**ATOC7500 – Application Lab #2**

**Regression, Autocorrelation, Red Noise Timeseries**

**in class Feb. 10/15, 2022**

**Notebook #1 – Autocorrelation and Effective Sample Size using Fort Collins, Colorado weather observations**

**ATOC5860\_applicationlab2\_AR1\_Nstar.ipynb**

**LEARNING GOALS:**

1) Calculate the autocorrelation at a range of lags using two methods available in python (np.correlate, dot products)

2) Estimate the effective sample size (N\*) using the lag-1 autocorrelation

3) Evaluate the influence of changing the sampling frequency and the specified weather variable on the memory/redness of the data as quantified by the autocorrelation and N\*.

**DATA and UNDERLYING SCIENCE:**

In this notebook, you will analyze the memory (red noise) in weather observations from Fort Colins, Colorado at Christman Field. The observations are from one year, but are sampled hourly. The default settings for the notebook analyze the air temperature in degrees F sampled once daily (every midnight). But other standard weather variables and sampling frequencies can also be easily analyzed. The file containing the data is called christman\_2016.csv and it is a comma-delimited text file.

**Non-exhaustive Questions to guide your analysis of Notebook #1:**

1) Start with the default settings in the code. In other words – Read in the data and find the air temperature every 24 hours (every midnight) over the entire year. Calculate the lag-1 autocorrelation using np.correlate and the direct method using dot products. Compare the python syntax for calculating the autocorrelation with the formulas in Barnes. Equation numbers are provided to refer you back to the Barnes Notes. What is the lag-1 autocorrelation?

*The lag-1 autocorrelation is 0.846.*

2) Calculate the autocorrelation at a range of lags using np.correlate and the direct method using dot products. Compare the python syntax for calculating the autocorrelation with the formulas in Barnes. Equation numbers are provided to refer you back to the Barnes Notes. How does the autocorrelation change as you vary the lag from -40 days to +40 days?  
  
*The python syntax for np.correlate and np.dot are identical except for the option in np.correlate to apply linear convolutions to the data. The autocorrelation is maximized at lag 0, since the data is perfectly correlated with itself, but diminishes with increasing lag magnitude. Since the autocorrelation diminishes with some linearity means that the daily temperature dataset is red.*

3) Calculate the effective sample size (N\*) and compare it to your original sample size (N). Equation numbers are provided to refer you back to the Barnes Notes. How much memory is there in temperature sampled every midnight?  
  
*The effective sample size is 31. The memory of the dataset can be described by the lag-1 autocorrelation, which for this dataset is 0.85. The dataset is therefore reasonably red, meaning there is significant memory in the temperature sampled every midnight.*

4) Now you are ready to tinker … i.e., make minor adjustments to the code with the parameters set in the code to see how your results change. *Suggestion: Make a copy of the notebook for your tinkering so that you can refer back to your original answers and the unmodified original code.* For example: Repeat steps 1-3) above with a different variable (e.g., relative humidity (RH), wind speed (wind\_mph)). Repeat steps 1-3) above with a different temporal sampling frequency (e.g., every 12 hours, every 6 hours, every 4 days). How do you answers change?

*Our investigation of wind\_mph at a daily interval:*

1. *The lag-1 autocorrelation is -0.045*
2. *The autocorrelation tends toward white noise which can be determined by the fact that it falls to a minimum values after only lag 1.*
3. *The effective sample size is 335 out of a total sample size of 366.*

*At an hourly interval:*

1. *The lag-1 autocorrelation is 0.861*
2. *The autocorrelation diminishes out to about 15 hours with a red noise signal, after which the noise appears to be white.*
3. *The effective sample size is 657 out of a total sample size of 8784.*

**Notebook #2 – Red noise time series generation, Regression, and Statistical Significance Testing While Regressing**

**ATOC5860\_applicationlab2\_AR1\_regression\_AO.ipynb**

**LEARNING GOALS:**

1) Calculate and analyze the autocorrelation at a range of lags using output from an EOF analysis (the Arctic Oscillation Index).

2) Generate a red noise time series with equivalent memory as an observed time series (i.e., given lag-1 autocorrelation).

3) Correlate two time series and calculate the statistical significance.

4) Evaluate the statistical significance obtained in the context of the number of chances provided for success. What happens when you go “fishing” for correlations and give yourself lots of opportunity for success? Can you critically evaluate the chances that your regression is statistically different than 0 just by chance?

**DATA and UNDERLYING SCIENCE:**

In this notebook, you will analyze the monthly Arctic Oscillation (AO) timeseries from January 1950 to present. The AO timeseries comes from an Empirical Orthogonal Function (EOF) analysis. We will implement EOFs in the next application lab so in this lab we are actually using multiple analysis methods introduced in this class, some that you have learned and some that you are still yet to learn ☺.

How do you find the AO value each month? To identify the atmospheric circulation patterns that explain the most variance, NOAA regularly applies EOF analysis to the monthly mean 1000-hPa height anomalies poleward of 20° latitude for the Northern Hemisphere. The AO spatial pattern (Figure 1 below) emerges as the first EOF (explaining the most variance, 19%). The AO timeseries we will analyze is a measure of the amplitude of the pattern in Figure 1 in a given month. In other words – the AO timeseries is the first principal component (a timeseries) associated with the first EOF (a spatial structure). More information on the EOF analysis here:

http://www.cpc.ncep.noaa.gov/products/precip/CWlink/daily\_ao\_index/history/method.shtml



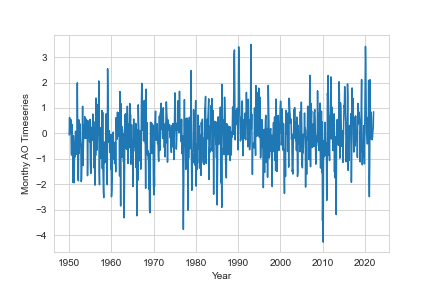
Figure 1. The loading pattern of the Arctic Oscillation (AO), i.e., the structure explaining the most variance of monthly mean 1000mb height during 1979-2000 period. In other words – this is the first EOF.

The data are available and regularly updated here:

<http://www.cpc.ncep.noaa.gov/products/precip/CWlink/pna/norm.nao.monthly.b5001.current.ascii>

You can work with the data directly on the web (assuming you have an internet connection). I have also downloaded the data and made them available – The name of the data file is “monthly.ao.index.b50.current.ascii”.

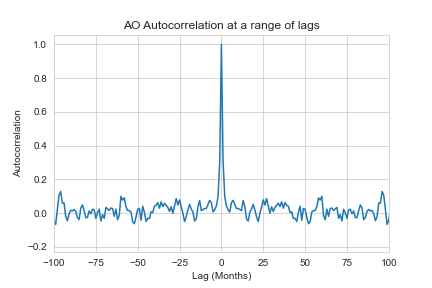
**Questions to guide your analysis of Notebook #2:**

1) Start with the default settings in the code. First read in the Arctic Oscillation (AO) data. Look at your data!! Plot it as a timeseries. Save the timeseries plot as a postscript file and put it in this document.  
  


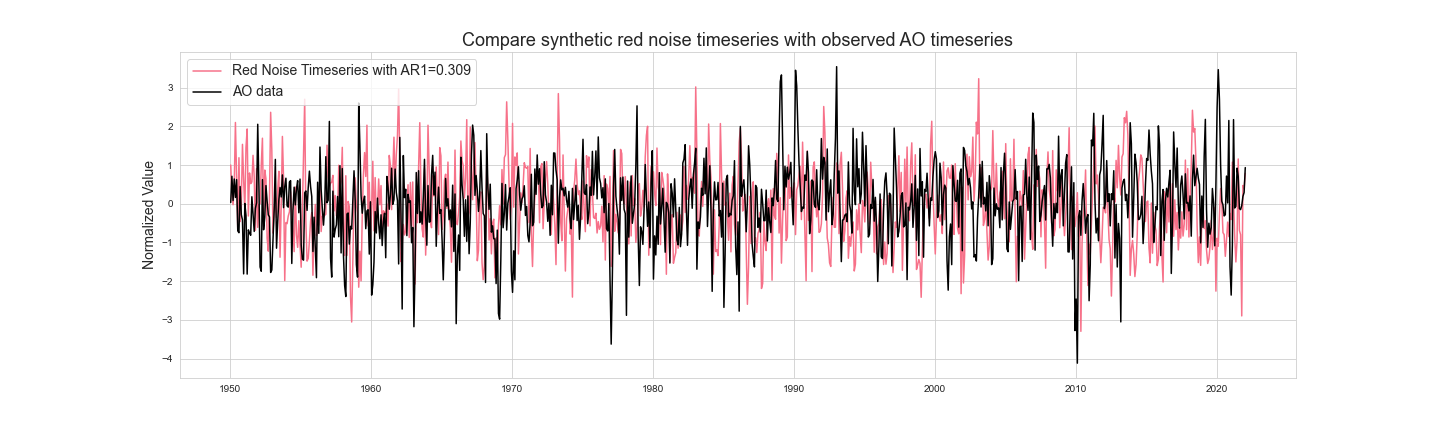
2) Calculate the lag-one autocorrelation (AR1) of the AO data and record it here. Use two methods (np.correlate, dot products). Check that they give you the same result. Interpret the value. How much memory (red noise) is there in the AO from month to month?  
  
*np.correlate autocorrelation: [0.30855]*

*direct calculation autocorrelation: 0.30855*

*The autocorrelation values do not look particularly red but we can further quantify the redness by looking at the e-folding time. Our e-folding time is 0.85 months (eqn. 85), so we know that the AO loses about 40% of its memory in about 26 days.*

3) Calculate and plot the autocorrelation of the AO data at all lags. Describe your results. How red are the data at lags other than lag=1? Is there any interesting behavior of the autocorrelation as a function of lag? What would you expect for red noise timeseries with an AR1=value reported in 2)?  


*The autocorrelation drops significantly after lag=1 such that the data is no longer red by lag=3. There does not appear to be any interesting behavior as a function of lag, and I would expect the results shown in the plot above since it was derived from the dataset.*

4) Generate a synthetic red noise time series with the same lag-1 autocorrelation as the AO data. Your synthetic dataset should have different time evolution but the same memory as the AO. Plot the AO timeseries and the synthetic red noise time series. Put the plot below.  
  


5) Do you expect to find any correlation between the two datasets, i.e., the synthetic red noise and the actual AO data? What is the correlation between the synthetic red noise and the actual AO data? Calculate a regression coefficient and other associated regression statistics.

*I do not expect to find any correlation between the two datasets because the synthetic red noise dataset only shares a red noise coefficient while the values of the two datasets are completely independent.   
  
The percentage variance explained is 0.1013%.*

6) Next -- Have some fun and go “fishing for correlations”. What happens if you try correlating subsets of the two datasets many times? When you try 200 times -- what is the maximum correlation/variance explained you can obtain between the synthetic red noise and the actual data? *Note: you are effectively searching for a high correlation with no a priori reason to do so.... THIS IS NOT good practice for science but we are doing it here because it is instructive to see what happens :)*

*Largest r\_value 0.59*

*Largest variance explained 34.4 %  
So, even though the datasets are completely independent, by fishing for correlations we found a maximum explained variance of 34.4%! That is huge!*

7) Calculate the correlation statistics for the highest correlation obtained in question 6). Two methods are provided - they should give you the same answers. Place a confidence interval on your correlation. Because you have found a correlation that is not equal to 0, use the Fisher-Z Transformation. Did your "fishing" for a statistically significant correlation work? Is your highest correlation statistically significant (i.e., can you reject the null hypothesis that the correlation is zero)? Write out the steps for hypothesis testing and use the values you calculate to formally assess.

*scipy.stats.linregress slope: 0.724*

*scipy.stats.linregress intercept: -0.614*

*scipy.stats.linregress r\_value: 0.587*

*direct method slope\_fast: 0.724*

*direct method intercept\_fast: -0.614*

*direct method rvalue\_fast: 0.587  
  
Rhomin (minimum 95% confidence interval for r\_value): 0.16*

*Rhomax (maximum 95% confidence interval for r\_value): 0.83*  
  
*i) We test at a 95% confidence level, so α = 0.05*

*ii) The null hypothesis is H0: the correlation is equal zero: r = 0*

*iii) Since our correlation is not equal to zero, we use a t-statistic on a Fisher-Z transformation.*

*iv) Our critical region is 0.16 to 0.83*

*v) Since our critical region does not overlap with zero, we can reject the null hypothesis that the data and rednoise datasets is zero.*

8) You went searching for correlations, you searched long and hard (200 times!) You should have been concerned that the largest correlation you found would be a false positive. Do you think you found a false positive? Explain what you found and potentially why you think it is important statistically but not physically. What lessons did you learn by “fishing for correlations”?

*I do believe I found a false positive since 0.59 is within the 95% confidence interval. This is important statistically because it provides evidence that about 1 in 20 random samples will include a significant correlation, even though the datasets are physically independent. I learned that it is important to consider the physical interconnectedness of variables of interest before looking for correlations.*

FOR FUN: Check out - <https://www.tylervigen.com/spurious-correlations>