Exploring Bee Behavior Through Survival Analysis

Jonah C. Bregstone

Johns Hopkins University, Whiting School of Engineering

Abstract

Bees are an essential part of our environment. Conserving bee populations is critical to maintaining our systems of agriculture and our environment.

**Exploring Bee Behavior Through Survival Analysis**

# Introduction

Our agricultural system and environment are reliant on pollinating insects, of which bees are some of the most efficient and effective. Bees visit pollinating flowers for nourishment, facilitating the transfer of pollen between flowers. This pollination allows for fruits and vegetables to grow year after year. For this reason, among many, it is critically important to the biodiversity of our ecosystem and to the sustainability of our agricultural systems that bees remain healthy and productive (Holm, 2022).

During the final year of my studies in the Applied Computational Mathematics program, I sought out opportunities to apply the math I was learning in the classroom to issues I cared about. I found an opportunity working as a technical researcher with the United States Geological Survey Native Bee Inventory and Monitoring Lab. While at the lab I had the opportunity to help design and pilot a bee monitoring experiment studying floral resource utilization by native bees.

**Methods of Data collection**

The Native Bee Lab uses repeated photography in our non-lethal pollinator monitoring experiment. The lab retro-fits donated cell-phones turning them into deployable bee-monitoring cameras. The phones are placed in the Patuxent wildlife preserve to take repeated photos of individual pollen producing blooms. After a day of photo capture, the phones are collected, and the data is uploaded for future analysis. Metadata about the capture event, including weather, time and location are all recorded.



Figure 1  
A recycled phone takes timed photos of yellow-tickseed (Coreopsis). A shade has been placed over the phone to keep the device from heating up in the Maryland sun.



Figure 2  
A honey bee lands on a  bur-marigold (bidens laevis). The flower is attached to a stake in order to keep the flower from moving during the capture period.

The research team records pollinator arrivals and approximate dwell times from the photos. Secondary analysis is completed by pollinator-ID specialists who attempt species-level classification of the photographed insects. The pollinator arrivals are a heuristic for pollinator flower-preference. Once complete, this flower-preference analysis can be disseminated to gardeners, ecologists and urban planners. This data will ensure that future pollinator-focused landscaping is conscious of native pollinator flower-preferences.

Through this project, I ended up with a tremendous amount of bee arrival data, roughly 67 hours of photo data and 25 hours of video data collected over the summer. This dataset collected through novel methods was ripe as an example of how applied computation mathematics can aid the shaping of social and environmental policy.

**Exploring the Data**

Firstly, I developed a lightweight data pipeline to categorize the data and turn the bee observations into a time-series dataframe. This step involved a tremendous amount of data cleaning and parsing. This was to take the bee arrivals and IDs of specimens noted by the team at the Native Bee Lab and convert this into an organized time-series of individual arrival events. During this process the data pipeline also produced metrics including arrival rates, dwell times, pollinator to pollinator interactions and other values that could be derived from the data.

Once the data had been collated and grouped in intervals, it was trivial to produce a model of the data using the Poisson probability distribution. Like many things in nature, the arrivals of bees to floral resources matches nicely with the Poisson distribution. The histogram below shows the percentage of 3-minute intervals were our cameras observed zeropollinator arrivals were detected, one pollinator arrival and two and three pollinator arrivals. About 8 percent have one pollinator arrival and very small number have two pollinators or three.

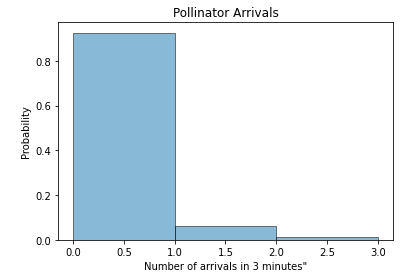


Figure 1  
Pollinator Arrivals grouped by 3 minute increments

Given that this fell into place so nicely, we wanted to see if we could calculate a more accurate Poisson model given a dynamic lambda. That is, consider the covariates of the bee arrivals and create a more specific curve. After exploring this, we decided a potentially good framework for this would be a Bayesian net model that would use the covariates to predict the presence of the bee. This type of Bayesian net ended up being difficult to produce because of the nature of our data. What we had was a bunch of collection events from one summer and it did not include repeated trails with certain covariates (i.e. weather) kept as a constant.

So finally, following the idea of a dynamic lambda poisson model we turned to the world of survival analysis.

**Survival Analysis**

**The math behind survival curves and hazard functions**

**Time-dependent covariates**

**Methods for dealing with time-dependent covariates in Survival Analysis**

**Neural Networks**

One approach to handling time-dependent covariates in survival analysis that has seen interest from the community is to employ the use of machine learning. Rather than doing a traditional regression, which is the method underlying the cox regression analysis, the data from a study can be fed to a neural network and the network can learn to predict the survival curve for the data.

In this study, several machine learning methods/frameworks were surveyed before selecting the PyCox framework to build a model with the dataset. The pycox framework builds off of the machine learning framework PyTorch. PyCox provides the Neural Network architecture and model classes to perform deep learning to predict survival curves. Using this framework involves splitting up the data into test, train and validation sets. A PyCox Logistic Hazard model was trained on the test data and then the completed model’s predictive capabilities were used to derive a population survival curve for the test data.

The Pycox framework for creating survival curves possesses a notable drawback, namely the issue of explainability. This limitation arises from the inherent nature of neural networks, which are unable to provide explicit explanations for weight allocation to the model. However, the inability to comprehend how and why certain nodes are assigned greater importance than others poses a challenge in survival analysis.

Understanding the weight of various covariates is crucial in survival analysis, particularly in a clinical context where studies aim to identify the connections between covariates (such as gender, age, and tobacco use) and diagnoses. Consequently, the inability to ascertain the weights of these covariates hampers the ability to draw meaningful conclusions from such studies and prescribe appropriate treatments. This drawback diminishes the practical utility of the Pycox framework.

Without the covariate weights, the trained neural network main utility is prediction. In the study, the model generates predictions for individual survival curves, which can then be aggregated to form population-level survival curves. However, the specific covariates influencing this function are not explicitly elucidated by the trained neural network. As a result, the framework fails to provide a clear understanding of the underlying factors that contribute to the observed survival outcomes. And, concerning our time-dependent variables even though the pycox method is able to handle time-dependent covariates, the resulting model does not give clear answers about how the effect of different covariates changes over time which would be part of the meaningful difference of a survival analysis method that does not need to adhere to the proportional hazards assumption. Luckily, there are methods to derive the covariate weights over time using multiple regression.

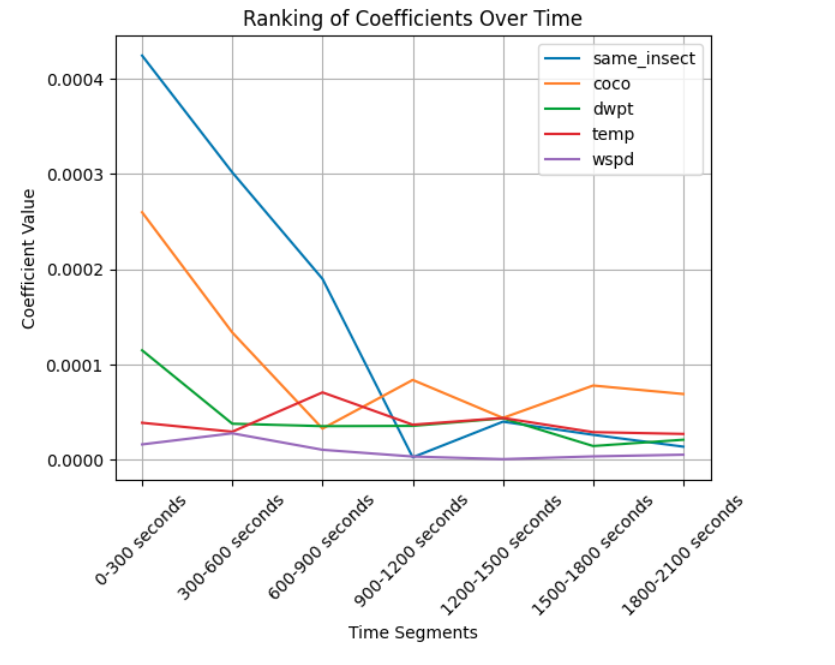
**Deriving covariate weights from survival curve (Tom’s Method)**

The method of multiple regression draws from the cox regression method but discretizes time along the curve to allow the change in ranking of covariate weights. By splitting the predicted survival curve into discrete time-segments a regression could be run on all “active” events during that time segment. The resulting coefficients would reveal the covariate weights at that time-segment. An event was considered “active” if the event had not terminated before the start of the time-segment.

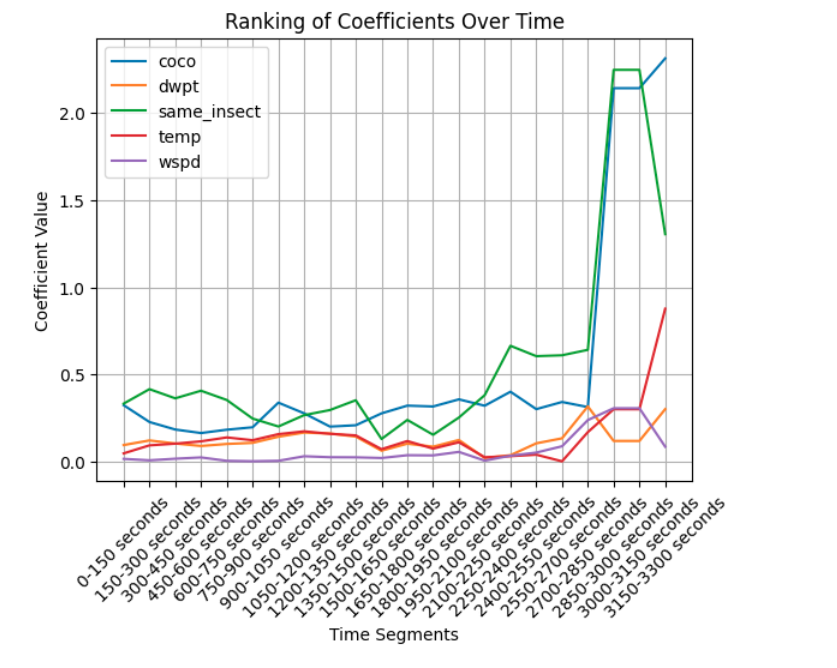
If thought about abstractly, this method of deriving the changing weights of time-dependent covariates is a very simple extension of the Cox regression method. By splitting up our data into time-segments and running a regression on each, we are following the logic of the cox regression analysis which expects the segment to have a stable proportion between it’s hazards. By doing this at each time-segment, we maintain the proportional hazard assumption at each regression while still exploring time-dependent covariates in the survival curve overall.

The specifics of the regression involve using the trained pycox model to predict survival curves for each “active” event in the time-segment. Then, the function calculates the slope of the survival curve during the relevant time-segment and this is used as the dependent variable for our regression. The independent variables are the covariates for each event enumerated above.

It is important to note that this method is not without its limitations. One limitation is that it can be computationally expensive depending on how many time-segments one has defined in the survival curve. However, this method can be a valuable tool for analyzing survival data with time-varying covariates and censored data.



Here is the time-segmented moving-regression method applied to the pycox model. We have graphed the coefficients of the covariates over time in 300 second segments from 0 to 2100 seconds. The most striking thing about this graph is the dominance and then steep drop-off of predictive capability with the same\_species covariate. What this might show is an issue with the data-coding. If the same bee is returning to the flower it is possible that the team at the Native Bee Lab encoded the same bee as new arrivals to the floral resource.



Here a similar method for graphing the weights with traditional cox analysis is shown. One big difference between the moving function of regression is that for the traditional cox, the function is not manually calculating the slope and using that as the dependent variable. Instead the “active” events are fed directly to the cox proportional hazard model `fit()` function from the lifelines package.

**Stacking Method**

Stacking is a form of data manipulation that cast survival analysis problems as classification problems (Tibshirani). This transformation allows the use of general classification methods (e.g. Logistic Regression, Random Forest) to be performed on the data with the goal of computing a survival curve.

The mechanics of the stacking method involve transforming the data into a matrix. For our data, each observation of a bee arrival becomes a new row in the matrix.

**Bibliography**

**Chadès, Iadine, et al. "MDPtoolbox: a multi‐platform toolbox to solve stochastic dynamic programming problems." Ecography 37.9 (2014): 916-920.**

**Boutilier, Craig, Richard Dearden, and Moisés Goldszmidt. "Stochastic dynamic programming with factored representations." *Artificial intelligence* 121.1-2 (2000): 49-107.**

**Bellman, Richard. "Dynamic programming." *Science* 153.3731 (1966): 34-37**

**Blackburn. “Reinforcement Learning: Markov-Decision Process.” *Towards Data Science* (2019)**

**Marescot, Lucile, et al. "Complex decisions made simple: a primer on stochastic dynamic programming" Methods in Ecology and Evolution (2013)**

**Chadès, Iadine, et al. “Quick Sat: Resolving a Markov decision process problem using the MDPtoolbox in Matlab” (2014)**

**Dubois, T., Pasquaretta, C., Barron, A. B., Gautrais, J., & Lihoreau, M. (2021). A model of resource partitioning between foraging bees based on learning. PLoS Computational Biology, 17(7), 1-19. [e1009260].** [**https://doi.org/10.1371/journal.pcbi.1009260**](https://doi.org/10.1371/journal.pcbi.1009260)

**Holm, Heather. “Common Native Bees of the Eastern United States.” Adventure Publications (2022)**

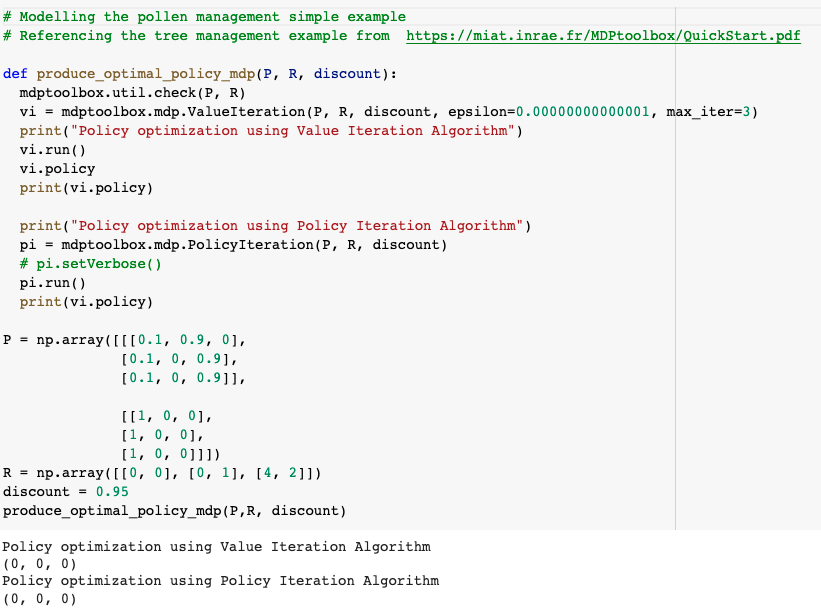
**Cordwell, Steven “Markov Decision Process (MDP) Toolbox for Python.“ (2015) Read The Docs** [**https://pyMDPtoolbox.readthedocs.io/en/latest/**](https://pymdptoolbox.readthedocs.io/en/latest/)

**Hicks, Damien et al. “Deep learning object detection to estimate the nectar sugar mass of flowering vegetation” Ecological Solutions and Evidence (2021).**

**Code Samples**

I wanted to highlight a few sections ­of the code used in this paper. The complete codebase can be found on github at: https://github.com/puckybreg/625.722-Markov-Decision-Process-Project

Code Sample 1  
This is a function written to produce the optimal policy for the MDP environment, using both the value iteration and policy iteration algorithms



Code Sample 2  
This simple function is used to calculate and plot the steady state distributions of a Markov Chain.

