**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 python script np.correlate(t1,t2,mode=’valid’) & np.dot(t1,t2) seem to do the same thing, as both are divided by the same values [(n-lag)(sigma\*\*2)] and yield the exact same result. The time series data is normalized.

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?

Chart, line chart

Description automatically generatedLag-1: 0.846

Lag-2: 0.779

Lag-3: 0.735

Lag-4: 0.741

Lag-5: 0.743

Lag-10: 0.725

Lag-20: 0.616

Lag-40: 0.403

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?

N\* = 31, compared to the N\_sample = 366! The sample is fairly auto-correlated, with a significant amount of memory (as we expected with the variable temperature).

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?

I’m going to re-run the 24-hour code with Surface Pressure. Min: 821.4 hPa; Max: 855.09 hPa

Lag-1 autocorrelation is 0.535 (much less)

Chart, line chart

Description automatically generated

N\*: 111 (N=366 again)

Very interesting, it seems that the autocorrelation is cyclical (~10-14 days?). Day to Day pressures (lag = 1) is fairly low, but there is *some* memory here.

* Change temporal sampling to 6 hours with temperature

Lag-1 = 0.762

As expected, Lag-4 mimics the Lag-1 previously—and the “worse” is the lag-multiples of 2 (midnight vs. noon).

For Lag-1, N\*= 198 (N=1464)

**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.

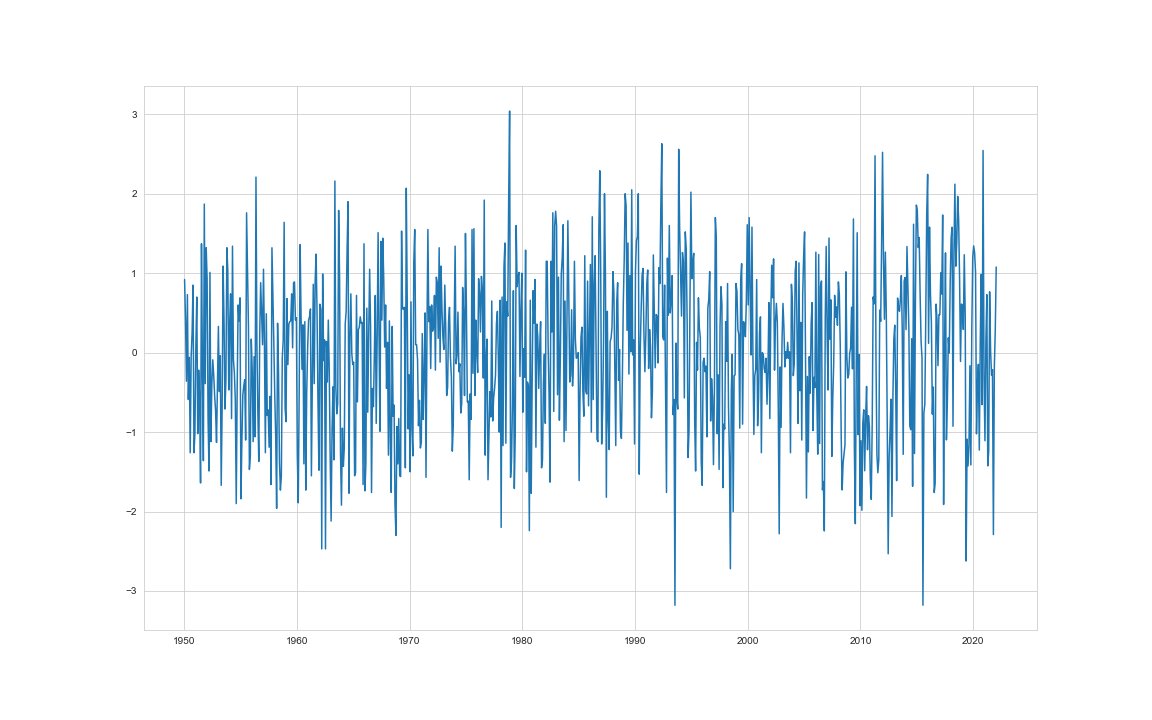
The data are available and regularly updated here:

<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.

(can’t load the .eps file into the doc, so saved as a png). 

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?

AR1 using np.correlate: 0.19906

AR1 using np.dot: 0.19906

Again, same result. This lag-1 autocorrelation is relatively low, which means there is *some* memory, but not a lot. The month-to-month AO is not very red.

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)?

Chart, line chart

Description automatically generated

The AO at lags higher than 1 are even less red than lag-1; more similar to white noise. The autocorrelation values go below 0 for lags 3-7.5 (except for 5), and then return to above 0. Unless the timeseries was seasonal or cyclical in nature, I would expect a dataset with a relatively white AR1 value (i.e., low AR1) to yield a relatively white AR values for lags > 1 as well.

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.

Chart, bar chart

Description automatically generated

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.

Given the low AR value used to create the synthetic red dataset (0.199), I would not expect the correlation to be high.

R-value = -0.006

R2 value = 0.0036%

Standard Error = 0.034

The lingress statistics back up the expectation that there is not much correlation between the AO data and the synthetic red data.

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 :)*

Maximum correlation (200 times, Length = 20 timesteps):

* R\_value = 0.54
* R2 value = 28.94% (pretty high!)

Maximum correlation (200 times, Length = 10 timesteps):

* R\_value = 0.83
* R2 value = 68.95% (very high!)

Maximum correlation (200 times, Length = 5 timesteps):

* R\_value = -0.99
* R2 value = 97.65% (extremely high!)

**Re-run to get a new Random Red Dataset:**

Maximum correlation (200 times, Length = 20 timesteps):

* R\_value = -0.68
* R2 value = 46.68% (even higher than before!)

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.

Highest Correlation: N = 196 🡪 timesteps [ 588, 608 ]

Lingress R\_Value: -0.638

1. State the Significant Level: alpha = 5
2. State the Null Hypothesis: the correlation between the highest 20-timestep long AO and synthetic red-noise dataset is 0 (H1 🡪 not equal to 0)
3. Statistic: Fisher-Z
4. Critical Region: t\_crit = 2.11 🡪 Rho\_min = -0.87m Rho\_max = -0.31
5. Conclusion: Null hypothesis is rejected!

Since the confidence range for the true correlation given by [ Rhomin, Rhomax ] does not overlap with 0, we can reject the null hypothesis that the correlation between fp\_data and fp\_rednoise is zero. If we reject the null hypothesis that the correlation is zero, we have found a false positive!

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”?

Yes, I believe we found a false positive. Given the length of the datasets, the fat that we were only taking 20 timesteps, and given that the red dataset had AR1 value identical to that of the AO dataset, I don’t think it’s necessarily that surprising that we found a false positive. However, given that we logically know that there shouldn’t be any correlation between the two datasets, the fact that we get an R2 of 0.47 is pretty surprising. The lessons learned here are that if you find a correlation in your own research, it’s always important to test the data in other ways to ensure you aren’t getting a false positive. It can be very easy to find correlations even if they don’t exist physically!

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

My favorite:

Chart, line chart

Description automatically generated