Pázmány Péter Katolikus Egyetem
Információs Technológiai és Bionikai Kar

Parameter Estimation

Exam Project Documentation

Dominik Csatári, FV1TW4
Software Engineer Msc

2024

Contents

1 Model **1**

1.1 The Michaelis-Menten Equation 1
1.2 Simulation 1
1.3 Computation Methods 3
1.3.1 Least Square Estimator 3
1.3.2 Maximum Likelihood Estimator 5
1.4 Summary 6

Chapter 1

Model

1.1. The Michaelis-Menten Equation

For this exam project I chose the 12th model, which was **Michaelis-Menten Equation**, which describes enzymatic reactions. The chemical reaction it describes is shown in Equation 1.1. The differential equations describing this model can be seen in the Equations (1.2) - (1.5).

$$\mathrm{E} + \mathrm{S} \underset{\text{k-1}}{\overset{\text{k1}}{\rightleftharpoons}} \mathrm{C} \xrightarrow{\text{k2}} \mathrm{P} + \mathrm{E} \tag{1.1}$$

$$\frac{dS(t)}{dt} = -k_1 E(t)S(t) + k_{-1}C(t) \tag{1.2}$$

$$\frac{dE(t)}{dt} = -k_1 E(t)S(t) + (k_{-1} + k_2)C(t) \tag{1.3}$$

$$\frac{dC(t)}{dt} = k_1 E(t)S(t) - (k_{-1} + k_2)C(t) \tag{1.4}$$

$$\frac{dP(t)}{dt} = k_2 C(t) \tag{1.5}$$

The values S(t), E(t), C(t), P(t) are the concentrations of the substrate, enzyme, complex and product. For initial values I have chose the following values: $k_1 = 2$, $k_{-1} = 1$, $k_2 = 1.5$. Also I chose the temperature to be constant, so the parameters values will depend on the time.

1.2. Simulation

For simulating the data I have chosen an app called *SimBiology*. With this I was able to define initial conditions and with creating the reactions I was able to generate the data. The simulation had two reactions with mass action kinetics generated the following way:

```
reaction1 = addreaction(model, 'E + S <-> C');
reaction2 = addreaction(model, 'C -> P + E');
```

The generated compartment which shows the reactions and parameters can be seen in Figure 1.1.

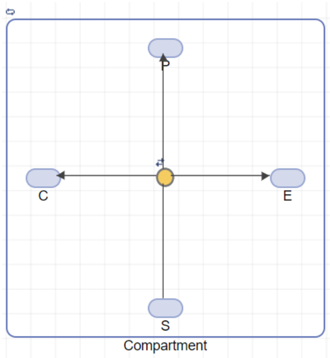

Figure 1.1: *SimBiology* generated model structure based on the equations introduced in Section 1.1

I have chosen the stop time to be 5, as for higher number the model reaches a steady-state. This behaviour can be seen in Figure 1.2, where I have generated the model. From this simulation I have saved the data of each concentration and the time which is an array of timesteps taken at each iteration. Note, that this array contains differing numbers as the simulation method uses an adaptive stepping mechanism.

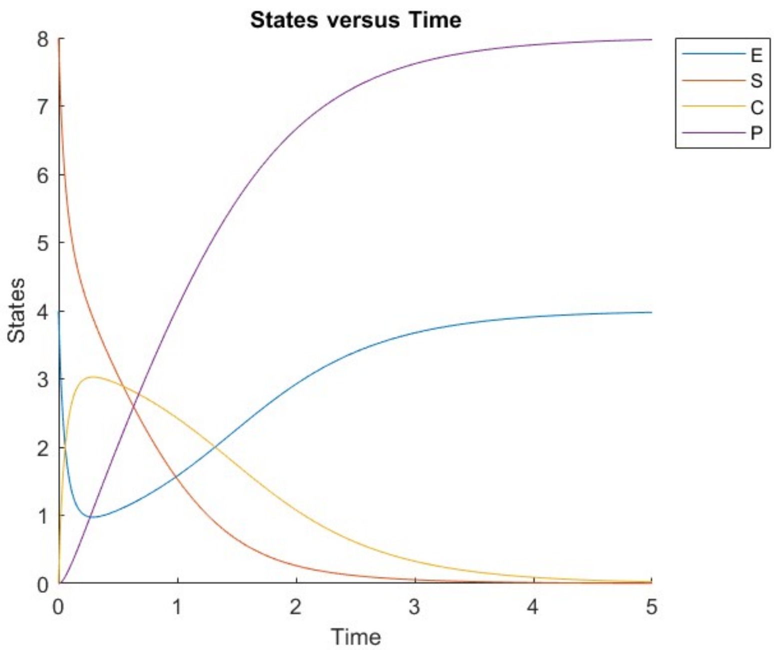

Figure 1.2: Simulated Model.

1.3. Computation Methods

For this task we had to choose two computation methods. For this I have chosen to implement a **Least Square Estimator** and a **Maximum Likelihood Estimator**.For both estimators, I have implemented the simulated data with noise to more closely represent measured data. For both calculations, I had to prepare the measured data. For implementing the concentrations I have used the Euler method, which lets me discretize the dynamical model. For each timestep `ts` I have calculated `ts = t(i)-t(i-1)`, where `i` represents the current instance.

1.3.1 Least Square Estimator

For Least Square Estimation, I prepared the data in a form that allows calculation of the parameters using pseudo-inverse as in Equation 1.6. I have predicted each parameter separately.

$$\hat{\theta}_{\mathrm{LS}} = (X^\top X)^{-1} X^\top Y \tag{1.6}$$

For k_2 I only had to use two arrays so it was a simpler task. For k_1 and k_{-1} I have used a combination of the equations mentioned in Equations (1.2) - (1.5). After this I have maintained the parameters, I have created a 100 simulations and averaged the results. I have done this for 5 noise levels each varying on `sigma` from the equation `sigma * randn(n,1)`. sigma values for LSQ were 0.2, 0.4, 0.6, 0.8 and 1.
After this, I have generated data based on the calculated parameters and plotted the results. I have also calculated squared losses for each noise level. The calculated parameters can be seen in Table 1.1 and the calculated squared differences in Table 1.2. I have also generated some figures showing the simulated data and the one calculated with estimated parameters. From this I decided to attach two. From Noise Level 1 and Noise Level 5 to compare. These can be seen in Figure 1.3 and in 1.4.

	Original Parameters	**Noise Level 1**	**Noise Level 2**	**Noise Level 3**	**Noise Level 4**	**Noise Level 5**
k1	**2**	**1.9472**	**1.9234**	**1.8617**	**1.8072**	**1.7375**
k-1	**1**	**0.76997**	**0.6955**	**0.53315**	**0.36386**	**0.25093**
k2	**1.5**	**1.482**	**1.4282**	**1.3518**	**1.2543**	**1.1473**

Table 1.1: Estimated parameters based on different Noise Levels

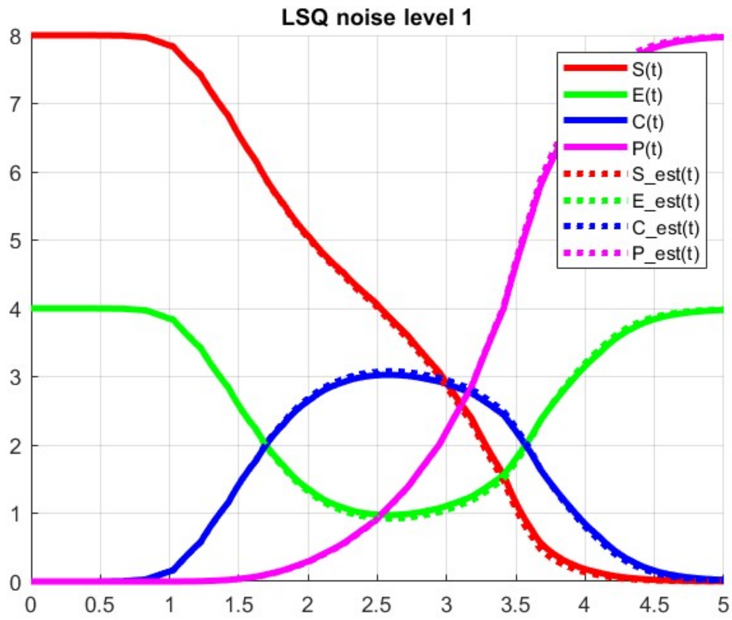

Figure 1.3: LSQ Estimation model at noise level 1 with the simulated and estimated data.

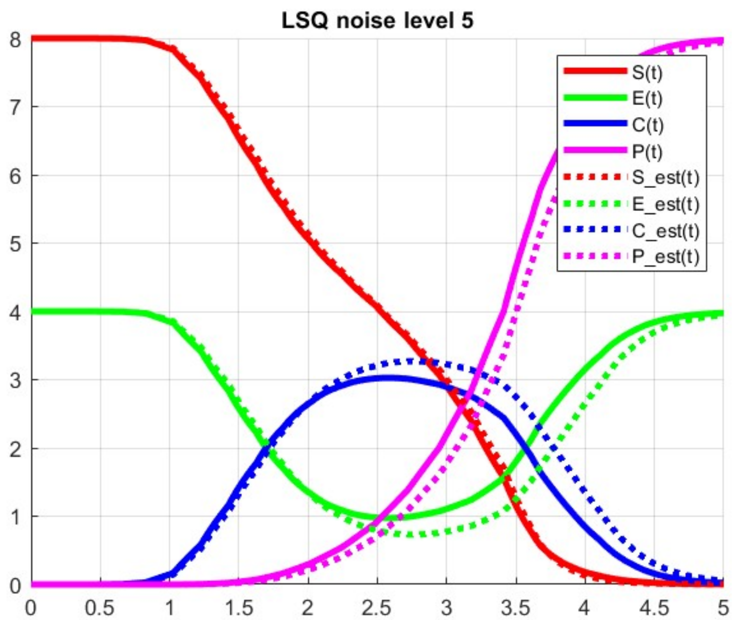

Figure 1.4: LSQ Estimation model at noise level 5 with the simulated and estimated data.

	Noise Level 1	Noise Level 2	Noise Level 3	Noise Level 4	Noise Level 5
E	**0.2433**	**0.1238**	**0.1032**	**0.1824**	**0.6643**
S	**0.1701**	**0.4492**	**1.7022**	**5.0202**	**11.1413**
C	**0.1701**	**0.4492**	**1.7022**	**5.0202**	**11.1413**
P	**0.1912**	**0.1532**	**1.2212**	**4.7391**	**13.5258**

Table 1.2: Squared differences between the original concentrations and the one's generated with estimated parameters based on different Noise Levels.

1.3.2 Maximum Likelihood Estimator

For Maximum Likelihood Estimation I chose to apply Likelihood estimation for different parameter values chosen from a grid with a predefined size. I have applied likelihood as shown in Equation 1.7, where *arg* is the right hand side of the equations introduced in Section 1.1. As the function is different for different parameters I have created two functions in my code called `log_likelihood()` and `log_likelihood_k2()`. The first calculates k_1 and k_{-1} the second one calculates k_2.

$$\ln f_y(\theta; y_N) = \sum_{k=1}^{N} \ln \frac{1}{\sigma\sqrt{2\pi}} + \ln \exp\left(-\left(\frac{y_k - arg}{2\sigma^2}\right)\right) \tag{1.7}$$

For the grid size from where I choose the most probable solutions, I have chosen `linspace(0,5,range)`, where range is 500. I have tested 3 noise levels, with sigma(squared, but this is the naming convention) values 0.1, 0.2 and 0.3. I have followed the same implementation model proposed in Subsection 1.3.1. The estimated parameters can be seen in Table 1.3 and the squared distances in Table 1.4. Most of the estimated parameters are similar so I only attached the one calculated for Noise Level 1. This can be seen in Figure 1.5

	Original Parameters	Noise Level 1	Noise Level 2	Noise Level 3
k1	**2**	**2.004**	**1.9739**	**1.9739**
k-1	**1**	**0.92184**	**0.84168**	**0.8517**
k2	**1.5**	**1.503**	**1.493**	**1.4529**

Table 1.3: Estimated parameters based on different Noise Levels.

	Noise Level 1	Noise Level 2	Noise Level 3
E	**0.1747**	**0.2050**	**0.0570**
S	**0.1351**	**0.1368**	**0.1799**
C	**0.1351**	**0.1368**	**0.1799**
P	**0.2283**	**0.2128**	**0.0664**

Table 1.4: Squared differences between the original concentrations and the one's generated with estimated parameters based on different Noise Levels.

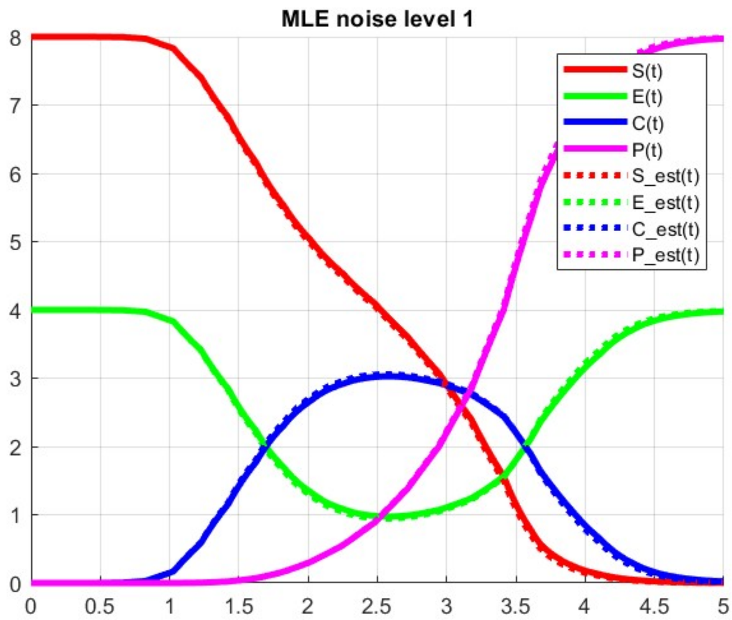

Figure 1.5: MLE Estimation model at noise level 1 with the simulated and estimated data.

1.4. Summary

In this project I have implemented two computation methods, LSQ and MLE. For LSQ I was able to try 5 different noise levels. Even for Noise Level 5, the model predicted the parameters and the concentrations quite well, according to the squared losses. MLE generally predicted better compared to LSQ based examples where they had the same noise level. However, for higher noise level, MLE failed to converge, that is why I only used 3 levels with smaller steps. It also requires a predefined grid for searching possible parameters. Since this grid is finite, we also need some intuition about the parameters beforehand. A great application for these methods could involve combining them. We could start with LSQ to gain some intuition about the parameters, and then refine our search for better parameters using MLE.