Random Factors and AUtocorrelation in Plankton Models

Curtis C. Bohlen, Casco Bay Estuary Partnership

4/04/2022

Contents

Introduction

This notebook is a follow up on earlier efforts to explore further changes in data analysis.

As before, this Notebook looks at:

1. Non-linear fits between zooplankton community metrics and possible environmental drivers, and
2. Examination of responses of one individual species to those same drivers.

I've trimmed down the analysis workflow, since I looked at the data distributions, etc. previously, but the major steps remain the same.

The major changes in this notebook address spatial and temporal structure. In particular, I want to consider:

1. Seasonal patterns
2. Temporal correlation
3. Spatial patterns
4. Spatial correlation

The available data is not collected on a regular grid in either space or time, so we can not use ARIMA models or their cousins directly to model autocorrelation (as ARIMA models are defined on a regular time series). But we can employ several different approaches to think through related questions:

1. ANOVA or ANCOVA models, and especially "hierarchical" or "mixed effects" models, fit groups that imply observations belong in "correlation classes" defined by the random and fixed terms in the model.
2. We can look at formal autocorrelation structure using either generalized least squares (for linear models) or GAMM models (for other models). These correlation structures can be fit to sample order (appropriate for regular time series) or to quantitative metrics, such as date and river mile.

The difference between the two is subtle, but potentially important. Method (1) says that observations collected at the same time (or in the same CLASS) are correlated, but it imposes no structure on the correlations between classes. Method (2) imposes structure based on some sort of a “distance” metric, typically based on time or geographic distance.

My conclusion is that autocorrelation structures don’t add much to the models, but that hierarchical models and explicit modeling of correlation groups using hierarchical models are very important.

Data Preparation

Load Libraries

```r
library(tidyverse)
library(readxl)
library(nlme)      # for generalized least squares and
                   # convenient access to correlations structures
library(lme4)
library(mgcv)      # for GAMM models
library(emmeans)   # For extracting useful "marginal" model summaries

library(ncf)      # one simple implementation of spatial correlograms

theme_set(theme_classic())
```

Folder References

```r
data_folder <- "Original_Data"
```

Load Data

```r
filename.in <- "penob.station.data EA 3.12.20.xlsx"
file_path <- file.path(data_folder, filename.in)
station_data <- read_excel(file_path,
                           sheet="Final", col_types = c("skip", "date",
                                                 "numeric", "text", "numeric",
                                                 "text", "skip", "skip",
                                                 "skip",
                                                 rep("numeric", 10),
                                                 "text",
                                                 rep("numeric", 47),
                                                 "text",
                                                 rep("numeric", 12))) %>%
  rename_with(~ gsub(" ", "_", .x)) %>%
  rename_with(~ gsub("\\.", "_", .x)) %>%
  rename_with(~ gsub("\\?", "", .x)) %>%
  rename_with(~ gsub("%", "pct", .x)) %>%
  rename_with(~ gsub("_Abundance", "", .x)) %>%
  filter(! is.na(date))
#> New names:
#> * `` -> ...61
```

Station names are arbitrary, and Erin previously expressed interest in renaming them from Stations 2, 4, 5 and 8 to Stations 1,2,3,and 4.

The `factor()` function by default sorts levels before assigning numeric codes, so a convenient way to replace the existing station codes with sequential numbers is to create a factor and extract the numeric indicator values with `as.numeric()`.

```
station_data <- station_data %>%
  mutate(station = factor(as.numeric(factor(station))))
head(station_data)
#> # A tibble: 6 x 76
#>   date                 year month month_num season riv_km station station_num
#>   <dttm>              <dbl> <chr>     <dbl> <chr>   <dbl> <fct>         <dbl>
#> 1 2013-05-28 00:00:00  2013 May           5 Spring  22.6  1                 1
#> 2 2013-05-28 00:00:00  2013 May           5 Spring  13.9  2                 2
#> 3 2013-05-28 00:00:00  2013 May           5 Spring   8.12 3                 3
#> 4 2013-05-28 00:00:00  2013 May           5 Spring   2.78 4                 4
#> 5 2013-07-25 00:00:00  2013 July          7 Summer  22.6  1                 1
#> 6 2013-07-25 00:00:00  2013 July          7 Summer  13.9  2                 2
#> # ... with 68 more variables: depth <dbl>, discharge_week_cftpersec <dbl>,
#> #   discharg_day <dbl>, discharge_week_max <dbl>, tide_height <dbl>,
#> #   Full_Moon <dbl>, Abs_Moon <dbl>, Spring_or_Neap <chr>, ave_temp_c <dbl>,
#> #   ave_sal_psu <dbl>, ave_turb_ntu <dbl>, ave_do_mgperl <dbl>,
#> #   ave_DO_Saturation <dbl>, ave_chl_microgperl <dbl>, sur_temp <dbl>,
#> #   sur_sal <dbl>, sur_turb <dbl>, sur_do <dbl>, sur_chl <dbl>, bot_temp <dbl>,
#> #   bot_sal <dbl>, bot_turb <dbl>, bot_do <dbl>, bot_chl <dbl>, ...
```

Subsetting to Desired Data Columns

I base selection of predictor variables here on the ones used in the manuscript.

```
base_data <- station_data %>%
  rename(Date = date,
         Station = station,
         Year = year) %>%
  select(-c(month, month_num)) %>%
  mutate(Month = factor(as.numeric(format(Date, format = '%m')),
                                                levels = 1:12,
                                                labels = month.abb),
         DOY = as.numeric(format(Date,format = '%j')),
         season = factor(season, levels = c('Spring', 'Summer', 'Fall')),
         Yearf = factor(Year)) %>%
  rename(Season = season,
         Temp = ave_temp_c,
         Sal = ave_sal_psu,
         Turb = sur_turb,
         AvgTurb = ave_turb_ntu,
         DOsat = ave_DO_Saturation,
         Chl = ave_chl_microgperl,
         RH = Herring
         ) %>%
  select(Date, Station, Year, Yearf, Month, Season, DOY, riv_km, Temp, Sal, Turb, AvgTurb,
         DOsat, Chl, RH,
         combined_density,H, SEI,
         Acartia, Balanus, Eurytemora, Polychaete, Pseudocal, Temora) %>%
  arrange(Date, Station)
head(base_data)
#> # A tibble: 6 x 24
#>   Date                Station  Year Yearf Month Season   DOY riv_km  Temp
#>   <dttm>              <fct>   <dbl> <fct> <fct> <fct>  <dbl>  <dbl> <dbl>
#> 1 2013-05-28 00:00:00 1        2013 2013  May   Spring   148  22.6  11.7 
#> 2 2013-05-28 00:00:00 2        2013 2013  May   Spring   148  13.9   9.40
#> 3 2013-05-28 00:00:00 3        2013 2013  May   Spring   148   8.12  6.97
#> 4 2013-05-28 00:00:00 4        2013 2013  May   Spring   148   2.78  9.51
#> 5 2013-07-25 00:00:00 1        2013 2013  Jul   Summer   206  22.6  18.5 
#> 6 2013-07-25 00:00:00 2        2013 2013  Jul   Summer   206  13.9  13.6 
#> # ... with 15 more variables: Sal <dbl>, Turb <dbl>, AvgTurb <dbl>,
#> #   DOsat <dbl>, Chl <dbl>, RH <dbl>, combined_density <dbl>, H <dbl>,
#> #   SEI <dbl>, Acartia <dbl>, Balanus <dbl>, Eurytemora <dbl>,
#> #   Polychaete <dbl>, Pseudocal <dbl>, Temora <dbl>
```

```
rm(station_data)
```

Add Transformed Predictors

We can treat the sampling history as “spring”, “summer” and “fall” observations each year from 2013 through 2017. This breaks the temporal pattern down into integer valued time, generating a “quasi regular” time series, and allowing us to simplify the analysis of temporal autocorrelation. The “real world” time difference across the winter is longer that between seasons, but I could not find a ready way to address that.

We need both the numerical sequence and a factor later, for different purposes.

```
base_data <- base_data %>%
  mutate(sample_seq = as.numeric(Season) + (Year-2013)*3,
         sample_event = factor(sample_seq))
```

Data Without Low Salinity Observations

```
low_sample <- which(base_data$Sal <= 5)
smaller_data <- base_data[-low_sample,]
```

Examining Autocorrelation

The form of autocorrelation structures is not immediately obvious, but there are common choices, which differ for temporal and spatial autocorrelation models. We have relatively few sample dates, and relatively few sample locations, so there is likely no way to formally test “goodness of fit” for different correlations structures.

For temporal autocorrelation structures, I'll examine a "conventional" "AR1" structure, which corresponds to an autoregressive model with order 1. It's not that I think this model is in any way "correct", but that it's a first order assumption. It basically says observations close in time are likely to more similar than observations further apart in time. The expected degree of correlation drops off fast (exponentially) with time.

I'm less certain where to start with the spatial autocorrelation. This is especially tricky, as we have only the "river mile" metric or the choice of four stations to work with (inducing a rank ordering of proximity). There's just not a lot of info to work with there.

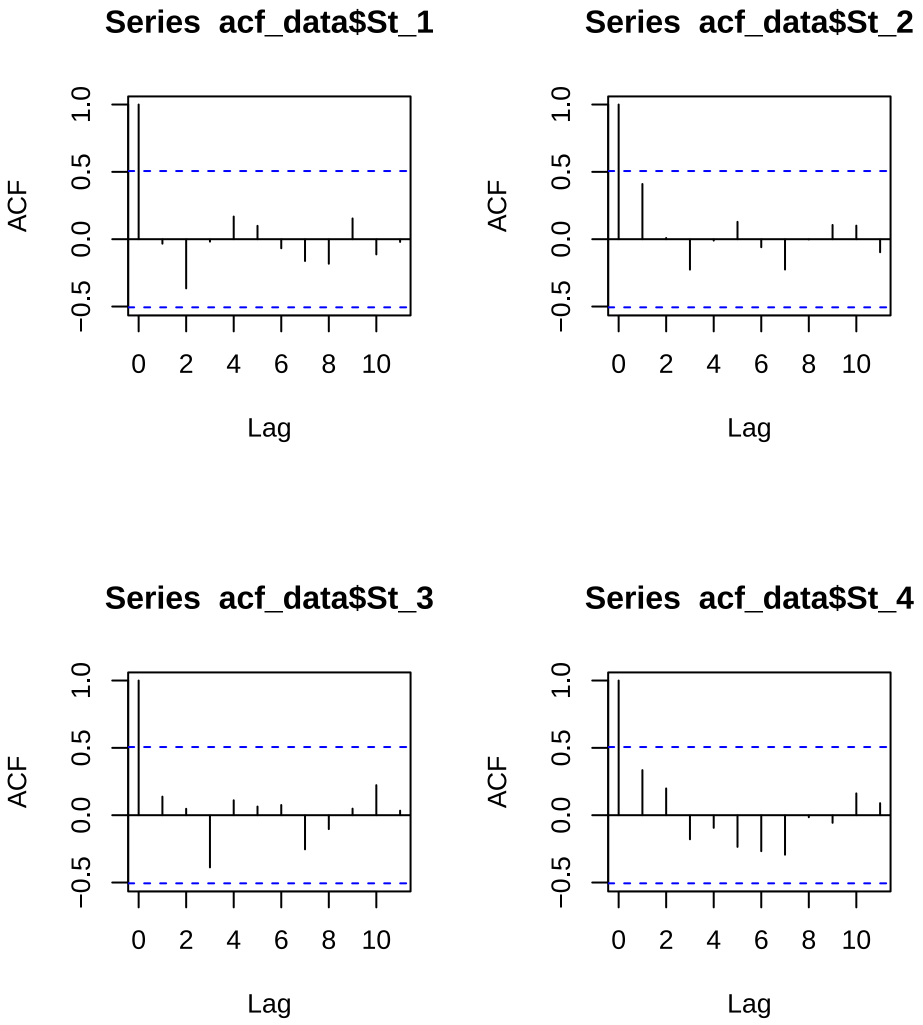
I plan to walk through graphic summary of the autocorrelation structure of the data, then take it step by step developing increasingly complex models and reviewing the autocorrelation structure of model residuals.

Temporal Autocorrelation

I start with temporal autocorrelation analysis. *A priori*, I doubt this is going to be very helpful. We have only 15 sampling events. . . .

Raw Data

```r
acf_data <- base_data %>%
  select(sample_event, Station, combined_density) %>%
  group_by(sample_event, Station) %>%
  mutate(combined_density = log(combined_density)) %>%
  pivot_wider(id_cols = c(sample_event, Station),
              values_from = 'combined_density',
              names_from = 'Station',
              names_prefix = 'St_' )

oldpar <- par(mfrow = c(2,2), mar = c(5,4, 6,2) + 0.1)
acf(acf_data$St_1)
acf(acf_data$St_2)
acf(acf_data$St_3)
acf(acf_data$St_4)
```



```
par(oldpar)
```

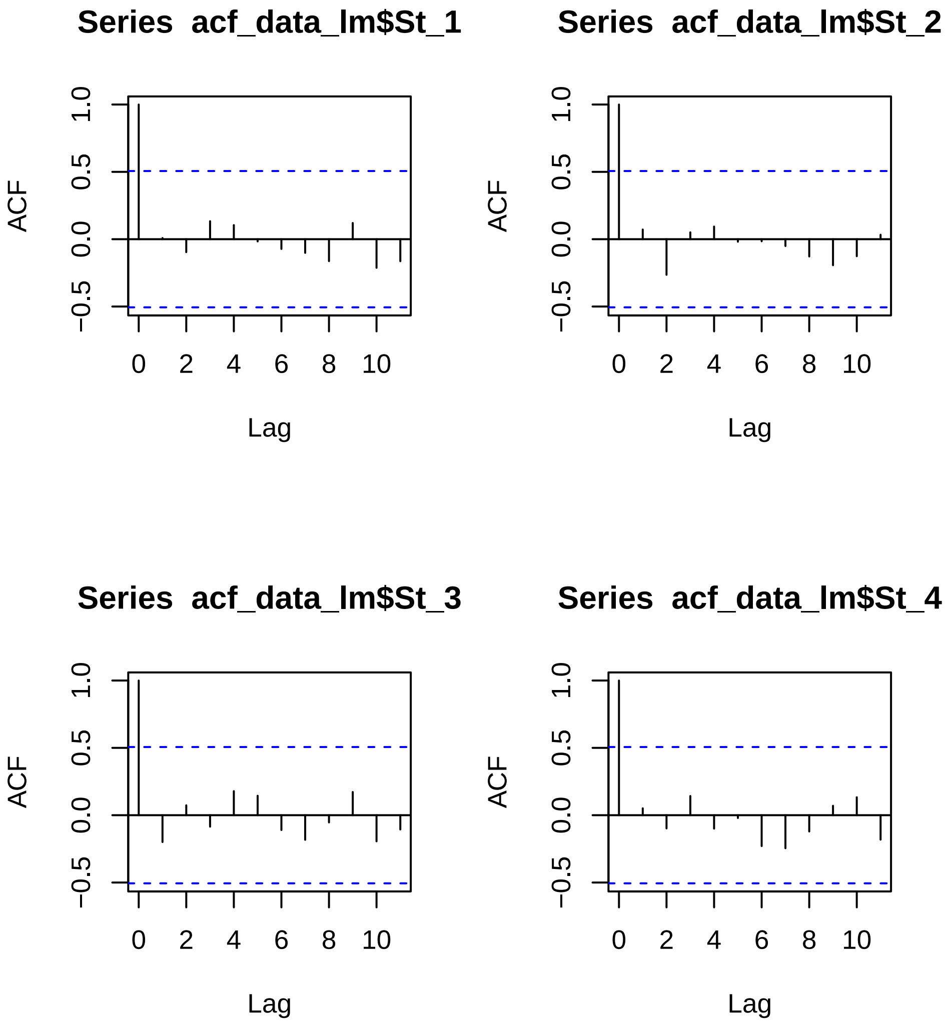
So, raw autocorrelations on individual stations don't reach statistical significance, but since the sample is so small (15 observations in time), that alone does not mean much. Note that the estimated autocorrelations at stations 2 and 4 show the kind of gradual decline one would expect if there is some autocorrelation, but any correlation is swamped by other sources of variability. (The time series of average densities across all four Stations DOES show some autocorrelation).

Linear Model

We want to look at autocorrelations in residuals after fitting likely predictors.

For simplicity, I'm sticking with simple log-linear models that fits all of the spatial and temporal factors that I use in later models. That imposes the same correlation structure as induced in the later models. Here, however, we don't fit any quantitative predictors. We are actually not inerested in statistical significnace here, we only want to loo kat teh structure of the residuals.

```r
the_lm <- lm(log(combined_density) ~ Station + Yearf + Season + sample_event,
             data = base_data)

r <- resid(the_lm)
p <- predict(the_lm)
r_data <- base_data %>%
  select(Yearf, Month, sample_event, Station, combined_density) %>%
  mutate(resid = r)
acf_data_lm <- r_data %>%
  pivot_wider(id_cols = c(sample_event, Station),
              values_from = resid,
              names_from = 'Station',
              names_prefix = 'St_' )

oldpar <- par(mfrow = c(2,2), mar = c(5,4, 6,2) + 0.1)
acf(acf_data_lm$St_1)
acf(acf_data_lm$St_2)
acf(acf_data_lm$St_3)
acf(acf_data_lm$St_4)
```



```
par(oldpar)
```

That has significantly reduced the autocorrelations. By fitting several models, I learned that the `Yearf` term is most important, but the `Season` term has an effect, reducing correlation of residuals at lags 1 and 2.

based on that analysis, would not immediately think that we need to fit a temporal autocorrelation structure to these data, but there are other indications that there may stil lbe som eremaining serial autocorrelation, just not very strong autocorrelation.

Generalized Least Squares Model

What I do here is fit a model that includes autocorrelation, just so I can see how large the autocorrelation term is after we have fit all of the spatial and temporal factors as predictors.

Here, I estimate temporal autocorrelation based on the sample sequence. I had trouble fitting models that would estimate autocorrelation based on dates or another quantitative metric of temporal proximity.

I can't fit `sample_event` here, as it produces a signular model, and `gls()` does not have a clear fallback strategy for singular models the way `lm()` does.

```
the_gls_order<- gls(log(combined_density) ~ Station + Yearf +
                       Season,
               correlation = corAR1(form = ~ sample_seq | Station),
               data = base_data)
```

```
the_gls_order$modelStruct$corStruct
#> Correlation structure of class corAR1 representing
#>       Phi
#> 0.1552237
```

So, when I do fit an autocorrelation term, the model fits a low value for the serial autocorrelation coefficient.

I don't know of a good method for deciding whether it is important to include serial autocorrelation i, a statistical model, although I'm fairly sure they exist. But based on the previous `acf()` plots,a nd this estimate of serial autocorrelaiton, I don't think it's important in this case.

Conclusions

It's important to fit either random or fixed effects for year, and probably for season in any model. Including `sample_event` may or may not be important. It appears there is little value to fitting models with formal temporal autocorrelation structure.

Spatial Autocorrelation

I decided to explore spatial autocorrelation using a "spatial correlogram". This is a graphic that depicts correlations at different distances, classically in either geographic space or as distances along some a connected graph.

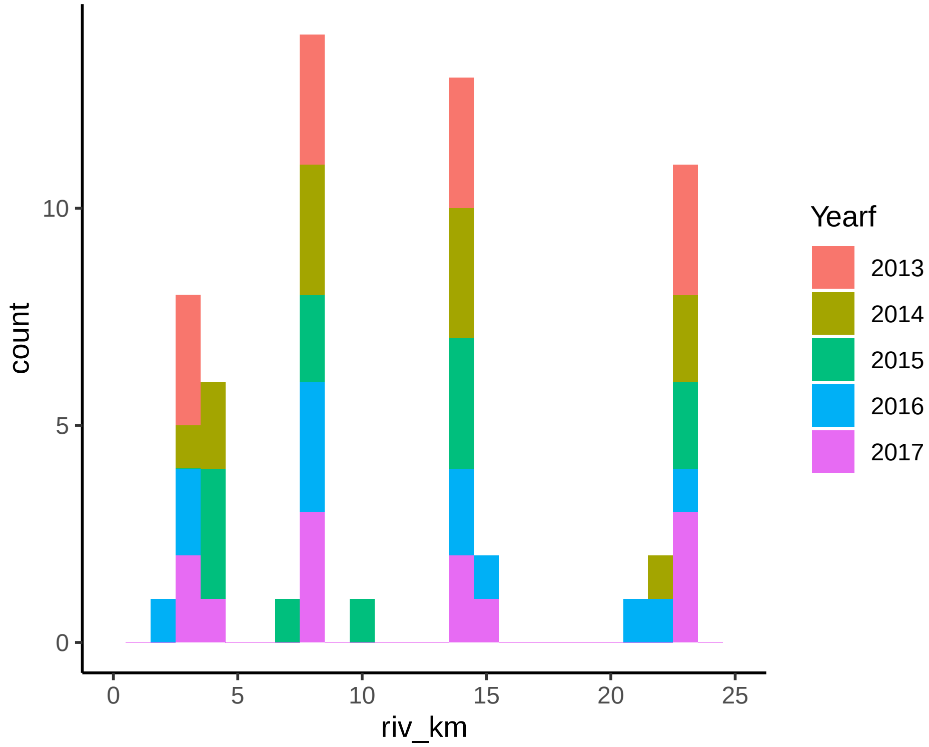
I took this approach because I'm fairly familiar with it. Poking around online, it looks like there are some more up to date methods that might be more useful, but I'm not sure we have enough spatial data to do very much with spatial autocorrelations, so this is a good start.

Again, we have only four sample stations, so I would be surprised if we have enough data for these comparisons to be very informative, but I proceed anyway.

Our best measure of physical proximity is river kilometer, which varies over a range of about 20. We have "distances" between observations from zero to about 20 kilometers.

Here's a histogram of river kilometer values:

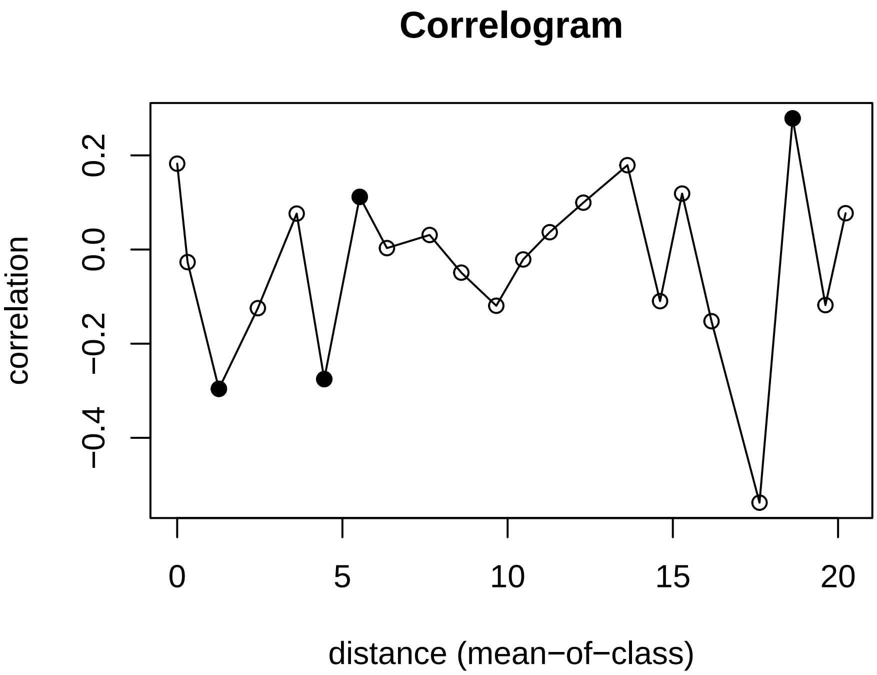
```
ggplot(base_data, aes(riv_km)) +
  geom_histogram(aes(fill = Yearf), binwidth = 1) +
  xlim(0,25)
#> Warning: Removed 10 rows containing missing values (geom_bar).
```



Raw Data

```
ncf.cor <- correlog(base_data$riv_km, rep(1, 15*4),
                    log(base_data$combined_density),
                    increment = 1)
#> 100  of  1000 200  of  1000 300  of  1000 400  of  1000 500  of  1000 600  of  1000 700  of  1000 80

plot(ncf.cor)
```

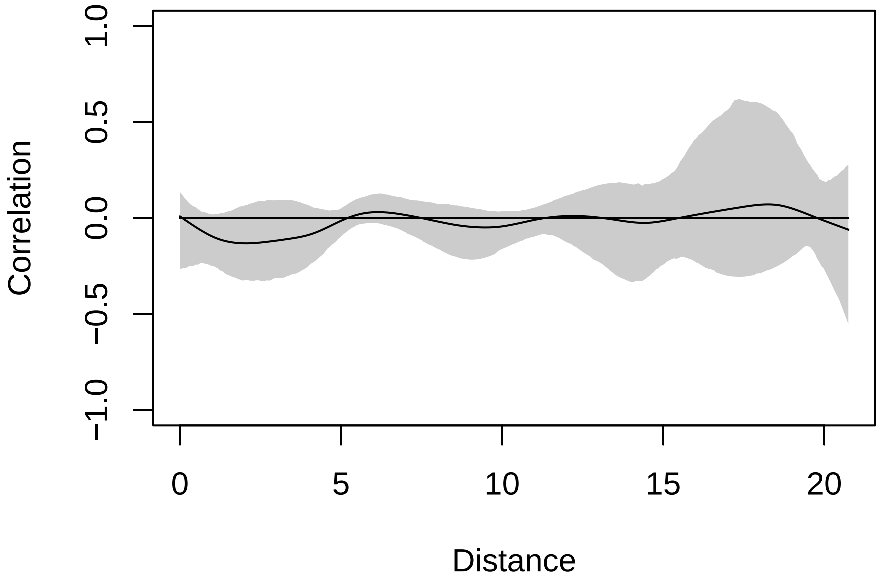
River kilometer



There is no evidence of spatial correlation in that plot. Values are mostly smal and jump back and forth from positive to negative, suggesting they are largely due to chance.

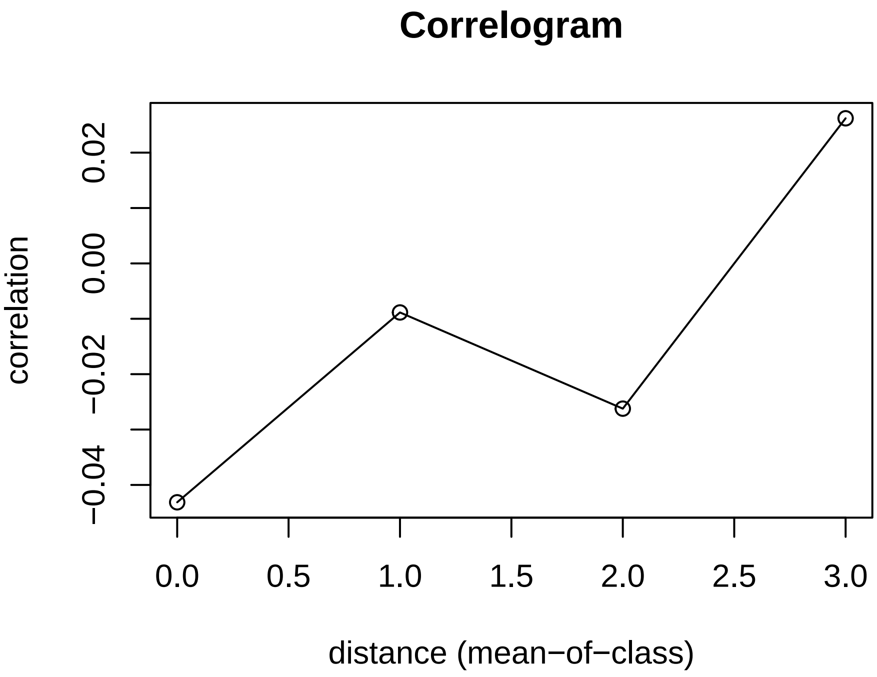
Looked at in a slightly different way:

```
ncf.spline <- spline.correlog(base_data$riv_km, rep(1, 15*4),
                              log(base_data$combined_density))
#> 100  of  1000 200  of  1000 300  of  1000 400  of  1000 500  of  1000 600  of  1000 700  of  1000 80

plot(ncf.spline)
```



We can also look at a correlogram by Station, but here we have only four groups, so there's not much to go on.

```
ncf.cor <- correlog(as.numeric(base_data$Station), rep(1, 15*4),
                    log(base_data$combined_density),
                    increment = 1)
#> 100  of  1000 200  of  1000 300  of  1000 400  of  1000 500  of  1000 600  of  1000 700  of  1000 80

plot(ncf.cor)
```



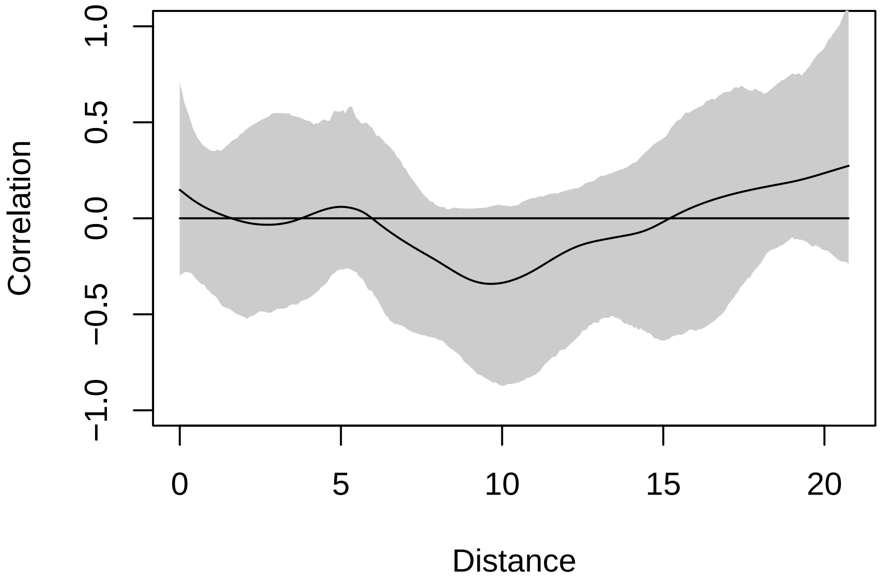
The estimated correlation coefficients are tiny. There is no evidence here that we should fit a spatial autocorrelation structure either.

Linear Model

Just as before, we may want to look at whether there is any clear spatial structure to model residuals after we fit our linear model.

```
r <- resid(the_lm)
p <- predict(the_lm)
r_data <- base_data %>%
  select(Yearf, Month, sample_event, Station, riv_km, combined_density) %>%
  mutate(resid = r)
```

```
ncf.cspline <- spline.correlog(r_data$riv_km, rep(1, 15*4),
                               log(r_data$resid), na.rm = TRUE)
#> Warning in log(r_data$resid): NaNs produced
#> Warning in spline.correlog(r_data$riv_km, rep(1, 15 * 4), log(r_data$resid), :
#> Missing values exist; Pairwise deletion will be used
#> 100  of  1000 200  of  1000 300  of  1000 400  of  1000 500  of  1000 600  of  1000 700  of  1000 80

plot(ncf.cspline)
```



Smoothed estimates of spatial autocorrelation are moderate, with a wide error band. The largest autocorrelation estimates are negative, at intermediate distances, which makes relatively little sense, from a scientific perspective.

I'm not convinced we have spatial correlations structure to model, but as for temporal autocorrelation, I don't know of a simple formal test that is likely to be informative with our relatively small sample size and few geographic locations.

Conclusion

We don't need to fit a spatial autocorrelation term, although it would not be unreasonable to look at spatial corellograms of the residual from or final models to check.

Modeling Strategy

Overall, I'll continue to mimic the linear models from the manuscript, only using GAMMs instead of linear models.

I'm shooting for a consistent model structure that we can use over and over again for each of the responses of interest. I poke at various models fairly hard based on the `combined_density` values, but after I've selected my prefered model structure for that response, I just go ahead and use the same model for each of the other responses of interest.

I've made a few other changes compared to the linear analyses in the original manuscript:

1. I transformed some predictor variables. This was to ease model fitting, but may not be the right call if the transformed variables make no sense, are hard to explain to readers, or or if there are scientific reasons to not look at transformed predictor variables.

2. I have shifted to ‘GAMM’ models These allow fitting fairly arbitrary smooth relationships between predictors and response variables. GAMMs allow use of “mixed models” (random factors) and auto-correlation structures, which we need here, so we can’t get away with just using GAMs.
3. I have added a term for Season into the models. These models thus have both Location in the estuary – via the Station factor – and time of year – via the `Season` factor included in the model. These are both classification variables.
4. In the interest of documenting alternative models, I have fit models both including an not including the extreme low salinity samples.

Mixed Models

In mixed models, we have the opportunity to address certain factors in either the fixed effects or the random effects components of the model.

The model will fit a series of indicator variables for different levels of a factor included in the fixed effects, while for a random effect, the model will fit only an estimate of the associated “variance component” or uncertainty associated with that factor.

Typically, we fit “random effects” to represent groups of observations that are all similar or related due to something about the sampling design that we are willing to consider “random” or unimportant to understand in detail. This might include the subject in a repeated measures design, the plot in an agricultural split plot design, or the sampling date in a long-term study. Random factors allow for correlations among observations within a group, without affecting the quantitative effects of primary interest.

In our context, we need to represent BOTH sample locations and time in the model somehow.

Each station was sampled repeatedly. Formally, this is a “repeated measures” analysis. That can be modeled either as a hierarchical model (with Station as a “Random Factor”), or by fitting Stations as a factor in the model. We can’t leave station out of the model entirely. I prefer to include Station as a fixed effect, because we may want to interpret differences between stations with some sort of causal explanation.

We face a similar choices regarding how to include temporal terms. From a scientific perspective, we are probably most interested in seasonal patterns. I decided to formally fit `Season` as a fixed effect in these models, but most temporal variables are of little interest, so they should be included in the models as random factors.

Selection of Predictors

The following models use the same predictor variables, although I vary the structure of the temporal random structure from model to model.

In particular, I **still** have not tried to fit any interaction terms between predictors, although from a scientific perspective, interactions might be enlightening.

I have transformed several predictors to ensure predictors are more evenly distributed across the model space. This is for purely statistical reasons, and one could argue on scientific grounds against any of these choices.

GAM models are built on a combination of zero or more “linear” predictors and one or more “smooth” predictors. GAMM models are similar, but can include a “random effects” structure or a correlations structure.

For all of the models that follow, I use the the same suite of predictors, although I will explore different ways of fitting random effects and covariance structures.

- Linear Terms

  - Station. This fits a mean value for each Station, which adjusts for conditions at that site. This is not necessarily a very robust way of addressing location in the estuary, since I would not be surprised if those differences themselves show seasonal patterns, especially based on the location of the turbidity maximum, but it's a reasonable starting point.
  - Season. We include a season term, not so much because we are interested in estimated effect sizes, as because we have reason to believe samples collected in certain seasons are likely to be more similar than samples collected in other seasons.
- Smoothed Terms
  - Temperature (untransformed)

  - Salinity (untransformed) but as you will see, there are still problems

  - log(Turbidity)

  - log(Chlorophyll)

  - log(River Herring + 1). I had to add one to deal with zero counts.
- Random Terms
  - Year. I added Year as a random factor (`Yearf`). The model fits a single estimate of variance from year to year, rather than fitting separate parameters for each year.
  - Sampling event (`sample_event`). This is an alternative that basically says every sampling event is sightly different, for reasons we don't examine.
- Correlation Structure
  - I test the effect of fitting an AR1 autocorrelation structure based on `sample_seq` within each Station.

Shrinkage Estimators

Each of these models uses "Shrinkage" estimates of the smoothing terms, which allow certain terms to be "shrunk" out of the model. It's an alternative to AIC based model selection. See the help file for `step.gam`, which explains why `mgcv` does not include a step procedure.

Model Alternatives

We are looking at values (density) over the positive number line. So we should principally look at models that assign zero probability (or negligibly small probability) to negative numbers.

Also, we expect our estimates of "density" to be pretty precise when density is low, but not if density is high. If we only count 5 zooplankton, it's unlikely that a replicate count would find 55, so an error of 50 is highly unlikely. On the other hand, if we count 3000, we might not be too surprised if a replicate count had 3050.

Our best model choices should reflect those two features of our data. I considered many model alternatives (see the other notebooks for deeper discussion), but have settled on using Gaussian models on log transformed (and log + 1 transformed) response variables. These models naturally assume higher deviations for higher density. The log transform assumes deviations are proportional to density, which feels like a reasonable assumption. But there's a wonky outlier in the log transformed models of `combined_density`. . . .

Testing Models With Different Random Structures

If you want to jump ahead, look at Model 4. That's the one I like best, and I develop the analysis more completely than for the other models.

Model 1: GAMM Model with random factor `Yearf`

```
density_gam <- gamm(log(combined_density) ~
                      Station +
                      Season +
                      s(Temp, bs="ts") +
                      s(Sal, bs="ts") +
                      s(log(Turb), bs="ts") +
                      s(log(Chl), bs="ts") +
                      s(log1p(RH),bs="ts"),
                    random = list(Yearf = ~ 1),
                    data = base_data, family = 'gaussian')
```

```
summary(density_gam$gam)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log(combined_density) ~ Station + Season + s(Temp, bs = "ts") +
#>     s(Sal, bs = "ts") + s(log(Turb), bs = "ts") + s(log(Chl),
#>     bs = "ts") + s(log1p(RH), bs = "ts")
#>
#> Parametric coefficients:
#>                Estimate Std. Error t value Pr(>|t|)
#> (Intercept)      9.3260     0.4314  21.618  < 2e-16 ***
#> Station2        -0.7172     0.3194  -2.245  0.02959 *
#> Station3        -0.4711     0.2972  -1.585  0.11977
#> Station4        -0.8472     0.3228  -2.624  0.01174 *
#> SeasonSummer    -1.2117     0.3996  -3.032  0.00398 **
#> SeasonFall      -1.1395     0.3815  -2.987  0.00451 **
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>                    edf Ref.df      F  p-value
#> s(Temp)      7.427e-01      9  0.361 0.056563 .
#> s(Sal)       3.246e+00      9 10.155  < 2e-16 ***
#> s(log(Turb)) 9.228e-01      9  0.882 0.008441 **
#> s(log(Chl))  1.063e+00      9  1.867 0.000411 ***
#> s(log1p(RH)) 2.084e-08      9  0.000 0.831759
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.334
#>   Scale est. = 0.22997   n = 58
```

Missing Values?

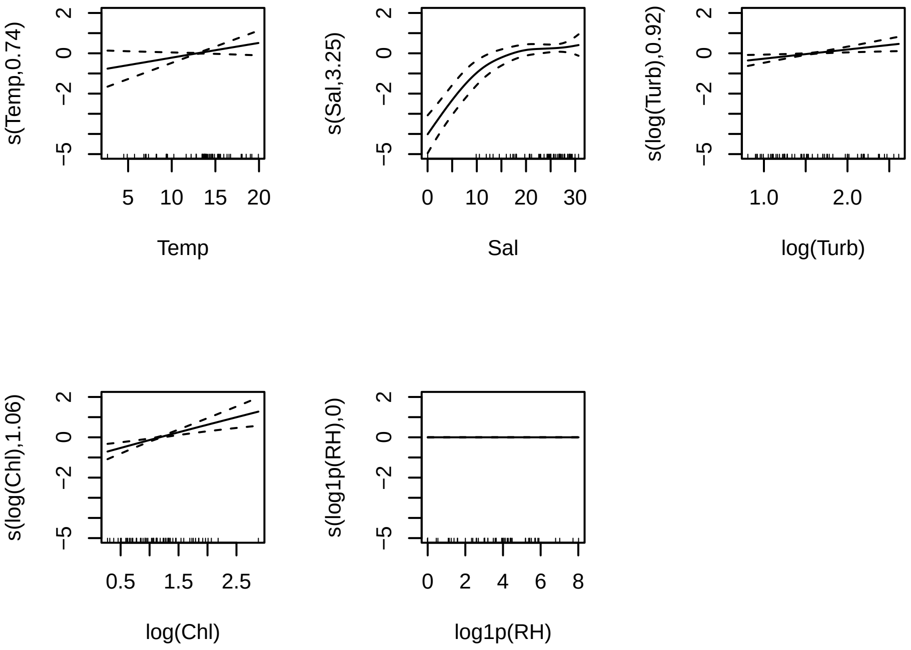
The last line of the output shows `n = 58`, but we started with 60 observations (5 years x 23 seasons x 4 Stations. Two samples were dropped from these models because we lack River Herring data.

We could include all samples if we drop herring as a predictor, but that was one of the motivating questiosn behind the original study!

```
base_data %>%
  select(combined_density, Station , Season, Temp,
          Sal, Turb, Chl, RH, Yearf) %>%
  filter(! complete.cases(.))
#> # A tibble: 2 x 9
#>   combined_density Station Season  Temp   Sal  Turb   Chl    RH Yearf
#>              <dbl> <fct>   <fct>  <dbl> <dbl> <dbl> <dbl> <dbl> <fct>
#> 1            1101. 4       Spring  8.42 24.5   4.08  1.94    NA 2015
#> 2            1138. 1       Spring 10.5   0.12  1.95  3.2     NA 2017
```

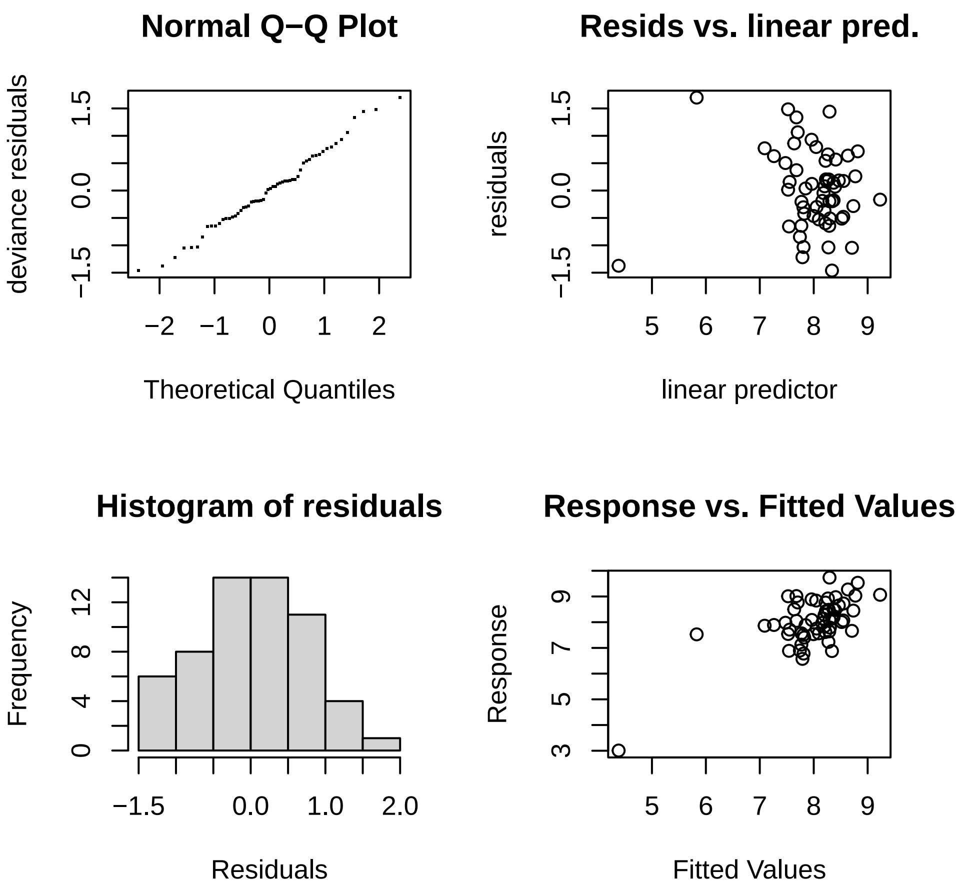
```
anova(density_gam$gam)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log(combined_density) ~ Station + Season + s(Temp, bs = "ts") +
#>     s(Sal, bs = "ts") + s(log(Turb), bs = "ts") + s(log(Chl),
#>     bs = "ts") + s(log1p(RH), bs = "ts")
#>
#> Parametric Terms:
#>         df     F p-value
#> Station  3 2.710  0.0559
#> Season   2 4.776  0.0130
#>
#> Approximate significance of smooth terms:
#>                    edf    Ref.df      F  p-value
#> s(Temp)      7.427e-01 9.000e+00  0.361 0.056563
#> s(Sal)       3.246e+00 9.000e+00 10.155  < 2e-16
#> s(log(Turb)) 9.228e-01 9.000e+00  0.882 0.008441
#> s(log(Chl))  1.063e+00 9.000e+00  1.867 0.000411
#> s(log1p(RH)) 2.084e-08 9.000e+00  0.000 0.831759
```

```
oldpar <- par(mfrow = c(2,3))
plot(density_gam$gam)
par(oldpar)
```



Model Checks

```
oldpar <- par(mfrow = c(2,2))
gam.check(density_gam$gam)
```

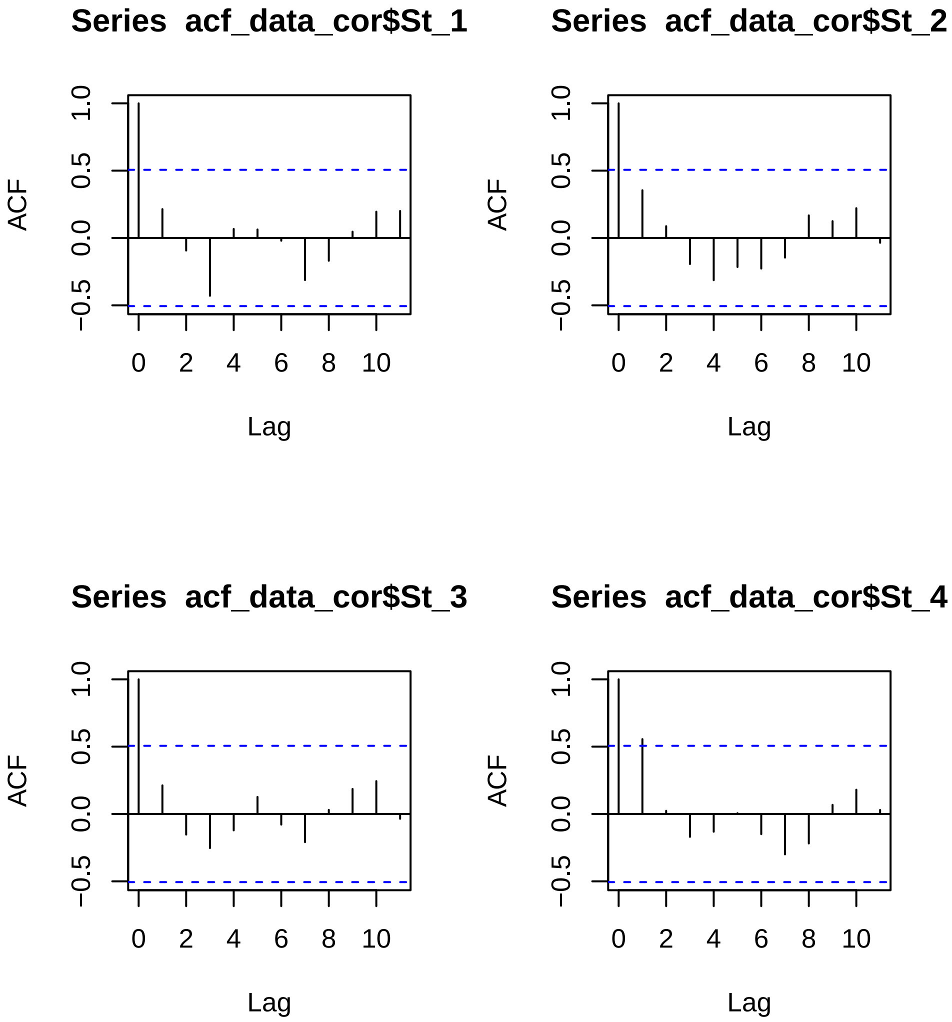


```
#>
#> 'gamm' based fit - care required with interpretation.
#> Checks based on working residuals may be misleading.
#> Basis dimension (k) checking results. Low p-value (k-index<1) may
#> indicate that k is too low, especially if edf is close to k'.
#>
#>                     k'      edf k-index p-value
#> s(Temp)       9.00e+00 7.43e-01    1.12    0.76
#> s(Sal)        9.00e+00 3.25e+00    1.08    0.68
#> s(log(Turb)) 9.00e+00 9.23e-01    0.93    0.24
#> s(log(Chl))  9.00e+00 1.06e+00    0.97    0.36
#> s(log1p(RH)) 9.00e+00 2.08e-08    0.94    0.27
par(oldpar)
```

The residuals are excellent, but the two low fitted values are problematic. As we saw before, those are low salinity samples.

Autocorrelation of Residuals

```
r <- resid(density_gam$gam)
r_data <- base_data %>%
  select(Yearf, Month, Season, sample_event, Station, Temp,
           Sal, Turb, Chl, RH, combined_density) %>%
  filter(complete.cases(.)) %>%
  mutate(resid = r)

acf_data_cor <- r_data %>%
  pivot_wider(id_cols = c(sample_event, Station),
                values_from = resid,
                names_from = 'Station',
                names_prefix = 'St_' )

oldpar <- par(mfrow = c(2,2), mar = c(5,4, 6,2) + 0.1)
acf(acf_data_cor$St_1, na.action = na.pass)
acf(acf_data_cor$St_2, na.action = na.pass)
acf(acf_data_cor$St_3, na.action = na.pass)
acf(acf_data_cor$St_4, na.action = na.pass)
```



```
par(oldpar)
```

Interestingly, there is slightly MORE evidence for temporal autocorrelation here than there was for our considerably simpler linear models.

Model 2: with Autocorrelation

Here , we add an autocorrelation term to the previous model.

```
density_gam_cor <- gamm(log(combined_density) ~ 
                          Station + 
                          Season +
                          s(Temp, bs="ts") +
                          s(Sal, bs="ts") + 
                          s(log(Turb), bs="ts") + 
                          s(log(Chl), bs="ts") + 
                          s(log1p(RH),bs="ts"),
                        random = list(Yearf = ~ 1),
                        correlation = corAR1(form = ~ sample_seq|Station),
                        data = base_data, family = 'gaussian')
summary(density_gam_cor$gam)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> log(combined_density) ~ Station + Season + s(Temp, bs = "ts") + 
#>     s(Sal, bs = "ts") + s(log(Turb), bs = "ts") + s(log(Chl), 
#>     bs = "ts") + s(log1p(RH), bs = "ts")
#> 
#> Parametric coefficients:
#>               Estimate Std. Error t value Pr(>|t|)    
#> (Intercept)     9.4140     0.4257  22.114  < 2e-16 ***
#> Station2       -0.6876     0.2828  -2.432  0.01901 *  
#> Station3       -0.4106     0.2538  -1.618  0.11253    
#> Station4       -0.7740     0.2860  -2.706  0.00953 ** 
#> SeasonSummer  -1.4157     0.4292  -3.298  0.00189 ** 
#> SeasonFall    -1.3094     0.3915  -3.345  0.00165 ** 
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> Approximate significance of smooth terms:
#>                    edf Ref.df      F  p-value    
#> s(Temp)      8.504e-01      9  0.534 0.029698 *  
#> s(Sal)       3.364e+00      9 13.509  < 2e-16 ***
#> s(log(Turb)) 9.766e-01      9  1.353 0.001784 ** 
#> s(log(Chl))  1.109e+00      9  2.146 0.000182 ***
#> s(log1p(RH)) 2.439e-08      9  0.000 0.933131    
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> R-sq.(adj) =  0.326   
#>   Scale est. = 0.21526   n = 58
```

Including a correlations structure reduces the standard errors, but there's trouble:

Autocorrelation Structure

```
density_gam_cor$lme$modelStruct$corStruct
#> Correlation structure of class corAR1 representing
#>        Phi
#> -0.3922034
```

THe autocorrelation coefficient is **negative**. That suggests successive observations are more different than expected, not less different as we would expect. I'd recommend **against** fitting an autocorrelation term. I don't think a negative autocorrelation makes any scientific sense in our context.

Model 3: With random term for `sampling_event`

Another way to address correlations is to treat each sampling event as a random factor in its own right. This does not suggest that sequential observations should be either more or less similar to each other, but does imply that some sample days are higher and some samples are lower, largely due to chance.

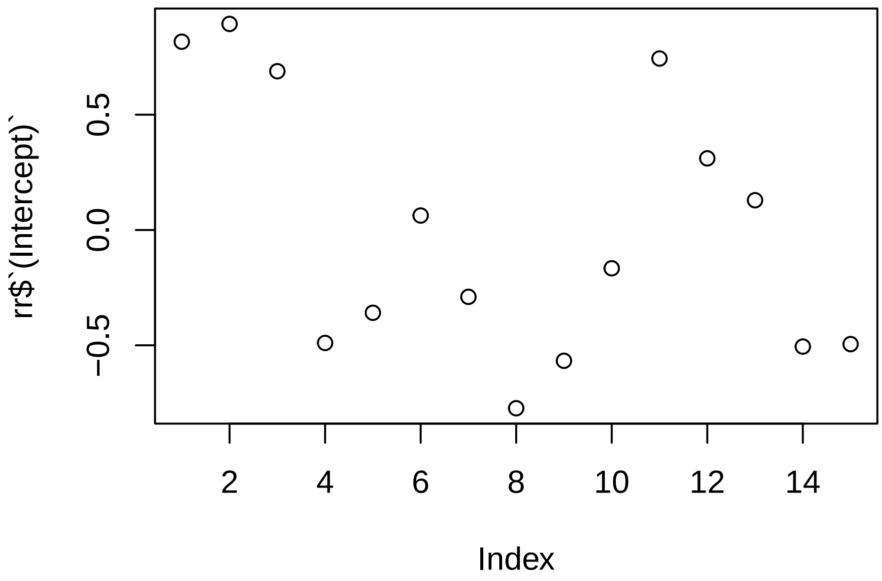
```
density_gam_seq <- gamm(log(combined_density) ~
                          Station +
                          Season +
                          s(Temp, bs="ts") +
                          s(Sal, bs="ts") +
                          s(log(Turb), bs="ts") +
                          s(log(Chl), bs="ts") +
                          s(log1p(RH),bs="ts"),
                        random = list(sample_event = ~ 1),
                        data = base_data, family = 'gaussian')
summary(density_gam_seq$gam)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> log(combined_density) ~ Station + Season + s(Temp, bs = "ts") + 
#>     s(Sal, bs = "ts") + s(log(Turb), bs = "ts") + s(log(Chl), 
#>     bs = "ts") + s(log1p(RH), bs = "ts")
#> 
#> Parametric coefficients:
#>              Estimate Std. Error t value Pr(>|t|)    
#> (Intercept)    9.5545     0.4849  19.706   <2e-16 ***
#> Station2      -0.8068     0.3085  -2.616   0.0120 *  
#> Station3      -0.5348     0.2994  -1.786   0.0807 .  
#> Station4      -0.9368     0.3378  -2.773   0.0080 ** 
#> SeasonSummer  -1.4629     0.6307  -2.320   0.0249 *  
#> SeasonFall    -1.3346     0.6001  -2.224   0.0311 *  
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> Approximate significance of smooth terms:
```

```
#>                      edf Ref.df      F  p-value    
#> s(Temp)        7.131e-01      9  0.295 0.077022 .  
#> s(Sal)         3.382e+00      9 11.470  < 2e-16 ***
#> s(log(Turb))   8.785e-01      9  0.567 0.027136 *  
#> s(log(Chl))    1.247e+00      9  1.824 0.000504 ***
#> s(log1p(RH))   5.696e-08      9  0.000 0.826288    
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> R-sq.(adj) =   0.34   
#>   Scale est. = 0.16865   n = 58
```

That model actually increases standard errors… Worse, it ceases to fit an annual correction in any meaningful way, and we think year to Year variation is important.

We can take a look at the random effects fitted.

```
rr <- ranef(density_gam_seq$lme)$sample_event
plot(rr$`(Intercept)`)
```

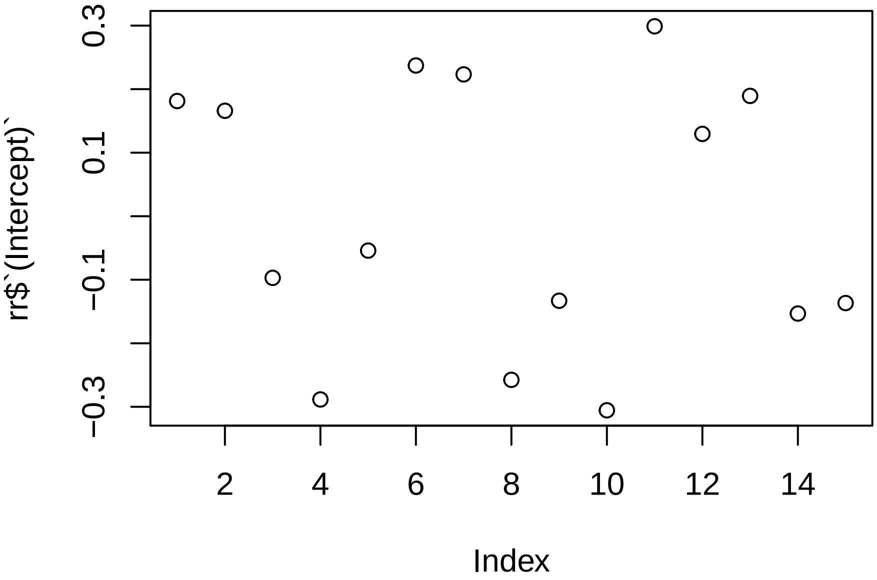


I think I can see some remaining temporal structure, with each annual grouping of three observations lookign “more similar” that expected. I think we need to keep the `yearf` term in the model somewhere (fixed effects or random effects).

Model 4: With random terms for `Yearf` and `sampling_event`

```
density_gam_seq_2 <- gamm(log(combined_density) ~
                          Station +
                          Season +
                          s(Temp, bs="ts") +
                          s(Sal, bs="ts") +
                          s(log(Turb), bs="ts") +
                          s(log(Chl), bs="ts") +
                          s(log1p(RH),bs="ts"),
                        random = list(Yearf = ~ 1, sample_event = ~ 1),
                        data = base_data, family = 'gaussian')
summary(density_gam_seq_2$gam)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> log(combined_density) ~ Station + Season + s(Temp, bs = "ts") + 
#>     s(Sal, bs = "ts") + s(log(Turb), bs = "ts") + s(log(Chl), 
#>     bs = "ts") + s(log1p(RH), bs = "ts")
#> 
#> Parametric coefficients:
#>              Estimate Std. Error t value Pr(>|t|)    
#> (Intercept)    9.3298     0.4471  20.869  < 2e-16 ***
#> Station2      -1.0127     0.2760  -3.669 0.000624 ***
#> Station3      -0.7621     0.2627  -2.900 0.005672 ** 
#> Station4      -1.1834     0.2943  -4.020 0.000211 ***
#> SeasonSummer  -0.8743     0.3377  -2.589 0.012798 *  
#> SeasonFall    -0.7889     0.3203  -2.463 0.017533 *  
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> Approximate significance of smooth terms:
#>                    edf Ref.df      F  p-value    
#> s(Temp)      3.688e-05      9  0.000 0.112923    
#> s(Sal)       3.437e+00      9 12.044  < 2e-16 ***
#> s(log(Turb)) 8.029e-01      9  0.420 0.049332 *  
#> s(log(Chl))  1.186e+00      9  2.021 0.000268 ***
#> s(log1p(RH)) 1.269e-05      9  0.000 0.800262    
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> R-sq.(adj) =  0.258   
#>   Scale est. = 0.17578   n = 58
```

```
rr <- ranef(density_gam_seq_2$lme)$sample_event
rYear <- ranef(density_gam_seq_2$lme)$Yearf
plot(rr$`(Intercept)`)
```



The random terms for sampling_events don't show structure any more. We definitely need the Year random factor.

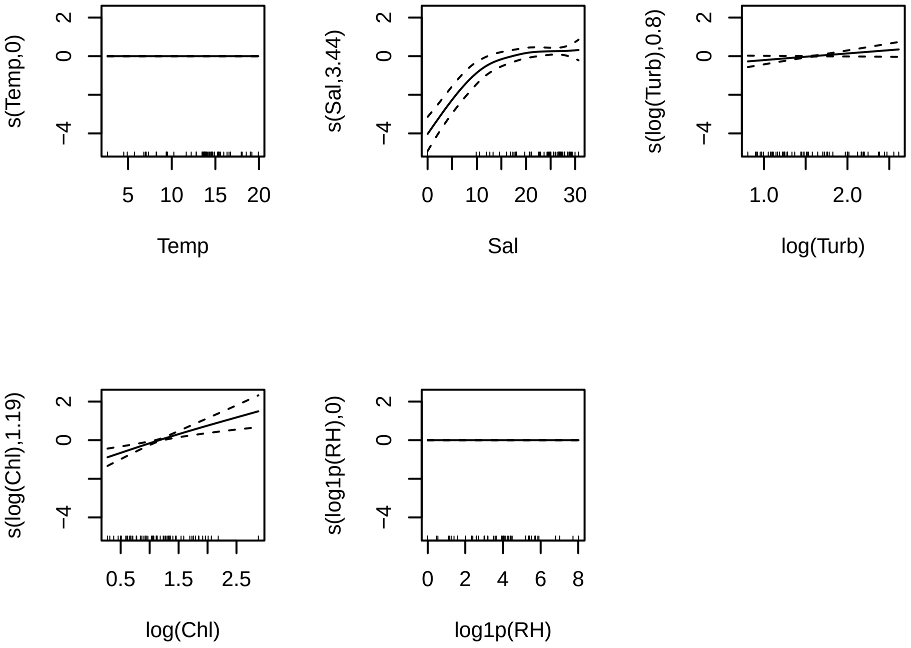
Extracting the variance components is a bit tricky, because they are buried in the `lme` object, which hides a lot of the complexity of the GAMM fitting.

```
vc <- VarCorr(density_gam_seq_2$lme)
# `VarCorr()` produces a text matrix.
# I want the last few rows,  I had to look at it to figure out which rows
# contain the information I wanted.

as_tibble(unclass(vc))[c(52, 54, 55),] %>%
  mutate(across(everything(), function(x) round(as.numeric(x), digits = 3)),
        name = rownames(vc)[c(51, 53, 55)]) %>%
  relocate(name)
#> # A tibble: 3 x 3
#>   name           Variance StdDev
#>   <chr>             <dbl>  <dbl>
#> 1 Yearf =           0.31   0.557
#> 2 sample_event =    0.096  0.309
#> 3 Residual          0.176  0.419
```

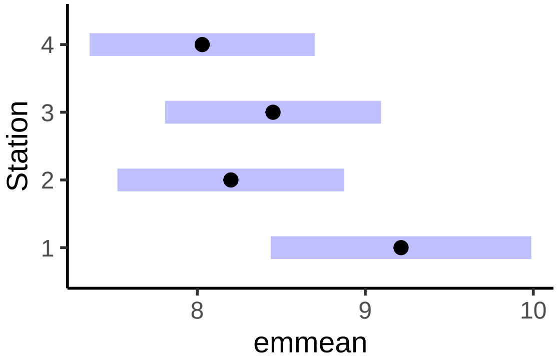
If I'm reading that correctly, the `Yearf` term is fit as a normal variate with mean zero and standard deviation Of 0.557, while the individual sampling events were fit as a normal variate with standard deviation 0.301 on top of that. The residual for the model is on the order of 0.419, so slightly higher than variation among the samples, but less than variation among the Years.

```
anova(density_gam_seq_2$gam)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> log(combined_density) ~ Station + Season + s(Temp, bs = "ts") + 
#>     s(Sal, bs = "ts") + s(log(Turb), bs = "ts") + s(log(Chl), 
#>     bs = "ts") + s(log1p(RH), bs = "ts")
#> 
#> Parametric Terms:
#>         df     F p-value
#> Station  3 6.020 0.00149
#> Season   2 3.802 0.02955
#> 
#> Approximate significance of smooth terms:
#>                    edf    Ref.df      F  p-value
#> s(Temp)      3.688e-05 9.000e+00  0.000 0.112923
#> s(Sal)       3.437e+00 9.000e+00 12.044  < 2e-16
#> s(log(Turb)) 8.029e-01 9.000e+00  0.420 0.049332
#> s(log(Chl))  1.186e+00 9.000e+00  2.021 0.000268
#> s(log1p(RH)) 1.269e-05 9.000e+00  0.000 0.800262
```

```
oldpar <- par(mfrow = c(2,3))
plot(density_gam_seq_2$gam)
par(oldpar)
```

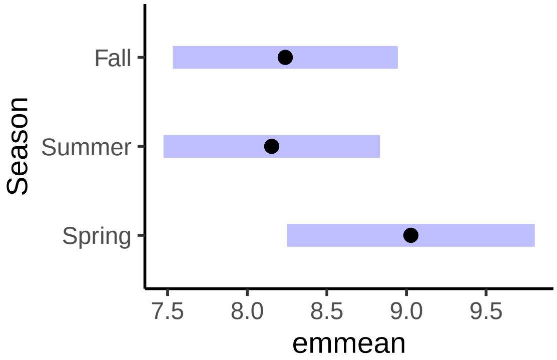


```
Sta_emms <- emmeans(density_gam_seq_2, ~Station, type = 'response',
                    data = base_data)
plot(Sta_emms)
```



```
pairs(Sta_emms, adjust ='bonferroni')
#>  contrast estimate    SE   df t.ratio p.value
#>  1 - 2       1.013 0.276 46.6   3.669  0.0037
#>  1 - 3       0.762 0.263 46.6   2.900  0.0340
#>  1 - 4       1.183 0.294 46.6   4.020  0.0013
#>  2 - 3      -0.251 0.188 46.6  -1.331  1.0000
#>  2 - 4       0.171 0.201 46.6   0.851  1.0000
#>  3 - 4       0.421 0.179 46.6   2.357  0.1362
#>
#> Results are averaged over the levels of: Season
#> P value adjustment: bonferroni method for 6 tests
```

```
Seas_emms <- emmeans(density_gam_seq_2, ~Season, type = 'response',
                     data = base_data)
plot(Seas_emms)
```



```
pairs(Seas_emms, adjust ='bonferroni')
#>  contrast        estimate    SE   df t.ratio p.value
#>  Spring - Summer   0.8743 0.338 46.6   2.589  0.0384
#>  Spring - Fall     0.7889 0.320 46.6   2.463  0.0526
#>  Summer - Fall    -0.0854 0.262 46.6  -0.326  1.0000
```

```
#> 
#> Results are averaged over the levels of: Station 
#> P value adjustment: bonferroni method for 3 tests
```

- The Station differences are significant by ANOVA F test. Pairwise comparisons show that Station 1 (upstream) shows the highest combined density, which is significantly higher than for Stations 2 and 4, but not different from Station 3 (by multiple comparisons test anyway). There are no meaningful differences among the three downstream Stations.
- While zooplankton density varies by season, none of the pairwise comparisons or marginal means are individually significant. Densities are somewhat higher in the spring.
- Salinity Shows a highly significant curved (~ 3 edf) pattern, driven largely by a couple of very low salinity, low density samples.
- Turbidity and Chlorophyll both fit close to linear (~ 1 edf) relationships that appear fairly robust to model specification. Zooplankton abundance is correlated with higher chlorophyll land higher turbidity. (it's not unreasonable to test for a significant interaction there, btu I have not done so.)

Model Comparisons

Compare Model 1 and Model 2

```
anova(density_gam$lme, density_gam_cor$lme)
#>                     Model df      AIC      BIC    logLik   Test  L.Ratio
#> density_gam$lme         1 13 142.0046 168.7904 -58.00230
#> density_gam_cor$lme     2 14 139.8911 168.7373 -55.94553 1 vs 2 4.113545
#>                     p-value
#> density_gam$lme
#> density_gam_cor$lme  0.0425
```

So the model that includes a correlation structure is better that a model without.

Compare Model 1 and Model 4

```
anova(density_gam$lme, density_gam_seq_2$lme)
#>                       Model df      AIC      BIC    logLik   Test L.Ratio
#> density_gam$lme           1 13 142.0046 168.7904 -58.00230
#> density_gam_seq_2$lme     2 14 139.2851 168.1313 -55.64257 1 vs 2 4.71947
#>                       p-value
#> density_gam$lme
#> density_gam_seq_2$lme  0.0298
```

The model that includes the random factor for each sampling event *also* outperforms Model 1.

We can't run a formal test between Models 2 and 4, because they are not nested, and have identical number of estimated parameters (degrees of freedom), but we can still informally compare models by AIC. Model 4 has(ever so slightly) smaller AIC / log likelihood compared to model 2.

Conclusions

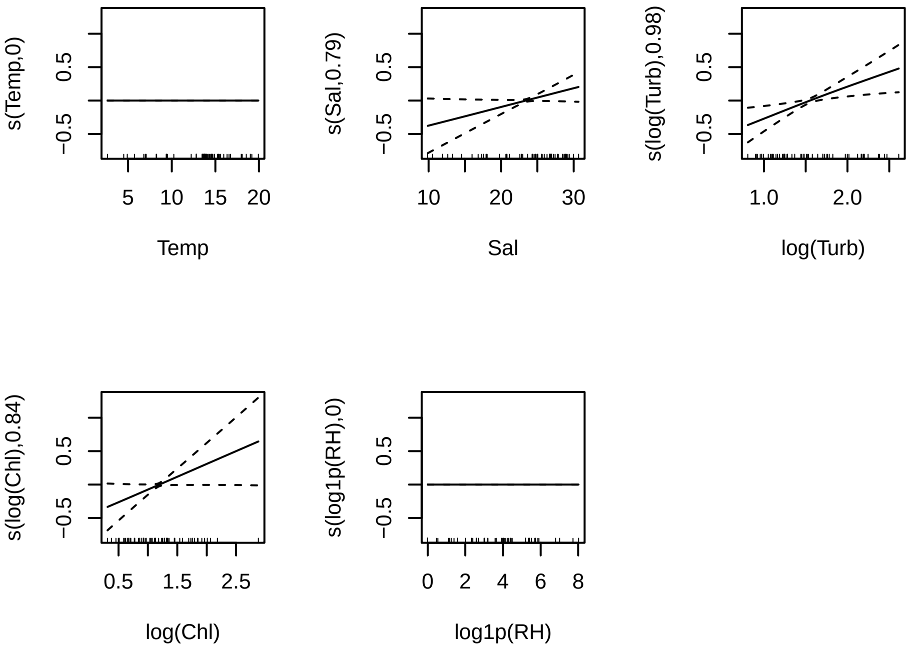
I don't think a negative autocorrelation structure makes any sense in this context. Since models 2 and 4 perform nearly as well (by AIC), I lean towards using model 4.

Model 4, but Without the Low Salinity Samples

```
density_gam_seq_2 <- gamm(log(combined_density) ~
                          Station +
                          Season +
                          s(Temp, bs="ts") +
                          s(Sal, bs="ts") +
                          s(log(Turb), bs="ts") +
                          s(log(Chl), bs="ts") +
                          s(log1p(RH),bs="ts"),
                        random = list(Yearf = ~ 1, sample_event = ~ 1),
                        data = smaller_data, family = 'gaussian')
summary(density_gam_seq_2$gam)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log(combined_density) ~ Station + Season + s(Temp, bs = "ts") +
#>     s(Sal, bs = "ts") + s(log(Turb), bs = "ts") + s(log(Chl),
#>     bs = "ts") + s(log1p(RH), bs = "ts")
#>
#> Parametric coefficients:
#>              Estimate Std. Error t value Pr(>|t|)
#> (Intercept)    8.8720     0.3756  23.624  < 2e-16 ***
#> Station2      -0.6906     0.2239  -3.085  0.00339 **
#> Station3      -0.4112     0.2097  -1.961  0.05577 .
#> Station4      -0.7515     0.2426  -3.098  0.00327 **
#> SeasonSummer  -0.3545     0.2866  -1.237  0.22229
#> SeasonFall    -0.4093     0.2707  -1.512  0.13713
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>                    edf Ref.df     F p-value
#> s(Temp)      5.439e-09      9 0.000  0.3268
#> s(Sal)       7.883e-01      9 0.431  0.0513 .
#> s(log(Turb)) 9.786e-01      9 1.019  0.0068 **
#> s(log(Chl))  8.389e-01      9 0.486  0.0419 *
#> s(log1p(RH)) 8.558e-09      9 0.000  0.9584
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.0854
#>   Scale est. = 0.13391   n = 56
```

```
anova(density_gam_seq_2$gam)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> log(combined_density) ~ Station + Season + s(Temp, bs = "ts") + 
#>     s(Sal, bs = "ts") + s(log(Turb), bs = "ts") + s(log(Chl), 
#>     bs = "ts") + s(log1p(RH), bs = "ts")
#> 
#> Parametric Terms:
#>         df     F p-value
#> Station  3 4.240 0.00982
#> Season   2 1.177 0.31701
#> 
#> Approximate significance of smooth terms:
#>                    edf    Ref.df     F p-value
#> s(Temp)      5.439e-09 9.000e+00 0.000  0.3268
#> s(Sal)       7.883e-01 9.000e+00 0.431  0.0513
#> s(log(Turb)) 9.786e-01 9.000e+00 1.019  0.0068
#> s(log(Chl))  8.389e-01 9.000e+00 0.486  0.0419
#> s(log1p(RH)) 8.558e-09 9.000e+00 0.000  0.9584
```

```
oldpar <- par(mfrow = c(2,3))
plot(density_gam_seq_2$gam)
par(oldpar)
```



That fits linear relationships to salinity, turbidity and chlorophyll. Differences between the Stations remain important, but there is little evidence here for a seasonal pattern. That suggests that some of the seasonal pattern we found before was the result of fitting those low salinity spring samples. Those low salinity spring samples had large effect on model fits.

Diversity (Shannon Index)

It would probably be better to check each of our four random model structures for the diversity data too, but instead, we dive right in using the same model form, and i tlooks like we can get away with it.

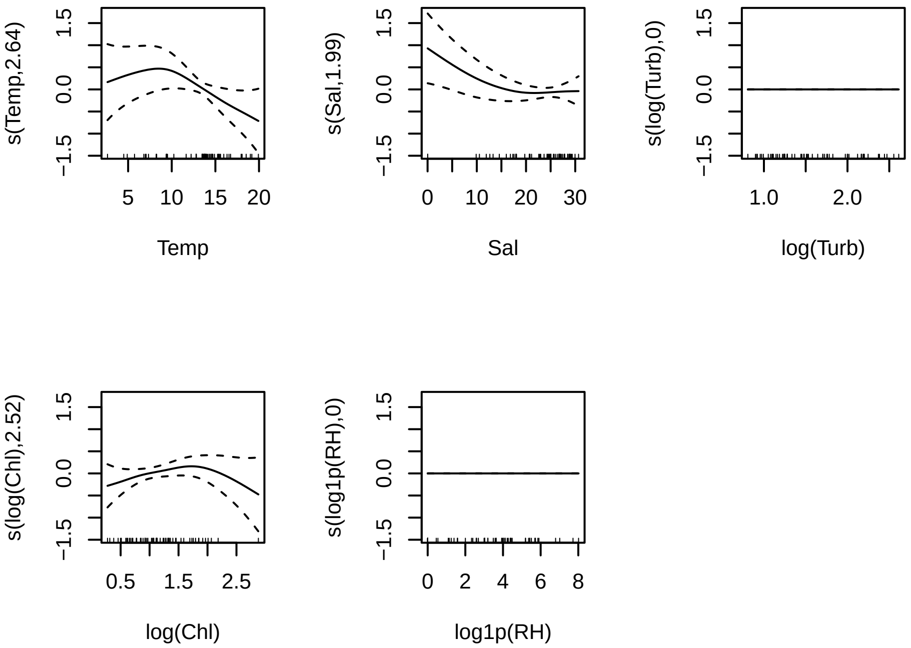
Gaussian GAM, with Identity Link

We are using "Shrinkage" estimates of the smoothing terms again, which allow certain terms to be "shrunk" out of the model

```
shannon_gam <- gam(H ~ Station +
                     Season +
                     s(Temp, bs="ts") +
                     s(Sal, bs="ts") +
                     s(log(Turb), bs="ts") +
                     s(log(Chl), bs="ts") +
                     s(log1p(RH),bs="ts"),
                   random = list(Yearf = ~ 1, sample_event = ~ 1),
                   data = base_data, family = 'gaussian')
summary(shannon_gam)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> H ~ Station + Season + s(Temp, bs = "ts") + s(Sal, bs = "ts") + 
#>     s(log(Turb), bs = "ts") + s(log(Chl), bs = "ts") + s(log1p(RH), 
#>     bs = "ts")
#> 
#> Parametric coefficients:
#>              Estimate Std. Error t value Pr(>|t|)  
#> (Intercept)    0.8738     0.3338   2.618    0.012 *
#> Station2       0.2543     0.2935   0.866    0.391  
#> Station3       0.3433     0.2698   1.272    0.210  
#> Station4       0.2077     0.2865   0.725    0.472  
#> SeasonSummer   0.4842     0.4026   1.203    0.235  
#> SeasonFall     0.3203     0.3957   0.810    0.422  
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> Approximate significance of smooth terms:
#>                    edf Ref.df     F p-value  
#> s(Temp)      2.642e+00      9 0.654  0.0753 .
#> s(Sal)       1.994e+00      9 0.757  0.0261 *
#> s(log(Turb)) 5.537e-08      9 0.000  0.5331  
#> s(log(Chl))  2.517e+00      9 0.510  0.1321  
#> s(log1p(RH)) 3.173e-08      9 0.000  0.3568  
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.319   Deviance explained = 46.4%
#> GCV = 0.26465  Scale est. = 0.20463   n = 58
```

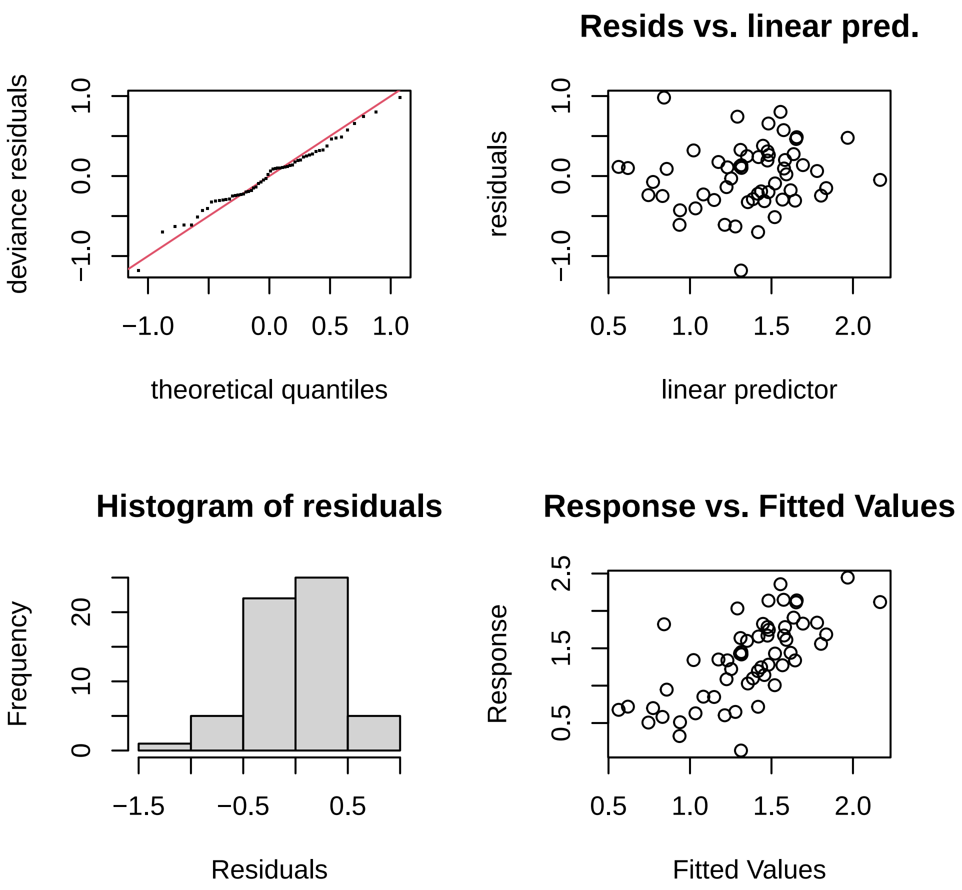
```
anova(shannon_gam)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> H ~ Station + Season + s(Temp, bs = "ts") + s(Sal, bs = "ts") +
#>     s(log(Turb), bs = "ts") + s(log(Chl), bs = "ts") + s(log1p(RH),
#>     bs = "ts")
#>
#> Parametric Terms:
#>         df     F p-value
#> Station  3 0.622   0.605
#> Season   2 1.010   0.372
#>
#> Approximate significance of smooth terms:
#>                    edf    Ref.df     F p-value
#> s(Temp)      2.642e+00 9.000e+00 0.654  0.0753
#> s(Sal)       1.994e+00 9.000e+00 0.757  0.0261
#> s(log(Turb)) 5.537e-08 9.000e+00 0.000  0.5331
#> s(log(Chl))  2.517e+00 9.000e+00 0.510  0.1321
#> s(log1p(RH)) 3.173e-08 9.000e+00 0.000  0.3568
```

```
oldpar <- par(mfrow = c(2,3))
plot(shannon_gam)
par(oldpar)
```



While the GAMM fits curves for several predictors, only the relationship with salinity is retained in the model as statistically significant. It appears much of that pattern is driven by a couple of low salinity samples.

```
oldpar <- par(mfrow = c(2,2))
gam.check(shannon_gam)
```



```
#>
#> Method: GCV   Optimizer: magic
#> Smoothing parameter selection converged after 19 iterations.
#> The RMS GCV score gradient at convergence was 5.565746e-08 .
#> The Hessian was positive definite.
#> Model rank =  51 / 51
#>
#> Basis dimension (k) checking results. Low p-value (k-index<1) may
#> indicate that k is too low, especially if edf is close to k’.
#>
#>                    k’      edf k-index p-value
#> s(Temp)      9.00e+00 2.64e+00    1.12    0.72
#> s(Sal)       9.00e+00 1.99e+00    0.96    0.33
#> s(log(Turb)) 9.00e+00 5.54e-08    1.30    0.98
#> s(log(Chl))  9.00e+00 2.52e+00    1.13    0.79
#> s(log1p(RH)) 9.00e+00 3.17e-08    1.06    0.62
par(oldpar)
```

Not a bad model from a diagnostics point of view.

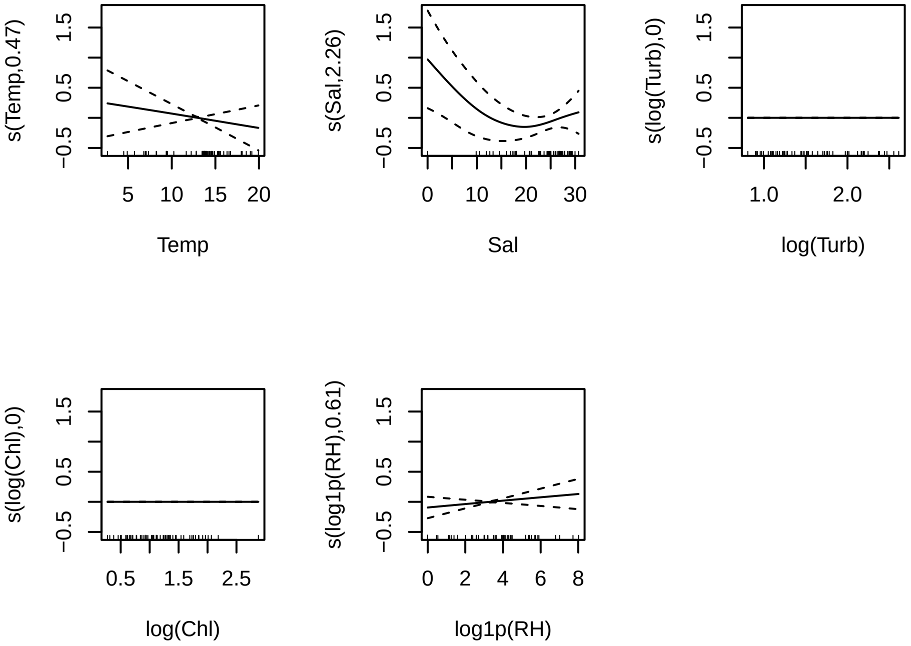
Model on Log Transformed data

Analysis of log-transformed data gives quaitatively siml

```
shannon_gam_2 <- gam(log(H) ~ Station +
                       Season +
                       s(Temp, bs="ts") +
                       s(Sal, bs="ts") +
                       s(log(Turb), bs="ts") +
                       s(log(Chl), bs="ts") +
                       s(log1p(RH),bs="ts"),
                     random = list(Yearf = ~ 1, sample_event = ~ 1),
                     data = base_data, family = gaussian)
summary(shannon_gam_2)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log(H) ~ Station + Season + s(Temp, bs = "ts") + s(Sal, bs = "ts") +
#>     s(log(Turb), bs = "ts") + s(log(Chl), bs = "ts") + s(log1p(RH),
#>     bs = "ts")
#>
#> Parametric coefficients:
#>              Estimate Std. Error t value Pr(>|t|)
#> (Intercept)   -0.3984     0.2793  -1.426   0.1602
#> Station2       0.5340     0.2723   1.961   0.0556 .
#> Station3       0.6425     0.2553   2.517   0.0152 *
#> Station4       0.4239     0.2678   1.583   0.1199
#> SeasonSummer   0.3130     0.2658   1.177   0.2447
#> SeasonFall     0.2202     0.2730   0.807   0.4238
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>                    edf Ref.df     F p-value
#> s(Temp)      4.652e-01      9 0.090 0.18339
#> s(Sal)       2.259e+00      9 1.172 0.00661 **
#> s(log(Turb)) 9.732e-09      9 0.000 0.56025
#> s(log(Chl))  1.209e-07      9 0.000 0.55597
#> s(log1p(RH)) 6.087e-01      9 0.124 0.17096
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.225   Deviance explained = 33.9%
#> GCV = 0.26901  Scale est. = 0.22573   n = 58
```

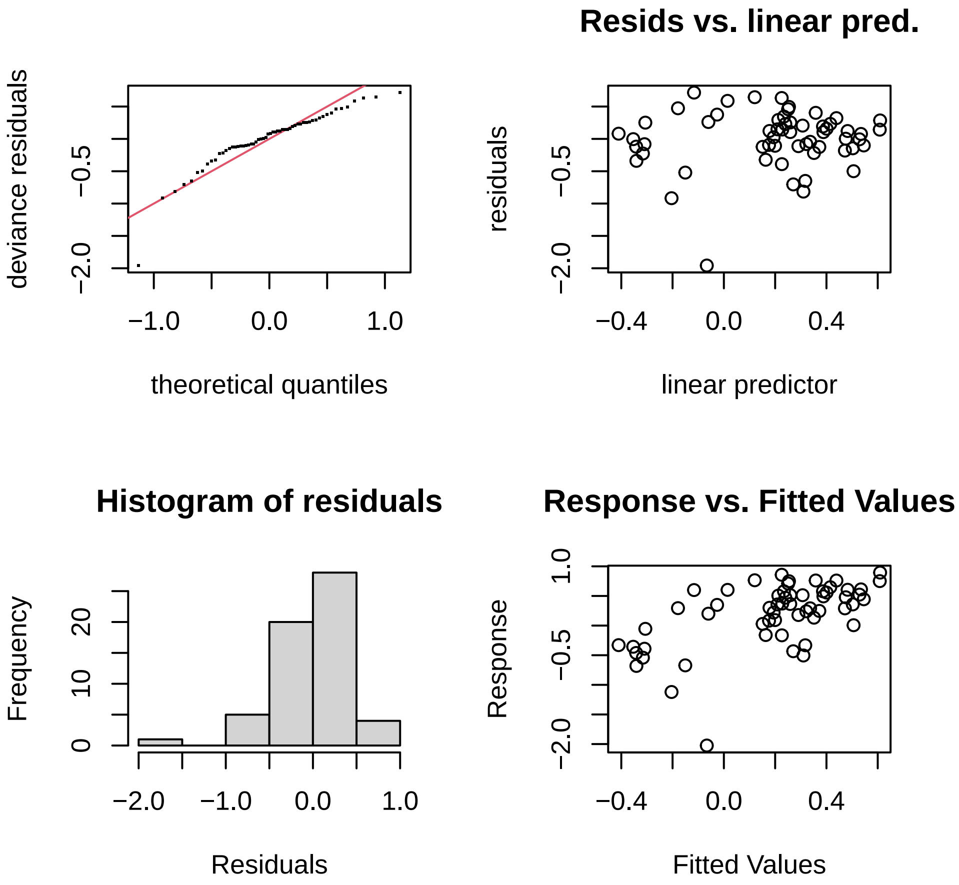
```
anova(shannon_gam_2)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log(H) ~ Station + Season + s(Temp, bs = "ts") + s(Sal, bs = "ts") +
#>     s(log(Turb), bs = "ts") + s(log(Chl), bs = "ts") + s(log1p(RH),
#>     bs = "ts")
#>
#> Parametric Terms:
#>         df     F p-value
#> Station  3 2.242  0.0952
#> Season   2 0.745  0.4802
#>
#> Approximate significance of smooth terms:
#>                    edf    Ref.df     F p-value
#> s(Temp)      4.652e-01 9.000e+00 0.090 0.18339
#> s(Sal)       2.259e+00 9.000e+00 1.172 0.00661
#> s(log(Turb)) 9.732e-09 9.000e+00 0.000 0.56025
#> s(log(Chl))  1.209e-07 9.000e+00 0.000 0.55597
#> s(log1p(RH)) 6.087e-01 9.000e+00 0.124 0.17096
```

```
oldpar <- par(mfrow = c(2,3))
plot(shannon_gam_2)
par(oldpar)
```



The only consistently significant relationship between diversity and any of the predictors relates to salinity. Unfortunately, most of that pattern appears again due to a handful of low salinity samples.

```
oldpar <- par(mfrow = c(2,2))
gam.check(shannon_gam_2)
```



```
#>
#> Method: GCV   Optimizer: magic
#> Smoothing parameter selection converged after 24 iterations.
#> The RMS GCV score gradient at convergence was 3.77895e-08 .
#> The Hessian was positive definite.
#> Model rank =  51 / 51
#>
#> Basis dimension (k) checking results. Low p-value (k-index<1) may
#> indicate that k is too low, especially if edf is close to k’.
#>
#>                   k’      edf k-index p-value
#> s(Temp)      9.00e+00 4.65e-01    1.02    0.47
#> s(Sal)       9.00e+00 2.26e+00    1.06    0.65
#> s(log(Turb)) 9.00e+00 9.73e-09    1.27    0.98
#> s(log(Chl))  9.00e+00 1.21e-07    1.07    0.61
#> s(log1p(RH)) 9.00e+00 6.09e-01    1.07    0.64
par(oldpar)
```

That extreme negative outlier is the main weakness of this model. This model is perhaps slightly less trustworthy than the Gaussian model. Luckily, results are consistent.

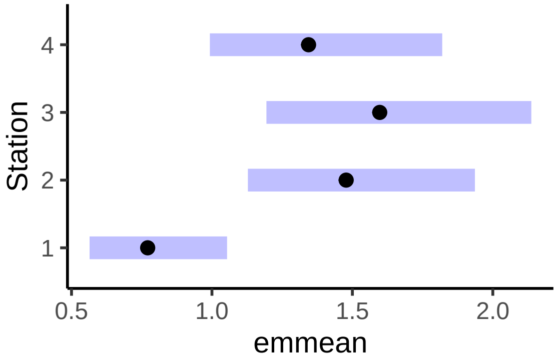
Model without Low Salinity Observations

```
shannon_gam_3 <- gam(log(H) ~ Station +
                       Season +
                       s(Temp, bs="ts") +
                       s(Sal, bs="ts") +
                       s(log(Turb), bs="ts") +
                       s(log(Chl), bs="ts") +
                       s(log1p(RH),bs="ts"),
                     random = list(Yearf = ~ 1, sample_event = ~ 1),
                     data = smaller_data, family = gaussian)
summary(shannon_gam_3)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log(H) ~ Station + Season + s(Temp, bs = "ts") + s(Sal, bs = "ts") +
#>     s(log(Turb), bs = "ts") + s(log(Chl), bs = "ts") + s(log1p(RH),
#>     bs = "ts")
#>
#> Parametric coefficients:
#>              Estimate Std. Error t value Pr(>|t|)
#> (Intercept)   -0.5818     0.2004  -2.903  0.00554 **
#> Station2       0.6505     0.2124   3.062  0.00357 **
#> Station3       0.7283     0.2127   3.423  0.00126 **
#> Station4       0.5556     0.2176   2.553  0.01386 *
#> SeasonSummer   0.3637     0.2373   1.533  0.13174
#> SeasonFall     0.3172     0.2318   1.369  0.17735
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>                    edf Ref.df     F p-value
#> s(Temp)      4.211e-01      9 0.069  0.2258
#> s(Sal)       4.392e-10      9 0.000  0.5514
#> s(log(Turb)) 5.227e-11      9 0.000  1.0000
#> s(log(Chl))  3.420e-10      9 0.000  0.4329
#> s(log1p(RH)) 7.604e-01      9 0.295  0.0636 .
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.216   Deviance explained = 30.4%
#> GCV = 0.25792  Scale est. = 0.22485   n = 56
```

```
anova(shannon_gam_3)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log(H) ~ Station + Season + s(Temp, bs = "ts") + s(Sal, bs = "ts") +
#>     s(log(Turb), bs = "ts") + s(log(Chl), bs = "ts") + s(log1p(RH),
#>     bs = "ts")
#>
#> Parametric Terms:
#>         df     F p-value
#> Station  3 4.297 0.00908
#> Season   2 1.207 0.30788
#>
#> Approximate significance of smooth terms:
#>                    edf    Ref.df     F p-value
#> s(Temp)      4.211e-01 9.000e+00 0.069  0.2258
#> s(Sal)       4.392e-10 9.000e+00 0.000  0.5514
#> s(log(Turb)) 5.227e-11 9.000e+00 0.000  1.0000
#> s(log(Chl))  3.420e-10 9.000e+00 0.000  0.4329
#> s(log1p(RH)) 7.604e-01 9.000e+00 0.295  0.0636
```

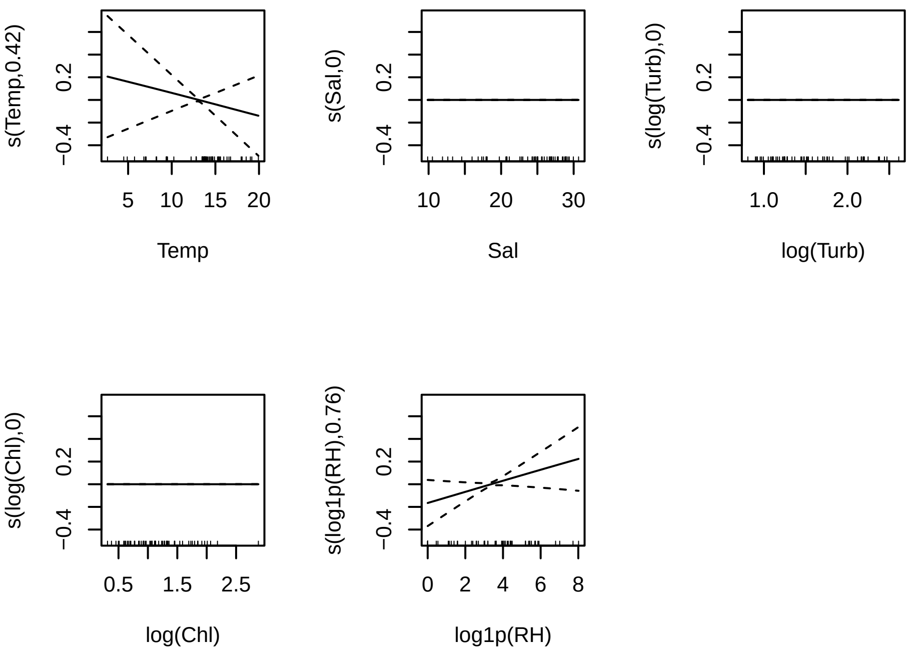
So, when you remove those low salinity samples, what emerges is a statistically robust pattern by station, with station 1 showing lower diversity. River herring has an almost statistically significant effect.

```
Sta_emms <- emmeans(shannon_gam_3, ~Station, type = 'response',
                    data = base_data)
plot(Sta_emms)
```



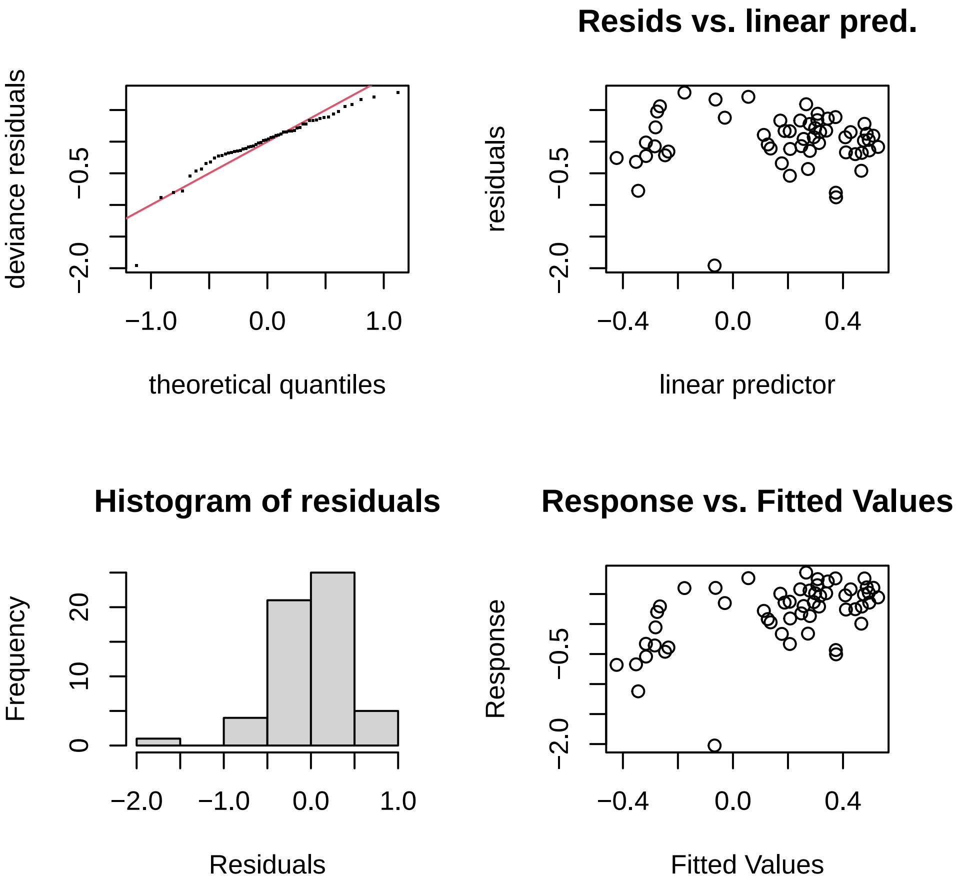
```
pairs(Sta_emms, adjust ='bonferroni')
#>  contrast ratio    SE   df null t.ratio p.value
#>  1 / 2    0.522 0.111 48.8    1  -3.062  0.0214
#>  1 / 3    0.483 0.103 48.8    1  -3.423  0.0076
#>  1 / 4    0.574 0.125 48.8    1  -2.553  0.0832
#>  2 / 3    0.925 0.163 48.8    1  -0.441  1.0000
#>  2 / 4    1.100 0.198 48.8    1   0.529  1.0000
#>  3 / 4    1.189 0.210 48.8    1   0.979  1.0000
#>
#> Results are averaged over the levels of: Season
#> P value adjustment: bonferroni method for 6 tests
#> Tests are performed on the log scale
```

```
oldpar <- par(mfrow = c(2,3))
plot(shannon_gam_3)
par(oldpar)
```



So, diversity is lowest in the spring, but shows no significant relationship to the other predictors.

```
oldpar <- par(mfrow = c(2,2))
gam.check(shannon_gam_3)
```



```
#>
#> Method: GCV   Optimizer: magic
#> Smoothing parameter selection converged after 33 iterations.
#> The RMS GCV score gradient at convergence was 4.130619e-08 .
#> The Hessian was positive definite.
#> Model rank =  51 / 51
#>
#> Basis dimension (k) checking results. Low p-value (k-index<1) may
#> indicate that k is too low, especially if edf is close to k’.
#>
#>                     k’      edf k-index p-value
#> s(Temp)      9.00e+00 4.21e-01    1.07    0.58
#> s(Sal)       9.00e+00 4.39e-10    1.06    0.57
#> s(log(Turb)) 9.00e+00 5.23e-11    1.24    0.97
#> s(log(Chl))  9.00e+00 3.42e-10    1.02    0.42
#> s(log1p(RH)) 9.00e+00 7.60e-01    1.08    0.66
par(oldpar)
```

The extreme low residual is problematic, but otherwise this model is not too bad. Failure to meet model assumption can degrade estimates of stnadard errors, so perhaps the nearly significant effect o river herring is worth more exploration here, presumably with a simpler model?

Single Species Models

Model Choices

Our model alternatives are basically similar to what we had for the Total Density models.

The problem is, we can't use any of the continuous data distributions in GAMS with zero values, at least relying on the canonical link functions, because (log(0) = -Inf; 1/0 = Inf, 1 / 0*0 = Inf). The easiest solution is to add some finite small quantity to the density data, and predict that. Here we predict Density + 1.

Automating Analysis of Separate Species

I'm going to automate analysis of all five selected species by using a "nested" Tibble. This is a convenient alternative to writing a "for" loop to run multiple identical analyses.

I create a "long" data source.

```
spp_data <- base_data %>%
  select(Yearf, Month, Season, sample_event, Station, Temp,
         Sal, Turb, Chl, RH,
         Acartia, Eurytemora, Polychaete, Pseudocal, Temora) %>%
  pivot_longer(-c(Yearf:RH), names_to = 'Species', values_to = 'Density')
```

Next, I create a function to run the analysis. This function takes a data frame or tibble as an argument. The tibble mush have data columns with the correct names.

The initial model fits for some species had a lot of wiggles in them, to an extent that I thought did not make much scientific sense, so I decided to reduce the dimensionality of the GAM smoothers, by adding the parameter `k= 4`. Lowe numbers constrain the GAM to fit smoother lines.

```
my_gamm <- function(.dat) {

  gam(log1p(Density) ~ Station +
        Season +
        s(Temp, bs="ts", k = 4) +
        s(Sal, bs="ts", k = 4) +
        s(log(Turb), bs="ts", k = 4) +
        s(log(Chl), bs="ts", k = 4) +
        s(log1p(RH),bs="ts", k = 4),
      random = list(Yearf = ~ 1, sample_event = ~ 1),
      data = .dat, family = "gaussian")
}
```

Next, I create the nested tibble, and conduct the analysis on each species. . . .

```
spp_analysis <- spp_data %>%
  group_by(Species) %>%
  nest() %>%
  mutate(gam_mods = map(data, my_gamm))
```

and finally, output the model results. I can do that in a "for" loop, but it's Awkward to look through a long list of output, so I step through each sspecies in turn.

Acartia

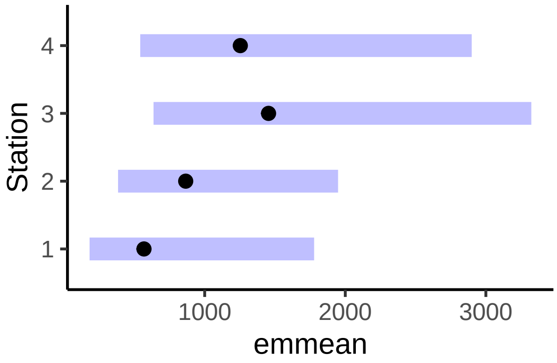
Summary and ANOVA

```
spp = 'Acartia'
mod <- spp_analysis$gam_mods[spp_analysis$Species == spp][[1]]
summary(mod)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> log1p(Density) ~ Station + Season + s(Temp, bs = "ts", k = 4) + 
#>     s(Sal, bs = "ts", k = 4) + s(log(Turb), bs = "ts", k = 4) + 
#>     s(log(Chl), bs = "ts", k = 4) + s(log1p(RH), bs = "ts", k = 4)
#> 
#> Parametric coefficients:
#>              Estimate Std. Error t value Pr(>|t|)    
#> (Intercept)    4.2233     0.6754   6.253  1.3e-07 ***
#> Station2       0.4199     0.5900   0.712   0.4804    
#> Station3       0.9390     0.5566   1.687   0.0985 .  
#> Station4       0.7905     0.6061   1.304   0.1988    
#> SeasonSummer   2.3134     1.0156   2.278   0.0275 *  
#> SeasonFall     2.6436     0.9928   2.663   0.0107 *  
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> Approximate significance of smooth terms:
#>                    edf Ref.df     F p-value   
#> s(Temp)      2.864e+00      3 4.239 0.00754 **
#> s(Sal)       1.424e-10      3 0.000 0.38719   
#> s(log(Turb)) 6.140e-01      3 0.663 0.07251 . 
#> s(log(Chl))  2.737e+00      3 3.844 0.01026 * 
#> s(log1p(RH)) 6.360e-01      3 0.633 0.08469 . 
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> R-sq.(adj) =   0.62   Deviance explained = 69.9%
#> GCV = 1.6621  Scale est. = 1.2938    n = 58
cat('\n')
anova(mod)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> log1p(Density) ~ Station + Season + s(Temp, bs = "ts", k = 4) + 
#>     s(Sal, bs = "ts", k = 4) + s(log(Turb), bs = "ts", k = 4) + 
#>     s(log(Chl), bs = "ts", k = 4) + s(log1p(RH), bs = "ts", k = 4)
#> 
#> Parametric Terms:
#>         df     F p-value
#> Station  3 1.158   0.336
#> Season   2 3.647   0.034
#>
#> Approximate significance of smooth terms:
#>                    edf    Ref.df     F p-value
#> s(Temp)      2.864e+00 3.000e+00 4.239 0.00754
#> s(Sal)       1.424e-10 3.000e+00 0.000 0.38719
#> s(log(Turb)) 6.140e-01 3.000e+00 0.663 0.07251
#> s(log(Chl))  2.737e+00 3.000e+00 3.844 0.01026
#> s(log1p(RH)) 6.360e-01 3.000e+00 0.633 0.08469
```

Comparison of Station and Season

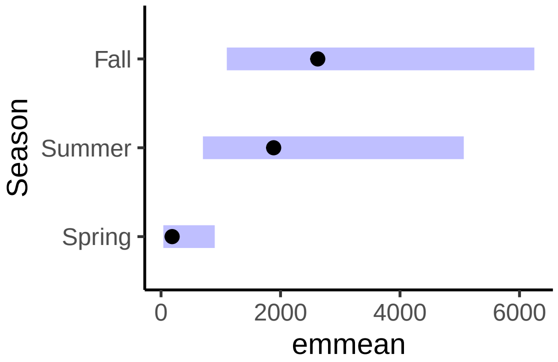
I'm showing "marginal" means – essentially means adjusted for the other predictors, at their mean values.

```
Sta_emms <- emmeans(mod, ~Station, type = 'response',
                    data = spp_analysis$data[spp_data$Species == spp][[1]])
plot(Sta_emms)
```



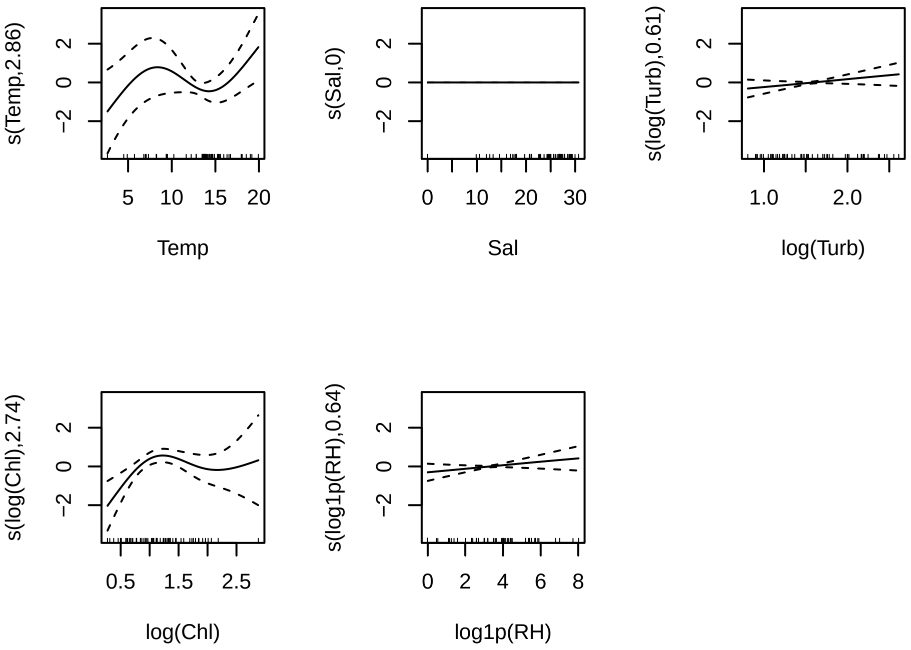
```
pairs(Sta_emms, adjust ='bonferroni')
#> Note: Use 'contrast(regrid(object), ...)' to obtain contrasts of back-transformed estimates
#>  contrast estimate    SE   df t.ratio p.value
#>  1 - 2      -0.420 0.590 45.1  -0.712  1.0000
#>  1 - 3      -0.939 0.557 45.1  -1.687  0.5911
#>  1 - 4      -0.790 0.606 45.1  -1.304  1.0000
#>  2 - 3      -0.519 0.445 45.1  -1.167  1.0000
#>  2 - 4      -0.371 0.465 45.1  -0.797  1.0000
#>  3 - 4       0.148 0.443 45.1   0.335  1.0000
#>
#> Results are averaged over the levels of: Season
#> Note: contrasts are still on the log1p scale
#> P value adjustment: bonferroni method for 6 tests
```

```
Seas_emms <- emmeans(mod, ~Season, type = 'response',
                    data = spp_analysis$data[spp_data$Species == spp][[1]])
plot(Seas_emms)
```

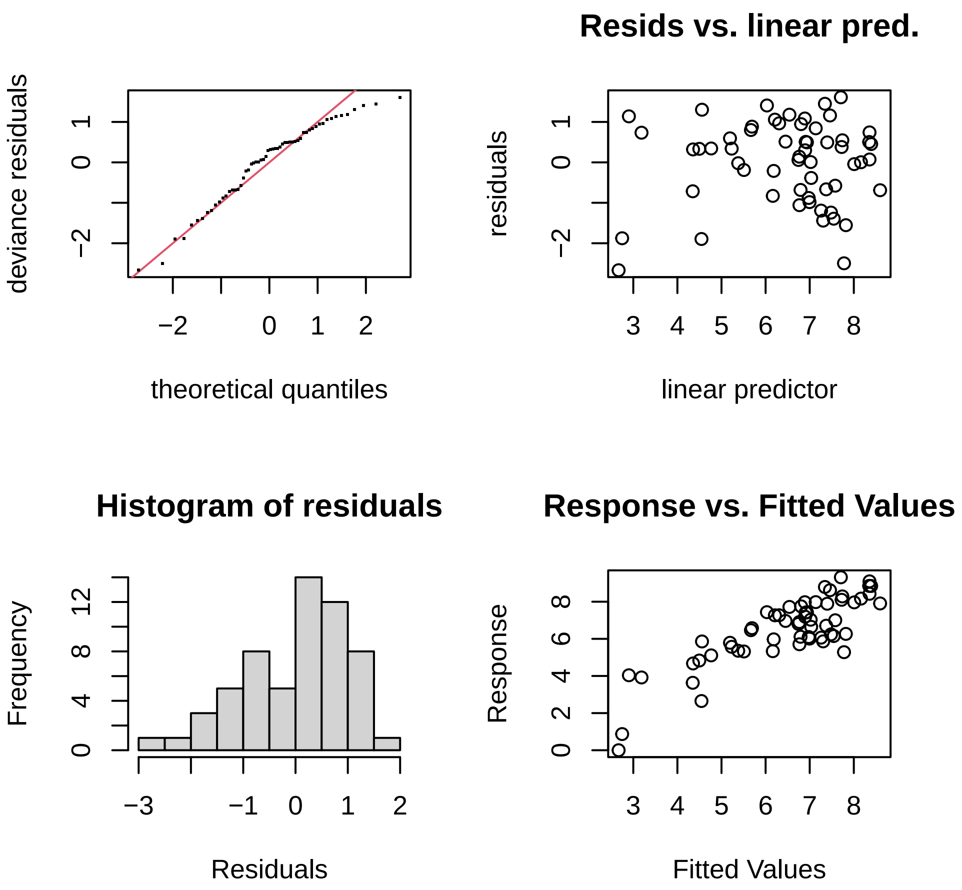


```
pairs(Seas_emms, adjust ='bonferroni')
#> Note: Use 'contrast(regrid(object), ...)' to obtain contrasts of back-transformed estimates
#>  contrast        estimate    SE   df t.ratio p.value
#>  Spring - Summer    -2.31 1.016 45.1  -2.278  0.0826
#>  Spring - Fall      -2.64 0.993 45.1  -2.663  0.0321
#>  Summer - Fall      -0.33 0.399 45.1  -0.827  1.0000
#>
#> Results are averaged over the levels of: Station
#> Note: contrasts are still on the log1p scale
#> P value adjustment: bonferroni method for 3 tests
```

```
oldpar <- par(mfrow = c(2,3))
plot(mod)
par(oldpar)
```



```
oldpar <- par(mfrow = c(2,2))
gam.check(mod)
```



```
#>
#> Method: GCV   Optimizer: magic
#> Smoothing parameter selection converged after 24 iterations.
#> The RMS GCV score gradient at convergence was 1.066668e-07 .
#> The Hessian was positive definite.
#> Model rank =  21 / 21
#>
#> Basis dimension (k) checking results. Low p-value (k-index<1) may
#> indicate that k is too low, especially if edf is close to k'.
#>
#>                    k'      edf k-index p-value
#> s(Temp)      3.00e+00 2.86e+00    1.01   0.445
#> s(Sal)       3.00e+00 1.42e-10    1.20   0.945
#> s(log(Turb)) 3.00e+00 6.14e-01    1.17   0.875
#> s(log(Chl))  3.00e+00 2.74e+00    0.79   0.045 *
#> s(log1p(RH)) 3.00e+00 6.36e-01    0.98   0.465
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
par(oldpar)
```

Eurytemora

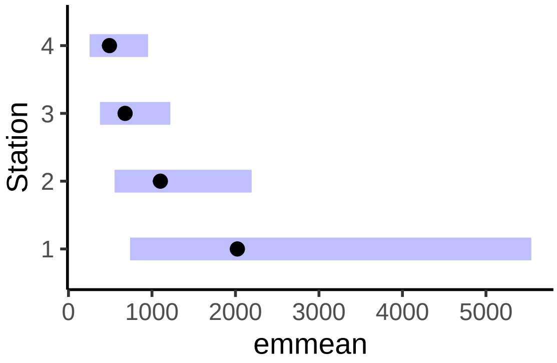
Summary and ANOVA

```
spp =  "Eurytemora"
mod <- spp_analysis$gam_mods[spp_analysis$Species == spp][[1]]
summary(mod)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log1p(Density) ~ Station + Season + s(Temp, bs = "ts", k = 4) +
#>     s(Sal, bs = "ts", k = 4) + s(log(Turb), bs = "ts", k = 4) +
#>     s(log(Chl), bs = "ts", k = 4) + s(log1p(RH), bs = "ts", k = 4)
#>
#> Parametric coefficients:
#>              Estimate Std. Error t value Pr(>|t|)
#> (Intercept)    8.3514     0.5513  15.149  < 2e-16 ***
#> Station2      -0.6082     0.5400  -1.126  0.26586
#> Station3      -1.0918     0.5006  -2.181  0.03432 *
#> Station4      -1.4144     0.5251  -2.694  0.00983 **
#> SeasonSummer  -1.6297     0.6682  -2.439  0.01864 *
#> SeasonFall    -1.7576     0.6663  -2.638  0.01134 *
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>                    edf Ref.df      F p-value
#> s(Temp)      2.122e+00      3  2.261 0.03666 *
#> s(Sal)       2.756e+00      3 15.226 < 2e-16 ***
#> s(log(Turb)) 9.291e-01      3  2.413 0.00556 **
#> s(log(Chl))  1.925e-01      3  0.084 0.25634
#> s(log1p(RH)) 1.243e-10      3  0.000 0.81723
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.522   Deviance explained = 61.4%
#> GCV = 0.92683  Scale est. = 0.73507   n = 58
cat('\n')
anova(mod)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log1p(Density) ~ Station + Season + s(Temp, bs = "ts", k = 4) +
#>     s(Sal, bs = "ts", k = 4) + s(log(Turb), bs = "ts", k = 4) +
#>     s(log(Chl), bs = "ts", k = 4) + s(log1p(RH), bs = "ts", k = 4)
#>
#> Parametric Terms:
#>         df     F p-value
#> Station  3 3.422  0.0248
#> Season   2 3.485  0.0390
#>
#> Approximate significance of smooth terms:
#>                   edf    Ref.df      F p-value
#> s(Temp)      2.122e+00 3.000e+00  2.261 0.03666
#> s(Sal)       2.756e+00 3.000e+00 15.226 < 2e-16
#> s(log(Turb)) 9.291e-01 3.000e+00  2.413 0.00556
#> s(log(Chl))  1.925e-01 3.000e+00  0.084 0.25634
#> s(log1p(RH)) 1.243e-10 3.000e+00  0.000 0.81723
```

Comparison of Station and Season

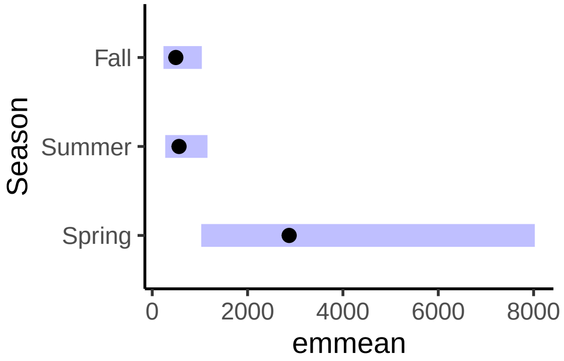
I'm showing "marginal" means – essentially means adjusted for the other predictors, at their mean values.

```
Sta_emms <- emmeans(mod, ~Station, type = 'response',
                    data = spp_analysis$data[spp_data$Species == spp][[1]])
plot(Sta_emms)
```



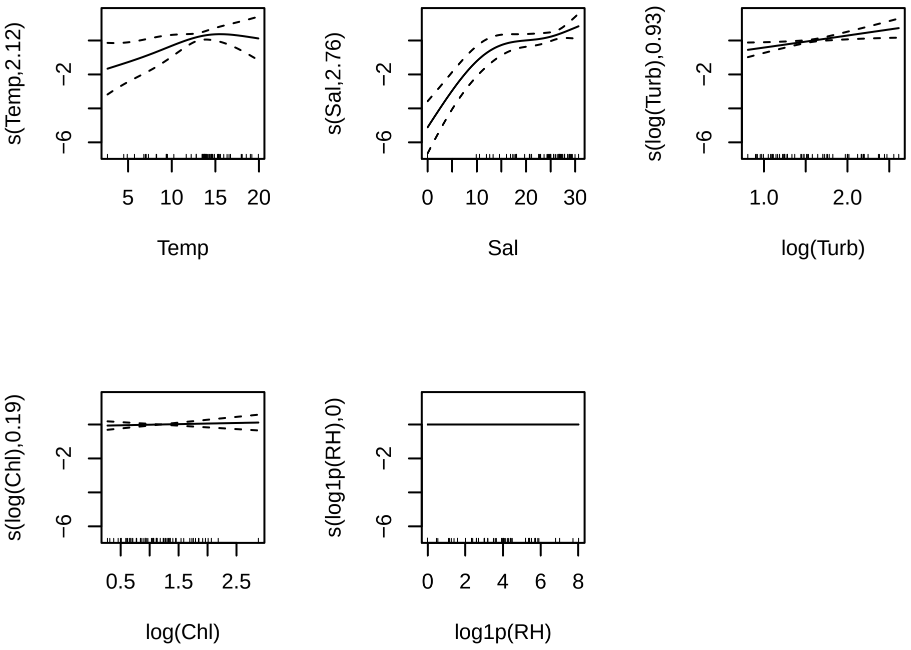
```
pairs(Sta_emms, adjust ='bonferroni')
#> Note: Use 'contrast(regrid(object), ...)' to obtain contrasts of back-transformed estimates
#>  contrast estimate    SE df t.ratio p.value
#>  1 - 2       0.608 0.540 46   1.126  1.0000
#>  1 - 3       1.092 0.501 46   2.181  0.2059
#>  1 - 4       1.414 0.525 46   2.694  0.0590
#>  2 - 3       0.484 0.350 46   1.382  1.0000
#>  2 - 4       0.806 0.354 46   2.280  0.1636
#>  3 - 4       0.323 0.328 46   0.983  1.0000
#>
#> Results are averaged over the levels of: Season
#> Note: contrasts are still on the log1p scale
#> P value adjustment: bonferroni method for 6 tests
```

```
Seas_emms <- emmeans(mod, ~Season, type = 'response',
                    data = spp_analysis$data[spp_data$Species == spp][[1]])
plot(Seas_emms)
```

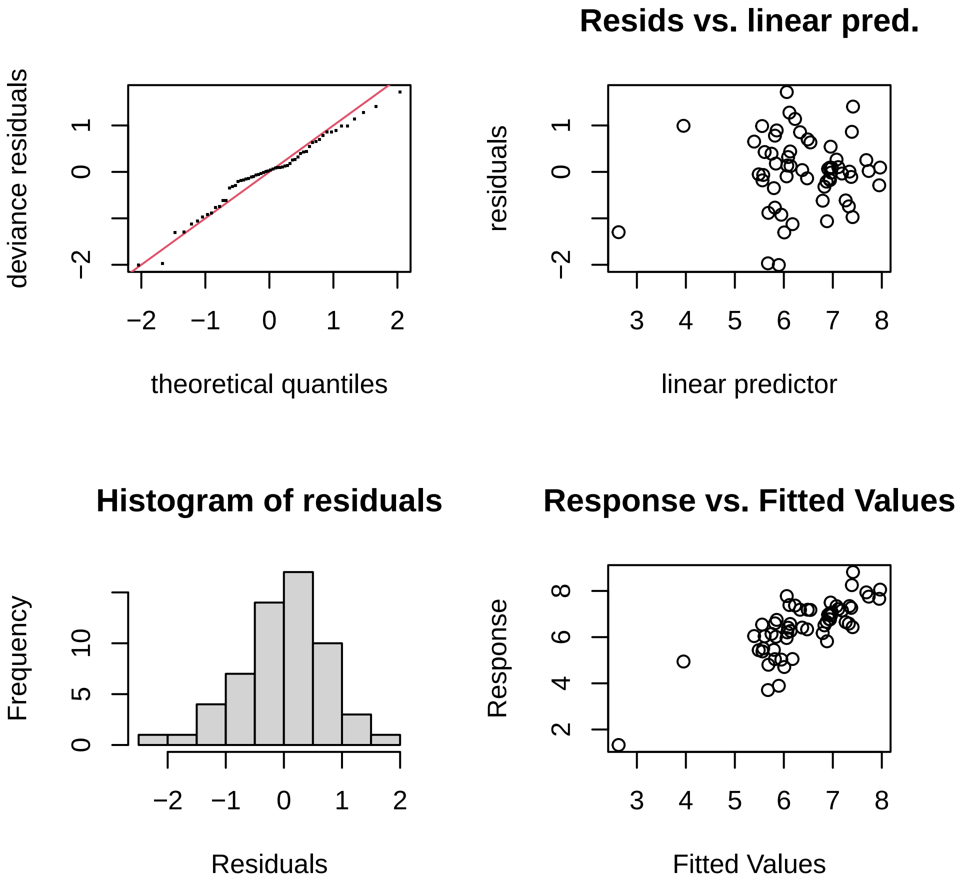


```
pairs(Seas_emms, adjust ='bonferroni')
#> Note: Use 'contrast(regrid(object), ...)' to obtain contrasts of back-transformed estimates
#>  contrast        estimate    SE df t.ratio p.value
#>  Spring - Summer    1.630 0.668 46   2.439  0.0559
#>  Spring - Fall      1.758 0.666 46   2.638  0.0340
#>  Summer - Fall      0.128 0.288 46   0.444  1.0000
#>
#> Results are averaged over the levels of: Station
#> Note: contrasts are still on the log1p scale
#> P value adjustment: bonferroni method for 3 tests
```

```
oldpar <- par(mfrow = c(2,3))
plot(mod)
par(oldpar)
```



```r
oldpar <- par(mfrow = c(2,2))
gam.check(mod)
```



```
#>
#> Method: GCV   Optimizer: magic
#> Smoothing parameter selection converged after 53 iterations.
#> The RMS GCV score gradient at convergence was 1.105685e-07 .
#> The Hessian was positive definite.
#> Model rank =  21 / 21
#>
#> Basis dimension (k) checking results. Low p-value (k-index<1) may
#> indicate that k is too low, especially if edf is close to k'.
#>
#>                    k'      edf k-index p-value
#> s(Temp)      3.00e+00 2.12e+00    1.09    0.74
#> s(Sal)       3.00e+00 2.76e+00    1.16    0.86
#> s(log(Turb)) 3.00e+00 9.29e-01    0.89    0.18
#> s(log(Chl))  3.00e+00 1.92e-01    0.95    0.24
#> s(log1p(RH)) 3.00e+00 1.24e-10    1.07    0.66
par(oldpar)
```

Polychaete

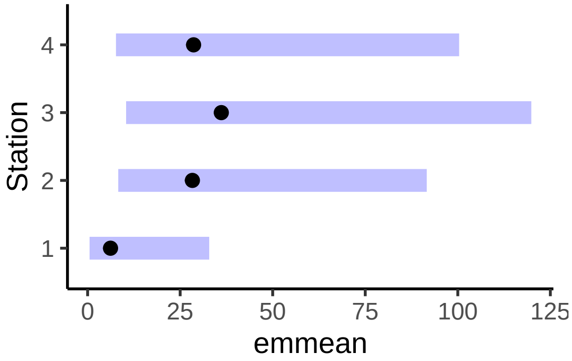
```
spp =  "Polychaete"
mod <- spp_analysis$gam_mods[spp_analysis$Species == spp][[1]]
summary(mod)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> log1p(Density) ~ Station + Season + s(Temp, bs = "ts", k = 4) + 
#>     s(Sal, bs = "ts", k = 4) + s(log(Turb), bs = "ts", k = 4) + 
#>     s(log(Chl), bs = "ts", k = 4) + s(log1p(RH), bs = "ts", k = 4)
#> 
#> Parametric coefficients:
#>              Estimate Std. Error t value Pr(>|t|)    
#> (Intercept)    4.0181     0.9547   4.209 0.000111 ***
#> Station2       1.4049     0.9529   1.474 0.146852    
#> Station3       1.6412     0.9084   1.807 0.077027 .  
#> Station4       1.4158     0.9844   1.438 0.156779    
#> SeasonSummer  -3.6977     1.4839  -2.492 0.016177 *  
#> SeasonFall    -2.3651     1.3738  -1.722 0.091510 .  
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> Approximate significance of smooth terms:
#>                    edf Ref.df     F p-value   
#> s(Temp)      1.846e+00      3 1.305 0.11752   
#> s(Sal)       9.830e-11      3 0.000 0.56768   
#> s(log(Turb)) 7.525e-01      3 0.918 0.05710 . 
#> s(log(Chl))  9.075e-01      3 2.149 0.00831 **
#> s(log1p(RH)) 5.935e-11      3 0.000 0.56093   
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> R-sq.(adj) =  0.437   Deviance explained = 52.1%
#> GCV = 4.5041  Scale est. = 3.7659    n = 58
cat('\n')
anova(mod)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> log1p(Density) ~ Station + Season + s(Temp, bs = "ts", k = 4) + 
#>     s(Sal, bs = "ts", k = 4) + s(log(Turb), bs = "ts", k = 4) + 
#>     s(log(Chl), bs = "ts", k = 4) + s(log1p(RH), bs = "ts", k = 4)
#> 
#> Parametric Terms:
#>         df     F p-value
#> Station  3 1.131  0.3458
#> Season   2 3.820  0.0288
#> 
```

```
#> Approximate significance of smooth terms:
#>                     edf    Ref.df     F p-value
#> s(Temp)       1.846e+00 3.000e+00 1.305 0.11752
#> s(Sal)        9.830e-11 3.000e+00 0.000 0.56768
#> s(log(Turb)) 7.525e-01 3.000e+00 0.918 0.05710
#> s(log(Chl))  9.075e-01 3.000e+00 2.149 0.00831
#> s(log1p(RH)) 5.935e-11 3.000e+00 0.000 0.56093
```

Comparison of Station and Season

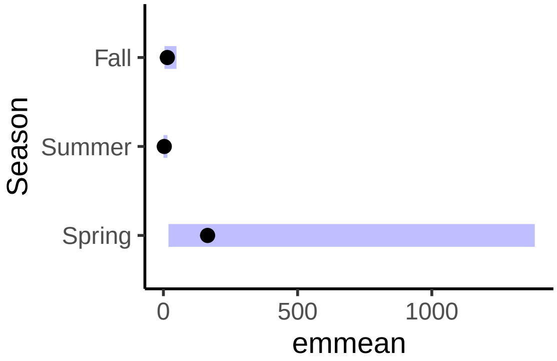
I'm showing "marginal" means – essentially means adjusted for the other predictors, at their mean values.

```
Sta_emms <- emmeans(mod, ~Station, type = 'response',
                    data = spp_analysis$data[spp_data$Species == spp][[1]])
plot(Sta_emms)
```



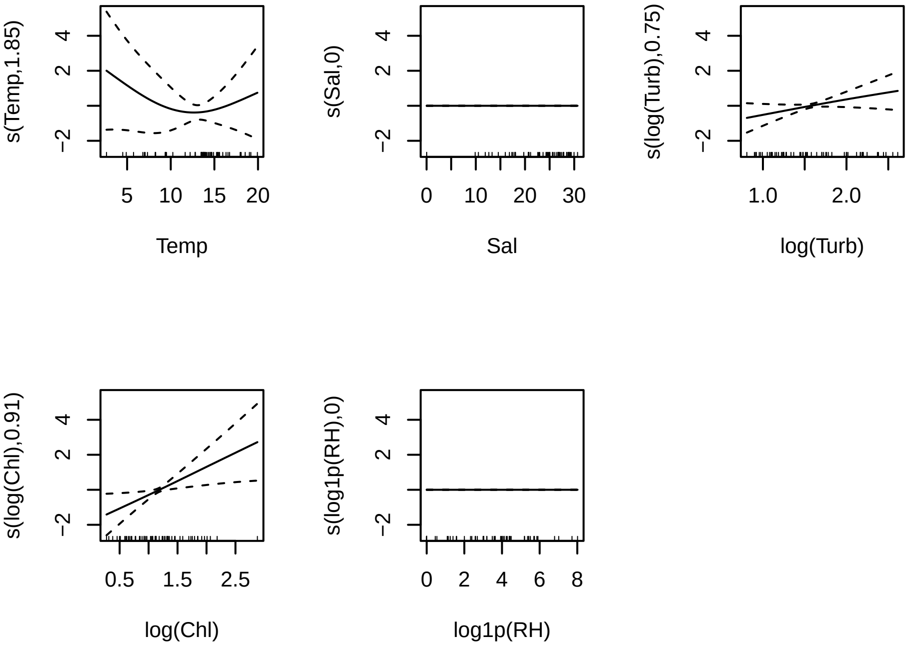
```
pairs(Sta_emms, adjust ='bonferroni')
#> Note: Use 'contrast(regrid(object), ...)' to obtain contrasts of back-transformed estimates
#>  contrast estimate    SE   df t.ratio p.value
#>  1 - 2     -1.4049 0.953 48.5  -1.474  0.8811
#>  1 - 3     -1.6412 0.908 48.5  -1.807  0.4622
#>  1 - 4     -1.4158 0.984 48.5  -1.438  0.9407
#>  2 - 3     -0.2363 0.745 48.5  -0.317  1.0000
#>  2 - 4     -0.0109 0.781 48.5  -0.014  1.0000
#>  3 - 4      0.2254 0.742 48.5   0.304  1.0000
#>
#> Results are averaged over the levels of: Season
#> Note: contrasts are still on the log1p scale
#> P value adjustment: bonferroni method for 6 tests
```

```
Seas_emms <- emmeans(mod, ~Season, type = 'response',
                    data = spp_analysis$data[spp_data$Species == spp][[1]])
plot(Seas_emms)
```

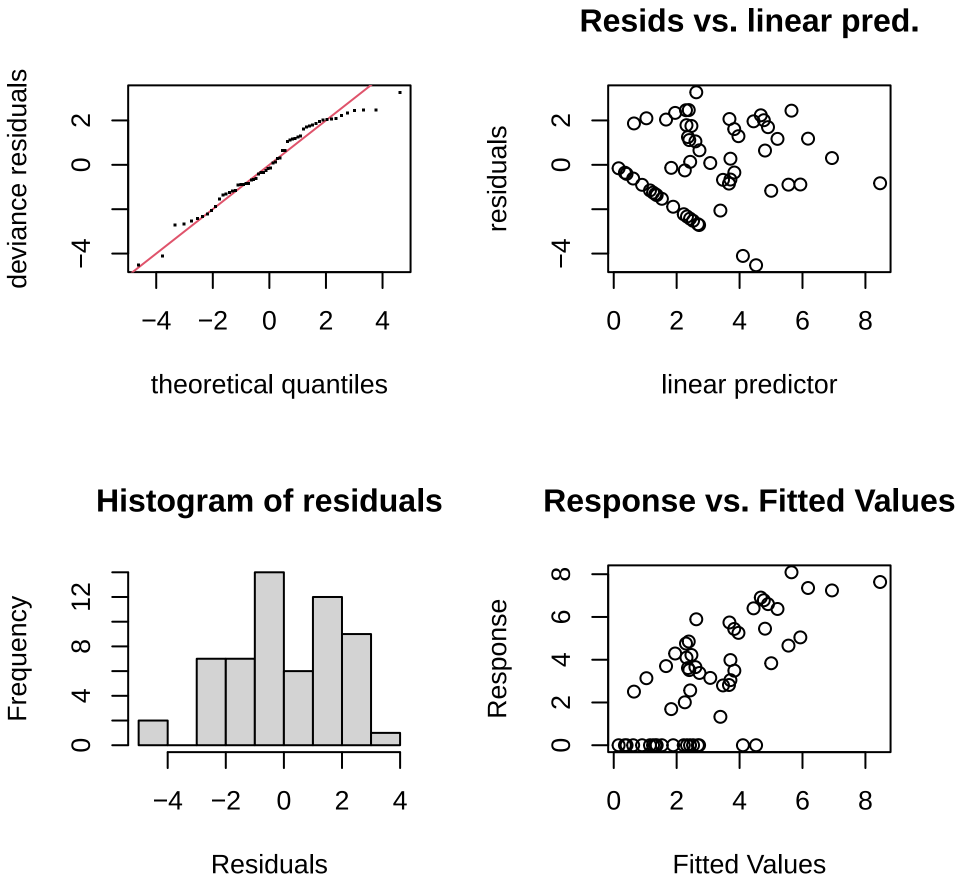


```
pairs(Seas_emms, adjust ='bonferroni')
#> Note: Use 'contrast(regrid(object), ...)' to obtain contrasts of back-transformed estimates
#>  contrast        estimate    SE   df t.ratio p.value
#>  Spring - Summer     3.70 1.484 48.5   2.492  0.0485
#>  Spring - Fall       2.37 1.374 48.5   1.722  0.2745
#>  Summer - Fall      -1.33 0.648 48.5  -2.057  0.1353
#>
#> Results are averaged over the levels of: Station
#> Note: contrasts are still on the log1p scale
#> P value adjustment: bonferroni method for 3 tests
```

```
oldpar <- par(mfrow = c(2,3))
plot(mod)
par(oldpar)
```



```
oldpar <- par(mfrow = c(2,2))
gam.check(mod)
```



```
#>
#> Method: GCV   Optimizer: magic
#> Smoothing parameter selection converged after 20 iterations.
#> The RMS GCV score gradient at convergence was 1.922952e-07 .
#> The Hessian was positive definite.
#> Model rank =  21 / 21
#>
#> Basis dimension (k) checking results. Low p-value (k-index<1) may
#> indicate that k is too low, especially if edf is close to k'.
#>
#>                    k'      edf k-index p-value
#> s(Temp)       3.00e+00 1.85e+00    1.03    0.49
#> s(Sal)        3.00e+00 9.83e-11    1.11    0.73
#> s(log(Turb)) 3.00e+00 7.52e-01    0.76    0.01 **
#> s(log(Chl))  3.00e+00 9.07e-01    0.94    0.26
#> s(log1p(RH)) 3.00e+00 5.93e-11    1.09    0.74
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
par(oldpar)
```

Pseudocal

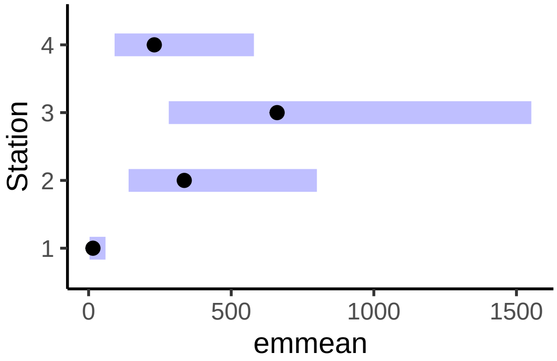
```
spp =  "Pseudocal"
mod <- spp_analysis$gam_mods[spp_analysis$Species == spp][[1]]
summary(mod)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> log1p(Density) ~ Station + Season + s(Temp, bs = "ts", k = 4) + 
#>     s(Sal, bs = "ts", k = 4) + s(log(Turb), bs = "ts", k = 4) + 
#>     s(log(Chl), bs = "ts", k = 4) + s(log1p(RH), bs = "ts", k = 4)
#> 
#> Parametric coefficients:
#>              Estimate Std. Error t value Pr(>|t|)    
#> (Intercept)    4.2268     0.7730   5.468 1.68e-06 ***
#> Station2       3.0302     0.7968   3.803 0.000410 ***
#> Station3       3.7070     0.7186   5.159 4.85e-06 ***
#> Station4       2.6556     0.7700   3.449 0.001195 ** 
#> SeasonSummer  -2.4303     0.9624  -2.525 0.014978 *  
#> SeasonFall    -4.0359     0.9639  -4.187 0.000122 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> Approximate significance of smooth terms:
#>                    edf Ref.df     F  p-value    
#> s(Temp)      2.165e+00      3 8.642 1.86e-05 ***
#> s(Sal)       9.069e-01      3 1.960   0.0109 *  
#> s(log(Turb)) 1.699e+00      3 0.678   0.3018    
#> s(log(Chl))  1.603e-10      3 0.000   0.5275    
#> s(log1p(RH)) 2.440e-10      3 0.000   0.7785    
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> R-sq.(adj) =  0.675   Deviance explained =   73%
#> GCV = 2.1139  Scale est. = 1.7213    n = 58
cat('\n')
anova(mod)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> log1p(Density) ~ Station + Season + s(Temp, bs = "ts", k = 4) + 
#>     s(Sal, bs = "ts", k = 4) + s(log(Turb), bs = "ts", k = 4) + 
#>     s(log(Chl), bs = "ts", k = 4) + s(log1p(RH), bs = "ts", k = 4)
#> 
#> Parametric Terms:
#>         df      F  p-value
#> Station  3  9.161 6.91e-05
#> Season   2 12.793 3.63e-05
#> 
```

```
#> Approximate significance of smooth terms:
#>                    edf    Ref.df     F  p-value
#> s(Temp)      2.165e+00 3.000e+00 8.642 1.86e-05
#> s(Sal)       9.069e-01 3.000e+00 1.960   0.0109
#> s(log(Turb)) 1.699e+00 3.000e+00 0.678   0.3018
#> s(log(Chl))  1.603e-10 3.000e+00 0.000   0.5275
#> s(log1p(RH)) 2.440e-10 3.000e+00 0.000   0.7785
```

Comparison of Station and Season

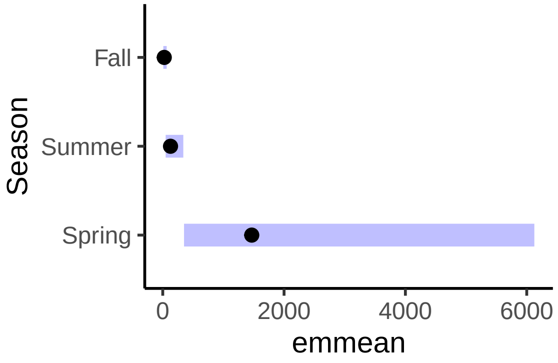
I'm showing "marginal" means – essentially means adjusted for the other predictors, at their mean values.

```
Sta_emms <- emmeans(mod, ~Station, type = 'response',
                    data = spp_analysis$data[spp_data$Species == spp][[1]])
plot(Sta_emms)
```



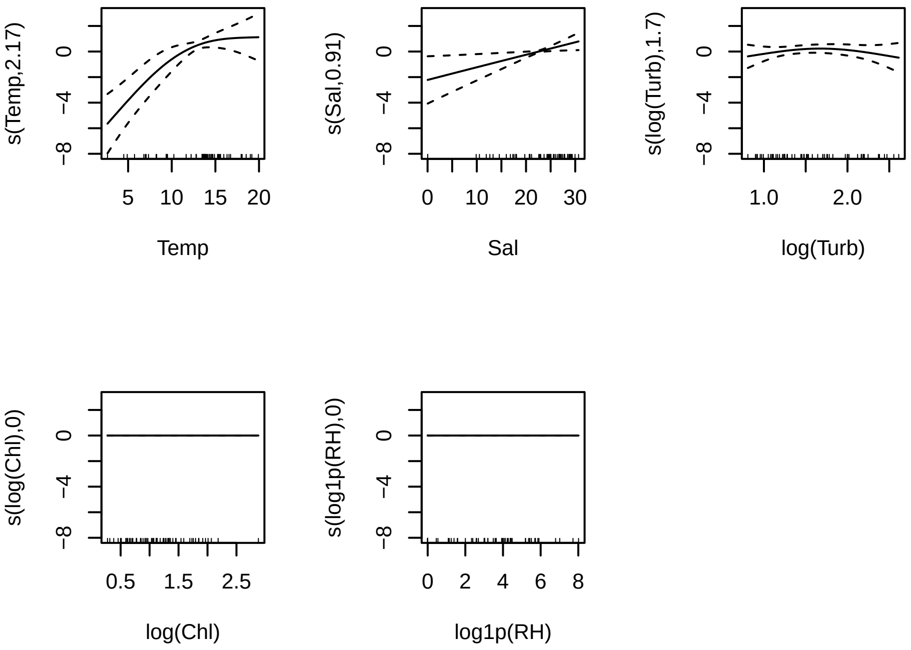
```
pairs(Sta_emms, adjust ='bonferroni')
#> Note: Use 'contrast(regrid(object), ...)' to obtain contrasts of back-transformed estimates
#>  contrast estimate    SE   df t.ratio p.value
#>  1 - 2      -3.030 0.797 47.2  -3.803  0.0025
#>  1 - 3      -3.707 0.719 47.2  -5.159  <.0001
#>  1 - 4      -2.656 0.770 47.2  -3.449  0.0072
#>  2 - 3      -0.677 0.522 47.2  -1.296  1.0000
#>  2 - 4       0.375 0.542 47.2   0.690  1.0000
#>  3 - 4       1.051 0.496 47.2   2.120  0.2357
#>
#> Results are averaged over the levels of: Season
#> Note: contrasts are still on the log1p scale
#> P value adjustment: bonferroni method for 6 tests
```

```
Seas_emms <- emmeans(mod, ~Season, type = 'response',
                    data = spp_analysis$data[spp_data$Species == spp][[1]])
plot(Seas_emms)
```

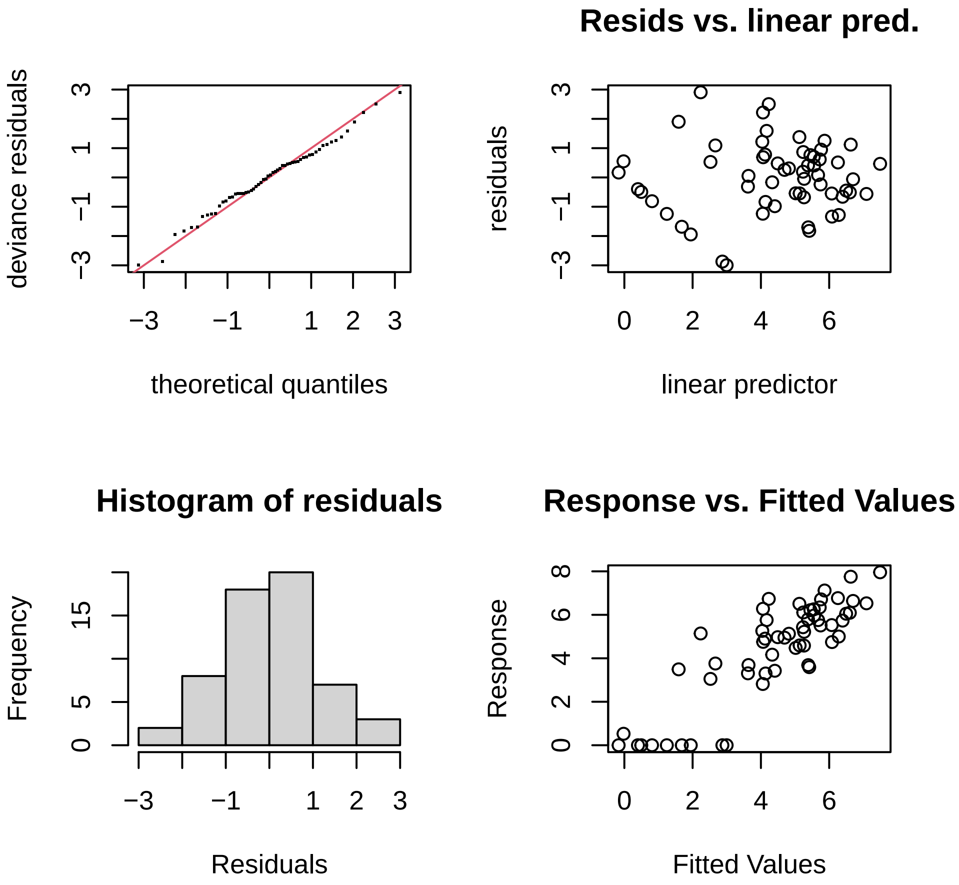


```
pairs(Seas_emms, adjust ='bonferroni')
#> Note: Use 'contrast(regrid(object), ...)' to obtain contrasts of back-transformed estimates
#>  contrast        estimate    SE   df t.ratio p.value
#>  Spring - Summer     2.43 0.962 47.2   2.525  0.0449
#>  Spring - Fall       4.04 0.964 47.2   4.187  0.0004
#>  Summer - Fall       1.61 0.432 47.2   3.716  0.0016
#>
#> Results are averaged over the levels of: Station
#> Note: contrasts are still on the log1p scale
#> P value adjustment: bonferroni method for 3 tests
```

```
oldpar <- par(mfrow = c(2,3))
plot(mod)
par(oldpar)
```



```
oldpar <- par(mfrow = c(2,2))
gam.check(mod)
```



```
#>
#> Method: GCV   Optimizer: magic
#> Smoothing parameter selection converged after 20 iterations.
#> The RMS GCV score gradient at convergence was 8.18504e-08 .
#> The Hessian was positive definite.
#> Model rank =  21 / 21
#>
#> Basis dimension (k) checking results. Low p-value (k-index<1) may
#> indicate that k is too low, especially if edf is close to k’.
#>
#>                     k’      edf k-index p-value
#> s(Temp)      3.00e+00 2.17e+00    0.83   0.085 .
#> s(Sal)       3.00e+00 9.07e-01    0.83   0.095 .
#> s(log(Turb)) 3.00e+00 1.70e+00    1.04   0.545
#> s(log(Chl))  3.00e+00 1.60e-10    1.34   0.995
#> s(log1p(RH)) 3.00e+00 2.44e-10    1.16   0.850
#> ---
#> Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
par(oldpar)
```

Temora

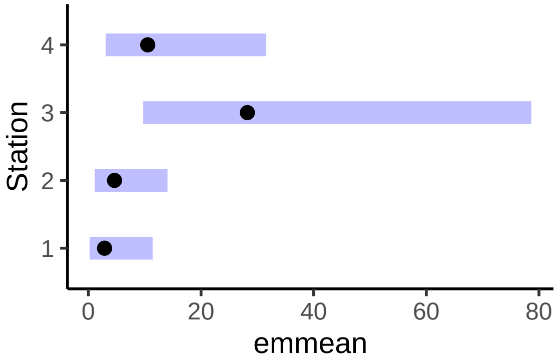
```
spp =  "Temora"
mod <- spp_analysis$gam_mods[spp_analysis$Species == spp][[1]]
summary(mod)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> log1p(Density) ~ Station + Season + s(Temp, bs = "ts", k = 4) + 
#>     s(Sal, bs = "ts", k = 4) + s(log(Turb), bs = "ts", k = 4) + 
#>     s(log(Chl), bs = "ts", k = 4) + s(log1p(RH), bs = "ts", k = 4)
#> 
#> Parametric coefficients:
#>              Estimate Std. Error t value Pr(>|t|)   
#> (Intercept)   1.27819    0.79945   1.599  0.11607   
#> Station2      0.37540    0.74772   0.502  0.61780   
#> Station3      2.02093    0.73328   2.756  0.00811 **
#> Station4      1.09087    0.79124   1.379  0.17405   
#> SeasonSummer -0.47093    0.76757  -0.614  0.54227   
#> SeasonFall   -0.08541    0.70719  -0.121  0.90435   
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> Approximate significance of smooth terms:
#>                   edf Ref.df     F p-value   
#> s(Temp)      2.395e-11      3 0.000 0.63955   
#> s(Sal)       1.071e-01      3 0.043 0.27433   
#> s(log(Turb)) 2.998e-01      3 0.151 0.22275   
#> s(log(Chl))  8.847e-01      3 2.300 0.00588 **
#> s(log1p(RH)) 1.190e-10      3 0.000 0.51267   
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> 
#> R-sq.(adj) =  0.219   Deviance explained = 30.5%
#> GCV = 3.8932  Scale est. = 3.4037    n = 58
cat('\n')
anova(mod)
#> 
#> Family: gaussian 
#> Link function: identity 
#> 
#> Formula:
#> log1p(Density) ~ Station + Season + s(Temp, bs = "ts", k = 4) + 
#>     s(Sal, bs = "ts", k = 4) + s(log(Turb), bs = "ts", k = 4) + 
#>     s(log(Chl), bs = "ts", k = 4) + s(log1p(RH), bs = "ts", k = 4)
#> 
#> Parametric Terms:
#>         df     F p-value
#> Station  3 3.110  0.0344
#> Season   2 0.267  0.7670
#> 
```

```
#> Approximate significance of smooth terms:
#>                    edf    Ref.df     F p-value
#> s(Temp)      2.395e-11 3.000e+00 0.000 0.63955
#> s(Sal)       1.071e-01 3.000e+00 0.043 0.27433
#> s(log(Turb)) 2.998e-01 3.000e+00 0.151 0.22275
#> s(log(Chl))  8.847e-01 3.000e+00 2.300 0.00588
#> s(log1p(RH)) 1.190e-10 3.000e+00 0.000 0.51267
```

Comparison of Station and Season

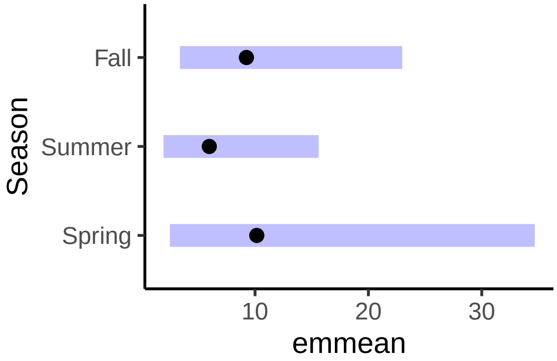
I'm showing "marginal" means – essentially means adjusted for the other predictors, at their mean values.

```
Sta_emms <- emmeans(mod, ~Station, type = 'response',
                    data = spp_analysis$data[spp_data$Species == spp][[1]])
plot(Sta_emms)
```



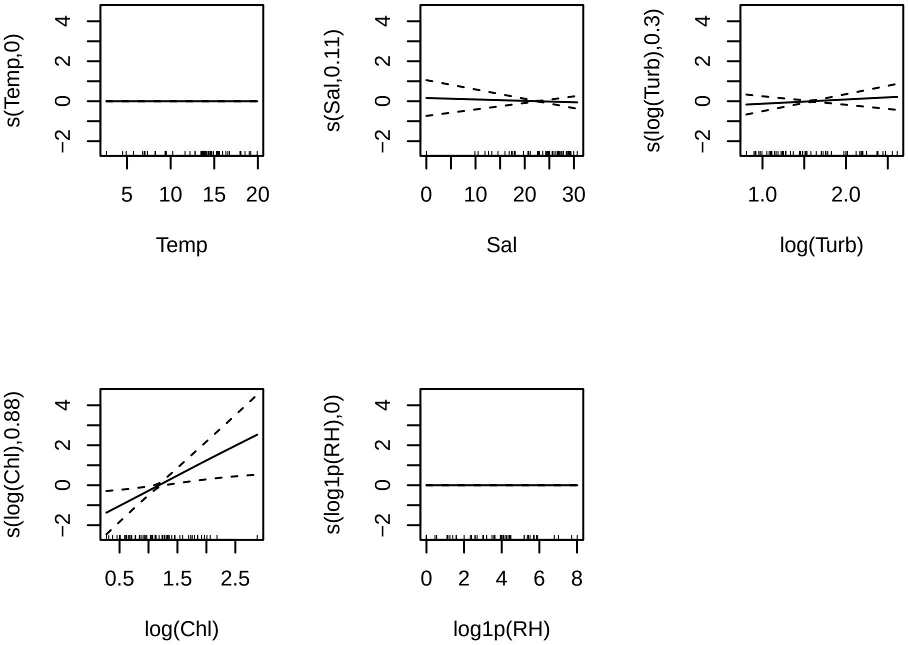
```
pairs(Sta_emms, adjust ='bonferroni')
#> Note: Use 'contrast(regrid(object), ...)' to obtain contrasts of back-transformed estimates
#>  contrast estimate    SE   df t.ratio p.value
#>  1 - 2      -0.375 0.748 50.7  -0.502  1.0000
#>  1 - 3      -2.021 0.733 50.7  -2.756  0.0487
#>  1 - 4      -1.091 0.791 50.7  -1.379  1.0000
#>  2 - 3      -1.646 0.687 50.7  -2.394  0.1223
#>  2 - 4      -0.715 0.711 50.7  -1.007  1.0000
#>  3 - 4       0.930 0.701 50.7   1.327  1.0000
#>
#> Results are averaged over the levels of: Season
#> Note: contrasts are still on the log1p scale
#> P value adjustment: bonferroni method for 6 tests
```

```
Seas_emms <- emmeans(mod, ~Season, type = 'response',
                    data = spp_analysis$data[spp_data$Species == spp][[1]])
plot(Seas_emms)
```

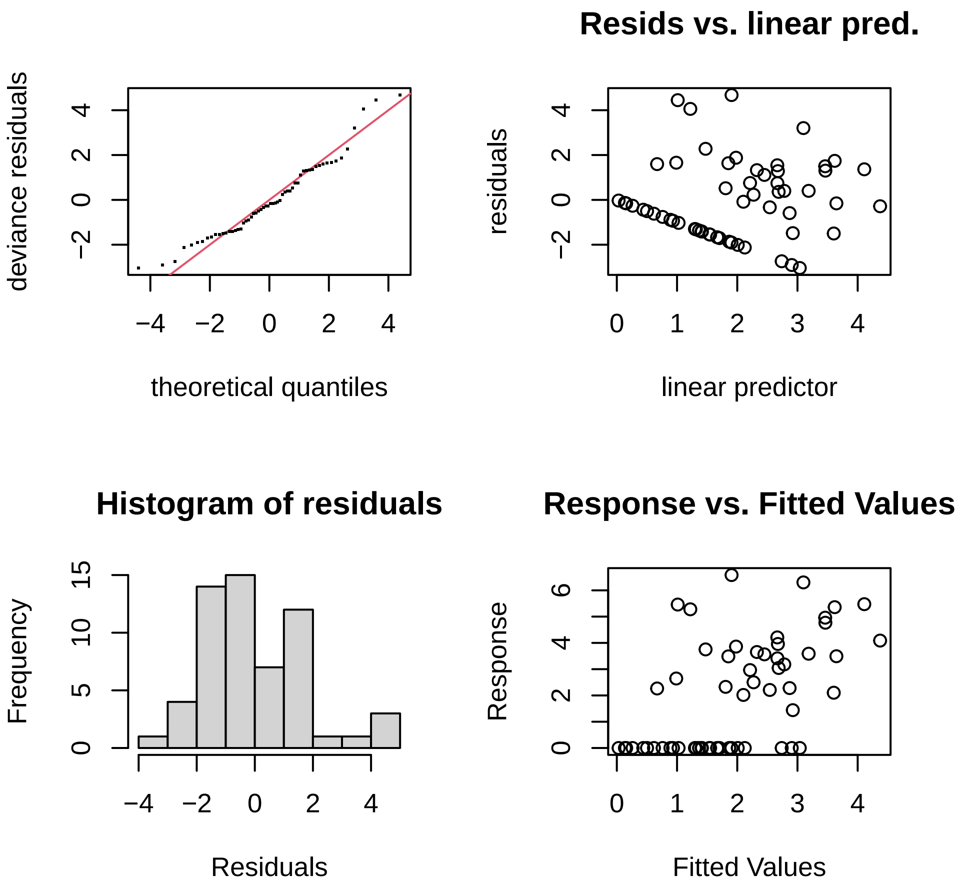


```
pairs(Seas_emms, adjust ='bonferroni')
#> Note: Use 'contrast(regrid(object), ...)' to obtain contrasts of back-transformed estimates
#>  contrast        estimate    SE   df t.ratio p.value
#>  Spring - Summer   0.4709 0.768 50.7   0.614  1.0000
#>  Spring - Fall     0.0854 0.707 50.7   0.121  1.0000
#>  Summer - Fall    -0.3855 0.598 50.7  -0.645  1.0000
#>
#> Results are averaged over the levels of: Station
#> Note: contrasts are still on the log1p scale
#> P value adjustment: bonferroni method for 3 tests
```

```
oldpar <- par(mfrow = c(2,3))
plot(mod)
par(oldpar)
```



```
oldpar <- par(mfrow = c(2,2))
gam.check(mod)
```



```
#>
#> Method: GCV   Optimizer: magic
#> Smoothing parameter selection converged after 16 iterations.
#> The RMS GCV score gradient at convergence was 2.765473e-07 .
#> The Hessian was positive definite.
#> Model rank =  21 / 21
#>
#> Basis dimension (k) checking results. Low p-value (k-index<1) may
#> indicate that k is too low, especially if edf is close to k'.
#>
#>                     k'      edf k-index p-value
#> s(Temp)       3.00e+00 2.39e-11    1.07    0.66
#> s(Sal)        3.00e+00 1.07e-01    1.13    0.85
#> s(log(Turb)) 3.00e+00 3.00e-01    1.10    0.76
#> s(log(Chl))  3.00e+00 8.85e-01    0.97    0.32
#> s(log1p(RH)) 3.00e+00 1.19e-10    1.25    0.94
par(oldpar)
```