Simulation & Modelling of Dynamical Systems

Offline Estimation of Dynamical System Parameters

Nikolaos Konstas

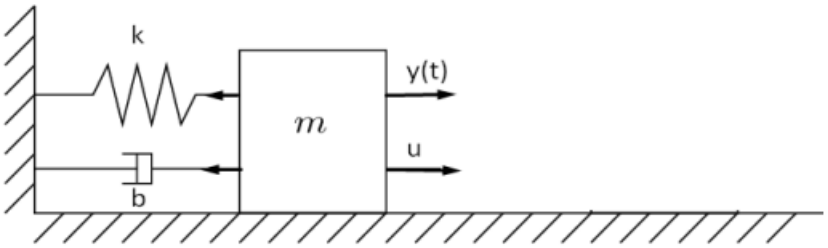
Department of Electrical & Computer Engineering
Aristotle University of Thessaloniki

April 2022

Contents

1 Problem 1 - Mass-Spring-Damper

In the first problem we were asked to model and estimate the parameters of the following system, which consists of a mass, a spring and a damper.



3 forces are exerted on the body: a) h force of the spring, b) h force of the damper, c) h external force u.
By $y(t)$ we denote the displacement of the mass from the equilibrium position.

1.1 Theoretical Analysis

The force exerted by the spring is expressed by the relation:

$$F_k = -k \cdot y$$

, while the force exerted by the damper:

$$F_b = -b \cdot \dot{y}$$

From the *Fundamental Law of Mechanics* it is:

$$\begin{aligned} m \cdot a &= \sum F \Rightarrow \\ m \cdot a &= F_b + F_k + u \Rightarrow \\ m \cdot \ddot{y} &= -b \cdot \dot{y} - k \cdot y + u \Rightarrow \\ \ddot{y} &= -\frac{b}{m} \cdot \dot{y} - \frac{k}{m} \cdot y + \frac{1}{m} \cdot u \end{aligned} \tag{1}$$

(1) can be easily expressed in tabular form as follows:

$$\ddot{y} = \begin{bmatrix} \frac{b}{m} & \frac{k}{m} & \frac{1}{m} \end{bmatrix} \cdot \begin{bmatrix} -\dot{y} & -y & u \end{bmatrix}^T \tag{2}$$

1.1.1 Linear Parameterization

We have put the differential equation that describes our system in the form $y^{(n)} = \theta^{*T} \cdot \Delta$, but the question is to parametrize the system linearly in the form $y = \theta_\lambda^{*T} \cdot \zeta$, where the signal ζ is produced by measurements of the external force u and the displacement y.

For this purpose I create the filter $\Lambda(s)$ which is of degree equal to the maximum order of the derivative of y, i.e. 2nd degree. This filter needs to be stable so I define it as:

$$\Lambda(s) = (s + \rho_1) \cdot (s + \rho_2) = s^2 + (\rho_1 + \rho_2) \cdot s + \rho_1 \cdot \rho_2$$

, where ρ_1 and ρ_2 are real positive numbers.

I also define the vector λ:

$$\lambda = \begin{bmatrix} \rho_1 + \rho_2 & \rho_1 \cdot \rho_2 \end{bmatrix}^T$$

Let θ_1 be the part of θ related to the terms of the output y and θ_2 the part related to the external force u. Thus $\theta_\lambda *$ becomes:

$$\theta_\lambda^{*T} = \begin{bmatrix} \theta_1^{*T} - \lambda^T & \theta_2^{*T} \end{bmatrix}$$
$$\theta_\lambda^{*T} = \begin{bmatrix} \frac{b}{m} - (\rho_1 + \rho_2) & \frac{k}{m} - \rho_1 \cdot \rho_2 & \frac{1}{m} \end{bmatrix}$$

As for table g, it is obtained by applying the Λ filter to the input and output as follows:

$$\zeta = \begin{bmatrix} -\frac{\Delta_1^{T(s)}}{\Lambda(s)} \cdot y & \frac{\Delta_1^{T(s)}}{\Lambda(s)} \cdot u \end{bmatrix}^T$$
$$\zeta = \begin{bmatrix} -\frac{\begin{bmatrix} s & 1 \end{bmatrix}}{\Lambda(s)} \cdot y & \frac{\begin{bmatrix} s & 1 \end{bmatrix}}{\Lambda(s)} \cdot u \end{bmatrix}^T$$
$$\zeta = \begin{bmatrix} -\frac{s}{\Lambda(s)} \cdot y & -\frac{1}{\Lambda(s)} \cdot y & \frac{s}{\Lambda(s)} \cdot u & \frac{1}{\Lambda(s)} \cdot u \end{bmatrix}^T$$

Thus, the system is now linearly parameterized and we can estimate its parameters by measurements of the output y and the external force u.

1.1.2 Least Squares Method

To estimate the parameters m, b, k of the system, the method of least squares will be used.

At this point we must take into account that y and u are measured signals and therefore discrete in time. For this reason I define the vector Y and the matrix Φ:

$$Y = \begin{bmatrix} y(1) & y(2) & ... & y(N) \end{bmatrix}^T$$

$$\Phi = \begin{bmatrix} \zeta_1(1) & \zeta_2(1) & \zeta_3(1) & \zeta_4(1) \\ \zeta_1(2) & \zeta_2(2) & \zeta_3(2) & \zeta_4(2) \\ \vdots & \vdots & \vdots & \vdots \\ \zeta_1(N) & \zeta_2(N) & \zeta_3(N) & \zeta_4(N) \end{bmatrix}$$

Where N is the number of measurements, Y is a $N \times 1$ vector that contains the measurements of y and Φ is a $N \times 4$ that contains in each column the measurements of the corresponding size of the array z.
So we can now express the matrix Y in the following way:

$$Y = \Phi \cdot \theta_\lambda^*$$

Suppose that:

$$\hat{Y} = \Phi \cdot \theta_0$$

the estimate of the matrix Y, where θ_0 is the estimate of the vector θ_λ^*. Define the estimation error:

$$E = Y - \hat{Y}$$

L.S.M. is based on the minimization of the square of the error. More specifically in the minimization of the quantity:

$$V = \left| \frac{E}{2} \right|^2 \Rightarrow$$
$$V = \left| \frac{Y - \hat{Y}}{2} \right|^2 \Rightarrow$$
$$V = \left| \frac{Y - \Phi \cdot \theta_0}{2} \right|^2$$

We notice that the function V is convex with respect to θ, and therefore it will have a total minimum at the zero point of the partial derivative with

respect to θ.

$$\left.\frac{\partial V}{\partial \theta}\right|_{\theta=\theta_0} = 0 \Rightarrow$$
$$(Y - \Phi \cdot \theta_0)^T \cdot (-\Phi) = 0 \Rightarrow$$
$$-Y^T \cdot \Phi + \theta_0^{\Phi} \cdot (\Phi^T \cdot \Phi) = 0 \Rightarrow$$
$$\theta_0^T \cdot (\Phi^T \cdot \Phi) = Y^T \cdot \Phi \tag{3}$$

We can also solve in terms of θ_0^T:

$$\theta_0^T = Y^T \cdot \Phi \cdot (\Phi^T \cdot \Phi)^{-1}$$

, but we will avoid this form, because there may be difficulties (instability) in inverting the matrix $\Phi^T \cdot \Phi$. There are also implemented numerical calculation methods which solve equation (3) faster and more accurately.

Therefore the optimal value of θ, where the estimation error of the output is minimized, will be at θ_0, as we have shown above, and we can continue with the implementation in MATLAB.

1.2 Implementation in MATLAB & Results

1.2.1 Implementation in MATLAB

The full implementation can be found in project: Mass-Spring-Damper contained in this folder, but schematically it states that the program follows these steps:

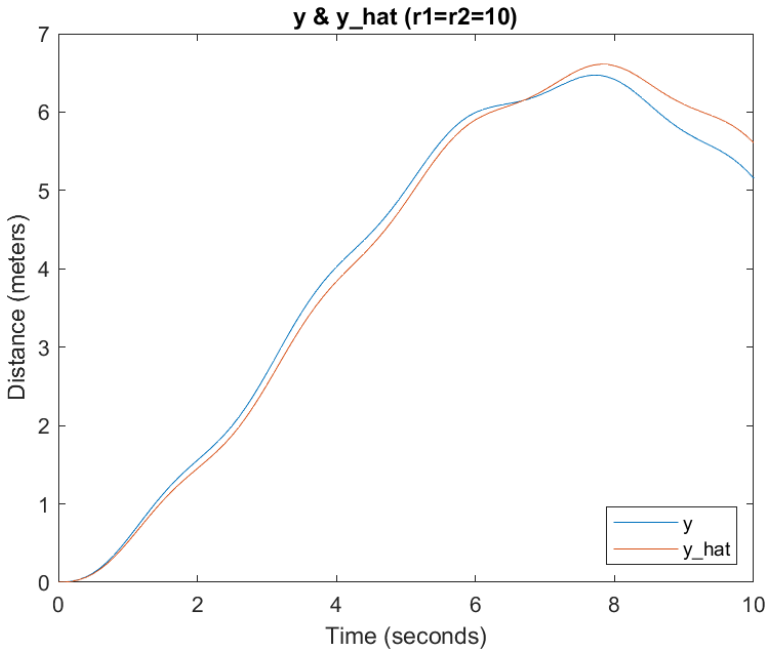
- Define the actual m,b,k values.
- Initialization of vector t (time), initial values $y(0)$ & $\dot{y}(0)$ and external force u.
- Find the vector Y by solving d.e.(1) with the help of **ode45** after defining appropriate state equations.
- Initializing the filter $\Lambda(s)$ and finding the matrix Φ by applying the filters of the matrix ζ.
- Definition of a method for the application of the L.S.M., making use of **mrdivide**, and the estimation of the parameters m,b,k.
- Iterative loop to find the optimal model for different poles of $\Lambda(s)$.

1.2.2 Results & Remarks

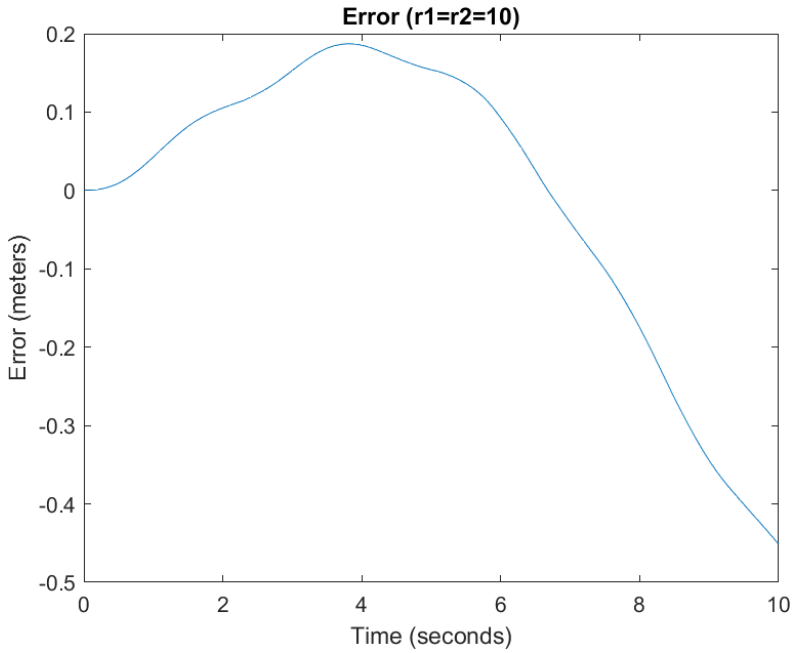
I start my tests with a very simple case, placing a double pole at -10 for the $\Lambda(s)$ filter. The estimated parameters take the values:

$\hat{m} = 10.8813$, $\hat{b} = 0.1932$ & $\hat{k} = 1.5030$

Comparing with the real values of the parameters, I notice that k is practically the same, m is quite close, while the biggest deviation exists in b. Here is a graph of $y(t)$ & $\hat{y}(t)$:



It is obvious that y & $\hat{y}$ have a similar form and up to about 8s the error is very small. Specifically, the average absolute error $(\frac{1}{N} \cdot |Y - \hat{Y}|)$ takes the value $e = 0.1510$, while below the error is shown as a function of time t. From the error diagram, we confirm that the largest deviations are for $t > 8s$ while we also notice that for $0 < t < 6.5s$ $\hat{y}$ is smaller than y, while for $t > 6.5s$ the opposite happens.



My next tests are for double pole at -1 and double pole at -100, with the respective results:

$(\rho_1 = \rho_2 = 1)$: $\hat{m} = 9.9850$, $\hat{b} = 0.3016$ & $\hat{k} = 1.4982$, $e = 0.0036$
$(\rho_1 = \rho_2 = 100)$: $\hat{m} = 10.00$, $\hat{b} = -0.3834$ & $\hat{k} = 1.5451$, $e = 4.0307$

For the double pole at -1 all parameters have negligible deviation from their true values, resulting in an error of the order of 10^{-3}.

Conversely for the double pole at -100, although we have good approximations for m and k, our algorithm returns a prohibitive value for b, which renders our model useless as is evident from the corresponding error.

We already suspect that the best pole values are in the $[-10, 0]$ interval, which we will study in detail below.

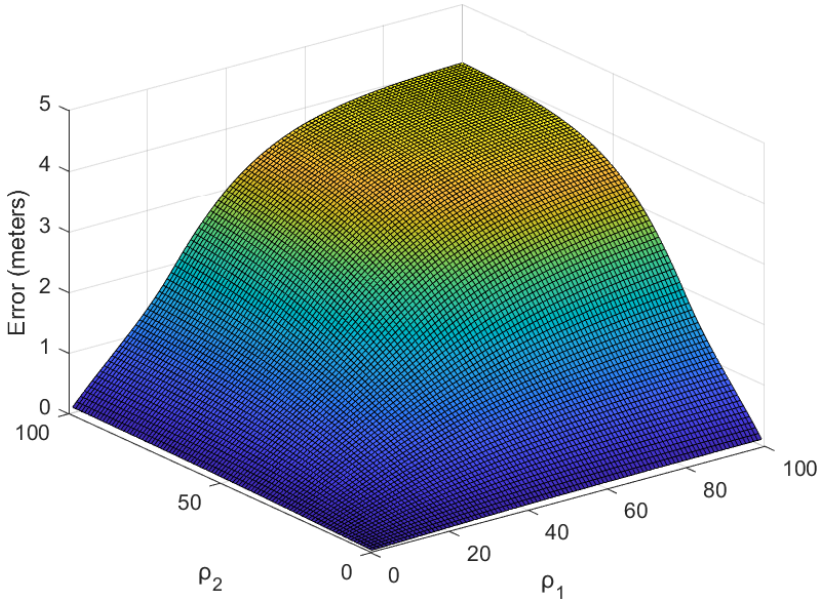
At this point, however, we do a test for poles at -1 and -10:

$(\rho_1 = 1, \rho_2 = 10)$: $\hat{m} = 10.0754$, $\hat{b} = 0.2816$, $\hat{k} = 1.4997$, $e = 0.0163$

I notice that the latter model is a very good approximation of the real system but the error it produces is larger than that of the double pole at 1. Thus, my suspicion is reinforced that the optimal model lies in the interval $[-10, 0]$, while we also have an indication that the double pole returns the best results.

I continue my tests with two iterative loops, looking for the best performance for poles in the interval $[-100, 0.1]$ and step 0.1.

The smallest error has the double-pole model at -1, while as ρ_1 and ρ_2 increase the error becomes larger and larger. This can be seen in the following graph which plots the error as a function of ρ_1 and ρ_2.



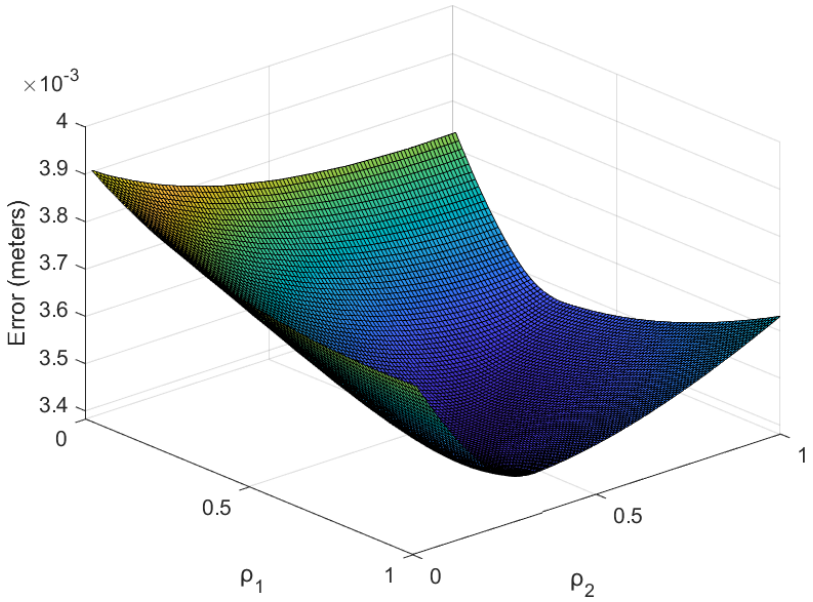
Now, I've established that the best performance is given for small values of ρ_1 and ρ_2 (poles of $\Lambda(s)$), so my next move is to investigate what happens for $\rho_1, \rho_2 < 1$.

So I reduce the step to 10^{-2} and rerun the algorithm for poles in the interval $[-1, -0.01]$.

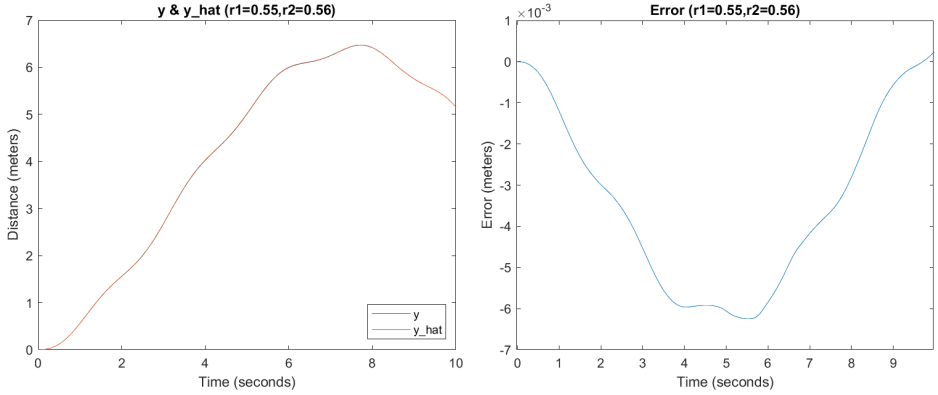
Here the situation is a little different, as there is no such 'monotony' as we encountered above. Specifically, the error is maximized as ρ_1 and ρ_2 approach 0 and 1, while it takes its minimum value for $\rho_1 = 0.55, \rho_2 = 0.56$ and the parameters take the values:

$\hat{m} = 9.9771$, $\hat{b} = 0.3034$, $\hat{k} = 1.4980$, $e = 0.0034$

All values are very close to actual values with small decimal deviations.

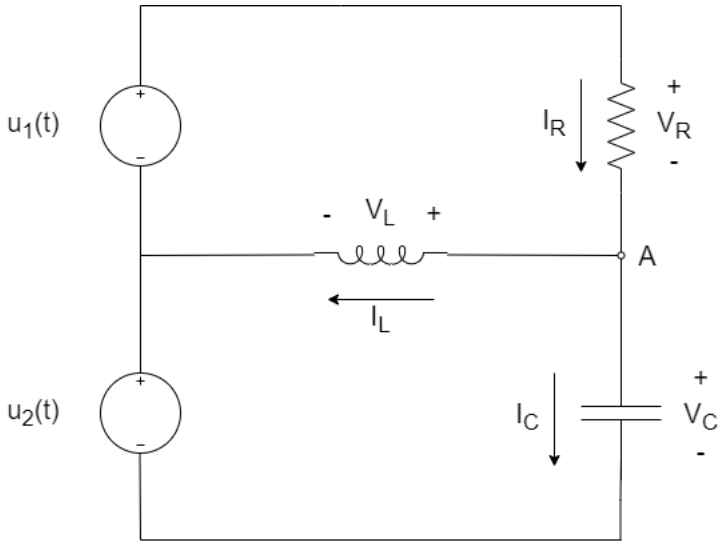


Probably, we could find an even more accurate model by further slicing the $[-1, -0.01]$ interval with a smaller step, but for the purposes of this project the result found is satisfactory, and the tests stop here.
Finally, with the implementation in MATLAB, we saw the utility and one of the applications of the model approach with the Method of Least Squares, facing any difficulties that arose and understanding the importance of the $\Lambda(s)$ filter, seeing how large has an effect on the formation of the final model. Here are the graphs of y, $\hat{y}$ & e for the optimal model ($\rho_1 = 0.55, \rho_2 = 0.56$). Notice, in the 1st graph, that $\hat{y}$ is so close to y that they practically coincide.



2 Problem 2 - Electric Circuit

In the 2nd problem, the goal was to estimate, with the Method of Least Squares, the transfer matrix of the circuit and to deal with some noise problems. Following is the diagram of the circuit enriched with voltages and currents that will be useful in the theoretical analysis.



2.1 Theoretical Analysis

2.1.1 Modeling

To begin with, I outline the relationships between V_R-I_R, V_C-I_C and V_L-I_L:

$$I_R = \frac{V_R}{R}$$
$$I_C = C \cdot \dot{V}_C$$
$$\dot{I}_L = \frac{V_L}{L}$$

Then I write the 3 loop equations and the equation of node A:

$$KVL_1 : u_1 = V_R + V_L \quad (1)$$
$$KVL_2 : u_2 = -V_L + V_C \quad (2)$$
$$KVL_3 : u_1 + u_2 = V_R + V_C \quad (3)$$
$$KVL_A : I_R = I_L + I_C \quad (4)$$

Where KVL_1 corresponds to the upper loop, KVL_2 corresponds to the lower loop, and KVL_3 corresponds to the outer loop.

I take relation (4) and derive:

$$(4) \xrightarrow{\frac{d}{dt}}$$

$$\dot{I}_R = \dot{I}_L + \dot{I}_C \xrightarrow{\text{From relations } V\text{-}I}$$

$$\frac{\dot{V}_R}{R} = \frac{V_L}{L} + C \cdot \ddot{V}_C \Rightarrow$$

$$V_L = \frac{L}{R} \cdot \dot{V}_R - LC \cdot \ddot{V}_C \tag{5}$$

Now I take (1) and replace V_L from (5):

$$(1) \xrightarrow{(5)}$$

$$u_1 = V_R + \frac{L}{R} \cdot \dot{V}_R - LC \cdot \ddot{V}_C \xrightarrow{(3)}$$

$$u_1 = V_R + \frac{L}{R} \cdot \dot{V}_R - LC \cdot \ddot{u}_1 - LC \cdot \ddot{u}_2 + LC \cdot \ddot{V}_R \Rightarrow$$

$$\ddot{V}_R + \frac{1}{RC} \cdot \ddot{V}_R + \frac{1}{LC} \cdot V_R = \ddot{u}_1 + \frac{1}{LC} \cdot u_1 + \ddot{u}_2 \Rightarrow$$

$$\ddot{V}_R = \begin{bmatrix} \frac{1}{RC} & \frac{1}{LC} & 1 & 0 & \frac{1}{LC} & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} -\dot{V}_R & -V_R & \ddot{u}_1 & \dot{u}_1 & u_1 & \ddot{u}_2 & \dot{u}_2 & u_2 \end{bmatrix}^T \tag{6}$$

Now to find the differential equation describing V_C, I substitute in (2) from (5):

$$(2) \xrightarrow{(5)}$$

$$u_2 = -\frac{L}{R} \cdot \dot{V}_R + LC \cdot \ddot{V}_C + V_C \xrightarrow{(3)}$$

$$u_2 = -\frac{L}{R} \cdot \dot{u}_1 - \frac{L}{R} \cdot \dot{u}_2 + \frac{L}{R} \cdot \dot{V}_C + LC \cdot \ddot{V}_C + V_C \Rightarrow$$

$$\ddot{V}_C + \frac{1}{RC} \cdot \dot{V}_C + \frac{1}{LC} \cdot V_C = \frac{1}{RC} \cdot \dot{u}_1 + \frac{1}{RC} \cdot \dot{u}_2 + \frac{1}{LC} \cdot u_2 \Rightarrow$$

$$\ddot{V}_C = \begin{bmatrix} \frac{1}{RC} & \frac{1}{LC} & \frac{1}{RC} & 0 & \frac{1}{RC} & \frac{1}{LC} \end{bmatrix} \cdot \begin{bmatrix} -\dot{V}_C & -V_C & \dot{u}_1 & u_1 & \dot{u}_2 & u_2 \end{bmatrix}^T \tag{7}$$

We have now constructed for both outputs equations of the form $\ddot{V} = \theta^T \cdot \Delta$ and we continue below for the linear parameterization.

2.1.2 Linear Parameterization

In a similar way to the 1st Problem, I define 2 filters $\Lambda_1(s)$ and $\Lambda_2(s)$, for V_R and V_C respectively:

$$\Lambda_1(s) = (s+\rho_1)\cdot(s+\rho_2) = s^2 + (\rho_1+\rho_2)\cdot s + \rho_1\rho_2$$
$$\Lambda_2(s) = (s+\rho_3)\cdot(s+\rho_4) = s^2 + (\rho_3+\rho_4)\cdot s + \rho_3\rho_4$$

, where $\rho_1, \rho_2, \rho_3, \rho_4 \in \mathbf{R}_+^*$.

As the analytical procedure has been described in the 1st Topic, the final forms are presented:

For V_R:

$$\theta_{\lambda_1}^* = \begin{bmatrix} \frac{1}{RC} - (\rho_1+\rho_2) & \frac{1}{LC} - \rho_1\rho_2 & 1 & 0 & \frac{1}{LC} & 1 & 0 & 0 \end{bmatrix}^T$$
$$\zeta_1 = \begin{bmatrix} -\frac{\begin{bmatrix} s & 1 \end{bmatrix}}{\Lambda_1(s)}\cdot V_R & \frac{\begin{bmatrix} s^2 & s & 1 \end{bmatrix}}{\Lambda_1(s)}\cdot u_1 & \frac{\begin{bmatrix} s^2 & s & 1 \end{bmatrix}}{\Lambda_1(s)}\cdot u_2 \end{bmatrix}^T$$

For V_C:

$$\theta_{\lambda_2}^* = \begin{bmatrix} \frac{1}{RC} - (\rho_3+\rho_4) & \frac{1}{LC} - \rho_3\rho_4 & \frac{1}{RC} & 0 & \frac{1}{RC} & \frac{1}{LC} \end{bmatrix}^T$$
$$\zeta_2 = \begin{bmatrix} -\frac{\begin{bmatrix} s & 1 \end{bmatrix}}{\Lambda_2(s)}\cdot V_C & \frac{\begin{bmatrix} s & 1 \end{bmatrix}}{\Lambda_2(s)}\cdot u_1 & \frac{\begin{bmatrix} s & 1 \end{bmatrix}}{\Lambda_2(s)}\cdot u_2 \end{bmatrix}^T$$

2.1.3 Transfer Table

To find the Transfer Matrix of the system I return to relations (6) and (7) and apply the *Laplace* transform.

$$(6) \overset{\mathcal{L}}{\Rightarrow}$$

$$\mathcal{L}\left(\ddot{V}_R + \frac{1}{RC}\cdot\ddot{V}_R + \frac{1}{LC}\cdot V_R\right) = \mathcal{L}\left(\ddot{u}_1 + \frac{1}{LC}\cdot u_1 + \ddot{u}_2\right) \xRightarrow[\dot{u}_1(0)=u_1(0)=0, \dot{u}_2(0)=u_2(0)=0]{\dot{V}_R(0)=V_R(0)=0}$$

$$s^2\cdot V_R(s) + \frac{1}{RC}s\cdot V_R(s) + \frac{1}{LC}\cdot V_R(s) = s^2\cdot u_1(s) + \frac{1}{LC}\cdot u_1(s) + s^2\cdot u_2(s) \Rightarrow$$

$$\left(s^2 + \frac{1}{RC}s + \frac{1}{LC}\right)\cdot V_R(s) = \left(s^2 + \frac{1}{LC}\right)\cdot u_1(s) + s^2\cdot u_2(s) \Rightarrow$$

$$V_R(s) = \left(\frac{s^2 + \frac{1}{LC}}{s^2 + \frac{1}{RC}s + \frac{1}{LC}}\right)\cdot u_1(s) + \left(\frac{s^2}{s^2 + \frac{1}{RC}s + \frac{1}{LC}}\right)\cdot u_2(s) \qquad (8)$$

So I have found the 1st row of the Transfer Table (which has dimension 2×2, since we have 2 inputs and 2 outputs). In a similar way I calculate the second line of the Transfer Table.

$$(7) \overset{\mathcal{L}}{\Rightarrow}$$

$$\mathcal{L}\left(\ddot{V}_C + \frac{1}{RC} \cdot \dot{V}_C + \frac{1}{LC} \cdot V_C\right) = \mathcal{L}\left(\frac{1}{RC} \cdot \dot{u}_1 + \frac{1}{RC} \cdot \dot{u}_2 + \frac{1}{LC} \cdot u_2\right) \xRightarrow[\dot{u}_1(0)=u_1(0)=0, \dot{u}_2(0)=u_2(0)=0]{\dot{V}_C(0)=V_C(0)=0}$$

$$s^2 \cdot V_C(s) + \frac{1}{RC} s \cdot V_C(s) + \frac{1}{LC} \cdot V_C(s) = \frac{1}{RC} s \cdot u_1(s) + \frac{1}{RC} s \cdot u_2(s) + \frac{1}{LC} \cdot u_2(s) \Rightarrow$$

$$\left(s^2 + \frac{1}{RC} s + \frac{1}{LC}\right) \cdot V_C(s) = \frac{1}{RC} s \cdot u_1(s) + \left(\frac{1}{RC} s + \frac{1}{LC}\right) \cdot u_2(s) \Rightarrow$$

$$V_C(s) = \left(\frac{\frac{1}{RC} s}{s^2 + \frac{1}{RC} s + \frac{1}{LC}}\right) \cdot u_1(s) + \left(\frac{\frac{1}{RC} s + \frac{1}{LC}}{s^2 + \frac{1}{RC} s + \frac{1}{LC}}\right) \cdot u_2(s) \qquad (9)$$

So finally the Transfer Table of the system is:

$$H(s) = \begin{bmatrix} \dfrac{s^2 + \frac{1}{LC}}{s^2 + \frac{1}{RC} s + \frac{1}{LC}} & \dfrac{s^2}{s^2 + \frac{1}{RC} s + \frac{1}{LC}} \\ \dfrac{\frac{1}{RC} s}{s^2 + \frac{1}{RC} s + \frac{1}{LC}} & \dfrac{\frac{1}{RC} s + \frac{1}{LC}}{s^2 + \frac{1}{RC} s + \frac{1}{LC}} \end{bmatrix}$$

That is:

$$\begin{bmatrix} V_R(s) \\ V_C(s) \end{bmatrix} = H(s) \cdot \begin{bmatrix} u_1(s) \\ u_2(s) \end{bmatrix}$$

The form of the matrix H may seem complicated, but I do not make simplifications, since this form is particularly convenient to substitute after we estimate the parameters $\frac{1}{RC}$ and $\frac{1}{LC}$.

Regarding the Least Squares Method, it has been analyzed in section 1.1.2 of this report, so we proceed with the implementation in MATLAB.

2.2 Implementation in MATLAB & Results

2.2.1 Implementation in MATLAB

As the code has been delivered in this folder this part will simply outline the steps followed:

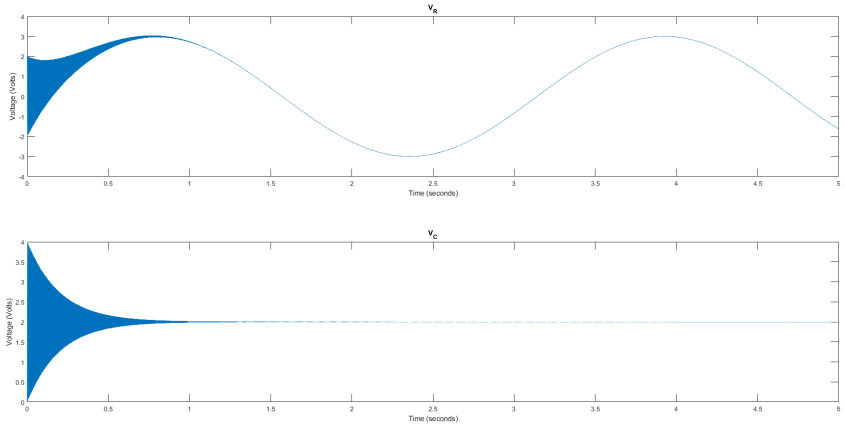
- Initialize the time vector t.
- Extract values for V_R and V_C via the $v.p$ file and initialize the voltages of the u_1 and u_2 sources.
- Initialization of the filter $\Lambda(s)$ and calculation of the z matrices.
- Calculate the circuit parameters and calculate $\hat{V_R}$ and $\hat{V_C}$ using $lsim$ and the transfer table we calculated above.
- Print the plots for presentation in the report.
- Finally, a script was developed which, with iterative loops, searches for the optimal model.

Before proceeding to the presentation of the results and their commentary, there are a few more things that should be noted.

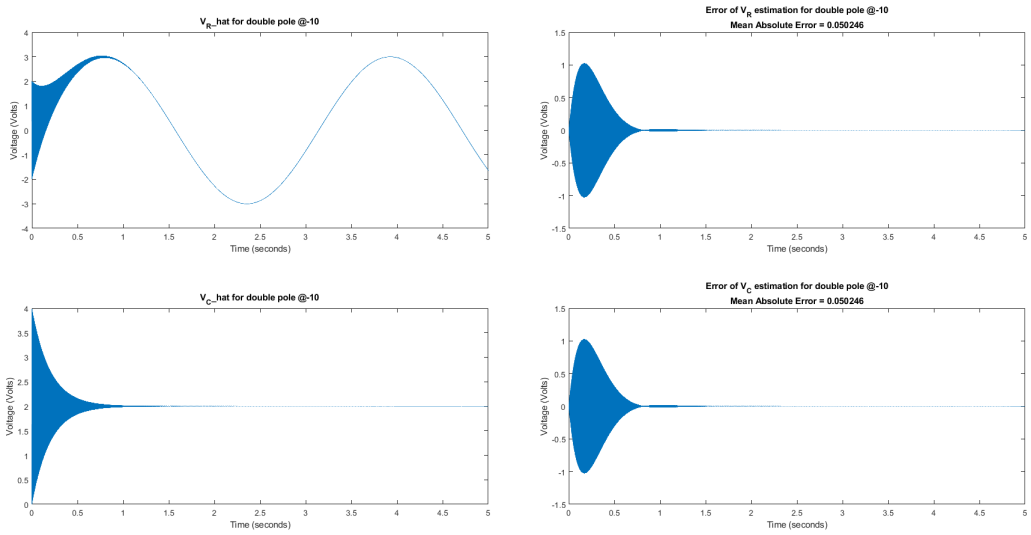
1. Strange results were observed regarding the θ vectors estimated with the L.S.M. Specifically, elements 3 & 7 of vector θ_R (coefficients $\ddot{u}_1$ and $\dot{u}_2$) and element 5 of vector θ_C (coefficient $\dot{u}_2$) showed strange behavior in the tests and had a large deviation from their expected values. For this reason they were ignored for the calculation of $\hat{V_R}$ and $\hat{V_C}$.
2. Continuing with 1., after a sufficient number of tests I came to the conclusion that the best estimates of the parameters $\frac{1}{RC}$ and $\frac{1}{LC}$ are in the first 2 elements of the vectors θ_R and θ_C (that is, the coefficients of $\dot{V}_R, V_R$ and $\dot{V}_C, V_C$ respectively), so only the average of these elements was used to approximate the quantities.
3. Additionally, after testing and some MATLAB warnings I ended up using $time\ step = 10^{-6}$, while the $tspan$ used was $[0, 5]$, to monitor the behavior of the estimate for a sufficiently long time after the circuit transient.
4. Finally, only filters with double poles have been used in the tests, while the same filter is used to estimate both θ vectors.

2.2.2 Results & Remarks

As in Topic 1, I start my tests by placing a dipole at -10 for the $\Lambda(s)$ filter. Below are shown first the actual values of V_R and V_C taken from the $v.p$ file, followed by the predictions and corresponding errors for the double pole at -10.



V_R has some fluctuations during the transient and then follows a sinusoidal form and V_C makes a damped oscillation and finally stabilizes at $2V$ as expected for both voltages.

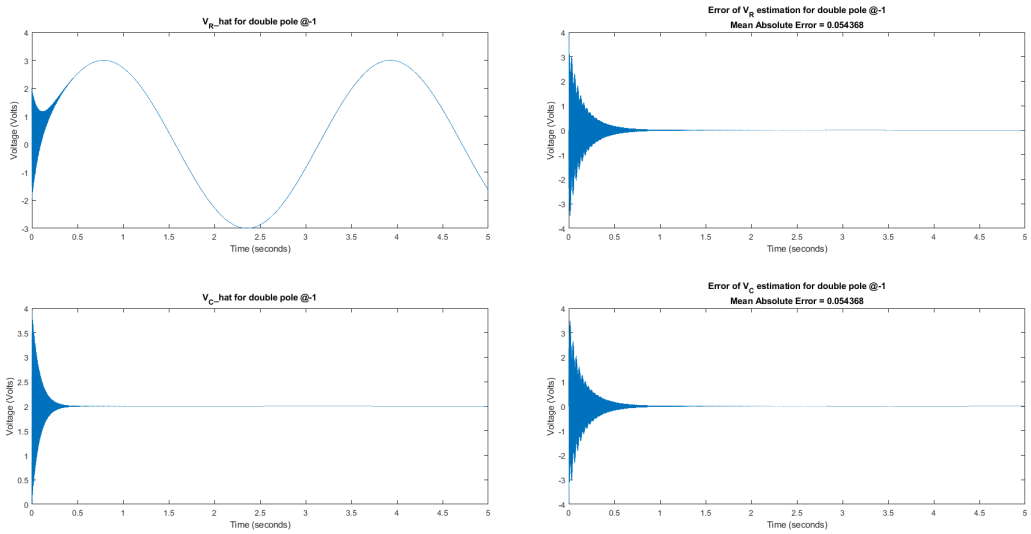


The signals $\hat{V_R}$ and $\hat{V_C}$ have the same form as the measured signals, but as

can be seen from the error graphs they follow with a delay, while in the steady state the error is zero. What is particularly interesting and for which I could not find any explanation is that the error is exactly the same for both signals (the average absolute error is 0.0502 in both cases).

My next step, again, is to try $\Lambda(s)$ filters with -1 and -100 poles.

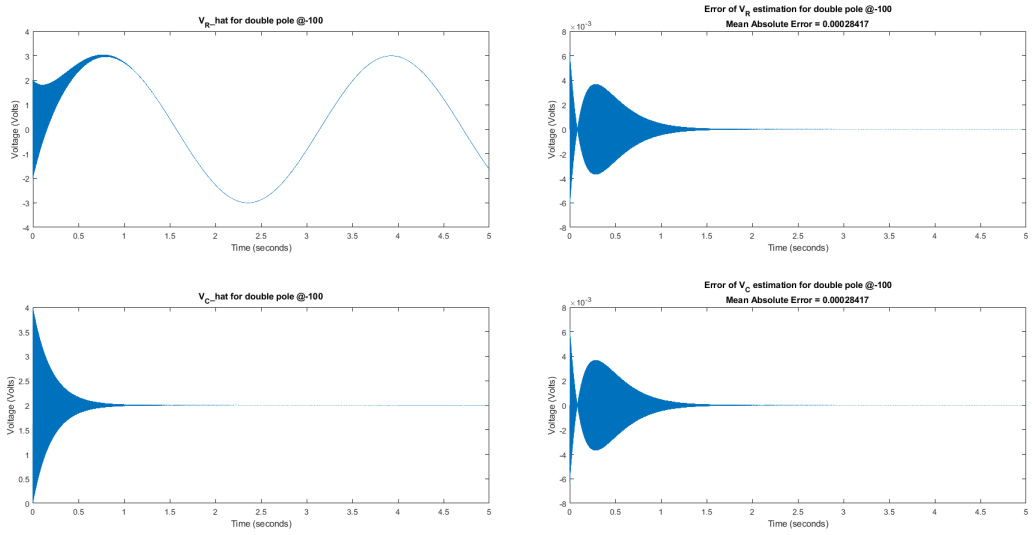
The results for double pole at -1:



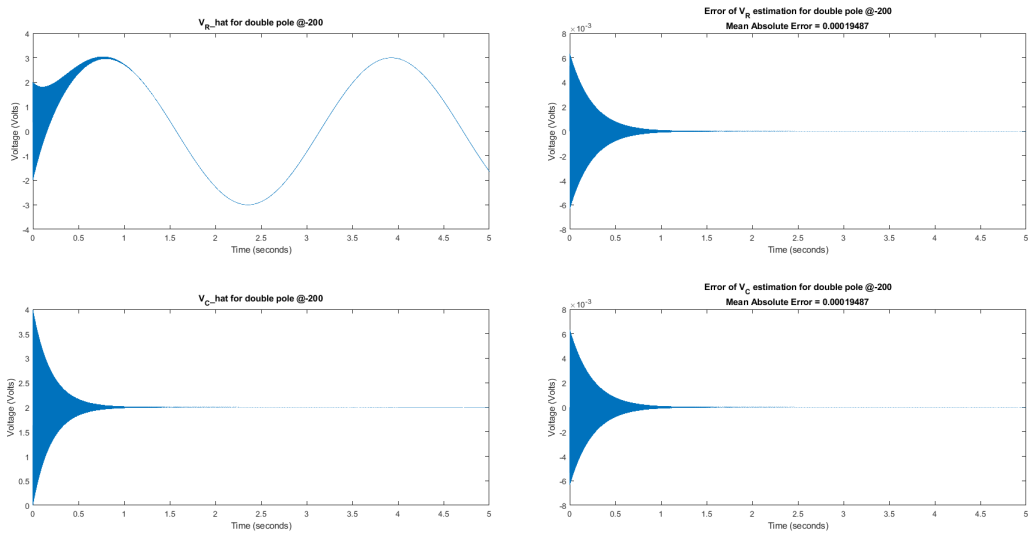
I notice that the error in the transient stage is now quite increased which is also reflected in the average absolute error which is now 0.5437, a relatively small increase as the longest period of the test is in the steady state, where this model also makes a very good estimate .

The results for double pole at -100:

In this case the error is very significantly reduced, since the maximum value it takes is of the order of $10^{-}3$, while earlier it took values close to unity. The reduction in error is so dramatic that it also has a large effect on the mean absolute error which is now 0.0003.

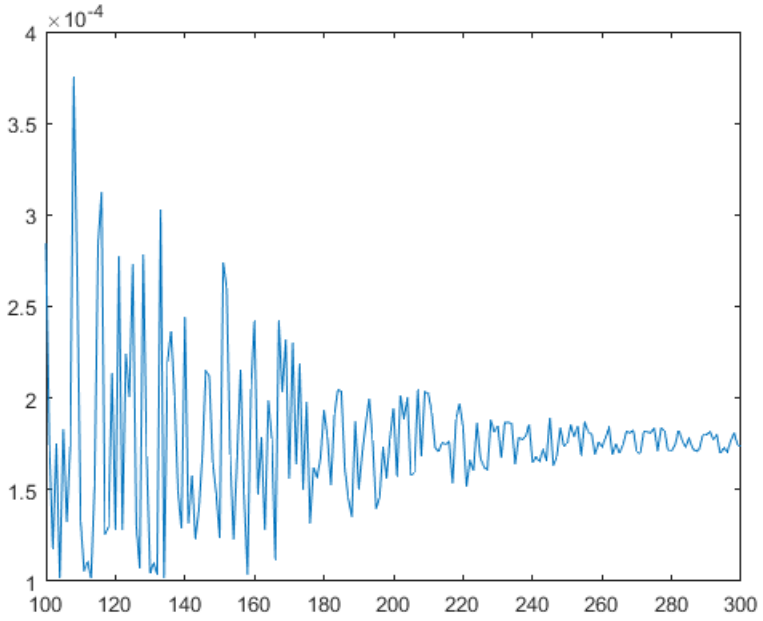


So I understand that in this particular problem I am looking for larger pole values, in contrast to the Topic 1 problem, where the optimal approximation was in the [0,1] interval. In an attempt to narrow down the interval in which to search for optimal pole values with an iterative method, I do one more test with a double pole at -200:



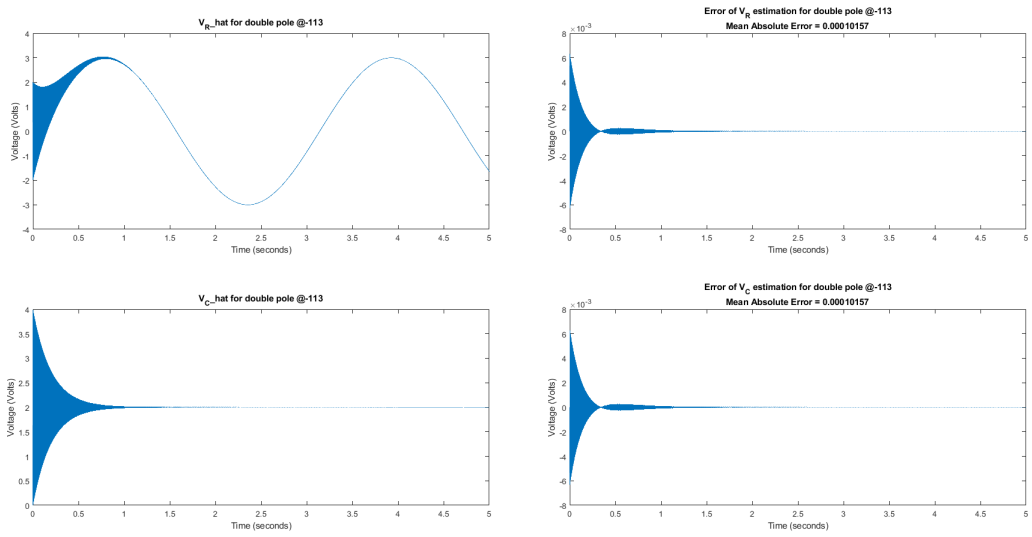
The error is even smaller in this case (0.0002) and prompts me to investigate

in the interval [100,300]. The diagram below shows the average absolute error as a function of the double pole.

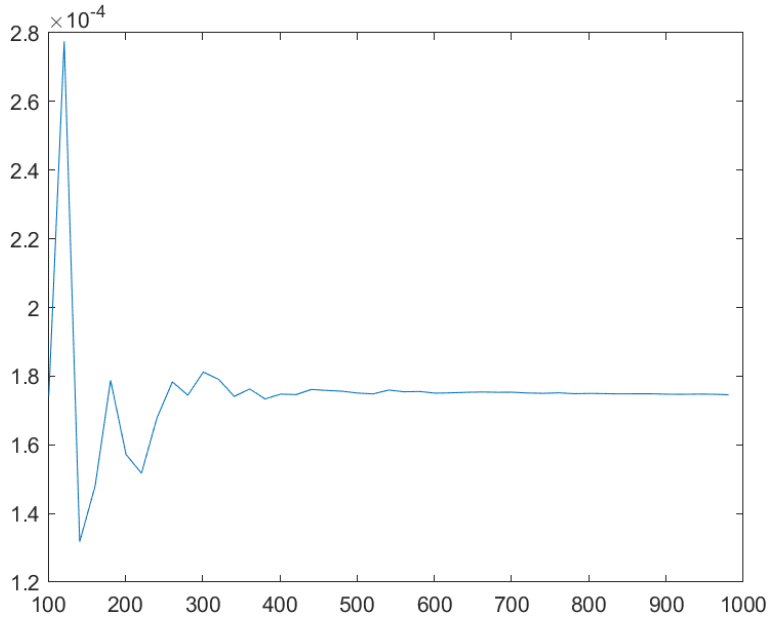


Unlike the error in the 1st problem, where a global minimum appeared, here the behavior of the error is strange with too many fluctuations, especially in the interval [100,200]. Perhaps this is due to the simplifications that have been made, Nevertheless we conclude that the smallest error returns the double pole at -113. Estimate and error plots for this model are shown below.

The error in this particular case shows a similar form to the previous models but approaches 0 noticeably faster, which is also reflected in the value of the average absolute error which has dropped to 0.0001.



In an attempt to understand the behavior of the error as a function of the poles of the $\Lambda(s)$ filter, I do one last test with pole values in the interval [1,1000], but with a smaller step this time as I'm not looking for a minimum.



Values for poles 0 to 100 have been left off the graph as they were much larger. We notice that the error has some local minima (with a possible total

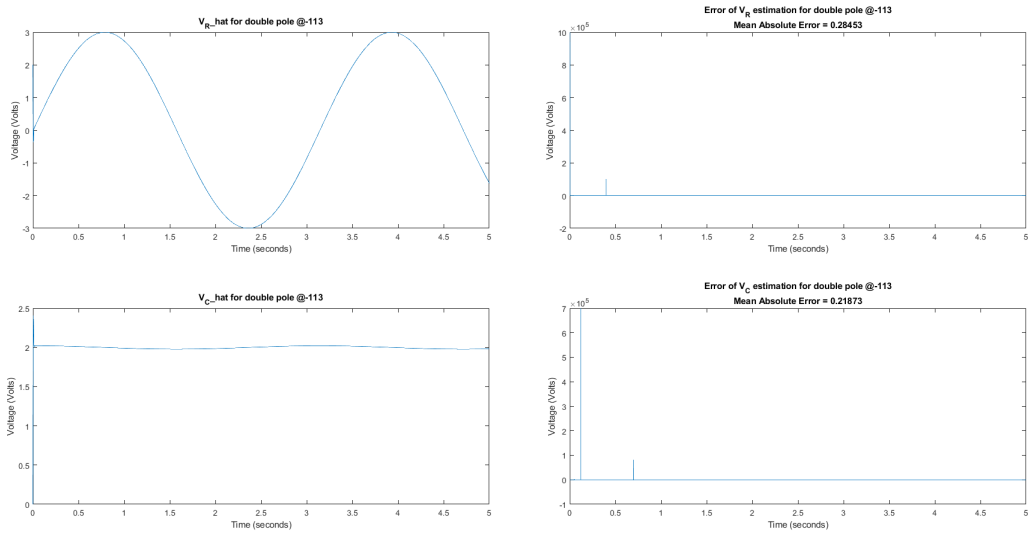
maximum at 113), while as it increases it stabilizes and takes a value of approximately $1.8 \cdot 10^{-4}$.

Estimates with measurement noise

At this point we reach the requirement b of Topic 2, where we have to add 3 noise points to the signals V_R and V_C and then estimate again with the L.S.M. the circuit parameters, observing what effects there are.

The 6 values added range in the interval $[10^3, 10^6]$, and have been inserted at random, but fixed points.

For the purposes of this test we are using the best filter we identified in the previous piece of project, namely the one with a double pole at -113.



Our model again manages to approximate the real signals in the steady state quite satisfactorily, but we notice that it is no longer able to follow the continuously changing shape of V_R and V_C in the transient stage. Finally, the largest errors appear to appear where noise has been introduced, indicating that these outliers have largely been "missed".