AMATH 563: COMPUTATIONAL REPORT 3

NGHI NGUYEN

Applied Math Department, University of Washington, Seattle, WA
nghiiuu@uw.edu

1. Introduction

In this report, we will implement the Kernel Ridge Regression (KRR) with the Radial Basis Function (RBF) and polynomial kernel to predict the dynamical differential equations and trajectories of the Lotka-Volterra Model, a predator and prey population model. The accuracy of the method will be examined by finding the error at the trajectory, derivative level, along with a 2D contour over both populations. For variation, the KRR population functions will examined with different initial conditions.

2. Theory

The Lotka-Volterra model is a dynamical system of ordinary differential equations that describes the population change of a group of prey and predators over time. Denote the prey population as $p_1(t)$ and the predator population as $p_2(t)$ at time t. The model is as follows:

$$\begin{cases} \frac{dp_1}{dt} = \alpha p_1 - \beta p_2 \\ \frac{dp_2}{dt} = -\gamma p_2 - \delta p_1 p_2 \end{cases}$$

where the hyperparameter $\alpha,\ \beta,\ \gamma,$ and δ are positive real numbers. In the system, each scaled $p_1 p_2$ terms accounts for the interactions between both population with the respective β and δ interaction rates.

Given a set of 50 observations on the true state of the prey and predator population, the true trajectories of the populations are shown with the solid lines in 1 with the parameters $(\alpha,\ \beta,\ \gamma,\ \delta)=$ $(1,\ 0.1,\ 0.075,\ 1.5)$.

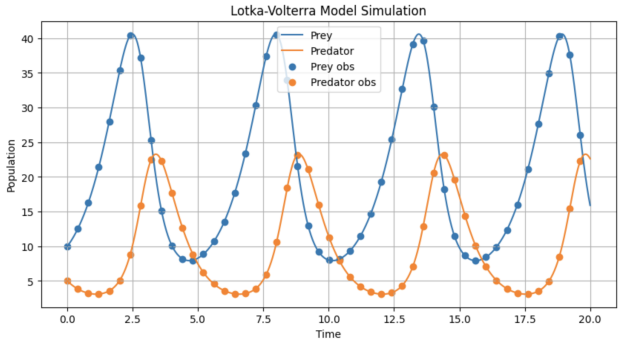

Figure 1. A plot showing the data.

Date: May 14, 2025.

In this report, suppose that the true trajectories and the four parameters governed in the system are unknown. To find the system of differential equations, we will train the Kernel Ridge Regression on the 50 observation pairs to find the system in the form

$$\begin{cases} \frac{dp_1}{dt} = f_1(p_1, p_2) \\ \frac{dp_2}{dt} = f_2(p_1, p_2) \end{cases}$$

.

3. Methods

To approximate the set of differential equations governing prey and predator change over time, the 3 following steps will be implemented:

Step 1: Kernel Ridge Regression (KRR) with the Radial Basis Function (RBF) kernel.

We begin by performing KRR, a regularized least square regression learning method in a Reproducing Kernel Hilbert Space (RKHS) H, on the observation pairs $y(t_n) = \{(p_1(t_n), p_2(t_n)\} \in \mathbb{R}^2$ where $t_n = .4n$ and $T_{50} = \{t_n\}_{n=0}^{49}$. Denote H_P as the RKHS associated with the positive definite symmetric RBF kernel. The KRR give rise to the following optimization problem:

$$\hat{\mathbf{p}}_i = \underset{p_i \in H_p}{\operatorname{argmin}} ||p_i(t) - y_i||_2^2 + \frac{\lambda}{2}||p_i(t)||_{H_p}^2$$

By Representor theorem, the solution $\hat{\mathbf{p}}_i$ can be represented as a linear combination of the RBF kernel as follow

$$\begin{aligned} \hat{\mathbf{p}}_i(t) &= \sum_{n=0}^{49} \hat{c}_n^i K(t, t_n) \\ &= \sum_{n=0}^{49} \hat{c}_n^i \exp(-\omega_i^2 (t - t_n)^2) \end{aligned}$$

where $\hat{c}_n^i$ is the optimal coefficient for each point-evaluated RBF kernel and $\omega_i = \frac{1}{2l^2}$ is the hyperparameter for RBF defined with the length-scale l. The vector of optimal coefficients can be solve with the following optimization problem,

$$\hat{c}_i = \underset{c_i \in \mathbb{R}^{50}}{\operatorname{argmin}} ||K(T_{50}, T_{50})c_i - y_i||_2^2 + \frac{\lambda}{2} c_i^T K(T_{50}, T_{50}) c_i, \qquad K(T_{50}, T_{50}) = \sum_{n=0}^{49}\sum_{m=0}^{49} K(t_n, t_m)$$

Denote $J(c_i) = ||K(T_{50}, T_{50})c_i - y_i||_2^2 + \frac{\lambda}{2} c_i^T K(T_{50}, T_{50}) c_i$; since $J(c_i)$ is convex, the optimal coefficient vector $\hat{c}_i$ can be calculated by taking the gradient such that $\nabla J(\hat{c}) = 0$. The population approximation functions and the coefficient vectors can be define as follow

$$\begin{cases} \hat{c}_i = (K(T_n, T_n) + \lambda I)^{-1} y \\ \hat{\mathbf{p}}_i(t) = K(t, T_n)^T (K(T_n, T_n) + \lambda I)^- 1 y_i \end{cases}$$

To implement the KRR on Python, we used import KernelRidge from sklearn.kernel_ridge. In addition, both Grid Search and Cross validation will be implemented onto the KRR and the training set. Grid Search finds the combination of the regularization parameter λ and the RBF kernel parameter ω that produces the lowest error between the training data and the approximation functions. Each combination will be tested on 10 folds of the training data and the errors for each fold will be averaged across, a process called Cross Validation. After this process, the approximating function $\hat{\mathbf{p}}_1$ and $\hat{\mathbf{p}}_2$ is examined at 950 testing time in the interval $[0, 20]$ excluding the training time. The accuracy of the population functions will be examined with the absolute error $|\hat{\mathbf{p}}_i - p_i|$.

Step 2: Compute the derivative of the Population Trajectory Approximation $\hat{\mathbf{p}}_i$.

In the first step, KRR was performed on 50 observation pairs to produce an approximation $\hat{\mathbf{p}}_i(t) \approx p_i(t)$ for $i = 1, 2$ to best fit the points, in this step, we take the derivative of $\hat{\mathbf{p}}_i$ on a denser time grid. Denote a uniform time grid with 100 points as $\tilde{T} = \{\tilde{t}_n\}_{n=1}^{100}$ where $\tilde{t}_n = .2n$, then derivative of $\hat{\mathbf{p}}_i(\tilde{t}_n)$ for $n = 1, ..100$ is as follow:

$$\frac{d}{dt}\hat{\mathbf{p}}_i(\tilde{t}_n) = \sum_{j=0}^{49} \hat{c}_j^i(-2\omega_i(\tilde{t}_n - t_j)), \qquad n = 1, ..100$$

In this step, 100 points of the derivative of the trajectory approximation are collected. To measure the precision of the KRR approximation, the absolute error $e_n = |\frac{d}{dt}\hat{\mathbf{p}}_i(\tilde{t}_n) - \frac{d}{dt}p_i|$ will be measured across the time interval $[0, 20]$ for $n = 1, .., 100$.

Step 3: KRR with the RBF and Polynomial kernel to approximate the dynamical equations.

Using the trajectory approximation $\hat{\mathbf{p}}_i(\tilde{t}_n)$ and the derivative $\frac{d}{dt}\hat{\mathbf{p}}_i(\tilde{t}_n)$ for the 100 points, we want to find the right hand sides of the Lotka-Volterra system with KRR. Denote $\mathcal{H}_i$ as the RKHS f_i lives in, then the approximation $\hat{\boldsymbol{f}}_i(\hat{\boldsymbol{p}}_1, \hat{\boldsymbol{p}}_2) \approx f_i(p_1, p_2)$ can be solved with the following optimization problem

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

where λ_i is the regularization parameter for the regression. For the KRR in this step, $\hat{\boldsymbol{f}}_i(\hat{\boldsymbol{p}}_1, \hat{\boldsymbol{p}}_2)$ will be expressed with the RBF and the polynomial kernel, $K(\hat{\boldsymbol{p}}_1, \hat{\boldsymbol{p}}_2) = (\hat{\boldsymbol{p}}_1^T \hat{\boldsymbol{p}}_2 + b)^d$ where $b > 0$ and $d \in \mathbb{N}$, to compare the kernels. Similar to step 1, a grid search and cross validation with 5 folds. Specifically, the grid search will find the best combination between the the parameters λ and ω for the RBF kernel and λ, d, and b for the polynomial kernel.

To examine the precision of the approximation $\hat{\boldsymbol{f}}_i(\hat{\boldsymbol{p}}_1, \hat{\boldsymbol{p}}_2) \approx f_i(p_1, p_2)$, the absolute error $|f_i - \hat{\mathbf{f}}_i|$ will be presented on a 2D contour plot in the domain $[0, 600] \times [0, 60]$ where $[0, 600]$ is the domain of the prey population and $[0, 60]$ is the domain of the predator population.

4. Results

Figure 2 shows the absolute error between the KRR trajectories $\hat{\mathbf{p}}_i$ and the true trajectories p_i. It is clear the magnitude of the error is small, and the RBF kernel did a good job recovering the behavior of the true population. However, the absolute error increases from a magnitude close to 0 to a magnitude greater than 1.25 for the predator error and 2.25 for the prey error around $t = 19$. This instability could be a result of the time grid T_n that the KRR first performed on. The KRR was only performed on $t_n \in [0, 19.6]$, whereas the time from 19.6 to 20 was outside the training set, leading to inaccurate trajectories outside $[0, 19.6]$.

Figure 3 shows the absolute error between the KRR population change $\frac{d}{dt}\hat{\mathbf{p}}_i(\tilde{t}_n)$ and the true population change $\frac{d}{dt}p_i(t)$. The absolute error for the population derivative shows similar behavior as the population as both errors blow up around $t = 19$. However. note that near $t = 0$, the error magnitude is relativity higher here than the interior time. This could be a result from the time grid change from Step 1 to Step 2. Step 1 uses the lower boundary point $t = 0$ to train the KRR, however in Step 2, the derivative did not include the $t = 0$ but rather $t = 20$. This inconsistency could lead to an inaccurate prediction at the boundary points.

Figure 4 shows the 2D contour of $|f_i - \hat{\mathbf{f}}_i|$ on the domain $[0, 600] \times [0, 60]$ where the predator population is the y-axis and the prey population is the x-axis. The white orbit is the natural orbit that Lotka-Volterra inhabits. As we get closer to the orbit, the error gets smaller for both kernels

and vice versa. This is expected as the KRR was only trained to predict within the trajectories. In addition, it is clear that the RBF kernel has a higher error, ranging up to 2700, while the polynomial kernel's absolute error only ranges up to 105 and 54.

Figure 5 and figure 6 shows the trajectories and phase portraits for the RBF and Polynomial kernel with various initial conditions. For the intial conditions (10,5), (30,5), and (10,15), both kernels inhabit similar trends and population number over time to the true trajectories. However, both kernels do not preserves the curvature and points of the the true trajectories well for the initial condition (5,5). While the polynomial kernel still inhabit similar curvature behaviors, the RBF kernel deviates both noticeably curvature and point-wise away from the true trajectories. This deviation is also shown in the phase portrait; while the polynomial has similar orbit curvature as the true obit, the RBF kernel does not inhibits the same curves as the true orbit. Nevertheless, this shows that the polynomial kernel performs better than the RBF kernel even for unfavorable initial conditions.

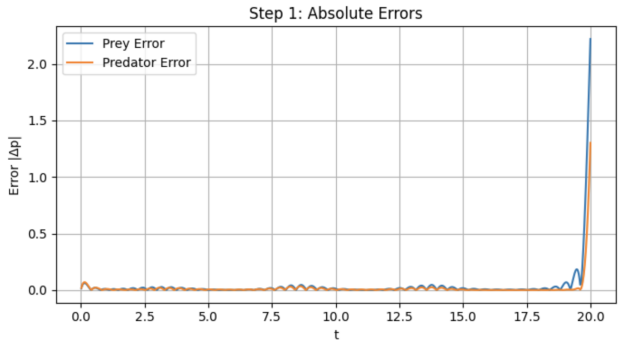

FIGURE 2. Step 1: The absolute error of the KRR Prey and Predator Trajectories

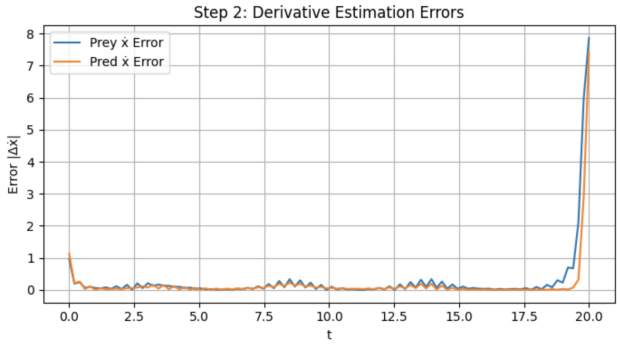

FIGURE 3. Step 2: The absolute error of the KRR Prey and Predator Population Change

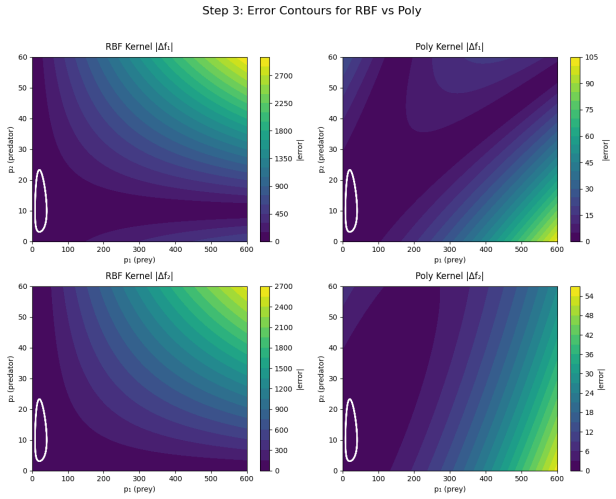

FIGURE 4. Step 3:The 2D Contour plot of $|f_i - \hat{\mathbf{f}}_i|$ over the $[0, 600][0, 60]$

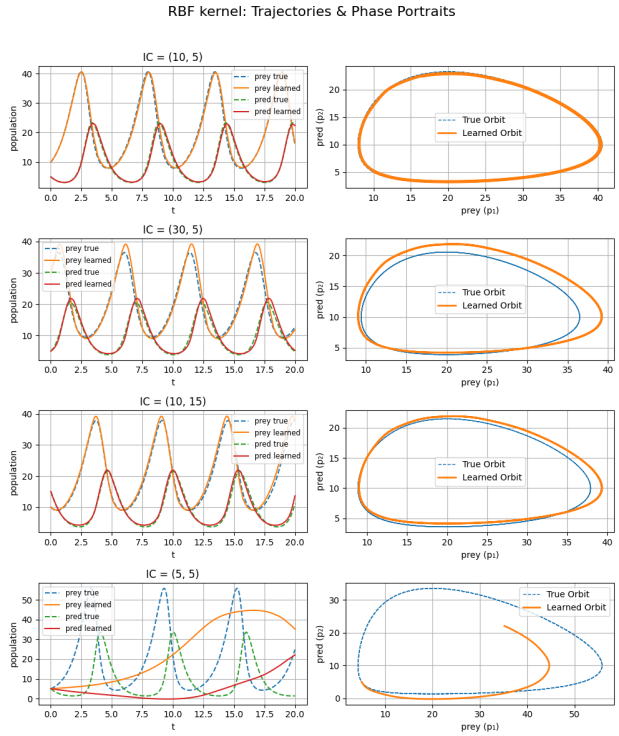

FIGURE 5. Trajectories and Phase Portraits with Different Initial Conditions and the RBF Kernel

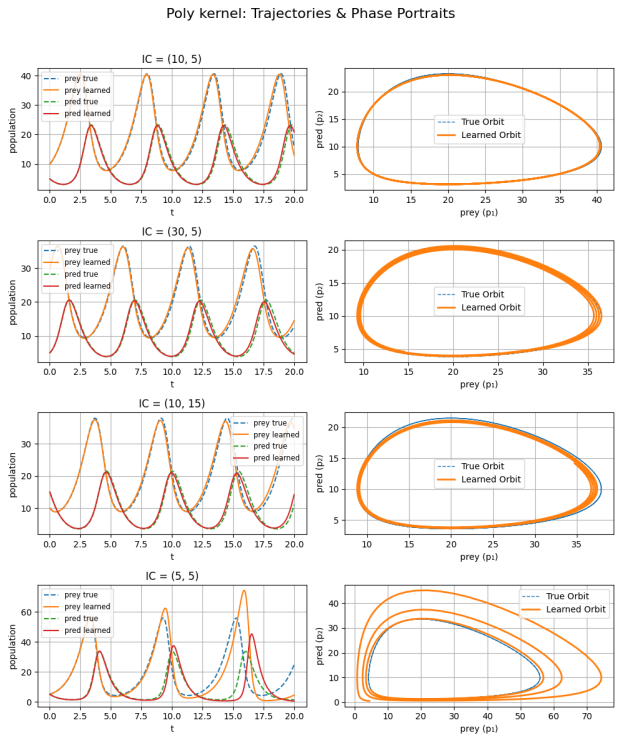

FIGURE 6. Trajectories and Phase Portraits with Different Initial Conditions and the Polynomial Kernel

5. Summary and Conclusions

In summary, the Kernel Ridge Regression (KRR) method was implemented to predict the dynamics of the Lokta-Volterra model on predator and prey population. Fifty observation pairs on both population were given and used to train the KRR with the Radial Basis Function (RBF) and predict the trajectories of both population. In addition, using Representor formula, we collected the derivative of the trained KRR trajectories on set of 100 temporal points to find the right-hand side of the Lokta-Volterra dynamical system. The absolute error of the KRR trajectories and derivatives to the true trajectories and derivatives shows consistent behavior as the errors were small on the interior time points and large on the boundary points. This could have been a result of modeling the trajectories outside of the trained data. For variation, the polynomial kernel was also used to find the dynamical equations and trajectories. The 2D contour of the populations for the polynomial kernel shows that the polynomial kernel has a lower error than the RBF kernel. This was also seen in the trajactories as the curvature of the true trajectories were preserved with the polynomial kernel among different initial conditions. Altogether, the Kernel Ridge Regression method allows us to model the dynamics of the prey and predator populations.

Acknowledgements

I would like to thank Emily, Tomas, Christina, Howard, and David for a helpful discussion.