Project 2

Matthew Callahan, Jiajing Guan, and Aneesh Malhotra

October 22, 2018

1 Trajectory Matching with Markov Chain Monte Carlo

1.1 Method

Since the ODE function given could be easily solved, the problem became estimating C and β in $Ce^{-\beta t}$.
We first initialized the Markov Chain Monte Carlo with the following parameters:

Variable	Value
C_0	0
β_0	0
Data Standard Deviation σ	0.03
Guess Jump	0.01
Burn Time	100
Iteration Limit	5000

For each iteration step, we make a new guess for $[C_n, \beta_n] = [C_{n-1}, \beta_{n-1}] + D * \text{randn}$. Then we calculate the SSE using $[C_n, \beta_n]$ to evaluate the ratio $e^{\frac{-\text{SSE}_{n-1}+\text{SSE}_n}{2\sigma^2}}$. If a random number between 0 and 1 is smaller than the ratio, the new guess is accepted.

1.2 Result

Using the method described above, we obtained the estimated parameters on two sets of data given.

- Smooth Data With the smoother version of the data, we obtained the following result:

C	3.1792
β	5.5447

We can also observe the convergence in the following figure:

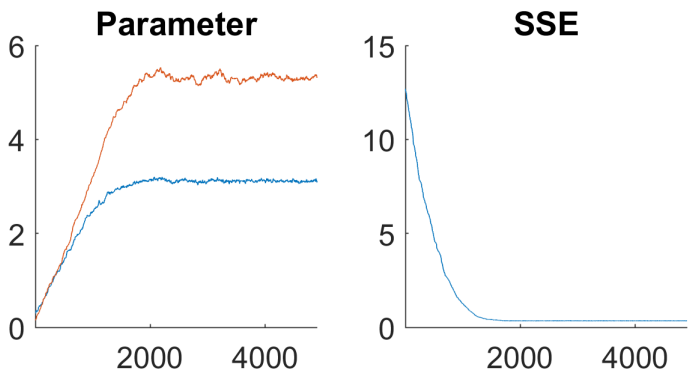

Figure 1: The figure on the left shows the convergence of the parameters as iteration step increases. The figure on the right shows the convergence of SSE.

We can also see how our model compares with the data:

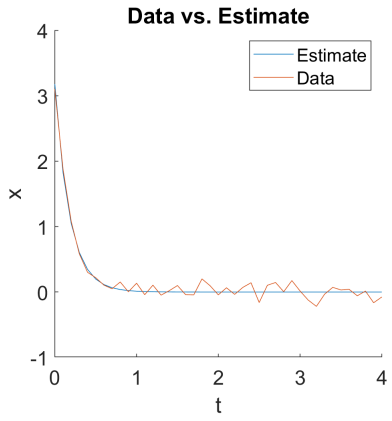

Figure 2: The model we obtained is the line in blue. We can see that our model captures the trend of the data.

- Noisy Data

  With the noisy version of the data, we obtained the following result:

C	2.6331
β	4.8364

We can also observe the convergence in the following figure:

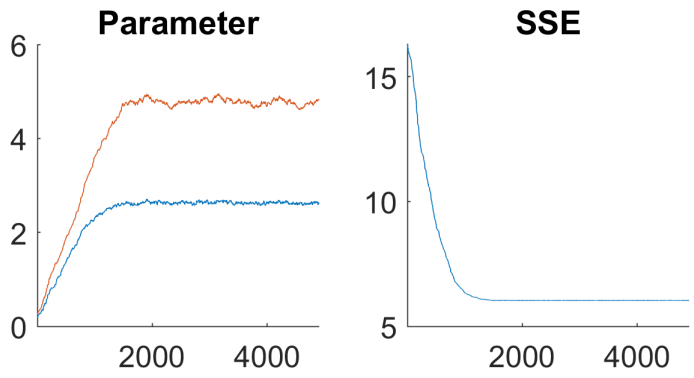

Figure 3: The figure on the left shows the convergence of the parameters as iteration step increases. The figure on the right shows the convergence of SSE.

We can also see how our model compares with the data:

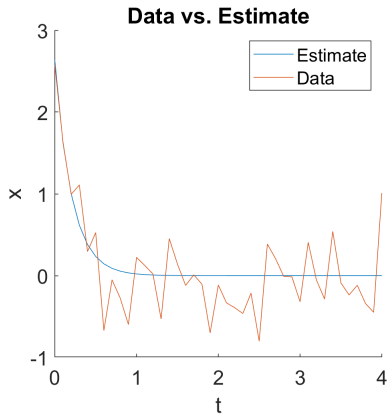

Figure 4: The model we obtained is the line in blue. We can see that our model captures the trend of the data.

1.3 Explore

We explored with the guess jump a little bit and obtained the following result:

Guess Jump	C	β	Convergence Step	Converged SSE
0.005	3.16566068356473	5.46532123092112	4326	0.346611467385126
0.01	3.12005507560874	5.35257792035497	2073	0.344644620650210
0.02	3.09761701625579	5.27931484301583	755	0.345845653405011
0.1	3.17838628697765	5.50513291919420	99	0.348208384804511

We can see that the different guess jump did not affect the results too much but did delay convergence when it is small.

2 Gradient Matching with Smoothing

2.1 Method

We can smooth the data using collocation. The basis ϕ could be created using *spcol* function in matlab. Then the coefficients $\mathbf{c} = (\phi^T\phi)^{-1}(\phi y)$, where y is the data given.

The gradient matching method is essentially using Gauss-Newton method to minimize the integrated SSE:

$$\text{ISSE} = \sum_{i=1}^{n} W_i[Dx(\hat{t}_i) - f(x(\hat{t}_i), \theta)]^2$$

where $\hat{x(t_i)} = \phi\mathbf{c}$ is the smoothed data using collocation.
For this problem, the ISSE could be simplied to

$$\text{ISSE} = \sum_{i=1}^{n} W_i[D\phi(t_i)\mathbf{c} + \beta\phi(t_i)\mathbf{c}]^2$$

We set $W_i = 1$ for all $i = 1, \ldots, n$. Then the Jacobian becomes $\hat{x}$. For each iteration step, we perform the following:

$$\begin{aligned} H &= J^T J \\ g &= J^T(D\hat{x} + \beta\hat{x}) \\ \beta_n &= \beta_{n-1} - H^{-1}g \end{aligned} \tag{1}$$

Due to the fact that ISSE does not take $x(t_0)$ into account, we use the initial value of the smoothed data as $x(t_0)$ for the following parts.

2.2 Result

- Smooth Data The estimated $\beta = 4.962410410929266$.
  We can also observe the convergence in the following figure:

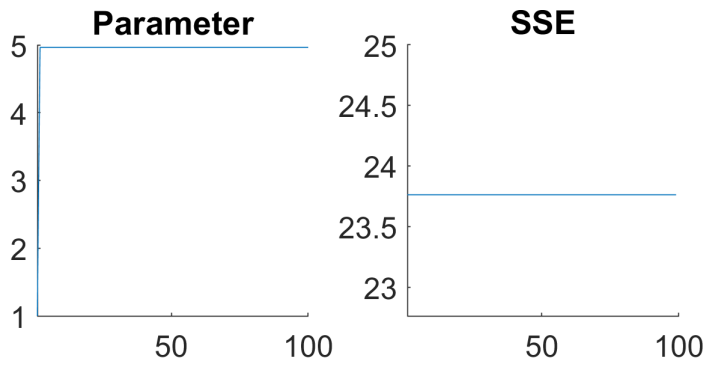

Figure 5: The figure on the left shows the convergence of the parameters as iteration step increases. The figure on the right shows the convergence of SSE.

We can also see how our model compares with the data:

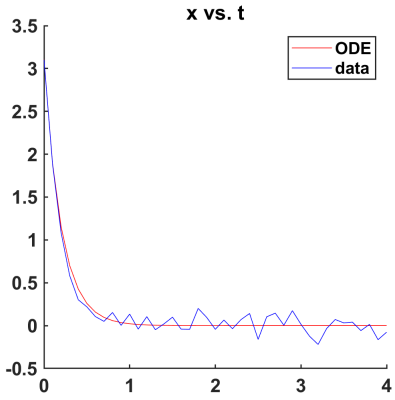

Figure 6: The model we obtained is the line in blue. We can see that our model captures the trend of the data.

- Noisy Data The estimated $\beta = 2.160787567688269$.
  We can also observe the convergence in the following figure:

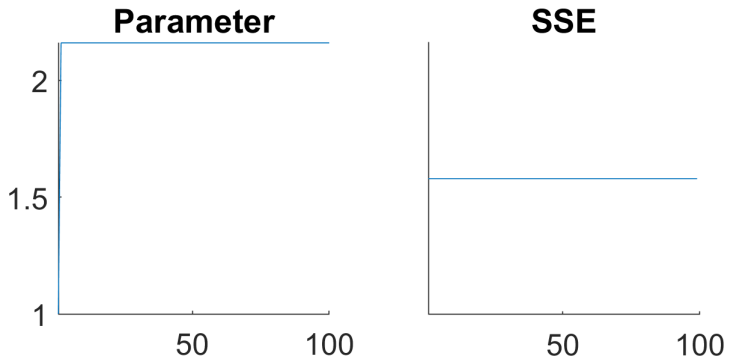

Figure 7: The figure on the left shows the convergence of the parameters as iteration step increases. The figure on the right shows the convergence of SSE.

We can also see how our model compares with the data:

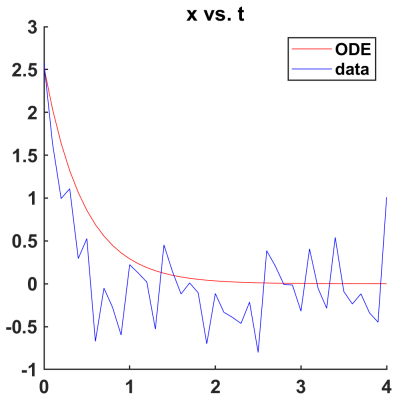

Figure 8: The model we obtained is the line in blue. We can see that our model captures the trend of the data.

2.3 Explore

- Without Smoothing
  When the method is used without smoothing, it fails catastrophically. The parameter blows up.
- Different Number of Basis
  When we smooth the data with collocations, we try different number of polynomials allowed for the basis and obtained the following result:

Number of Polynomials	β
9	4.97277486579170
10	4.94682728787471
11	4.96241041092927
12	4.96995381483431
13	4.97336941885107

Therefore, the number of polynomials in the collocation basis does not affect the results drastically.

3 Integral Matching with Smoothing

3.1 Method

We can solve this system analytically, which removes the need for numerical integration. Our goal is then to find parameters B, C that minimize the sum of squared errors given by

$$\text{ISSE}(B, C) = \sum_{i=1}^{N} (Ce^{-Bt} - y_i)^2$$

We achieved this using a Gauss-Newton Iteration using both the smoothed data and the original data

3.2 Results

The first thing we did was observe the fitting of the solution to the data in both the case of smoothed and unsmoothed data.

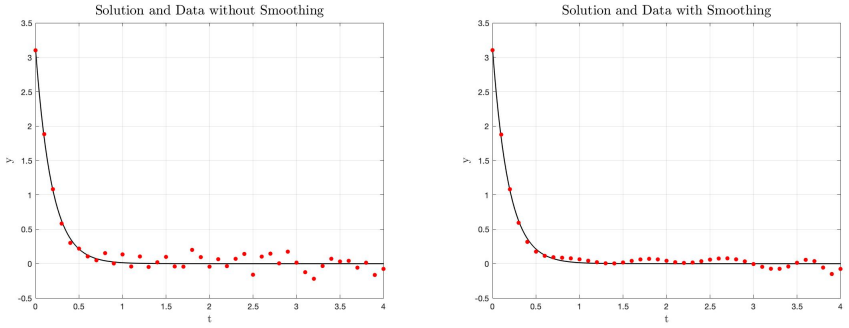

Figure 9: Integral matching fits the smoothed data in both cases

In the case of data that was not smoothed, we found the following parameter convergence

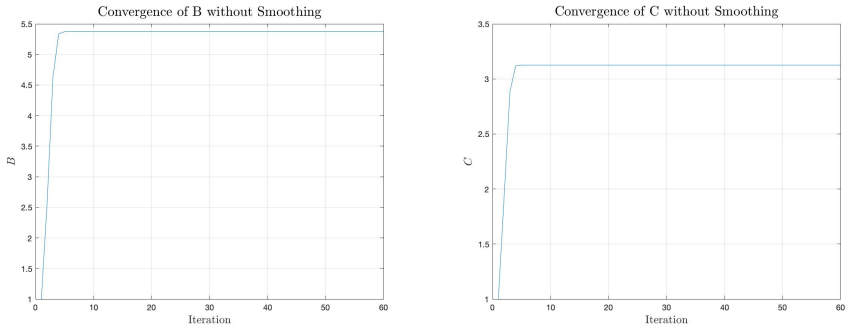

Figure 10: Convergence of Parameters without Smoothing. Both paramters converge in ≈ 6 iterations

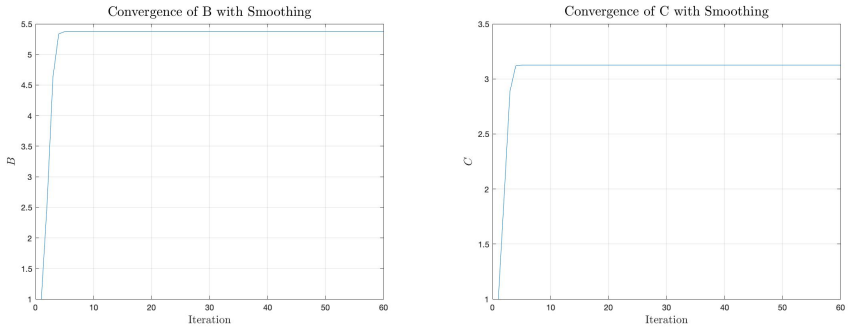

Figure 11: Convergence of Parameters with Smoothing. Both parameters converge in ≈ 6 iterations

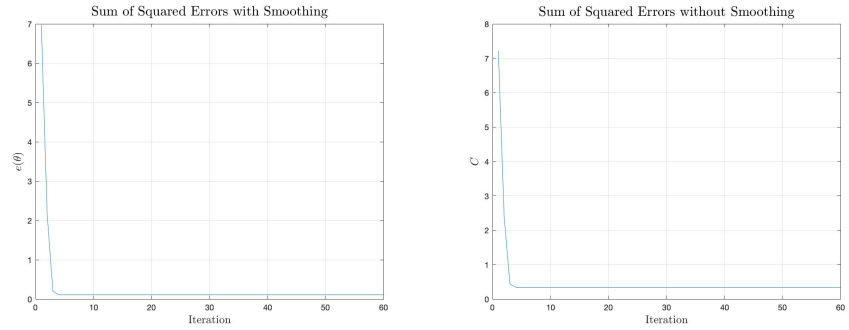

Figure 12: SSE Error Convergence

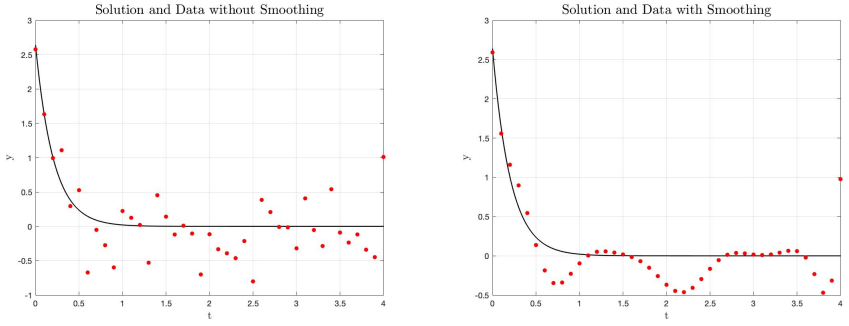

Figure 13: Integral Matching with Noisy Data

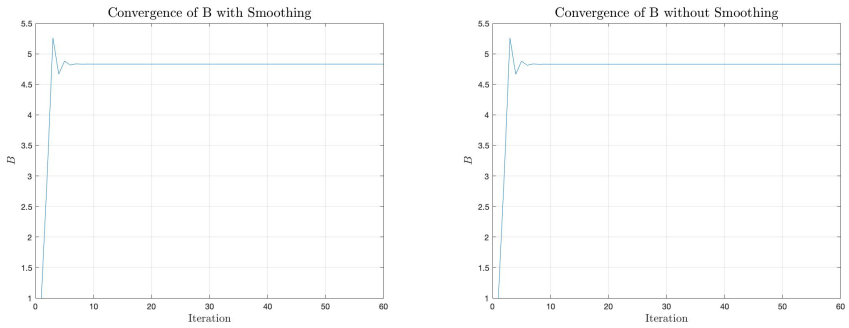

Figure 14: Convergence of the parameter B with noisy data

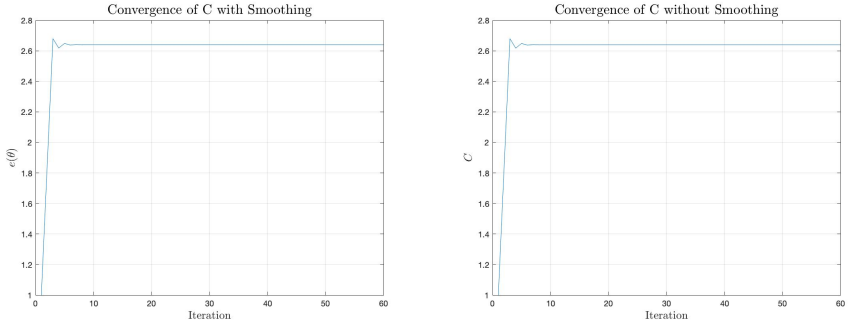

Figure 15: Convergence of the parameter C with noisy data

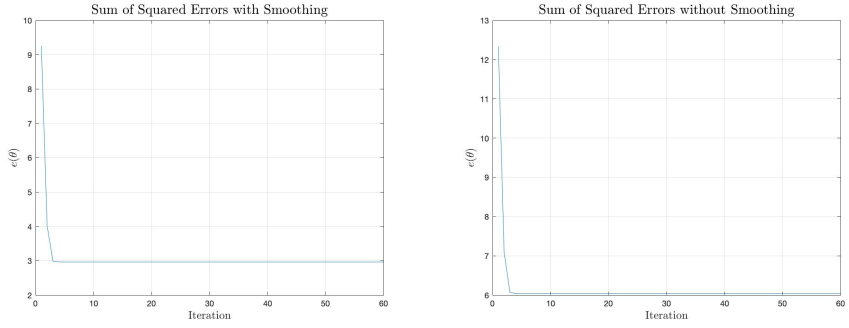

Figure 16: Convergence of the parameter C with noisy data

3.3 Integral Matching with Noisy Data

4 Parameter Cascading with Smoothing

4.1 Method

Parameter cascading aims to approximate the same values for C and β as the previous two problems but goes about it in a way that combines smoothing and matching. The steps are as follows:

- First we approximate the solution, $\hat{x}(t) = \sum_k^N c_k \phi_k$, using a B-spline similar to the other methods
- Next we use an inner optimization function given by

$$J(c|\theta, \rho) = (1-\rho)\sum_j^n [y_j - x(t_j)]^2/n + \rho \int_0^T [D^m x(t) - f[x(t)]]^2 dt/T$$

  to further refine $\hat{x}(t)$. The left hand term of this equation corresponds solely to data-fitting and the right corresponds to matching the differential. Therefore if ρ is set to 0 we are only using our data, and if ρ is 1 then we are only fitting to our differential.
- The last step is to use the outer optimization function to find a θ that most closely models our smoothed $\hat{x}(t)$. This is just done via SSE, but can be done with other methods.
- Steps 2 and 3 are then repeated until the best approximation is found.

4.2 Results

For our test we used the initial data

c	[1,...,1]
C	1
β	1

When ρ was chosen to be .9 the results were found to be the following. Note that C is taken to be the first value of $\hat{x}$.

ρ	.9
β	5.33
C	3.08

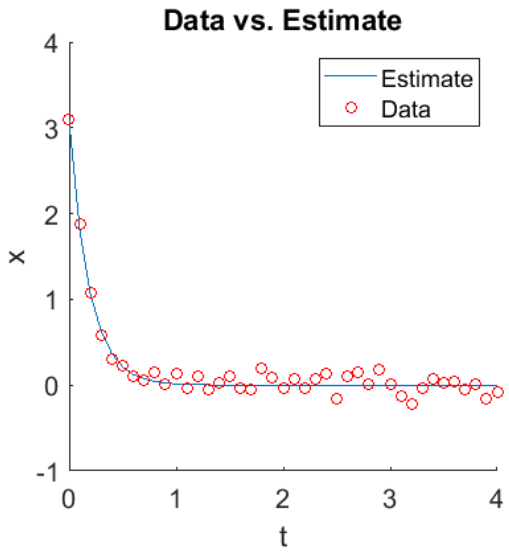

Figure 17: Cascading Parameter Estimation

The convergence of β can be seen below. It takes 14 steps to get the SSE within the set tolerance of 10^{-10}.

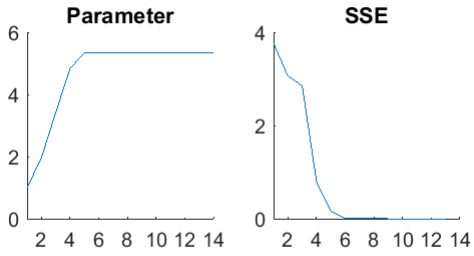

Figure 18: Cascading Parameter Convergence

4.3 Explore

The first thing to experiment with in Parameter Cascading is ρ values. Changing the ρ value had very little effect on the estimation of our parameters, changing the hundreds place by 1 or 2. As seen in the graph below, a wide range of ρ values all overlap with each other.

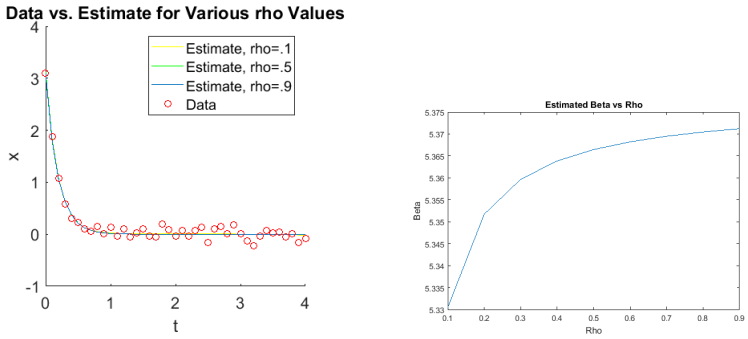

Figure 19: Cascading Parameter Convergence

The graphs only began to differ when the number of approximating polynomials was significantly reduced. Because the system being approximated was so simple there isn't a lot of room for variance in the solutions.

5 Comparison

First, we think that the gradient descent method is most helpful when the initial value does not to be approximated. Otherwise, it limits the accuracy of the model.

Second, we think that the MCMC and integral matching method are powerful when the ODE equations given can be easily solved. The MCMC method is easier than the integral matching method because we don't need to take the Jacobian matrix.

Profile cascading method required the most time coding in all of these methods, but the method is overall more powerful than others. It converges fast and provides accurate estimation for the parameters.