Lab - Species Distribution Modeling with GAMs

BIOHOPK 143H - Winter 2021

Getting Set Up

Place the `EBS_survey_data.csv`, `extrapolation_grid.csv`, and `plot_functions.R` files in the same folder, and set your working directory to that folder.

Install the `tidyverse`, `mgcv`, and `dismo` packages (if you don't already have them installed) and load them.

```r
library(tidyverse)
library(mgcv)
library(dismo)
```

Also, source the `plot_functions.R` script, which provides a couple handy functions to plot the model predictions, once we've gotten to that step:

```r
source("plot_functions.R")
```

Finally, load the data, which are contained in the `EBS_survey_data.csv` file:

```r
EBS_data <- read.csv("EBS_survey_data.csv")
```

The variables contained in this dataset are:

- `year` and `day_of_year` - the year and day within year (i.e. "julian date") of the survey
- `station_id` - a unique identifier for the survey station, which are locations that are sampled repeatedly throughout time
- `E_km` and `N_km` - the UTM coordinates of the survey in Eastings and Northings
- `distance_fished` - the distance of the trawl survey, in kilometers
- `depth` - the bottom depth for the trawl survey, in meters
- `bottom_temp` - the bottom temperature for the trawl survey, in degrees celsius
- `halibut`, `flounder`, and `pollock` - a binary variable (0 or 1) indicating whether the given species (Pacific halibut, arrowtooth flounder, or juvenile walleye pollock) was observed in the trawl survey

Discussion Question You can use any of these variables in building your species distribution model. Which of these variables might be considered "environmental" predictors? Which aren't environmental predictors, but seem like they should be accounted for anyway?

Fitting a Generalized Additive Model

Model Form

We'll be fitting models of the form

$$Y \sim \text{Bernoulli}(p)$$
$$\text{logit}(p) = \alpha + f_1(x_1) + f_2(x_2) + \cdots$$

Where Y is whether or not our species of interest occurs ($Y = 1$) or is absent ($Y = 0$) from a given sampling site. The probability of occurrence p is a linear function (on the log-odds scale) of an intercept α and a combination of functions $f_i(x_i)$ for each predictor x_i. These functions can be pretty much *anything* - when we fit a GAM, we're usually interested in fitting **smooth functions**, which represent the weighted sum of k polynomial "basis" functions, but we can also fit linear effects, or categorical predictors, etc.

Recall that the Bernoulli distribution is a special case of the Binomial distribution (when $n = 1$), so to fit a Bernoulli GAM (with a logit link function) in R we'll specify `family = binomial(link = "logit")` when we run our model.

Model predictors

We'll use the `gam` function, in the MGCV package, to run our models. To fit a thin-plate regression spline of temperature as the only predictor of pollock occurrence, we could run:

```r
btemp_model <- gam(pollock ~ s(bottom_temp, k = 5), data = EBS_data,
                   family = binomial(link = "logit"))
```

In this code chunk, `pollock ~ s(bottom_temp, k = 5)` is a **formula** - it tells R that our predictor, `pollock`, depends on (`~`) some variables. Implicitly, an intercept is included. The `s(bottom_temp, k = 5)` term tells `mgcv` that we want to fit a smooth function (by default, a thin-plate regression spline) of bottom temperature, with a maximum `k = 5` basis functions.

Generally, it's good to keep the number of basis functions small when we expect there not to a complex relationship - in the case of temperature, for example, we expect species to have a "hump-shaped" temperature preference. If we allow the model to use too many basis functions, however, it will generally shrink the weights of some of the basis functions towards zero, effectively eliminating the extra complexity.

We can fit smooth terms for depth, or year, or the number of days in a year, in a similar way (although we might want to play around with `k` for those terms). We can use two-dimensional smoothers to fit spatial models - With these, it's good to give a large number of basis functions to work with, in each direction. We can try it here with $k = 100$. There are several ways we can fit these, but we'll use the default thin-plate regression splines here too:

```r
spatial_model <- gam(pollock ~ s(E_km, N_km, k = 100), data = EBS_data,
                     family = binomial(link = "logit"))
```

We can add a temporal term with:

```r
sp_temp_model <- gam(pollock ~ s(E_km, N_km, k = 100) + s(year, k = 10),
                     data = EBS_data, family = binomial(link = "logit"))
```

We can even fit **spatio-temporal** smoothers by including spatial, temporal, and a spatio-temporal smoother with space and time in it. Here, we'll use a tensor interaction smooth (`ti()`), and indicate with `d = c(2,1)` that we want an interaction between a 2-dimensional spatial smoother and a 1-dimension temporal smoother. This model might take a few minutes to run:

```r
spatiotemp_model <- gam(pollock ~ s(E_km, N_km, k = 100) + s(year, k = 10) +
                          ti(E_km, N_km, year, d = c(2, 1)),
                        data = EBS_data, family = binomial(link = "logit"))
```

In this lab, play around with several different models - including models with just environmental predictors, ones with just spatial/temporal/spatio-temporal predictors, and ones with both.

Assessing predictor effects

Plotting smoothers

Once we've fit a model with our chosen combination of predictors, we can plot the smoothers from these models by just using the "plot" function. We can plot spatial, temporal, and spatio-temporal smoothers in the same way.

Here's the temperature smoother from the model we fit above with only bottom temperature. **What do you expect this to look like? What does it look like for your species? When you include other covariates, does this smoother change?**

```r
plot(btemp_model)
```

Significance of model terms

We can assess the statistical significance of model terms (that is, does this term explain significantly more variance than a horizontal line?) using the `anova` function.

The `edf`, or "estimated degrees of freedom" given by this function tell us something imporant as well - if the estimated degrees of freedom are much lower than the number of basis functions we specified in our model, than the model estimated the relationship between the response and our predictor to be simpler than we expected. However, if the estimated degrees of freedom is very close to the maximum we allowed in the model, it is likely that the relationship would be better described by a more complex function, and we should increase the number of basis functions for that term.

```r
anova(spatiotemp_model)
```

Characterizing model accuracy

AIC and R-squared

We can obtain R-squared (the amount of variance in the data explained by the model) with the `summary()` function, and an AIC value with the `AIC` function. R-squared is a useful metric, but we shouldn't use it to directly compare models, since it will always go up as we add more predictors. We *can* use AIC to compare models though - generally, we prefer models with lower AIC.

```r
btemp_rsq <- summary(btemp_model)$r.sq ## R-squared
btemp_AIC <- AIC(btemp_model) ## AIC
```

Area-Under-the-Curve

One common tool for assessing the performance of a model for classifying binary outcomes is the ROC curtve (or "receiver operator characteristic" curve), which plots the true positive rate (the number of occurrences that the model got right over the total number of observations for which the species occured) against the false positive rate (the number of observations for which the species did not occur, but the model predicted it did, divided by the total number of observations for which the species did *not* occur) as we increase the prediction point. The prediction point is the threshold of the predicted probability at which we decide to classify a site as suitable habitat for the species of interest, and therefore expect it to occur there.

The grater the area under the ROC curve (AUC), the more accurate the model is at predicting presences and absences. A model whose predictions are *always* right has an AUC of 1, while a model whose predictions are always wrong has an AUC of 0 - anything with an AUC above about 0.8 is pretty good, and anything above 0.9 is *excellent*.

We can plot the AUC curve by extracting the fitted values using the `predict` function with `response = TRUE` to indicate that we want predictions as probability of occurrence and not as log-odds of occurrence (the logit scale). Than, we store the fitted probabilities for which our species of interest was observed, and for those where it wasn't observed, and pass those to the `evaluate` function from the `dismo` package:

```r
fitted <- as.numeric(predict(spatiotemp_model, type = "response")) ## fitted probabilities
fitted_p <- fitted[EBS_data$pollock == 1] ## fitted probabilities for observed occurrences
fitted_a <- fitted[EBS_data$pollock == 0] ## fitted probabilities for observed absences
```

```r
ROC <- dismo::evaluate(fitted_p, fitted_a) ## compute ROC curve
plot(ROC, "ROC") ## plot ROC curve
```

We can extract the AUC score with:

```r
ROC@auc
```

Predicting on new data

Finally, once we've selected a model that we feel makes sense and has some decent predictive power, we can apply our model predictions to current, past, or future environmental and/or spatial data (although whether or not it makes sense to extrapolate a model with spatial terms depends on *how* the spatial terms were included in the model).

The dataset `extrapolation_grid.csv` contains depth and bottom temperature for a grid of latitude and longitude points across the Eastern Bering Sea every year between 1982 and 2017, as well as the average julian date of sampling in that year, and the average distance fished overall. Make sure to take a look at this dataset to understand its structure.

```r
extrap_grid <- read.csv("extrapolation_grid.csv")
```

We can predict the probability of occurrence for our model using this data:

```r
extrap_grid$fitted <- predict(spatiotemp_model, newdata = extrap_grid, type = "response")
```

We can plot these predictions using `ggplot2` and the `geom_raster` function, using `facet_wrap` to create a panel for each year. Just for convenience, I've provided a function which will also plot the coastline of Alaska. You can plot model predictions with this function with:

```r
with(extrap_grid, map_EBS_grid(E_km, N_km, fill = fitted, facet = year)) +
  labs(fill = "probability of occurrence")
```

Use these maps to assess how spatial and temporal patterns in the fitted probability of occurrence change as you change the predictors in the model.

Northward Movement

One of the basic expectations we have about marine species ranges under climate change is that they will, on average, move poleward. We can assess how a species range is tracking in multiple ways, but one way is just to calculate a mean latitude for the species range, weighted by probability of occurrence. We can do this for the observed data (weighting each observation by whether or not the species occurs), and for the fitted models (weighting each grid cell by probability of occurrence), and plot them together.

Discussion Question Is your species moving north over time? Does the answer to this question depend on the model, or is it the same regardless of the model you fit? Do some models seem to describe the temporal change in mean latitude better?

```r
### annual mean northings from observed data for pollock
N_obs <- EBS_data %>% group_by(year) %>%
  summarize(mean_N = weighted.mean(N_km, w = pollock))

### annual mean northings from model predictions
N_fit <- extrap_grid %>% group_by(year) %>%
  summarize(mean_N = weighted.mean(N_km, w = fitted))

### plot together
```

```r
N_obs %>% ggplot(aes(year, mean_N)) + geom_point() +
  geom_line(data = N_fit)
```

Saving plots

At the end of this lab, we'll share some of the models and plots we made and our interpretations of them - is the range of the species changing over time, or not? Does the species have an apparent thermal (or depth) niche? Does it seem like it's moving over time in order to track with that niche?

In order to save plots made by ggplot, use the `ggsave` function. The first argument is the file name we want to save to, the second is the plot object, and we can specify other options as follows:

- `height` and `width` arguments for the dimensions of the saved plot - also make sure to include units for the dimensions (e.g., `units = "in"`)
- `dpi` - the resolution of the saved plot. For images with lots of detail, `dpi = 300` or greater is good
- `scale` - if it's hard to make out text on the saved plot, we can make things more readable by lowering the scale below 1, the default - try, for example, 0.8 or 0.9.