How Should We Decide Whether to use Gamma or Lognormal Models?"

Curtis C. Bohlen, Casco Bay Estuary Partnership

2/11/2022

Contents

Introduction

This notebook is a another follow up on previous analyses. Here I focus on whether to use lognormal (Gaussian models on log transformed data) or Gamma
models with log link.

The two models are close cousins, and so it is not surprising that results are often (but not always) similar. There's actually a fairly large literature on this question.

While I'm at it, I model fitting GAM models with random factors.

Load Libraries

```
library(tidyverse)
library(readxl)
library(mgcv)      # for GAM models
library(emmeans)   # For extracting useful "marginal" model summaries

theme_set(theme_classic())
```

Folder References

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

Load Data

```
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)
```

Modeling with GAMs

I'm trying to mimic the linear models from the manuscript, only using GAMs instead of linear models.

I've made three big changes:

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 'GAM' models These allow fitting fairly arbitrary smooth relationships between predictors and response variables.
3. Since I'm using GAMs, I also explore alternatives to assumptions of normally distributed errors. Here I only look at the gamma GAM with a log link.

Repeated Measures / Mixed models?

I'm running `Year` as a random effect, but `Station` as a fixed effect. The reason is that I think we DO want to be able to interpret differences in abundance by location in the estuary. Station is one measure of location. We have no real interest in interpreting year to year variation, only accounting for it in the models. I've run these models including Station as a random factor too, and it makes little difference to the other conclusions of any of these models.

I poked at fitting random effects using both `gam()` and `gamm()` functions. For "simple" random effects, either one can handle it, but the R output is different. I prefer `gam()` , largely because the R output is simpler. It's worth pointing out that I could not get `gamm()` to fit quite the same statistical model as the one I fit with `gam()`. Results are similar, but not quantitatively identical.

For a quick intro to random effects models using `gam()`, here: https://fromthebottomoftheheap.net/2021/02/02/random-effects-in-gams/ But that discussion is mostly aimed at estimating variance components in the context of (linear) GLM models.

Total Density Data

Selection of Predictors

All the following models use the same predictor variables. These are based on the linear models in the manuscript. I STILL have not tried to fit any interaction terms between predictors.

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.

- Linear (Fixed) Effects
  - Station. This will fit three parameters, to distinguish between the four Stations. By default, the contrasts used in the model fitting compare each of stations 2,3, and 4 against station 1, which is fit as the model intercept. This convention can be altered by (1) specifying a model without an intercept (`0 + Station + ...`) or by specifying a different contrast for the Stations. I don't bother, principally because I am going to look at marginal means (predictions) by station (rather than the model coefficients themselves), and the marginal means won't be affected by how Station is fit.
- Random Factor
  - Year. I added Year as a random factor. This is fit in `gam()` as a "smooth" term with `s(Station, bs = 're')`. The `bs = 're'` specifies a "random effect". I don't understand the mathematical details. In `gamm()`, this is fit as a conventional random factor, using `random = list(Yearf = ~ 1)`.
- Smoothed Terms
  - Temperature (untransformed)
  - Salinity (untransformed)
  - log(Turbidity)
  - log(Chlorophyll)
  - log(River Herring + 1). I had to add one to deal with zero counts.

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.

Our best model choices should reflect those two features of our data. I considered several model strategies (see the other notebooks for more discussion). Here I focus on comparing two types of models.

1. Model assuming normally distributed errors, on log transformed data.
2. A Gamma family GAM with a Log link.

Testing Different Models

Log Transform (lognormal model)

I'll fit this both with `gam()` and with `gamm()`, to provide code examples.

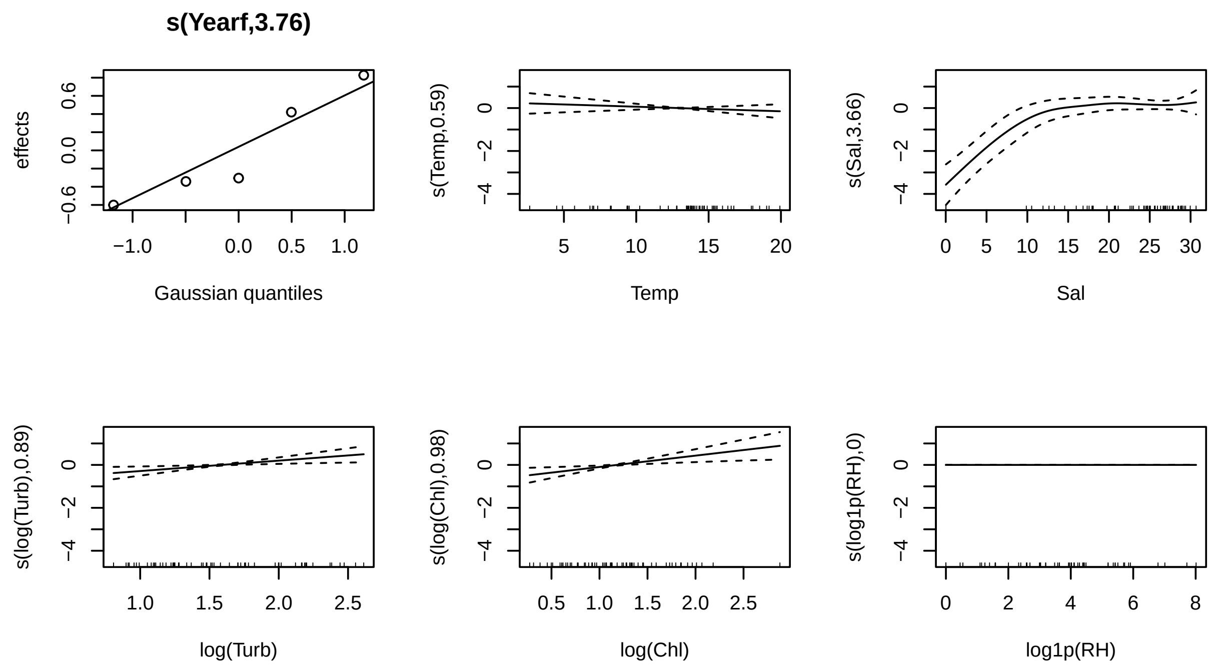
```
combined_density_gam_l <- gam(log(combined_density) ~
                                Station +
                                s(Yearf, bs = 're') +
                                s(Temp, bs="ts") +
                              s(Sal, bs="ts") +
                              s(log(Turb), bs="ts") +
                              s(log(Chl), bs="ts") +
                              s(log1p(RH),bs="ts"),
                            data = base_data, family = 'gaussian')
summary(combined_density_gam_l)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log(combined_density) ~ Station + s(Yearf, bs = "re") + 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)   8.4668     0.3902  21.699   <2e-16 ***
#> Station2     -0.6724     0.3241  -2.075   0.0439 *
#> Station3     -0.4182     0.3041  -1.375   0.1760
#> Station4     -0.7521     0.3251  -2.313   0.0254 *
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>                    edf Ref.df      F  p-value
#> s(Yearf)     3.757e+00      4 11.774 9.50e-07 ***
#> s(Temp)      5.906e-01      9  0.171  0.22136
#> s(Sal)       3.656e+00      9 10.437 2.86e-05 ***
#> s(log(Turb)) 8.916e-01      9  1.152  0.00471 **
#> s(log(Chl))  9.796e-01      9  3.175  0.00274 **
#> s(log1p(RH)) 8.468e-08      9  0.000  0.29545
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.685   Deviance explained = 75.6%
#> GCV = 0.3934  Scale est. = 0.29929   n = 58
```

Fitting with `gam()` The random effect is statistically significant, meaning we probably should not drop it from the model. But we knew that already…

Otherwise, we see Salinity, Turbidity, and Chlorophyll are "Significant" predictors.

Also, I finally noticed that two observations have missing values for River Herring, so those observations are dropped from the analysis, giving us n = 58. One of those is a low salinity sample.

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

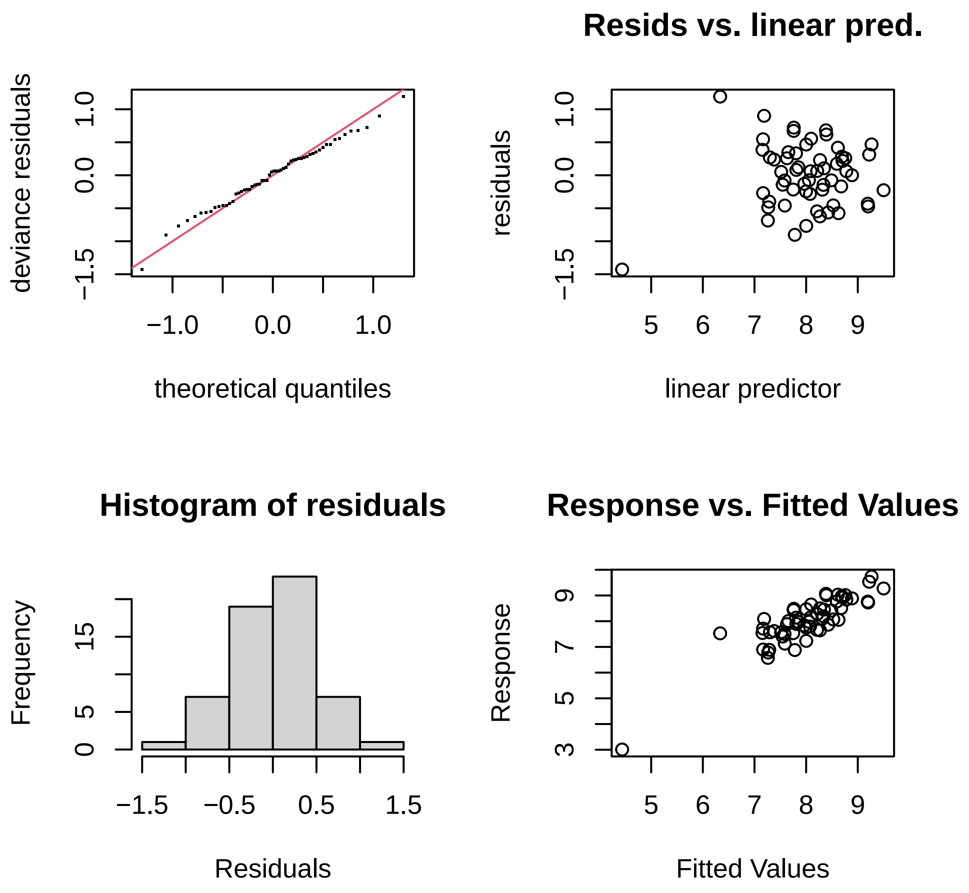


```
par(oldpar)
```

This model suggests:

1. Low salinity observations are different
2. Abundance increases with (log of) turbidity
3. Abundance increases with (log of) chlorophyll.

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



```
#>
#> Method: GCV   Optimizer: magic
#> Smoothing parameter selection converged after 21 iterations.
#> The RMS GCV score gradient at convergence was 1.214577e-07 .
#> The Hessian was positive definite.
#> Model rank =  54 / 54
#>
#> 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(Yearf)      5.00e+00 3.76e+00      NA      NA
#> s(Temp)       9.00e+00 5.91e-01    1.16    0.85
#> s(Sal)        9.00e+00 3.66e+00    0.98    0.40
#> s(log(Turb)) 9.00e+00 8.92e-01    0.84    0.08 .
#> s(log(Chl))  9.00e+00 9.80e-01    1.14    0.84
#> s(log1p(RH)) 9.00e+00 8.47e-08    0.99    0.47
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
par(oldpar)
```

The extreme low fitted value is still a problem, but that's one of the low salinity observations.

Fitting with `gamm()` The random effects have to be specified as a list of one-sided formulae. I find the output of `gamm()` much more confusing, so if I can get away with it, I'm likely to want to stick to `gam()`....

```
combined_density_gamm_l <- gamm(log(combined_density) ~
                                  Station +
                                  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')
```

The object returned by `gamm()` combines both an `lme` object and a `gam` object.

```
summary(combined_density_gamm_l)
#>     Length Class Mode
#> lme 18     lme   list
#> gam 31     gam   list
```

Generally, you need to refer to one or the other component explicitly to make sense of the analysis. I mostly focus on the `$gam` component. I think the `$lme` component is mostly hidden fitting machinery. I have never been able to make much sense of the `$lme` component, although there are several references that provide details of how to pull out important model diagnostics.

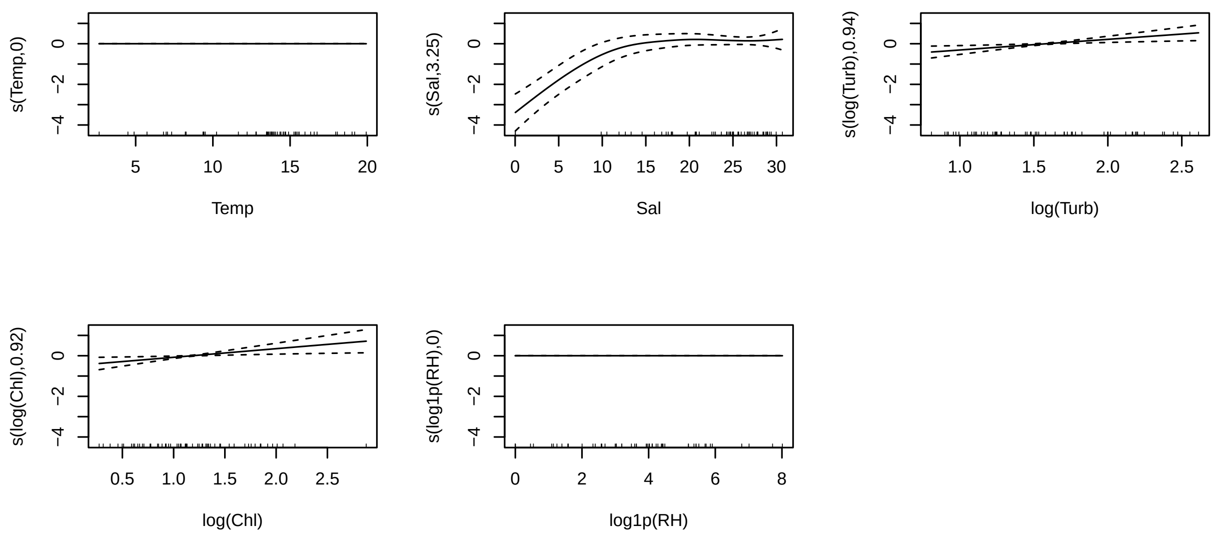
```
summary(combined_density_gamm_l$gam)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log(combined_density) ~ Station + s(Temp, bs = "ts") + s(Sal,
#>     bs = "ts") + s(log(Turb), bs = "ts") + s(log(Chl), bs = "ts") +
#>     s(log1p(RH), bs = "ts")
#>
#> Parametric coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept)   8.3519     0.3106  26.891   <2e-16 ***
#> Station2     -0.5209     0.2811  -1.853   0.0699 .
#> Station3     -0.2811     0.2681  -1.049   0.2995
#> Station4     -0.5811     0.2776  -2.094   0.0415 *
#> ---
#> 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)      4.081e-06      9 0.000 0.36455
#> s(Sal)       3.248e+00      9 7.377 < 2e-16 ***
#> s(log(Turb)) 9.389e-01      9 0.978 0.00449 **
#> s(log(Chl))  9.156e-01      9 0.798 0.01012 *
#> s(log1p(RH)) 4.971e-05      9 0.000 0.29991
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#>
#> R-sq.(adj) =  0.338
#>   Scale est. = 0.29049   n = 58
```

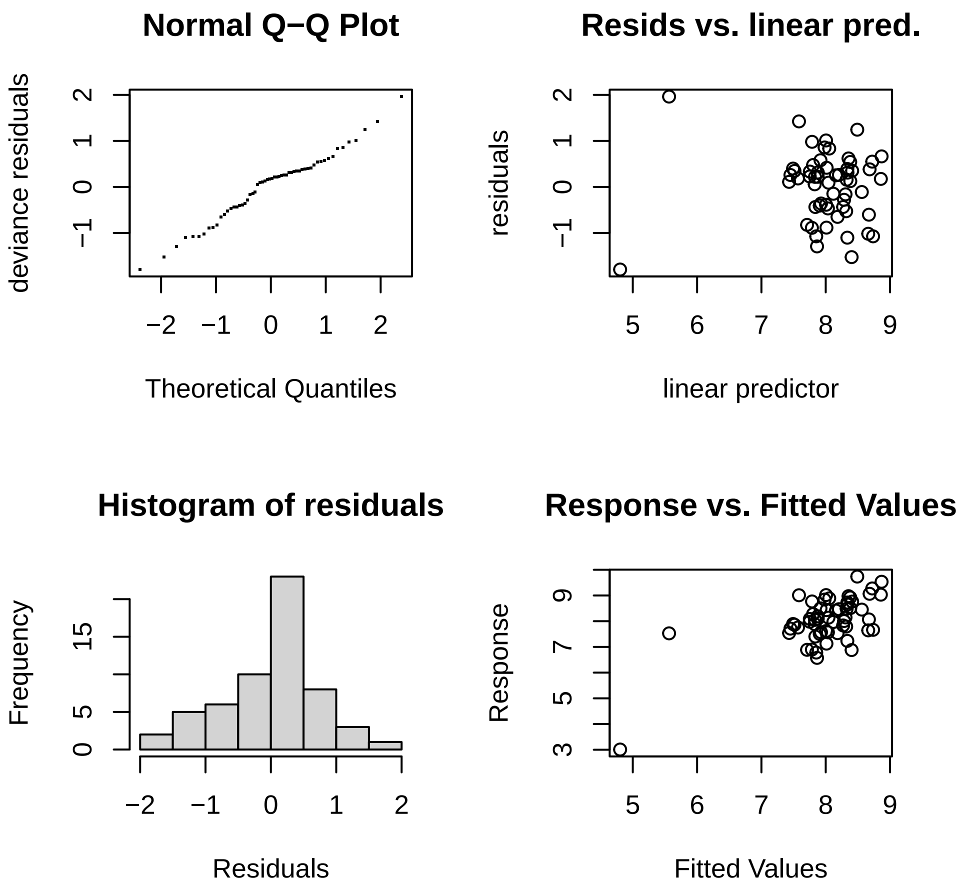
Results appear similar, but not identical to those from the prior model. I believe the difference is due to how the two functions incorporate (or fail to incorporate) levels of the random factor (here, the year). But I'm not 100% sure.... The two functions use different fitting algorithms, so it is also possible that they converge on slightly different solutions.

You also have to specify that you want to plot the `gam` component of the object returned by `gamm()`.

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



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



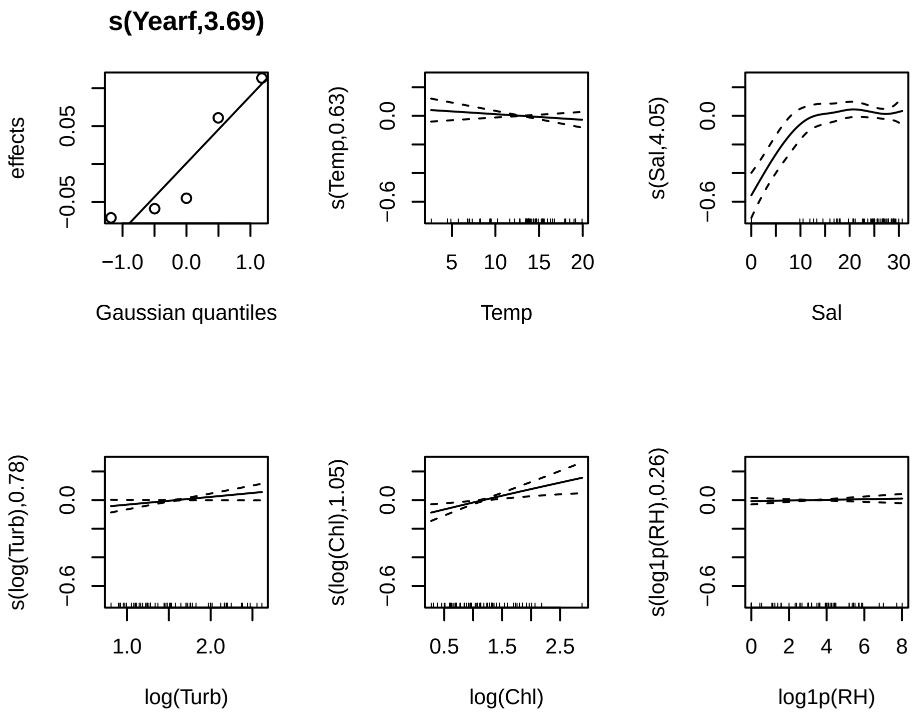
```
#>
#> 'gamm' based fit - care required with interpretation.
#> Checks based on working residuals may be misleading.
#> Basis dimension (k) checking results. Low p-value (k-index<1) may
#> indicate that k is too low, especially if edf is close to k'.
#>
#>                     k'      edf k-index p-value
#> s(Temp)      9.00e+00 4.08e-06    1.02    0.50
#> s(Sal)       9.00e+00 3.25e+00    1.06    0.64
#> s(log(Turb)) 9.00e+00 9.39e-01    0.92    0.22
#> s(log(Chl))  9.00e+00 9.16e-01    0.99    0.49
#> s(log1p(RH)) 9.00e+00 4.97e-05    0.88    0.10
par(oldpar)
```

Gamma Model, Log Link

A Gamma GLM is often selected because it can model a wide range of positive valued random variables, where variability increases with expected value. After doing a little more reading and thinking, I settled on the log link as probably making the most sense here.

```
combined_density_gam_g <- gam(log(combined_density) ~
                               Station +
                               s(Yearf, bs = 're') +
                               s(Temp, bs="ts") +
                               s(Sal, bs="ts") +
                               s(log(Turb), bs="ts") +
                               s(log(Chl), bs="ts") +
                               s(log1p(RH),bs="ts"),
                             data = base_data, family = Gamma(link = 'log'))
summary(combined_density_gam_g)
#>
#> Family: Gamma
#> Link function: log
#>
#> Formula:
#> log(combined_density) ~ Station + s(Yearf, bs = "re") + 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)  2.13364    0.05900  36.162   <2e-16 ***
#> Station2    -0.08493    0.05387  -1.577   0.1221
#> Station3    -0.05644    0.05053  -1.117   0.2701
#> Station4    -0.09980    0.05397  -1.849   0.0712 .
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>                 edf Ref.df     F  p-value
#> s(Yearf)     3.6852      4 8.263 1.43e-05 ***
#> s(Temp)      0.6307      9 0.197  0.19650
#> s(Sal)       4.0526      9 9.504 2.57e-05 ***
#> s(log(Turb)) 0.7830      9 0.549  0.02696 *
#> s(log(Chl))  1.0470      9 3.488  0.00157 **
#> s(log1p(RH)) 0.2602      9 0.048  0.19715
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.631   Deviance explained = 69.1%
#> GCV = 0.010915  Scale est. = 0.0078685  n = 58
```

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

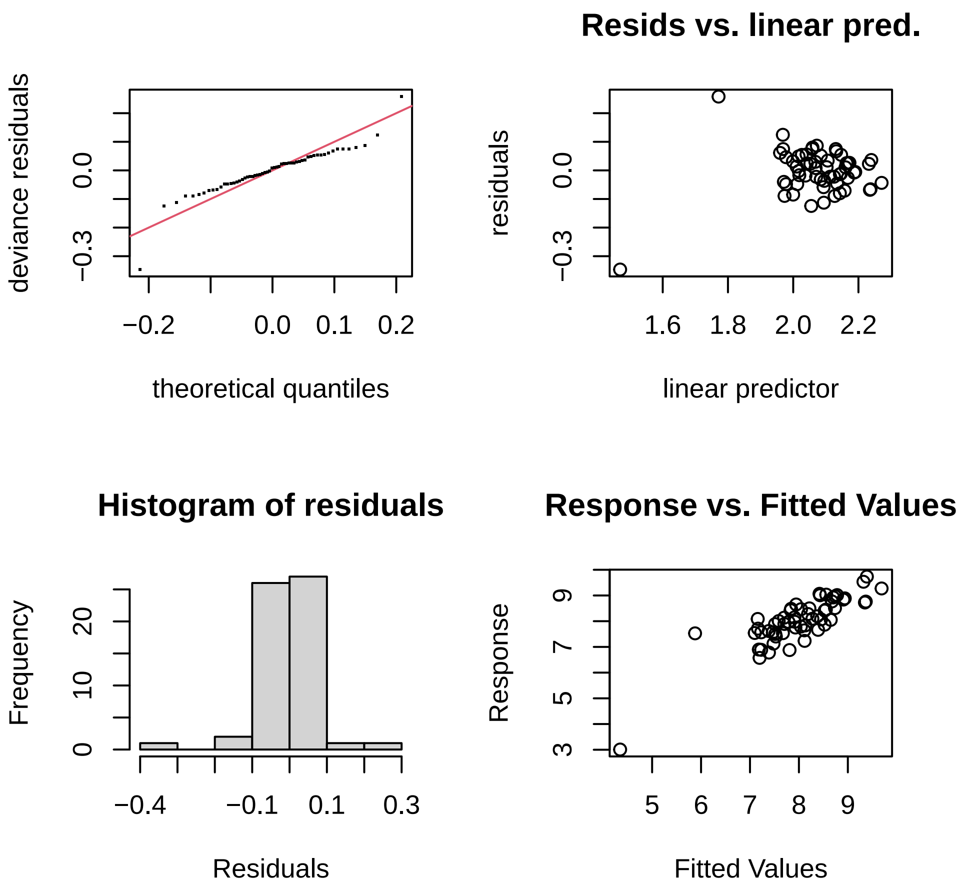


```
par(oldpar)
```

So,

1. Low salinity samples are different. After that the effect of salinity is relatively small.
2. Abundance increases with turbidity.
3. Abundance increases with Chlorophyll

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



```
#>
#> Method: GCV   Optimizer: outer newton
#> full convergence after 10 iterations.
#> Gradient range [-2.135249e-09,1.828163e-10]
#> (score 0.01091475 & scale 0.007868505).
#> Hessian positive definite, eigenvalue range [1.1611e-05,0.0001010756].
#> Model rank =  54 / 54
#>
#> 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(Yearf)     5.000 3.685      NA      NA
#> s(Temp)      9.000 0.631    1.18    0.92
#> s(Sal)       9.000 4.053    1.10    0.71
#> s(log(Turb)) 9.000 0.783    0.90    0.20
#> s(log(Chl))  9.000 1.047    1.06    0.74
#> s(log1p(RH)) 9.000 0.260    0.88    0.17
par(oldpar)
```

This model has OK residual structure, except for that one extreme low predicted value. As we will see below, that one weird sample balloons the confidence intervals on this model a lot more than on the lognormal model.

Conclusions

1. One very low abundance, low salinity sample dominates the relationship of abundance and salinity. Without that low value, I doubt there would be much of a pattern to point to. Nevertheless, that relationship is real.
2. Abundance increases strongly with turbidity.
3. In most models, there is also a positive association between zooplankton abundance and chlorophyll.

Is there a Rational Way to decide which model works "Better"?

Lets compare graphic output, showing prediction lines, 95% confidence intervals for those predictions, and the raw data.

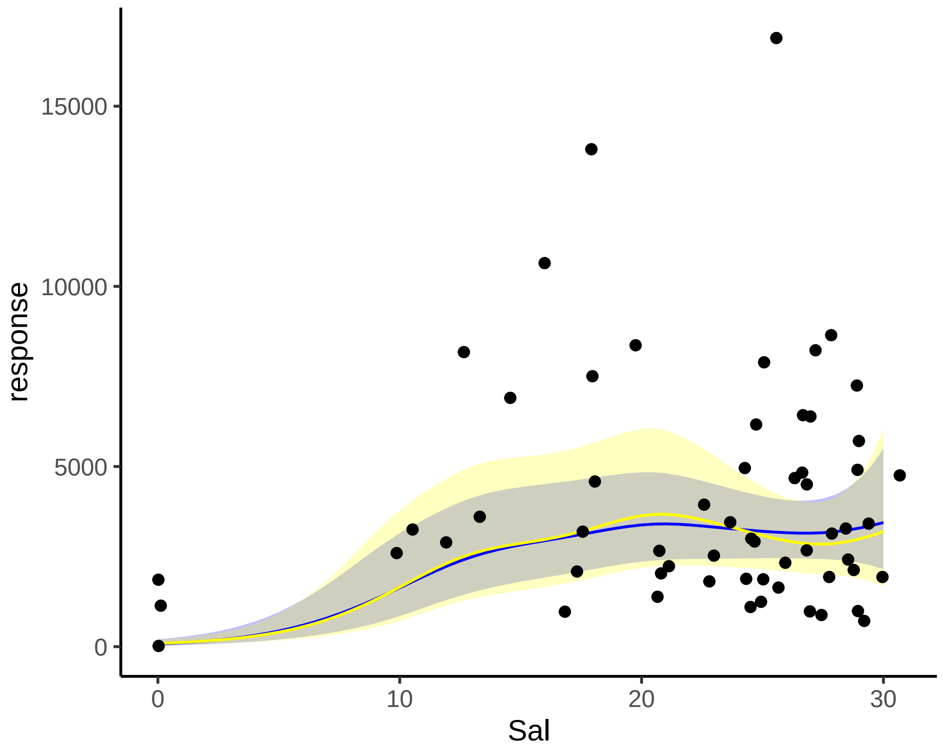
We'll follow the convention that the log linear model is in blue, and the gamma a model is in yellow.

Salinity

```r
emm_sal_l <- as.data.frame(emmeans(combined_density_gam_l, "Sal",
                    at = list(Sal = seq(0, 30, 0.5)),
                    cov.reduce = median,
                    type = 'response'))

emm_sal_g <- as.data.frame(emmeans(combined_density_gam_g, "Sal",
                    at = list(Sal = seq(0, 30, 0.5)),
                    cov.reduce = median,
                    type = 'response'))
```

```r
ggplot() +
  geom_ribbon(data = emm_sal_l, mapping = aes(x = Sal,
                                              ymin = lower.CL,
                                              ymax = upper.CL),
              fill = 'blue', alpha = 0.25) +
  geom_ribbon(data = emm_sal_g, mapping = aes(x = Sal,
                                              ymin = lower.CL,
                                              ymax = upper.CL),
              fill = 'yellow', alpha = 0.25) +
  geom_line(data = emm_sal_l, mapping = aes(x = Sal, y = response),
            color = 'blue') +
  geom_line(data = emm_sal_g, mapping = aes(x = Sal, y = response),
            color = 'yellow') +
  geom_point(data = base_data, mapping = aes(x = Sal, y = combined_density))
```



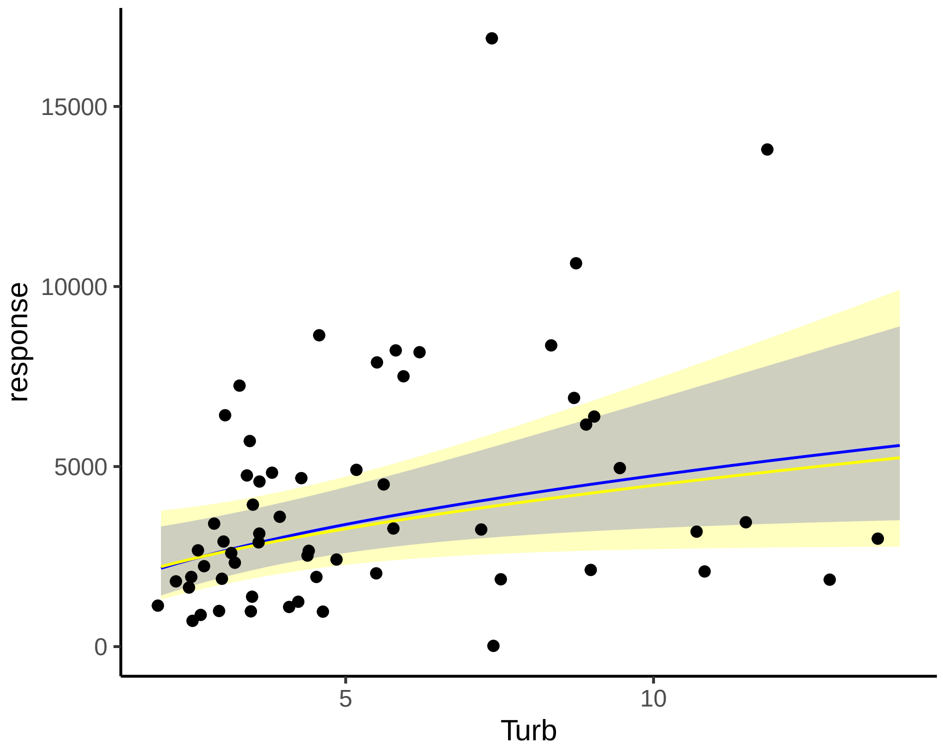
The two models provide statistically similar predictions. Here, the log-linear model fit has both fewer wiggles and a narrower error band.

Turbidity

```
emm_turb_l <- as.data.frame(emmeans(combined_density_gam_l, "Turb",
                    at = list(Turb = seq(2, 14, 0.25)),
                    cov.reduce = median,
                    type = 'response'))

emm_turb_g <- as.data.frame(emmeans(combined_density_gam_g, "Turb",
                    at = list(Turb = seq(2, 14, 0.25)),
                    cov.reduce = median,
                    type = 'response'))
```

```
ggplot() +
  geom_ribbon(data = emm_turb_l, mapping = aes(x = Turb,
                                               ymin = lower.CL,
                                               ymax = upper.CL),
              fill = 'blue', alpha = 0.25) +
  geom_ribbon(data = emm_turb_g, mapping = aes(x = Turb,
                                               ymin = lower.CL,
                                               ymax = upper.CL),
              fill = 'yellow', alpha = 0.25) +
  geom_line(data = emm_turb_l, mapping = aes(x = Turb, y = response),
            color = 'blue') +
  geom_line(data = emm_turb_g, mapping = aes(x = Turb, y = response),
            color = 'yellow') +
  geom_point(data = base_data, mapping = aes(x = Turb, y = combined_density))
```



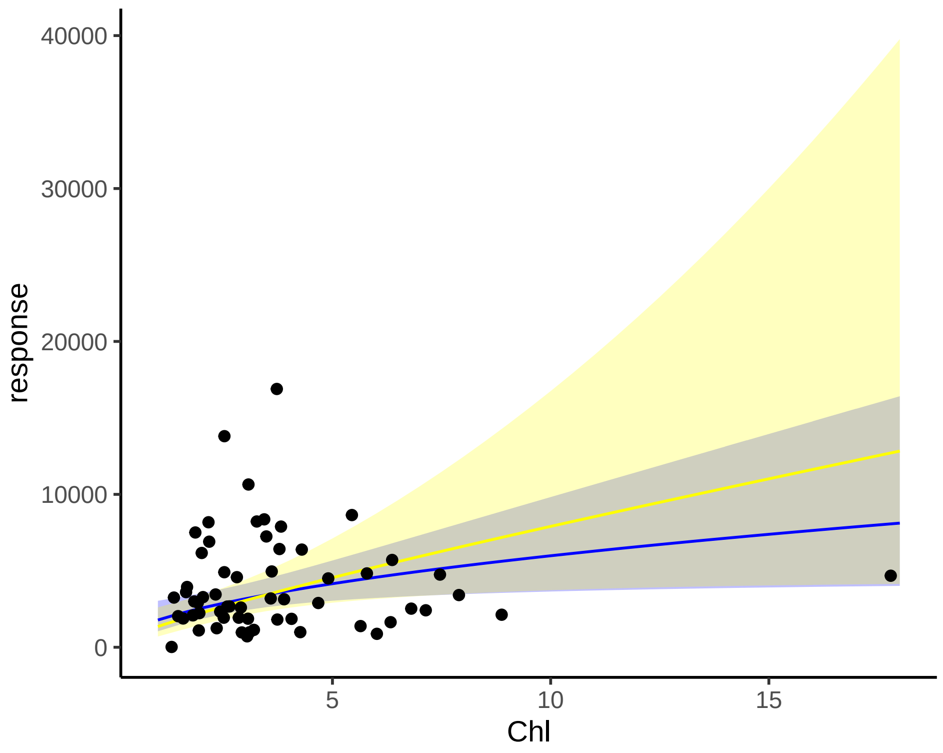
Again, the log transformed model has narrower error bands. That's not too surprising, since we are looking at the same models as we just examined. We are just looking at a different marginal view. (Remember, the model was fit on transformed Turbidity data, and `emmeans` handles that correctly, so we get the slightly curved prediction lines here.)

Chlorophyll

```
emm_chl_l <- as.data.frame(emmeans(combined_density_gam_l, "Chl",
                    at = list(Chl = seq(1, 18, 0.5)),
                    cov.reduce = median,
                    type = 'response'))

emm_chl_g <- as.data.frame(emmeans(combined_density_gam_g, "Chl",
                    at = list(Chl = seq(1, 18, 0.5)),
                    cov.reduce = median,
                    type = 'response'))
```

```
ggplot() +
  geom_ribbon(data = emm_chl_l, mapping = aes(x = Chl,
                                              ymin = lower.CL,
                                              ymax = upper.CL),
              fill = 'blue', alpha = 0.25) +
  geom_ribbon(data = emm_chl_g, mapping = aes(x = Chl,
                                              ymin = lower.CL,
                                              ymax = upper.CL),
              fill = 'yellow', alpha = 0.25) +
  geom_line(data = emm_chl_l, mapping = aes(x = Chl, y = response),
            color = 'blue') +
  geom_line(data = emm_chl_g, mapping = aes(x = Chl, y = response),
            color = 'yellow') +
  geom_point(data = base_data, mapping = aes(x = Chl, y = combined_density))
```



That points to the impact of that one outlier. . . . We have very wide error bands out by that one sample, especially for the Gamma model. Again, the error bands for the log transformed model are preferable.

Drop the Low Salinity Observations

Let's see what happens if we drop our handful of low salinity observations. Here we drop all samples with salinity below 5, which amounts to three samples from Station 1 in May of 2013, 2014, and 2017.

The goal here is to evaluate my guess that those low salinity observations dominate model fitting.

```
low_sample <- which(base_data$Sal <= 5)
base_data[low_sample,]
#> # A tibble: 3 x 24
#>   Date                  Station  Year Yearf Month Season   DOY riv_km  Temp    Sal
#>   <dttm>                <fct>   <dbl> <fct> <fct> <fct>  <dbl>  <dbl> <dbl>  <dbl>
#> 1 2013-05-28 00:00:00 1        2013 2013  May   Spring   148   22.6 11.7  0.0200
#> 2 2014-05-02 00:00:00 1        2014 2014  May   Spring   122   22.6  7.33 0.03
#> 3 2017-05-12 00:00:00 1        2017 2017  May   Spring   132   22.9 10.5  0.12
#> # ... with 14 more variables: 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>
smaller_data <- base_data[-low_sample,]
```

Total Density

Log Transformed Model

```
drop_density_gam_l <- gam(log(combined_density) ~
                            Station +
                            s(Yearf, bs = 're') +
                            s(Temp, bs="ts") +
                            s(Sal, bs="ts") +
                            s(log(Turb), bs="ts") +
                            s(log(Chl), bs="ts") +
                            s(log1p(RH),bs="ts"),
                          data = smaller_data, family = 'gaussian')
summary(drop_density_gam_l)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log(combined_density) ~ Station + s(Yearf, bs = "re") + 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)   8.4793     0.3268  25.944   <2e-16 ***
#> Station2     -0.5881     0.2649  -2.220   0.0318 *
#> Station3     -0.2311     0.2463  -0.939   0.3533
#> Station4     -0.5959     0.2735  -2.179   0.0350 *
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>                   edf Ref.df      F  p-value
#> s(Yearf)     3.756e+00      4 13.684  < 2e-16 ***
#> s(Temp)      3.848e+00      9  1.601 0.210871
#> s(Sal)       5.069e-01      9  0.186 0.130707
#> s(log(Turb)) 9.969e-01      9  2.386 0.000291 ***
#> s(log(Chl))  8.005e-01      9  1.084 0.018524 *
#> s(log1p(RH)) 1.503e-08      9  0.000 0.700934
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#>
#> R-sq.(adj) =  0.627   Deviance explained = 71.5%
#> GCV = 0.25418  Scale est. = 0.19105   n = 56
```

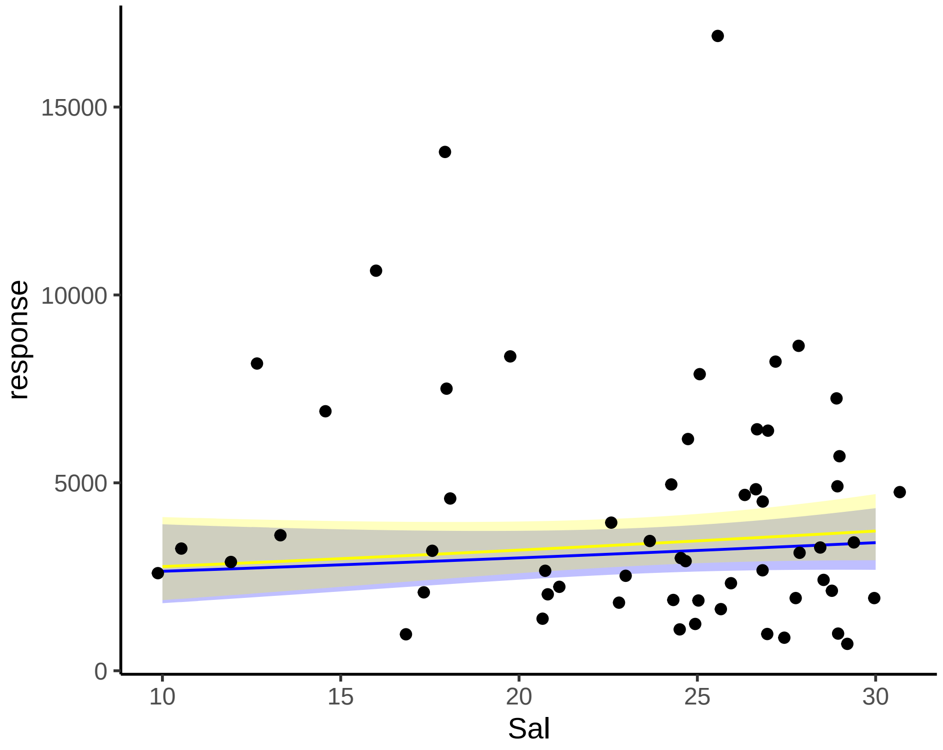
Gamma Model, Log Link

```
drop_density_gam_g <- gam(combined_density ~
                            Station +
                            s(Yearf, bs = 're') +
                            s(Temp, bs="ts") +
                            s(Sal, bs="ts") +
                            s(log(Turb), bs="ts") +
                            s(log(Chl), bs="ts") +
                            s(log1p(RH),bs="ts"),
                          data = smaller_data, family = Gamma(link = 'log'))
summary(drop_density_gam_g)
#>
#> Family: Gamma
#> Link function: log
#>
#> Formula:
#> combined_density ~ Station + s(Yearf, bs = "re") + 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)   8.5914     0.3486  24.647   <2e-16 ***
#> Station2     -0.6712     0.2554  -2.629    0.012 *
#> Station3     -0.2438     0.2374  -1.027    0.310
#> Station4     -0.6691     0.2636  -2.539    0.015 *
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>                    edf Ref.df      F  p-value
#> s(Yearf)     3.819e+00      4 14.843  < 2e-16 ***
#> s(Temp)      4.167e+00      9  2.615 0.121330
#> s(Sal)       5.590e-01      9  0.268 0.096394 .
#> s(log(Turb)) 1.037e+00      9  2.762 0.000155 ***
#> s(log(Chl))  8.213e-01      9  1.347 0.009665 **
#> s(log1p(RH)) 1.785e-05      9  0.000 0.768802
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.537   Deviance explained = 72.5%
#> GCV = 0.24583  Scale est. = 0.17221   n = 56
```

```
emm_sal_l <- as.data.frame(emmeans(drop_density_gam_l, "Sal",
                       at = list(Sal = seq(10, 30, 0.5)),
                       cov.reduce = median,
                       type = 'response'))

emm_sal_g <- as.data.frame(emmeans(drop_density_gam_g, "Sal",
                       at = list(Sal = seq(10, 30, 0.5)),
                       cov.reduce = median,
                       type = 'response'))
```

```
ggplot() +
  geom_ribbon(data = emm_sal_l, mapping = aes(x = Sal,
                                              ymin = lower.CL,
                                              ymax = upper.CL),
              fill = 'blue', alpha = 0.25) +
  geom_ribbon(data = emm_sal_g, mapping = aes(x = Sal,
                                              ymin = lower.CL,
                                              ymax = upper.CL),
              fill = 'yellow', alpha = 0.25) +
  geom_line(data = emm_sal_l, mapping = aes(x = Sal, y = response),
            color = 'blue') +
  geom_line(data = emm_sal_g, mapping = aes(x = Sal, y = response),
            color = 'yellow') +
  geom_point(data = smaller_data, mapping = aes(x = Sal, y = combined_density))
```



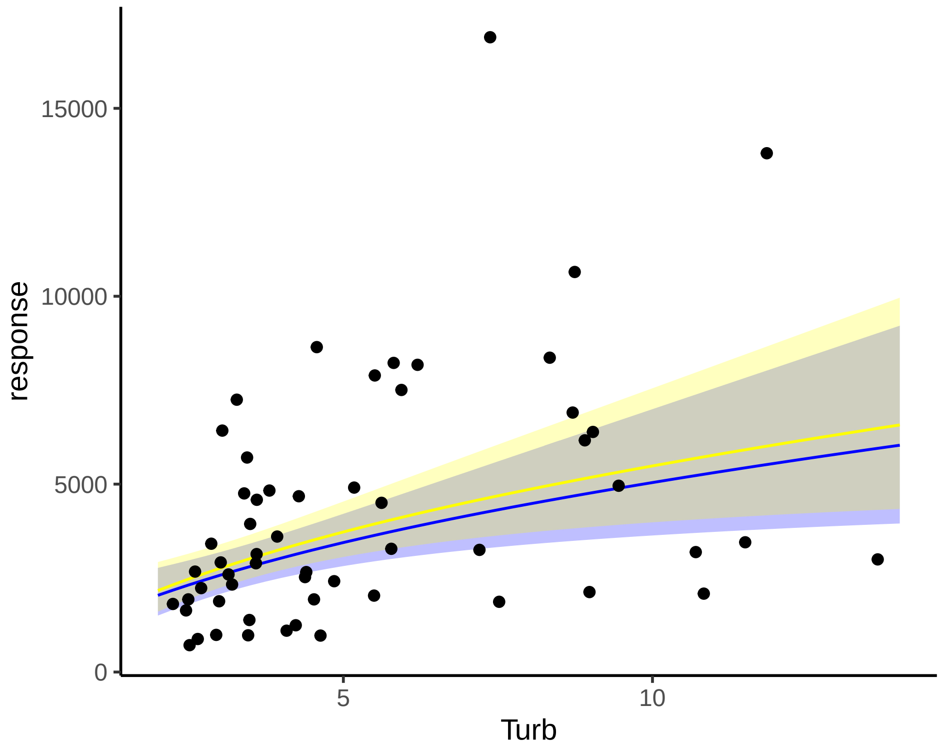
Here, the error bands are nearly identical. Apparently, the gamma model was much more strongly influenced by those extreme values.

Turbidity

```
emm_turb_l <- as.data.frame(emmeans(drop_density_gam_l, "Turb",
                    at = list(Turb = seq(2, 14, 0.25)),
                    cov.reduce = median,
                    type = 'response'))

emm_turb_g <- as.data.frame(emmeans(drop_density_gam_g, "Turb",
                    at = list(Turb = seq(2, 14, 0.25)),
                    cov.reduce = median,
                    type = 'response'))
```

```
ggplot() +
  geom_ribbon(data = emm_turb_l, mapping = aes(x = Turb,
                                               ymin = lower.CL,
                                               ymax = upper.CL),
              fill = 'blue', alpha = 0.25) +
  geom_ribbon(data = emm_turb_g, mapping = aes(x = Turb,
                                               ymin = lower.CL,
                                               ymax = upper.CL),
              fill = 'yellow', alpha = 0.25) +
  geom_line(data = emm_turb_l, mapping = aes(x = Turb, y = response),
            color = 'blue') +
  geom_line(data = emm_turb_g, mapping = aes(x = Turb, y = response),
            color = 'yellow') +
  geom_point(data = smaller_data, mapping = aes(x = Turb, y = combined_density))
```



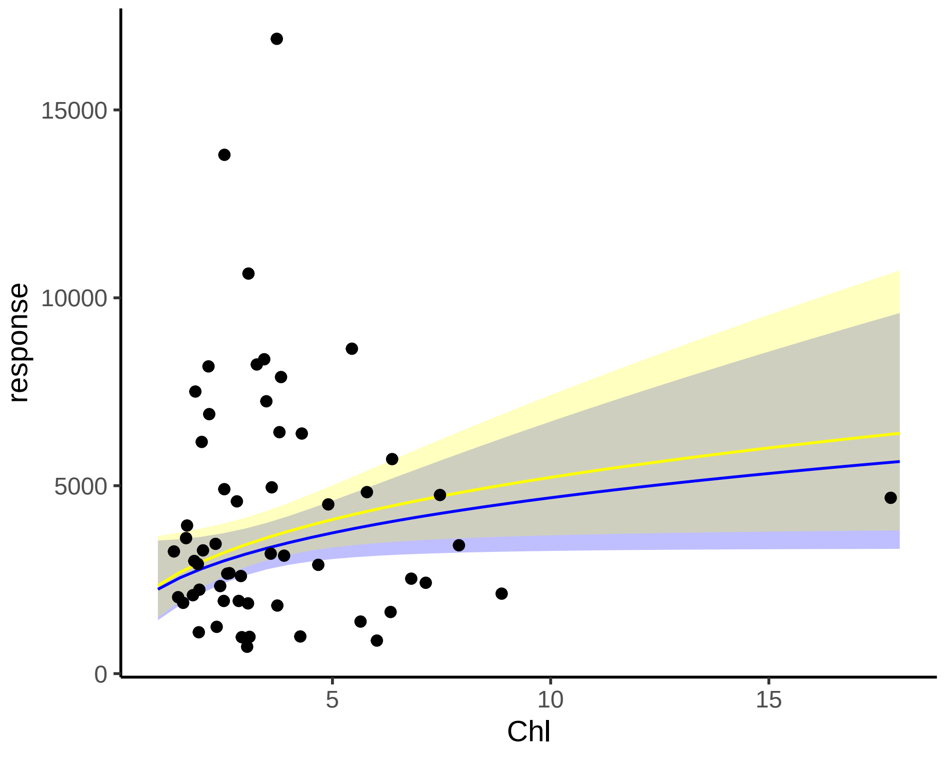
Again, the error bands are similar.

Chlorophyll

```r
emm_chl_l <- as.data.frame(emmeans(drop_density_gam_l, "Chl",
                    at = list(Chl = seq(1, 18, 0.5)),
                    cov.reduce = median,
                    type = 'response'))

emm_chl_g <- as.data.frame(emmeans(drop_density_gam_g, "Chl",
                    at = list(Chl = seq(1, 18, 0.5)),
                    cov.reduce = median,
                    type = 'response'))
```

```r
ggplot() +
  geom_ribbon(data = emm_chl_l, mapping = aes(x = Chl,
                                              ymin = lower.CL,
                                              ymax = upper.CL),
              fill = 'blue', alpha = 0.25) +
  geom_ribbon(data = emm_chl_g, mapping = aes(x = Chl,
                                              ymin = lower.CL,
                                              ymax = upper.CL),
              fill = 'yellow', alpha = 0.25) +
  geom_line(data = emm_chl_l, mapping = aes(x = Chl, y = response),
            color = 'blue') +
  geom_line(data = emm_chl_g, mapping = aes(x = Chl, y = response),
            color = 'yellow') +
  geom_point(data = smaller_data, mapping = aes(x = Chl, y = combined_density))
```



Note that the extreme chlorophyll sample remains in the data set, but standard error of prediction from the Gamma model does not balloon to the same extent. Apparently, those low salinity samples had a big effect on the Gamma model, but less of an effect on the log-linear model.