**ATOC5860 – Application Lab #5**

**Filtering Timeseries**

**Spring 2022**

**Notebook #1 – ATOC5860\_applicationlab5**

**ATOC5860\_applicationlab5\_check\_python\_convolution.ipynb**

**LEARNING GOAL**

1) Understand what is happening “under the hood” in different python functions that are used to smooth data in the time domain.

Use this notebook to understand the different python functions that can be used to smooth data in the time domain. Compare with a “by hand” convolution function. Look at your data by printing its shape and also values. Understand what the python function is doing, especially how it is treating edge effects.

Edge effects are important when filtering, because different filters handle the edge of the data differently. The edge points are unable to get the full range of data compared to points in the center. By default, np.convolve uses ‘full’ which add additional points on the end. ‘same’ makes no changes to the data and returns a smaller array.

**Notebook #2 – Filtering Synthetic Data**

**ATOC5860\_applicationlab5\_synthetic\_data\_with\_filters.ipynb**

**LEARNING GOALS:**

1) Apply both non-recursive and recursive filters to a synthetic dataset

2) Contrast the influence of applying different non-recursive filters including the 1-2-1 filter, 1-1-1 filter, the 1-1-1-1-1 filter, and the Lanczos filter.

3) Investigate the influence of changing the window and cutoff on Lanczos smoothing.

**DATA and UNDERLYING SCIENCE:**

In this notebook, you analyze a timeseries with known properties. You will apply filters of different types and assess their influence on the resulting filtered dataset.

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

1. Create a red noise timeseries with oscillations. Plot your synthetic data – Look at your data!! Look at the underlying equation. What type of frequencies might you expect to be able to remove with filtering?

Chart

Description automatically generated with medium confidence

I would expect frequencies that are high. Such as frequencies produced by the random addition. We could also setup the filter in a way to reduce the written-in frequencies found in the data.

1. Apply non-recursive filters in the time domain (i.e., apply a moving average to the original data) to reduce power at high frequencies. Compare the filtered time series with the original data (top plot). Look at the moving window weights (bottom plot). You are using the function “filtfilt” from scipy.signal, which applies both a forward and a backward running average. Try different filter types – What is the influence of the length of the smoothing window or weighted average that is applied (e.g., 1-1-1 filter vs. 1-1-1-1-1 filter)? What is the influence of the amplitude of the smoothing window or the weighted average that is applied (e.g., 1-1-1 filter vs. 1-2-1 filter)?

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

**Chart, histogram

Description automatically generated**

**This filter is crazy and doesn’t do anything useful, but it does look cool!**

1. Apply a Lanczos filter to remove high frequency noise (i.e., to smooth the data). What is the influence of increasing/decreasing the window length on the smoothing and the response function (Moving Window Weights) in the Lanczos filter? What is the influence of increasing/decreasing the cutoff on the smoothing and the response function?

Chart, line chart, histogram

Description automatically generated

Window 50

Chart, line chart

Description automatically generated

Window 20

Reducing the window length definitely reduces the smoothing.

1. Apply a Butterworth filter, a recursive filter. Compare the response function (Moving Window Weights) with the non-recursive filters analyzed above.

Chart, line chart

Description automatically generated

Lanczos filter, lots of oscillations

Chart, line chart

Description automatically generated

Butterworth filter, has a better cutoff

**Notebook #3 – Filtering ENSO data**

**ATOC5860\_applicationlab5\_mrbutterworth\_example.ipynb**

**LEARNING GOALS:**

1) Assess the influence of filtering on data in both the time domain (i.e., in time series plots) and the spectral domain (i.e., in plots of the power spectra).

2) Apply a Butterworth filter to remove power of specific frequencies from a time series.

3) Contrast the influence of differing window weights on the filtered dataset both in the time domain and the spectral domain.

4) Calculate the response function using the Convolution Theorem.

5) Assess why the python function filtfilt is filtering twice.

**DATA and UNDERLYING SCIENCE:**

In this notebook, you analyze monthly sea surface temperature anomalies in the Nino3.4 region from the Community Earth System (CESM) Large Ensemble project fully coupled 1850 control run (http://www.cesm.ucar.edu/projects/community-projects/LENS/). A reminder that an pre-industrial control run has perpetual 1850 conditions (i.e., they have constant 1850 climate). The file containing the data is in netcdf4 format: CESM1\_LENS\_Coupled\_Control.cvdp\_data.401-2200.nc

*Does this all look and sound really familiar? It should!! This dataset is the same one you analyzed in Homework #4.*

**Questions to guide your analysis of Notebook #3:**

1. Look at your data! Read in your data and Make a plot of your data. Make sure your data are anomalies (i.e., the mean has been removed). Look at your data. Do you see variance at frequencies that you might be able to remove?

A picture containing bar chart

Description automatically generated

The data has been standardized. It’s tough to identify specific frequencies, but I would expect a cycle roughly on the period of el nino/la nina. Maybe a 5 year period.

1. Calculate the power spectrum of your original data. Calculate the power spectra of the Nino3.4 SST index (variable called “nino34”) in the fully coupled model 1850 control run. Apply the analysis to the first 700 years of the run. Use Welch’s method (WOSA!) with a Hanning window and a window length of 50 years. Make a plot of normalized spectral power vs. frequency. Where is their power that you might be able to remove with filtering?

Chart, line chart

Description automatically generated

The peak is almost exactly at a 5 year period! I bet I could remove the high frequency spike around 0.008. I could also remove the higher frequencies to really isolate the 5-year peak.

1. Apply a Butterworth Filter. Use a Butterworth filter to remove all spectral power at frequencies greater than 0.04 per month (i.e., less than 2 year). Use an order 1 Butterworth filter (N=1, 1 weight). Replot the original data and the filtered data. Calculate the power spectra of your filtered data. Assess the influence of your filtering in both in time domain (i.e., by comparing the original data time series and filtered time series data) and the frequency domain (i.e., by comparing the power spectrum of the original data and the power spectrum of the filtered data). Look at the response function of the filter in spectral domain using the convolution theorem. Well that was pretty boring… we still have most of the power retained….

Graphical user interface

Description automatically generated

This worked pretty well. It did lose a lot of lower frequencies though. While this was helpful with removing noise, I am not sure if it was great at retaining the original data. I am not used to filtering in practice so I may just have too high of expecations.

1. Let’s apply another Butterworth Filter and this time really get rid of ENSO power!. Let’s really have some fun with the Butterworth filter and have a big impact on our data... Let’s remove ENSO variability from our original timeseries. Apply the Butterworth filter but this time change the frequency that you are cutting off to 0.01 per month (i.e., remove all power with timescales less than 8 years). Use an order 1 filter (N=1). Replot the original data and the filtered data. Calculate the power spectra of your filtered data. Assess the influence of your filtering in both in time domain (i.e., by comparing the original data time series and filtered time series data) and the frequency domain (i.e., by comparing the power spectrum of the original data and the power spectrum of the filtered data). Look at the response function of the filter in spectral domain using the convolution theorem.

Graphical user interface

Description automatically generated

Wow, we really just cut out a lot of frequencies. I think this would only useful if we were interested in really long term processes that were longer than El Nino.

1. Let’s apply yet another Butterworth Filter – and this time one with more weights. Repeat step 4) but this time change the order of the filter. In other words, increase the number of weights being used in the filter by increasing the parameter N in the jupyter notebook. What is the impact of increasing N on the filtered dataset, the power spectra, and the moving window weights? You should see that as you increase N – a sharper cutoff in frequency space occurs in the power spectra. Why?

Graphical user interface

Description automatically generated with medium confidence

With a filter cutoff at 0.015 and an N of 10, we can effectively isolate the low frequency signal. I would say that this is a very effective use of the butterworth filter.

1. Assess what is “under the hood” of the python function. How are the edge effects treated? Why is the function filtfilt filtering twice?

Filtfilt is applied twice for a both forward and backward spectra. That way we know that we are seeing the frequencies in both directions. This removes potential edge effects, especially when looking at low frequencies. The default method for edges in filtfilt is odd padding, so extra data points are extrapolated to the end for calculating filtering. Since it is odd, the added data points will have odd symmetry with the endpoint data.