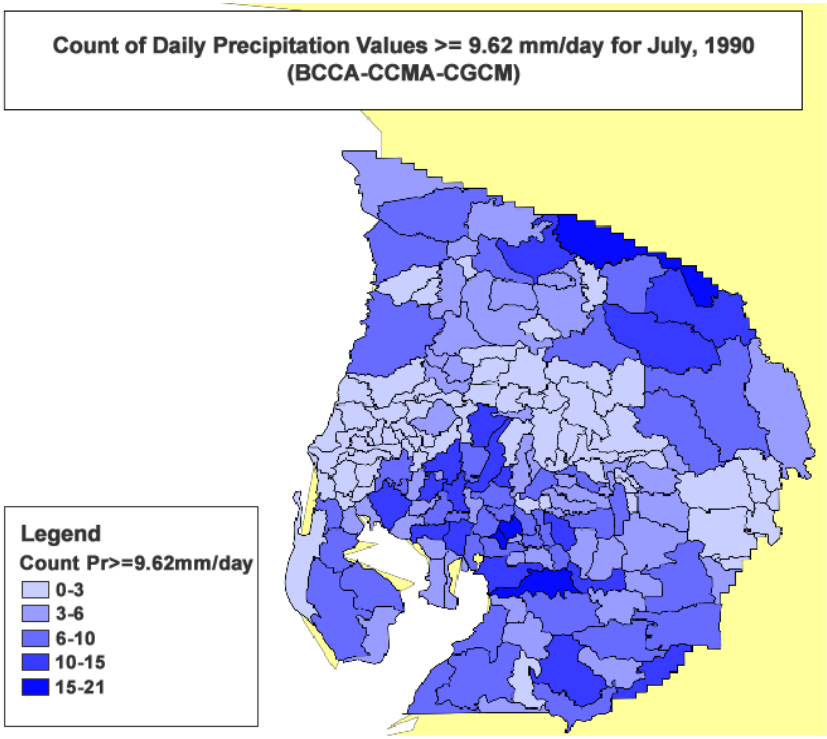
Optionally, you may specify a *time range* (i.e. a start and stop time), and/or a *time selection* (one or more months, and/or one or more years). If both are specified, the *time range* must contain the *time selection*.
If neither *time range* or *time selection* is specified,the full dataset temporal extent will be processed.

Time Range: Start **Stop** [Format: YYYY-MM-DD HH:MM:SS]

Time Selection: Months Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec

Time Selection: Years [Format: YYYY, YYYY, ... or YYYY-YYYY]

Next >

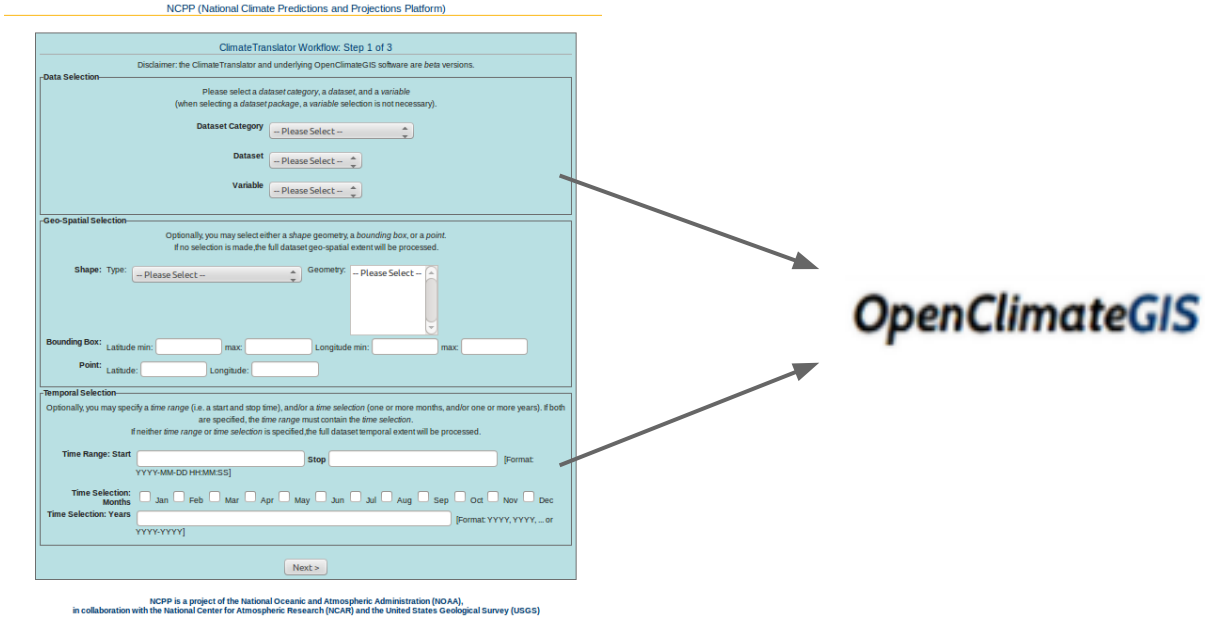


About ClimateTranslator and OpenClimateGIS

- Overcome barriers of usability of climate projections in adaptation planning and resource management
  - Translate out of climate data formats
  - Select geographical regions of interest
  - Select times of interest
  - Compute application-relevant indices
  - Translate into end-user and analysis-ready formats
- ClimateTranslator is a web framework developed by NCPP designed to simplify access to climate information
- The ClimateTranslator web interface is based on the OpenClimateGIS (OCGIS) Python toolkit - a standalone, open source software library enabling dynamic access to and manipulation of climate data

The ClimateTranslator Web Interface

- The ClimateTranslator web interface uses OCGIS on the backend formatting user selections and executing a single call to OCGIS
- Functionality from the interface is available in the standalone OCGIS library
- A note on versions:
  - OCGIS is in beta
  - ClimateTranslator web interface is in alpha



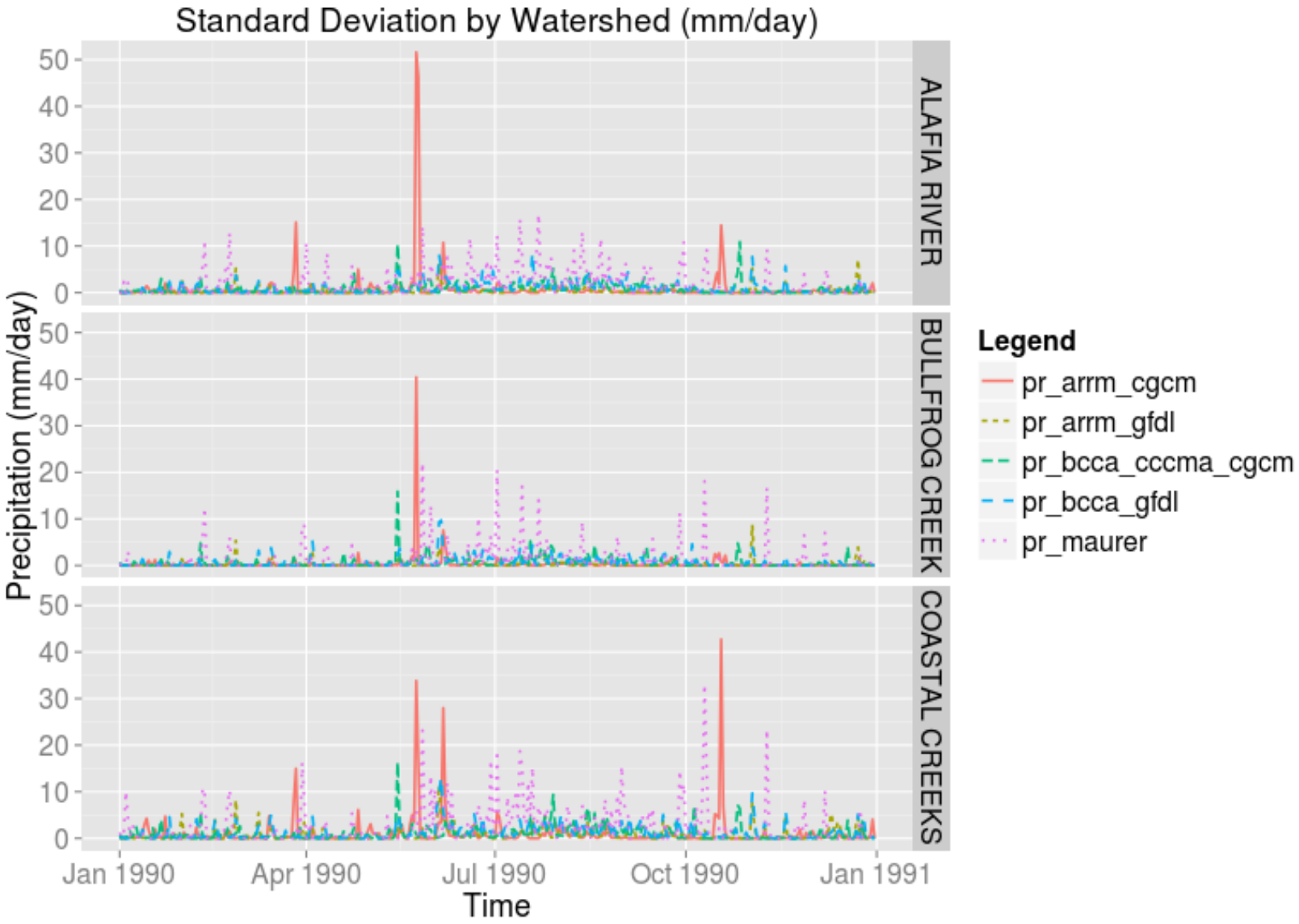
Application/Demo Outline

1. Extract a daily precipitation time series for all Tampa Bay watersheds for the year 1990
2. Base data will be the precipitation variable pulled from four downscaled and one observational dataset
3. Inspect output data and auxiliary files
4. Examine time series plot generated in R

Live Demo

http://hydra-ncpp.fsl.noaa.gov/ncpp/open_climate_gis/
user: ncppuser
password: qed2013

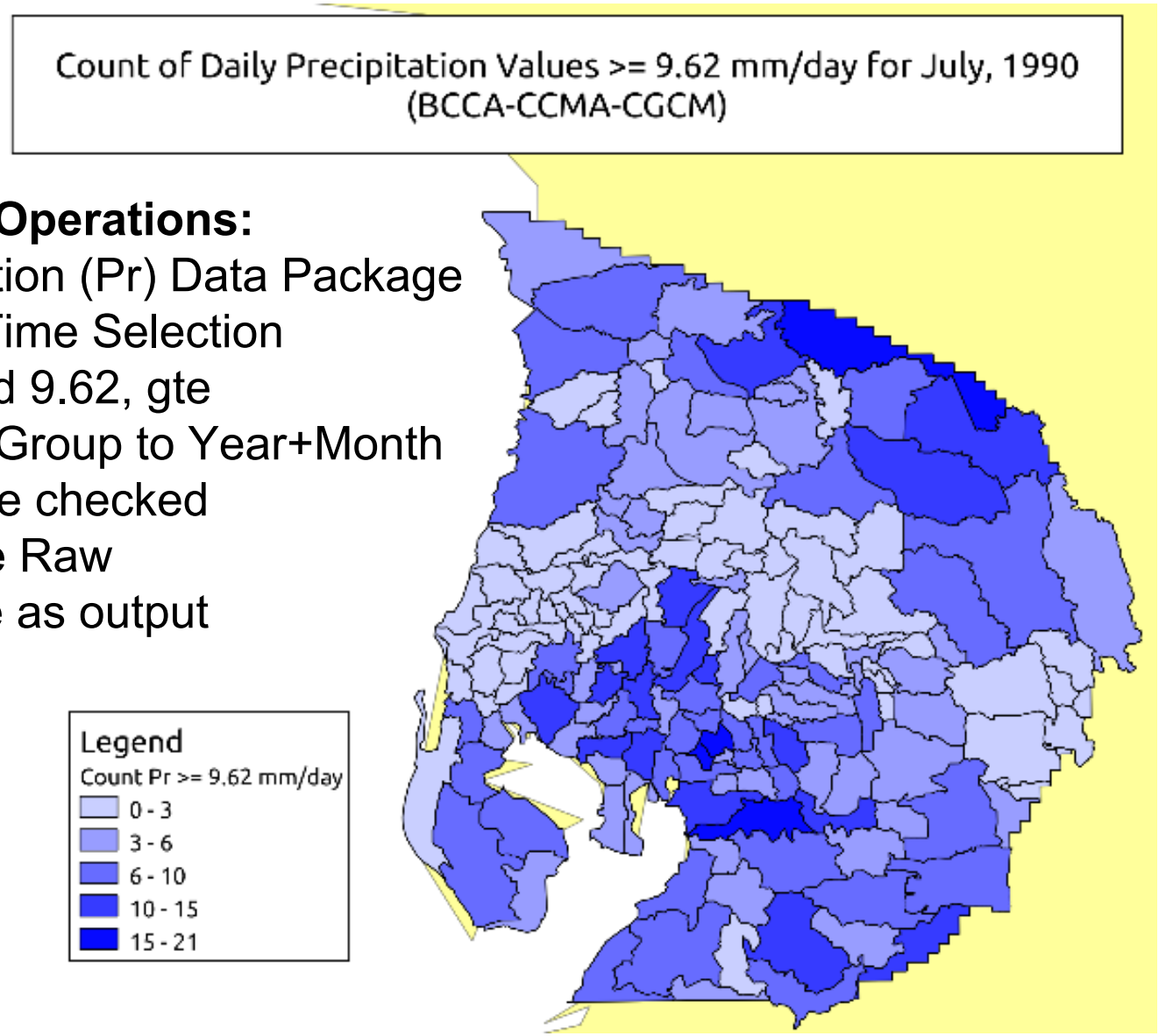
R Plot for Precipitation Time Series



Precipitation Threshold for Tampa Bay Watershed Basins

ClimateTranslator Operations:

1. Select Precipitation (Pr) Data Package
2. Select 1990 in Time Selection
3. Select Threshold 9.62, gte
4. Set Calculation Group to Year+Month
5. Leave Aggregate checked
6. Check Calculate Raw
7. Select Shapefile as output



Summary

- Demonstrated one path through the workflow:
  - Spatial and temporal subsetting of multiple climate datasets in a single request
  - Subsetted data pushed to one tabular output file
  - Output data quickly ingested by R or GIS to generate derivative products
- Many other potential paths depending on the end-user's data needs (i.e. compute thresholds or percentiles)
- Flexible and adaptable workflow model not tied to a single data source, region, or agency
- Software and framework may be integrated with a variety of specialized climate data systems

Future Work

- OpenClimateGIS:
  - Expand calculation library
  - Unit-aware conversions
  - Integration with a regridding package to generate a wider variety of gridded outputs
  - Probabilistic data outputs - anticipate needs of ensemble-based, application-oriented evaluations
- ClimateTranslator web interface:
  - Incorporate translational metadata
  - Web mapping and visualization capabilities
  - Custom configuration of data packages
  - Ability to apply multiple calculations as well as multivariate calculations (e.g. more than one dataset)
  - Improved help functionality and aesthetics

Core Capabilities of OpenClimateGIS

- Read local or remotely served (i.e. OPeNDAP) ~CF-compliant netCDF datasets
- Geospatial subsetting by arbitrary vector geometries (e.g. watersheds) and time/level bounds
- Common spatial operations such as intersects, clip, and aggregation on point or polygon (e.g. bounded coordinates) data representations
- Geometry wrapping and unwrapping to maintain a “GIS-friendly” -180 to 180 longitudinal spatial domain
- Support for geographic (e.g. latitude/longitude) and projected climate datasets (e.g. Lambert Conformal)
- Option to apply temporally-grouped computations to data subsets
- Write climate data to GIS and tabular formats