```
0  # Define a set of tightly-ranged bounding coordinates to extract precip,
1
2  #tempmax, and tempmin daily data from the University of Idaho OPeNDAP
3
4  #server; a massive repository of NetCDF .nc files.
5
6  #Coord set is for 31 specific "weather stations" selected by SWAT
7
8  #for use with 31 subbasins in a model of the Clackamas River, OR.
9
10 #Daily weather data from HadGEM2 general climate model; 2086-2099
11
12
13 '''
14 helpful:
15 http://docs.opendap.org/index.php/QuickStart
16
17 repository selection sequence:
18 http://thredds.northwestknowledge.net:8080/thredds/catalog.html
19 http://thredds.northwestknowledge.net:8080/thredds/nw.csc.is.catalog.html
20 http://thredds.northwestknowledge.net:8080/thredds/nw.csc.climate.html
21 http://thredds.northwestknowledge.net:8080/thredds/catalog/
       NWCSC_INTEGRATED_SCENARIOS_ALL_CLIMATE/catalog.html
22 http://thredds.northwestknowledge.net:8080/thredds/catalog/
       NWCSC_INTEGRATED_SCENARIOS_ALL_CLIMATE/macav2livneh/catalog.html
23 http://thredds.northwestknowledge.net:8080/thredds/catalog/
       NWCSC_INTEGRATED_SCENARIOS_ALL_CLIMATE/macav2livneh/HadGEM2-ES365/catalog.
       html
24 http://thredds.northwestknowledge.net:8080/thredds/catalog/
       NWCSC_INTEGRATED_SCENARIOS_ALL_CLIMATE/macav2livneh/HadGEM2-ES365/catalog.
       html?dataset=NWCSC_IS_ALL_SCAN/macav2livneh/HadGEM2-ES365/
       macav2livneh_pr_HadGEM2-ES365_r1i1p1_rcp85_2086_2099_CONUS_daily.nc
25 '''
26
27 '''
28 code works when see this output:
29 opened url for 1x time
30 /home/bmarron/Desktop/25
31 lat lon data collected
32 coords found
33 coords df created
34 wrote ws coords
35 ...
36 '''
37
38
39 '''
40 Original code by Madeline Steele, Mike Psaris, GIS Programming 2012
41 Modified Bruce Marron, Portland State University, May 2016
42 '''
43
44
45
46 #%% SWAT-selcted weather stations from the 352 weather stations offered by
47
48 # by PRISM; sorted min to max
49
50 #
51
```

```
52 L=[30,27,25,51,52,77,100,96,122,95,120,164,147,143,166,175,198,171,218,209,
53 190,214,212,234,237,261,276,281,326,328,301]
54 L = sorted(L)
55
56
57 #%% the packages required for this script
58
59
60 import csv
61 import os
62 import numpy
63 import pydap    # Import the OPenDAP python library
64 from pydap.client import open_url
65 from pydap.client import open_dods
66     #import pandas as pd ??
67
68 import pydataframe as df
69 import datetime as dt
70
71 #%%
72
73 # Have each weather station tightly bounded so that only one
74
75 # "weather station" is pulled from the .nc database (tricky!)
76
77 # Save the tightly-bounded coords as .csv, with the following fields:
78
79 # ['gage_id', 'lon_max', 'lon_min', 'lat_max', 'lat_min']
80
81 # use ==> "Clackamas_ws_360_HadGEM2.csv"
82
83
84 output_path = "/home/bmarron/Desktop"
85 output_array = numpy.array([])
86
87 #%%
88
89 # HadGEM2 years in this .nc file are 2086 to 2099
90
91 # The .nc file is named with the prefix "pr",
92
93 #but the variables in the dataset use longer names
94
95 var_name_list = 'precipitation'
96
97 #Some of the variable names are the same.  I've changed them for more clarity
       when they are written to text files.
98
99 col_header_list = 'precipitation'
100
101
102 # Read in the list of XY min and max for each (buffered) polygon from QGIS
103
104 xy_bounds = "/home/bmarron/Desktop/Clackamas_ws_360_HadGEM2.csv"
105 reader = csv.reader(open(xy_bounds))
106 headers = reader.next() # This skips the headers row
107
108
109 #%%  Run script in one-shot from here
110
```

```
111 # Loop through each row (watershed bounding box)
112
113 # and get the coordinates from table
114
115
116 #Determine the index values of the weather station
117
118 #coordinates using the the "precipitation" variable.
119
120 #The indices are the same for all variables in this HadGEM2 series.
121
122
123 url = ("http://thredds.northwestknowledge.net:8080/thredds/dodsC/
        NWCSC_INTEGRATED_SCENARIOS_ALL_CLIMATE/macav2livneh/HadGEM2-ES365/
        macav2livneh_pr_HadGEM2-ES365_r1i1p1_rcp85_2086_2099_CONUS_daily.nc")
124 dataset = open_url(url)
125 print "opened url for 1x"
126
127 for row in reader:
128     gage_id = row[0]
129     lon_max = float(row[1]) # They need to be converted from string type
130     lon_min = float(row[2])
131     lat_max = float(row[3])
132     lat_min = float(row[4])
133     lat_list = []
134     lon_list = []
135     id_list = []
136
137     #Unique identifier for each grid point
138
139     dataset_num = 1
140
141     # Create an output folder for each gage
142
143     output_folder = (output_path + "/" + gage_id)
144     print output_folder
145     if not os.path.exists(output_folder):  # Check to see if output folder has
            been created. If not, make it.
146         os.makedirs(output_folder)
147
148     lat = dataset['lat'] # Accesses the latitude array
149     lon = dataset['lon'] # Accesses the longitude array
150     lat_ind = numpy.arange(0,(len(lat[:])))
151     lon_ind = numpy.arange(0,(len(lon[:])))
152     sub_lat_ind = lat_ind[(lat > lat_min) & (lat < lat_max)]
153     sub_lon_ind = lon_ind[(lon > lon_min) & (lon < lon_max)]
154     print "lat lon data collected "
155
156     #For each grid point, a time series of all the variables are written to a
            tab-delimited text file.
157
158     for lat_index in sub_lat_ind:
159         for lon_index in sub_lon_ind:
160             lat_list.append(float(lat[lat_index]))
161             lon_list.append(float(lon[lon_index]))
162             id_list.append(dataset_num)
163             dataset_num += 1
164     print "coords found"
165
166     coords = df.DataFrame({"ID": id_list, "LAT": lat_list, "LON": lon_list})
```

```
167     print "coords df created "
168 
169     file_name = output_path + "/" + str(gage_id) + "/" + "ws_Coords.txt"
170     with open(file_name, 'wb') as f:
171         write_data = f.write(str(coords))
172     f.closed
173     print "wrote ws coords"
```