Regularized Inverse Laplace Transform

Objective – In today's assignment, we had multiple objectives.

- Generate three signals as given: x1, x2 and x3
    1. Obtain theta parameter without using regularizing parameter
    2. Obtain theta parameter using regularizing parameter
- Add noise in the three signals: x1, x2 and x3
    1. Obtain theta parameter using regularizing parameter
    2. Obtain theta parameter without using regularizing parameter

Input parameters:

Initially time was defined which was varied between 0 to 50 us in the time steps of 0.01 giving us 5001 samples of the exponential signal x1, x2 and x3 each making the dimensions of the signals **1x5001**.

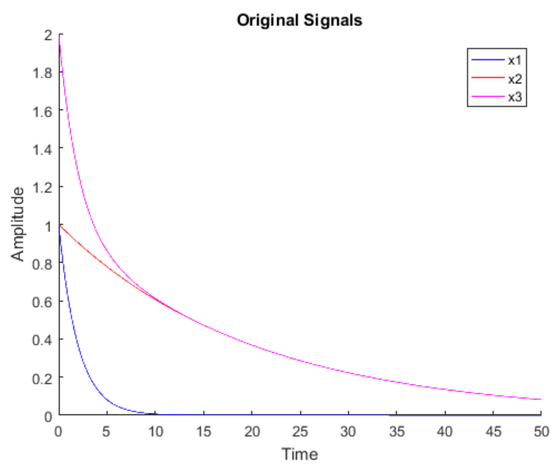
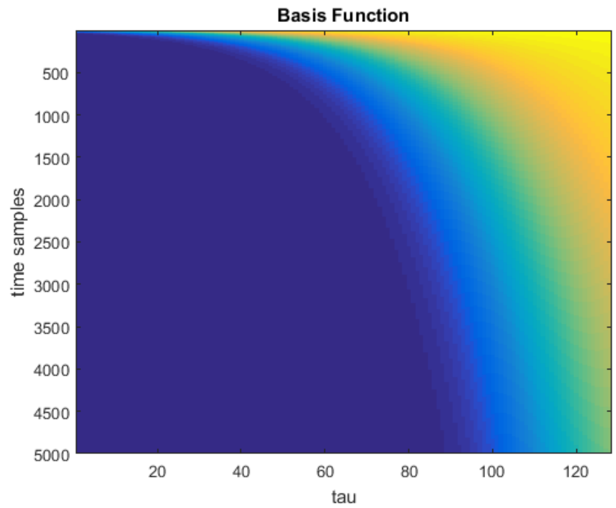


Fig 1: Original signals

The next parameter to generate was the **basis function**. We used the logspace function in Matlab for the same. Tau was to vary from 0.1 to 100us. According to logspace function we had to take the log to the 10 of 0.1 and 100 which is -1 and 2 as input to logspace along with length of the array which is equal to 128. Thus, the basis function dimension was **128x5001** and also the pseudo-inverse had the same dimensions.

Result from Basis function: We can see both the basis function and the pseudo inverse.

- In the basis function, we have a series of exponential signals with t ranging from 0 to 50 for 128 different tau at different logspace.
- The pseudo-inverse is the matrix of interest as it shows a definite pattern with certain yellow spots in between designating low values.



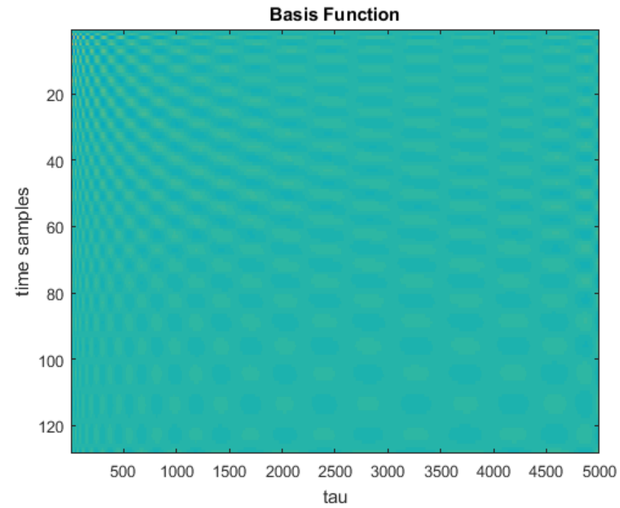
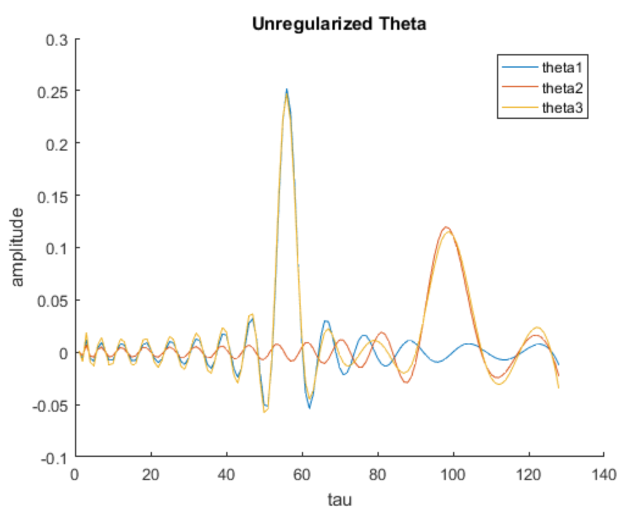


Fig 2: Basis function

Fig 3: pseudo -inverse of basis function

Theta calculation:

Unregularized – For unregularized method, we calculated pseudo-inverse of the matrix by using the pinv() function in MATLAB which gave us a well-conditioned pseudo-inverse. We tried with the traditional method but it was ill-conditioned with the factor of -21 which was giving us inaccurate results.



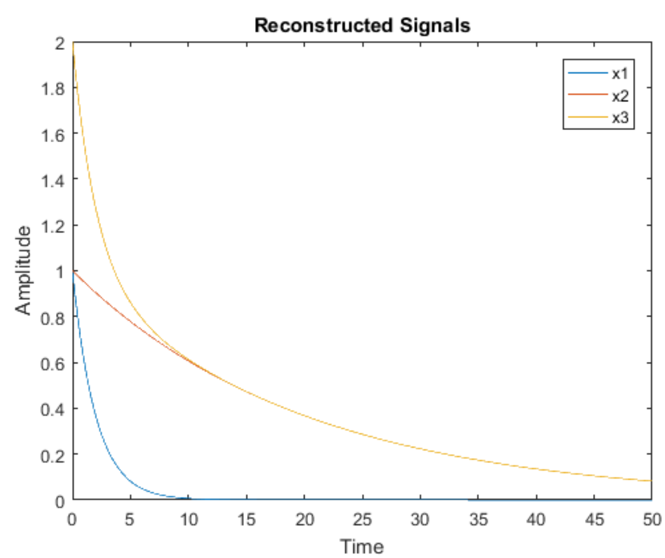


Fig 4: Unregularized theta

Fig 5: Reconstructed signal

After multiplying the pseudo-inverse of basis function with each signal we received the theta for each signal. Unregularized theta are given in Fig 4.

We reconstructed the signal by multiplying the basis function with theta. Reconstructed signals are given in Fig 5.

Result: We can see in Fig 4 that we have obtained theta but there are large number fluctuations which signifies the noise. Our motive is to generate theta values with no fluctuation in turn eliminating the noise component in the signal.

Regularized – For regularized method, we used the theta obtained from the unregularized method with the motive to minimize the error between x and x_cap = basis*theta, x is the desired value and x_cap is the actual value using prss. We used the fminunc function in MATLAB for the same. As we didn't have the final gradient, fminunc used to use the quasi-newton method to minimize the error.

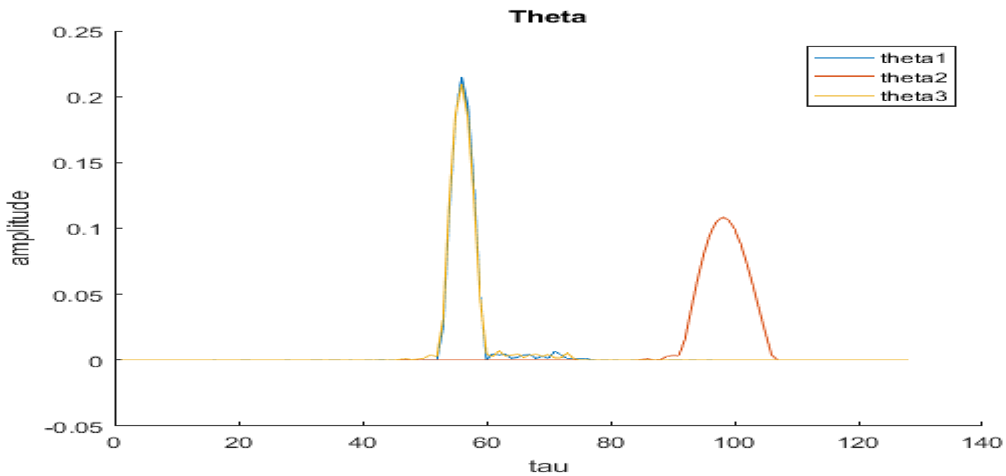


Fig 6: Regularized theta

Initially, when we tried implementing the code, the function used to run for the **length of theta (128)** and used to stop thus, giving us the local minima which used to be the same output as we get with lambda= 0. Probably, it was also because the starting point of the function was always the **unregularized theta**.

But to contemplate this mistake, we increased the number of iterations using the options variable incorporated in the function to increase the number of iteration. Once we done that, with right choice of lambda obtained by trial and error we obtained the regularized theta for each signal.

Result: We can see that we get just one spike with minimal noise fluctuations for each thetas. Hence, we can say that we have regularized successfully.

Signals with noise:

Our second objective was to repeat the regularization and unregularization techniques with noise added to the signals.

Input parameters: We took the same signals as generated in the previous procedure and added white gaussian noise with **standard deviation 0.05**.

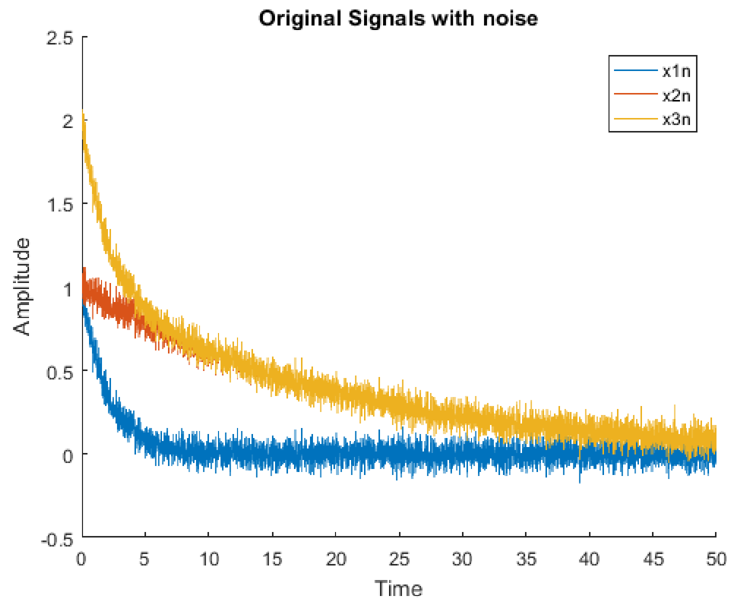


Fig 7: Noisy signals

The basis function remains the same so the pseudo-inverse of the basis function remains the same.

Unregularized theta:

We used to get random theta every time as the noise keeps on changing. Thus, unregularized theta always used to have an oscillating nature.

Regularized theta:

As the noise keeps on changing, it's difficult to have same lambda for every value. It used to give different output for same lambdas. But we could generalized that the lambda here had to be in the range of 10^5 to 10^9. Having these values of lambda gave us better regularized output. If we increased the power the oscillations used to increase.

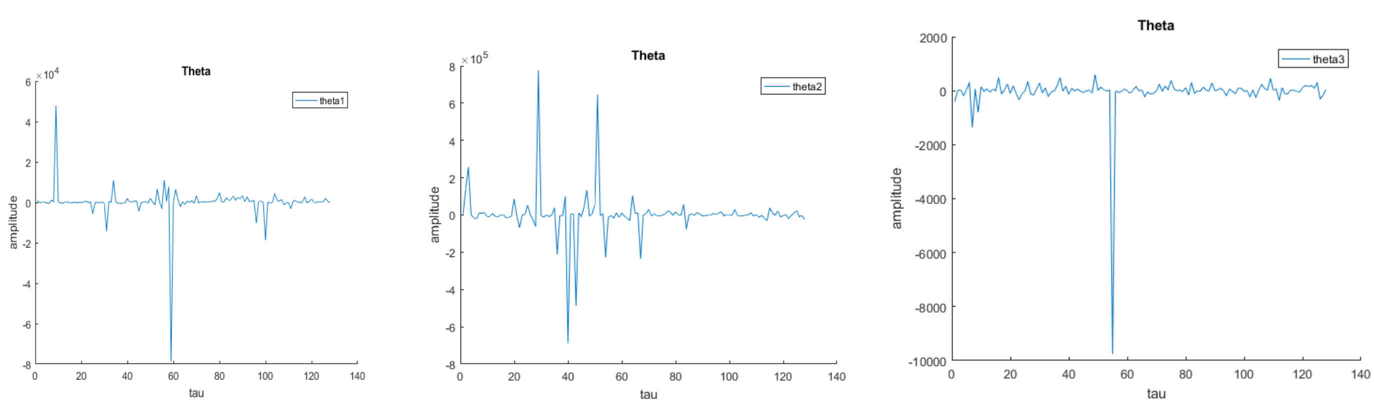


Fig 8: Regularized thetas of noisy signals

Result: Thus, we obtained the regularized theta for noisy signals. We can still see few oscillations, it was quite expected because of the nature of the noisy signal. We had to vary lambda for every time we run the code as the noise element was random and keeping same lambda value didn't give the same output.

Conclusion: Thus, we achieved all the required objectives successfully.