Effect of C0linearity on Mixed Effects Linear Models of Plankton Community Data

Curtis C. Bohlen, Casco Bay Estuary Partnership

7/21/2022

Contents

Introduction

This notebook looks closely at the effect of colinearity on LMER models for plankton community composition in Penobscot Bay.

Colinearities

Correlations among predictors are fairly high, so different predictors 'confound' each other. Values (and sometimes even the sign) of model parameters are dependent on which other terms are retained in each model.

The problem with colinearity is that the interpretation of relationships between predictors and response variables can depend on which other terms are retained in the model.

Estuaries are full of expected correlations among environmental variables, driven by seasonality, river discharge, estuary mixing dynamics etc. In Maine, during the warmer months of the year, river water is significantly warmer than ocean water, so location in the estuary (upstream vs. downstream), seasonality, temperature and salinity are all interrelated. Estuarine processes can further generate correlations among other environmental variables.

This complicates fitting of regression models in estuarine environments, as environmental variables are not independent, leading to difficulties with model specification.

A Note on Degrees of Freedom

We have just under 60 complete cases available, and as our models grow increasingly complex, we burn up degrees of freedom. The "full" linear mixed effects model used here has the following degrees of freedom:

Source	Degrees of Freedom
Intercept	1
Year (Random)	1 *
Sample Day (Random)	1 *
Station	3
Season	2
is_sp_up	1
Temp	1
Sal	1
log(Turb)	1
log(Chl)	1
log1p(Fish)	1
Total	*14*

Adding an interaction term between Station and Season (instead of fitting the "is_sp_up" term) adds $3 \times 2 - 1 - 1 = 4$ further degrees of freedom.

The "GAM" models we tested use multiple degrees of freedom to estimate each non-linear fit, so we often need to restrict the dimension of the smoothers to avoid using up all available degrees of freedom. The GAM models are complex models to fit to a fairly small data set.

Load Libraries

```
library(mgcv)
#> Loading required package: nlme
#> This is mgcv 1.8-40. For overview type 'help("mgcv-package")'.
library(lmerTest)  # Automatically loads lme4
#> Loading required package: lme4
#> Loading required package: Matrix
#>
#> Attaching package: 'lme4'
#> The following object is masked from 'package:nlme':
#>
#>     lmList
#>
#> Attaching package: 'lmerTest'
#> The following object is masked from 'package:lme4':
#>
#>     lmer
#> The following object is masked from 'package:stats':
#>
#>     step
library(tidyverse)
#> -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
#> v ggplot2 3.3.6     v purrr   0.3.4
#> v tibble  3.1.7     v dplyr   1.0.9
#> v tidyr   1.2.0     v stringr 1.4.0
#> v readr   2.1.2     v forcats 0.5.1
#> -- Conflicts ------------------------------------------ tidyverse_conflicts() --
#> x dplyr::collapse() masks nlme::collapse()
```

```
#> x tidyr::expand()   masks Matrix::expand()
#> x dplyr::filter()   masks stats::filter()
#> x dplyr::lag()      masks stats::lag()
#> x tidyr::pack()     masks Matrix::pack()
#> x tidyr::unpack()   masks Matrix::unpack()
library(readxl)
library(car)       # for vif() function
#> Loading required package: carData
#>
#> Attaching package: 'car'
#> The following object is masked from 'package:dplyr':
#>
#>     recode
#> The following object is masked from 'package:purrr':
#>
#>     some
library(emmeans)   # For extracting useful "marginal" model summaries
```

Set Graphics Theme

This sets `ggplot()`graphics for no background, no grid lines, etc. in a clean format suitable for (some) publications.

```
theme_set(theme_classic())
```

Input Data

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`
```

```
names(station_data)[10:12]
#> [1] "discharge_week_cftpersec" "discharg_day"
#> [3] "discharge_week_max"

names(station_data)[10:12] <- c('disch_wk', 'disch_day', 'disch_max')
```

Station names are arbitrary, and Erin 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>, disch_wk <dbl>, disch_day <dbl>,
#> #   disch_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>, max_temp <dbl>, max_sal <dbl>, ...
```

Scaling values

I divide two predictor variables by 1000 to reduce numerical problems with model fitting.

```
range(station_data$disch_wk)
#> [1]  4092 48125
range(station_data$`___61`, na.rm = TRUE)
#> [1]    0.00 3041.59
```

```
station_data <- station_data %>%
  mutate(Fish = `___61` / 1000,
         disch_wk = disch_wk/1000)  %>%  # to reduce scale issues in model fit
  select(-`___61`)
```

```
range(station_data$disch_wk)
#> [1]  4.092 48.125
range(station_data$Fish, na.rm = TRUE)
#> [1] 0.00000 3.04159
```

Subsetting to Desired Data Columns

I base selection of predictor variables here on the ones used in the manuscript, plus River Discharge. The three Discharge estimators are highly correlated. I selected the weekly discharge figure because I felt it probably reflected recent conditions better than a daily discharge value, but honestly, it makes little difference.

```
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')),
         is_sp_up = season == 'Spring' & Station == 1,
         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, is_sp_up, DOY, riv_km,
         disch_wk, disch_day, disch_max,
         Temp, Sal, Turb, AvgTurb, DOsat, Chl,
         Fish, RH,
         combined_density, H, SEI,
         Acartia, Balanus, Eurytemora, Polychaete, Pseudocal, Temora) %>%
  arrange(Date, Station)
head(base_data)
#> # A tibble: 6 x 29
#>   Date                Station  Year Yearf Month Season is_sp_up   DOY riv_km
#>   <dttm>              <fct>   <dbl> <fct> <fct> <fct>  <lgl>    <dbl>  <dbl>
#> 1 2013-05-28 00:00:00 1        2013 2013  May   Spring TRUE       148  22.6
#> 2 2013-05-28 00:00:00 2        2013 2013  May   Spring FALSE      148  13.9
#> 3 2013-05-28 00:00:00 3        2013 2013  May   Spring FALSE      148   8.12
#> 4 2013-05-28 00:00:00 4        2013 2013  May   Spring FALSE      148   2.78
```

```
#> 5 2013-07-25 00:00:00 1        2013 2013  Jul   Summer FALSE      206  22.6
#> 6 2013-07-25 00:00:00 2        2013 2013  Jul   Summer FALSE      206  13.9
#> # ... with 20 more variables: disch_wk <dbl>, disch_day <dbl>, disch_max <dbl>,
#> #   Temp <dbl>, Sal <dbl>, Turb <dbl>, AvgTurb <dbl>, DOsat <dbl>, Chl <dbl>,
#> #   Fish <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)
```

Complete Cases

This drops two samples, one with missing Zooplankton data (leading to our normal sample size of 59 samples), and one for missing fish data. We need this reduced data set to compare sequential models where we want to evaluate the importance of either fish or zooplankton as predictors (at least if we want to use model comparisons via AIC or likelihood).

```
complete_data <- base_data %>%
  select(Season, Station, Yearf, disch_wk,
         is_sp_up, Temp, Sal, Turb, Chl, DOsat, Fish, RH,
         combined_density, H,
         Acartia, Balanus, Eurytemora, Polychaete, Pseudocal, Temora) %>%
  filter(complete.cases(.))
```

Drop Low Salinity Observations

The low salinity spring samples are doing something rather different, and they complicate model fitting. Models are far better behaved if we exclude a few extreme samples. These are low salinity low zooplankton samples. We have two complementary ways to specify which samples to omit, without just omitting "outliers". The first is to restrict modeling to "marine" samples over a certain salinity, and the other is to omit spring upstream samples, which include most of the problematic samples.

```
drop_low <- complete_data %>%
  filter(Sal > 10)    # Pulls three samples, including one fall upstream sample
drop_v_low <- complete_data %>%
  filter(Sal > 5)     # drops two Samples, both spring Samples
drop_sp_up <- complete_data %>%
  filter(! is_sp_up)  # drops four samples
```

Correlations Among Predictors

Pairwise comparisons are a good place to start. To make it marginally easier to look at Season and Station via `cor()`, I convert them to numeric values.
This means the correlations are only linear associations, and don't reflect any non-linearities (which we actually know exist). But it's a good first step.

The following uses Spearman Rank correlations, to minimize the potential effect of different monotonic transformations of predictors on the reported correlations.

The Complete Data

```
complete_data %>%
  select(Season, Station, c(disch_wk:Fish),  -is_sp_up) %>%
  mutate(Season = as.numeric(Season),
         Station = as.numeric(Station)) %>%
  cor(method = 'spearman', use = 'pairwise') %>%
  round(3)
#>          Season Station disch_wk   Temp    Sal   Turb    Chl  DOsat   Fish
#> Season    1.000   0.000   -0.811  0.647  0.455 -0.083  0.473 -0.735 -0.213
#> Station   0.000   1.000    0.007 -0.242  0.380 -0.528  0.314  0.023 -0.201
#> disch_wk -0.811   0.007    1.000 -0.701 -0.269 -0.055 -0.431  0.547 -0.027
#> Temp      0.647  -0.242   -0.701  1.000 -0.038 -0.034  0.211 -0.437 -0.013
#> Sal       0.455   0.380   -0.269 -0.038  1.000 -0.363  0.508 -0.378 -0.095
#> Turb     -0.083  -0.528   -0.055 -0.034 -0.363  1.000 -0.219  0.013  0.080
#> Chl       0.473   0.314   -0.431  0.211  0.508 -0.219  1.000 -0.413 -0.081
#> DOsat    -0.735   0.023    0.547 -0.437 -0.378  0.013 -0.413  1.000  0.342
#> Fish     -0.213  -0.201   -0.027 -0.013 -0.095  0.080 -0.081  0.342  1.000
```

- Season is associated with Discharge, Temperature, Salinity, Chlorophyll and dissolved oxygen saturation.
- Station is also moderately correlated with Salinity, Turbidity and chlorophyll
- Discharge is highly correlated (in absolute value) with multiple possible predictor variables, especially Season, temperature, and Chlorophyll and dissolved oxygen.
- Dissolved oxygen is correlated with many predictors at intermediate values.

Reduced Data

Drop Low Salinity Samples

While dropping extreme samples (drop three low salinity samples)

```
drop_low %>%
  select(Season, Station, c(disch_wk:Fish),  -is_sp_up) %>%
  mutate(Season = as.numeric(Season),
         Station = as.numeric(Station)) %>%
  cor(use = 'pairwise') %>%
  round(3)
#>          Season Station disch_wk   Temp    Sal   Turb    Chl  DOsat   Fish
#> Season    1.000  -0.041   -0.623  0.672  0.392 -0.051  0.231 -0.695 -0.269
#> Station  -0.041   1.000    0.155 -0.302  0.423 -0.443  0.270  0.052 -0.257
#> disch_wk -0.623   0.155    1.000 -0.828 -0.254  0.119 -0.238  0.384 -0.050
#> Temp      0.672  -0.302   -0.828  1.000 -0.005 -0.167  0.155 -0.396  0.057
#> Sal       0.392   0.423   -0.254 -0.005  1.000 -0.276  0.353 -0.257 -0.412
#> Turb     -0.051  -0.443    0.119 -0.167 -0.276  1.000 -0.163 -0.047 -0.007
#> Chl       0.231   0.270   -0.238  0.155  0.353 -0.163  1.000 -0.044 -0.099
#> DOsat    -0.695   0.052    0.384 -0.396 -0.257 -0.047 -0.044  1.000  0.308
#> Fish     -0.269  -0.257   -0.050  0.057 -0.412 -0.007 -0.099  0.308  1.000
```

- Discharge is still highly correlated with Season and Temperature. The correlation with salinity is somewhat reduced.
- Season is still highly correlated with temperature, but the connection with salinity is slightly reduced.
- Station is still moderately correlated with Temperature, Salinity, and turbidity.

Drop all upstream spring samples

```
drop_sp_up%>%
  select(Season, Station, c(disch_wk:Fish), -is_sp_up) %>%
  mutate(Season = as.numeric(Season),
         Station = as.numeric(Station)) %>%
  cor(use = 'pairwise') %>%
  round(3)
#>          Season Station disch_wk   Temp    Sal   Turb    Chl  DOsat   Fish
#> Season    1.000  -0.141   -0.655  0.707  0.298 -0.045  0.188 -0.683 -0.106
#> Station  -0.141   1.000    0.159 -0.312  0.327 -0.460  0.227  0.115 -0.060
#> disch_wk -0.655   0.159    1.000 -0.828 -0.304  0.120 -0.245  0.394 -0.072
#> Temp      0.707  -0.312   -0.828  1.000  0.002 -0.165  0.161 -0.409  0.102
#> Sal       0.298   0.327   -0.304  0.002  1.000 -0.300  0.303 -0.197 -0.019
#> Turb     -0.045  -0.460    0.120 -0.165 -0.300  1.000 -0.161 -0.046 -0.131
#> Chl       0.188   0.227   -0.245  0.161  0.303 -0.161  1.000 -0.010  0.081
#> DOsat    -0.683   0.115    0.394 -0.409 -0.197 -0.046 -0.010  1.000  0.354
#> Fish     -0.106  -0.060   -0.072  0.102 -0.019 -0.131  0.081  0.354  1.000
```

That is more or less the same as our prior data subsets.

PCA of Predictors

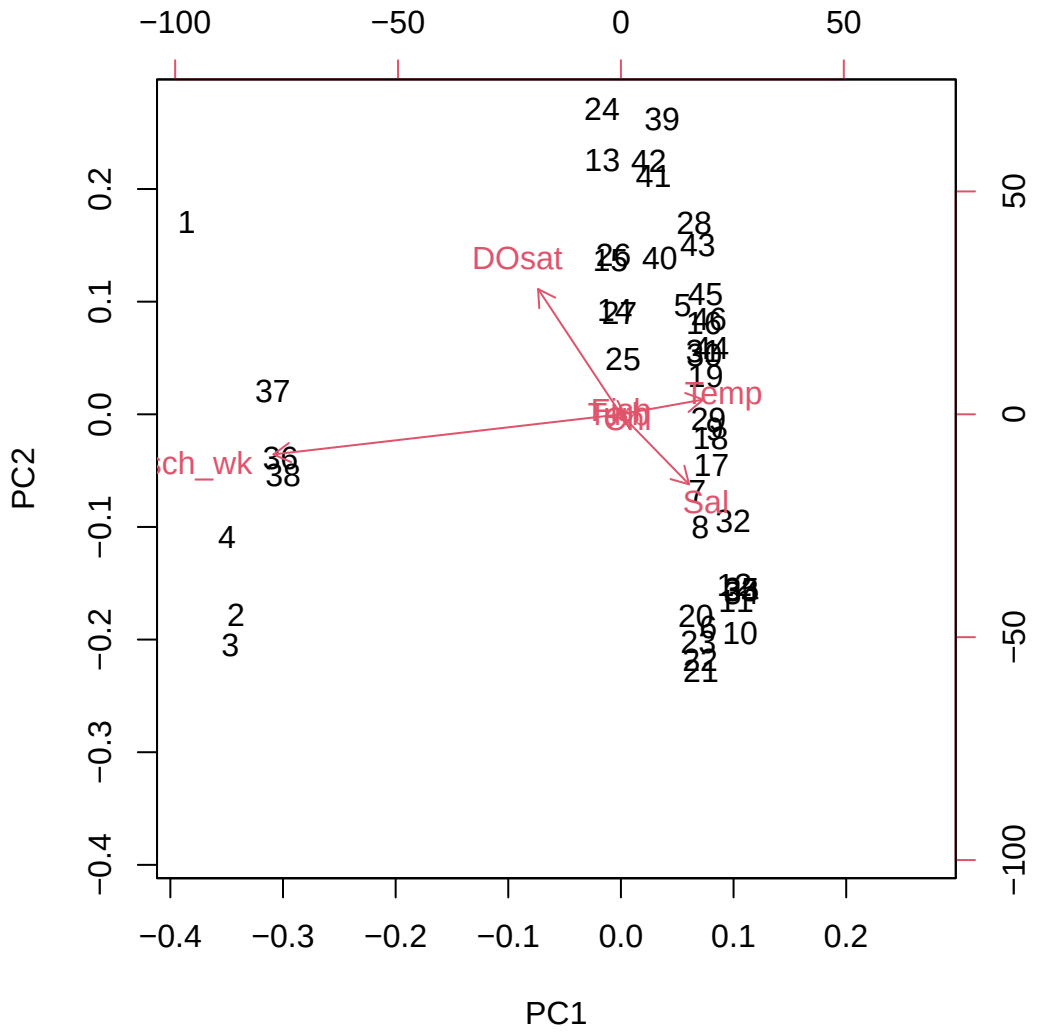
One way to look at correlations among variables is to look at a principal components analysis of explanatory variables. Variables that have a high loading on important PCA axes are highly colinear with the other variables included in the PCA.

The problem in using a PCA this way is that results depend somewhat on how each predictor will be transformed in the eventual models. We transform the predictors the way we have been including them in models, based on prior work.

The PCA based on the covariance matrix (equivalent to not scaling variables before conducting the PCA) is dominated by discharge, which alone accounts for high proportion of overall pattern.

High discharge samples are associated with high discharge spring events in 2013 and 2017. Quickly perusing the data shows that they are cooler temperature, slightly low salinity, high dissolved oxygen samples. That impression is confirmed by the PCA.

```
test <- complete_data %>%
  mutate(Turb = log(Turb),
         Chl = log(Chl),
         Fish = log1p(Fish)) %>%
  select(c(disch_wk:Fish ), - is_sp_up)  %>%
  prcomp(scale. = FALSE)
biplot(test)
```




```
summary(test)
#> Importance of components:
#>                           PC1    PC2     PC3     PC4     PC5     PC6     PC7
#> Standard deviation     15.4161 6.1942 5.39665 1.89837 0.48633 0.43932 0.24107
#> Proportion of Variance  0.7685 0.1241 0.09418 0.01165 0.00076 0.00062 0.00019
#> Cumulative Proportion   0.7685 0.8926 0.98677 0.99842 0.99919 0.99981 1.00000
```

```
test$rotation
#>                    PC1           PC2           PC3          PC4           PC5
#> disch_wk -0.9318569029 -0.267982438  0.007703295 -0.24368438 -9.073760e-03
#> Temp      0.2206687920  0.098439832  0.231079260 -0.93852224 -5.931587e-02
#> Sal       0.1827275313 -0.467470333 -0.837429580 -0.20653294 -5.187889e-02
#> Turb     -0.0046490113  0.005409442  0.024903099  0.07693657 -9.504576e-02
#> Chl       0.0144095463 -0.022867311 -0.026107816 -0.06151475  9.923035e-01
#> DOsat    -0.2221013525  0.836028337 -0.493912993 -0.08554754  2.823533e-03
#> Fish      0.0006219682  0.021701224  0.001704530  0.01078840  2.295371e-05
```

```
#>                  PC6          PC7
#> disch_wk  0.01443000 -0.009805692
#> Temp      0.06130794 -0.010780764
#> Sal       0.03469432 -0.015584106
#> Turb      0.99076822 -0.052481934
#> Chl       0.10046913 -0.006596693
#> DOsat     0.01454192  0.015483610
#> Fish     -0.05387232 -0.998252046
```

Axis 1, which accounts for just over 3/4 of the variance, is nearly synonymous with discharge.

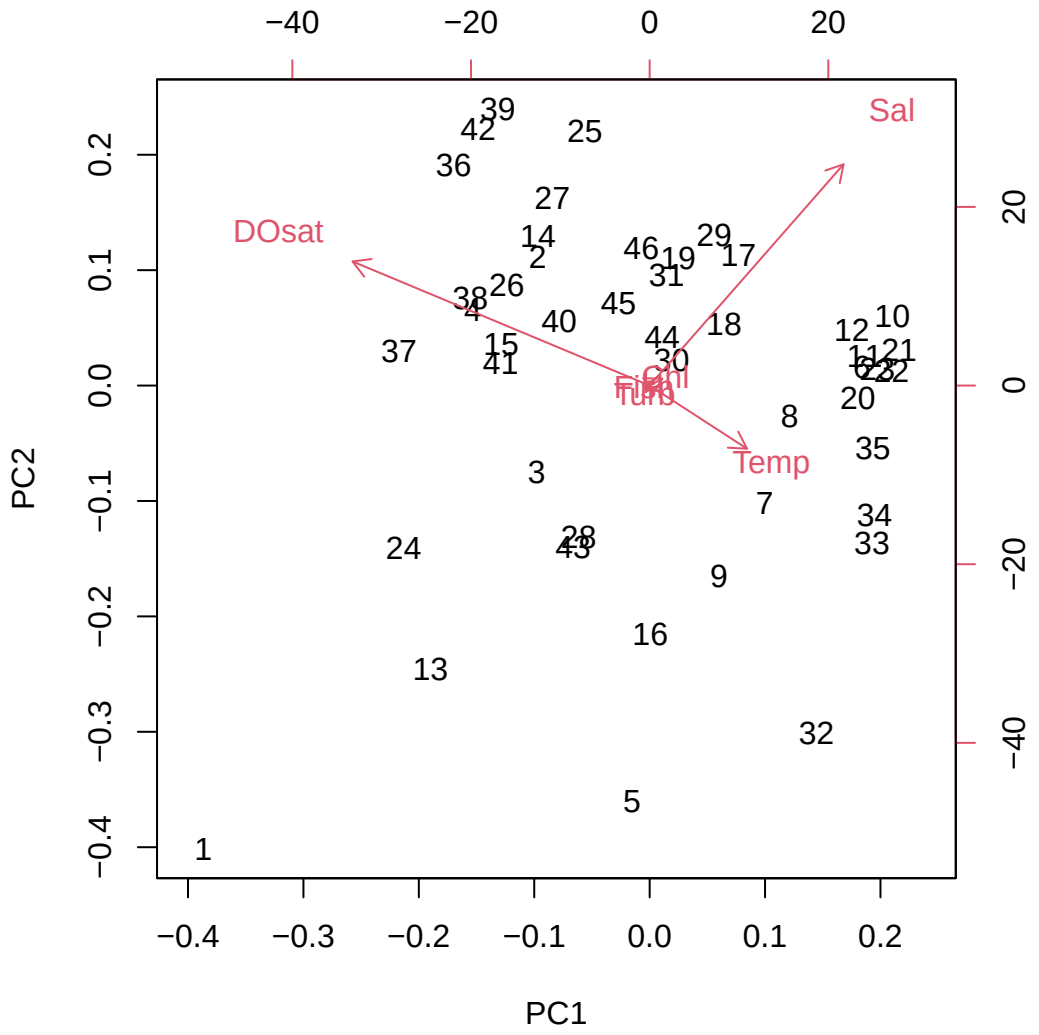
Axis 2 is "mostly" dissolved oxygen, but it explains much less of the structure in the data, and we are missing dissolved oxygen data from 2013.

Axis 3 is associated with salinity, and explains nearly as much of the structure among predictors as dissolved oxygen.

Cumulatively, we've got about 89% of the variance (information) in the correlation matrix explained by the first two PCA axes, and 99% explained by the first three. But most of the structure is in Axis 1, suggesting any model that includes Discharge is going to be problematic.

In the absence of Discharge

```
test <- complete_data %>%
  mutate(Turb = log(Turb),
         Chl = log(Chl),
         Fish = log1p(Fish)) %>%
  select(c(Temp:Fish), -is_sp_up)  %>%
  prcomp(scale. = FALSE)
biplot(test)
```




```
summary(test)
#> Importance of components:
#>                           PC1    PC2    PC3     PC4     PC5     PC6
#> Standard deviation     7.5949 5.3972 3.5469 0.49092 0.44882 0.24912
#> Proportion of Variance 0.5774 0.2916 0.1259 0.00241 0.00202 0.00062
#> Cumulative Proportion  0.5774 0.8690 0.9950 0.99736 0.99938 1.00000
```

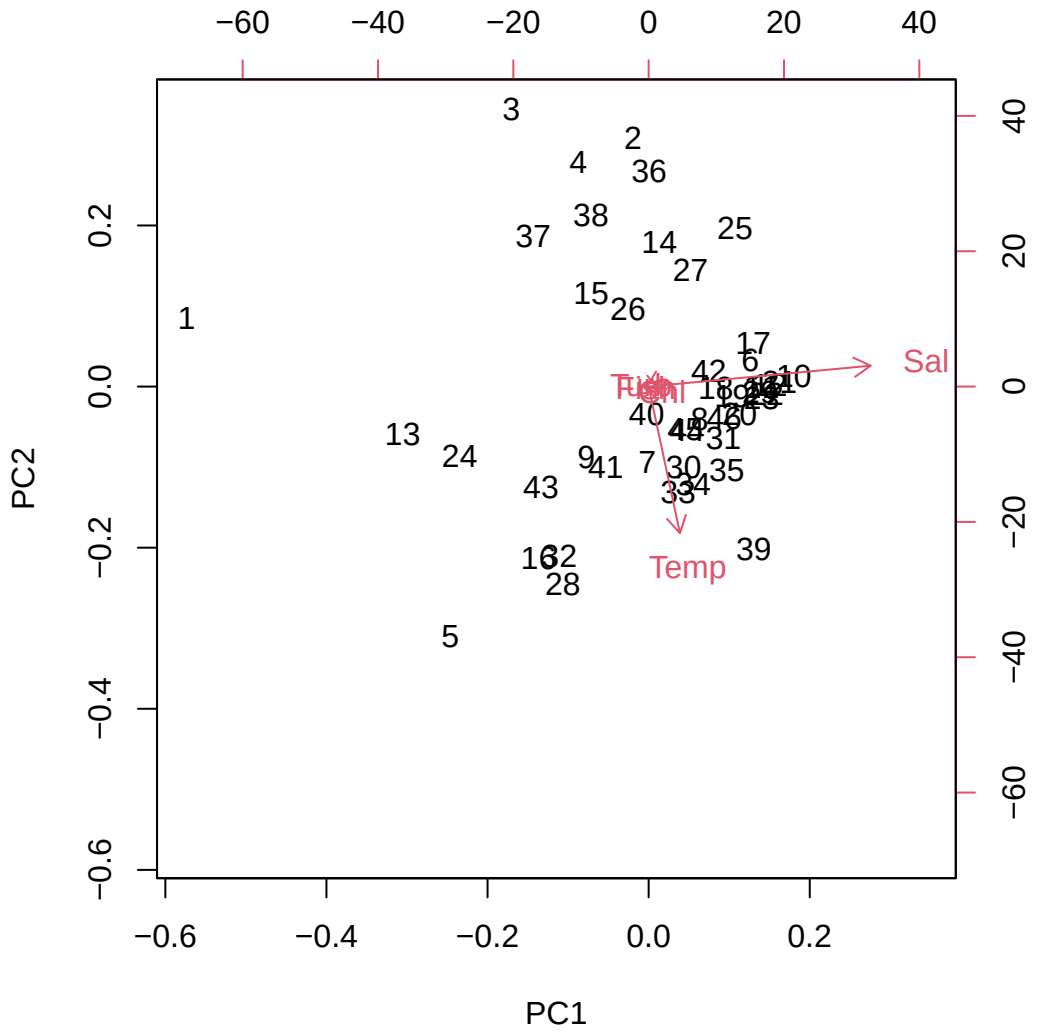
So Axis 1 now explains only 58% of the structure in the predictor variables. We are likely to have fewer problems with colinearity if we omit Discharge from our models.

```
test$rotation
#>                PC1          PC2         PC3          PC4         PC5
#> Temp   0.26442065 -0.241321635 -0.93276458 -0.035385320  0.01413726
#> Sal    0.52670467  0.845784188 -0.06753927 -0.048307403  0.01722141
#> Turb  -0.01074947 -0.024849953  0.02713954 -0.243631653  0.96907117
#> Chl    0.03433072  0.026317224 -0.03049870  0.967835856  0.24504192
```

```
#> DOsat -0.80698186  0.474444105 -0.35103900  0.001578658  0.01358892
#> Fish  -0.01224912 -0.002377262 -0.02221681 -0.018685689 -0.01320675
#>                PC6
#> Temp  -0.018543187
#> Sal    0.006290286
#> Turb   0.008663149
#> Chl    0.021138789
#> DOsat -0.016356541
#> Fish   0.999413402
```

Drop DO Saturation

```
test <- complete_data %>%
  mutate(Turb = log(Turb),
         Chl = log(Chl),
         Fish = log1p(Fish)) %>%
  select(c(Temp:Fish), -is_sp_up, -DOsat)  %>%
  prcomp(scale. = FALSE)
biplot(test)
```




```
summary(test)
#> Importance of components:
#>                           PC1    PC2     PC3     PC4     PC5
#> Standard deviation     6.117 4.0359 0.49103 0.45629 0.26623
#> Proportion of Variance 0.690 0.3004 0.00445 0.00384 0.00131
#> Cumulative Proportion  0.690 0.9904 0.99485 0.99869 1.00000
```

Axis 1 explains 69%% of the structure in the predictor variables, but pretty much all of the structure in the predictors is explained by axis 1, with high loading on Salinity, and axis 2, with high loading on Temperature.

```
test$rotation
#>              PC1          PC2         PC3         PC4          PC5
#> Temp  0.13926394 -0.989538565  0.03622145 -0.00665461  0.007918792
#> Sal   0.98866897  0.140918109  0.04850516 -0.01382377 -0.011485422
#> Turb -0.02662528  0.010386463  0.22073531 -0.97299320 -0.061183131
#> Chl   0.04814885 -0.027686689 -0.97312396 -0.21985655 -0.040099854
```

```
#> Fish -0.01058361 -0.009004703  0.02531688  0.06864367 -0.997223147
```

Conclusions

- The large loading of Discharge on the first PCA axis tells us that it is close to colinear with some linear combination of the other predictors. Any model including Discharge is going to have problems with colinearity.
- The moderate loading on dissolved oxygen on Axis 2 is less problematic, but we lack dissolved oxygen data for 2013, so we chose to omit DO from our models as well.
- This analysis does not address strong statistical associations between our experimental factors, Season and Station, and the measured environmental variables.
  - Season tends to be associated fairly strongly with temperature, as water temperatures are lower across the board in spring (especially in the upper estuary).
  - Station tends to be associated with salinity, with lower salinity in the upper watershed, especially in spring.
- We are likely to have trouble interpreting model coefficients for models that include both Season and Temperature or both Station and Salinity.

LMER Models of Zooplankton Density

We focus on linear models instead of GAMS because we can use standard methods to evaluate the importance of colinearity. The only random factor we include in these models is Year. much of this modelling could be managed as linear models with only limited loss of information.

Full Model

We start with a "full" model and demonstrate the problem of colinearity that arises.

```
density_lmer <- lmer(log1p(combined_density) ~
                       Station + Season +
                        Temp +
                        Sal +
                        disch_wk +
                        log(Turb) +
                        log(Chl) +
                        log1p(Fish) +
                       (1 | Yearf),
                     data = base_data, na.action = na.omit, REML = FALSE)
vif(density_lmer)
#>                   GVIF Df GVIF^(1/(2*Df))
#> Station       5.027723  3        1.308866
#> Season       12.062232  2        1.863618
#> Temp         13.642149  1        3.693528
#> Sal           4.149246  1        2.036970
#> disch_wk      5.548167  1        2.355455
#> log(Turb)     1.540687  1        1.241244
```

```
#> log(Chl)     2.102863  1        1.450125
#> log1p(Fish)  1.496782  1        1.223431
```

We have many problems with colinearity, especially with Temperature, salinity, discharge, Season and possibly Station

Reduced Model

First I drop discharge.

```
density_lmer <- lmer(log1p(combined_density) ~
                     Station + Season +
                      Temp +
                      Sal +
                      #disch_wk +
                      log(Turb) +
                      log(Chl) +
                      log1p(Fish) +
                     (1 | Yearf),
                   data = base_data, na.action = na.omit, REML = FALSE)
vif(density_lmer)
#>                  GVIF Df GVIF^(1/(2*Df))
#> Station      4.813035  3        1.299381
#> Season      11.750245  2        1.851449
#> Temp         8.432985  1        2.903960
#> Sal          3.787705  1        1.946203
#> log(Turb)    1.523441  1        1.234278
#> log(Chl)     2.074541  1        1.440327
#> log1p(Fish)  1.450380  1        1.204317
```

We see Season, Temperature and perhaps Station continue to be problems.

Final Model

We drop Season from the model next.

```
density_lmer <- lmer(log1p(combined_density) ~
                     Station +
                      #Season +
                      Temp +
                      Sal +
                      #disch_wk +
                      log(Turb) +
                      log(Chl) +
                      log1p(Fish) +
                     (1 | Yearf),
                   data = base_data, na.action = na.omit, REML = FALSE)
vif(density_lmer)
#>                 GVIF Df GVIF^(1/(2*Df))
#> Station     4.650954  3        1.291984
#> Temp        2.387465  1        1.545142
```

```
#> Sal         2.690435  1        1.640254
#> log(Turb)   1.515444  1        1.231034
#> log(Chl)    1.767940  1        1.329639
#> log1p(Fish) 1.088004  1        1.043075
```

No remaining terms have especially high VIF (when appropriately scaled to degrees of freedom), so we can stop here.

Alternative Model Family

We could go the other direction, and instead of dropping Season, we could retain it and drop highly correlated predictor variables, especially temperature. Salinity is some what correlated with Station, so dropping it also assists with model colinearity problems.

```
density_lmer_alt <- lmer(log1p(combined_density) ~
                      Station +
                      Season +
                      # Temp +
                      #Sal +
                       #disch_wk +
                       log(Turb) +
                       log(Chl) +
                       log1p(Fish) +
                      (1 | Yearf),
                    data = base_data, na.action = na.omit, REML = FALSE)
vif(density_lmer_alt)
#>                 GVIF Df GVIF^(1/(2*Df))
#> Station     1.712565  3        1.093808
#> Season      2.215958  2        1.220086
#> log(Turb)   1.445432  1        1.202261
#> log(Chl)    2.051420  1        1.432278
#> log1p(Fish) 1.183599  1        1.087934
```

Conclusions

In effect, what this shows is that we can **either** talk in terms of season and Station, or in terms of salinity and temperature, but that if we try to do both at once, we are going to have problems with colinearity, making model interpretation messy.

LMER Models of Fish Abundance

Full model

```
fish_lmer <- lmer(log1p(Fish) ~
                      Station +
                      Season +
                       Temp +
                       Sal +
```

```
                      disch_wk +
                      log(Turb) +
                      log(Chl) +
                      log1p(combined_density) +
                     (1 | Yearf),
                   data = base_data, na.action = na.omit, REML = FALSE)
vif(fish_lmer)
#>                              GVIF Df GVIF^(1/(2*Df))
#> Station                  4.341306  3        1.277233
#> Season                  10.108408  2        1.783079
#> Temp                    15.006133  1        3.873775
#> Sal                      5.221024  1        2.284956
#> disch_wk                 5.776604  1        2.403457
#> log(Turb)                1.878087  1        1.370433
#> log(Chl)                 2.132617  1        1.460348
#> log1p(combined_density)  1.711654  1        1.308302
```

We have many problems with colinearity here, as expected.

Reduced Model

First lets drop discharge. Note that i this fit, `Yearf` has so little effect on the model that the `lmer()` algorithm judges it to have zero variance. In other words, the random structure of the model provides no benefit.

```
fish_lmer <- lmer(log1p(Fish) ~
                    Station + Season +
                     Temp +
                     Sal +
                     #disch_wk +
                     log(Turb) +
                     log(Chl) +
                     log1p(combined_density) +
                    (1 | Yearf),
                  data = base_data, na.action = na.omit, REML = FALSE)
#> boundary (singular) fit: see help('isSingular')
vif(fish_lmer)
#>                              GVIF Df GVIF^(1/(2*Df))
#> Station                  4.133868  3        1.266853
#> Season                   8.613329  2        1.713140
#> Temp                     6.595740  1        2.568217
#> Sal                      3.680945  1        1.918579
#> log(Turb)                1.796728  1        1.340421
#> log(Chl)                 2.066176  1        1.437420
#> log1p(combined_density) 1.541807  1        1.241695
```

We see Season, temperature continue to problems, just as for the zooplankton models.

Final Model

We drop Season next.

```
fish_lmer <- lmer(log1p(Fish) ~
                    Station +
                    #Season +
                     Temp +
                     Sal +
                     #disch_wk +
                     log(Turb) +
                     log(Chl) +
                     log1p(combined_density) +
                    (1 | Yearf),
                  data = base_data, na.action = na.omit, REML = FALSE)
#> boundary (singular) fit: see help('isSingular')
vif(fish_lmer)
#>                             GVIF Df GVIF^(1/(2*Df))
#> Station                 4.013986  3        1.260654
#> Temp                    2.020736  1        1.421526
#> Sal                     2.810159  1        1.676353
#> log(Turb)               1.760341  1        1.326779
#> log(Chl)                1.430518  1        1.196043
#> log1p(combined_density) 1.338286  1        1.156843
```

We again see that no remaining terms have especially high VIF, so we can stop here.

Alternative Model Family

```
fish_lmer <- lmer(log1p(Fish) ~
                    Station +
                    Season +
                     #Temp +
                     Sal +
                     #disch_wk +
                     log(Turb) +
                     log(Chl) +
                     log1p(combined_density) +
                    (1 | Yearf),
                  data = base_data, na.action = na.omit, REML = FALSE)
#> boundary (singular) fit: see help('isSingular')
vif(fish_lmer)
#>                             GVIF Df GVIF^(1/(2*Df))
#> Station                 3.459742  3        1.229817
#> Season                  2.638864  2        1.274542
#> Sal                     3.508161  1        1.873008
#> log(Turb)               1.711835  1        1.308371
#> log(Chl)                1.907803  1        1.381232
#> log1p(combined_density) 1.314407  1        1.146476
```

So, dropping Discharge and Temperature (NOT salinity this time) produces a model with limited problems due to colinearity. Things do get slightly better (from a VIF perspective) if you drop Salinity from the model as well.

Models on Reduced Data

Even if we fit models to the reduced data, we find colinearities are still fairly high, so we still need to drop both Discharge and Season to ensure decent VIF.

```
density_lmer_drop <- lmer(log1p(combined_density) ~
                      Station + Season +
                       Temp +
                       Sal +
                       disch_wk +
                       log(Turb) +
                       log(Chl) +
                       log1p(Fish) +
                      (1 | Yearf),
                    data = drop_low, na.action = na.omit, REML = FALSE)
vif(density_lmer_drop)
#>                   GVIF Df GVIF^(1/(2*Df))
#> Station       6.403442  3        1.362706
#> Season       17.218686  2        2.037042
#> Temp         17.386112  1        4.169666
#> Sal           4.329827  1        2.080824
#> disch_wk      4.969930  1        2.229334
#> log(Turb)     2.215011  1        1.488291
#> log(Chl)      2.477387  1        1.573972
#> log1p(Fish)   1.667574  1        1.291346
```

```
density_lmer_drop <- lmer(log1p(combined_density) ~
                      Station +
                       #Season +
                       Temp +
                       Sal +
                       #disch_wk +
                       log(Turb) +
                       log(Chl) +
                       log1p(Fish) +
                      (1 | Yearf),
                    data = drop_low, na.action = na.omit, REML = FALSE)
vif(density_lmer_drop)
#>                   GVIF Df GVIF^(1/(2*Df))
#> Station       5.215948  3        1.316908
#> Temp          2.324974  1        1.524787
#> Sal           2.720372  1        1.649355
#> log(Turb)     1.732306  1        1.316171
#> log(Chl)      1.888849  1        1.374354
#> log1p(Fish)   1.223124  1        1.105949
```

Conclusions

Our "Standard" models will exclude discharge, and take two forms:

1. **Season and Station Models** include Season and Station, but omit Temperature and Salinity.

2. **Environment Models** that omit Season, but include Temperature and Salinity. They will include Station where it does not pose high colinearity or concurvity problems, but omit Station if it does.