Multivariate Statistics: Introduction to Ordination

Tim D. Malinich, PhD
Timothy.Malinich@wildlife.ca.gov
Edition 2022

Introduction and Goals

This document will serve as a supplement to the R script (Ordination_Intro_Course.R) and presentation "Multivariate Statistics An Introduction to Ordination". My goals within this document are to provide a written instruction on the use of multivariate statistics, specifically ordination methods such as Principal Components Analysis (PCA), and to provide instruction on using the program R to conduct PCA. This document will assume that readers have a beginners grasp of working in R and have some background in basic statistics such as univariate statistic techniques (i.e. t-test, ANOVA, and Linear Regression).

To gain the most from this document you should also have:

- The Powerpoint presentation or recording of: Multivariate Statistics An Introduction to Ordination (IEP Workshop 2022)
- The R script document: Ordination_Intro_Course.R
- The csv file: Supplemental STN_flatfile.csv.
    - Note that this version of the STN flatfile has an additional column containing the Delta Smelt Life Cycle Model Regional assignments.

Setting up

For each project in R that I begin, I prefer to keep them all contained in their own separate folder. My script therefore begins with setting my working directory to this folder. Following this you will want to bring in the csv file containing the 2021 Summer Townet dataset. This dataset contains an additional column, the Delta Smelt Life Cycle Model (DSLCM regional assignments which has been added specifically for this exercise. The current Summer Townet Dataset can be found online at this California Department of Fish and Wildlife website. Also, at this weblink, you can find the relevant metadata documents describing each column (except the DSLCM regional assignments). As best practices, I highly recommend examining the metadata documents to understand the dataset. For a short introduction to the Summer Townet Survey, visit this website.

To check the dataset, I recommend using the View() function to open the dataset in a separate window. I also find it useful to use the summary() function with the original dataset, to see what types of values (i.e. discrete, continuous, factor, numeric etc) are contained in the dataset. In this dataset, you will find many types of values and note that some columns will contain blank values, read as NA by R, in addition to 0 values. Blank values in the Summer Townet dataset represent values that were not collected. For environmental variables it usually

means equipment failure or a time period when this metric was not measured, and for fish presence it could be a time period when the fish was either not counted by the survey or not yet present in the Estuary.

R Code:

```
setwd("C:/Users/tmalinich/Desktop/Ordination_Intro")#Connect to files

STN<-read.csv("CatchPerStation1959-2021.csv",fileEncoding="UTF-8-BOM")#STN
      #Dataset

View(STN)#See the dataset in all its glory.
names(STN)#useful when referencing columns
```

After we bring in the dataset, we'll want to prepare the packages we intend to use. The base R already includes several functions, including a principal components analysis that we'll use later. The remaining packages are not required, but are used in my example script. For instance, a 'must have' package for multivariate and ordination techniques is the vegan package. Ordination techniques often rely on plots for interpretation, and my plotting package of choice is ggplot2. Finally, dplyr and tidyr packages are useful packages for working with datasets and keeping a 'tidy' R script.

R Code:

```
###Many ordination techniques in 'vegan' package

library(vegan)        #Statistics
library(ggplot2)      #graphing
library(dplyr)        #tools for dataset work
library(tidyr)        #tools for dataset work
```

Finally, before we start using our ordination functions we will do two more steps to prepare our dataset. First, to make the analysis and interpretation easier, we will remove rows containing missing information (NAs). You will note that in this step I change the object name from STN to ds (dataset). This preserves my original dataset in case I need to refer to the original file. Next, we will limit our analysis to the last 10 years. If you wish to examine the whole dataset, simply skip this step.

R Code:

```
#--------------#Datasetprep techniques#-----------------------#

ds<-na.omit(STN)          #For this example we'll omit rows with NA's (blanks in the
                          #dataset)
```

```
ds<-ds[ds$Year>2011,]       #lets examine only the last 10 years.
```

There are many ordination techniques and below I include several for those interested in trying different analyses. Each function has a useful help page that contains a list of references for those in need of more information on different ordination techniques. You can also check out this book for different multivariate techniques and ordination methods in R:

> Borcard, D., F. Gillet, P. Legendre. 2018. Numerical Ecology with R. Springer. New York, NY. https://doi.org/10.1007/978-1-4419-7976-6.

I've included two functions for conducting principal components analysis (PCA), but will use prcomp() for my example. These functions are great if you have continuous variables that you wish to condense into fewer dimensions. They will scale and center your data, although if you have many zero's in your dataset they may not scale well. Out of all the ordination techniques, PCA is the easiest to pick up and is a great tool for data exploration. For this reason I've chosen to focus on this type of analysis for our introduction to multivariate analysis and ordination.

A commonly used ordination technique is non-metric multidimensional scaling and the R function from vegan to run this is MetaDMS(). If you have community data, this is a great technique for dimension reduction as you can specify the number of target dimensions *(k)*. However, the fewer dimensions, the more difficult it will be to summarize the variance in your dataset. Without going into depth, the nMDS uses a randomization technique to optimize new dimensions and grades them using a stress value. You should aim to have a stress values less than 0.2, with less being better. If your analysis is unable to reduce stress, then you could consider increasing your dimensions (k+1). One final note, because of the random starts used in this analysis, everytime you run the nMDS you may end up with axes that look different each time. You can use the set.seed() function to get the same result each time.

Correspondence Analysis is similar to PCA, but works on categorical values.

Finally, redundancy analysis is valuable if you have specific groupings that you want to use to distinguish variation in your response variables.

R Code:

```
#--------------#Ordination techniques#-----------------------#

#Principal Components Analysis (Base R, Stats packages)
#--#prcomp()
#--#princomp()

#Non-Metric Multidimensional Scaling (vegan)
```

```
#--#MetaDMS()

#Correspondence Analysis (vegan)
#--#cca()

#Redundancy Analysis (vegan)
#--#rda()
```

The Summer Townet Survey identifies larval and Juvenile fish to the lowest taxonomic level possible and collects a suite of environmental data at each station sampled. This environmental data includes water clarity (secchi in cm), turbidity (NTU), conductivity (microsiemens/cm), temperature (C), and it ranks in increasing severity (1-5) the weather, Microcystis presence and wave presence. Since many of these environmental variables correlate to one another, there is redundancy among them and that makes them prime candidates for ordination such as principal components analysis. This analysis will utilize that redundancy to summarize the variance in environmental information into fewer dimensions, called principal components.

R code:

```
#------------------------Demonstration--------------------------#

#Summer Townet Dataset

names(ds)#Here is the list of columns
```

In this analysis, I want to see if the Delta Smelt Life Cycle regional assignments can capture the different environmental conditions observed in stations across the Upper San Francisco Estuary. I could do this individually for each environmental variable using a linear model.

R code:

```
#-------------------------------------------------------------#
#------question and analysis-----------------------------------#
#-------------------------------------------------------------#


summary(lm(log(ds$Secchi)~ds$DSLCM.Region))#are there differences among my
    #regional variables?
```

But this would not help capture how a group of environmental variables change in each region. I could then try to plot each variable out, but I am quickly limited by how many different dimensions (i.e. environmental variables) I can plot and perceive at the same time.

Here I plot temperature and secchi together. Then I examine them with the regional assignments colored over the points. But beyond this I start to reach the limits of what we can perceive in a single graph. Maybe we could sneak another dimension in by going 3D, but we're starting to get too complex to reasonably interpret what is happening.

R Code:

```
plot<-ggplot(ds,aes(Temperature.Top,Secchi))+geom_point()
plot     #two metrics

plot<-ggplot(ds,aes(Temperature.Top,Secchi,color=DSLCM.Region))+geom_point()
plot     #two metrics, and region
```

Here we can begin to take advantage of multivariate statistics and dimension reduction. To set up our principal components analysis, we'll need to create a separate matrix for our analysis. I like to print out all my column names to make it easier to select columns for my new matrix. I use pipes (%>%) to simplify this process and highly recommend others to try it out when writing their own code. I select 8 environmental variables and save them as a new object ds_m (dataset matrix).

R Code:

```
names(ds)#nice to have the names printed out for selecting

ds_m<-ds%>%              # This line sets up a 'pipe' using %>%, meaning that everything
                         #following will come from the dataset ds.
  select(Temperature.Top, # This function pulls columns, by name, from the dataset 'ds'
         Secchi,
         Conductivity.Top,
         Depth.Bottom,
         Microcystis,
         Turbidity.Top,
         Weather,
         Waves)
```

The PCA function requires only our matrix. I recommend using the ?prcomp() to see what aspects of the analysis are already set in the function, to fully understand what will happen to your matrix. In this case, you would see that the function is set to automatically center and scale the data. The PCA needs to be saved as an object, I call mine 'pca_ds'.

To see the components of this pca, use the names() function. You will see that this object contains; sdev, rotation, center, scale, and x. These represent the standard deviations used to

calculate the eigenvalues, the rotation represents the eigenvectors, following this will be the values used to center and scale variables, and finally x contains the projection of our dataset for the new principal components.

R Code:

```
#-----------------------------------------------------------#
#-----------------------------------------------------------#
#--------------------The PCA--------------------------------#

pca_ds<-prcomp(ds_m,center=TRUE,scale.=TRUE)#run the PCA

names(pca_ds)#check out the components of pca_ds
```

The summary of our pca object provides a good table showing the variance represented by each principal component (PC) as well as the cumulative variance as you progress through each component. These values should be included whenever you plot the new PCs. To make plotting easier, save the projected PCs (PCscores) and the eigenvectors (rotation) as new objects. I found it also useful to save the rownames from the rotation to help with labeling in our graphs.

R Code:

```
summary(pca_ds)#examine the summary of Principal components
PCscores<-as.data.frame(pca_ds$x)#scores, save them here for plotting
rotation<-as.data.frame(pca_ds$rotation) #is the relationship between the original
values and PC scores
vectors<-rownames(rotation)#save here for printing in plot later
```

Before plotting out all my points, lets first demonstrate how our new principal components work. Each principal component is considered orthogonal, independent, to all other principal components. This means we can plot our PCs at right angles to one another (i.e. an x and y-axis plot). Figure 1 shows us just over half of our total variation at 52%. The vectors moving positively toward the right along the x-axis are primarily Temperature, Microcystis, Secchi and Depth. Towards the left x-axis, stations moving in this direction are strongly influenced by increased conductivity, Turbidity, and more severe waves. The second PC shares some of these same variables, but is notably impacted by weather, moving positively up the y-axis. Already we can start to make predictions on where we might see some of our DSLCM regions, such as the San Pablo Bay region more towards the left-hand side of figure 1.

As you plot your own figures and alter which components, make sure you update the PC label for all aspects of the ggplot2 code (in ggplot(), geom_segment(), geom_text and xlab(). Note, you will see that I multiply the rotation values in geom_segment(). This is to provide greater

magnitude to discern differences among the different vectors, otherwise they would all graph on top of one another.

R Code:

```
#-------------------------------------------------------------------------#
#----------plotting principal components-------------------------------------#

figure1<-ggplot(data=PCscores,aes(PC1,PC2))+
  geom_segment(data=rotation,aes(x = 0,xend=PC1*10, y =0,yend= PC2*10),
               arrow = arrow(),size=1)+
  geom_text(data=rotation,aes(PC1*10+1,PC2*10+1),label=vectors,size=4)+

  xlab("PC 1 (37%)")+ylab("PC 2 (15%)")+

  theme_bw()+
  theme(axis.ticks=element_line(color="black"),
        axis.title.x=element_text(size=16, color="Black"),
        axis.title.y=element_text(size=16, color="Black"),
        axis.text.x=element_text(size=14, color = "Black",angle=90),
        axis.text.y=element_text(size=14, color="Black"))
```

figure1

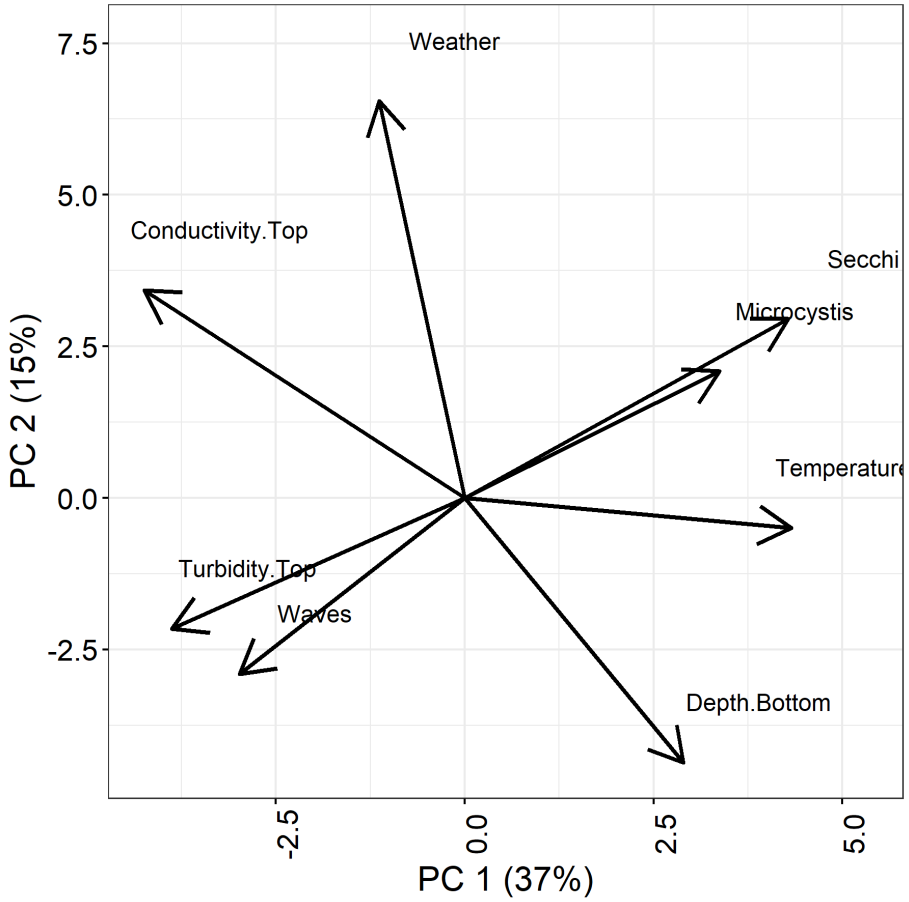

Figure 1. Principal components 1 and 2.

If we repeat this process, but with a different principal component (PC3) we will see slightly different patterns and interpretations of stations across the axes. In Figure 2, we have moved the patterns of PC2 to the x-axis. Along PC3, we see that stations found higher on the y-axis will have greater turbidity, and temperature. While moving down the y-axis stations would have more severe weather, waves and greater depth.

R Code:

```
############different axes

figure2<-ggplot(data=PCscores,aes(PC2,PC3))+
  geom_segment(data=rotation,aes(x = 0,xend=PC2*10, y =0,yend= PC3*10),
               arrow = arrow(),size=1)+
  geom_text(data=rotation,aes(PC2*10+1,PC3*10+1),label=vectors,size=4)+
  xlab("PC 2 (15%)")+ylab("PC 3 (11%)")+
```

```
  theme_bw()+
  theme(axis.ticks=element_line(color="black"),axis.title.x=element_text(size=16,
color="Black"), axis.title.y=element_text(size=16,
color="Black"),axis.text.x=element_text(size=14, color = "Black",angle=90),
axis.text.y=element_text(size=14, color="Black"))
figure2
```

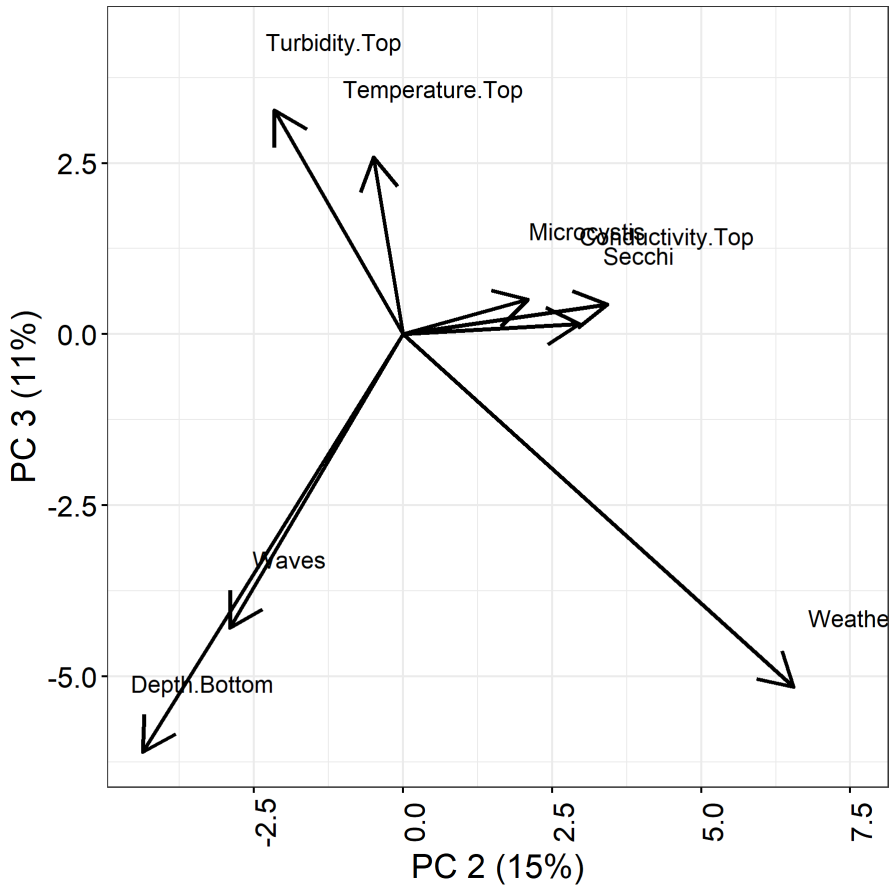

Figure 2. PC2 and PC3 plotted.

Now that we have a better understanding of how our vectors help interpret our principal components, lets add in our points representing different stations. I use geom_point() to add in the new projected points, which we saved as PCscores earlier. I adjust the alpha code in this function to make it easier to see overlapping points.

R Code:

```
figure3<-ggplot()+
  geom_point(data=PCscores,aes(PC1,PC2),alpha=.3)+
  geom_segment(data=rotation,aes(x = 0,xend=PC1*10, y =0,yend= PC2*10),
               arrow = arrow(),size=1.0)+
```

```
  geom_text(data=rotation,aes(PC1*10+1,PC2*10+1),label=vectors,size=4)+
  xlab("PC 1 (37%)")+ylab("PC 2 (15%)")+
  theme_bw()+
  theme(axis.ticks=element_line(color="black"),
        axis.title.x=element_text(size=16, color="Black"),
        axis.title.y=element_text(size=16, color="Black"),
        axis.text.x=element_text(size=14, color = "Black",angle=90),
        axis.text.y=element_text(size=14, color="Black"))
figure3
```

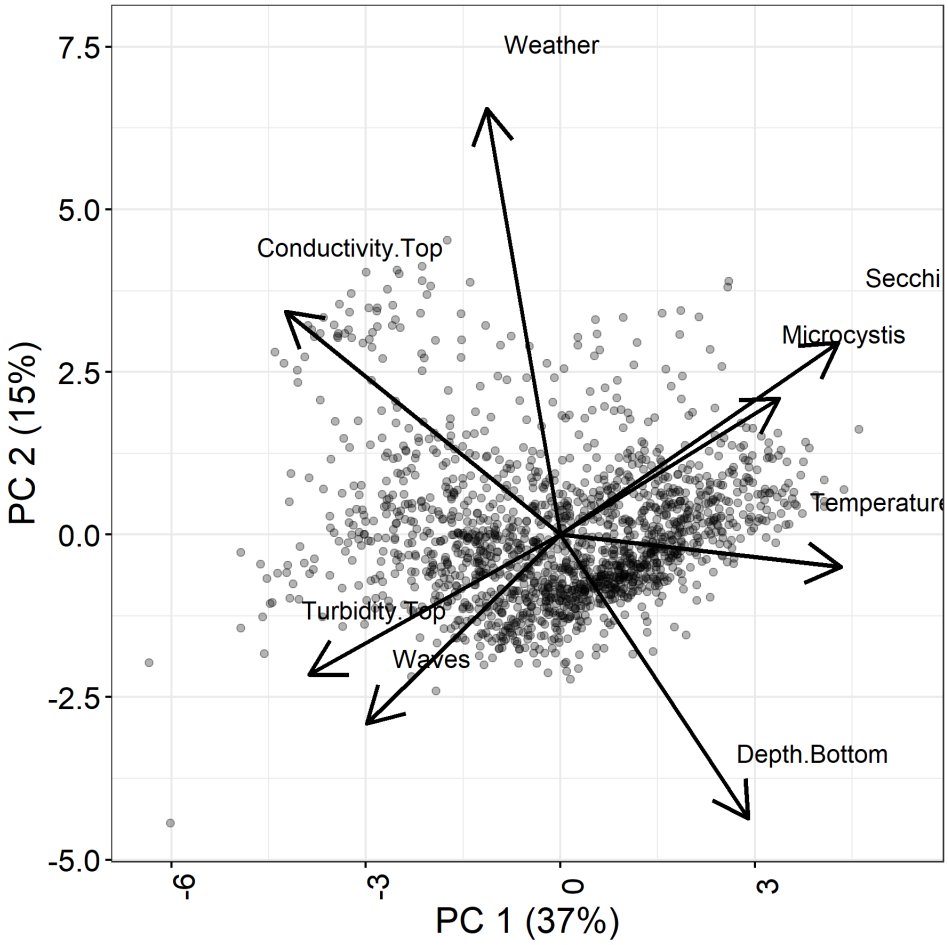

Figure 3. PC1 and PC2 with projected points.

In Figure 3 we can see how points are spread further across PC1 compared to PC2. There are outlying points that may be driving differences in some of our axes and we can use labeling to identify these points in the future. In severe cases you may find some PCs completely influenced by one or two points. In such cases you might consider dropping these outliers and conducting the PCA again. In our next figure, we'll add in the regional coloring.

R Code:

```
figure4<-ggplot()+
  geom_point(data=PCscores,aes(PC1,PC2,color=ds$DSLCM.Region),alpha=.2)+
  geom_segment(data=rotation,aes(x = 0,xend=PC1*10, y =0,yend= PC2*10),
               arrow = arrow(),size=1.0)+
  geom_text(data=rotation,aes(PC1*10+1,PC2*10+1),label=vectors,size=4)+
  xlab("PC 1 (37%)")+ylab("PC 2 (15%)")+
  theme_bw()+
  theme(axis.ticks=element_line(color="black"),
        axis.title.x=element_text(size=16, color="Black"),
        axis.title.y=element_text(size=16, color="Black"),
        axis.text.x=element_text(size=14, color = "Black",angle=90),
        axis.text.y=element_text(size=14, color="Black"))
figure4
```

We can see in Figure 4 that the majority of group separation appears to occur along PC1. San Pablo Bay and the South Delta are driven the furthest apart by this component, but the North and West Delta remain greatly overlapping. PC2 does little to drive our regions apart so we will examine additional components.

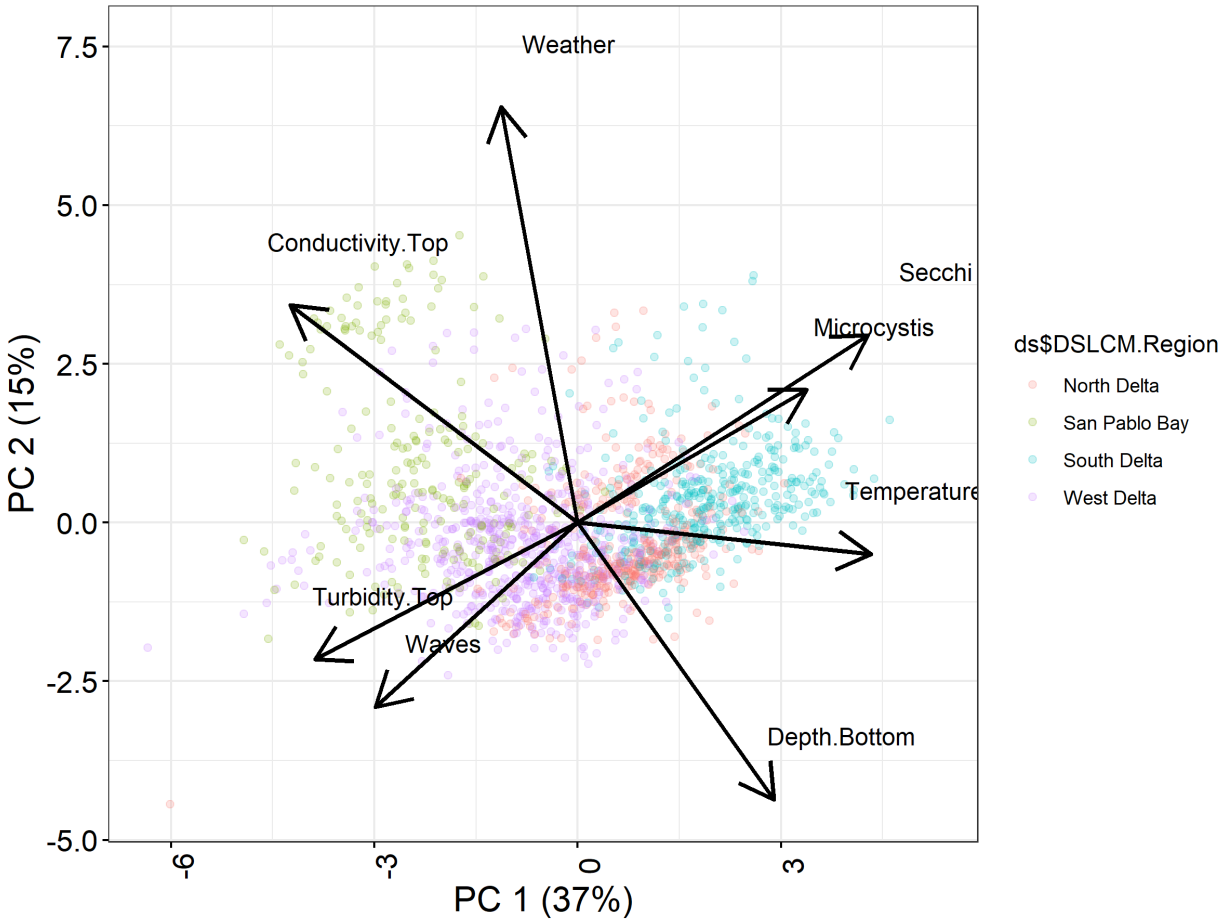

Figure 4. PC1 and PC2 with projected points and colored by DSLCM regional assignments.

R Code:

```
##################use different Principal components

figure5<-ggplot()+
  geom_point(data=PCscores,aes(PC1,PC5,color=ds$DSLCM.Region),alpha=.2)+
  geom_segment(data=rotation,aes(x = 0,xend=PC1*10, y =0,yend= PC5*10),
               arrow = arrow(),size=1.0)+
  geom_text(data=rotation,aes(PC1*10+1,PC5*10+1),label=vectors,size=4)+
  xlab("PC 1 (37%)")+ylab("PC 5 (9%)")+
  theme_bw()+
  theme(axis.ticks=element_line(color="black"),
        axis.title.x=element_text(size=16, color="Black"),
        axis.title.y=element_text(size=16, color="Black"),
        axis.text.x=element_text(size=14, color = "Black",angle=90),
        axis.text.y=element_text(size=14, color="Black"))
```

figure5

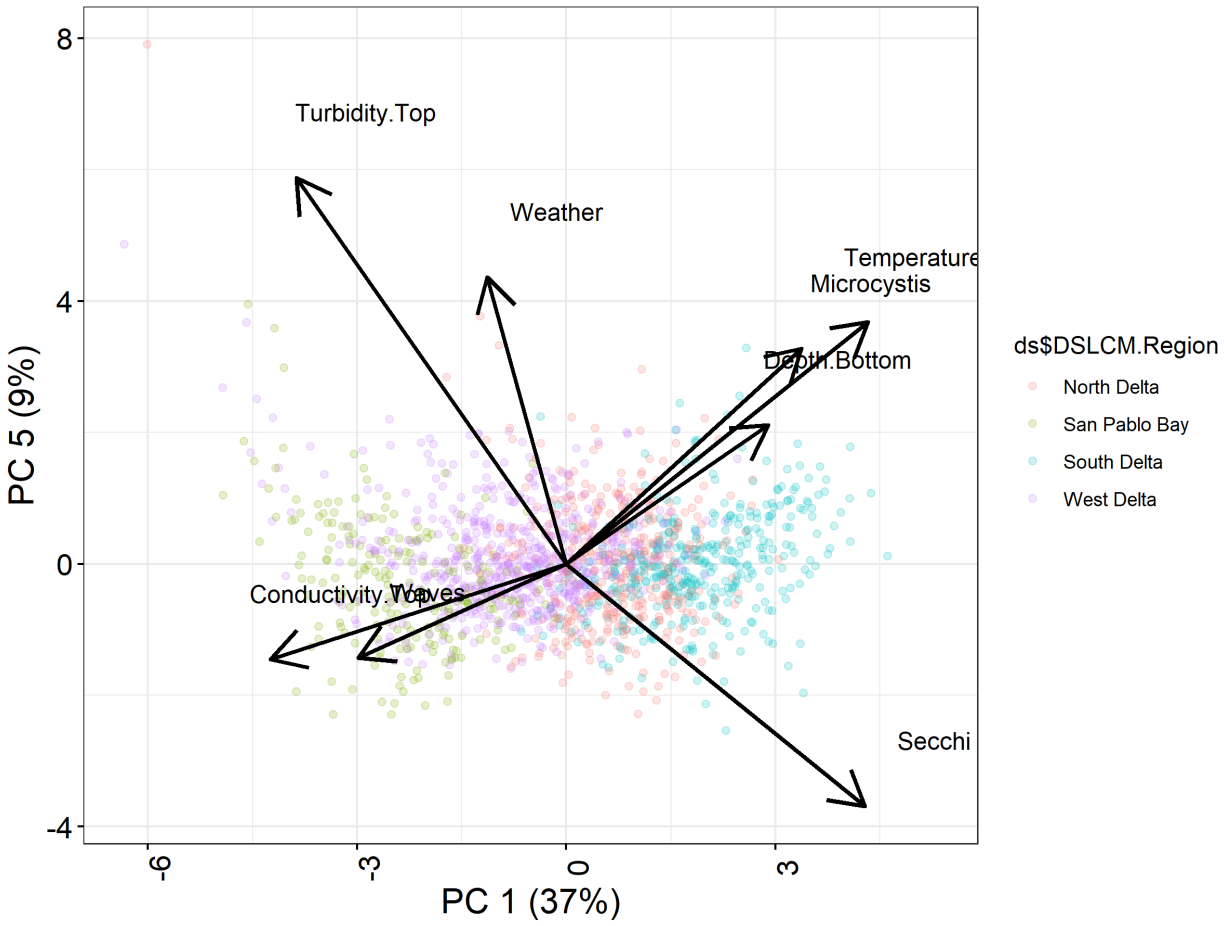

Figure 5. PC1 and PC5, with projected points and regional assignments.

Figure 5, like Figure 4 shows little separation among groups when we examine principal components other than PC1. Since ordination is often interpreted visually, you will need to continue comparing different PC combinations. You should consider your goals while examining components. Are you aiming to capture the most variance? In that case retain the principal components that represent the most variation. If your goal is to find the best group separation, then select components that best drive differences, possibly losing more of the total variation in the process.

R Code:

```
##############continue examining different components

figure6<-ggplot()+
  geom_point(data=PCscores,aes(PC1,PC8,color=ds$DSLCM.Region))+
```

```
  geom_segment(data=rotation,aes(x = 0,xend=PC1*10, y =0,yend= PC8*10),
               arrow = arrow(),size=1.5)+
  geom_text(data=rotation,aes(PC1*10+1,PC8*10+1),label=vectors,size=4)+
  xlab("PC 1 (37%)")+ylab("PC 8 (5%)")+
  theme_bw()+
  theme(axis.ticks=element_line(color="black"),
        axis.title.x=element_text(size=16, color="Black"),
        axis.title.y=element_text(size=16, color="Black"),
        axis.text.x=element_text(size=14, color = "Black",angle=90),
        axis.text.y=element_text(size=14, color="Black"))
figure6
```

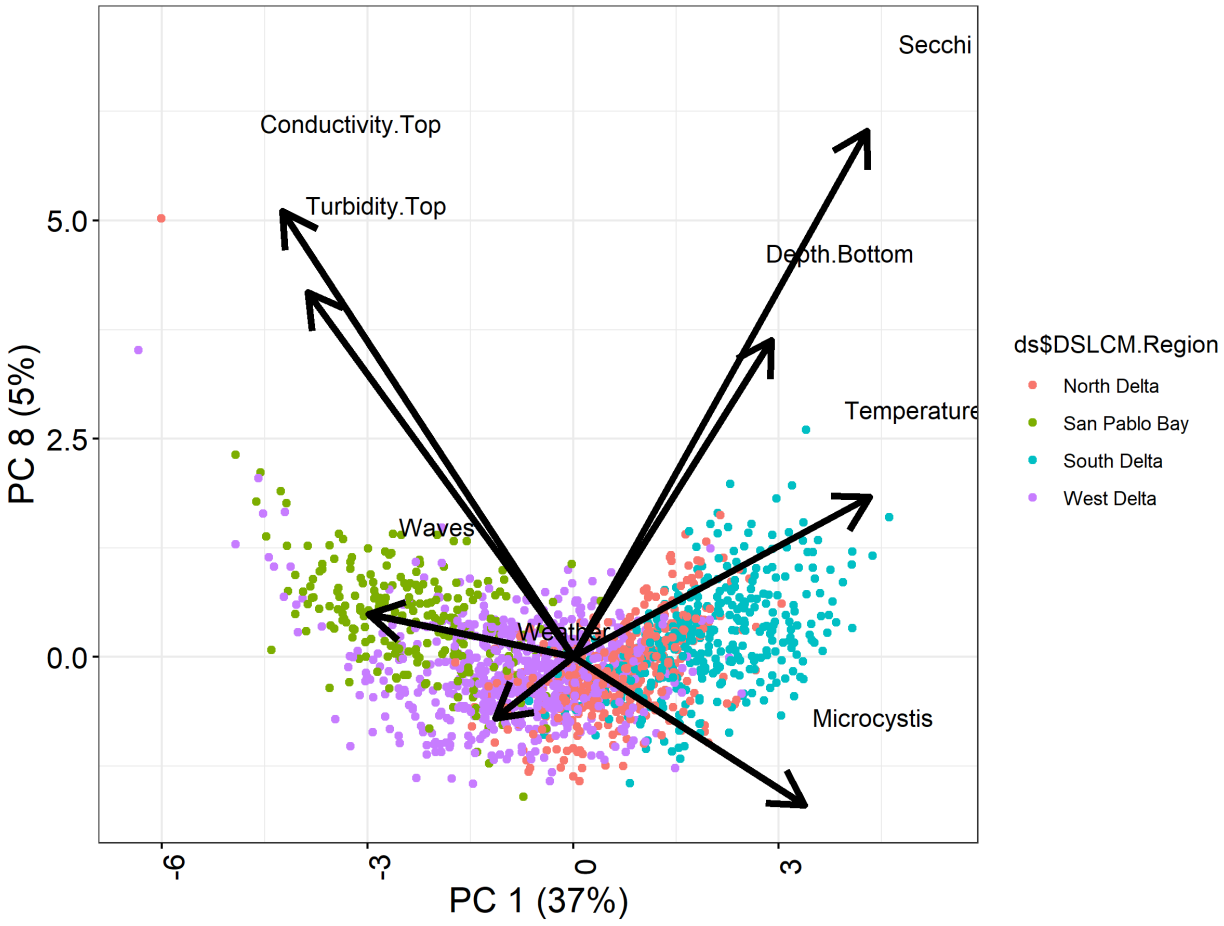

Figure 6. PC 1 and 8 with projected points and regional assignments.

In figure 6, we see our final principal component with the least variation. Some of our vectors here will have challenging interpretations, and we may consider not using this combination. We see both Secchi and Turbidity moving positively up PC8. These environmental variables should

be related inversely to one another (i.e. greater water clarity should indicate lower turbidity). Therefore, I want to caution users to consider the interpretation of their vectors, particularly with PCs containing low amounts of variation. Linear combinations of our environmental variables may not always preserve their true relationships.

Following up your analysis

Where to go from here? To help determine the separation of your regions, you can plot ellipses to highlight the majority of your points (Figure 7).

R Code:

```
###############use ellipses to distinguish regions

figure7<-ggplot()+
  geom_point(data=PCscores,aes(PC1,PC5,color=ds$DSLCM.Region),size=1)+
  stat_ellipse(data=PCscores,aes(PC1,PC5,color=ds$DSLCM.Region),size=1.5)+#ellipse
plotted based on multivariate t distribution
  geom_segment(data=rotation,aes(x = 0,xend=PC1*10, y =0,yend= PC5*10),
               arrow = arrow(),size=1.0)+
  geom_text(data=rotation,aes(PC1*10+1,PC5*10+1),label=vectors,size=4)+
  scale_color_discrete(name="")+
  xlab("PC 1 (37%)")+ylab("PC 5 (9%)")+
  theme_bw()+
  theme(legend.position="bottom")+
  theme(axis.ticks=element_line(color="black"),
        axis.title.x=element_text(size=16, color="Black"),
        axis.title.y=element_text(size=16, color="Black"),
        axis.text.x=element_text(size=14, color = "Black",angle=90),
        axis.text.y=element_text(size=14, color="Black"))
figure7
```

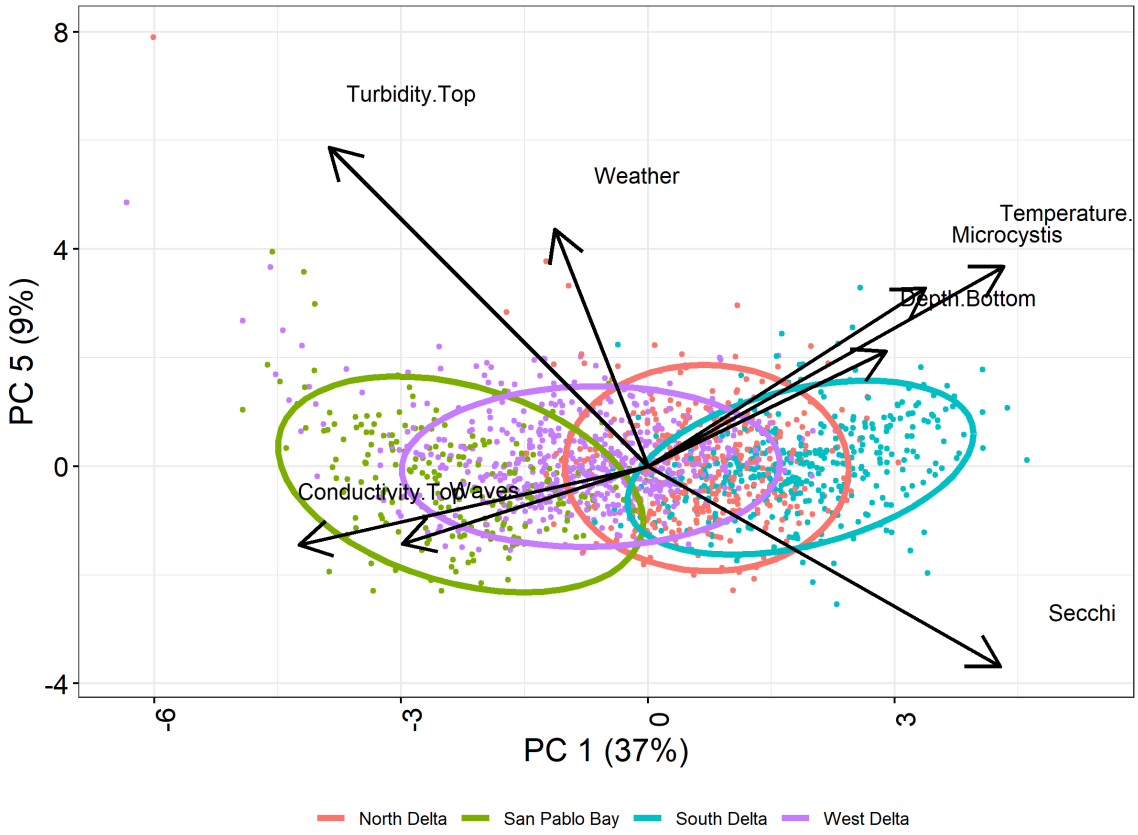

Figure 7. Ellipses showing regional assignments, plotted on PC1 and 5.

There are a variety of other packages and choices when it comes to examining your PCA results. I include the package ggfortify below as one example.

R Code:

```
#######################################
############Plenty of other R packages that can help with plotting your PCA results
library(ggfortify)

autoplot(pca_ds,loadings=TRUE,loadings.label=TRUE)

plot(pca_ds$x)#base R package
```

We began this analysis wanting to examine how well our groups distinguished different environmental conditions. If you wish to use a more analytical approach, as opposed to our visual approach, to examine regions, you could consider a variety of analyses, such as k-means to look at clusters. Or more simply you could test a singular PC with an ANOVA, or multiple PCs using a MANOVA or perMANOVA. Of course, if your goal is to separate groups I might suggest redundancy analysis (RDA) or discriminant function analysis (DFA) instead of PCA.

R Code:

```
#######################################Do I have good groups?

summary(lm(PCscores$PC1~ds$DSLCM.Region))
summary(manova(as.matrix(PCscores[,1:3])~ds$DSLCM.Region))

#to identify better regions, I suggest RDA DFA.
```

Finally, one of the great strengths of PCA, and other ordination techniques, is dimension reduction. We can use the principal components in further analysis. For example, we may be interested in Striped Bass catch as it relates to Tripletooth goby catch. As a part of this analysis, we will want to include environmental variables that also have strong impacts on fish catch. But including all of our environmental variables could lead to overfitting. Particularly if we had a small dataset. I attempt to demonstrate this below. First, a model looking at the Striped Bass catch response to our 8 environmental variables, in addition to the goby catch. We can use our PCs to include the variation due to environmental variables but with 1 or two dimensions, rather than 8. We will lose some information from our dimension reduction, but we will retain most of the variation in favor of fewer variables. Further, some of our independent variables may be correlated with one another and may violate some assumptions in our model. But our principal components are orthogonal, so multiple PCs may be included within the same model.

R Code:

```
####################Can use principle components in further analyses
summary(lm(Age.0.Striped.Bass~Conductivity.Top+Turbidity.Top+Waves+Weather+Micr
ocystis+Temperature.Top+Depth.Bottom+Secchi+Tridentiger.spp,data=ds))
#R2 ok, but due in part to the increased number of variables and not better fit.

summary(lm(ds$Age.0.Striped.Bass~PCscores$PC1+ds$Tridentiger.spp))#Account for
environmental metrics, but examine the relationship between invasive gobies and SB
#note the low R2 value

summary(lm(ds$Longfin.Smelt~PCscores$PC1+ds$Tridentiger.spp))#Account for
environmental metrics, but examine the relationship between invasive gobies and
longfin
#note the low R2 value
```

Thank you for working your way through this document. If you have additional questions feel free to contact me at timothy.malinich@wildlife.ca.gov.