Ph11 Second Hurdle (2023)
Amphibians on Oceanic Islands

Enoch Guo
eguo@caltech.edu

November 27, 2023

Abstract

In this hurdle, the goal is to model the colonization of Sao Tome, a volcanic island. First, we answer whether it is odd that the island has 7 species of amphibian despite colonization events being rare. We do this using a binary classification algorithm that classifies species as either being capable of immigrating to Sao Tome or not. Secondly, we give a probabilistic model of the colonization of *Ptychadena newtoni*, a type of frog. We use the same classifier to determine the probability with which this species makes it to the island. Then using some fundamental principles of Island Biogeography Theory, we derive an expression for the colonization rate based on carrying capacity, birth rate, and death rate. Lastly, using R, we model a sample population of *Ptychadena newtoni*.

1 Introduction

There are two portions to this problem. One, we must determine if it is probable or even possible for *Ptychadena newtoni* to migrate from the Western coast of Africa to the island of Sao Tome. Second, we must consider upon reaching the island, what is the probability that the species can actually propagate and successfully colonize the island without going extinct rapidly. We solve the first part using a binary classifier, and the second using probabilistic modeling based on established theory.

2 Assumptions

I make the following assumptions throughout my method:

1. Islands are Isolated Systems: The theory assumes that islands are isolated systems, whether they are true oceanic islands or habitat fragments surrounded by different landscapes. This isolation limits the interaction and exchange of species with the mainland or other islands.

2. Limited Immigration: Islands have a limited potential for new species to immigrate and colonize due to their isolation. Immigration rates are influenced by factors such as distance from the mainland and the size of the island.

3. Dynamic Equilibrium: Island biogeography theory proposes that the species richness of an island is in a state of dynamic equilibrium, balancing the colonization (immigration) and extinction rates. This equilibrium establishes the total number of species that an island can support.

4. Birth rates and death rates of a species are population density-dependent (that is, they are proportional to the population at a given time).

5. The probability that a species can immigrate to Sao Tome can be determined based on some intrinsic factors of a species e.g. size, weight, length of mating season, etc. (This is a necessary condition as we assume these parameters to influence the migration probability but we do not really know as we use a neural network).

6. *Ptychadena newtoni*'s population dynamics behave similarly to its close relatives on mainland Africa.

3 The Dataset

In order to use a classifier algorithm, first we had to compile our own training data. We scraped the web, mainly Wikipedia and IUCN red list, for data on (approx. all) 150 species of animal on the island of Sao Tome. We also gathered the data for an additional 350 species of animals from the Eastern and Western coast of Africa that did not historically appear on Sao Tome. This resulted in a total of 500 data points. The data we gathered included the following categories: [Average Size (cm) ,Weight (g), Lifespan (years), Peak Breeding Month (1 for Jan, 2 for Feb, etc.), Duration of breeding period (months), Number of Known Predators, Diet Classification (1 for herbivore, 2 for carnivore, 3 for omnivore), Distance from endemic habitat to Sao Tome (km), Number of Offspring, Water Survivability (0 for no, 1 for yes)]. This resulted in a 1 by 10 vector representing the traits for

each species, and each species also received a label of [0] for not having made it to Sao Tome or [1] for capable of immigrating to Sao Tome (the 150 species were labeled [1]). For example, the corresponding vector for *Ptychadena newtoni* was [8,10,4,3,1,2,2,300,300,1]. After exporting all this data into a .csv file, we then converted it to a custom dataset using Pytorch's Dataloader. The dataset was split into 80 percent training data and 20 percent test data. *Ptychadena newtoni* was not included in either, we used our fully calibrated model to make the prediction for it.

4 The Architecture

We employed two binary classification algorithms. One of them was a simple neural network with 3 fully connected layers. For this algorithm, we had a learning rate of 0.001, used the Adam optimizer, BCE loss function, ReLU activation functions, and trained for 1000 epochs with a batch size of 25. The model returned a number between 0 and 1 representing the probability of reaching the island of Sao Tome. The second algorithm we used was a vanilla XGBoost classifier.

5 Reducing Overfitting

In order to combat overfitting, we randomly sampled from our training data to generate batches that were evenly distributed (50 percent each) between data points that were labeled 0 and 1. This way, our data would be balanced and the model would not simply replicate the distribution of the data.

6 Results

The neural network finished with a loss of 0.017 and a test accuracy of 91 percent. It predicted *Ptychadena newtoni* a 86 percent chance of immigrating to Sao Tome. The XGBoost algorithm finished with a test accuracy of 93 percent and predicted a 90 percent chance of immigration. From this we can conclude that *Ptychadena newtoni* would have most likely been capable of reaching the island Sao Tome on one of these rare colonization events. Moreover, after testing the other 7 species of amphibian, we found that it is not too surprising that they are all on Sao Tome because after repeated iterations of the simulation, between 4 and 6 of the species were consistently classified as having made it to the island.

7 XGBoost Code

```
import csv

from sklearn.model_selection import train_test_split
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score
import numpy as np

csv_file_path1 = '/Users/PycharmProjects/ph11hurdle/x_data.csv'
x_data = []
with open(csv_file_path1, 'r') as file:
    csv_reader = csv.reader(file)
    for row in csv_reader:
        # Convert each row into a list of integers
        row_list = [float(val) for val in row[0].split(',')]
        x_data.append(row_list)

x_data = np.asarray(x_data, dtype=np.float32)

csv_file_path2 = '/Users/PycharmProjects/ph11hurdle/labels.csv'
y = []
```

```
with open(csv_file_path2, 'r') as file:
    csv_reader = csv.reader(file)
    for row in csv_reader:
        # Convert each row into a list of single-element lists
        row_list = [[int(val)] for val in row[0].split(',')]
        y.extend(row_list)

# Convert the DataFrame to a Python list
y = np.asarray(y, dtype=np.float32)

# Assume X is your feature vectors (2D array or DataFrame) and y is your labels
X, y = x_data, y
# Splitting the dataset into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2,random_state=42)

print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)

print(y_train[y_train == 1].shape)
print(y_train[y_train == 0].shape)
print(y_test[y_test == 1].shape)
print(y_test[y_test == 0].shape)

# Create an instance of the XGBClassifier
xgb_model = XGBClassifier()

# Train the XGBoost model
xgb_model.fit(X_train, y_train)

# Predict on the test set
y_pred = xgb_model.predict(X_test)

# Evaluate the accuracy of the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}")
```

8 Neural Network Code

```
import csv
import torch.nn as nn
import torch.optim as optim
import torch
import pandas as pd
import numpy as np
from torch.utils.data import Dataset, DataLoader, random_split, TensorDataset

# Check for GPU availability
# device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

# Read the CSV file
csv_file_path1 = '/Users/PycharmProjects/ph11hurdle/x_data.csv'
data = pd.read_csv(csv_file_path1, header=None)
```

```
# Convert the data to a Python list of 10x1 matrices
x_data = []
with open(csv_file_path1, 'r') as file:
    csv_reader = csv.reader(file)
    for row in csv_reader:
        # Convert each row into a list of integers
        row_list = [float(val) for val in row[0].split(',')]
        x_data.append(row_list)
x_data = np.asarray(x_data, dtype=np.float32)
tensor_x = torch.tensor(x_data)

# Read the CSV file
csv_file_path2 = '/Users/PycharmProjects/ph11hurdle/labels.csv'

# Read the CSV file and parse each line into a list of single-element lists
labels = []
with open(csv_file_path2, 'r') as file:
    csv_reader = csv.reader(file)
    for row in csv_reader:
        # Convert each row into a list of single-element lists
        row_list = [[int(val)] for val in row[0].split(',')]
        labels.extend(row_list)

# Convert the DataFrame to a Python list
labels = np.asarray(labels, dtype=np.float32)
tensor_y = torch.tensor(labels)

train_data = TensorDataset(tensor_x, tensor_y)

# Define the sizes for train and test datasets
total_size = len(train_data)
train_size = int(0.8 * total_size)
test_size = total_size - train_size

# Splitting the dataset into train and test sets
train_dataset, test_dataset = random_split(train_data, [train_size, test_size])

# Create DataLoader for train and test sets
batch_size = 25
train_loader = DataLoader(train_dataset, batch_size=batch_size, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=batch_size, shuffle=False)


# Define the neural network
class BinaryClassifier(nn.Module):
    def __init__(self):
        super(BinaryClassifier, self).__init__()
        self.fc1 = nn.Linear(10, 64)  # Input size: 10, Output size: 64
        self.fc2 = nn.Linear(64, 32)  # Input size: 64, Output size: 32
        self.fc3 = nn.Linear(32, 1)  # Input size: 32, Output size: 1
        self.sigmoid = nn.Sigmoid()  # Sigmoid activation for binary classification

    def forward(self, x):
        x = torch.relu(self.fc1(x))
        x = torch.relu(self.fc2(x))
        x = self.fc3(x)
```

```
        return self.sigmoid(x)  # Applying sigmoid activation for binary classification


# Create an instance of the neural network and move it to the GPU (if available)
model = BinaryClassifier()

# Define the loss function and optimizer
criterion = nn.BCELoss()  # Binary Cross Entropy loss
optimizer = optim.Adam(model.parameters(), lr=0.001)

num_epochs = 10  # Replace with the desired number of epochs
for epoch in range(num_epochs):
    size = len(train_dataset)
    running_loss = 0.0
    for i, data in enumerate(train_loader):
        inputs, labels = data
        optimizer.zero_grad()
        output = model(inputs)
        loss = criterion(output, labels)
        loss.backward()
        optimizer.step()

        running_loss += loss.item()
        if i % 2 == 1:  # print every 2 mini-batches
            print(f'[{epoch + 1}, {i + 1:5d}] loss: {running_loss / size:.3f}')
            running_loss = 0.0

    test_loss, correct = 0, 0

correct1 = 0
total = 0
with torch.no_grad():
    for data in test_loader:
        animals, labels = data
        outputs = model(animals)
        _, predicted = torch.max(outputs.data, 1)
        total += labels.size(0)
        correct1 += (predicted == labels).sum().item()

print(f'Accuracy: {100 * correct // total} %')


test_input = torch.randn(1, 10).to(device)
predicted_output = model(test_input)
print("Predicted Output Probability:")
print(predicted_output)
```

9 Predicting the Colonization Rate

Now that we have established that *Ptychadena newtoni* has a good chance of ever making it to the island, we must consider the population dynamics that come into play on the island and see if the species would actually be able to reproduce successfully enough to not die out, and at what rate it colonizes the island. Define colonization rate as the probability that a species successfully colonizes and thrives on the island. First we notice the chance of an individual eventually growing to the carrying capacity K is about $\frac{\lambda-\mu}{\lambda}$, where λ is birth rate and μ is death rate. Define λ_x as the birth probability of the population changing from size x to $x+1$ across some time interval. Similarly

define μ_x as the death probability of the population changing from size x to $x-1$ across the same interval. Then the probability of any population size change on some interval h, positive or negative, is given by $(\lambda_x+\mu_x)h$. Also the fraction of population changes that are positive and negative would be $\lambda_x/(\lambda_x+\mu_x)$ and $\mu_x/(\lambda_x+\mu_x)$ respectively. Across some time interval h, the population is then modeled as $P(t+h) = P(t)[1-(\lambda_x+\mu_x)h]$, where $P(t)$ is the exponential population growth function $e^{-(\lambda_x+\mu_x)t}$. So the change in population with respect to time is $\frac{dP(t)}{dt} = \lim_{h\to 0} \frac{P(t+h)-P(t)}{h} = -P(\lambda_x+\mu_x)$. This gives us a p.d.f with expected value $\int_0^\infty t(\lambda_x+\mu_x)e^{-(\lambda_x+\mu_x)t}\,dt = \frac{1}{\lambda_x+\mu_x}$. The expected time per change is $\frac{1}{\lambda_x+\mu_x}$. Using this we can derive a formula for time until extinction of one propagule of a species, which is a good measure for its colonization rate. The expected time for a population to go extinct is then recursively defined $T_x = \frac{1}{\lambda_x+\mu_x} + \lambda_x/(\lambda_x+\mu_x)T_{x+1} + \mu_x/(\lambda_x+\mu_x)T_{x-1}$. Because we assume density dependent birth and death rates, we can replace the coefficients with $\lambda_x = \lambda x$ and $\mu_x = \mu x$, and taking T_K for K the carrying capacity gives us a relation with λ, μ, and K which we can use in tandem with our first equation to compute estimates of the colonization rate.

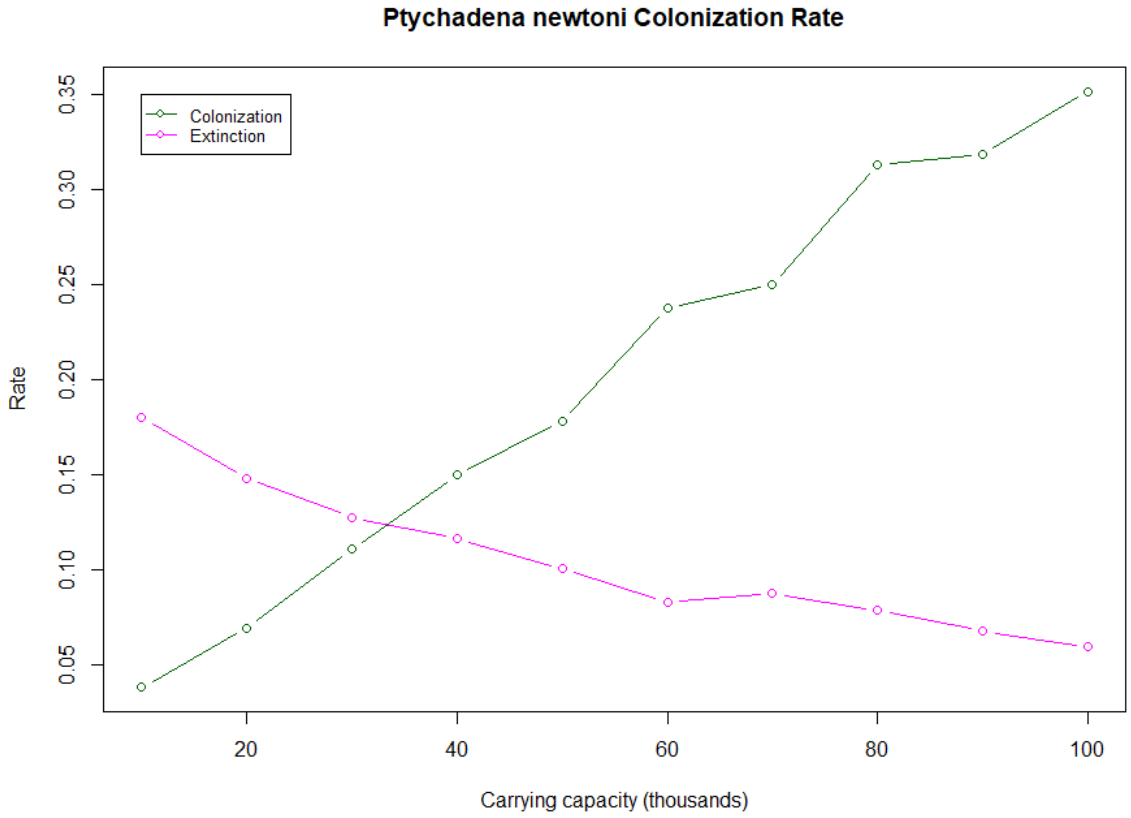

Figure 1: Sample simulation to find the most probable colonization rate and carrying capacity.

10 R Code for Predicting Colonization Rate

With the help of a package called "island", we can model this and run simulations for different values of K, the carrying capacity. Below, beta is the birthrate, delta the death rate, mu the immigration rate, and K the carrying capacity. These are hyperparameters which can be adjusted accordingly. For our particular case of *Ptychadena newtoni*, we choose these values based on knowledge of its close relatives in Africa. Because the frog has less natural predators on Sao Tome, we decrease the death rate, and because we know colonization events are extremely rare, we will keep immigration rate rather small. The birth rate would stay about the same, based on our statistics of breeding month and duration of breeding time. We will vary K. So birth rate = 0.2, death rate = 0.3, immigration rate = 0.004.

```
ts <- 0:100 #Time-vector
```

```
ccc <- seq(10, 100, 10) #Carrying capacities
out <- NULL #Initializing output
for (i in ccc){
pops <- matrix(nrow = 100, ncol = 101)
for (j in 1:100){
  sim <- ibd_models(n0 = 0, beta = 0.2, delta = 0.3, mu = 0.004, K = i, time_v = ts,
                    type = "Alonso") #Simulations
  pops[j, ] <- (sim[, 2] > 0) * 1.0
}
out <- rbind(out, c(i, regular_sampling_scheme(pops, 1:101)))
}

#Plotting
plot(out[, 1], out[, 2], type = "b", xlab = "Carrying capacity (thousands)",
ylab = "Rate",
     col = "darkgreen", main = "Ptychadena newtoni Colonization Rate")
lines(out[, 1], out[, 5], type = "b", col = "magenta")
legend(10, .35, legend=c("Colonization", "Extinction"),
       col=c("darkgreen", "magenta"), pch = 21, lty=1, pt.bg = "White",  cex=0.8)
```

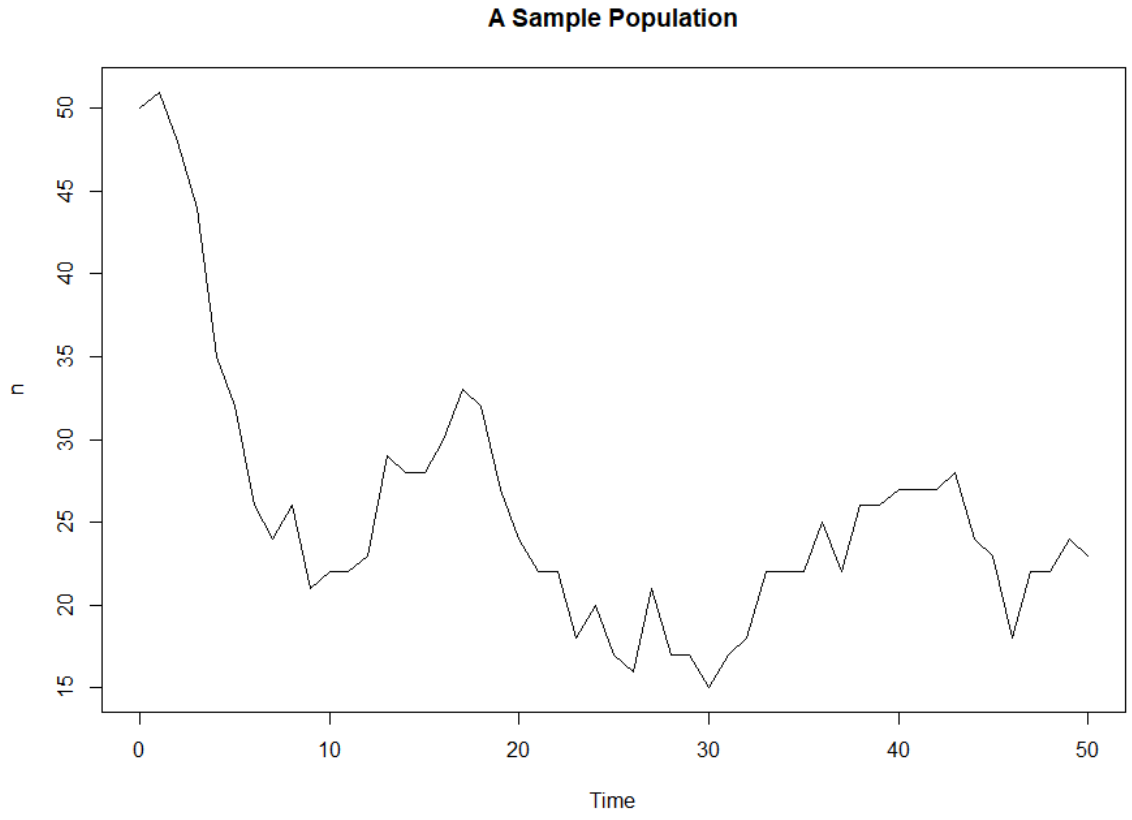

Figure 2: A sample simulation of a Ptychadena newtoni population with 50 initial immigrants. Time is in years and n in thousands.

11 A Sample Population

We can also generate a hypothetical instance of a colonization by this species and see how the population fluctuates. All the hyperparameters should remain the same, with the exception of K which

we plug in from earlier and n0, the initial amount of immigrant individuals in a species, which we arbitrarily set to 50.

12 R Code for A Sample Population

```
set.seed(101100111)
dynamic <- ibd_models(n0 = 50, beta = 0.2, delta = 0.3, mu = 0.004, K = 40,
time_v = 0:50, type = "Alonso")
plot(dynamic, type = "l", main = "A Sample Population")
```

13 Conclusion

From this we can conclude that *Ptychadena newtoni* would colonize Sao Tome at a rate of 0.12 (probability of 12%), reaching a carrying capacity of 35,000.

14 Discussion

Ideally, we could have estimated the colonization rate based directly on the equilibrium point of colonization and extinction rates of all species, without access at all to carrying capacity. This would be a more community based approach as opposed to modeling based only on the individual species' statistics. Based on the basic model of Equilibrium Theory of Island Biogeography,

$$\frac{dS_s}{dt} = c(S_p - S_s) - eS_s$$

where S_s is the number of species present at a site, S_p is the number of species in the regional pool, and c and e are colonization and extinction rates respectively. This equation can be solved for a single species, allowing us to compute the absence or presence of a species at a specified time, which we can use to construct a Markov model of transition probabilities (the probability whether a species will go from present to absent ($1 \to 0$), absent to present ($0 \to 1$), or stay the same ($0 \to 0$ or $1 \to 1$) from one time interval to the next):

$$T_{00} = 1 - \frac{c}{e+c}(1 - exp(-(e+c)\Delta t)) \tag{1}$$

$$T_{10} = \frac{c}{e+c}(1 - exp(-(e+c)\Delta t)) \tag{2}$$

$$T_{01} = \frac{c}{e+c}(1 - exp(-(e+c)\Delta t)) \tag{3}$$

$$T_{11} = 1 - \frac{c}{e+c}(1 - exp(-(e+c)\Delta t)) \tag{4}$$

T_{01} is the probability of extinction, T_{10} the probability of colonization, T_{11} of repeated presence and T_{00} of repeated absence. However, using this method of calculating colonization rate requires a data set of random samples of multiple species' presence/absence across discretized time intervals, which is something we cannot obtain easily. In addition, given more time, we also could have modeled environmental factors that come into play and more importantly, the ecological theory of variable niches and invasibility; that is, if there are existing species that already fulfill another species niche, that will adversely affect its chances of colonization through direct competition, but we do not know which species arrived earlier or later so we cannot compute this.

15 Sources

- MacArthur, R. H., & Wilson, E. O. (1967). The Theory of Island Biogeography. Princeton University Press.
- Wildlife of São Tomé and Príncipe, Wikipedia

- Alonso et al. (2019). Colonization and extinction rates estimated from temporal dynamics of ecological communities: The island r package