pycovjson Documentation

Release 0.3.7

University of Reading eScience Centre

October 10, 2016

Contents

Contents:

CHAPTER 1

Introduction to pycovjson

1.1 Introduction

pycovjson is a python utility library for creating CoverageJSON files from common scientific data formats (e.g NetCDF).

1.2 What does pycovjson offer?

The library provides a packaged Python API for the generation of CoverageJSON which is compliant with the CoverageJSON Format Specification.

CHAPTER 2

Quickstart

2.1 Purpose

The following document explains how to quickly get up and running with pycovjson. It explains how to execute the key commands and explains (at a high level) what those commands are doing e.g. what input and output we can expect. More detail on expressive use of the various API's including function level API documentation can be found in subsequent pages of this documentation guide.

2.2 CoverageJSON Generation

2.2.1 pycovjson-viewer

To quickly view the structure of a NetCDF file, pycovjson-viewer can be used. Usage is simple:

```
$ pycovjson-viewer (netcdf file)

Dimensions:  (depth: 20, lat: 171, lon: 360)
Coordinates:
   * depth    (depth) float32 5.0 15.0 25.0 35.0 48.0 67.0 96.0 139.0 204.0 ...
   * lat      (lat) float32 -81.0 -80.0 -79.0 -78.0 -77.0 -76.0 -75.0 -74.0 ...
   * lon      (lon) float32 0.0 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0 11.0 ...
Data variables:
 ICEC     (lat, lon) float64 nan nan nan nan nan nan nan nan nan nan nan nan ...
 ICETK    (lat, lon) float64 nan nan nan nan nan nan nan nan nan nan nan nan ...
 M        (lat, lon) float64 nan nan nan nan nan nan nan nan nan nan nan nan ...
 SALTY    (depth, lat, lon) float64 nan nan nan nan nan nan nan nan nan nan ...
 TMP      (depth, lat, lon) float64 nan nan nan nan nan nan nan nan nan nan ...
 U        (lat, lon) float64 nan nan nan nan nan nan nan nan nan nan nan nan ...
 V        (lat, lon) float64 nan nan nan nan nan nan nan nan nan nan nan nan ...
 Attributes:
 title: MET OFFICE FOAM GLOBAL 1 DEG DATA
 field_date: 2011-01-01 00:00:00
```

2.2.2 pycovjson-convert

This is very simple...

```
#

# pycovjson-convert -i foam.nc -o coverage.covjson -v [SALTY]
```

More on using pycovjson functions later...

2.3 Conclusion

That concludes the quick start. Hopefully this has been helpful in providing an overview of the main pycovjson features. If you have any issues with this document then please register them at the issue tracker. Please use labels to classify your issue.

CHAPTER 3

Installation:

If you already have netCDF4 and hdf5 installed, the installation process is simple. Open up a command line and type the following:

```
$ pip install pycovjson
```

If not, you will need to download conda for your operating system, details of how to do this can be found here. After you have installed conda, type the following in the command line:

```
$ conda install netcdf4
$ pip install pycovjson
```

Conda will install all of the required binaries for your OS.

CHAPTER 4

Command Line Interface

4.1 Convert Module

class pycovjson.cli.convert.**main**

Command line interface for pycovjson - Converts Scientific Data Formats into CovJSON and saves to disk.

Parameters

- **-i** – Input file path.
- **-o** – Output file name.
- **-t** – Use Tiling.
- **-v** – Which variable to populate coverage with.
- **-s** – [tile shape]: Tile shape.
- **-n** – Use interactive mode.

Members

4.1.1 Usage

:: pycovjson-convert(*path to netcdf file*, *name of output file*, *[list of vars as strings]* [-t] *tiled* [-s] *tile_shape as list*

4.2 Viewer Module

class pycovjson.cli.viewer.**main**

4.2.1 Usage

:: pycovjson-viewer (netcdf file)

CHAPTER 5

model.py

5.1 Coverage

class pycovjson.model.**Coverage**(*domain*, *ranges*, *params*, *reference*)

5.2 Domain

class pycovjson.model.**Domain**(*domain_type*, *x_values=[]*, *y_values=[]*, *z_values=[]*, *t_values=[]*)

5.3 Range

class pycovjson.model.**Range**(*range_type*, *data_type={}*, *axes=[]*, *shape=[]*, *values=[]*, *variable_name=''*, *tile_sets=[]*)

populate(*data_type={}*, *axes=[]*, *shape=[]*, *values=[]*, *variable_name=''*)
Function to populate Range object with values

5.4 TileSet

class pycovjson.model.**TileSet**(*tile_shape*, *url_template*)

get_tiles(*tile_shape: object*, *array*) → object
Function which yields a generator which can be leveraged to return tile arrays from an input array :param tile_shape:

Parameters **variable** –

Returns

CHAPTER 6

convert.py

6.1 Convert

Convert.py is a library API to group the read and write classes together into a single file, this allows it to be called in a much more simple way

class pycovjson.convert.**main**

CHAPTER 7

read_netcdf.py

class pycovjson.read_netcdf.**NetCDFReader**(*dataset_path*)

convert_time(*t_variable*)
Formats time objects to CovJSON compliant strings.

Parameters **t_variable** – Time Variable

Returns list of formatted datetime strings

extract_var_data(*var_names*)
Returns dictionary containing the values in each variable specified in the variable list.

Parameters **var_names** (*String*) –

:return variable_dict - Dictionary containing key-val pairs

get_description(*variable*)

Parameters **variable** – input variable

Returns long_name

get_dimensions(*variable*)
Return dimension of specified variable.

Parameters **variable** – Input variable

Returns Tuple - Array dimension of specified variable

get_metadata(*variable*)
Returns metadata for a specified variable.

Parameters **variable** – Name of specified

Returns dset[variable]

get_name(*variable*)

Parameters **variable** – input variable

Returns name - string

get_shape(*variable*)
Get shape of specifed variable, as list :param variable: String specifying variable name :return: shape_list - List containing shape of specified variable

get_std_name(*variable*)
Return standard name of variable.

Parameters **variable** – input variable

Returns standard_name

get_type(*variable*)

Parameters

- **dset** – NetCDF dataset object
- **variable** – Specified

Returns var_type with digits stripped

get_units(*variable*)
Return units of specified variable. :param variable: :return: units

get_values(*variable*)

Parameters **variable** –

Returns variable values as ndarray

get_var_group(*variable*)
Return group which specified variable belongs to.

Parameters **variable** –

Returns group as string

:type string

is_x(*var*)
Detect whether or not specified variable is an x coord :param var: :return: Boolean value

is_y(*var*)
Detect whether or not specified variable is a y coord :param var: :return: Boolean value

CHAPTER 8

pycovjson setup.py

CHAPTER 9

write.py

class pycovjson.write.**Writer**(*output_name: object*, *dataset_path: object*, *vars_to_write: object*, *tiled=False*, *tile_shape=[]*) → object

Writer class

Writer class constructor :param output_name: Name of output file :param dataset_path: Path to dataset :param vars_to_write: List of variables to write :param tiled: Boolean value (default False) :param tile_shape: List containing shape of tiles

save_covjson_tiled(*obj*, *path*)

Skip indentation of certain fields to make JSON more compact but still human readable :param obj: :param path:

write()

Writes Coverage object to disk

CHAPTER 10

Indices and tables

- genindex
- modindex
- search

p