Indexing Examples

Carver J. Bierson

July 10, 2017

This is the example used in class on July 20, 2017. This file shows two equivalent ways to manipulate data via loops and logical indexing. I highly reccomend if you want to get a better grasp of these techniques, make a small array (maybe 5 elements) and add a nan in the middle.

```
1
2  import numpy as np
3
4  # Load data
5  filename='co2_mm_mlo.txt'
6
7  Data=np.loadtxt(filename)
8
9  # Move data from 1 2D array to multiple 1D arrays for readability
10
11 Ddate=Data[:,2] # Decimal Date
12
13 CO2=Data[:,3] # Measured CO2 data (ppm)
14
15 CO2_I=Data[:,4] # Measured values with bad data interpolated
16
17 #%% Replace bad data in CO2 with nan (not a number) values
18
19 # Loop Method
20 for i in range(len(CO2)): # for each index in the length of CO2
21     if CO2[i] < 0: # check if that value is less than 0 (i.e. a bad
       value)
22         CO2[i]=np.nan  # If it is bad, replace with nan so we don't
       use it
23
24 # logical indexing method
25 CO2[CO2<0]=np.nan # replace all values where CO2<0 with np.nan
26
27 #%% Take the mean of the data between the dates Dmin and Dmax
28
29 # Define range
30 Dmin=2005
31 Dmax=2015
32
33 # If all the data were good (like in CO2_I) and we wanted the mean
       of all values we could do
34 simple_mean=np.mean(CO2_I)
35
36 # Loop method
37 # mean is sum of the data points / number of data points
38 sum1=0 # track the sum (note that the name sum is generally a
       function so I added a 1)
39 N=0 # Total number of points used
40 for i in range(len(CO2)): # loop with i as the index in CO2
```

```
  # check that the date of the current data point is within the
    range
  # ( less than the max and more than the min)
    if Ddate[i]>=Dmin and Ddate[i]<=Dmax:
        if np.isnan(CO2[i]): # if the data point is a nan, skip it
            continue
        # if it is not a nan, count it and add it to the sum
        N+=1
        sum1+=CO2[i]

meanval=sum1/N    # calculate mean

# Logical indexing method
meanval2= np.nanmean(               #use the version of mean that
    ignores nans
            CO2[np.logical_and(         # index the CO2 array where
    both these logical arrays are true
                Ddate<=Dmax,  # Check if the date is less than
    the max
                Ddate>=Dmin   # and greater than the min
                )])

#%% Linear fitting

 # polyfit really doesn't like nans. To get around this we can use
    the interpolated data like
fit=np.polyfit(Ddate,CO2_I,1)

# sometimes though you won't have interpolated data. In this case
    we need to remove the nan values from our arrays
# In other words, we want to use points that are not nans
fit2=np.polyfit(Ddate[ # index the dates array so that it is the
    same size as CO2 with nans removed
            np.logical_not( # all points where that are not
            np.isnan(CO2) # where the nans are
            )],
                CO2[np.logical_not(np.isnan(CO2))], # same exact
    indexing expression for CO2 so they end up the same size
                1)
```