OpenClimateGIS (O CGIS) v0.02a
Overview & Demo
Presented at ESGF f2f, Nov 2012

Ben Koziol (NESII/CIRES/NOAA-ESRL)
ben.koziol@noaa.gov
Presentation Outline

1. Code location and important links
2. Software overview
3. API overview
4. Demo
5. Future work

Also acknowledge Tyler Erickson (Google), K. Arthur Endsley (Michigan Tech Research Institute), Ricky Rood (UM), GLISA
1. Code location and important links

- Current version tag is v0.02a
- Code is available on GitHub: https://github.com/NCPP/ocgis
- Documentation and examples evolving on the OCGIS CoG workspace: http://www.earthsystemcog.org/projects/openclimategis/
- Feel free to use the GitHub issue tracker or email ben.koziol@noaa.gov with questions or comments
2. Software Overview

- Python v2.7 + Python/C Dependencies
- Subsetting of spatially explicit CF climate datasets using arbitrary vector geometries or bounding boxes, time, and level (variable dependent) ranges
- Perform calculations on subsetted data in optionally grouped by temporal categories
- Output climate data to common vector, gridded, and tabular formats (i.e. shapefile, csv, nc)
  - Straightforward to add additional output formats
- Operates as a standalone geoprocessing library or behind a RESTful web service
Climate Dataset Compatibility

- CF-compliant NetCDFs and OPeNDAP datasets
  - An accessible URI (universal resource identifier) is required for the dataset
    - Local or remote storage
- More generally, data must have spatial, temporal, and level (if applicable) dimensional variables
  - Spatial variables assumed latitude/longitude (WGS84 datum) or must contain sufficient metadata to properly define projection
  - Bounded latitudes and longitudes ideal
  - Point geometries are supported
  - Temporal variable must have common calendar and unit definitions
- OCGIS will "auto-discover" variable names but naming conventions may require minor code modifications
  - Interface parameters may be overloaded as arguments
- Data masks are respected (and updated) throughout processing chain
Geometric Operations

- Sources of geometry bounds:
  - Bounding box
  - Geometries pulled from shapefiles stored in OCGIS-referenced directory structure
  - Point-based requests currently not supported (can be hacked with buffering)
- Intersects and clip (intersection) spatial operations
  - Support multi-geometries tagging coincident data with geometry identifier
- Prototype spatial index and geometry wrapping for overlays and vector output (-180/180 to 0/360)
- For aggregation, weights determined by cell area normalized by maximum area in the geometry set
  - Raw (pre-weighting) and aggregated values stored
  - Point geometries simply assigned 1.0

https://github.com/NCPP/ocgis/blob/master/src/ocgis/util/shp_cabinet.py
Calculations

- Calculations are objects with an associated algorithm implemented as a static method
  - Class attributes provide metadata (i.e. key, long name, description, argument definitions)
- Calculation must be aware of the correct data reduction procedure
  - Point-wise or aggregated
- Data slices passed to the calculation method are determined by temporal groups composed of any combination of day, month, or year (other groupings desirable)
- Uni- and multi-variate calculations supported
- Current calculation framework is sufficient but opportunities exist for a more extensible and performance-oriented architecture

https://github.com/NCPP/ocgis/blob/master/src/ocgis/calc/library.py
3. API Overview

- Single entry point to request execution through Python “operations” object
  - Supports argument validation, etc.
  - Update of operational definitions transfers to RESTful API (add an argument it is immediately exposed as a RESTful parameter)
- RESTful API uses a slug to specify major operational category (i.e. subset)
  - Additional arguments appended as query parameters (key-value pairs)

```python
from definition import * #UnusedWildImport
class OcgOperations(object):
    def __init__(self, dataset=None, spatial_operation=None, geom=None, aggregate=None,
                 time_range=None, level_range=None, calc=None, calc_grouping=None,
                 calc_raw=None, interface=None, snippet=None, backend=None, request_url=None,
                 prefix=None, output_format=None, output_grouping=None, agg_selection=None,
                 **kwds):

from ocgis.api.operations import OcgOperations
from ocgis.api.interpreter import OcgInterpreter

kwds = {'dataset': 'http://some.opendap.dataset.nc',
        'variable': 'tas',
        'spatial_operation': 'clip',
        'aggregate': True}
ops = OcgOperations(**kwds)
ret = OcgInterpreter(ops).execute()
```

```
subset?variable=tas&geom=-127|40|-120|42&time_range=2000-1-1|2009-12-31&uri=http://some.opendap.dataset
```
4. “Demo”

- Demonstration of RESTful URL format (small subset of parameters and combinations)
- Working with two ESGF-hosted OPeNDAP datasets:
  - CLT – total cloud fraction (MODIS-derived), global coverage
  - TA – temperature (AIRS-derived), global coverage (land masked)
- URIs for datasets will not be displayed to save space
Selection Geometry Download

/shp/state_boundaries (1.3 sec)

/shp/state_boundaries?select_ugid=25&prefix=california (0.05 sec)

/shp/world_countries (1.1 sec)
Dataset Inspection


=== Temporal =============

Start Date = 2000-03-16 12:00:00
End Date = 2011-09-16 00:00:00
Calendar = standard
Units = days since 2000-3-01
Resolution (Days) = 30
Count = 139

=== Spatial ==============

Spatial Reference = WGS84
Extent = (-180.0, -90.0, 180.0, 90.0)
Interface Type = SpatialInterfacePolygon
Resolution = 1.99721448468
Count = 64800

=== Level ================

No level dimension found.

=== Dump =================

++ GLOBAL ATTRIBUTES ++
- institution :: National Aeronautics and Space Administration, Goddard Space Flight Center
- institute_id :: NASA-GSFC
- experiment_id :: obs
- source :: MODIS MOD08_M3 Cloud Fraction Mean Mean output prepared for CMIP5 historica
- model_id :: Obs-MODIS
- contact :: Steven A. Ackerman (stevea@ssec.wisc.edu) Steven E. Platnick (steven.e.platnick@nasa.gov)
- tracking_id :: ce60d0eb-2b5d-4f9d-9c82-c3a6dc4db3d7
- instrument :: MODIS
- obs_project :: MODIS
- source_type :: satellite_retrieval
- data_structure :: grid
- processing_level :: L3
- processing_version :: C5
- mipspecs :: CMIP5
- realm :: atmosChem
- source_id :: MODIS
- product :: observations
- frequency :: mon
Snippets

/snippet?s_abstraction=point&variable=ta&uri=…&prefix=snippet_ta_point (13.0 sec)

/snippet?s_abstraction=polygon&variable=clt&uri=…&prefix=snippet_clt_polygon (46.6 sec)

/snippet?variable=clt&uri=…&geom=state_boundaries&prefix=snippet_states (9.0 sec)

/snippet?variable=clt&uri=…&geom=state_boundaries&select_ugid=25&prefix=snippet_california (2.9 sec)

/snippet?variable=clt&uri=…&geom=world_countries&prefix=snippet_world (135.7 sec)
Subsetting

/subset?variable=clt&geom=-127.79297|32.24997|-112.98340|42.29356&prefix=bb_ca&uri=... (4.3 sec)

/subset?variable=clt&geom=-127.79297|32.24997|-112.98340|42.29356&prefix=bb_ca&spatial_operation=intersects&time_range=none&level_range=none&output_format=keyed&uri=...
Subsetting w/ Calculation

/subset?variable=clt&uri=...&time_range=2000-1-1|2000-12-31&spatial_operation=clip&aggregate=true&geom=state_boundaries&output_format=meta&calc=mean~mean_clt|min~min_clt|max~max_clt|std~std_clt&calc_raw=false&calc_grouping=year

/subset?...&output_format=shp (65.7 sec)
Requested URL:
http://testserver/subset?variable=clt&uri=http://esg-datanode.jpl.nasa.gov/thredds/dodsC/esg_dataroot/obs4MIPs/observations/atmos/clt/mon/grid/NASA-GSFC/MODIS/v20111130/clt_MODIS_L3_C5_200003-201109.nc&time_range=2000-1-1%7C2000-12-31&spatial_operation=clip&aggregate=true&geom=state_boundaries&output_format=meta&calc=mean~mean_clt%7Cmin~min_clt%7Cmax~max_clt%7Cstd~std_clt&calc_raw=false&calc_grouping=year

++++ Parameter and Slug Definitions ++++

Parameter descriptions for an OpenClimateGIS call based on operational dictionary values. The key/parameter names appears first in each "===" group with the name of URL-encoded slug name if it exists ("None" otherwise).

snipet=Flase

All data returned.

calc=[{'ref': <class 'ocgi...<truncated>

* n :: Statistical sample size.

* mean_clt :: Mean value for the series.

* min_clt :: Min value for the series.

* max_clt :: Max value for the series.

* std_clt :: Standard deviation for the series.

++++ Potential Header Names and Definitions ++++

CALC_NAME :: User-supplied name for a calculation.

CID :: Unique identifier for a calculation name.

DAY :: Day extracted from time string.

GID :: Geometry identifier assigned by OpenClimateGIS to a dataset geometry. In the case of "aggregate=True" this is equivalent to "UGID".

LEVEL :: Level name.

LID :: Level identifier unique within a variable.

MONTH :: Month extracted from time string.

TGID :: Unique grouped time identifier.

TID :: Unique time identifier.

TIME :: Time string.

UGID :: User geometry identifier pulled from a provided set of selection geometries. Reduces to "1" for the case of no provided geometry.

VALUE :: Value associated with a variable or calculation.

VARIABLE :: Name of request variable.

VID :: Unique variable identifier.

VLID :: Globally unique level identifier.

YEAR :: Year extracted from time string.
netcdf clt_usa {

dimensions:
  d_time = 139;
  d_lat = 54;
  d_lon = 113;
  d_bnds = 2;

variables:
  double time(d_time);
  time:calendar = "standard";
  time:units = "days since 2000-3-01";
  double lat(d_lat);
  lat:projection = "+proj=longlat +datum=WGS84 +no_defs ";
  double lon(d_lon);
  lon:projection = "+proj=longlat +datum=WGS84 +no_defs ";
  double lat_bnds(d_lat, d_bnds);
  lat_bnds:projection = "+proj=longlat +datum=WGS84 +no_defs ";
  double lon_bnds(d_lon, d_bnds);
  lon_bnds:projection = "+proj=longlat +datum=WGS84 +no_defs ";
  float clt(d_time, d_lat, d_lon);
  clt:FillValue = 1.e+20f;
  clt:fill_value = 1.e+20;

data:
  time = 15.5, 46, 76.5, 107, 137.5, 168.5, 199, 229.5, 260, 290.5, 321.5,
          351, 380.5, 411, 441.5, 472, 502.5, 533.5, 564, 594.5, 625, 655.5, 686.5,
          716, 745.5, 776, 806.5, 837, 867.5, 898.5, 929, 959.5, 990, 1020.5,
          1051.5, 1081, 1110.5, 1141, 1171.5, 1202, 1232.5, 1263.5, 1294, 1324.5,
          1355, 1385.5, 1416.5, 1446.5, 1476.5, 1507, 1537.5, 1568, 1598.5, 1629.5,
          1660, 1690.5, 1721, 1751.5, 1782.5, 1812, 1841.5, 1872, 1902.5, 1933,
          1963.5, 1994.5, 2025, 2055.5, 2086, 2116.5, 2147.5, 2177, 2206.5, 2237,
          2267.5, 2298, 2328.5, 2359.5, 2390, 2420.5, 2451, 2481.5, 2512.5, 2542,
          2571.5, 2602, 2632.5, 2663, 2693.5, 2724.5, 2755, 2785.5, 2816, 2846.5,
          2877.5, 2907.5, 2937.5, 2968, 2998.5, 3029, 3059.5, 3090.5, 3121, 3151.5,
          3182, 3212.5, 3243.5, 3273, 3302.5, 3333, 3363.5, 3394, 3424.5, 3455.5,
          3486, 3516.5, 3547, 3577.5, 3608.5, 3638, 3667.5, 3698, 3728.5, 3759,
          3789.5, 3820.5, 3851, 3881.5, 3912, 3942.5, 3973.5, 4003, 4032.5, 4063,
          4093.5, 4124, 4154.5, 4185.5, 4216;
5. Future Work

- **Infrastructure:**
  - Redeployment of web service to AWS
  - Access to additional OPeNDAP datasets
- **Development:**
  - Prototype interface to ESGF, NCPP
  - Expand test coverage
  - Reconnect web interface to new URL format
  - On-the-fly dataset concatenation
- **Documentation!**

ben.koziol@noaa.gov
Supported Operations on Climate Datasets

• Subset by time and/or level and arbitrary vector-based user geometries
  o Intersects and clip geometric operations
    ▪ Prototype spatial index for large, complex geometries
    ▪ Geometry wrapper/unwrapper for 0 to 360 coordinate systems
  o Value aggregation to selection geometry bounds (area-weighted)
  o Works with multiple geometries in a single request tagging climate data outputs with its enclosing geometry identifier

• Calculations on raw or geometrically aggregated data points grouped in temporal categories
  o Uni- and multivariate calculations

• Dump data to multiple formats
  o shapefile, csv, numpy, nc
Dependencies

- GDAL (Geospatial Data Abstraction Library)
  - shapefile I/O, spatial reference systems
  - requires: PROJ4, GEOS
- Shapely
  - geometric operations (intersects, touches, intersection, union)
  - requires: GEOS
- NumPy
  - array manipulation, computational library
- netcdf4-python
  - requires: HDF, NetCDF4, NumPy

http://www.earthsystemcog.org/projects/openclimategis/dependencies
3. System Diagram

User Request

- Returned Data (Python, URI, HTTP Response)
- Python or RESTful API

Converter → OcgCollection → Calculation Engine

OcgInterpreter → Subset Operation → OcgDataset

Global Interface

Temporal Interface → Level Interface → Spatial Interface

Variable Discovery or User Overloading

NetCDF of OPeNDAP Dataset

- Point or Polygon Vectorization
- Spatial Reference Management
- Bounding Box Comparisons and Full Geometric Operations
What OCGIS is not...

- High performance software per se
  - Simple parallelization and minimal data manipulations make it relatively fast within the constraints of Python and software dependencies
  - Numpy-based data storage and calculation
  - Speed and memory dependent on size, structure, and origin of data request
  - Many opportunities for eventual speed-ups
  - Current bottlenecks: remote data transfer, I/O, calculations
- Data analysis and visualization software
  - Designed to get data into "user-friendly" formats ready for analysis or visualization in other software packages
3. Brief introduction to request handling

- Data request organized in an operations objects that may be parameterized with a Python dictionary
- Request is validated
- Request definition object passed to an "interpreter"
  - Possible to implement additional interpreters (GDP for example)
- Interpreter executes operations
- Data output returned
  - Path in the case of file formats
  - Dictionary of NumPy arrays otherwise
- For the RESTful interface, web service (Django) simply reduces a URL to a request dictionary before calling the OCGIS library (in most cases)
class SampleSize(OcgFunction):
    description = 'Statistical sample size.'
    Group = groups.BasicStatistics
    name = 'n'
    dtype = int

@staticmethod
def _calculate_(values):
    ret = np.sum(~values.mask, axis=0)
    return(ret)

@staticmethod
def _aggregate_(values, weights):
    return(np.sum(values))

class HeatIndex(OcgCvArgFunction):
    description = 'Heat Index following: http://en.wikipedia.org/wiki/Heat_index.'
    Group = groups.MultivariateStatistics
    dtype = float
    nargs = 2
    keys = ['tas', 'rhs']
    name = 'heat_index'

@staticmethod
def _calculate_(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

    hi = np.mean(hi, axis=0)

    return(hi)
NCPP

OpenClimateGIS

Data Selection

Spatial

Area-of-Interest (AOI):

Spatial Filters:

Aggregate Geometries

Temporal

Grouping interval:

Available Statistics:

Basic Statistics:

Threshholds:

Output Format:

GeoJSON Text File

Data Request URL:

http://openclimategis.org/api

Map data ©2012 Google - Terms of Use
Report a map error