Using GAMs to Analyze Plankton Comunity NMDS Data

Curtis C. Bohlen, Casco Bay Estuary Partnership

5/17/2022

Contents

Introduction

This notebook takes the output of an NMDS analyses of plankton community composition and the relationship of the community composition to major environmental variables.

The flow of analyses is as follows:

1. Conduct the NMDS analysis (mimicking Ambrose's original NMDS plot)
2. Plot the results, color coded by different environmental variables to examine major relationships with predictors on a graphic basis.
3. Conduct a linear analysis of the relationship between GAM output and each predictor, using the `envfit()` function from `vegan`.
4. Conduct GAM analyses of the synthetic NMDS axis scores, based on GAM models that follow the same model form as we used to analyze zooplankton abundance.

Load Libraries

```
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::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
library(vegan)
#> Loading required package: permute
#> Loading required package: lattice
#> This is vegan 2.6-2
library(readxl)
library(mgcv)      # for GAM models
#> Loading required package: nlme
#>
#> Attaching package: 'nlme'
#> The following object is masked from 'package:dplyr':
#>
#>     collapse
#> This is mgcv 1.8-40. For overview type 'help("mgcv-package")'.
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())
```

Folder References

I use folder references to allow limited indirection, thus making code from GitHub repositories more likely to run "out of the box".

```
data_folder <- "Original_Data"


dir.create(file.path(getwd(), 'figures'), showWarnings = FALSE)
```

Input Data

Environmental Data

```
filename.in <- "penob.station.data EA 3.12.20.xlsx"
file_path <- file.path(data_folder, filename.in)

station_data_2 <- read_excel(file_path,
                           sheet="Final", col_types = c("skip", "date",
                                                  "numeric", "text", "skip",
                                                  "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)) %>%
  filter(! (station == 8 & month == 'May' & year == 2015))
#> New names:
#> * `` -> `...60`
```

Station names are arbitrary, and Ambrose 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_2 %>%
  mutate(station = factor(as.numeric(factor(station)))) %>%
  mutate(season = case_when(month == 'May' ~ 'Spring',
                            month == 'July' ~ 'Summer',
                            TRUE ~ 'Fall')) %>%
  relocate(season, .after = month) %>%
  relocate(station, .after = season)
```

Here I mostly select the depth-averaged water chemistry parameters, create short names that will work in later analyses and graphics and convert some variables to factors to control later analyses.

```
station_data <- station_data %>%
  rename(Date = date,
         Station = station,
         Year = year) %>%
  select(-c(month)) %>%
  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,
         Fish = `___60`
```

```
        ) %>%
  select(Date, Station, Year, Yearf, Month, Season, DOY, riv_km, Temp, Sal, Turb, AvgTurb,
         DOsat, Chl, RH, Fish) %>%
  arrange(Date, Station)
head(station_data)
#> # A tibble: 6 x 16
#>   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 7 more variables: Sal <dbl>, Turb <dbl>, AvgTurb <dbl>, DOsat <dbl>,
#> #   Chl <dbl>, RH <dbl>, Fish <dbl>
```

Composition Data

```
filename.in <- "Penobscot_Zooplankton and field data_EA_2.13.20.xlsx"
file_path <- file.path(data_folder, filename.in)
zoopl <- read_excel(file_path,
                    sheet = "NMDS Happy",
                    col_types = c("date",
                                  "text", "numeric", "numeric", "text",
                                  "text", "text", "text", "text", "text",
                                  "text", "numeric", "text", "text",
                                  "numeric", "numeric", "numeric",
                                  "text", "text", "text", "numeric",
                                  "numeric", "numeric", "numeric")) %>%
  select(-c(`...20`:`...24`)) %>%
  rename_with(~ gsub(" ", "_", .x))
#> New names:
#> * `` -> `...20`
#> * `` -> `...21`
#> * `` -> `...22`
#> * `` -> `...23`
#> * `` -> `...24`
```

We renumber the stations here as well. The code is similar.

```
zoopl <- zoopl %>%
  mutate(STATION = factor(as.numeric(factor(STATION))))
zoopl
#> # A tibble: 814 x 19
#>    DATE                Month  Year STATION PHYLUM CLASS `SUB-CLASS` ORDER FAMILY
#>    <dttm>              <chr> <dbl> <fct>   <chr>  <chr> <chr>       <chr> <chr>
#>  1 2015-09-16 00:00:00 Sept~  2015 4       Arthr~ Maxi~ Copepoda    <NA>  <NA>
#>  2 2014-05-02 00:00:00 May    2014 1       Arthr~ Maxi~ Copepoda    Cala~ <NA>
#>  3 2017-07-12 00:00:00 July   2017 3       Unkno~ <NA>  <NA>        <NA>  <NA>
#>  4 2016-07-20 00:00:00 July   2016 4       Unkno~ <NA>  <NA>        <NA>  <NA>
```

```
#>  5 2015-09-16 00:00:00 Sept~  2015 3       Unkno~ <NA>  <NA>        <NA>  <NA>
#>  6 2017-10-11 00:00:00 Octo~  2017 1       Unkno~ Unid~ <NA>        <NA>  <NA>
#>  7 2016-07-20 00:00:00 July   2016 3       Unkno~ <NA>  <NA>        <NA>  <NA>
#>  8 2016-05-25 00:00:00 May    2016 2       Unkno~ <NA>  <NA>        <NA>  <NA>
#>  9 2013-09-25 00:00:00 Sept~  2013 3       Unkno~ <NA>  <NA>        <NA>  <NA>
#> 10 2013-07-25 00:00:00 July   2013 4       Unkno~ <NA>  <NA>        <NA>  <NA>
#> # ... with 804 more rows, and 10 more variables: GENUS <chr>, SPECIES <chr>,
#> #   QUANTITY <dbl>, LOWEST_TAXA <chr>, NAME <chr>, `TOTAL_#_ORGANISMS` <dbl>,
#> #   CORRECTED_PERCENT_ABUNDANCE <dbl>, `NET_MESH_SIZE_(MICRONS)` <dbl>,
#> #   NOTES <chr>, Picture_number <chr>
```

Turn Data from Long to Wide

This code generates a total abundance for each taxa by site and date and pivots it to wide format. The code is more compact that what Erin used, but slightly more opaque because it relies on several options of the `pivot_wider()` function.

```
zoopl2 <- zoopl %>%
  pivot_wider(c(DATE, Month, Year, STATION),
              names_from = NAME,
              names_sort = TRUE,
              values_from = CORRECTED_PERCENT_ABUNDANCE,
  values_fn = sum,
  values_fill = 0)
zoopl2
#> # A tibble: 59 x 53
#>    DATE                Month  Year STATION Acartia Amphipod `Arrow worm` Balanus
#>    <dttm>              <chr> <dbl> <fct>     <dbl>    <dbl>        <dbl>   <dbl>
#>  1 2015-09-16 00:00:00 Sept~  2015 4         43.4   0            0          3.28
#>  2 2014-05-02 00:00:00 May    2014 1          6.85  0            0          3.43
#>  3 2017-07-12 00:00:00 July   2017 3         32.5   0            0.0219    22.6
#>  4 2016-07-20 00:00:00 July   2016 4         49.8   0            0.00463   0.422
#>  5 2015-09-16 00:00:00 Sept~  2015 3         49.1   0.00942      4.96       7.66
#>  6 2017-10-11 00:00:00 Octo~  2017 1         68.6   0            0          0
#>  7 2016-07-20 00:00:00 July   2016 3         49.3   0            0          0
#>  8 2016-05-25 00:00:00 May    2016 2          6.45  0.00466      0          1.38
#>  9 2013-09-25 00:00:00 Sept~  2013 3         45.6   0            0.00236    3.88
#> 10 2013-07-25 00:00:00 July   2013 4         48.2   0            0          9.16
#> # ... with 49 more rows, and 45 more variables: Bivalve <dbl>,
#> #   `Brittle Star` <dbl>, Bryozoan <dbl>, `Calanoid spp` <dbl>, Calanus <dbl>,
#> #   Caligus <dbl>, Centropages <dbl>, Cladoceran <dbl>, `Crab larvae` <dbl>,
#> #   Crangon <dbl>, Ctenophore <dbl>, Cumacean <dbl>, Decapod <dbl>,
#> #   Diacyclops <dbl>, Eucyclops <dbl>, Eurytemora <dbl>, `Fish larvae` <dbl>,
#> #   Gastropod <dbl>, `Halicyclops fosteri` <dbl>, Harpacticoid <dbl>,
#> #   Hermit <dbl>, Hydrozoan <dbl>, Isopod <dbl>, Leptodiaptomus <dbl>, ...
```

Check for Dropped Sample

Erin Ambrose dropped 5/20/15 Station 8 from both datasheets. Environmental and zooplankton data should each have 59 rows, and they do.

Erin notes that there was no zooplankton "sample" (?) only nekton for that sample. I'm not sure if that means no sample was collected or there were no zooplankton in the sample. Anyway, she noted that this sample "threw off calculation of percent abundances."

Note that that sample is one of the Spring "washout" samples that cause trouble on our other analyses as well.

```
sum(! is.na((zoopl2 %>%
  filter((STATION == 4 & Month == 'May' & Year == 2015))))) == 0
#> [1] TRUE
sum(! is.na(station_data %>%
  filter((Station == 4 & Month == 'May' & Year == 2015)))) == 0
#> [1] TRUE
```

Correct Sample Row Alignment

I had some funny artifacts popping up in my initial (re) analyses. I finally tracked down the cause. My data was in a different order from Ambrose's version, apparently because I used different tools that have different default ordering. Since NMDS is determined only by a "distance" metric, the NMDS solution is unique only to rotations and reflections. In fact, after the NMDS is fit, the default behavior is to rotate the solution to align the first axis with the largest apparent axis of variation, so the solution returned is unique only to reflections. It looks like ordering of samples can affect which solution the algorithm presents. Only by ensuring uniform ordering can we ensure that output is similar to Ambrose's prior analysis.

Align Data Tables

```
zoopl2 <- zoopl2 %>%
  arrange(DATE, STATION)
station_data<- station_data %>%
  arrange(Date, Station)

head(zoopl2[,c(1,4)])
#> # A tibble: 6 x 2
#>   DATE                STATION
#>   <dttm>              <fct>
#> 1 2013-05-28 00:00:00 1
#> 2 2013-05-28 00:00:00 2
#> 3 2013-05-28 00:00:00 3
#> 4 2013-05-28 00:00:00 4
#> 5 2013-07-25 00:00:00 1
#> 6 2013-07-25 00:00:00 2
head(station_data[,c(1,2)])
#> # A tibble: 6 x 2
#>   Date                Station
#>   <dttm>              <fct>
#> 1 2013-05-28 00:00:00 1
#> 2 2013-05-28 00:00:00 2
#> 3 2013-05-28 00:00:00 3
#> 4 2013-05-28 00:00:00 4
#> 5 2013-07-25 00:00:00 1
#> 6 2013-07-25 00:00:00 2
```

Matrix of Species for vegan

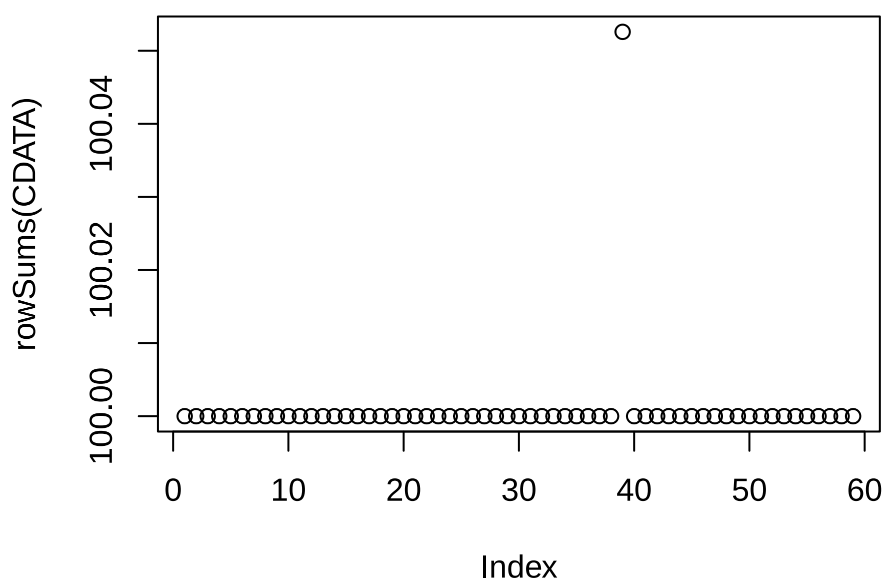
The `vegan` package likes to work with a matrix of species occurrences. Although the matrix can have row names that provide sample identifiers, that was not done here. The "matrix" I produce here is really a data frame with nothing but numeric values. While those are different data structures internally, `vegan` handles the conversion in the background.

```
CDATA <- zoopl2[,-c(1:4)]
```

Data Sanity Checks

We should have no `NA`s, and row sums should all be 1 (100%), at least within reasonable rounding error.

```
anyNA(CDATA)
#> [1] FALSE
plot(rowSums(CDATA))
```



One sample is slightly off the calculation of totals, but the deviation is tiny, so not of any interest.

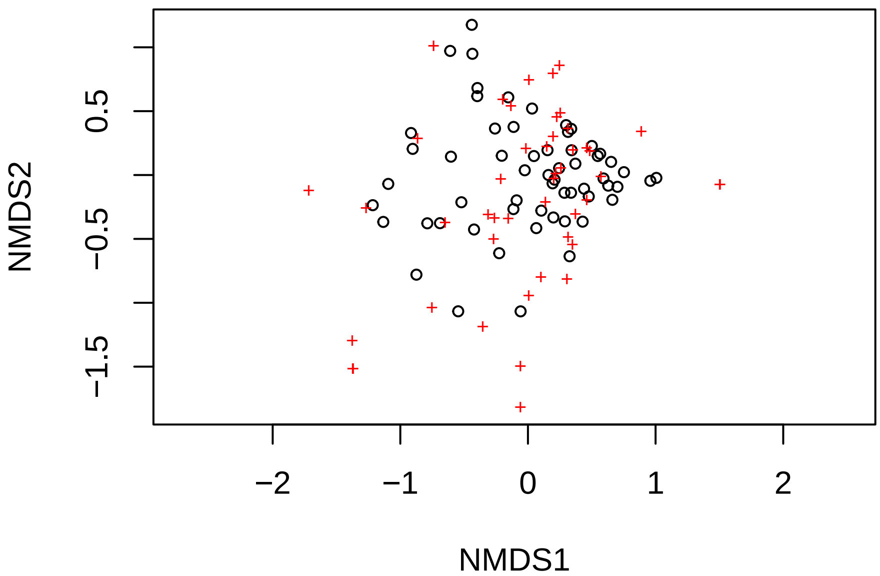
NMDS Analyses

```
NMDSE <- metaMDS(CDATA, autotransform = FALSE, k = 2, trymax = 75)
#> Run 0 stress 0.1509449
#> Run 1 stress 0.1493367
```

```
#> ... New best solution
#> ... Procrustes: rmse 0.01801924  max resid 0.1200198
#> Run 2 stress 0.1671366
#> Run 3 stress 0.1634485
#> Run 4 stress 0.1505044
#> Run 5 stress 0.15446
#> Run 6 stress 0.1790177
#> Run 7 stress 0.1508907
#> Run 8 stress 0.168418
#> Run 9 stress 0.1634487
#> Run 10 stress 0.180874
#> Run 11 stress 0.1669607
#> Run 12 stress 0.1577084
#> Run 13 stress 0.1493367
#> ... Procrustes: rmse 5.821323e-06  max resid 2.408775e-05
#> ... Similar to previous best
#> Run 14 stress 0.1796661
#> Run 15 stress 0.1709702
#> Run 16 stress 0.1628329
#> Run 17 stress 0.23126
#> Run 18 stress 0.15446
#> Run 19 stress 0.1994601
#> Run 20 stress 0.2375539
#> *** Solution reached
NMDSE
#>
#> Call:
#> metaMDS(comm = CDATA, k = 2, trymax = 75, autotransform = FALSE)
#>
#> global Multidimensional Scaling using monoMDS
#>
#> Data:     CDATA
#> Distance: bray
#>
#> Dimensions: 2
#> Stress:     0.1493367
#> Stress type 1, weak ties
#> Two convergent solutions found after 20 tries
#> Scaling: centring, PC rotation, halfchange scaling
#> Species: expanded scores based on 'CDATA'
```

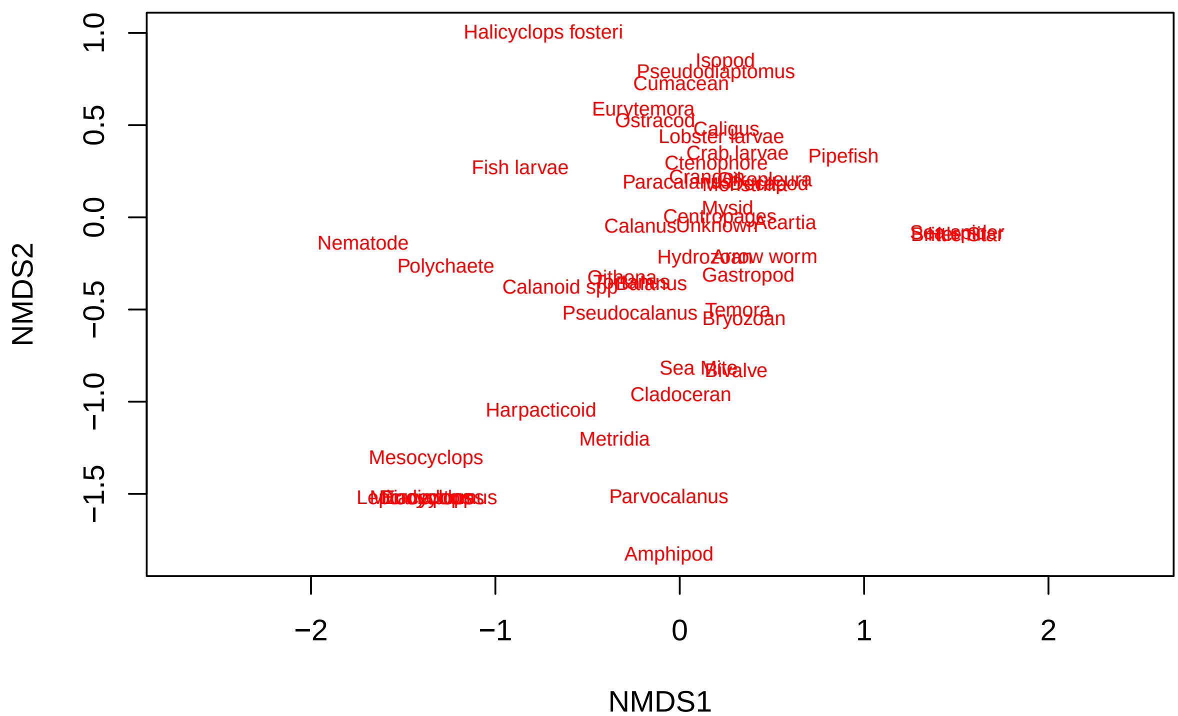
Plot

```
plot(NMDSE, type = 'p')
```



Plot Species

```
plot(NMDSE, 'species', type = 't')
```



Combining the NMDS Results with Environmental Data

I want to use the names of these variables as labels in graphics later. I capitalize variable names here, so they will appear capitalized in graphics without further action on my part.

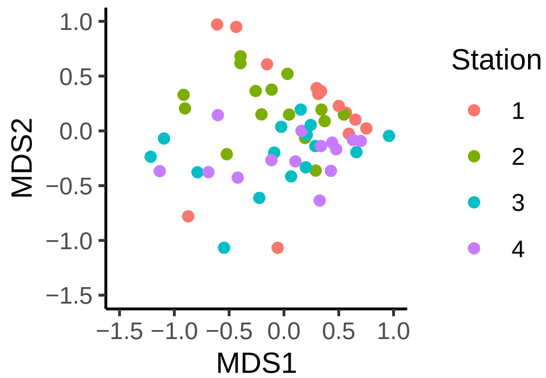
```
envNMDS <- station_data %>%
  select(-Date, -Month, -DOY, -riv_km, -AvgTurb) %>%
  mutate(Turb2 = log(Turb),
         Chl2 = log(Chl),
         RH2 = log1p(RH),
         Fish2 = log1p(Fish)) %>%
  mutate(sample_seq = as.numeric(Season) + (Year-2013)*3,
         sample_event = factor(sample_seq)) %>%
  cbind(as_tibble(NMDSE$points))
```

Graphic Exploration

These plots are intended principally to help us understand the NMDS from a more intuitive perspective. The idea is to plot the ordination, but colored by various predictor variables.

By Station

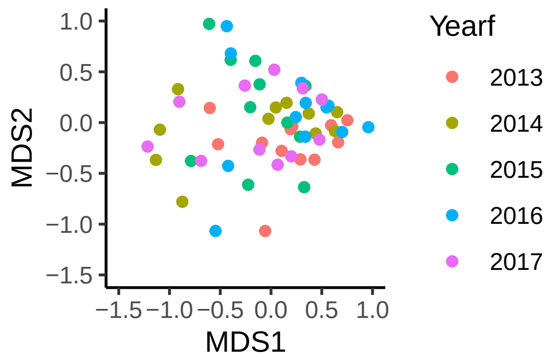
```r
ggplot(envNMDS, aes(MDS1, MDS2)) +
    geom_point(aes(color=Station)) +
  xlim(c(-1.5,1)) +
  ylim(c(-1.5,1)) +
  theme(aspect.ratio=1)
#> Warning: Removed 2 rows containing missing values (geom_point).
```



Note that station 1 is split into a group along the upper edge and two points along the lower edge. The stations don't segregate fully, but there are trends. Other than those two spring samples, Station 1 is upper edge. Station 2 is upper zone as well. I suspect those two samples are "washout" event samples.

By Year

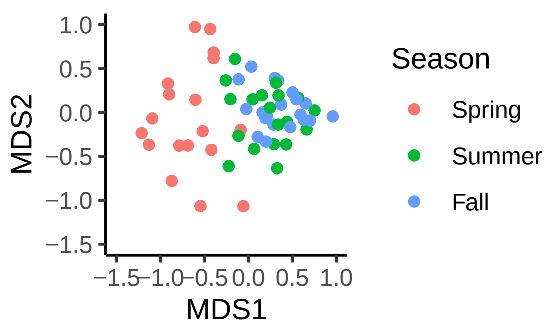
```r
ggplot(envNMDS, aes(MDS1, MDS2)) +
  geom_point(aes(color=Yearf)) +
  xlim(c(-1.5,1)) +
  ylim(c(-1.5,1)) +
  theme(aspect.ratio=1)
#> Warning: Removed 2 rows containing missing values (geom_point).
```



MAYBE 2016 is towards the upper edge, but it's not clear at all. I don't see a robust pattern here.

By Season

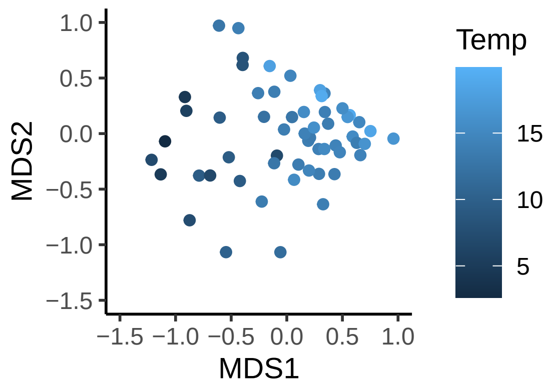
```
ggplot(envNMDS, aes(MDS1, MDS2)) +
  geom_point(aes(color=Season)) +
  xlim(c(-1.5,1)) +
  ylim(c(-1.5,1)) +
  theme(aspect.ratio=1)
#> Warning: Removed 2 rows containing missing values (geom_point).
```



Note the VERY strong association here, with Spring samples all to the left on the plot. Summer and Fall plots are fairly mixed up, but all to the left. That means Axis 1 can be interpreted as largely a "season" signal.

By Temperature

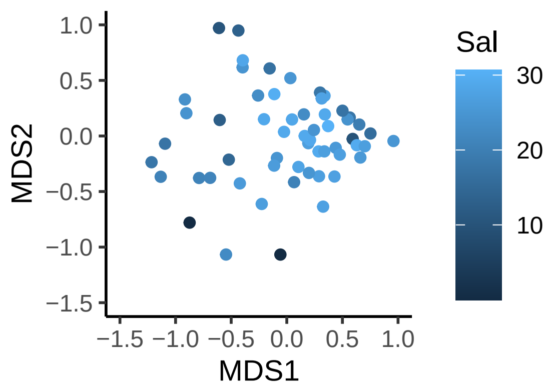
```r
ggplot(envNMDS, aes(MDS1, MDS2)) +
  geom_point(aes(color=Temp)) +
  xlim(c(-1.5,1)) +
  ylim(c(-1.5,1)) +
  theme(aspect.ratio=1)
#> Warning: Removed 2 rows containing missing values (geom_point).
```



This reveals the same pattern as the last graphic, only filtered through the correlation between season and temperature. Cool temperatures in spring to the left.

By Salinity

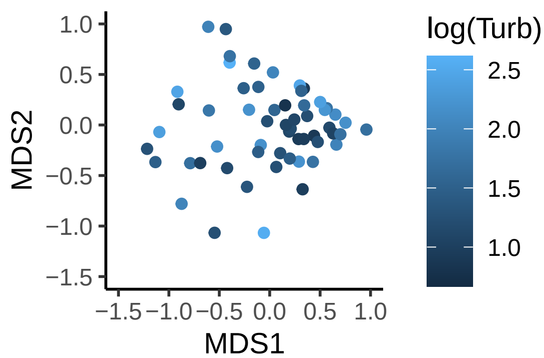
```r
ggplot(envNMDS, aes(MDS1, MDS2)) +
  geom_point(aes(color=Sal)) +
  xlim(c(-1.5,1)) +
  ylim(c(-1.5,1)) +
  theme(aspect.ratio=1)
#> Warning: Removed 2 rows containing missing values (geom_point).
```



This one is hard to interpret. What jumps out at me here is the two VERY low salinity sites at the bottom, and the tendency for other lower salinity samples to fall to the left (spring) and along the upper edge (Station 1).

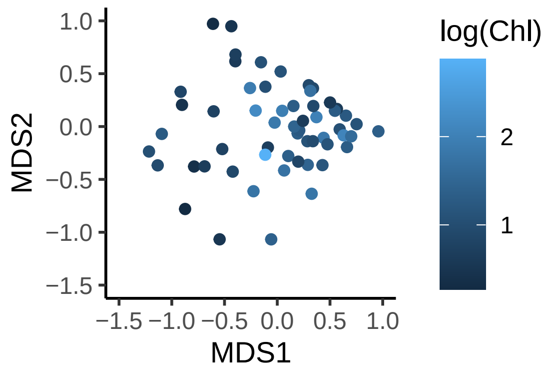
By Turbidity

```
ggplot(envNMDS, aes(MDS1, MDS2)) +
  geom_point(aes(color=log(Turb))) +
  xlim(c(-1.5,1)) +
  ylim(c(-1.5,1)) +
  theme(aspect.ratio=1)
#> Warning: Removed 2 rows containing missing values (geom_point).
```



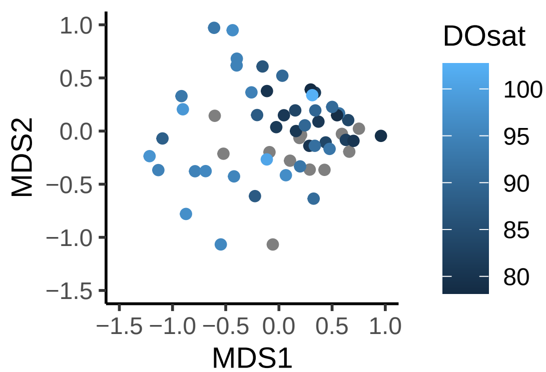
By Chlorophyll

```
ggplot(envNMDS, aes(MDS1, MDS2)) +
  geom_point(aes(color=log(Chl))) +
  xlim(c(-1.5,1)) +
  ylim(c(-1.5,1)) +
  theme(aspect.ratio=1)
#> Warning: Removed 2 rows containing missing values (geom_point).
```



By Oxygen Saturation

```
ggplot(envNMDS, aes(MDS1, MDS2)) +
  geom_point(aes(color=DOsat)) +
  xlim(c(-1.5,1)) +
  ylim(c(-1.5,1)) +
  theme(aspect.ratio=1)
#> Warning: Removed 2 rows containing missing values (geom_point).
```

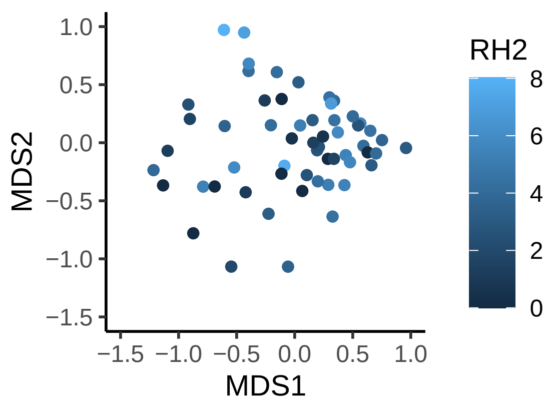


Note that highest DO is to the left, providing an alternate "explanation" to considering axis 1 a seasonal axis.

By Herring

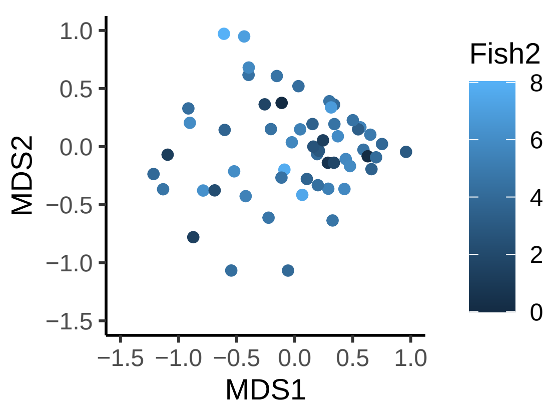
```
ggplot(envNMDS, aes(MDS1, MDS2)) +
  geom_point(aes(color=RH2)) +
  xlim(c(-1.5,1)) +
```

```
  ylim(c(-1.5,1)) +
  theme(aspect.ratio=1)
#> Warning: Removed 2 rows containing missing values (geom_point).
```



By Fish

```
ggplot(envNMDS, aes(MDS1, MDS2)) +
  geom_point(aes(color=Fish2)) +
  xlim(c(-1.5,1)) +
  ylim(c(-1.5,1)) +
  theme(aspect.ratio=1)
#> Warning: Removed 2 rows containing missing values (geom_point).
```



Using `envfit` to Estimate Correlations

The `envfit()` function is fitting linear predictors to the two NMDS axes jointly. The related help file says "The environmental variables are the dependent variables that are explained by the ordination scores, and each dependent variable is analyzed separately.” The model is always linear, which is different from our GAM models.

That means the `envfit()` output is NOT a single multivariate statistical test, but a separate statistical fit for each predictor variable.

Coefficients are the coordinates of a unit-length vector that points along the “direction” in ordination space that shows maximum correlation with the NMDS scores. (Since these are unit vectors, if one coordinate goes up, the other necessarily goes down.) The R2 term “is a”goodness of fit statistic” like the one from multiple regression models. The higher the number, the better the ability of the ordination scores to predict environmental variables

These results are apparently based on randomization methods, so results change somewhat between repeated runs of the following code. The relatively high number of permutations specified here helps keep those effects small.

`envfit()` Single Variable Relationships

```
ef <- envfit(NMDSE, envNMDS[,c(1, 3:15)], permu = 9999, na.rm = TRUE)
ef
#> 
#> ***VECTORS
#> 
#>          NMDS1    NMDS2     r2 Pr(>r)    
#> Temp   0.97606  0.21749 0.7597 0.0001 ***
#> Sal    0.98679 -0.16200 0.0978 0.1068    
#> Turb  -0.33503  0.94221 0.1477 0.0291 *  
#> DOsat -0.97723 -0.21219 0.3353 0.0003 ***
#> Chl    0.72751 -0.68610 0.0782 0.1630    
#> RH    -0.22967  0.97327 0.2436 0.0012 ** 
#> Fish  -0.35281  0.93570 0.1317 0.0463 *  
#> Turb2 -0.34247  0.93953 0.1501 0.0289 *  
#> Chl2   0.85555 -0.51772 0.1660 0.0187 *  
#> RH2    0.25697  0.96642 0.2385 0.0030 ** 
#> Fish2 -0.27060  0.96269 0.0506 0.3376    
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> Permutation: free
#> Number of permutations: 9999
#> 
#> ***FACTORS:
#> 
#> Centroids:
#>                NMDS1   NMDS2
#> Station1      0.1458  0.3011
#> Station2     -0.1540  0.3187
#> Station3     -0.1657 -0.2508
#> Station4      0.0648 -0.2420
#> Yearf2014    -0.0620 -0.0441
#> Yearf2015    -0.1254  0.1201
#> Yearf2016     0.1830  0.0678
#> Yearf2017    -0.1448 -0.0128
#> SeasonSpring -0.7443  0.0037
#> SeasonSummer  0.1803  0.0012
```

```
#> SeasonFall    0.3760  0.0872
#>
#> Goodness of fit:
#>             r2 Pr(>r)
#> Station 0.1959 0.0077 **
#> Yearf   0.0441 0.6838
#> Season  0.4689 0.0001 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> Permutation: free
#> Number of permutations: 9999
#>
#> 13 observations deleted due to missingness
```

Missing Values

Note 13 observations deleted due to missingness. Those are the 2013 data, which lacks DO saturation data. We can refit to include those data by dropping DO as a predictor. That might alter some fits.

`envfit()` Not Including Oxygen

```
ef_2 <- envfit(NMDSE, envNMDS[,c(1, 3:7, 9:15)], permu = 9999, na.rm = TRUE)
ef_2
#>
#> ***VECTORS
#>
#>          NMDS1    NMDS2     r2 Pr(>r)
#> Temp   0.96696  0.25494 0.7434 0.0001 ***
#> Sal    0.88012  0.47475 0.0737 0.1188
#> Turb  -0.57815  0.81593 0.0299 0.4286
#> Chl    0.77421 -0.63293 0.0623 0.1684
#> RH    -0.28443  0.95870 0.1316 0.0211 *
#> Fish  -0.43623  0.89984 0.0847 0.0837 .
#> Turb2 -0.53319  0.84600 0.0355 0.3668
#> Chl2   0.86764 -0.49720 0.1631 0.0070 **
#> RH2    0.34366  0.93909 0.1315 0.0195 *
#> Fish2 -0.36716  0.93016 0.0268 0.4744
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> Permutation: free
#> Number of permutations: 9999
#>
#> ***FACTORS:
#>
#> Centroids:
#>                NMDS1   NMDS2
#> Station1      0.2065  0.1600
#> Station2     -0.1257  0.2123
#> Station3     -0.0805 -0.2292
#> Station4      0.0459 -0.2259
#> Yearf2013     0.1633 -0.2201
```

```
#> Yearf2014    -0.0620 -0.0441
#> Yearf2015    -0.1254  0.1201
#> Yearf2016     0.1830  0.0678
#> Yearf2017    -0.1448 -0.0128
#> SeasonSpring -0.6495 -0.0713
#> SeasonSummer  0.2508 -0.0440
#> SeasonFall    0.3557  0.0494
#>
#> Goodness of fit:
#>             r2 Pr(>r)
#> Station 0.1305 0.0207 *
#> Yearf   0.0739 0.3908
#> Season  0.4338 0.0001 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#> Permutation: free
#> Number of permutations: 9999
#>
#> 1 observation deleted due to missingness
```

Note that River Herring and log(River Herring + 1) are significantly correlated with the MSDS, while total fish is at best marginally significantly correlated.

Extracting Vector Information

The `ef` or `ef_2` object is an `envfit` S3 object, with three named slots. The vector information we need to plot the environment arrows is available in `vectors`. But that object is itself also an S3 object, with five named items. The help page for `envfit()` tells us that the information on the direction of the arrows is in the `arrows` component. We are told that arrows contain "Arrow endpoints from `vectorfit`. The arrows are scaled to unit length."

```
ef_2$vectors$arrows
#>            NMDS1      NMDS2
#> Temp   0.9669560  0.2549435
#> Sal    0.8801207  0.4747500
#> Turb  -0.5781501  0.8159305
#> Chl    0.7742128 -0.6329254
#> RH    -0.2844316  0.9586963
#> Fish  -0.4362265  0.8998369
#> Turb2 -0.5331901  0.8459954
#> Chl2   0.8676350 -0.4972016
#> RH2    0.3436627  0.9390931
#> Fish2 -0.3671601  0.9301577
#> attr(,"decostand")
#> [1] "normalize"
```

The information we need to determine the magnitude of those vectors is in the `r` component of the `vectors` component, which (according to the `envfit()` help file) contains "Goodness of fit statistic: Squared correlation coefficient".

The correlation coefficient is (formally) a bivariate statistic, but here it is being used to indicate the strength of association between two predictors (NMDS Axis 1 and NMDS axis 2) and each environmental variable. I have not been able to find clear documentation of what is going on, but it appears the R squared value is (or is analogous to) the R squared value reported the implied (two predictor, one response) linear regression. If that is the case, the value of r can be (roughly) interpreted as the correlation coefficient between the best linear combination of NMDS Axis 1 and Axis 2 and each environmental variable.

It's worth pointing out that the r values are mostly pretty low. The NMDS ordination does not do a very good job of "predicting" environmental variables, except for Temperature, which it does quite well. Here are the implied correlation coefficients:

```
sqrt(ef_2$vectors$r)
#>      Temp       Sal      Turb       Chl        RH      Fish     Turb2      Chl2
#> 0.8621835 0.2715296 0.1730423 0.2496265 0.3627923 0.2910821 0.1884242 0.4038664
#>       RH2     Fish2
#> 0.3626774 0.1636947
```

For plotting, we scale the length of each of the arrows by the associated correlation coefficient (square root of the r squared value). The higher the correlation coefficient, the longer the arrow. Thus arrow direction shows the mix of Axis 1 and Axis 2 that correlates best with each environmental variable, while the length of the arrow shows the relative ability of the ordination to predict environmental variables.

```
arrows <- ef_2$vectors$arrows
rsq    <- ef_2$vectors$r
scaled_arrows <- as_tibble(arrows*sqrt(rsq)) %>%
  mutate(parameter = rownames(arrows))
```

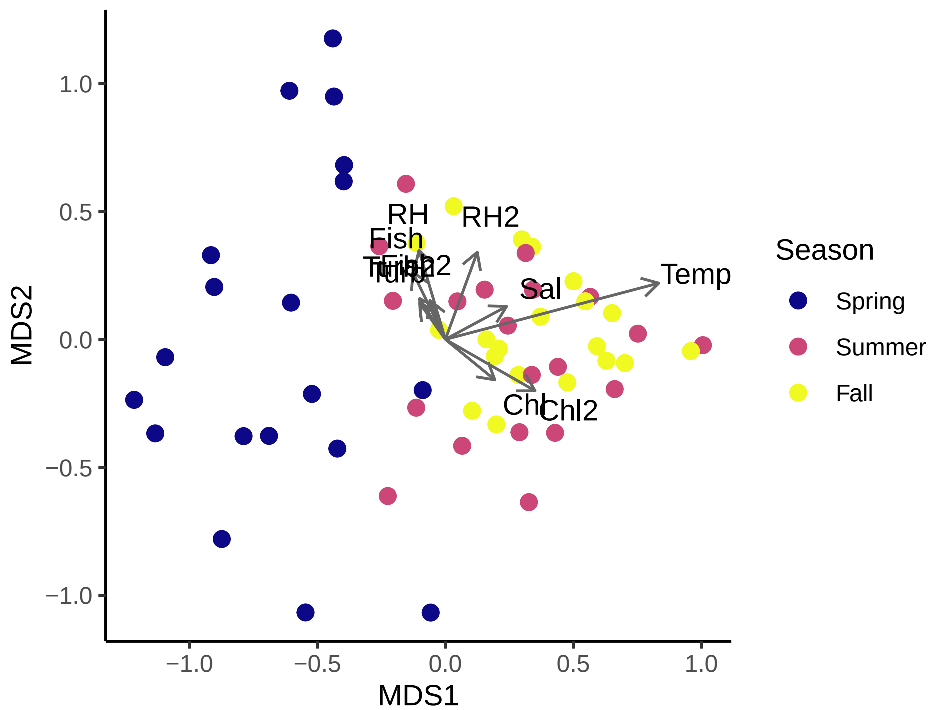
```
scale_factor = .15  # Scale position of text annotations this far off
                    # end or arrow

scaled_arrows <- scaled_arrows %>%
  mutate(ann_xpos = NMDS1 + arrows[,1] * scale_factor,
         ann_ypos = NMDS2 + arrows[,2] * scale_factor)
```

While we are creating vectors, we also want to create points for placing the annotations identifying each vector. We want to space the labels so they are a fixed distance beyond the end of each vector. We do that with a little vector addition.

Draft Graphic

```
plt <- ggplot(data = envNMDS, aes(MDS1, MDS2)) +
  geom_point(aes(color = Season), size = 2.5) +
  geom_segment(data=scaled_arrows,
               mapping = aes(x=0,xend=NMDS1,y=0,yend=NMDS2),
               arrow = arrow(length = unit(0.25, "cm")) ,colour="grey40") +
  geom_text(data=scaled_arrows,
            mapping = aes(x= ann_xpos,
                          y= ann_ypos,label=parameter),
            size=4, nudge_x =0, nudge_y = 0, hjust = .5)+
  scale_color_viridis_d(option = 'C', name = 'Season') +
  coord_fixed()
plt
```



That's too messy. We want to plot a smaller number of arrows. I recommend showing only the transformed variables, since that is what we use in the GAMs.

Possible Publication Graphics

The instructions to authors suggests figure widths should line up with columns, and proposes figure widths should be: 39, 84, 129, or 174 mm wide, with height not to exceed 235 mm. Presumably that corresponds to 1,2,3,or 4 columns wide?

39 mm is about one and one half inches, which his quite small, so we will use the 84 mm wide option, which is about 3.3 inches.

All (Transformed) Environmental Variables

```
tmp_arrows <- scaled_arrows %>%
  filter(parameter %in% c('Temp', 'Sal', 'Turb2', 'Chl2',
                          'Fish2', 'RH2')) %>%
  mutate(parameter = factor(parameter,
                            levels = c('Temp', 'Sal', 'Turb2', 'Chl2',
                                       'Fish2', 'RH2'),
                            labels = c('Temp', 'Sal', 'Turb', 'Chl',
                          'Fish', 'RH')))

plt <- ggplot(data = envNMDS, aes(MDS1, MDS2)) +
  geom_point(aes(color = Season, shape = Station), size = 1.5) +
  geom_segment(data=tmp_arrows,
```

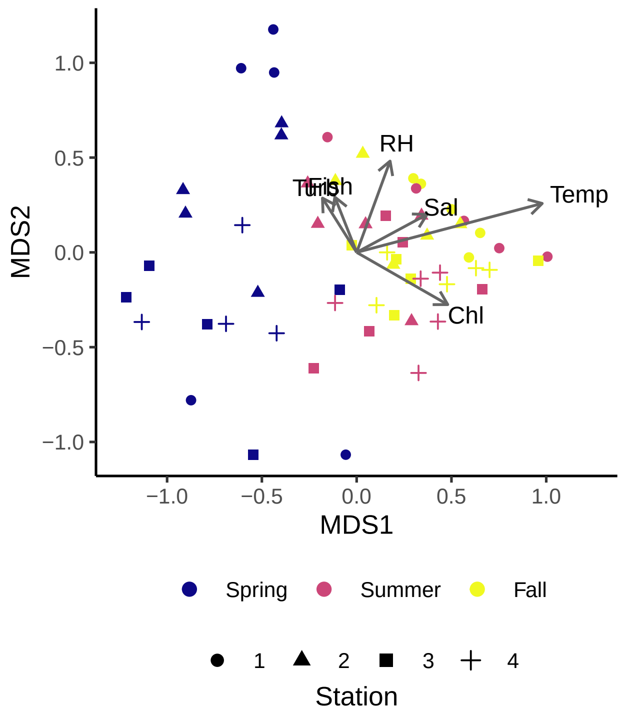
```
               mapping = aes(x=0,xend=ann_xpos,y=0,yend=ann_ypos),
               arrow = arrow(length = unit(0.2, "cm")) ,colour="grey40") +
  geom_text(data=tmp_arrows,
            mapping = aes(x=1.2 * ann_xpos,
                          y=1.2 * ann_ypos,label=parameter),
            size=3, nudge_x =0, nudge_y = 0, hjust = .5)+

  scale_color_viridis_d(option = 'C', name = '') +
  scale_shape(name = 'Station') +

  coord_fixed(xlim =c(-1.25, 1.25)) +

  guides(color = guide_legend(title.position = "bottom",
                              title.hjust = 0.5,
                              override.aes = list(size = 2),
                              order = 1),
         shape = guide_legend(title.position = "bottom",
                              title.hjust = 0.5,
                              override.aes = list(size = 2),
                              order = 2))
```

```
plt +
  theme_classic(base_size = 10) +
  theme(
        legend.position = 'bottom',
        legend.box = 'Vertical',
        legend.spacing.y = unit(0, 'cm'),
        legend.margin = margin(0,0,0,0)
        )
```




```
ggsave('figures/nmds_env.png', type='cairo',
         width = 3.3, height = 3.4)
ggsave('figures/nmds_env.pdf', device = cairo_pdf,
       width = 3.3, height = 3.4)
```

That is still too busy.

1. It is perhaps problematic to show both Fish and River Herring.
2. We show several arrows that are not statistically robust. We can simplify by showing only statistically significant correlations with environmental variables.

Statistically Significant Environmental Variable

```
tmp <- tmp_arrows %>%
  filter(parameter %in% c('Temp', 'Chl', 'RH'))

plt <- ggplot(data = envNMDS, aes(MDS1, MDS2)) +
  geom_point(aes(color = Season, shape = Station), size = 1.5) +
  geom_segment(data=tmp,
               mapping = aes(x=0,xend=ann_xpos,y=0,yend=ann_ypos),
               arrow = arrow(length = unit(0.2, "cm")) ,colour="grey40") +
  geom_text(data=tmp,
            mapping = aes(x=1.2 * ann_xpos,
```

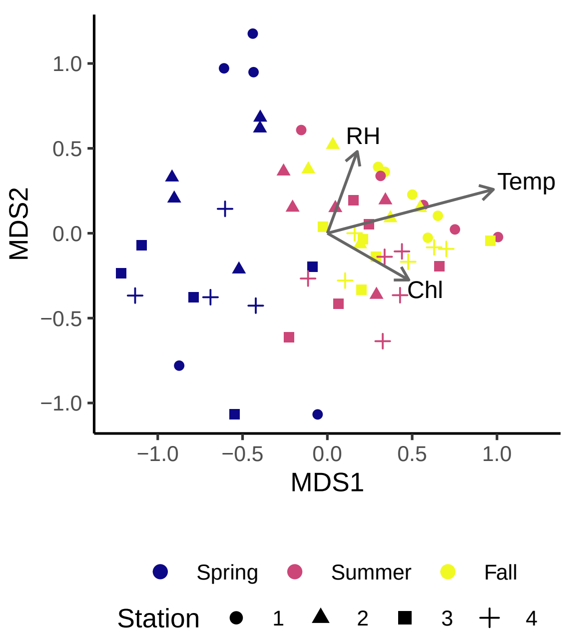
```
                          y=1.2 * ann_ypos,label=parameter),
              size=3, nudge_x =0, nudge_y = 0, hjust = .5)+
  scale_color_viridis_d(option = 'C', name = '') +
  scale_shape(name = 'Station') +

  coord_fixed(xlim =c(-1.25, 1.25)) +

  guides(color = guide_legend(title.position = "top",
                              title.hjust = 0.5,
                              override.aes = list(size = 2),
                              order = 1),
         shape = guide_legend(title.position = "left",
                              title.hjust = 0.5,
                              override.aes = list(size = 2),
                              order = 2))
rm(tmp)
```

We can't change the figure width, but we can alter the figure height. With that in mind, let's reorient the legends and juggle dimensions

```
plt +
  theme_classic(base_size = 10) +
  theme(
        legend.position = 'bottom',
        legend.box = 'Vertical',
        legend.spacing.y = unit(0, 'cm'),
        legend.box.just = 'top',
        legend.margin = margin(0,0,0,0),
        legend.box.margin = margin(0,0,0,0)
        )
```



```
ggsave('figures/nmds_env_significant.png', type='cairo',
         width = 3.3, height = 3.4)
ggsave('figures/nmds_env_significant.pdf', device = cairo_pdf,
       width = 3.3, height = 3.4)
```

Drop River Herring

If we don't EVER want to drag river herring in as a predictor in the manuscript, we should not show it in the NMDS Plots either.

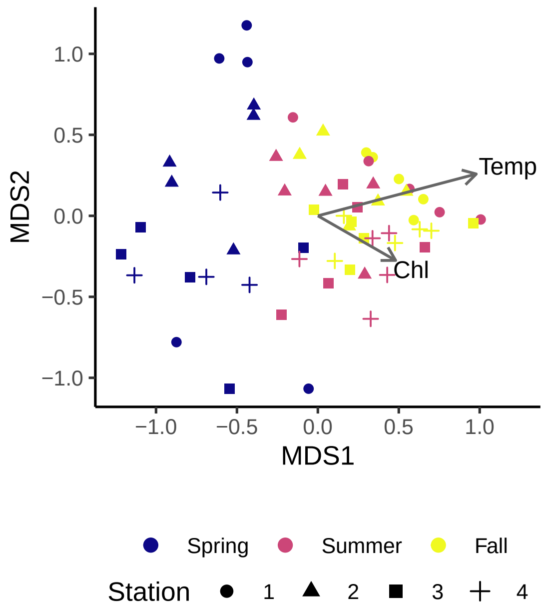
```
tmp <- tmp_arrows %>%
  filter(parameter %in% c('Temp', 'Chl'))

plt <- ggplot(data = envNMDS, aes(MDS1, MDS2)) +
  geom_point(aes(color = Season, shape = Station), size = 1.5) +
  geom_segment(data=tmp,
               mapping = aes(x=0,xend=ann_xpos,y=0,yend=ann_ypos),
               arrow = arrow(length = unit(0.2, "cm")) ,colour="grey40") +
  geom_text(data=tmp,
            mapping = aes(x=1.2 * ann_xpos,
                          y=1.2 * ann_ypos,label=parameter),
            size=3, nudge_x =0, nudge_y = 0, hjust = .5)+
  scale_color_viridis_d(option = 'C', name = '') +
  scale_shape(name = 'Station') +

  coord_fixed(xlim =c(-1.25, 1.25)) +

  guides(color = guide_legend(title.position = "top",
                              title.hjust = 0.5,
                              override.aes = list(size = 2),
                              order = 1),
         shape = guide_legend(title.position = "left",
                              title.hjust = 0.5,
                              override.aes = list(size = 2),
                              order = 2))
rm(tmp)
```

```
plt +
  theme_classic(base_size = 10) +
  theme(
        legend.position = 'bottom',
        legend.box = 'Vertical',
        legend.spacing.y = unit(0, 'cm'),
        legend.box.just = 'top',
        legend.margin = margin(0,0,0,0),
        legend.box.margin = margin(0,0,0,0)
        )
```




```
ggsave('figures/nmds_env_significant_no_RH.png', type='cairo',
         width = 3.3, height = 3.4)
ggsave('figures/nmds_env_significant_no_RH', device = cairo_pdf,
       width = 3.3, height = 3.4)
```

Qualitative Conclusions

- Axis 1 is highly correlated with season, and thus highly correlated with temperature and oxygen saturation (which was dropped from this analysis because of missing 2013 data).
- Axis 2 is closely related to Station, with upstream stations to the upper right and downstream stations to the lower left. Two "oddball" Station 1 samples are extreme low salinity spring samples – the same samples that cause problems fitting many of our models. The second axis is not especially correlated with ANY of the environmental variables.
- I am uncertain how to interpret the Chlorophyll association.

GAM Analysis

Axis 1

Note I increase the iterations to fit the model. This probably means the gradient near the solution is low, so parameter estimates may not bee very good.

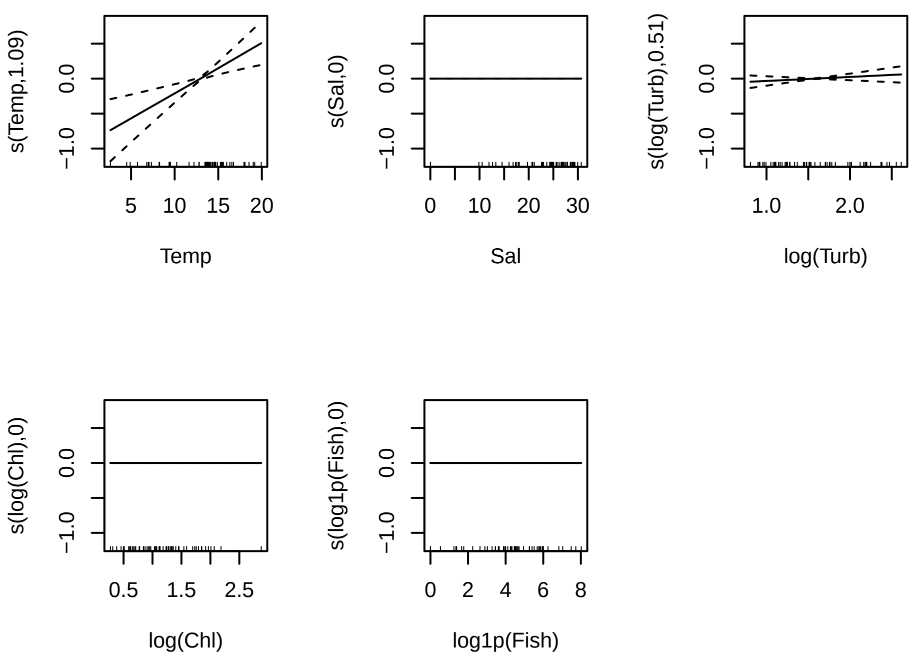
```
gam_1 <- gamm(MDS1 ~
               Station +
               Season +
               s(Temp, bs="ts") +
               s(Sal, bs="ts") +
               s(log(Turb), bs="ts") +
```

```
               s(log(Chl), bs="ts") +
               s(log1p(Fish),bs="ts"),
              random = list(Yearf = ~ 1, sample_event = ~ 1),
             data = envNMDS, family = 'gaussian',
             control = list(msMaxIter = 1000, msMaxEval = 1000, niterEM=0))
summary(gam_1$gam)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> MDS1 ~ Station + Season + s(Temp, bs = "ts") + s(Sal, bs = "ts") +
#>     s(log(Turb), bs = "ts") + s(log(Chl), bs = "ts") + s(log1p(Fish),
#>     bs = "ts")
#>
#> Parametric coefficients:
#>               Estimate Std. Error t value Pr(>|t|)
#> (Intercept)   -0.35894    0.10912  -3.289  0.00184 **
#> Station2      -0.01671    0.11852  -0.141  0.88846
#> Station3       0.00953    0.11343   0.084  0.93338
#> Station4       0.10775    0.12010   0.897  0.37388
#> SeasonSummer   0.42155    0.16930   2.490  0.01612 *
#> SeasonFall     0.56096    0.16082   3.488  0.00102 **
#> ---
#> 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)         1.094e+00      9 1.423 0.00163 **
#> s(Sal)          8.419e-08      9 0.000 0.30865
#> s(log(Turb))    5.057e-01      9 0.129 0.17844
#> s(log(Chl))     1.337e-07      9 0.000 0.84906
#> s(log1p(Fish)) 2.453e-07      9 0.000 0.60731
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =   0.76
#>   Scale est. = 0.050924  n = 58
```

```
anova(gam_1$gam)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> MDS1 ~ Station + Season + s(Temp, bs = "ts") + s(Sal, bs = "ts") +
#>     s(log(Turb), bs = "ts") + s(log(Chl), bs = "ts") + s(log1p(Fish),
#>     bs = "ts")
#>
#> Parametric Terms:
#>         df     F p-value
#> Station  3 0.686 0.56487
#> Season   2 6.867 0.00231
```

```
#>
#> Approximate significance of smooth terms:
#>                      edf    Ref.df     F p-value
#> s(Temp)        1.094e+00 9.000e+00 1.423 0.00163
#> s(Sal)         8.419e-08 9.000e+00 0.000 0.30865
#> s(log(Turb))   5.057e-01 9.000e+00 0.129 0.17844
#> s(log(Chl))    1.337e-07 9.000e+00 0.000 0.84906
#> s(log1p(Fish)) 2.453e-07 9.000e+00 0.000 0.60731
```

```
oldpar <- par(mfrow = c(2,3))
plot(gam_1$gam)
par(oldpar)
```

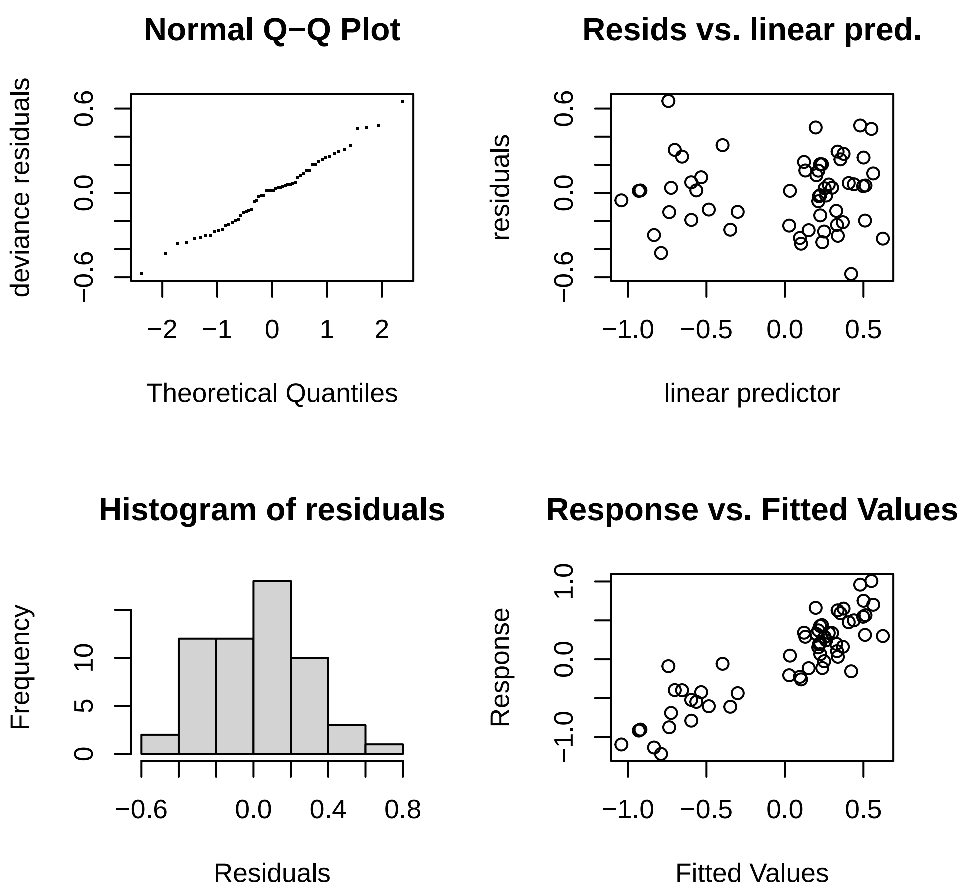



```
(emms <- emmeans(gam_1, pairwise ~ Season, type = 'response',
                 data = envNMDS))
#> $emmeans
#>  Season emmean     SE   df lower.CL upper.CL
#>  Spring -0.329 0.1263 50.4  -0.5822  -0.0749
#>  Summer  0.093 0.0926 50.4  -0.0930   0.2790
#>  Fall    0.232 0.0870 50.4   0.0577   0.4071
#>
#> Results are averaged over the levels of: Station
#> Confidence level used: 0.95
#>
#> $contrasts
#>  contrast        estimate     SE   df t.ratio p.value
#>  Spring - Summer   -0.422 0.1693 50.4  -2.490  0.0419
```

```
#>  Spring - Fall     -0.561 0.1608 50.4  -3.488  0.0029
#>  Summer - Fall     -0.139 0.0804 50.4  -1.733  0.2029
#>
#> Results are averaged over the levels of: Station
#> P value adjustment: tukey method for comparing a family of 3 estimates
```

As we saw from the `envfit()` analysis, Axis 1 is associated with the SEASON, with spring clearly separated from Summer and Fall. Temperature is also important in the GAM analysis (note that the relationship is essentially linear). While Turbidity is not shrunk out of the model entirely, the relationship does not reach statistical significance. Essentially, low Axis 1 Scores are spring, with lower temperatures.

```
oldpar <- par(mfrow = c(2,2))
gam.check(gam_1$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 1.09e+00    1.01    0.50
#> s(Sal)         9.00e+00 8.42e-08    1.12    0.80
#> s(log(Turb))   9.00e+00 5.06e-01    0.88    0.19
#> s(log(Chl))    9.00e+00 1.34e-07    0.96    0.33
#> s(log1p(Fish)) 9.00e+00 2.45e-07    1.08    0.68
par(oldpar)
```

The model is fairly well behaved. No obvious problems here.

Axis 2

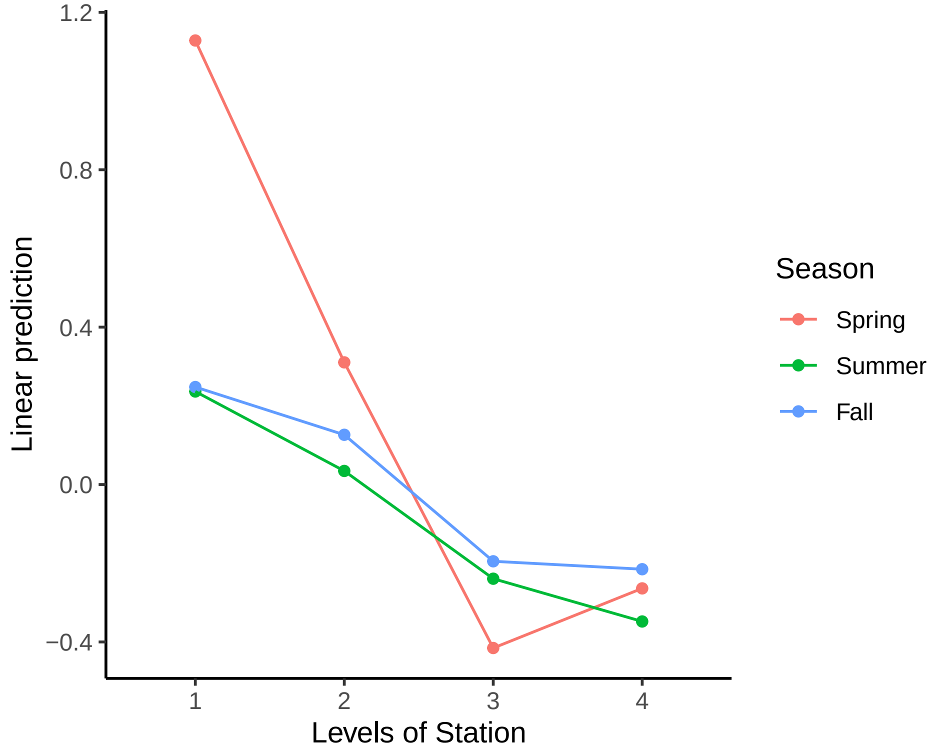
This one converges readily, but only if I include the Season:Station interaction term..

```
gam_2 <- gamm(MDS2 ~
               Station *
               Season +
               s(Temp, bs="ts") +
               s(Sal, bs="ts") +
               s(log(Turb), bs="ts") +
               s(log(Chl), bs="ts") +
               s(log1p(Fish),bs="ts"),
              random = list(Yearf = ~ 1, sample_event = ~ 1),
             data = envNMDS, family = 'gaussian')
summary(gam_2$gam)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> MDS2 ~ Station * Season + s(Temp, bs = "ts") + s(Sal, bs = "ts") +
#>     s(log(Turb), bs = "ts") + s(log(Chl), bs = "ts") + s(log1p(Fish),
#>     bs = "ts")
#>
#> Parametric coefficients:
#>                      Estimate Std. Error t value Pr(>|t|)
#> (Intercept)            1.0880     0.2276   4.780 2.21e-05 ***
#> Station2              -0.8180     0.2605  -3.140 0.003105 **
#> Station3              -1.5440     0.2632  -5.867 6.39e-07 ***
#> Station4              -1.3924     0.2615  -5.324 3.79e-06 ***
#> SeasonSummer          -0.8921     0.2415  -3.693 0.000638 ***
#> SeasonFall            -0.8807     0.2431  -3.623 0.000786 ***
#> Station2:SeasonSummer   0.6161     0.2881   2.139 0.038383 *
#> Station3:SeasonSummer   1.0682     0.2918   3.661 0.000702 ***
#> Station4:SeasonSummer   0.8079     0.2971   2.720 0.009495 **
#> Station2:SeasonFall     0.6966     0.2872   2.426 0.019702 *
#> Station3:SeasonFall     1.1011     0.2994   3.678 0.000669 ***
#> Station4:SeasonFall     0.9294     0.2986   3.112 0.003351 **
#> ---
#> 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)        3.469e-09      9 0.000    0.797
```

```
#> s(Sal)         3.742e+00      9 7.485 1.51e-06 ***
#> s(log(Turb))   4.207e-09      9 0.000    0.577
#> s(log(Chl))    2.897e-09      9 0.000    0.830
#> s(log1p(Fish)) 6.751e-01      9 0.239    0.158
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.606
#>   Scale est. = 0.052015  n = 58
```

```
anova(gam_2$gam)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> MDS2 ~ Station * Season + s(Temp, bs = "ts") + s(Sal, bs = "ts") +
#>     s(log(Turb), bs = "ts") + s(log(Chl), bs = "ts") + s(log1p(Fish),
#>     bs = "ts")
#>
#> Parametric Terms:
#>                df      F  p-value
#> Station         3 15.380 7.04e-07
#> Season          2  7.568  0.00157
#> Station:Season  6  2.892  0.01899
#>
#> Approximate significance of smooth terms:
#>                      edf    Ref.df     F  p-value
#> s(Temp)        3.469e-09 9.000e+00 0.000    0.797
#> s(Sal)         3.742e+00 9.000e+00 7.485 1.51e-06
#> s(log(Turb))   4.207e-09 9.000e+00 0.000    0.577
#> s(log(Chl))    2.897e-09 9.000e+00 0.000    0.830
#> s(log1p(Fish)) 6.751e-01 9.000e+00 0.239    0.158
```

```
emmip(gam_2 , Season ~Station,  data = envNMDS)
```




```
(emms <- emmeans(gam_2, pairwise ~ Station:Season,  data = envNMDS))
#> $emmeans
#>  Station Season  emmean    SE   df lower.CL upper.CL
#>  1       Spring  1.1285 0.247 41.6   0.6306   1.6264
#>  2       Spring  0.3105 0.126 41.6   0.0566   0.5644
#>  3       Spring -0.4155 0.124 41.6  -0.6660  -0.1649
#>  4       Spring -0.2639 0.141 41.6  -0.5479   0.0202
#>  1       Summer  0.2365 0.149 41.6  -0.0642   0.5371
#>  2       Summer  0.0346 0.145 41.6  -0.2588   0.3279
#>  3       Summer -0.2393 0.125 41.6  -0.4909   0.0123
#>  4       Summer -0.3480 0.144 41.6  -0.6392  -0.0568
#>  1       Fall    0.2478 0.142 41.6  -0.0379   0.5336
#>  2       Fall    0.1264 0.149 41.6  -0.1744   0.4272
#>  3       Fall   -0.1950 0.151 41.6  -0.4995   0.1095
#>  4       Fall   -0.2152 0.159 41.6  -0.5361   0.1058
#>
#> Confidence level used: 0.95
#>
#> $contrasts
#>  contrast                          estimate    SE   df t.ratio p.value
#>  Station1 Spring - Station2 Spring   0.8180 0.260 41.6   3.140  0.1067
#>  Station1 Spring - Station3 Spring   1.5440 0.263 41.6   5.867  <.0001
#>  Station1 Spring - Station4 Spring   1.3924 0.262 41.6   5.324  0.0002
#>  Station1 Spring - Station1 Summer   0.8921 0.242 41.6   3.693  0.0274
#>  Station1 Spring - Station2 Summer   1.0939 0.280 41.6   3.909  0.0153
#>  Station1 Spring - Station3 Summer   1.3678 0.276 41.6   4.963  0.0007
#>  Station1 Spring - Station4 Summer   1.4765 0.280 41.6   5.275  0.0003
#>  Station1 Spring - Station1 Fall     0.8807 0.243 41.6   3.623  0.0330
#>  Station1 Spring - Station2 Fall     1.0021 0.285 41.6   3.518  0.0432
```

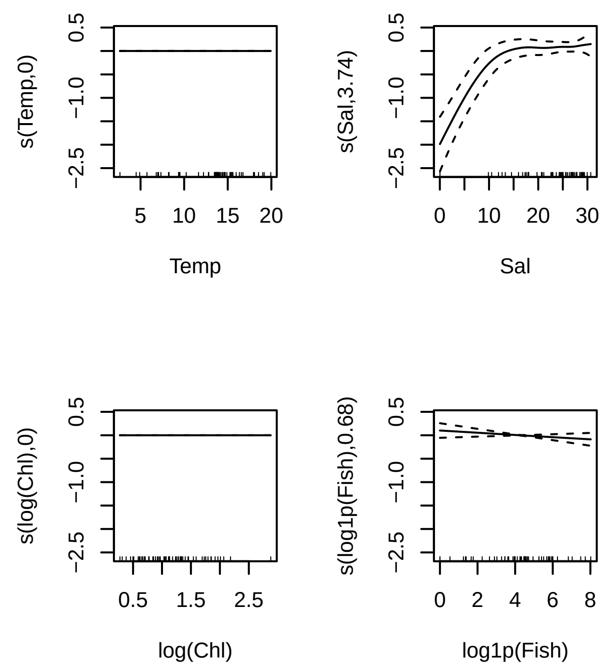
```
#>  Station1 Spring - Station3 Fall     1.3235 0.286 41.6   4.630  0.0019
#>  Station1 Spring - Station4 Fall     1.3437 0.288 41.6   4.662  0.0017
#>  Station2 Spring - Station3 Spring   0.7260 0.169 41.6   4.289  0.0052
#>  Station2 Spring - Station4 Spring   0.5744 0.178 41.6   3.231  0.0868
#>  Station2 Spring - Station1 Summer   0.0740 0.176 41.6   0.421  1.0000
#>  Station2 Spring - Station2 Summer   0.2759 0.172 41.6   1.608  0.8963
#>  Station2 Spring - Station3 Summer   0.5498 0.166 41.6   3.309  0.0722
#>  Station2 Spring - Station4 Summer   0.6585 0.170 41.6   3.868  0.0172
#>  Station2 Spring - Station1 Fall     0.0627 0.174 41.6   0.360  1.0000
#>  Station2 Spring - Station2 Fall     0.1841 0.175 41.6   1.055  0.9951
#>  Station2 Spring - Station3 Fall     0.5055 0.175 41.6   2.892  0.1812
#>  Station2 Spring - Station4 Fall     0.5257 0.179 41.6   2.937  0.1652
#>  Station3 Spring - Station4 Spring  -0.1516 0.175 41.6  -0.866  0.9991
#>  Station3 Spring - Station1 Summer  -0.6519 0.172 41.6  -3.783  0.0216
#>  Station3 Spring - Station2 Summer  -0.4500 0.184 41.6  -2.445  0.4004
#>  Station3 Spring - Station3 Summer  -0.1762 0.170 41.6  -1.037  0.9957
#>  Station3 Spring - Station4 Summer  -0.0675 0.184 41.6  -0.368  1.0000
#>  Station3 Spring - Station1 Fall    -0.6633 0.169 41.6  -3.933  0.0144
#>  Station3 Spring - Station2 Fall    -0.5419 0.187 41.6  -2.902  0.1777
#>  Station3 Spring - Station3 Fall    -0.2204 0.187 41.6  -1.177  0.9880
#>  Station3 Spring - Station4 Fall    -0.2003 0.193 41.6  -1.040  0.9956
#>  Station4 Spring - Station1 Summer  -0.5003 0.180 41.6  -2.774  0.2284
#>  Station4 Spring - Station2 Summer  -0.2985 0.189 41.6  -1.575  0.9083
#>  Station4 Spring - Station3 Summer  -0.0246 0.178 41.6  -0.138  1.0000
#>  Station4 Spring - Station4 Summer   0.0841 0.189 41.6   0.446  1.0000
#>  Station4 Spring - Station1 Fall    -0.5117 0.177 41.6  -2.899  0.1788
#>  Station4 Spring - Station2 Fall    -0.3903 0.192 41.6  -2.038  0.6662
#>  Station4 Spring - Station3 Fall    -0.0688 0.192 41.6  -0.359  1.0000
#>  Station4 Spring - Station4 Fall    -0.0487 0.196 41.6  -0.249  1.0000
#>  Station1 Summer - Station2 Summer   0.2019 0.191 41.6   1.055  0.9951
#>  Station1 Summer - Station3 Summer   0.4757 0.185 41.6   2.567  0.3300
#>  Station1 Summer - Station4 Summer   0.5844 0.192 41.6   3.044  0.1318
#>  Station1 Summer - Station1 Fall    -0.0114 0.164 41.6  -0.069  1.0000
#>  Station1 Summer - Station2 Fall     0.1100 0.196 41.6   0.563  1.0000
#>  Station1 Summer - Station3 Fall     0.4315 0.195 41.6   2.207  0.5539
#>  Station1 Summer - Station4 Fall     0.4516 0.198 41.6   2.275  0.5086
#>  Station2 Summer - Station3 Summer   0.2739 0.172 41.6   1.593  0.9018
#>  Station2 Summer - Station4 Summer   0.3826 0.166 41.6   2.310  0.4861
#>  Station2 Summer - Station1 Fall    -0.2133 0.187 41.6  -1.141  0.9907
#>  Station2 Summer - Station2 Fall    -0.0918 0.164 41.6  -0.558  1.0000
#>  Station2 Summer - Station3 Fall     0.2296 0.163 41.6   1.406  0.9557
#>  Station2 Summer - Station4 Fall     0.2498 0.164 41.6   1.524  0.9253
#>  Station3 Summer - Station4 Summer   0.1087 0.169 41.6   0.644  0.9999
#>  Station3 Summer - Station1 Fall    -0.4871 0.179 41.6  -2.718  0.2535
#>  Station3 Summer - Station2 Fall    -0.3657 0.173 41.6  -2.113  0.6172
#>  Station3 Summer - Station3 Fall    -0.0443 0.173 41.6  -0.256  1.0000
#>  Station3 Summer - Station4 Fall    -0.0241 0.177 41.6  -0.136  1.0000
#>  Station4 Summer - Station1 Fall    -0.5958 0.188 41.6  -3.173  0.0990
#>  Station4 Summer - Station2 Fall    -0.4744 0.169 41.6  -2.812  0.2121
#>  Station4 Summer - Station3 Fall    -0.1530 0.167 41.6  -0.917  0.9985
#>  Station4 Summer - Station4 Fall    -0.1328 0.167 41.6  -0.795  0.9996
#>  Station1 Fall - Station2 Fall       0.1214 0.191 41.6   0.636  1.0000
#>  Station1 Fall - Station3 Fall       0.4429 0.191 41.6   2.322  0.4779
```

```
#>  Station1 Fall - Station4 Fall       0.4630 0.194 41.6   2.387  0.4367
#>  Station2 Fall - Station3 Fall       0.3214 0.163 41.6   1.972  0.7085
#>  Station2 Fall - Station4 Fall       0.3416 0.165 41.6   2.070  0.6454
#>  Station3 Fall - Station4 Fall       0.0201 0.163 41.6   0.124  1.0000
#>
#> P value adjustment: tukey method for comparing a family of 12 estimates
```

Station 1 Spring is not meaningfully different from Station 2 Spring, but it differs from all the other values.

Station 2 spring differs from Stations 3 and 4 Spring, but not other values.

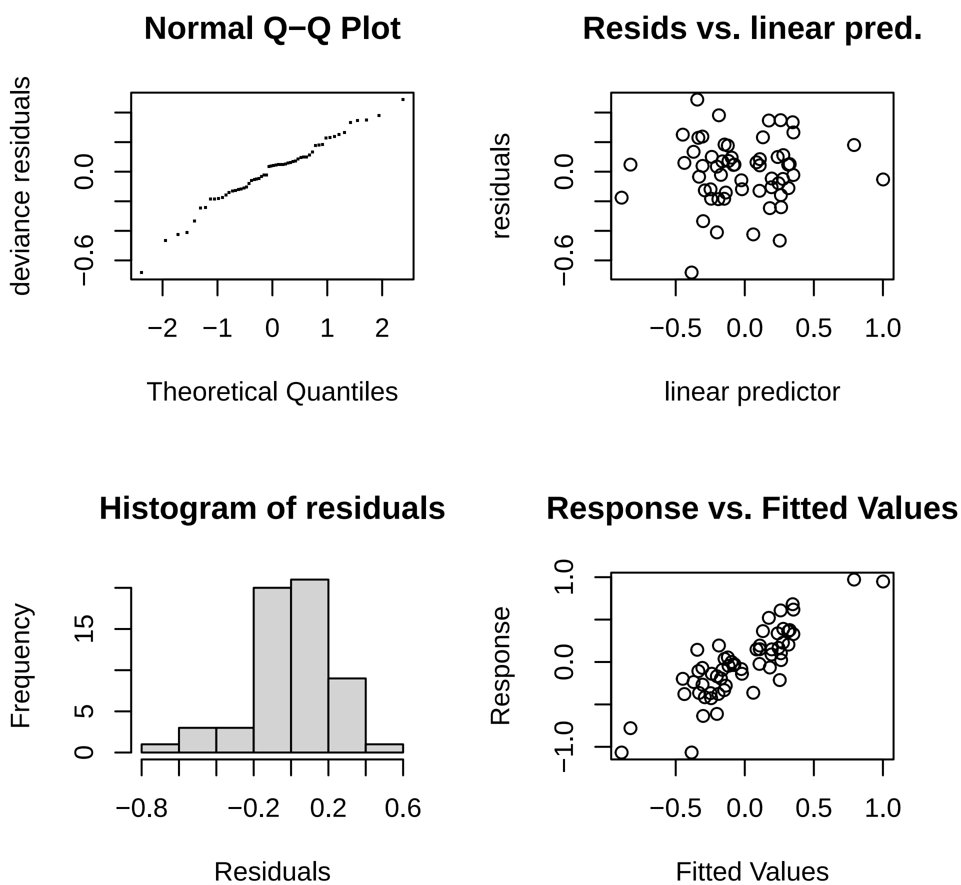
```
oldpar <- par(mfrow = c(2,3))
plot(gam_2$gam)
par(oldpar)
```



Axis 2 shows the effect of a couple of low salinity samples. Those Samples plot in NMDS space in the lower left, far from most Station 1 samples, so the GAM smoother fits a strong change in axis 2 scores to better predict those low salinity, early spring "washout" samples.

The other big story here is the connection between Axis 2 and early spring, upstream stations. Those samples have the highest values on Axis 2, meaningfully different from most other values.

```
oldpar <- par(mfrow = c(2,2))
gam.check(gam_2$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 3.47e-09    1.10    0.74
#> s(Sal)         9.00e+00 3.74e+00    1.07    0.66
#> s(log(Turb))   9.00e+00 4.21e-09    0.88    0.14
#> s(log(Chl))    9.00e+00 2.90e-09    1.03    0.52
#> s(log1p(Fish)) 9.00e+00 6.75e-01    0.98    0.42
par(oldpar)
```

Again, it looks like the low salinity samples have a disproportionate effect on model fit, but the models are otherwise well behaved. . . .

Axis 2 is associated with the differences between Stations 1 and 2 (high values) and Stations 3 and 4.