Encountering GAMs for modeling encounters

ER Deyle

Fall 2023; Marine Semester Block 3 v2

For these exercises you will need `tidyverse`, `bbmle`, and `mgcv`.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(bbmle)
```

```
## Loading required package: stats4
##
## Attaching package: 'bbmle'
##
## The following object is masked from 'package:dplyr':
##
##     slice
```

```
library(mgcv)
```

```
## Loading required package: nlme
##
## Attaching package: 'nlme'
##
## The following object is masked from 'package:dplyr':
##
##     collapse
##
## This is mgcv 1.9-0. For overview type 'help("mgcv-package")'.
```

Market Squid

Background

Navarro et al. 2018 https://doi.org/10.2983/035.037.0313

Spatial variation in embryo capsule density and location appears dependent on environmental conditions, whereas the temporal pattern of year-round spawning is not. Embryos require [O2] greater than 160 mmol and pH greater than 7.8. Temperature does not appear to be limiting (range: 9.9°C–15.5°C).

Perretti & Sederat 2016 https://doi.org/10.1111/fog.12167

Data

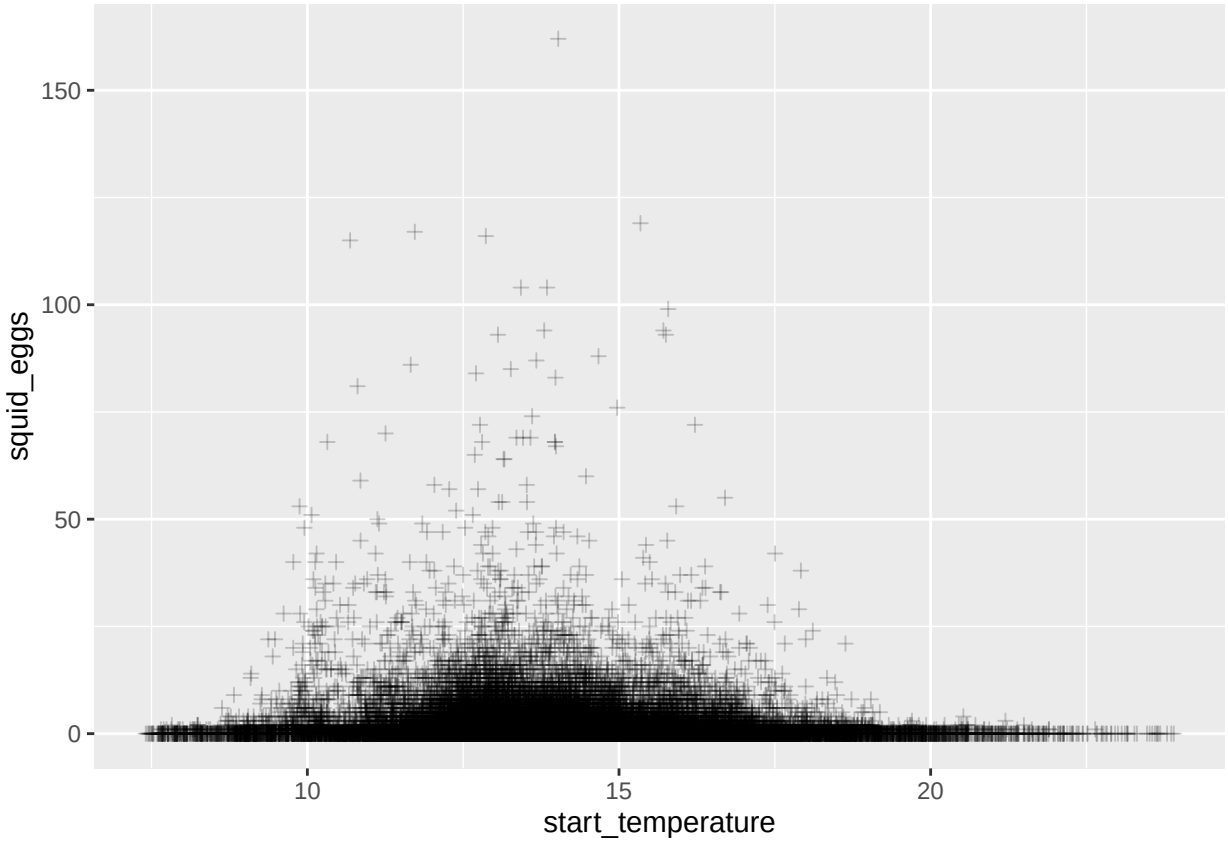
CalCOFI squid egg counts

```
df_CUFES <- read_csv("./data-raw/CalCOFI/erdCalCOFIcufes_4770_0498_b2aa.csv",
                     col_types="nTdddd?Tddddnnnnn")
```

```
## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)
```

```
df_CUFES %>% ggplot(aes(x=start_temperature,y=squid_eggs)) + geom_point(pch=3,alpha=.2)
```

```
## Warning: Removed 9919 rows containing missing values (`geom_point()`).
```



DIGRESSION:

All alternative to a thinplate spline would be a uni-modal function like the Briere function like they use in Shocket et al. 2020:

$$f_B(T) = q \cdot T(T - T_{min})\sqrt{T_{max} - T}$$

or the oppositely skewed

$$f_{iB}(T) = q \cdot T(T_{max} - T)\sqrt{T - T_{min}}$$

However, because of the way logistic regression works, a quadratic response of temperature, i.e. `a0 + a1*T + a2*T^2` behaves quite well. When the quadtric gives very large negative values, this just corresponds to the probability of occurence going to 0.

Modeling options

The Gruss et al 2018 is wicked technical. Equation (1) is really the gist of the modeling, though, where they describe a binomial model for the probability of encounter, η using a thinplate spline of environmental covariates like T along with... a bunch of other stuff. This piece is all we're going to work with for now:

$$\text{logit}(\eta) = s(T)$$

Logistic regression

- The dependent variable (y) can only take values 0 or 1.
- The probability of taking 1 is a function of the independent variable(s) (x).
- Tend to write the probability of the kth observation as $p_k = p(x_k)$.

The likelihood is

$$L = \prod_{1's} p_k \prod_{0's} (1 - p_k)$$

This should look at least vaguely like our scallop dredge modeling.

In a GLM, the form of the linear dependence of the probability on x is mediated through the logistic function.

This should look more than vaguely like our scallop dredge model.

$$p(x) = \text{logistic}((a + bx)) = \frac{e^{(a+bx)}}{1 + e^{(a+bx)}} = \frac{1}{1 + e^{-(a+bx)}}$$

Often this is written using the inverse of the logistic function, the logit function.

$$\text{logit}(p(x)) = \ln\left(\frac{p(x)}{1 - p(x)}\right) = a + bx$$

Analyze data from 1996-2010

```
df_squid_fit <- df_CUFES %>%
  filter(year(time) < 2010) %>%
  mutate(tdegC = 1/2*(start_temperature+stop_temperature)) %>%
  mutate(eta_squid = squid_eggs>0) %>%
  select(eta_squid,tdegC) %>%
  filter(complete.cases(.))
```

```
fit_out_1 <- mle2(eta_squid~dbinom(size=1,prob=1/(1 + exp(-(a1 + a2*tdegC)))),
                  data=df_squid_fit,
                  start=list(a1=0,a2=0))

# fit_out_1 <- mle2(eta_squid~dbinom(size=1,prob=1/(1 + exp(-alpha))),
#                   parameters = list(alpha~tdegC),
#                   data=df_squid_fit,
#                   start=list(alpha=c(0,0)))
```

```
fit_out_2 <- mle2(eta_squid~dbinom(size=1,prob=1/(1 + exp(-(a1 + a2*tdegC + a3*tdegC^2)))),
                  # parameters=list(a1,a2,a3),
                  data=df_squid_fit,
                  start=list(a1=0,a2=0,a3=0))


bbmle::anova(fit_out_1,fit_out_2)
```

```
## Likelihood Ratio Tests
## Model 1: fit_out_1, eta_squid~dbinom(size=1,prob=1/(1+exp(-(a1+a2*tdegC))))
## Model 2: fit_out_2, eta_squid~dbinom(size=1,prob=1/(1+exp(-(a1+a2*tdegC+
##            a3*tdegC^2))))
##   Tot Df Deviance Chisq Df Pr(>Chisq)
## 1      2    23950
## 2      3    23160 789.9  1  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Fit GAM data from 1996-2010

Now we need to do the whole GAM. First take a peek at the manual:

```
?mgcv
```

This unveils a suit of inputs and knobs. First up is `family` (try `?family`). This is which flavor of GLM we're using as the point of departure. For the encounter rate, we've already set this up as `binomial(link="logit")`. I.e. the function linking our `y` data to something we can model in a linear way is the inverse logistic, i.e. the `logit`.

The argument `subset` is potentially useful. We can manipulate the "in-sample" and "out-of-sample" treatment of the data outside the model structure if we want, but `mgcv` gives us the means to do it within the model call too.

Ok, `method`. This is how the software is going to handle the smoothness "knob". Let's look at the Gruss paper.

> We employed the restricted maximum likeli- hood (REML) optimization method (Wood 2011).

Cool. We'll do the same. Here's another piece of the methods they describe:

> We used thin-plate regression splines with shrinkage (bs = "ts" in the specification of the smooth function in "gam" from the "mgcv" library).

What does that mean? Well once we go to actually write the model formula, we're going to use `s()` to specify a smooth term, i.e. a potentially nonlinear response to a given variable. We can look at the manual entry of `s()` but actually that will ultimately point to the following:

```
?smooth.terms
```

> Thin plate regression splines bs="tp". These are low rank isotropic smoothers of any number of covariates. By isotropic is meant that rotation of the covariate co-ordinate system will not change the result of smoothing. By low rank is meant that they have far fewer coefficients than there are data to smooth. They are reduced rank versions of the thin plate splines and use the thin plate spline penalty. They are the default smooth for s terms because there is a defined sense in which they are the optimal smoother of any given basis dimension/rank (Wood, 2003). Thin plate regression splines do not have 'knots' (at least not in any conventional sense): a truncated eigen-decomposition is used to achieve the rank reduction. See tprs for further details. bs="ts" is as "tp" but with a modification to the smoothing penalty, so that the null space is also penalized slightly and the whole term can therefore be shrunk to zero.

Since we're not throwing in tons of possible co-variates, the `bs="ts"` doesn't really have a role. We should stick to `bs="tp"` for now. Also, you should see within this manual a lot of things reminiscent of `smooth.spline`. Some buisness about knots, etc.

Finally, this part is also worth snipping out for the coming projects:

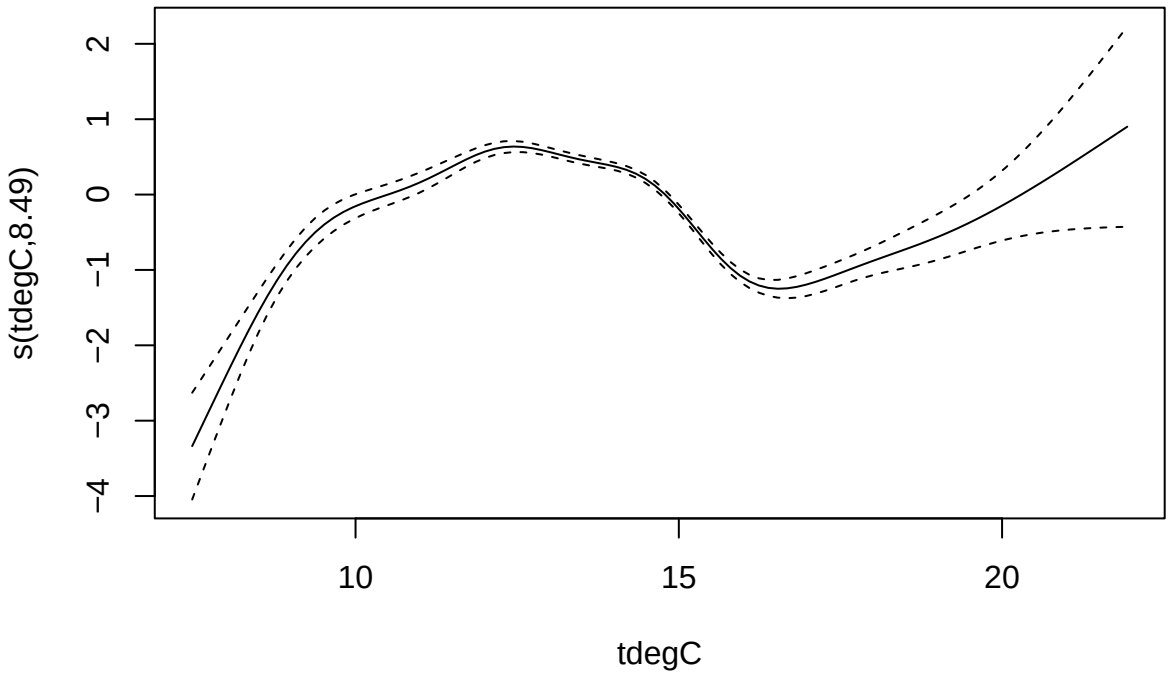
> We evaluated the validity of the fitted binomial GAMs by means of an iterative, cross-validation procedure called "Leave Group Out Cross Validation," in which the set of surveys from the comprehensive survey database for the functional group, species, or stanza of interest was randomly split into training and test data sets (Hastie et al. 2001; Kuhn and Johnson 2013; Gru€ss et al. 2016c); 60% of the data were used for model training and 40% for model evalua- tion. Binomial GAMs were fitted to the training data set using the fitting process described above and then evalu- ated by means of specific performance metrics using the test data set. This procedure was repeated 10 times. Thus, for each individual GAM, 10 models were trained using distinct training data sets, which were then evaluated using distinct test data sets, each corresponding to a specific training data set. Adopting the Leave Group Out Cross Validation procedure allowed us to quantify uncertainty around the performance metrics of GAMs.

This kind of cross-validation is good to do, but distinct from a second challenge we'd like to give these types of modeling approaches: predicting "non-stationary" futures. In that case we are specifically interested in the hardest challenge— predicting forward in time when driving variables are changing with long-term trends.

```
fit_out_GAM <- gam(formula=eta_squid~s(tdegC,bs="tp"),
    data=df_squid_fit,
    family="binomial",
    method = "REML")
```

What exactly did this get us? Well, one thing we can do with a GAM object is plot it and see what the splines look like. In this case there's only one:

```
plot(fit_out_GAM)
```



Check the axes. This is the spline that predicts `logit(eta)`. It says that there is an optimal temperature around 12.5ºC, but also that temperature preference might again start to incease after 16ºC. Is that realistic?

Can we compare this GAM fit to the others? Well, there's a function `AIC` which can gives us the "AIC" for

each of those fits.

```
AIC(fit_out_1,fit_out_2,fit_out_GAM)
```

```
##        AIC       df
## 1 23953.97 2.000000
## 2 23166.07 3.000000
## 3 22700.90 9.724062
```

```
# BIC(fit_out_1,fit_out_2,fit_out_GAM)
```

Collaborate:

Rewind. Our goal is along the lines of "predicting forward in time when driving variables are changing with long-term trends." What about those long-term trends in environment?

Task: Visualize temperature shifts across the full time period

```
# library(sf)
#
# sf_CUFES <- df_CUFES %>%
#   select(time,latitude,longitude,start_temperature,stop_temperature) %>%
#   mutate(month=month(time),year=year(time)) %>%
#   filter(complete.cases(.)) %>%
#   st_as_sf(coords=c("longitude","latitude"))
```

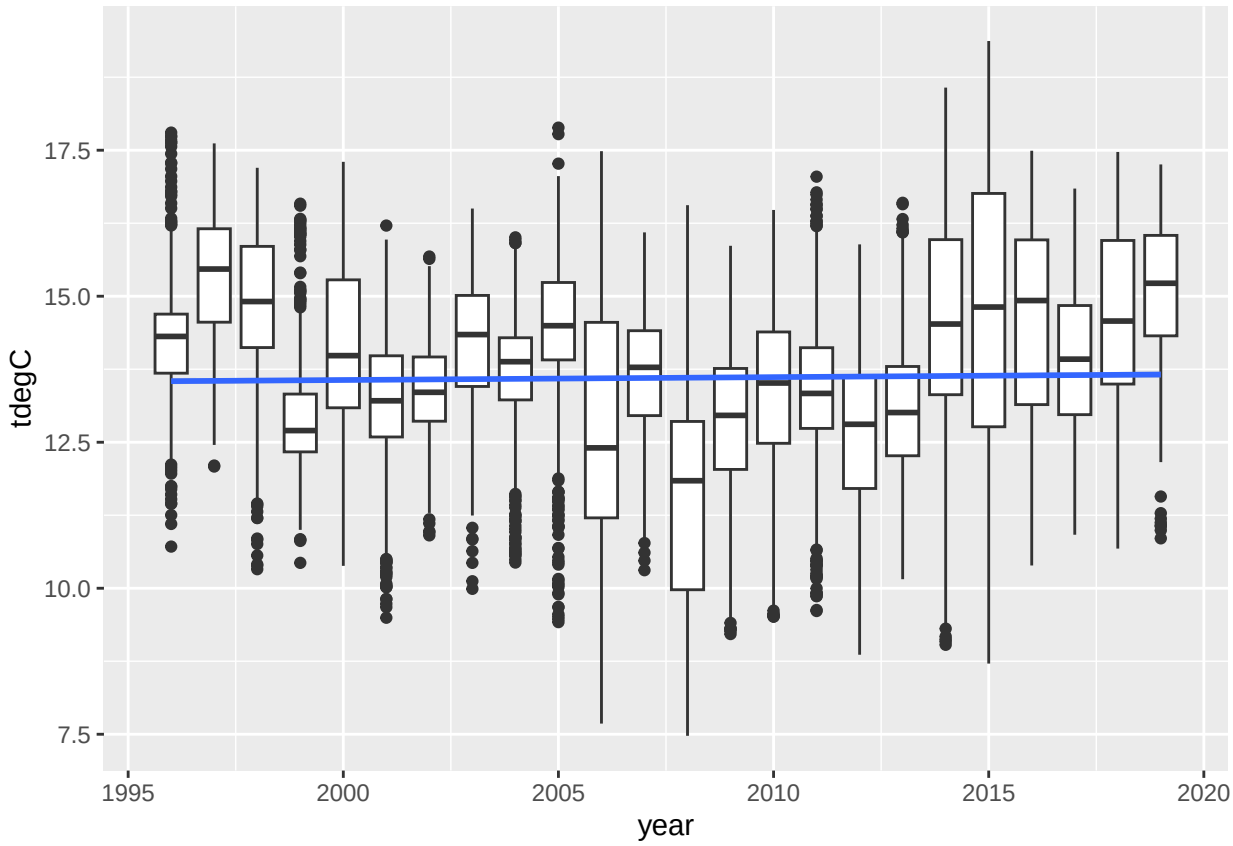
```
df_plot_T_trends <- df_CUFES[-1,] %>%
  mutate(tdegC = 1/2*(start_temperature+stop_temperature)) %>% # get avg temp
  mutate(year = year(time),month = month(time)) %>% # pull out year and month
  select(year,month,tdegC)

df_plot_T_trends  %>%
  filter(month==4) %>% # look at just April when bulk of data are collected
  ggplot(aes(x=year,group=year,y=tdegC)) +
  geom_boxplot() +
  geom_smooth(aes(group=NULL),method="lm") # add a trend line?
```

```
## Warning: Removed 56 rows containing non-finite values (`stat_boxplot()`).
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 56 rows containing non-finite values (`stat_smooth()`).
```



Now, we can use the `predict` function with our model fits to look at the accuracy out-of-sample.

Task: Test predictive power of GLM, "GQM", and GAM from 2010-2019

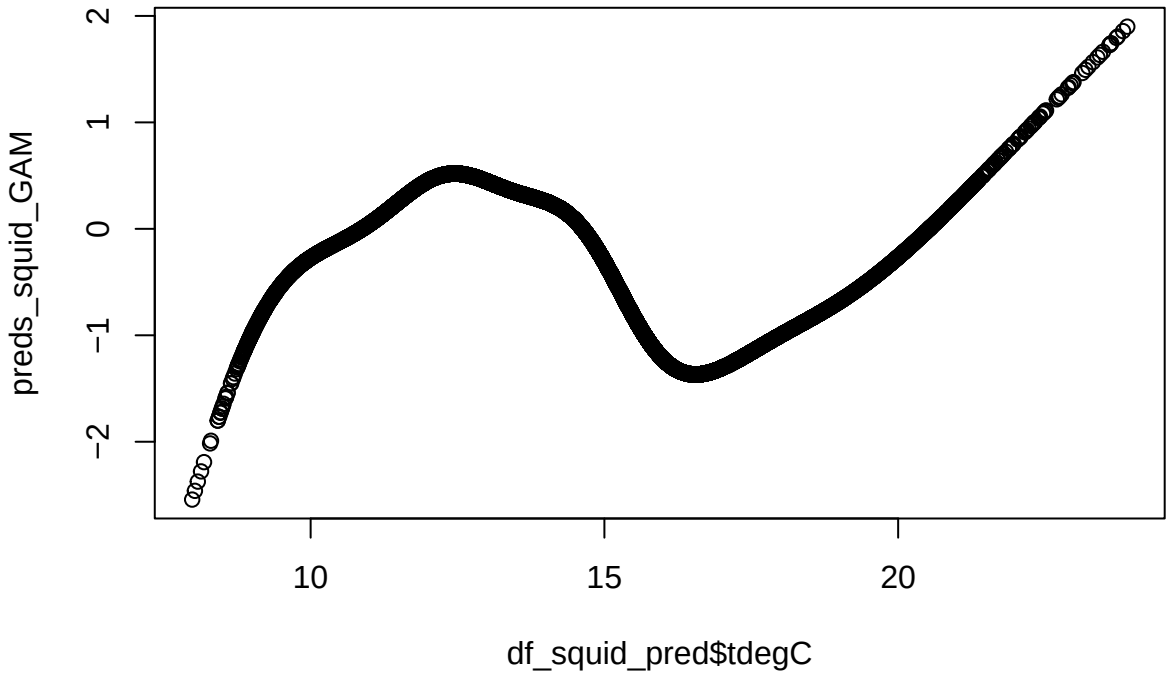
First we need to figure out how to use `predict` to get the GAMs predictions.

```
df_squid_pred <- df_CUFES %>%
  filter(year(time) >= 2010) %>%
  mutate(tdegC = 1/2*(start_temperature+stop_temperature)) %>%
  mutate(eta_squid = squid_eggs>0) %>%
  select(eta_squid,tdegC) %>%
  filter(complete.cases(.))


preds_squid_GAM <- predict(fit_out_GAM,df_squid_pred)
# type="response"
# type="link"
```

What did it actually predict?

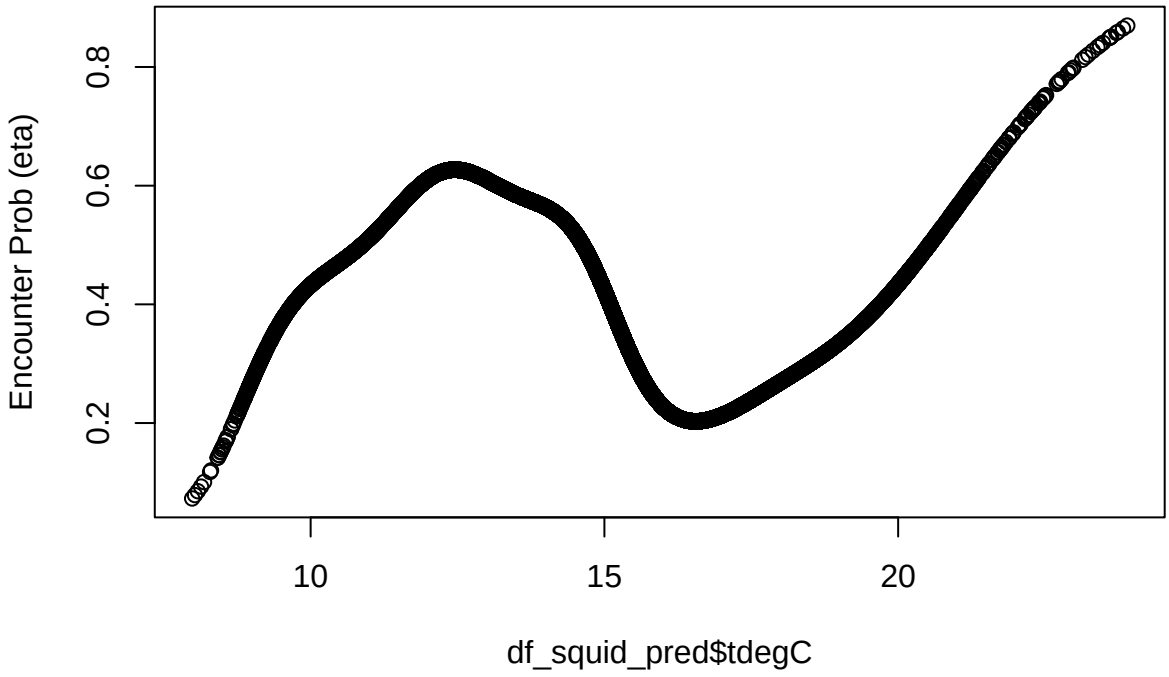
```
plot(df_squid_pred$tdegC,preds_squid_GAM)
```



Remember, however, that the GAM is modeling `logit(eta)`, i.e. the log-odds ratio of observing one or more eggs to not observing one or more eggs. We need to apply the inverse of the logit function to get back to just the "eta".

```
preds_squid_GAM <- predict(fit_out_GAM,df_squid_pred,type="response")

plot(df_squid_pred$tdegC,preds_squid_GAM,ylab="Encounter Prob (eta)")
```



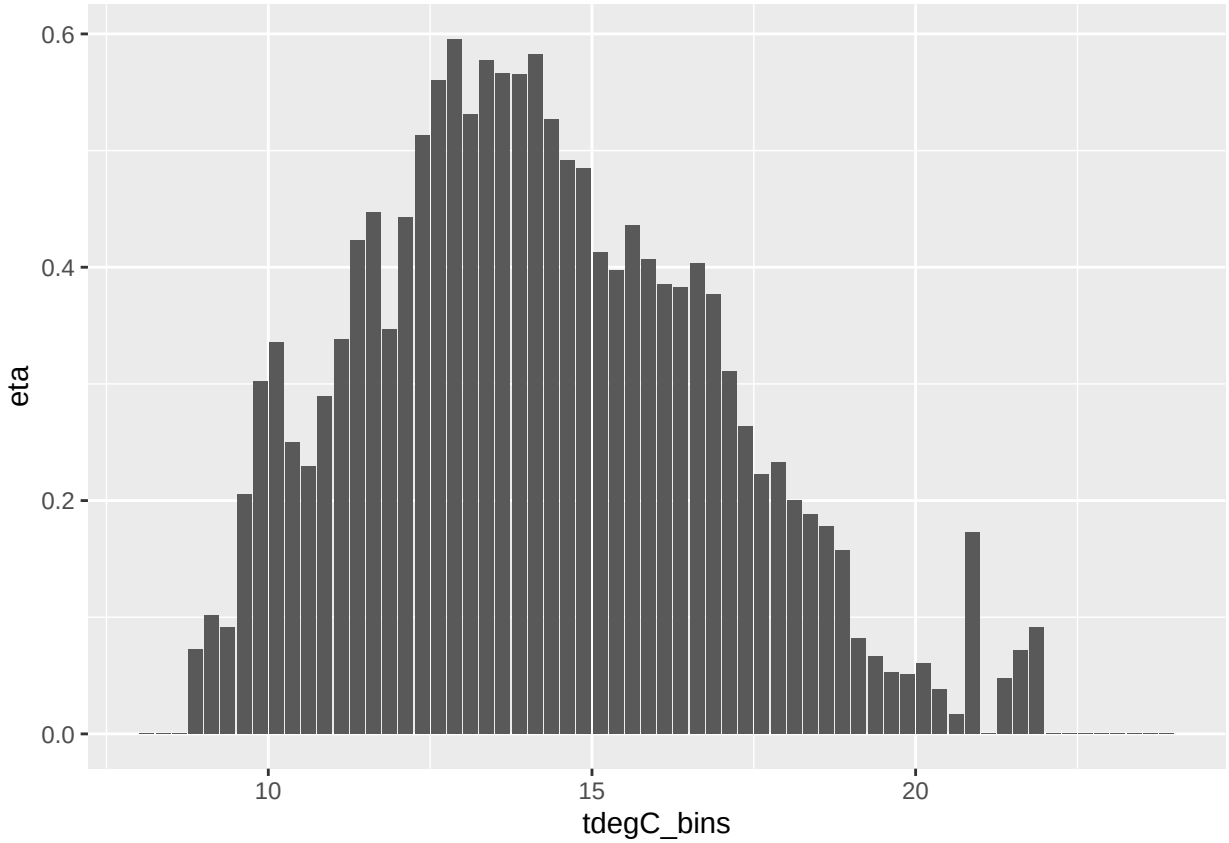
This isn't exactly straight forward to visualize the predictions versus observations, though. Instead, we can turn our presence/absence into bins.

```
bin_edges <- seq(8,24,by=0.25)
bin_mids <- (bin_edges[1:64] + bin_edges[2:65])/2

binned_squid_pred <- df_squid_pred %>%
  mutate( tdegC_bins = cut( tdegC, breaks = seq(8,24,by=0.25) ) ) %>%
  group_by(tdegC_bins) %>%
  summarise(eta = sum(eta_squid) / n()) %>%
  mutate(tdegC_bins=(bin_mids[tdegC_bins]))

binned_squid_pred %>%
  ggplot(aes(x=tdegC_bins,y=eta)) + geom_col()
```

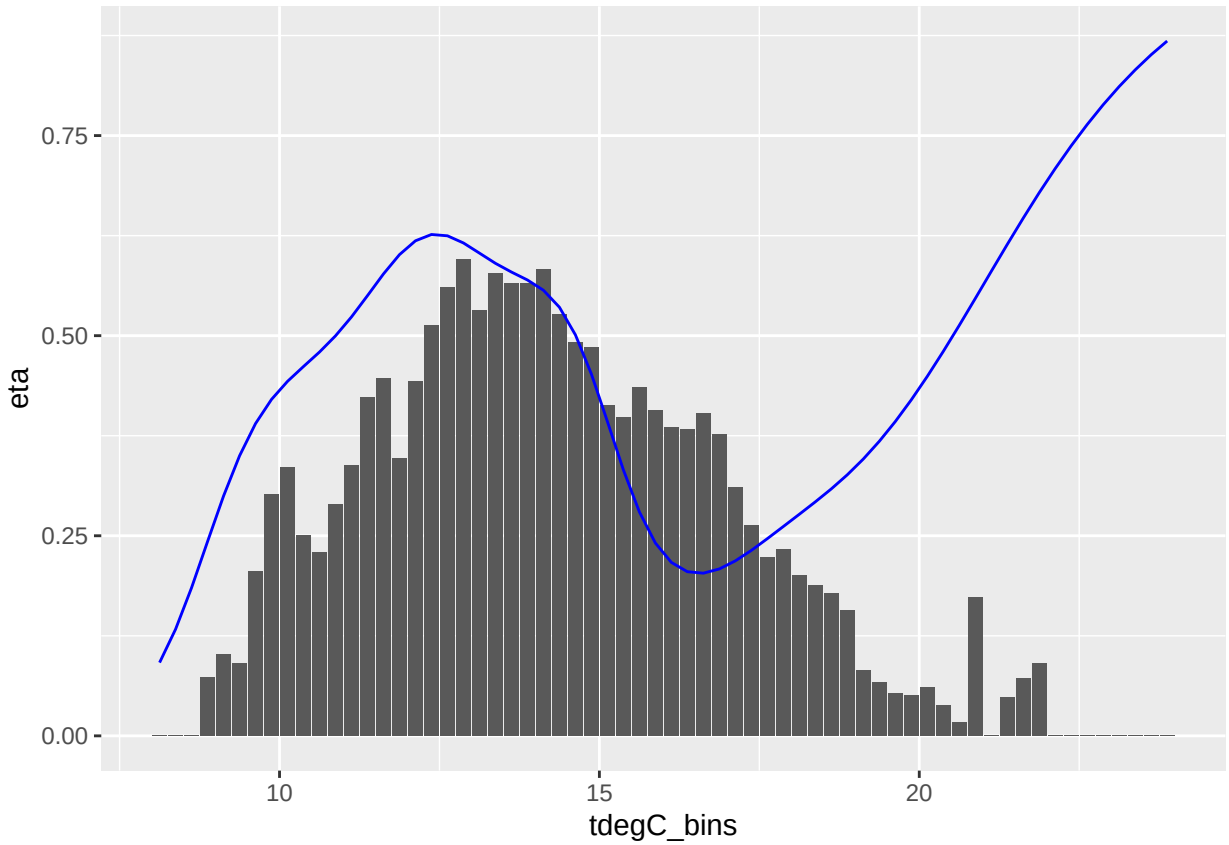
```
## Warning: Removed 1 rows containing missing values (`position_stack()`).
```



Then we need to use the GAM to predict the frequency at the midpoint of each bin.

```
binned_squid_pred <- binned_squid_pred %>%
  mutate(eta_GAM=predict(fit_out_GAM,
                         data.frame(tdegC=tdegC_bins),type="response"))

binned_squid_pred %>%
  ggplot(aes(x=tdegC_bins,y=eta)) +
  geom_col() +
  geom_line(aes(y=eta_GAM),color="blue")
```



Of course, this wasn't really taking full advantage of the "GAM" framework; we only considered a single environmental driver.

Optional 1: Include Salinity and Temperature in additive ways

Additional Options

Optional 2: Try the same thing but using the original count data and doing Poisson regression.

Optional 2: Try the same thing but using one of the other taxa in the data set.

Optional 3: Try the same thing but using data from the South Florida Reef Visual Census.

https://myfwc.com/research/saltwater/reef-fish/monitoring/fim-fl-keys-visual-sampling/

There is an R package for that data set! https://github.com/jeremiaheb/rvc.

```
# fk11_12 = getRvcData(years = 2011:2012, regions = "FLA KEYS")

fk01_20 = getRvcData(years = 2001:2020, regions = "FLA KEYS")
filter(fk01_20$taxonomic_data,COMNAME == "stoplight parrotfish")
df_stop_par <- fk01_20$sample_data %>% filter(SPECIES_CD == "SPA VIRI")
```

Unfortunately, despite there being discussion of temperature measurements, none of the data frames seem to contain it. However, there's times and locations which means we can pull temperature data from other data sets.