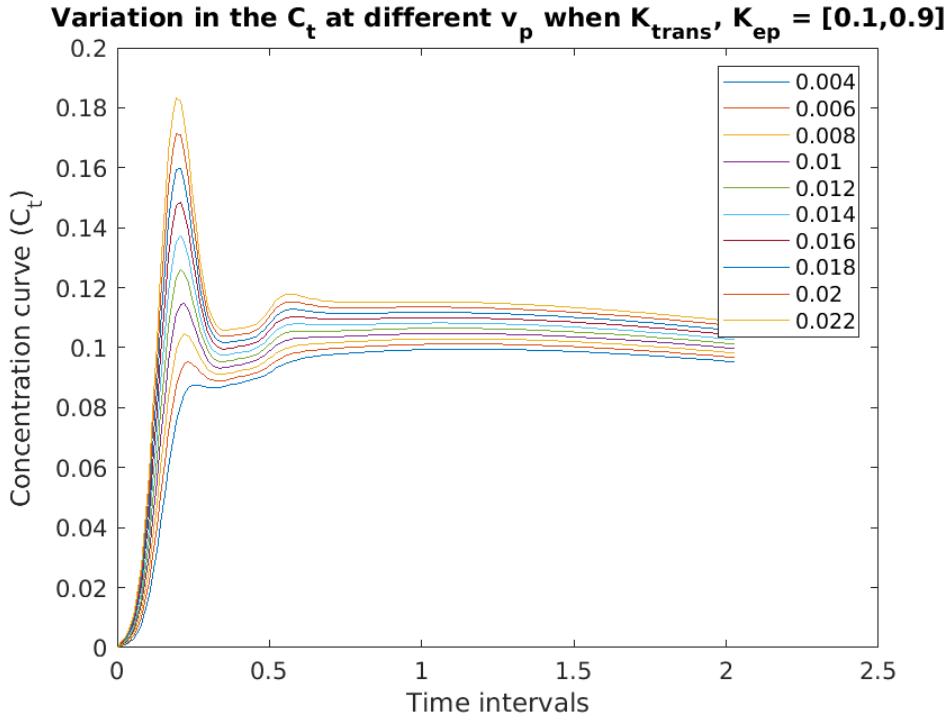
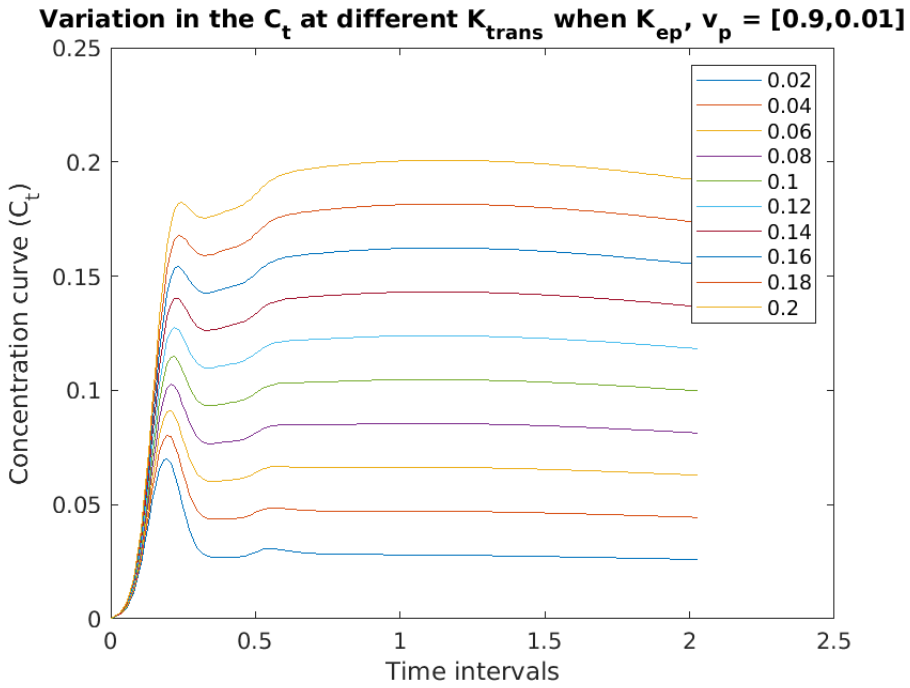
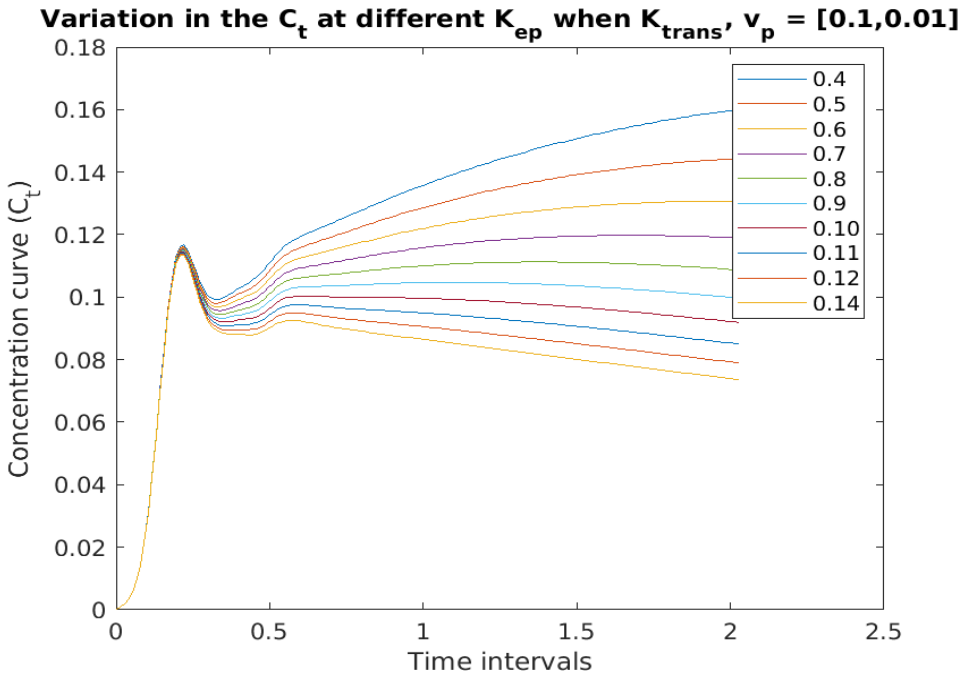
Report 3

1. Graphs at different K_trans, K_ep, V_p values:

Below are the graphs to show the variation of C_t (Concentration curve) as the parameter values are changed.

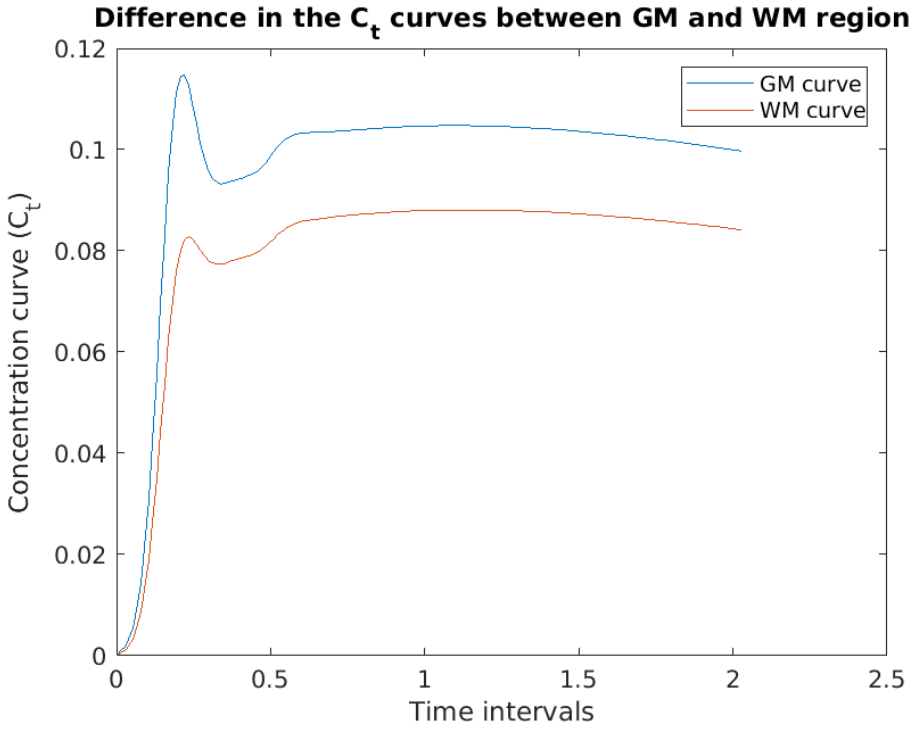






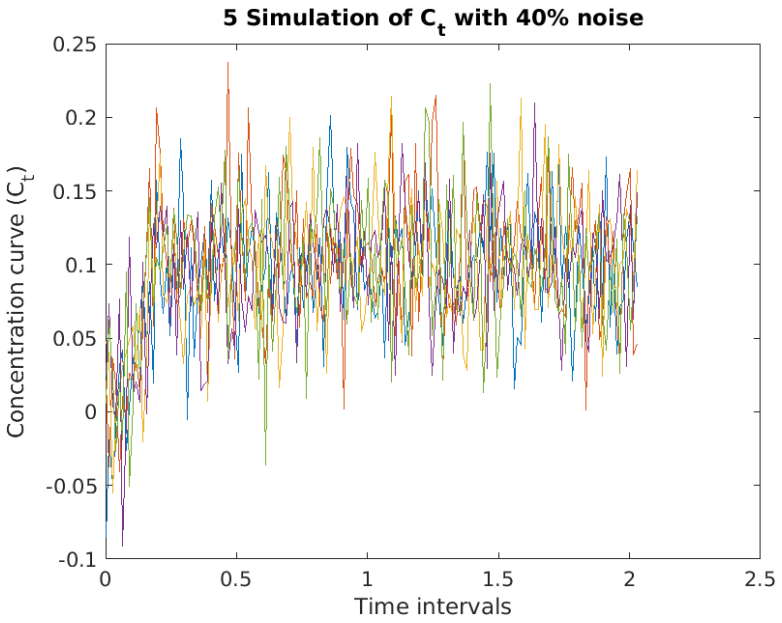
2. The Difference between Ct curves in the White matter and the Grey Matter region.

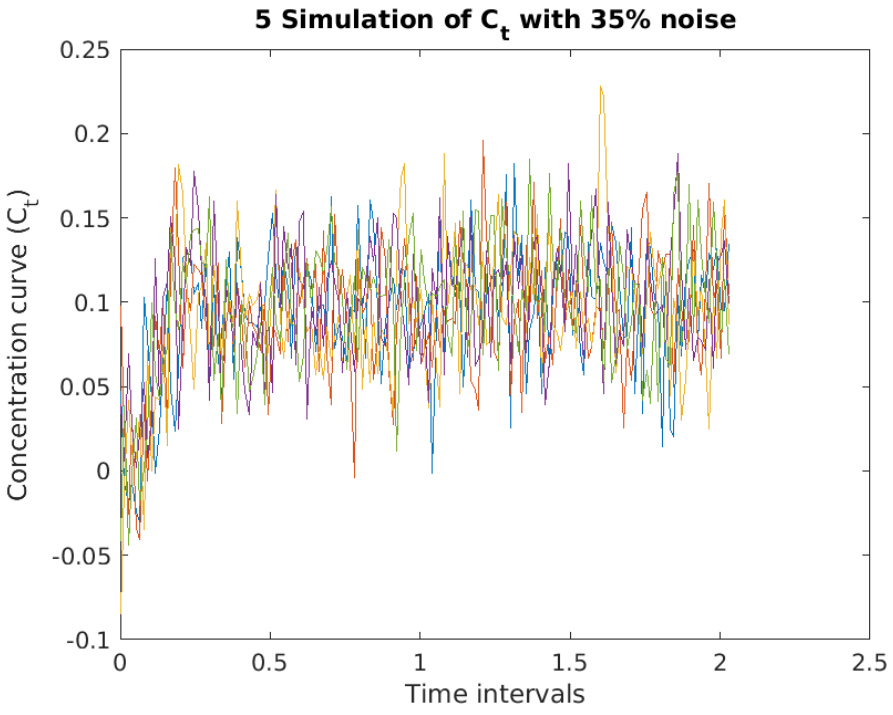
The Amplitude of the signal in the Grey Matter (GM) is lower than the White Matter (WM) curve. But in this report, we will only work with the GM curve as the same method can be extrapolated to the WM curve.

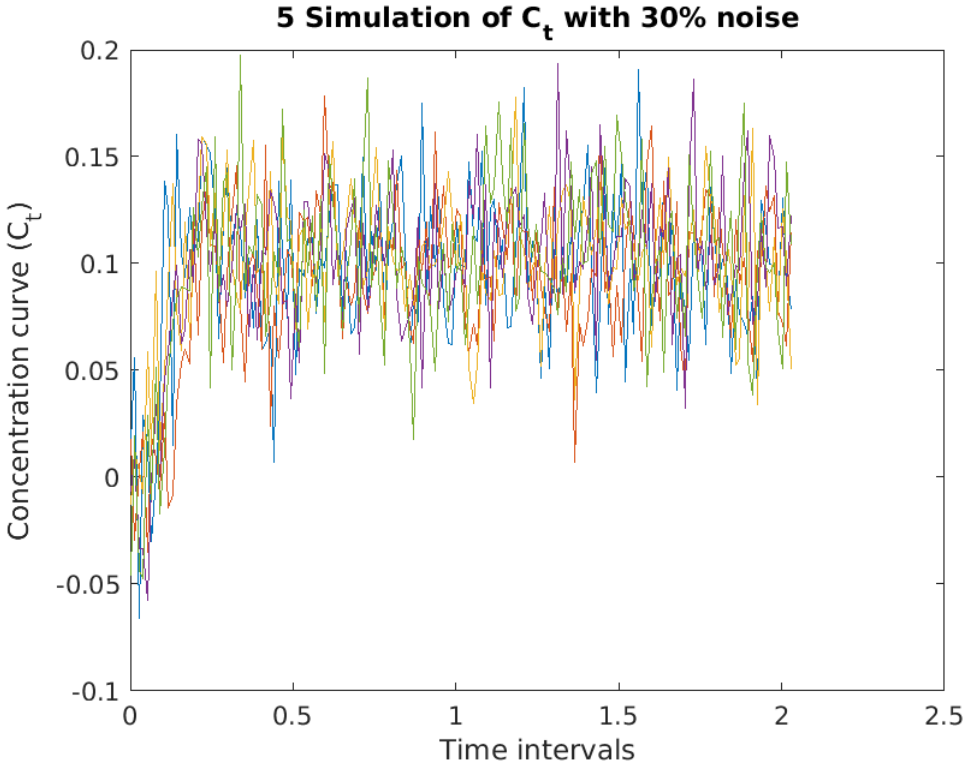


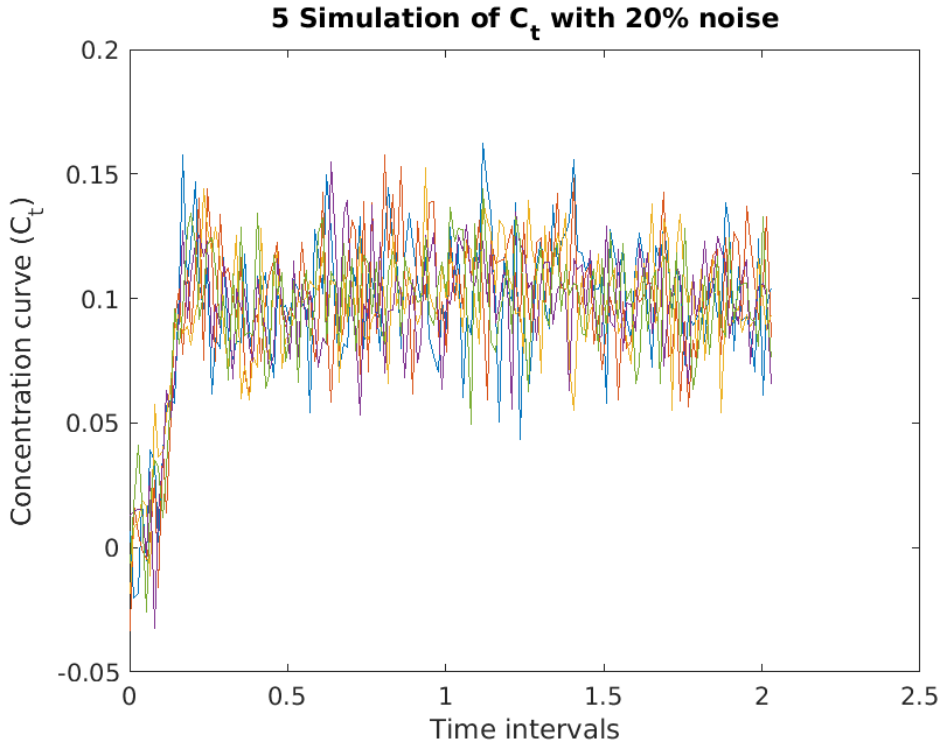
3. The CURVE Below are made with parameters value: K_trans, K_ep, V_p = [0.1, 0.9, 0.01] and further noise was added into them at different levels. 5 simulations of each noise level are given in the figure below.

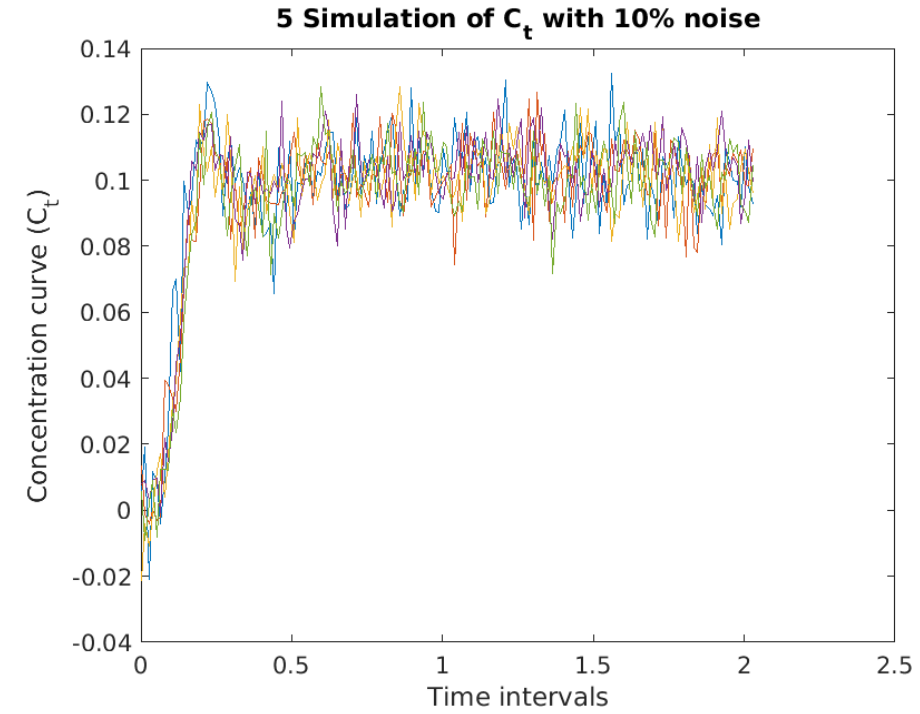
Noise is added using the function (MATLAB): maxConc*noiseLevel*randn(length(T),1); %% random noise with normal distribution (u=0, sd=1)
Here maxConc = 0.1, and Noise levels = 0.1(10%), 0.2 (20%)… similarly. Also 100 curves for each noise level were taken.

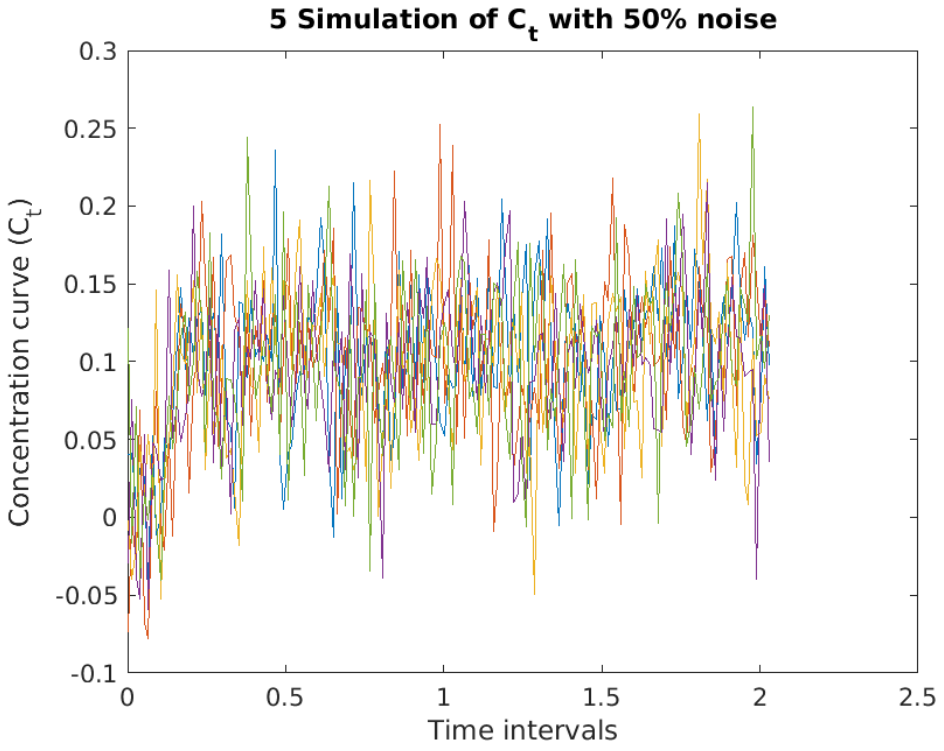





5 Simulation of C_t with 30% noise
Concentration curve (C_t)
Time intervals
0.2
0.15
0.1
0.05
0
-0.05
-0.1
0
0.5
1
1.5
2
2.5

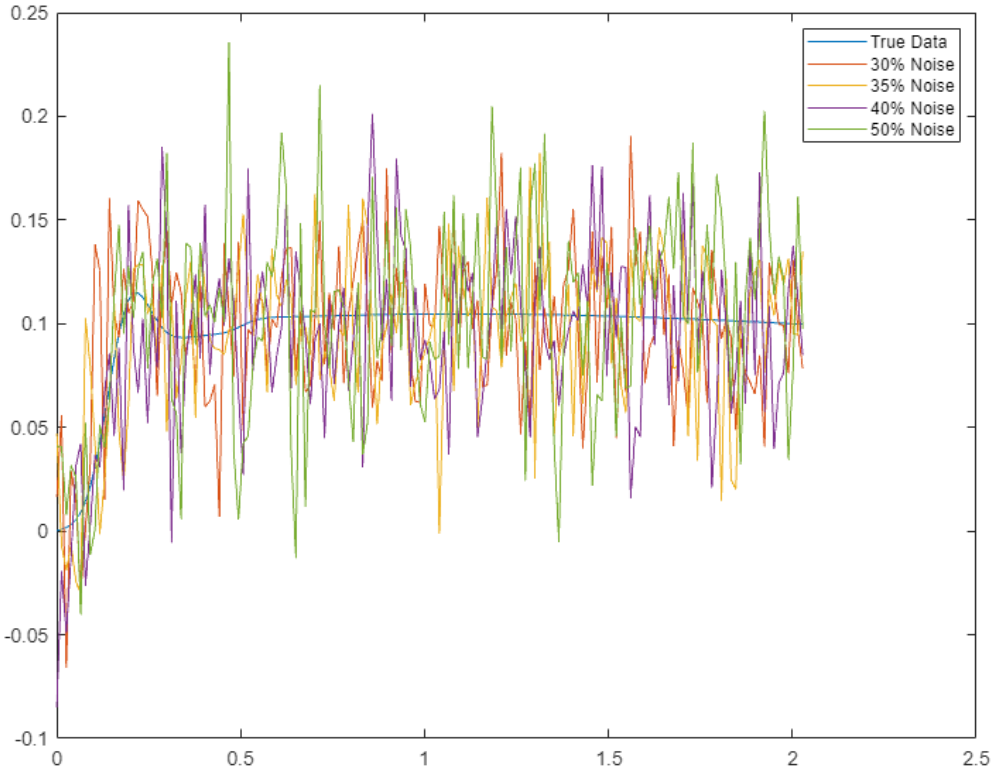

5 Simulation of C_t with 20% noise
Concentration curve (C_t)
Time intervals
0.2
0.15
0.1
0.05
0
-0.05
0
0.5
1
1.5
2
2.5





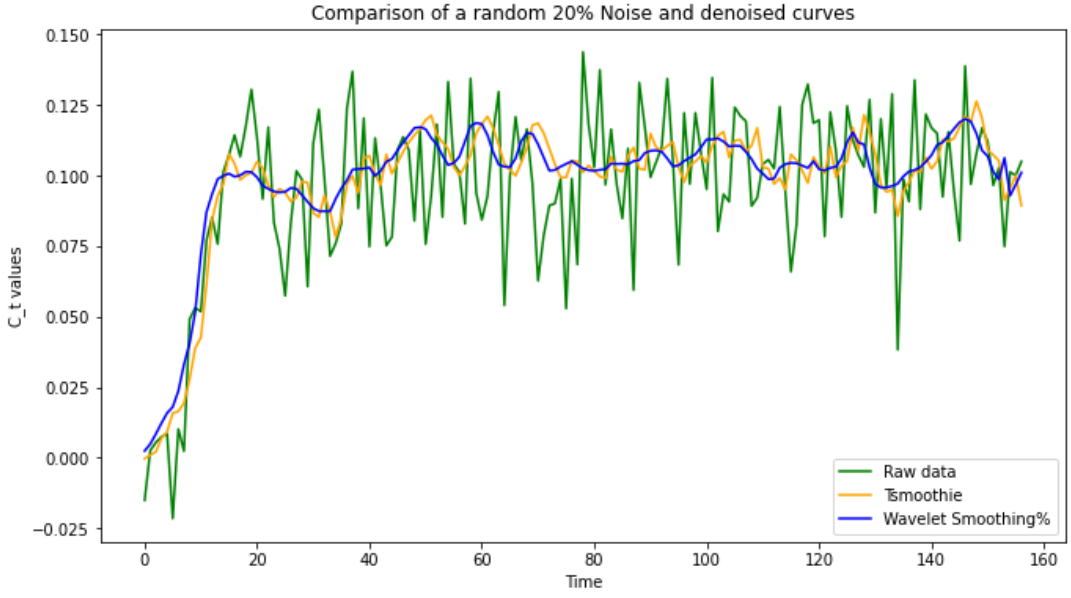
As is clearly visible from the graphs, The more noise level is added to the signal the more distorted the curve becomes. Which is also evident from the below curve, where the blue line denotes the true curve, but as the noise level increases the deviation increases.

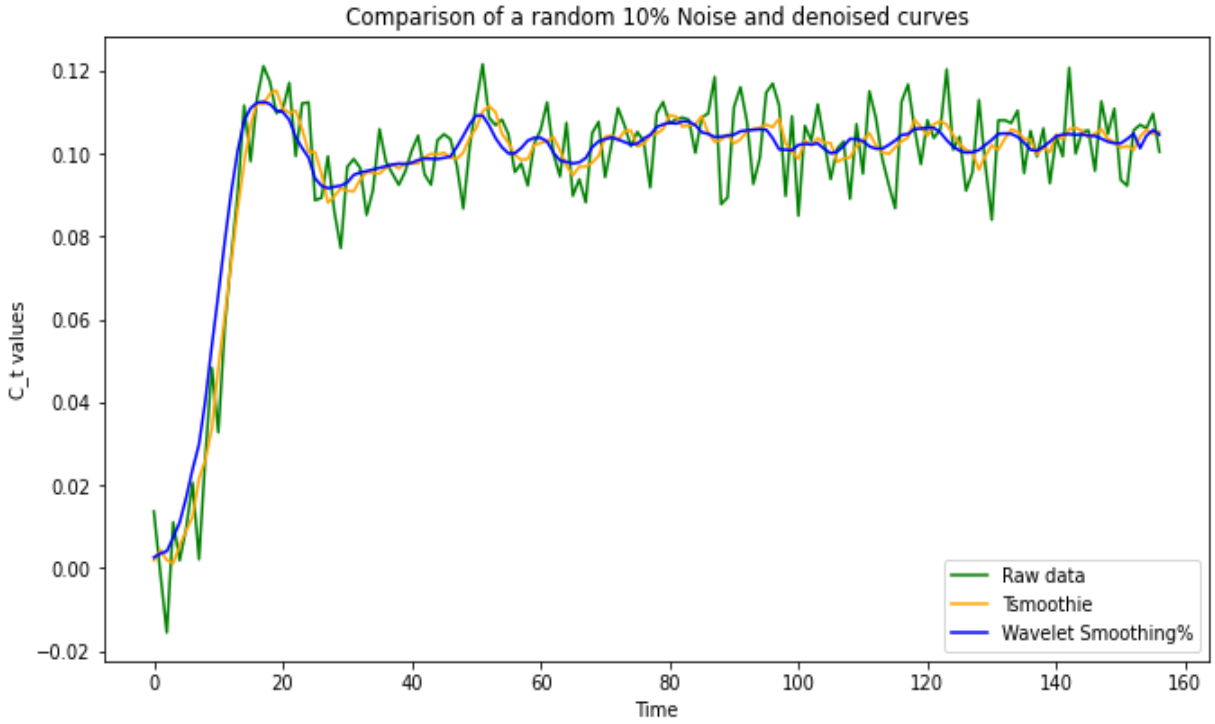
Simulation at noise levels from 30 to 50

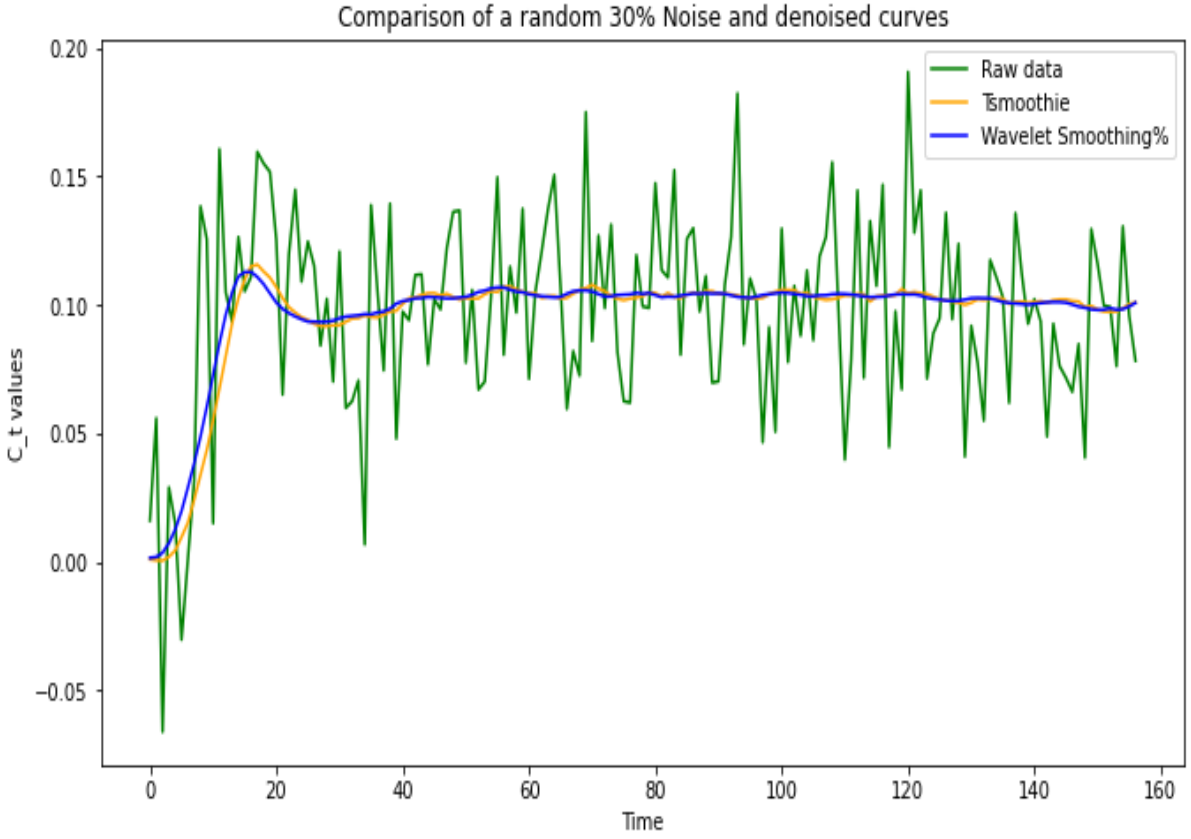


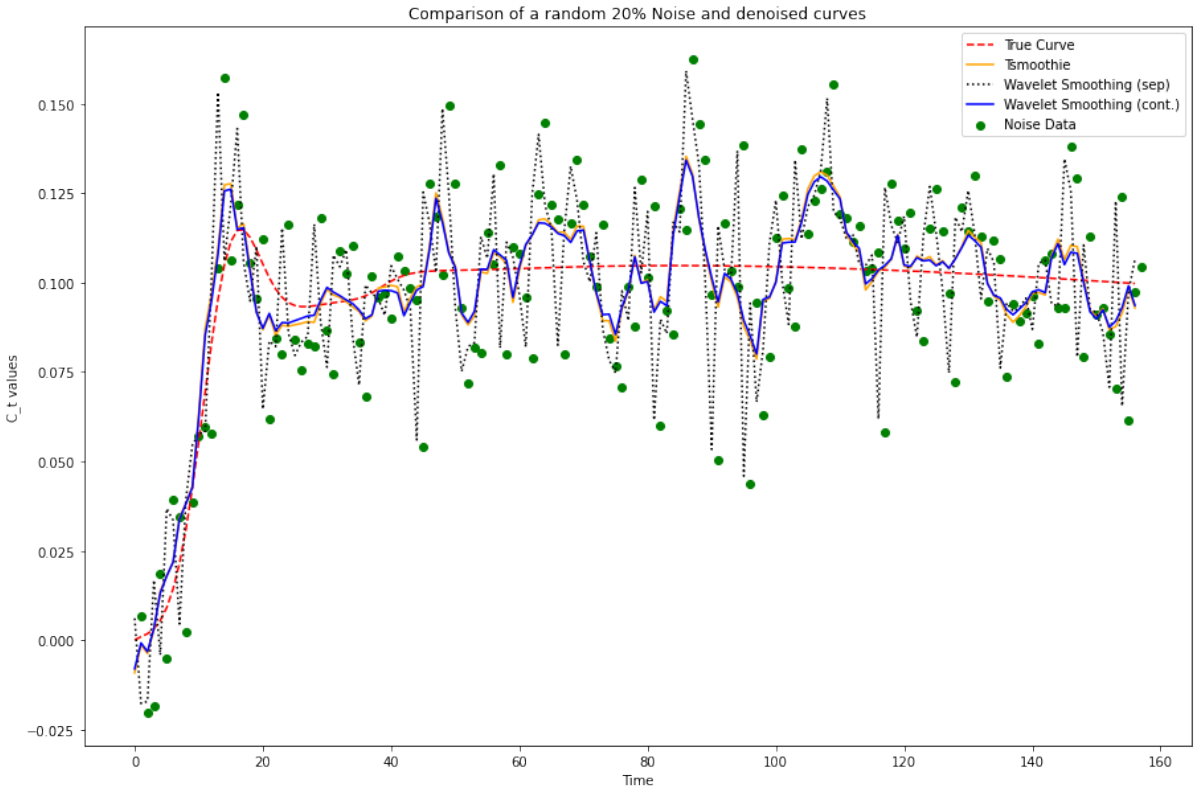
4. Smoothening Curves:

Convolution Filtering was used in Tsmoothie (python package), further Wavelet Denoising was also applied. In the resulting curves as it is clearly visible the filtering techniques reduced down noise by a huge margin.




Comparison of a random 10% Noise and denoised curves
C_t values
Time
Raw data
Tsmoothie
Wavelet Smoothing%

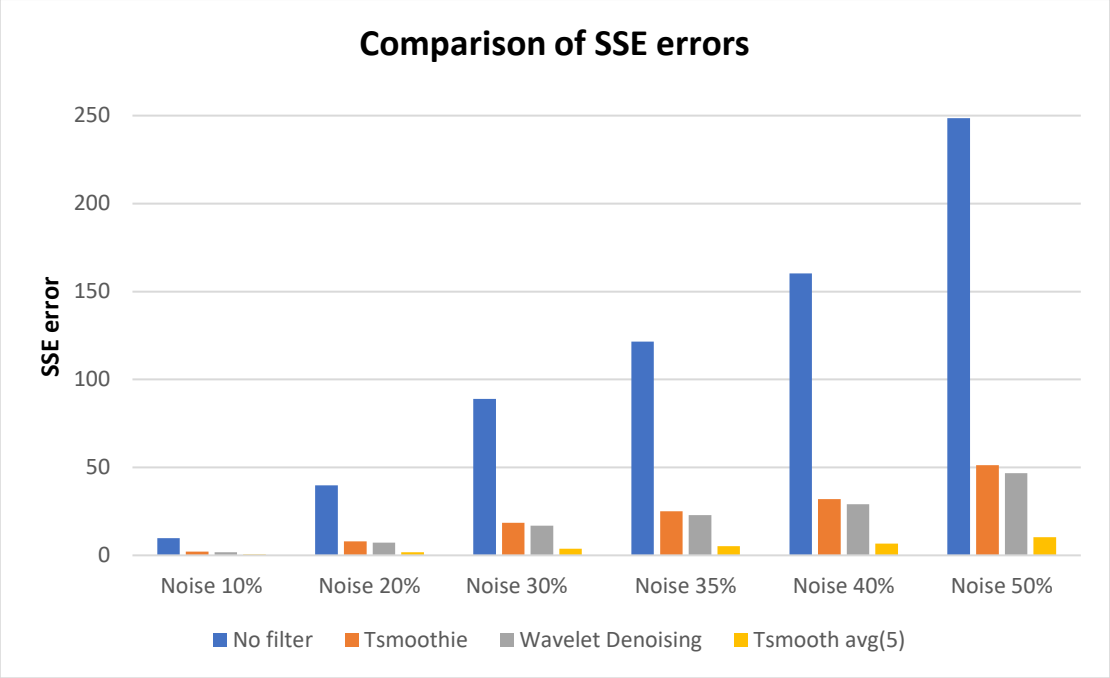

Comparison of a random 30% Noise and denoised curves
C_t values
Time
Raw data
Tsmoothie
Wavelet Smoothing%



Here Wavelet Smoothing (sep.) is when wavelet smoothing is applied to the noised data, and wavelet smoothing (contd.) is when it is applied to data denoised after tsmoothie. As, seen from this curve, The Wavelet smoothing and the Tsmoothie curve overlaps each other thus indicating that not a substantial improvement has been rendered by wavelet smoothing. This point is confirmed when Wavelet denoising is applied with T smoothie on the raw Noise data. So, in this graph, the convolutional filtering was able to reduce to reduce down the noise from the 20% Noise curve, but there is still a noticeable level of dissimilarity between the true curve and the denoised curve. In an attempt to remove this Moving average filtering was performed.

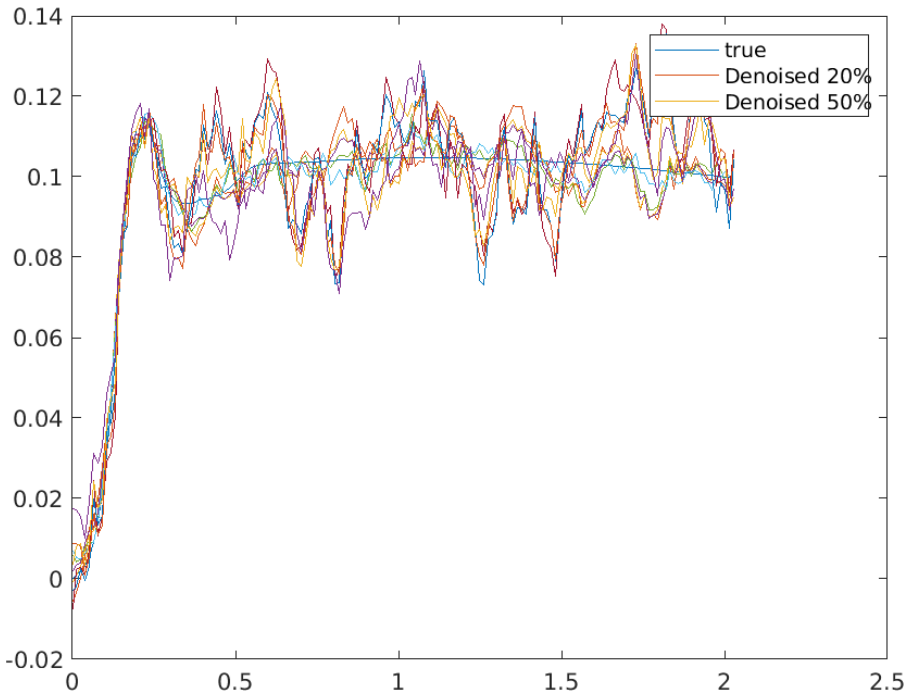
Also, to further smoothen the curve and reduce the noise levels, 5 point moving average of the T smoothie curve was then taken which then again slashed down the error. This error is measured as Sum of Square error. Below is the table which quantifies this error.

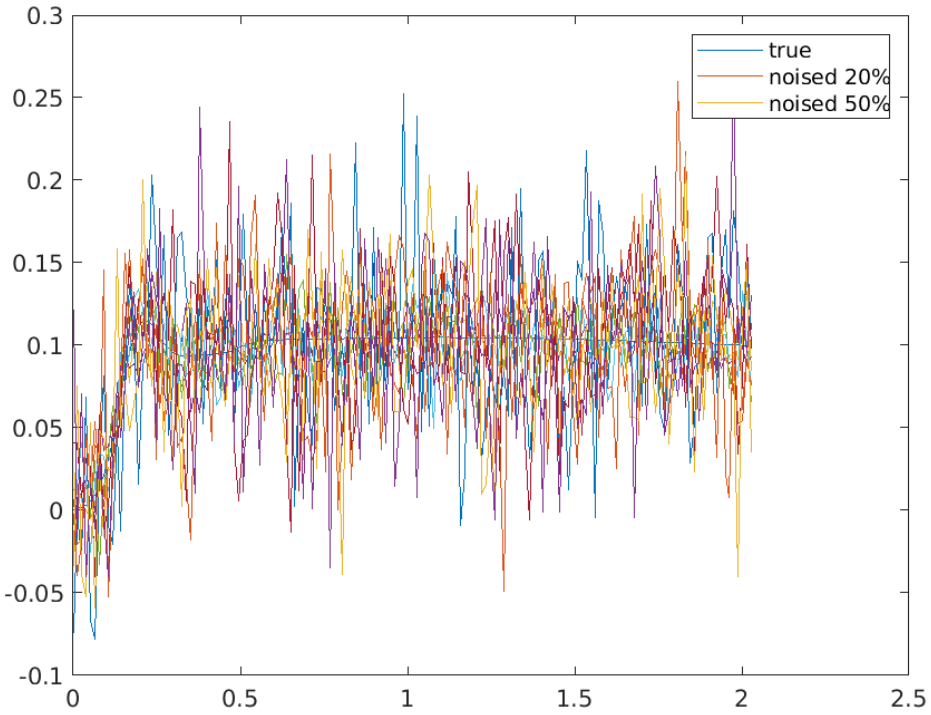
SSE(e-03)	No filter	Tsmoothie	Wavelet Denoising	Tsmooth Avg.(5)
Noise 10%	9.8	2.1	1.8	0.481
Noise 20%	39.8	8	7.2	1.7
Noise 30%	89	18.6	16.9	3.7
Noise 35%	121.5	25	22.8	5.2
Noise 40%	160.2	31.9	29.1	6.7
Noise 50%	248.5	51.2	46.8	10.4



From the Graph, we conclude that the Sum of square errors increases as the noise level is increasing, which is something expected as more noise leads to more error. The SSE reduced down when applying convolution filtering, and the results were improved further when Wavelet denoising was applied after convolutional filtering. But this improvement was not very noticeable. Therefore, we employed 5 point moving average, which was able to reduce the noise significantly.

5. Side by Side comparison of the Noise and the Denoised Curves against true Ct:



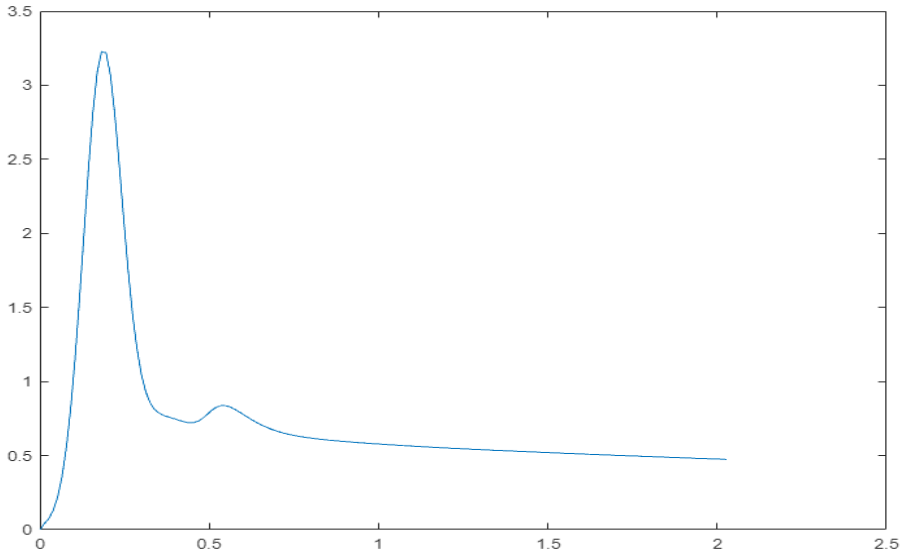


Evident enough our denoised curves looks much less cluttered than the noised curves.

6. Parameter Estimation:

Since we were using lsqcurvefit, we made an initial guess for the parameters of Ct, and the curve according to it looks like this K_trans, K_ep, V_p = [0.5, 4, 0.5];

Ct curve based on initial guess parameters

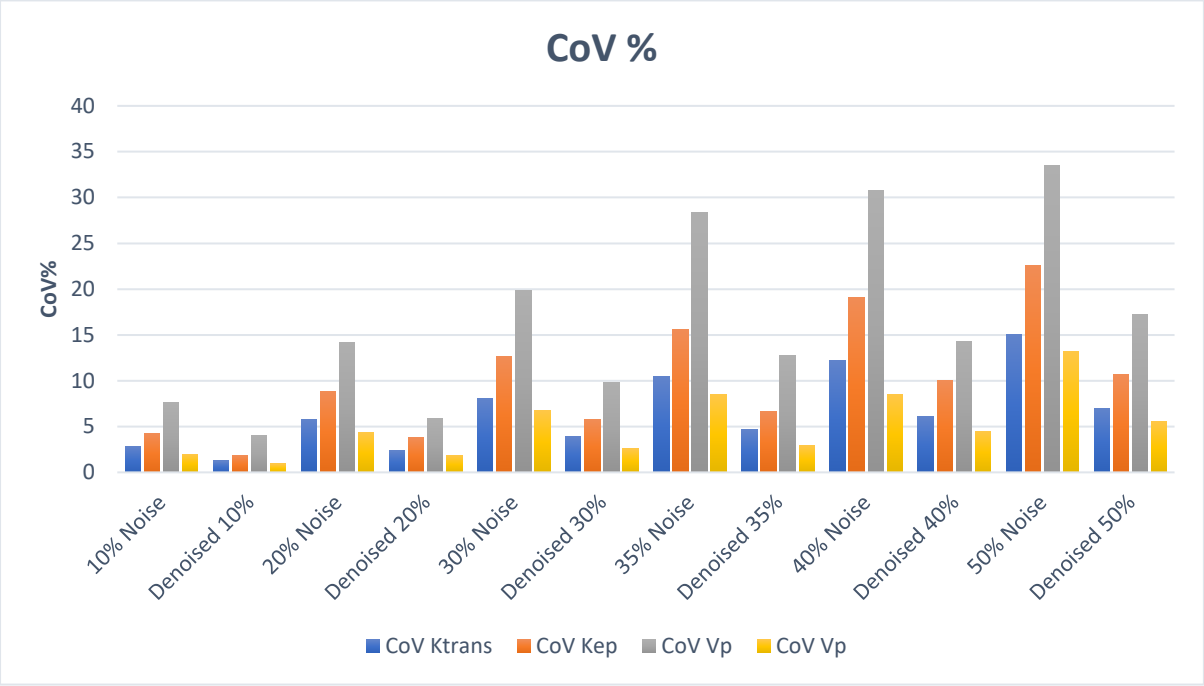


7. Fitting Parameters to our Denoised Curve.

After denoising our curve, we fitted our curve again the true Ct curve using the non-linear fitting function lsqcurvefit in MATLAB.

CoV (coefficient of Variation) of fitted parameters:

	CoV K_trans	CoV Kep	CoV Vp	CoV Ve
10% Noise	2.828	4.272	7.64	1.989
Denoised 10%	1.244	1.862	4.014	0.9314
20% Noise	5.804	8.871	14.19	4.3553
Denoised 20%	2.359	3.818	5.934	1.824
30% Noise	8.103	12.687	19.841	6.7503
Denoised 30%	3.868	5.77	9.801	2.575
35% Noise	10.43	15.58	28.418	8.482
Denoised 35%	4.626	6.657	12.799	2.883
40% Noise	12.215	19.054	30.8	8.4697
Denoised 40%	6.119	10.003	14.295	4.502
50% Noise	15.068	22.561	33.532	13.2449
Denoised 50%	7	10.686	17.205	5.53



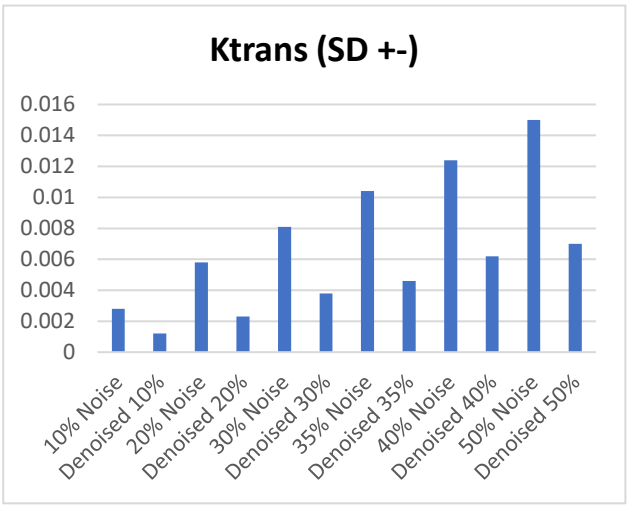
As it is visible from the graph, as the Coefficient of Variation increase with the increase in the noise level both in the NOISED and the DENOISED curves. Further this increase is maximum of Vp parameter and the minimum for K_trans parameter.

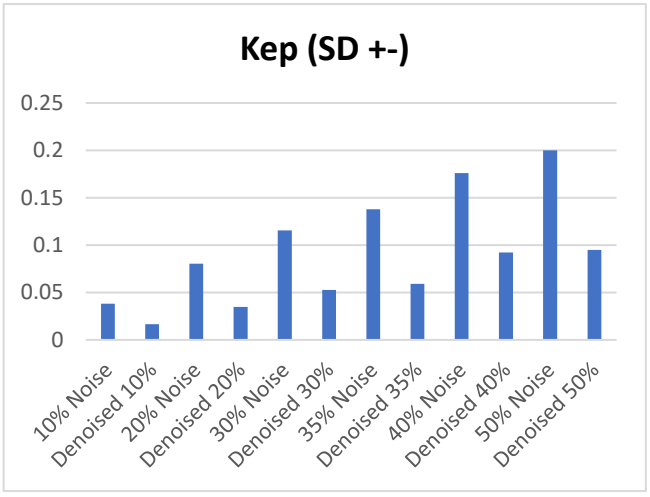
Tabular Summary of the Fit Parameters:

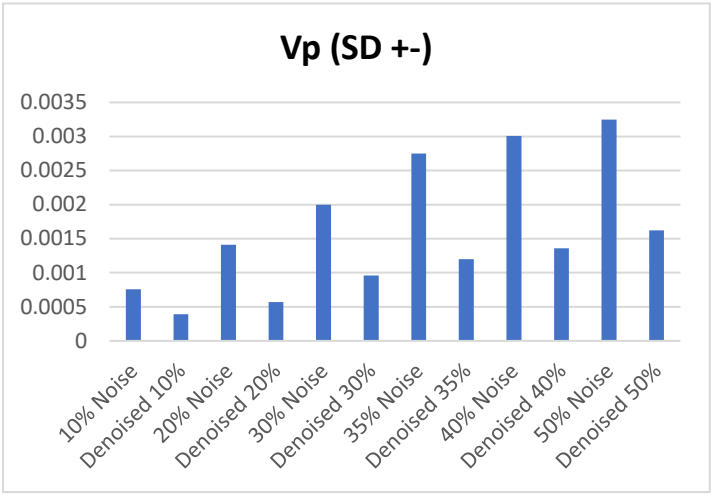
	$k_{trans} \pm sd$	$k_{ep} \pm sd$	$V_p \pm sd$	$V_e \pm sd$
10 % Noise	0.0995 ± 0.0028	0.893 ± 0.0381	0.01 ± 0.0007	0.111 ± 0.0022
Denoised 10%	0.1 ± 0.0012	0.896 ± 0.0166	0.00972 ± 0.0004	0.111 ± 0.0010
20% Noise	0.1 ± 0.0058	0.908 ± 0.0805	0.0099 ± 0.0014	0.111 ± 0.0048
Denoised 20%	0.1011 ± 0.0023	0.911 ± 0.0348	0.0096 ± 0.0006	0.111 ± 0.0020
30% Noise	0.1 ± 0.0081	0.911 ± 0.1157	0.01 ± 0.0020	0.111 ± 0.0075
Denoised 30%	0.1 ± 0.0038	0.913 ± 0.0527	0.0098 ± 0.0009	0.110 ± 0.0028
35% Noise	0.0997 ± 0.0104	0.885 ± 0.1379	0.0096 ± 0.0027	0.113 ± 0.0096
Denoised 35%	0.0998 ± 0.0046	0.886 ± 0.059	0.0094 ± 0.0012	0.113 ± 0.0032
40% Noise	0.102 ± 0.0124	0.924 ± 0.1761	0.0097 ± 0.0030	0.111 ± 0.0095
Denoised 40%	0.102 ± 0.0062	0.923 ± 0.0932	0.00954 ± 0.0013	0.111 ± 0.005
50% Noise	0.099 ± 0.015	0.886 ± 0.2	0.0097 ± 0.0032	0.115 ± 0.0152
Denoised 50%	0.1 ± 0.007	0.89 ± 0.0951	0.0094 ± 0.0016	0.112 ± 0.0062

It is visible from the table that higher noise levels mean more dispersion of parameters from the true value as highlighted from the Graphs below. The Denoised curves reduce this dispersion but the overall trend to increase with the noise levels remains the same.

From the Table it is interesting to see that whatever may be the Noise level the mean parameter values tend to be the same as the true parameter values. But this is expected as the we had introduced random noise into the curves so the positive and the negative biases in our estimated parameter values tend to cancel out each other when we take their average. So, better approach was to take the Relative error for each of the parameter value and take and average of that, this would work because the relative error is strictly positive.







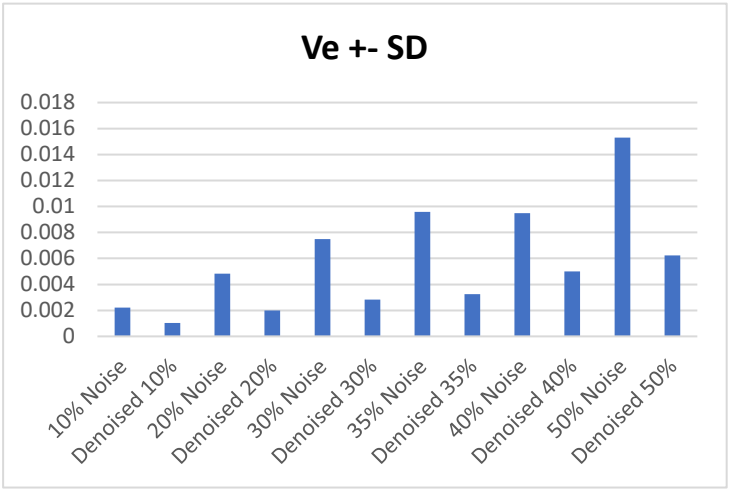
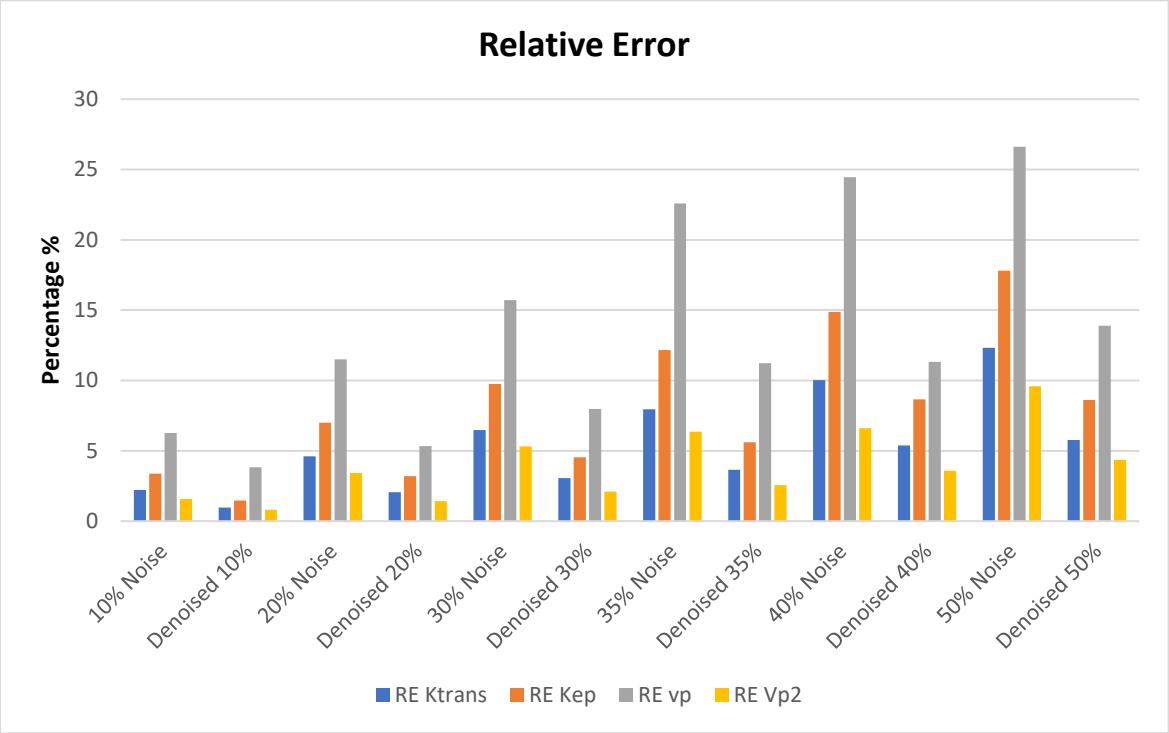


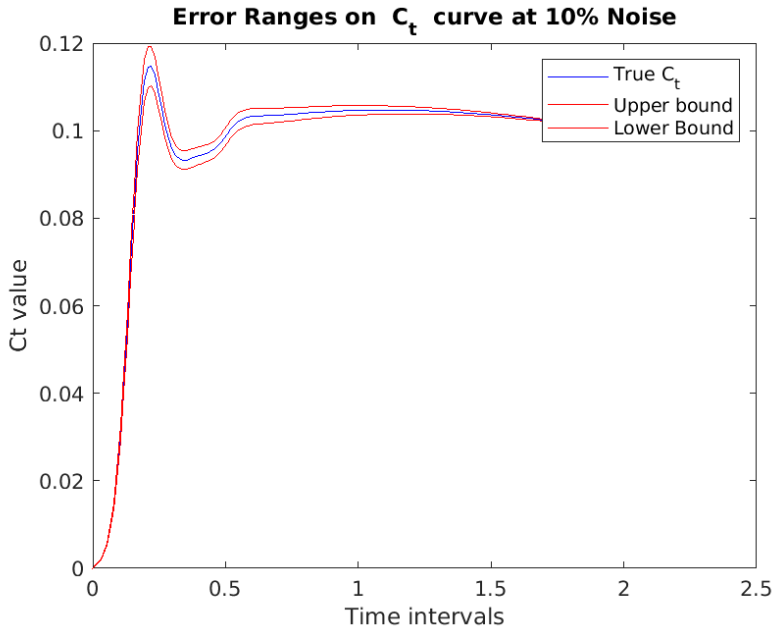
Table for Relative Error(in Percentage)

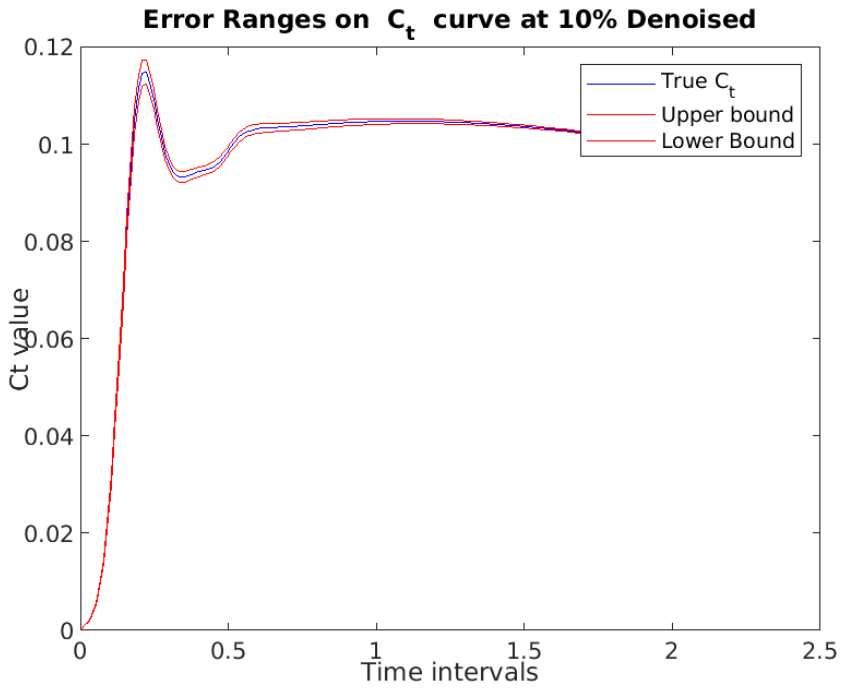
Relative % error	$k_{trans} \pm sd$	$k_{ep} \pm sd$	$V_p \pm sd$	$V_e \pm sd$
10 % Noise	2.22 ± 1.77	3.39 ± 2.63	6.26 ± 4.34	1.57 ± 1.27
Denoised 10%	0.96 ± 0.77	1.46 ± 1.20	3.83 ± 2.83	0.81 ± 0.64
20% Noise	4.60 ± 3.615	6.98 ± 5.623	11.50 ± 8.148	3.42 ± 2.681
Denoised 20%	2.06 ± 1.61	3.19 ± 2.52	5.33 ± 3.95	1.43 ± 1.13
30% Noise	6.48 ± 4.91	9.75 ± 8.42	15.72 ± 12.41	5.32 ± 4.10
Denoised 30%	3.05 ± 2.52	4.55 ± 3.97	7.98 ± 5.62	2.11 ± 1.56
35% Noise	7.94 ± 6.67	12.16 ± 9.39	22.59 ± 15.85	6.37 ± 6.38
Denoised 35%	3.65 ± 2.81	5.61 ± 3.66	11.23 ± 7.10	2.57 ± 2.06
40% Noise	10.02 ± 7.67	14.87 ± 12.93	24.44 ± 17.55	6.61 ± 5.41
Denoised 40%	5.38 ± 3.75	8.66 ± 6.035	11.32 ± 8.95	3.58 ± 2.70
50% Noise	12.33 ± 8.56	17.81 ± 13.25	26.63 ± 18.87	9.60 ± 10.58
Denoised 50%	5.77 ± 3.91	8.62 ± 6.14	13.89 ± 10.00	4.36 ± 3.83

The Table highlights following results, that a 10% random noise in the signal produces a 2% error in Ktrans, 3.4% error in Kep and more than 6% error in Vp. This error is reduced to less than 1% on denoised Ktrans, around 1.5% for denoised Kep and around 4% for denoised Vp. But it is noticeable throughout the table that the maximum error is reflected in Vp then in Kep and least in Ktrans.

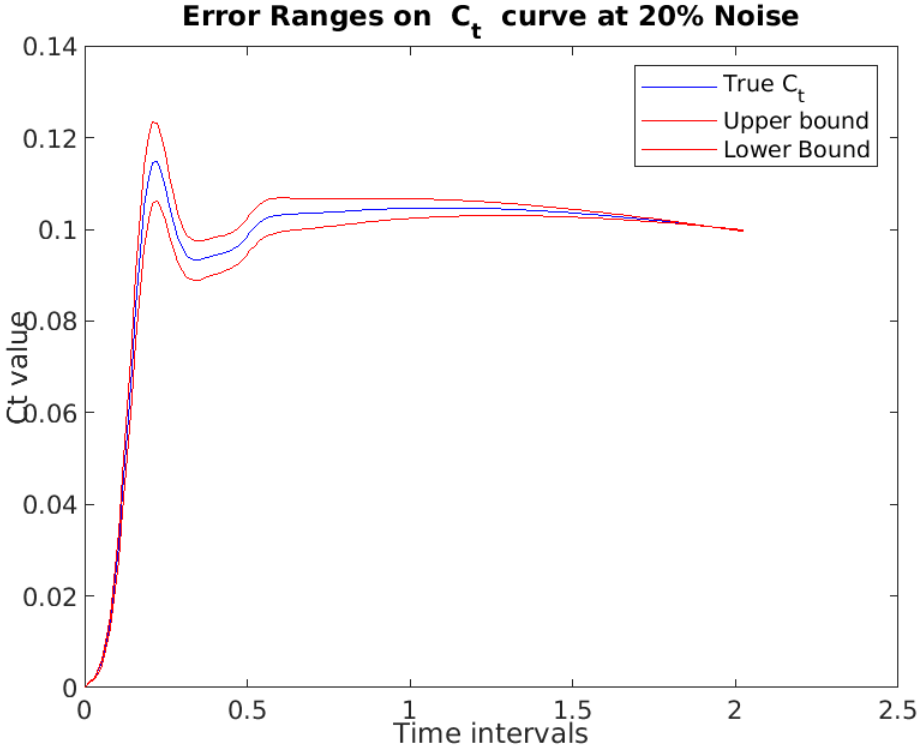


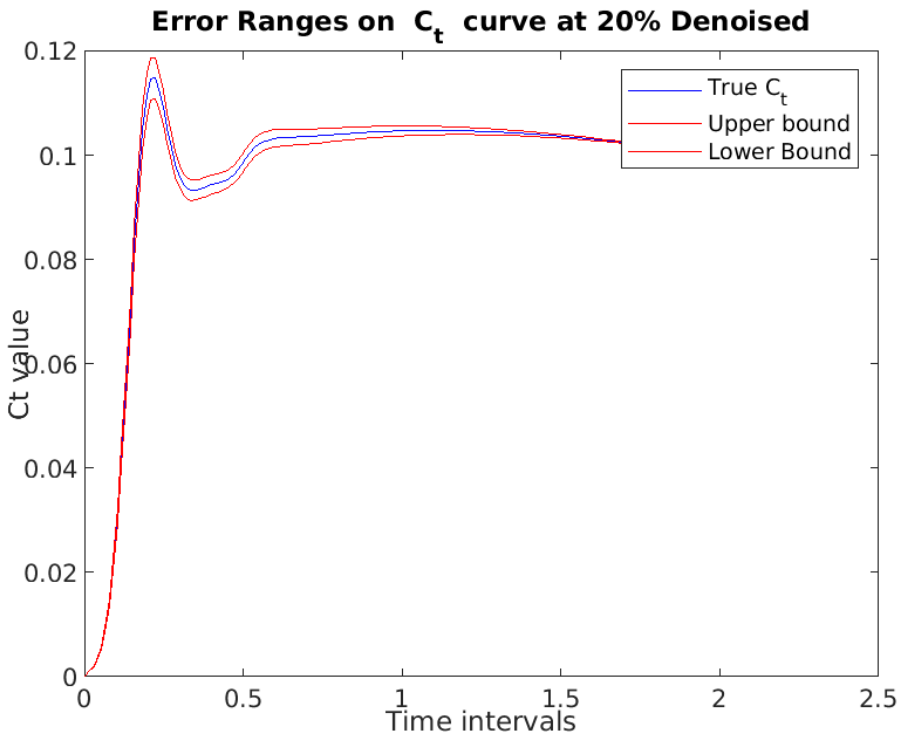
8. Plots to highlight the error ranges in the curve.

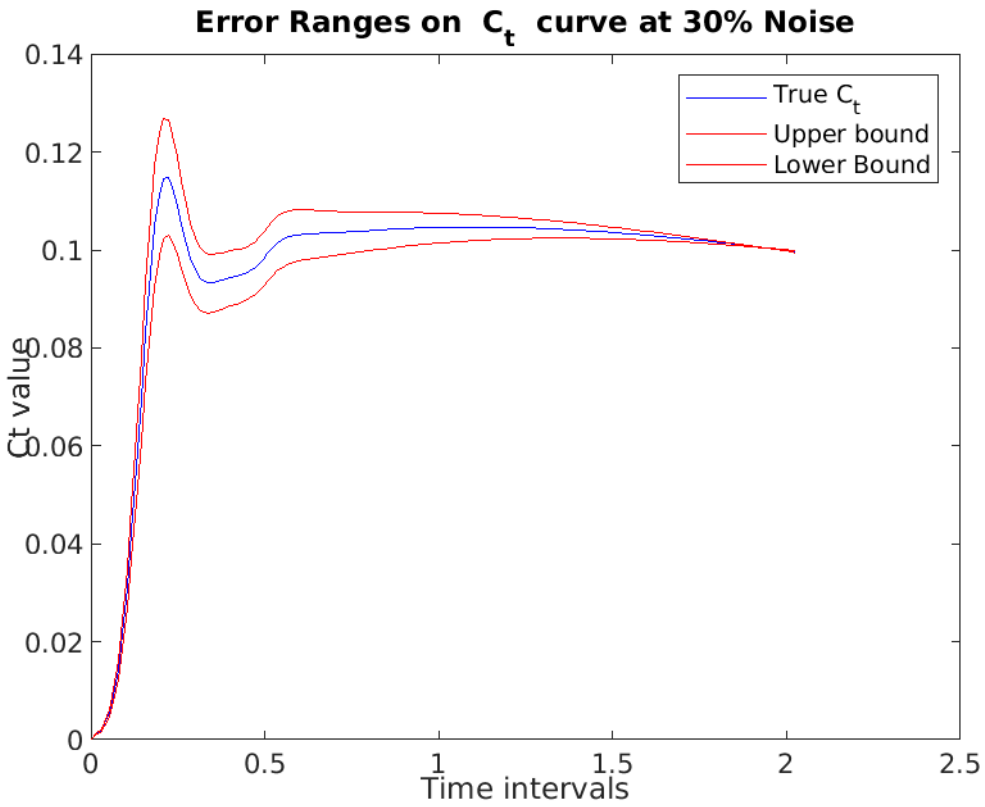


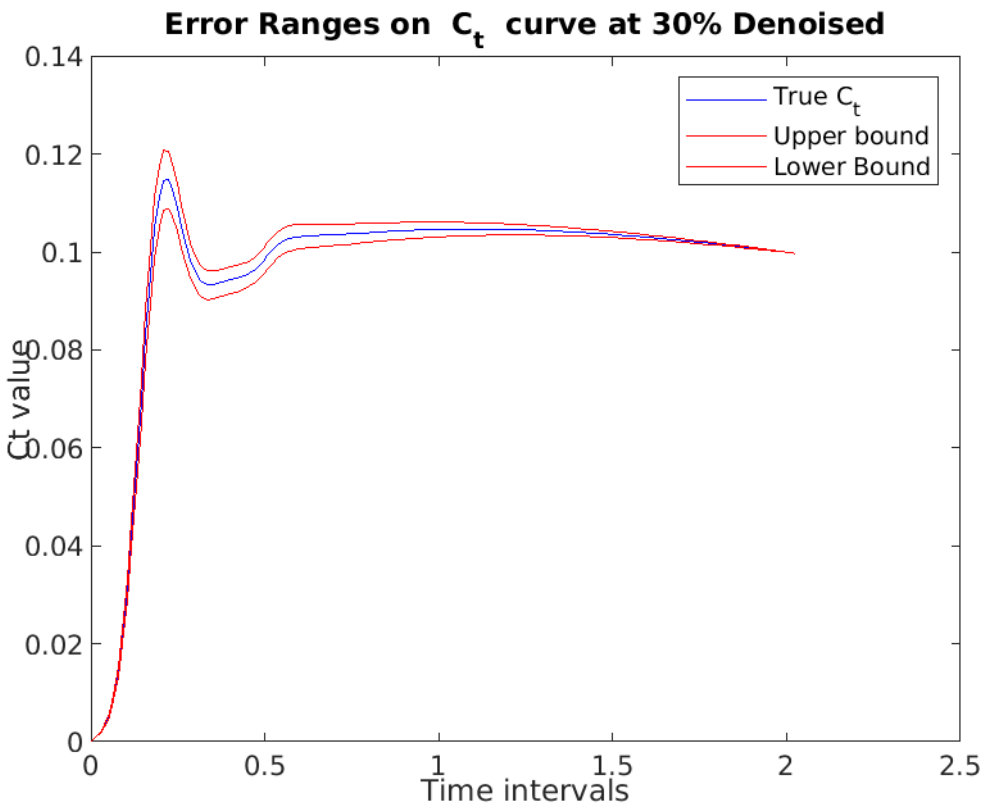


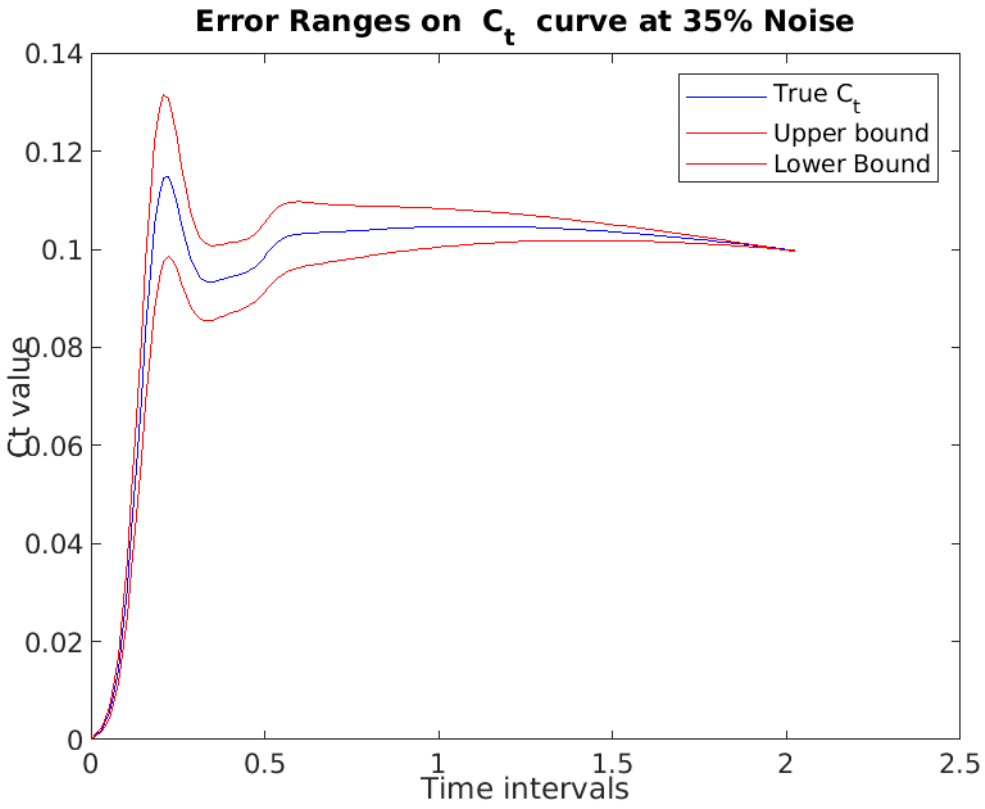
As, can be noticed in the above two graphs how the lower and the upper bounds tend to narrow for the denoised curve as compared to noised curve, indicating to the fact the denoised curves have lower error and narrow range. Thus, it is better model than the noised curves. Similarly, we can observe the same things in for the other graphs as well.

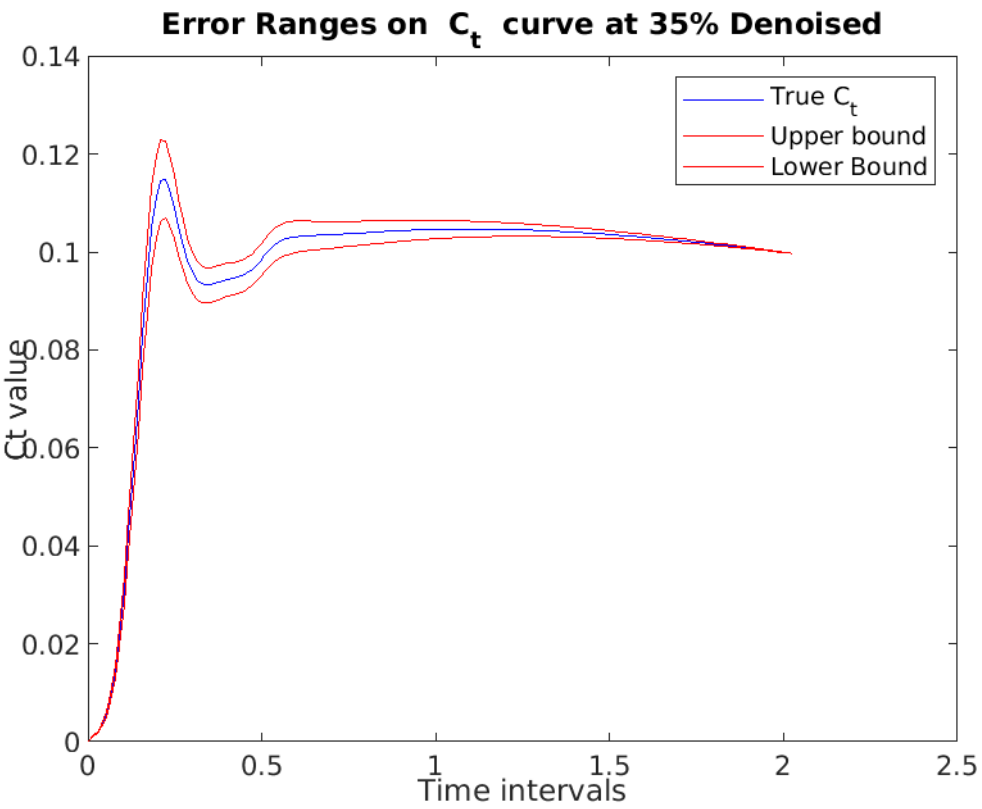


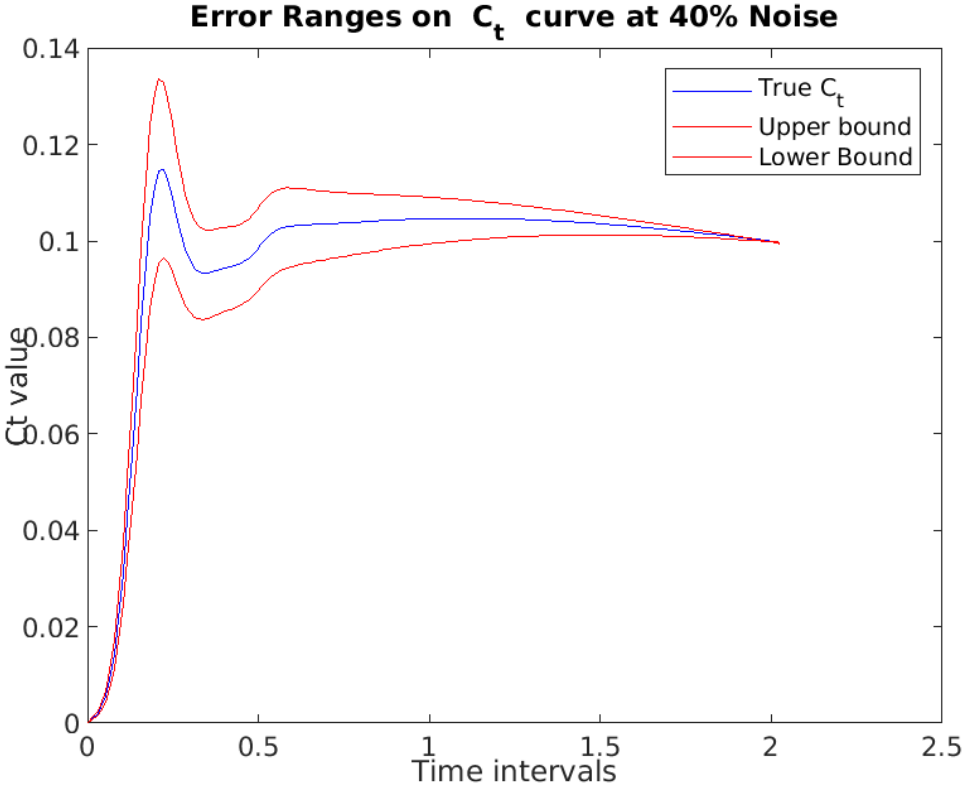

Error Ranges on C_t curve at 20% Denoised
True C_t
Upper bound
Lower Bound
Ct value
Time intervals
0.12
0.1
0.08
0.06
0.04
0.02
0
0
0.5
1
1.5
2
2.5

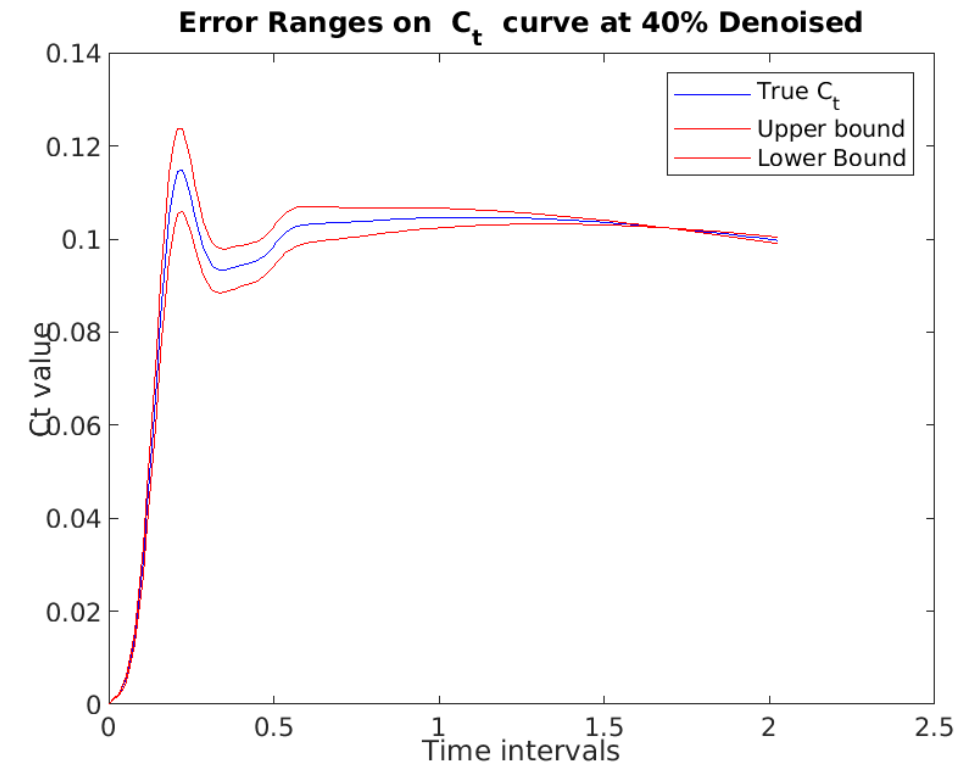

Error Ranges on C_t curve at 30% Noise
True C_t
Upper bound
Lower Bound
Ct value
Time intervals
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
0.5
1
1.5
2
2.5

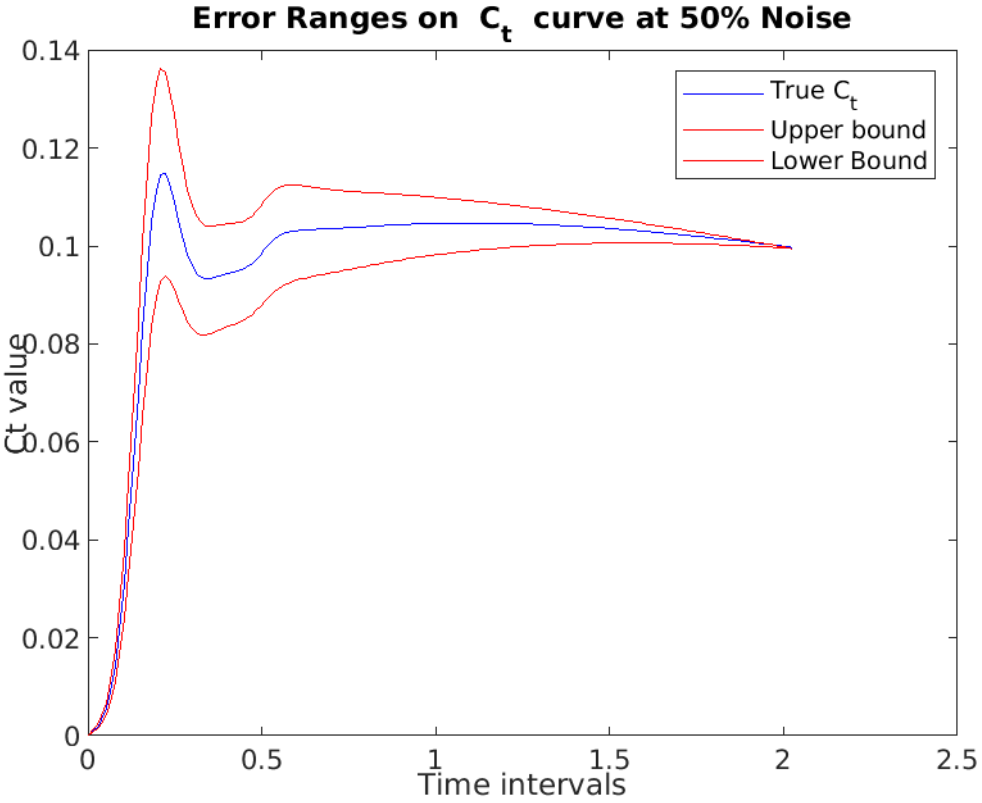

Error Ranges on C_t curve at 30% Denoised
True C_t
Upper bound
Lower Bound
Ct value
Time intervals
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.5
1
1.5
2
2.5

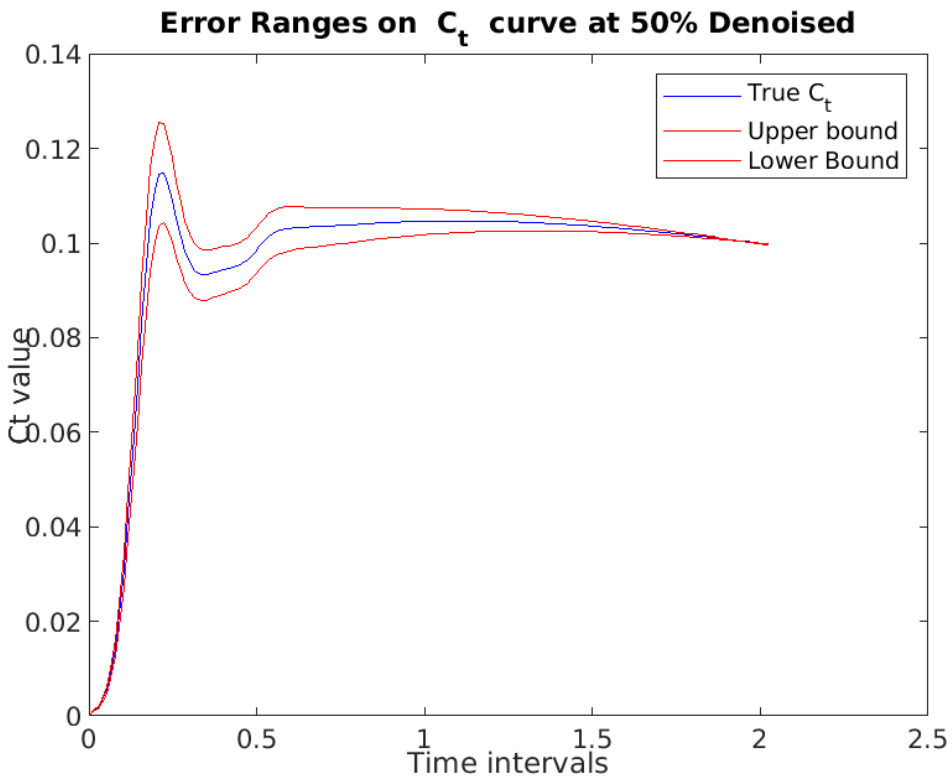

Error Ranges on C_t curve at 35% Noise
True C_t
Upper bound
Lower Bound
Ct value
Time intervals
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.5
1
1.5
2
2.5


Error Ranges on C_t curve at 35% Denoised
True C_t
Upper bound
Lower Bound
Ct value
Time intervals
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0
0.5
1
1.5
2
2.5


Error Ranges on C_t curve at 40% Noise
True C_t
Upper bound
Lower Bound
Ct value
Time intervals
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0
0.5
1
1.5
2
2.5


Error Ranges on C_t curve at 40% Denoised
True C_t
Upper bound
Lower Bound
Ct value
Time intervals
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
0.5
1
1.5
2
2.5


Error Ranges on C_t curve at 50% Noise
True C_t
Upper bound
Lower Bound
Ct value
Time intervals
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
0.5
1
1.5
2
2.5



9. Types of Filters Used:

1. **Median Filter:**

In median filter the way to reduce noise is to take median values of the signal in a specified window and replace all the other values in the same window by that median value. But as our noise was not “salt and pepper”, no significant improvement was found from this filtering method.

2. **Wavelet Denoising Filter.**

The basic idea behind wavelet denoising, or wavelet thresholding, is that the wavelet transform leads to a sparse representation for many real-world signals and images. What this means is that the wavelet transform concentrates signal and image features in a few large-magnitude wavelet coefficients. Wavelet coefficients which are small in value are typically noise and you can "shrink" those coefficients or remove them without affecting the signal or image quality. After you threshold the coefficients, you reconstruct the data using the inverse wavelet transform.

3**. Convolutional Smoother:**

ConvolutionSmoother operates convolutions of fixed dimensions on the series using a weighted-windows. The weights can assume different format but they are the same for all the windows and fixed for the whole procedure. The series are padded, reflecting themself,

with a quantity equal to the window size in both ends to avoid loss of information. The ConvolutionSmoother automatically vectorizes, in an efficient way, the desired smoothing operation on all the series received.

```
Parameters
----------
window_len : int
    Greater than equal to 1. The length of the window used to compute
    the convolutions.

window_type : str
    The type of the window used to compute the convolutions.
    Supported types are: 'ones', 'hanning', 'hamming', 'bartlett', 'blackman'.

copy : bool, default=True
    If True, the raw data received by the smoother and the smoothed
    results can be accessed using 'data' and 'smooth_data' attributes.
    This is useful to calculate the intervals. If set to False the
    interval calculation is disabled. In order to save memory, set it to
    False if you are interested only in the smoothed results.

Attributes
----------
smooth_data : array of shape (series, timesteps)
    Smoothed data derived from the smoothing operation. It is accessible
    after computhing smoothing, otherwise None is returned.

data : array of shape (series, timesteps)
    Raw data received by the smoother. It is accessible with 'copy'=True
    and  after computhing smoothing, otherwise None is returned.
```