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?

From Barnes Eq. 67, we find the turbulent quantities of each timeseries by subtracting the mean from each timestamp value. These departures from the mean are multiplied with each other, summed, and divided by the length minus the lag. The equations calculate the length of the timeseries by subtracting $(t_N - t_1)$. To receive $\rho(\mathrm{T})$, the autocovariance is normalized by $\gamma(0)$, or the variance $\overline{x'^2}$.

In the python notebooks, we also start by finding departures from the mean by subtracting the mean from the value at each timestamp. We then take a dot product and divide by the length of the timeseries minus the lag as before. Then, to normalize to the variance, we calculate the standard deviation, square it, and divide.

The lag-1 autocorrelation is 0.846. This is the amount of memory, or red noise, in the timeseries. If the autocorrelation is large, then successive timestamps have memory and affect each other.

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?
At +-40 days, the autocorrelation is smaller. This means that the temperature today has some effect on the temperature in 40 days (and 40 days ago). As the numbers of days reduces, the autocorrelation increases. This means that the temperature today has a strong influence on the temperature tomorrow, and vice versa. This also means there is no periodicity in the data.

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. This is a large decrease from 366 which is indicative of strong memory in the timeseries of temperature. Here the memory, or autocorrelation, is 0.846.

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?
Using pressure with a lag of 1 day, we see that is the same as temperature. The autocorrelation is 0.535 and the effective sample size is again 111.
Using pressure with a lag of hours, the autocorrelation increases to 0.92 and the effective sample size has reduced to 61.
Using pressure with a lag of 12 hours, the autocorrelation is 0.8 with an effective sample size of 80.

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).

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

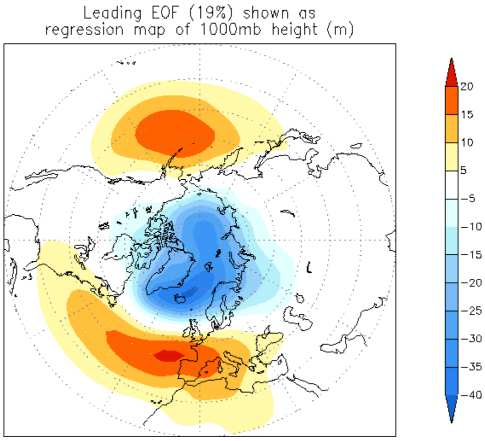

Leading EOF (19%) shown as
regression map of 1000mb height (m)
20
15
10
5
-5
-10
-15
-20
-25
-30
-35
-40

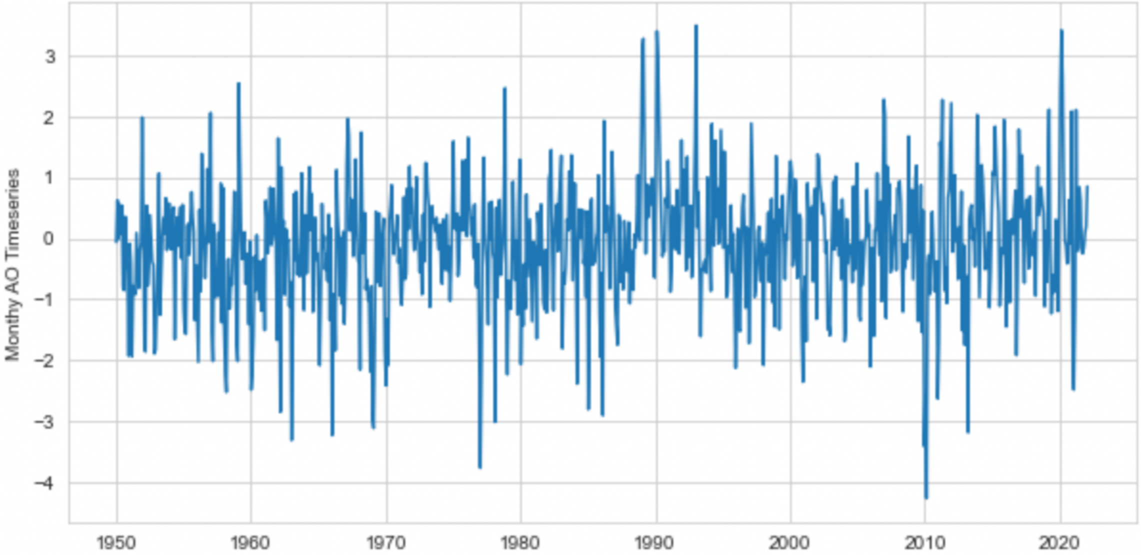
Figure 1. The loading pattern of the Arctic Oscillation (AO), i.e., the structure explaining the most variance of monthly mean 1000mb height during 1979-2000 period. In other words – this is the first EOF.

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?

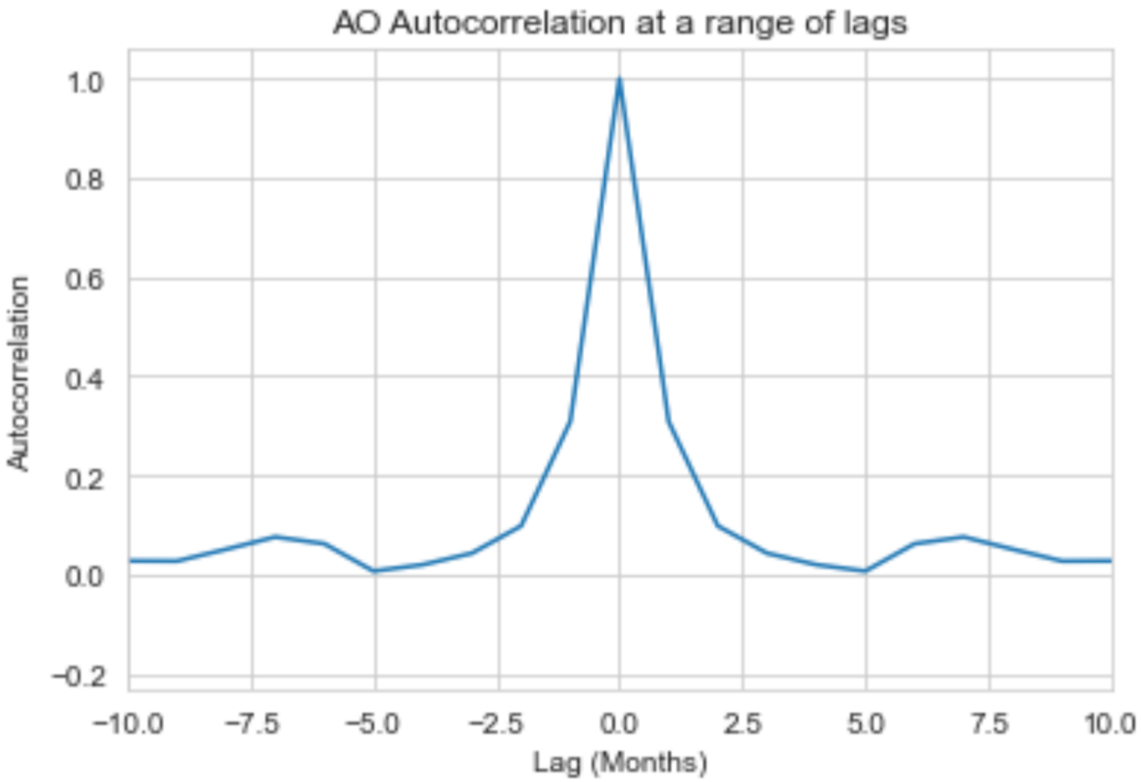
Both methods yield the same result.

With a lag of 1 month, the autocorrelation is 0.31. This is a moderate amount of memory.

With a lag of 6 months, the autocorrelation is 0.06. This does not have much memory at all.

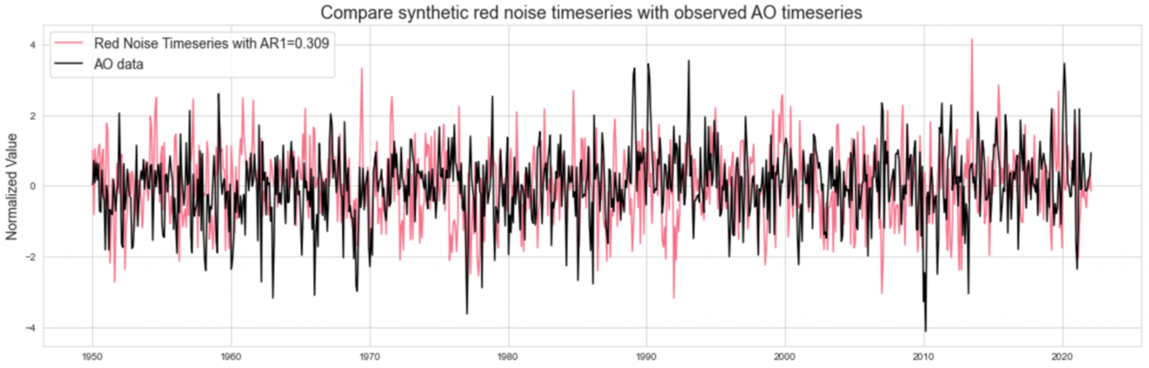
With a lag of 12 months, the autocorrelation is 0.01. This essentially has no memory.

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 data is not very red at lags greater than 1. The interesting behavior is that there is an increase in memory at 6 months, indicating a biannual cycle in AO. Although the AR value reported in 2 is the same for the synthetic data, I don't think this necessarily means that there will be a lag of exactly 6 months in the synthetic data since it is randomly generated.

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 expect to find some correlation between the two datasets although I imagine this is mostly up to chance. Having the same autocorrelation means the datasets correlate more or less well with themselves, not with each other. The regression coefficient between the datasets is only 0.1512% with a slope of 0.02 and an intercept of 0.0006.

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 :)*
When fishing for correlations 200 times, we retrieve a maximum correlation of 41.48%. However, the largest Pearson's r was -0.67, so these datasets increase and decrease with each other.

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.
The correlation statistics for the highest correlation are an r value of -0.681, a slope of -0.726, and an intercept of -0.207. These values remain the same between the different methods. Fishing for a correlation did work, since |-0.68|>0.5, so there is more correlation than not.
This correlation is statistically significant and we can reject the null hypothesis that the correlation is zero.

1) We want a 95% confidence level, so $\alpha = 0.05$.
2) H0: The null hypothesis as that the two timeseries do not correlate with each other.
3) We use the t-statistic with a two-sided test because the two timeseries have lengths of 20, we assume the underlying distributions are normal, and we do not have a priori information regarding the correlation sign.
4) From Barnes Eq. 54, we calculate a confidence interval between -0.87 and -0.31. This interval does not contain 0, and thus the null hypothesis that the correlation is 0 is rejected.

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”?

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

Yes, I think the strong correlation we found is a false positive. I think this because the two timeseries as a whole don’t correlate well with each other and because the probability of correctly rejecting the null hypothesis is only 0.0035%. This is important statistically since it means that there exists a period of time where the datasets do correlate well and researchers investigating that period need to be careful with the results. Physically though, this means that the two timeseries are not correlated as a whole, in which case we can conclude that there is no physical relationship.