AMATH 563: APPLYING KERNEL RIDGE REGRESSION TO THE LOTKA-VOLTERRA MODEL

WINNIE LAU

Department of Applied Mathematics, University of Washington, Seattle, WA
wwlau@uw.edu

1. Introduction

In this report, we will apply Kernel Ridge Regression with the Radial Basis Function (RBF) kernel to observations from the Lotka-Volterra (LV) Model in order to find the best fit functions for the populations of prey and predator over time. The parameters for the model are determined via cross validation. The derivative is estimated from our best fit functions, and the absolute error is plotted for both the estimated derivative and our best fit function compared to the true LV derivative and population values. Using these values, we will apply the RBF kernel and a second degree polynomial kernel to find the optimal model for our differential equations of the LV model. The error will be plotted between the model and true LV model on a 2D contour plot, and three additional initial conditions will be applied to find how well our new model performs. We observed that the model from the RBF kernel was unable to fully represent the LV model as it gradually loses its oscillations over time. The polynomial kernel retained its oscillations over time, but given new initial conditions, it was unable to follow the true LV model closely, likely due to overfitting on the training data.

2. Methods

Consider the Lotka-Volterra (LV) predator prey model defined as

$$
\begin{aligned}
\frac{dp_1}{dt} &= \alpha p_1 - \beta p_1 p_2 \\
\frac{dp_2}{dt} &= -\gamma p_2 + \delta p_1 p_2
\end{aligned}
\tag{1}
$$

We define p_1 and p_2 as the populations of prey and predators, respectively, while α, β, γ, δ are positive constants, with the first two representing growth rates of the prey and predator populations, and the other two representing the interaction rates between the two populations. The system was defined with $(\alpha, \beta, \gamma, \delta) = (1, 0.1, 0.075, 1.5)$ and an initial condition of $p_0 = (10, 5)$ over the time period $[0, 20]$. Without knowledge of Eq. 1, we consider a system of the form

$$
\frac{dp_i}{dt} = f_i(p_1, p_2) \quad i = 1, 2 \tag{2}
$$

where f_i are unknown functions. Our goal is to learn the $f_i' s$ from a set of observations of the system given by

$$
y(t_n) = (p_1(t_n), p_2(t_n)) \in \mathbb{R}^2, \quad n = 0, \dots, 49, \quad t_n = 0.4\, n.
$$

Date: May 13, 2025.

2.1. **Step 1.** Given some data $Y = (y(t_n))_{n=0}^{49}$, we will use the RBF kernel defined as

$$K(t,t') = \exp(-\gamma_r \|t-t'\|^2)$$
$$\hat{p}_i(t) = \sum_{j=0}^{n} \alpha_j K(t,t')$$

to find the best model approximation $\hat{p}_i$ with optimal coefficients α_i. We will use cross-validation to find the optimal length scale γ_r and regularization parameter λ for the best fit. Our parameter search space was defined as $\lambda \in (1e-6, 1e2)$ and $\gamma_r \in (1e-2, 1e2)$.

The true derivative was calculated with finite differences over a small step size of 0.001. The error is calculated as $|\hat{p}_i - p_i|$ and plotted over the time scale $[0, 20]$.

2.2. **Step 2.** Now, we consider a denser uniform time grid over $[0, 20]$ of 100 points defined as $\tilde{t}_1, \ldots, \tilde{t}_{100}$. We compute the derivative of the function $\hat{p}_i$ using the representer formula for KRR over the dense grid as

$$\hat{p}_i(t) = \sum_{i=0}^{n} \alpha_i K(t,t') = \sum_{i=0}^{n} \alpha_i \exp(-\gamma_r \|t-t'\|^2)$$
$$\frac{d}{dt}\hat{p}_i = \sum_{i=0}^{n} \alpha_i \exp(-\gamma_r \|t-t'\|^2) * (-2\gamma_r(t-t'))$$

The true derivative $\frac{d}{dt}p_i$ was calculated with finite differences using `np.gradient`. Our error was calculated as $|\frac{d}{dt}\hat{p}_i - \frac{d}{dt}p_i|$ and plotted over $\tilde{t}_n$.

2.3. **Step 3.** From Step 1 and 2, we derived approximations for our population functions with $\hat{p}_i(\tilde{t}_n)$ and their derivatives $\frac{d}{dt}\hat{p}_i(\tilde{t}_n)$. Now, we want to solve the optimization problem

$$\hat{f}_i = \operatorname*{arg\,min}_{h_i \in \mathcal{H}_i} \sum_{n=1}^{100} \left(\frac{d}{dt}\hat{p}_i(\tilde{t}_n) - h_i\big(\hat{p}_1(\tilde{t}_n), \hat{p}_2(\tilde{t}_n)\big) \right)^2 + \lambda_i \|h_i\|_{\mathcal{H}_i}^2,$$

where $\mathcal{H}_i$ are the RKHSs associated with the kernels H_i. We considered the BRF and second order polynomial kernels to compute $\hat{f}_i$, using the same cross validation approach as in Step 1 for the best length scale and regularization parameter. The error $|f_i - \hat{f}_i|$ is shown in a 2D contour plot over the domain $[0, 600] \times [0, 60]$. The functions $\hat{f}_i$ were then used to simulate trajectories given new initial conditions $p_0 = \{(15, 5), (20, 5), (10, 10)\}$ and compared with the true LV trajectories.

3. Results

3.1. **Step 1 and Step 2.** In Step 1, cross validation yielded the parameters $\lambda = 1.2966e-06$, $\gamma_r = 0.773922$ for $\hat{p}_1$, and $\lambda = 0.000111$, $\gamma_r = 0.596375$ for $\hat{p}_2$. The derivatives using these functions were then computed on the dense grid. The error for both approximations is presented in Figure 1. We observe oscillations in the error, which increases and decreases multiple times. For example, the intervals $(2.5, 5)$, $(7.5, 10)$ and $(12.5, 15)$ contain an increase in error compared to the other intervals, before leading to a large error on the end.

3.2. **Step 3.** Errors between the true solution and fitted solution from the RBF Kernels and polynomial kernel was plotted on a contour plot (Figure 2). For the RBF Kernel, we see the error increases as both p_1 and p_2 are increasing for both the fitted models for $\hat{f}_1$ and $\hat{f}_2$. Meanwhile, the polynomial kernel has increasing error when p_1 increased, with the highest error occurring around $p_1 = 600$, regardless of p_2.

The predicted trajectories for the fitted models were then plotted given different initial conditions, along with the true trajectory from the LV model (Figure 3). From this, we see the RBF Kernel

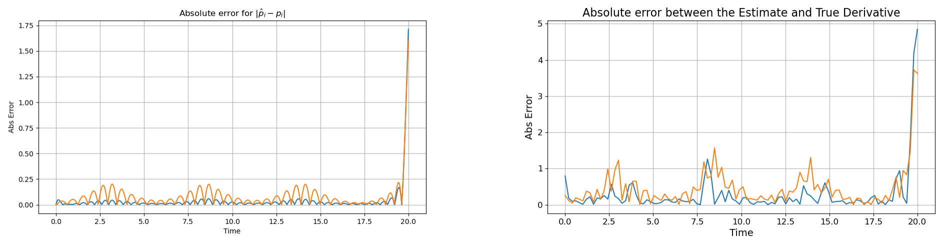

FIGURE 1. Plot of the absolute error between the estimated solution and true solution for the population and its derivative, given by $|\hat{p}_i - p_i|$ (left) and $|\frac{d}{dt}\hat{p}_i - \frac{d}{dt}p_i|$ (right).

was generally able to follow the true trajectory by increasing to one peak until between $(2.5, 5)$ where it then flattens out and loses its oscillations.

Meanwhile, the polynomial kernel was able to closely match the original initial conditions and maintain its oscillations and general shape. However, with new initial conditions, it was not able to fit the true LV as closely, especially when the new initial conditions moved further away from the original training data. This may be due to overfitting of the model to the original data set, causing the learned model to look closer to its trained data, even with new conditions.

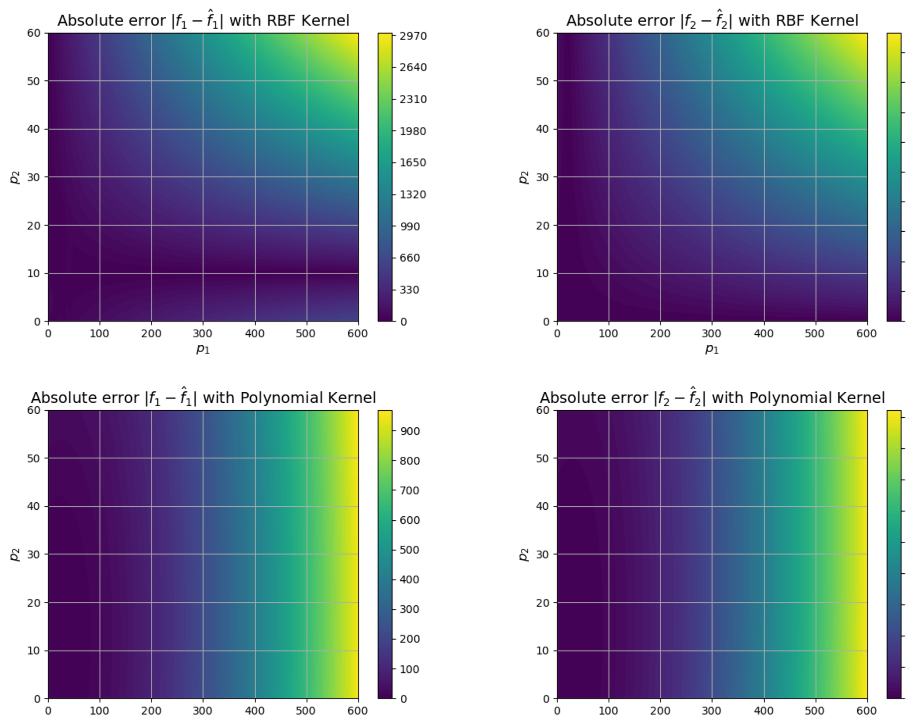

FIGURE 2. Contour plots over the domain $[0, 600] \times [0, 60]$ of the absolute errors $|f_i - \hat{f}_i|$, with $\hat{f}_i$ generated from a RBF kernel and a second degree polynomial kernel. The value for the error is indicated by the color bar next to each plot.

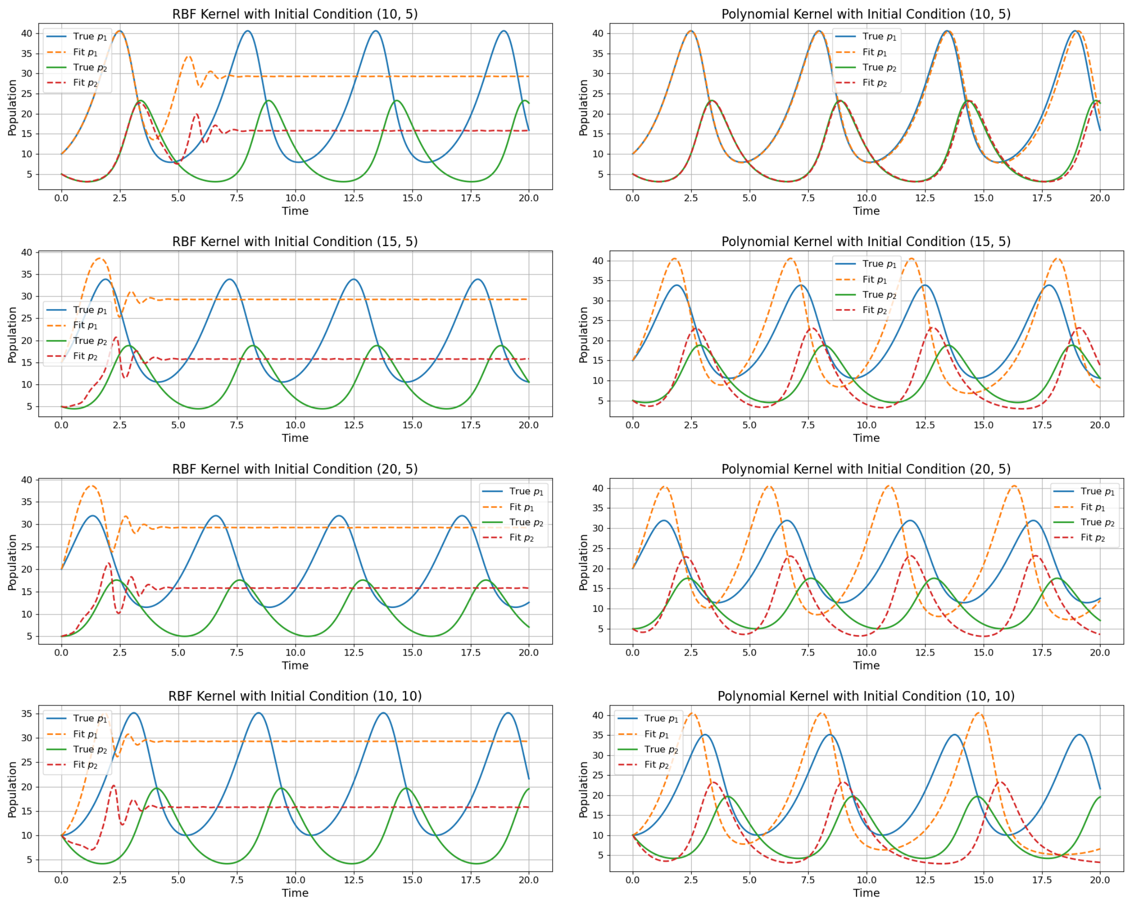

FIGURE 3. Simulation of learned models (dashed lines) created from the RBF Kernel and second degree polynomial kernel over the original initial condition and three new initial conditions, compared to the true LV model (solid lines).

4. Summary and Conclusions

From step 1 and step 2, we observed oscillating error values that eventually lead to a large error at the end of the interval. Meanwhile, in step 3, we observed for our RBF kernel that error increased as both p_1 and p_2 increased; for the polynomial model, error mainly increased when p_1 increased. In changing the initial conditions for our model, we found that the RBF kernel was unable to match the true LV trajectories as it loses its oscillations and flattens out over time. For the polynomial kernel, it was able to retain its oscillations, but it was best fit to the original model it was trained on, with it more closely following its original trajectory even as the initial conditions changed.

Future directions to consider would be to alter and tune the parameters further to prevent the overfitting observed from our models. In addition, it would be interesting to explore how well other kernels may perform, such as a linear kernel or an exponential kernel.

Acknowledgements

The author is thankful to Prof. Bamdad Hosseini for useful discussions about Kernel Ridge Regression and cross validation. I am also very thankful to David Kieke for providing insight into the implementation of KRR and cross-validation in Python.