Phys 641 Assignment 2

Emma Ellingwood 260811207

Question 1 (Code Q1.py)

$$m = \left(A^T N^{-1} A\right)^{-1} A^T N^{-1} d \tag{1}$$

$$m = \left((QR)^T N^{-1} QR\right)^{-1} (QR)^T N^{-1} d \tag{2}$$

$$m = \left(R^T Q^T N^{-1} QR\right)^{-1} R^T Q^T N^{-1} d \tag{3}$$

$$m = R^{-1} Q^{-1} N Q R^{T-1} R^T Q^T N^{-1} d \tag{4}$$

If we say all the matrices are orthogonal such that $XX^T = \mathbb{1}$ and $XX^{-1} = \mathbb{1}$ where $\mathbb{1}$ is just the identity matrix

$$m = R^{-1} Q^{-1} N N^{-1} d \tag{5}$$

$$m = R^{-1} Q^T d \tag{6}$$

Classical fits usually break down at higher orders which is where the QR factorization thrives. So to show that sometimes the QR fit is better than the classical I set the order to 25 and plotted the classical equation and the QR factorization fit. The initial equation is just a third order polynomial I created. It is easy to see visually that the classical fit in blue is very far off the black data points while the QR fit almost perfectly follows the data.

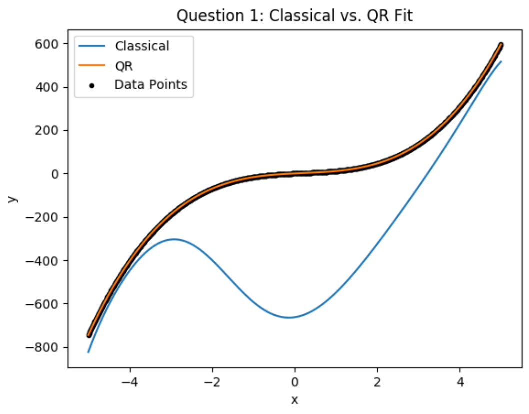


Figure 1: Example of classical and QR factorization fits for a third order polynomial with a fit order of 25

Question 2

Part (a) (Code Q2.py) Below (Fig.**??**) is a the exponential data and the fit for different order Chebyshev polynomials. Besides the order=2 plot, it is hard to distinguish the higher orders from the actual exponential data besides zooming in very far to a small section as seen in Fig **??**. Fig **??** shows that the fit gets better at higher orders and so the fit is very stable after even just a few orders.

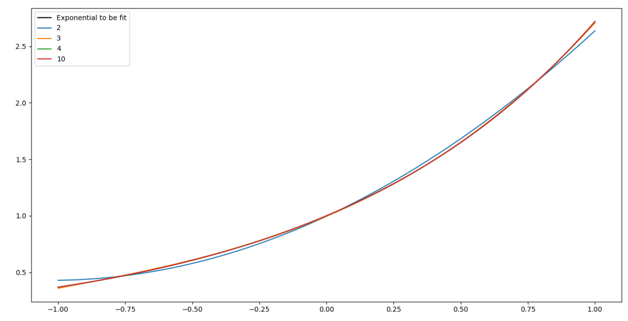


Figure 2: Plot of y=exp(x) and the Chebyshev polynomial fit for different polynomial orders

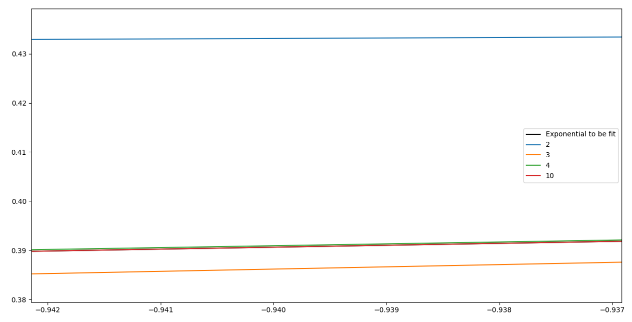


Figure 3: Zoomed in version of Fig. **??** to show the fit improvement with increasing polynomial order

Part (b) (Code Q2_ b.py)

The following are the results of fitting a 6th order Chebyshev polynomial and then a 50th order Chebyshev polynomial.

For order=6

RMS Error = 1.972×10^{-6}

Max Error = 8.141×10^{-6}

For order=50, truncated to order 6

RMS Error = 2.258×10^{-6}

Max Error = 8.141×10^{-6}

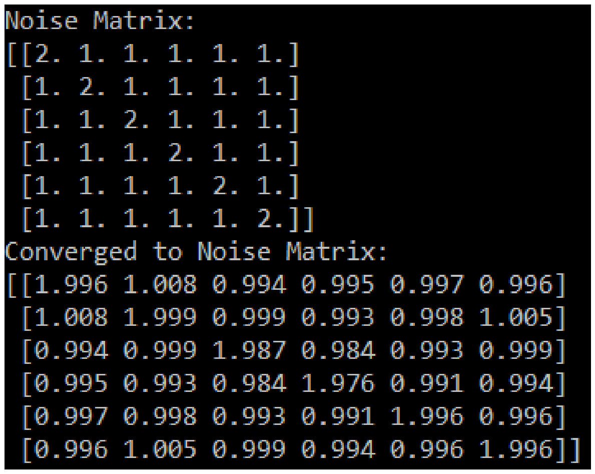
Based on these values, the RMS error has increased 14.5% between the order 6 and order 50 cases and the max error went down by a factor of 2.39. This agrees with the values stated in the problem set.

The max error without truncating is 9.770 times 10^{-15} , so if I had not truncated the terms my max error would have been much smaller than the max error with truncating.

Question 3 (Code Q3.py)

The code averages over 100000 realization of $\langle dd^T \rangle$, where d is the dot product of the Cholesky decomposition of the noise matrix and a vector of random Gaussian distributed values.

Figure **??** is a screen shot of the code output showing the initial noise matrix, and the averaged output of the code given to three decimal places.



```
Noise Matrix:
[[2. 1. 1. 1. 1. 1.]
 [1. 2. 1. 1. 1. 1.]
 [1. 1. 2. 1. 1. 1.]
 [1. 1. 1. 2. 1. 1.]
 [1. 1. 1. 1. 2. 1.]
 [1. 1. 1. 1. 1. 2.]]
Converged to Noise Matrix:
[[1.996 1.008 0.994 0.995 0.997 0.996]
 [1.008 1.999 0.999 0.993 0.998 1.005]
 [0.994 0.999 1.987 0.984 0.993 0.999]
 [0.995 0.993 0.984 1.976 0.991 0.994]
 [0.997 0.998 0.993 0.991 1.996 0.996]
 [0.996 1.005 0.999 0.994 0.996 1.996]]
```

Figure 4: Output of Q3.py showing initial noise matrix and calculated matrix

Question 4 (Code Q4.py)

Part (b)

For the code I set the fit equation to have the same standard deviation and mean value as the data that I added to the noise, but it still finds the amplitude which I use to find the error on the amplitude.

In Figure **??**, the blue is the green is the initial template, the blue is the template plus the noise and then the orange is the fit for a=0.5 , σ=0.5 for the noise. It is possible to visually see that there is a difference in the amplitude between the template and the fit. My results match the sanity check given in the problem set so that my error for a=0.5 , σ=0.5 is 0.276. Figure **??** shows the output for all pairs of a and σ .

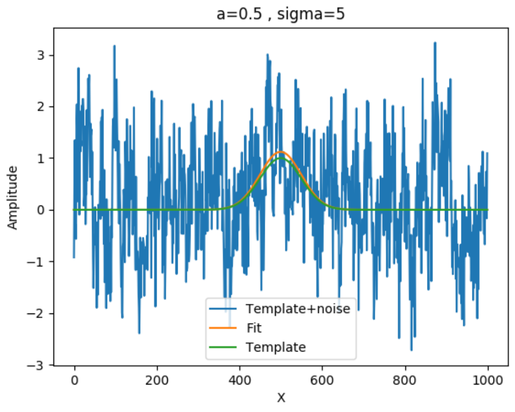
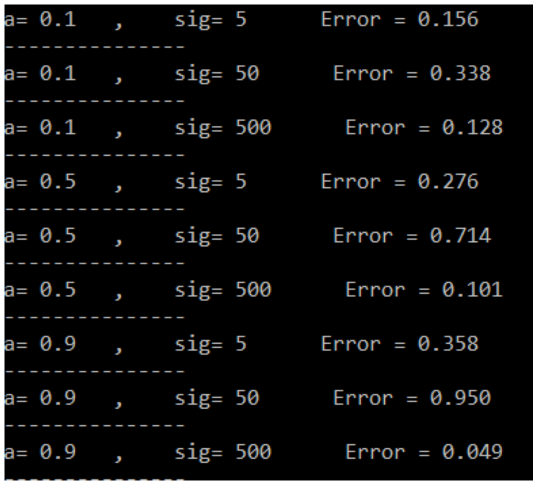


Figure 5: Example of the fit of the template with noise compared to just the template



```
a= 0.1    ,    sig= 5       Error = 0.156
---------------
a= 0.1    ,    sig= 50       Error = 0.338
---------------
a= 0.1    ,    sig= 500       Error = 0.128
---------------
a= 0.5    ,    sig= 5       Error = 0.276
---------------
a= 0.5    ,    sig= 50       Error = 0.714
---------------
a= 0.5    ,    sig= 500       Error = 0.101
---------------
a= 0.9    ,    sig= 5       Error = 0.358
---------------
a= 0.9    ,    sig= 50       Error = 0.950
---------------
a= 0.9    ,    sig= 500       Error = 0.049
---------------
```

Figure 6: Screenshot of Q4.py output

Part (b) Figure **??** is my attempt at plotting the data with noise both with and without data. With data is the template is in grey and without the data is with the red so then it is possible to see where the template causes the most changes based on how much individual grey and red is visible compared to the overlapping maroon colour.

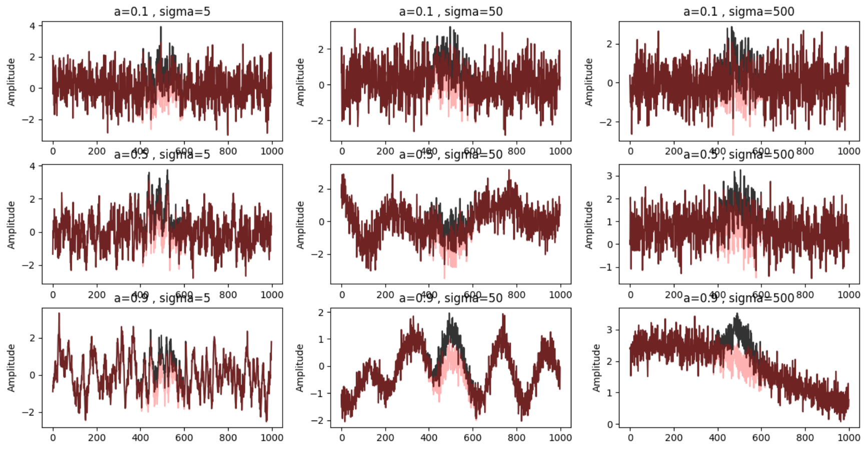


Figure 7: Q4.py Output

For the explanation about which one error is worst/best and why as requested, it is all put as a comment in the code. This is just a copy

Large 'a' means that the noise is mostly correlated and the amplitude of the noise will be larger, this results in larger error bars as 'a' increases. When sigma is very small, especially compared to the signal width, the variations are very pronounced and they vary on shorter scales so with a larger width. Similar to when sigma is high compared to the signal width because the deviations are on much longer scales so it is still possible to see the signal. When sigma is on the same scale as the signal width because it is very hard to distinguish any deviation as being part of a signal or just from the noise.

The a=0.9, sig=50 combination has the worst error bar. The sigma is on the scale of the signal width and it is very correlated which makes it hard to distinguish the signal. The a=0.9,sig=500 has the best error bar, high correlation and sigma makes a smoothish noise, makes it easier to see the signal

Noise with sigma=50 is the type that I should worry about more, specifically those with higher correlation because the variations in the noise is on the same scale as the width of the signal added in. So the variations are able to hide the signal in a way, to where it is less obvious. The high correlation also contributes to this to make the wiggles in the noise more pronounced.