An Introduction to R for the Geosciences: Ordination I

Gavin Simpson

April 29, 2013

Summary

This practical will use the PONDS dataset to demonstrate methods of indirect gradient analysis (PCA, CA, and DCA) of species and environmental data. The file pondsenv.csv contains the species data (48 taxa) and pondsenv.csv contains the transformed environmental variables (15 variables) for 30 sites. You will use R and the `vegan` package to analyse these data using a variety of indirect ordination graphical display techniques.

1 Principal Components Analysis

Principal Components Analysis (PCA) is a common statistical methods and is implemented in R by two functions in the standard installation (`prcomp()` and `princomp()`), as well as in numerous guises as part of other add-on packages. In this practical you will use the implimentation provided in the `vegan` package by Jari Oksanen. This is because you will be using `vegan` for the direct gradient analysis class and the `vegan` package provides a rich set of analysis and graphical tools that can be applied to a variety of ordination techniques. As such, you only have to learn a single set of commands to run a wide variety of analyses.

Start R and load the `vegan` package for use. Read in the two data sets and the associated bstick function file:

```
> library(vegan)
> pondsenv <- read.csv("pondsenv.csv")
> pondsdiat <- read.csv("ponddiat.csv")
> bstick <- function(n, tot.var = 1) rev(cumsum(tot.var/n:1)/n)
```

PCA is fitted using the `rda()` function, which was designed to implement Redundancy Analysis (RDA). RDA is the constrained form of PCA so running `rda()` without any constraints yields PCA. Run a PCA of the Ponds environmental data using `rda()` and display the results. The `scale` argument to `rda()` scales the variables to zero mean and unit standard deviation. This results in a PCA on a correlation rather than covariance matrix. This is appropriate in situations where the variables are measured in different units, as is the case with the hydrochemical data analysed here. It may also be appropriate in situations where species abundance (counts etc.) are being analysed and you wish to focus on explaining variation in all species not just the most abundant ones.

```
> pondspca <- rda(pondsenv, scale = TRUE)
> pondspca

Call: rda(X = pondsenv, scale = TRUE)

              Inertia Rank
Total              15
Unconstrained      15   15
Inertia is correlations

Eigenvalues for unconstrained axes:
 PC1  PC2  PC3  PC4  PC5  PC6  PC7  PC8  PC9 PC10 PC11 PC12 PC13
4.96 3.09 2.33 1.43 0.77 0.63 0.53 0.40 0.31 0.29 0.13 0.07 0.03
PC14 PC15
0.02 0.01
```

The ouput shows the results of the PCA, displaying the eigenvalues for each axis. Note that these values differ from the ones report by CANOCO, which scales the total variance (called inertia in `vegan`) to 1, `rda()` does not. To achieve a comparable set of eigenvalues simply divide each eigenvalue (stored within `pondspca$CA$eig`) by the total inertia (`pondspca$tot.chi`). This conveniently gives the proportions of the inertia (variance) explained by each axis.

```
> pondspca$CA$eig / pondspca$tot.chi

     PC1      PC2      PC3      PC4      PC5      PC6      PC7
0.330929 0.206249 0.155262 0.095185 0.051454 0.041739 0.035581
     PC8      PC9     PC10     PC11     PC12     PC13     PC14
0.026529 0.020602 0.019523 0.008518 0.004623 0.001911 0.001244
    PC15
0.000653
```

Q and A

1. What are the values of λ_1 and λ_1, the eigenvalues for axes one and two?
2. How much of the total variance in the environmental data is explained by axes one and two individually and cummulatively?

For a longer print out of the results of the PCA and to display the species and site scores a `summary` method is available. This truncates the output to the first six axes but this can be changed using the `axes` argument.

```
> summary(pondspca, scaling = 2)

Call:
rda(X = pondsenv, scale = TRUE)

Partitioning of correlations:
              Inertia Proportion
Total              15          1
Unconstrained      15          1

Eigenvalues, and their contribution to the correlations

Importance of components:
                        PC1    PC2    PC3     PC4     PC5     PC6
....
Eigenvalue            0.5337 0.3979 0.3090 0.2928 0.12777
Proportion Explained  0.0356 0.0265 0.0206 0.0195 0.00852
Cumulative Proportion 0.9164 0.9429 0.9635 0.9830 0.99157
                         PC12    PC13    PC14    PC15
Eigenvalue            0.06934 0.02866 0.01865 0.00979
Proportion Explained  0.00462 0.00191 0.00124 0.00065
Cumulative Proportion 0.99619 0.99810 0.99935 1.00000

Scaling 2 for species and site scores
* Species are scaled proportional to eigenvalues
....
Mg          -0.9397 -0.4269 -0.0276 -0.346 -0.06561 -0.251229
```

```
Ca           -0.6044  0.4924  0.6841 -0.204  0.14851 -0.000079
Cl           -0.8681 -0.5860 -0.3273  0.129 -0.14424  0.214247
SO4          -0.9684 -0.0729  0.3048 -0.185  0.06011 -0.163432
Chla         -0.3783  0.7311 -0.5759  0.239  0.19308  0.098566
Secchi        0.2349 -0.1047  0.8568  0.175 -0.38849 -0.474071
....
```

The use of the `scaling` argument is redundant in this case as the default is `scaling = 2` but it is included here to illustrate a common argument that can be specified by the user. `scaling = 2` relates to the option "inter-species correlations" in CANOCO. `rda()` does not allow the option to post transform the species scores (by dividing by the standard deviation) but this is easy to achieve in R.

```
> apply(scores(pondspca, choices = 1:6, display = "species"), 2,
+       function(x) x / sd(x))
```

```
                 PC1      PC2      PC3      PC4      PC5
pH           -0.4664  1.39775  1.34732  0.75636  0.79131
Conductivity -2.3832  0.07391  0.45100 -0.62598 -0.36653
Alkalinity   -1.5226  1.42407  0.87208 -0.29398 -0.02629
TP           -0.4541  1.47323 -1.04192  1.41896 -0.38174
....
```

1.1 The meaningful components

To assess the likely statistical significance of the axes, it is useful to both plot a scree plot and to compare the sizes of the actual PCA axes with the sizes expected under a random (null) model, such as the broken stick distribution. A scree plot of the results of a PCA can be produced using the `plot()` method (Figure 1).

```
> plot(pondspca$CA$eig, type = "o", col = "red",
+      xlab = "PCA Axis", ylab = "Variance",
+      main = "Scree plot for the PCA of the Ponds data")
```

Q and A

1. How many axes does the scree plot suggest are significant?

The broken-stick distribution is simple to calculate and the expected values of the *pieces* of the broken stick are given by Equation 1:

$$E_j = \frac{1}{n}\sum_{x=j}^{n}\frac{1}{x} \tag{1}$$

where E_j is the expected value of the j^{th} piece, and n is the number of pieces (axes in this case). Our `bstick()` function implements the broken stick distrbution. `bstick()` takes two arguments, the number of pieces (`n`) and the total variance (`tot.var`, defaults to 1). As `rda()` does not scale the eigenvalues to 1, which would give a total variance of 1, `tot.var` should be set at 15.

```
> bstick.env <- bstick(15, tot.var = 15)
> bstick.env
```

```
 [1] 3.31823 2.31823 1.81823 1.48490 1.23490 1.03490 0.86823
 [8] 0.72537 0.60037 0.48926 0.38926 0.29835 0.21502 0.13810
[15] 0.06667
```

```
> pondspca
```

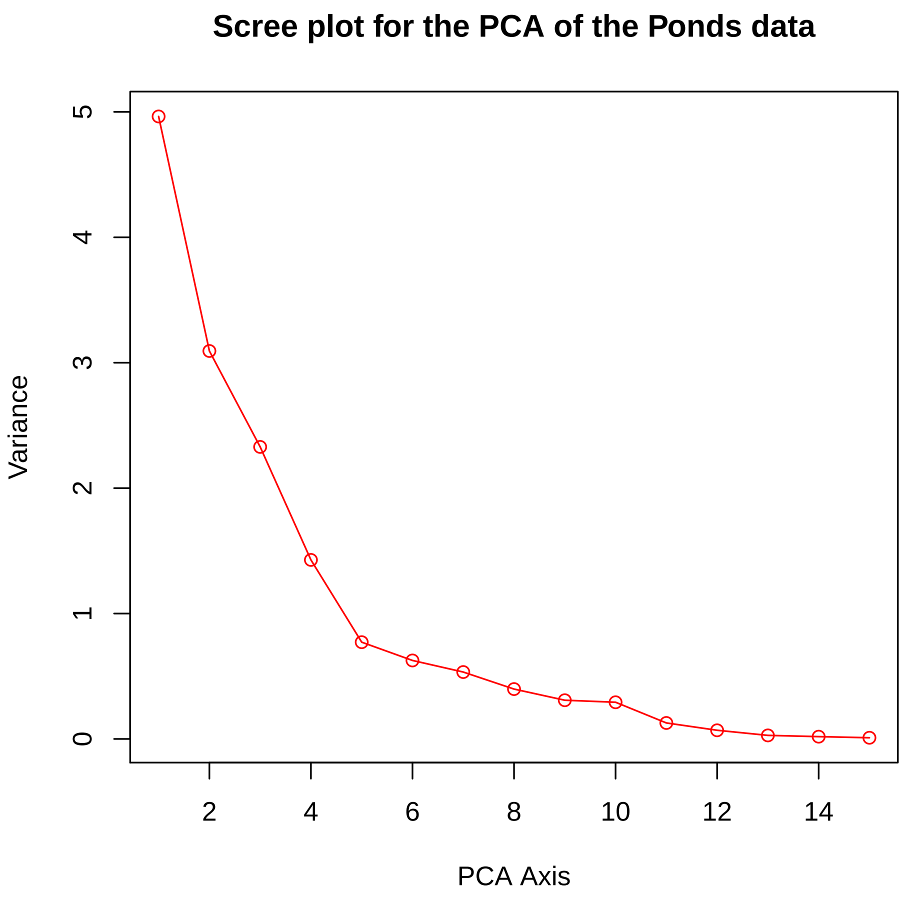

Figure 1: Scree plot of the eigenvalues obtained from a PCA of the Ponds hydrochemistry data.

```
Call: rda(X = pondsenv, scale = TRUE)

              Inertia Rank
Total              15
Unconstrained      15   15
Inertia is correlations

Eigenvalues for unconstrained axes:
 PC1  PC2  PC3  PC4  PC5  PC6  PC7  PC8  PC9 PC10 PC11 PC12 PC13
4.96 3.09 2.33 1.43 0.77 0.63 0.53 0.40 0.31 0.29 0.13 0.07 0.03
PC14 PC15
0.02 0.01
```

Compare the variances of the pieces expected under the broken-stick model with the eigenvalues obtained from the reaults of the PCA of the Ponds environmental data. The broken-stick distribution represents the null model, so significant axes are those that have an eigenvalue that exceeds the expected value for the j^{th} piece of the broken-stick distribution.

Q and A

1. How many axes does the broken-stick distribution suggest are significant?

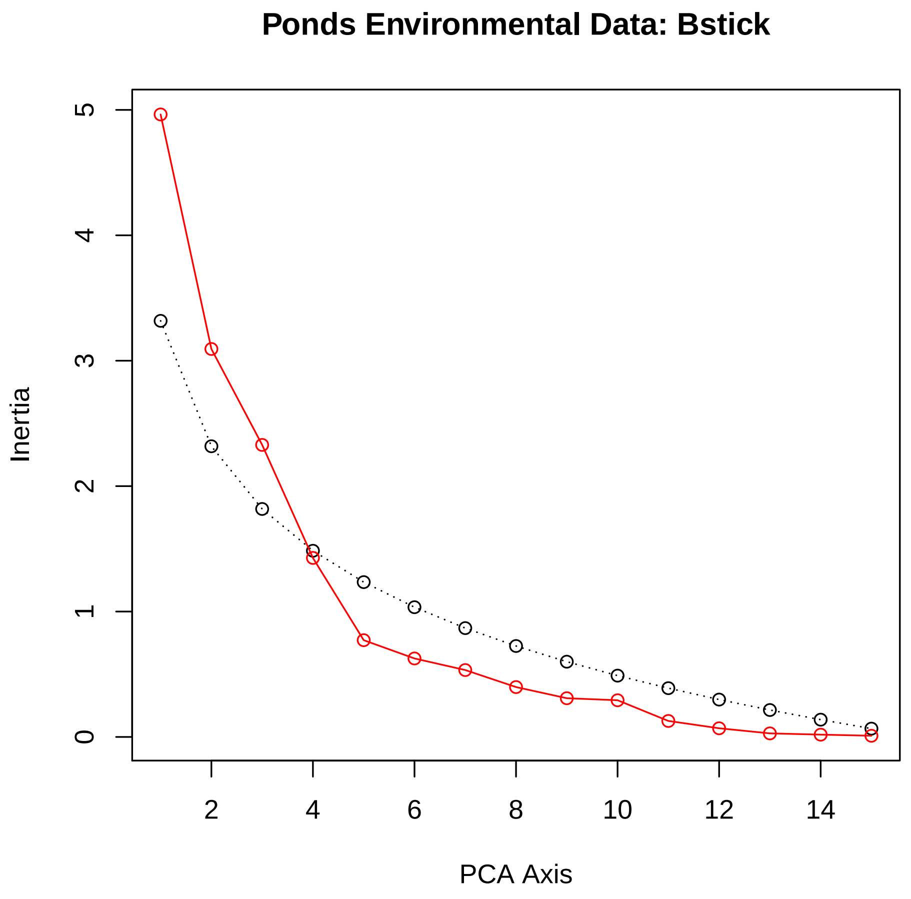

Figure 2: Scree plot of the eigenvalues obtained from a PCA of the Ponds hydrochemistry data. The solid line represents the observed eigenvalues. The dotted line indicates the expected values under a null model obtained from the broken-stick distribution.

A simple way to visualise which axes have eigenvalues that are greater than those expected under the null model is to overlay the values for the broken-stick distribution on to a scree plot of the PCA eigenvalues (Figure 2).

```
> plot(bstick.env, type = "o", lty = "dotted",
+      ylim = range(bstick.env, pondspca$CA$eig),
+      xlab = "PCA Axis", ylab = "Inertia",
+      main = "Ponds Environmental Data: Bstick")
> points(pondspca$CA$eig, type = "o", col = "red")
```

1.2 PCA Biplots

Two types of biplot may be used to visualise the results of a PCA; a distance biplot and a correlation biplot. In this analysis of the Ponds environmental data, the focus is on the correlations between the environmental variables ("species" in common parlance, even though environmental data are being analysed!), therefore, a correlation biplot is appropriate. The type of biplot produced is determined by the scaling applied to one or both of the species and site scores. For a correlation biplot, scaling 2 is used, in which the scores for the k^{th} species (the k^{th} eigenvector) are scaled to length $\sqrt{\lambda_k}$[1], whilst the sites are

[1]The scores for the k^{th} are then proportional to the square root of the k^{th} eigenvalue.

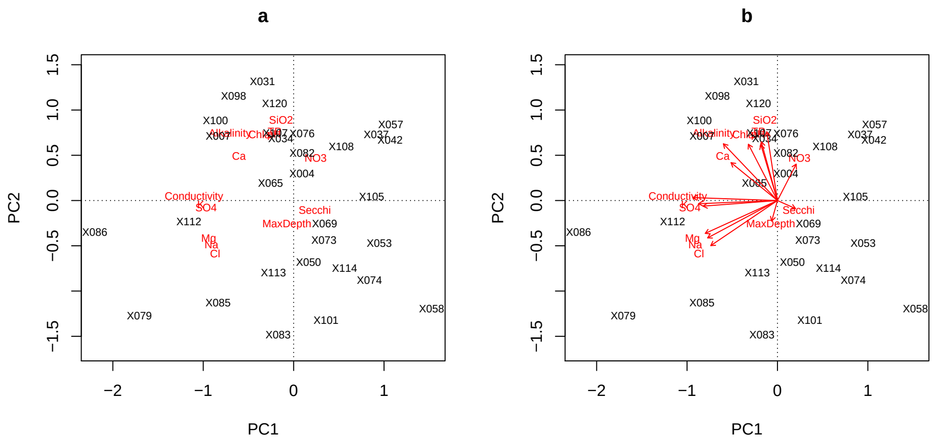
Figure 3: PCA correlation biplot of the Ponds environmental data, with species represented by points (a) or with biplot arrows (b)

left unscaled. Biplots are easily obtained using the `plot()` method of `rda()`.

```
> plot(pondspca, scaling = 2)
```

Species are traditionally represented by biplot arrows

```
> env.sc <- scores(pondspca)$species
> plot(pondspca, scaling = 2)
> arrows(0, 0, env.sc[,1] * 0.85, env.sc[,2] * 0.85,
+        col = "red", length = 0.05)
```

Q and A

1. What are the main chemical gradients represented by axes one and two?
2. Are there any outliers sites on axes one or two?

Interpretting ordination diagrams can be improved by enhancing the plot with additional information, commonly in the form of response surfaces. Function `ordisurf()` generates response surfaces using a generalised additive model (GAM) to predict a response variable for the ordination configuration (Figure 4). The irregular configuration is then interpolated to a regular grid using linear interpolation routines.

```
> plot(pondspca, scaling = 2, display = "sites")
> ordisurf(pondspca, pondsenv$TP, main = "temp", add = TRUE)
```

Firstly, the biplot is redisplayed with the species suppressed (`display = "sites"`), then a response surface for the variables TP (Total Phosphorus) produced using `ordisurf()`. The argument `add = TRUE)` is used to add the response surface to the existing plot without clearing the graphics device first.

Q and A

1. Make response surface plots for selected variables.

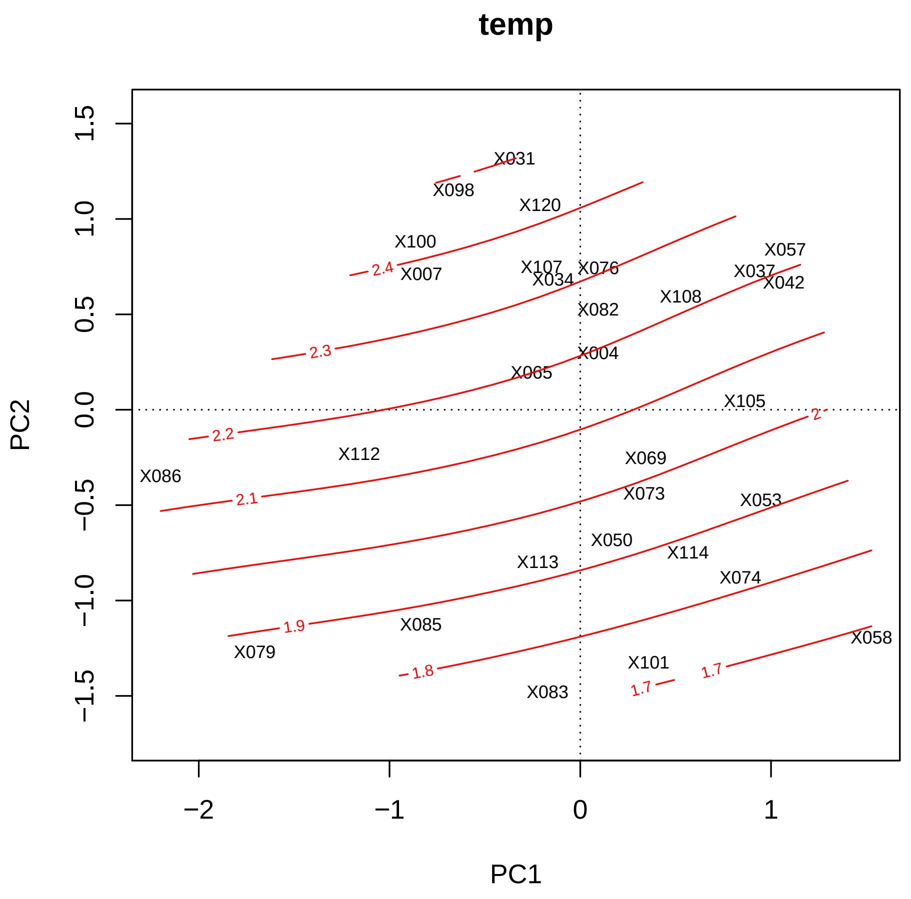

Figure 4: PCA correlation biplot of the Ponds environmental data overlayed with a GAM response surface of the TP values recorded at each pond.

2 Correspondence Analysis

Species often show unimodal responses to environmental gradients. PCA assumes a linear response model and as such may not be best suited to the analysis of species data exhibiting unimodal responses as PCA is unlikely to fit the data adequately. Correspondence Analysis (CA) is an indirect ordination technique that assumes an *idealised* unimodal response in species. The response is idealised in that it assumes that species responses are symmetrical, of equal height and width and are equally spaced.

A CA is performed using function `cca()`, also in package `vegan`. CA can produced unstable results when there are rare species or odd samples in the data set, therefore, rare species tend to be downweighted. Function `downweight()` provides this capability, replicating the behaviour of CANOCO.

```
> pondsca <- cca(downweight(pondsdiat))
> pondsca

Call: cca(X = downweight(pondsdiat))

              Inertia Rank
Total               5
Unconstrained       5   29
Inertia is mean squared contingency coefficient
```

```
Eigenvalues for unconstrained axes:
  CA1   CA2   CA3   CA4   CA5   CA6   CA7   CA8
0.673 0.516 0.420 0.362 0.341 0.325 0.291 0.266
(Showed only 8 of all 29 unconstrained eigenvalues)
```

```
> summary(pondsca)

Call:
cca(X = downweight(pondsdiat))

Partitioning of mean squared contingency coefficient:
              Inertia Proportion
Total               5          1
Unconstrained       5          1

Eigenvalues, and their contribution to the mean squared contingency coefficient

Importance of components:
                        CA1    CA2    CA3    CA4    CA5    CA6
....
Proportion Explained  0.0286 0.0266 0.0232 0.021 0.0169 0.0153
Cumulative Proportion 0.8298 0.8564 0.8796 0.901 0.9175 0.9328
                        CA19   CA20   CA21   CA22    CA23   CA24
Eigenvalue            0.0678 0.0639 0.0519 0.0359 0.03477 0.0235
Proportion Explained  0.0136 0.0128 0.0104 0.0072 0.00696 0.0047
Cumulative Proportion 0.9464 0.9592 0.9696 0.9768 0.98371 0.9884
                         CA25   CA26    CA27    CA28    CA29
Eigenvalue            0.01980 0.0145 0.01244 0.00931 0.00190
Proportion Explained  0.00396 0.0029 0.00249 0.00186 0.00038
Cumulative Proportion 0.99237 0.9953 0.99776 0.99962 1.00000

Scaling 2 for species and site scores
* Species are scaled proportional to eigenvalues
....
ST002A  0.99607  0.34930  0.11290  0.1855 -0.58966  0.13994
ST010A  0.86913  0.33400 -0.04098  0.4670  0.99007  0.33373
SU016A -0.58320 -1.00389 -0.31811 -0.6186  0.40217 -2.08693
SY002A  0.19660 -0.83703  0.12121  1.1096  0.21693 -0.35600
SY003A  0.36333  0.01010  0.00855  0.3060 -0.09488  0.17631
SY003C  0.32933 -0.42619 -0.19360  0.9700  1.45765  0.02937
SY010A  0.17065 -1.18427 -0.26521  1.5898  1.59697 -0.15965
....
```

Q and A

1. What are the values of λ_{1-4}, the eigenvalues for axes one to four?
2. How much of the total variance (inertia) in the species data is explained by axes 1 and 2, individually and combined?

2.1 The meaningful components

As with PCA, a scree plot is a useful way of visualising the important components for further analysis. Comprison of the eignevalues of the CA with those expected under the broken-stick model is also useful (Figure 5) . To apply the broken-stick model we need to know the total variance or inertia in the data. This is stored within the result of the CA in the form of `pondsca$tot.chi`.

```
> pondsca$tot.chi
```

```
[1] 4.996

> (bstick.diat <- bstick(29, tot.var = 4.996))

 [1] 0.682497 0.510221 0.424084 0.366658 0.323589 0.289134
 [7] 0.260421 0.235811 0.214276 0.195134 0.177907 0.162245
[13] 0.147889 0.134637 0.122332 0.110847 0.100079 0.089945
[19] 0.080375 0.071307 0.062694 0.054490 0.046659 0.039169
[25] 0.031991 0.025100 0.018474 0.012093 0.005941

> (ca.eig <- pondsca$CA$eig)

     CA1      CA2      CA3      CA4      CA5      CA6      CA7
0.672754 0.516221 0.419944 0.361907 0.340848 0.325353 0.291272
     CA8      CA9     CA10     CA11     CA12     CA13     CA14
0.266320 0.221947 0.211718 0.208879 0.165873 0.142697 0.132653
    CA15     CA16     CA17     CA18     CA19     CA20     CA21
0.116053 0.104829 0.084330 0.076460 0.067805 0.063924 0.051878
    CA22     CA23     CA24     CA25     CA26     CA27     CA28
0.035947 0.034768 0.023473 0.019797 0.014481 0.012439 0.009305
    CA29
0.001902

> plot(bstick.diat, type = "o", lty = "dotted",
+      ylim = range(bstick.diat, ca.eig),
+      xlab = "CA Axis", ylab = "Inertia",
+      main = "Ponds Diatom Data: Bstick")
> points(ca.eig, type = "o", col = "red")
```

Q and A

1. How many axes are significant when compared with the null model?

2.2 CA Biplots

Biplots, or *joint plots* as they are commonly called in the literature, are generate by plotting the site and species scores obtained from CA of the species matrix. As with PCA, these scores can be scaled so as to focus the resulting diagram on relationships among sites (`scaling = 1`) or species (`scaling = 2`) or some compromise of the two `scaling = 3`). So-called *biplot* and *Hill's* scaling are the two *types* of scaling that can be applied to three scalings mentioned above. These two types dictate how information on the species data is gleaned from the resulting biplot. With biplot scaling, the biplot rule applies and is most suited for short gradients. Hill's scaling equalises the average niche breadth for all axes and allows, for long gradients, interpretation via the distance rule. Consult your class notes for the two different rules. The `plot()` method for `cca()` produces a biplot of the results of a CA (Figure 6).

```
> plot(pondsca)
```

The `display` argument for the `plot()` method can be used to control what aspects of the results are plotted on the biplot. There are a number of valid options for `display`, but the most useful here are `"species"` and `"sites"`.

```
> plot(pondsca, display = "species")
> plot(pondsca, display = "sites")
```

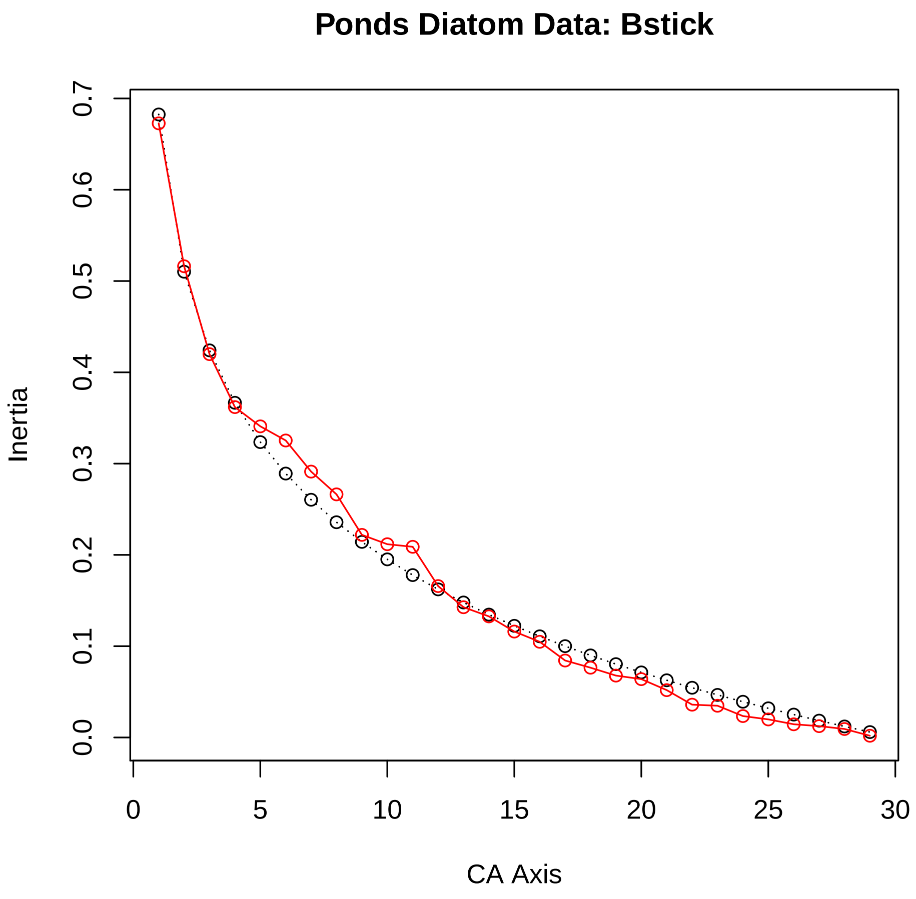

Figure 5: Scree plot of the eigenvalues obtained from a CA of the Ponds diatom data. The solid line represents the observed eigenvalues. The dotted line indicates the expected values under a null model obtained from the broken-stick distribution.

Q and A

1. Are there outliers in the plot (species or samples)?
2. Is there an arch apparent in the biplot?

It is useful to enhance these plots with further information to aid interpretation. One useful addition is to overlay measures of diversity on to the biplot. Diversity indices can be calculated using the `diversity()` function of `vegan`. One particular measure of diverist is Hill's N_2, which gives the effective number of occurences of a species across all sites or the effective number of species in an individual plot. Hill's N_2 is equivalent to the Inverse Simpson diversity measure we we tell `diversity()` to use. The third argument is what Jari Oksanen refers to as the MARGIN; rows (1) or columns (2). For the effective number of occurences of a particular species then the calculation should be over the columns, but rows should be used to calculate the effective numbers of species per sample.

```
> spp.n2 <- renyi(t(pondsdiat), scales = 2, hill = TRUE)
> site.n2 <- renyi(pondsdiat, scales = 2, hill = TRUE)
```

Having calculated the relevant diversity information, this can be used to augment the biplot. A simple way to use these data is to scale the plotting symbol according to the Hill's N_2 value, using the `cex` graphical parameter. The `identify` function is used to label the plotting sybols after plotting (Figure 7). Right click the graph when you are finished labelling.

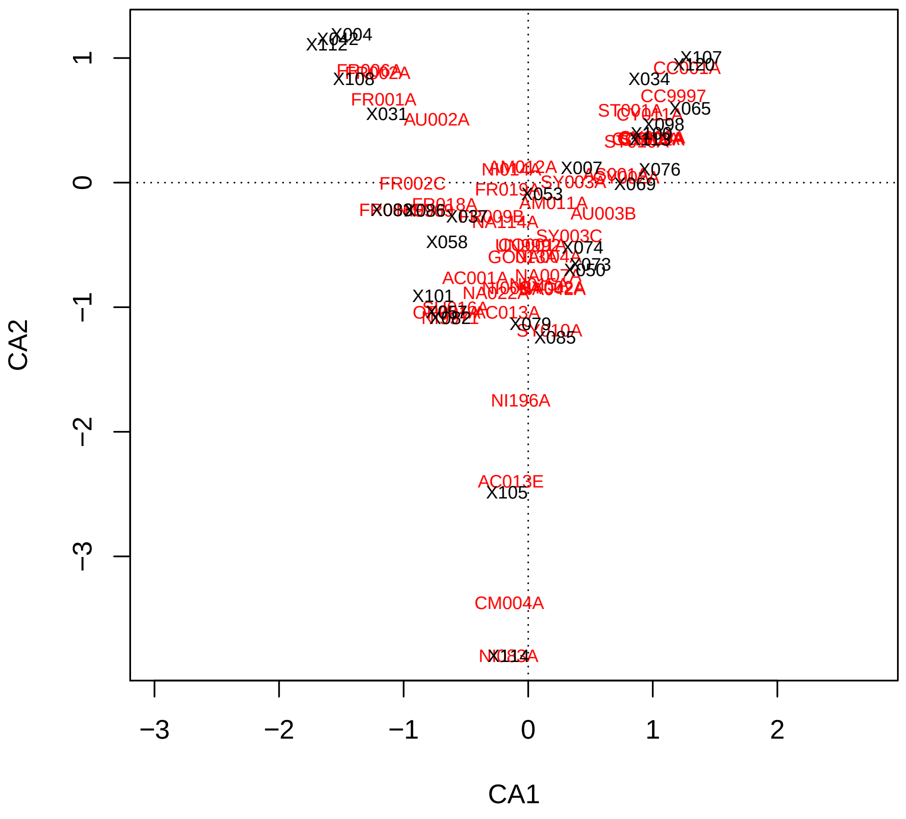

Figure 6: Correspondence Analysis biplot of the Ponds diatom data set.

```
> ca.plot <- plot(pondsca, type = "n")
> points(pondsca, display = "species", cex = 0.3 * spp.n2)
> identify(ca.plot, what = "species", col = "red", ps = 10)
> plot(pondsca, type = "n")
> points(pondsca, display = "sites", cex = 0.5 * site.n2)
> identify(ca.plot, what = "sites", col = "red", ps = 10)
```

Q and A

1. Are outlying species common or rare?
2. Are outlying samples dominated by a few, rare, species?
3. What effect does the choice of scaling have on the ordination plots? Use the code below to display the biplots with two different scalings.

```
> oldpar <- par(mfrow = c(1,2), pty = "s")
> plot(pondsca, scaling = 2, main = "Inter-species distance & Biplot scaling")
> plot(pondsca, scaling = -1, main = "Inter-sample distance & Hills scaling")
> par(oldpar)
```

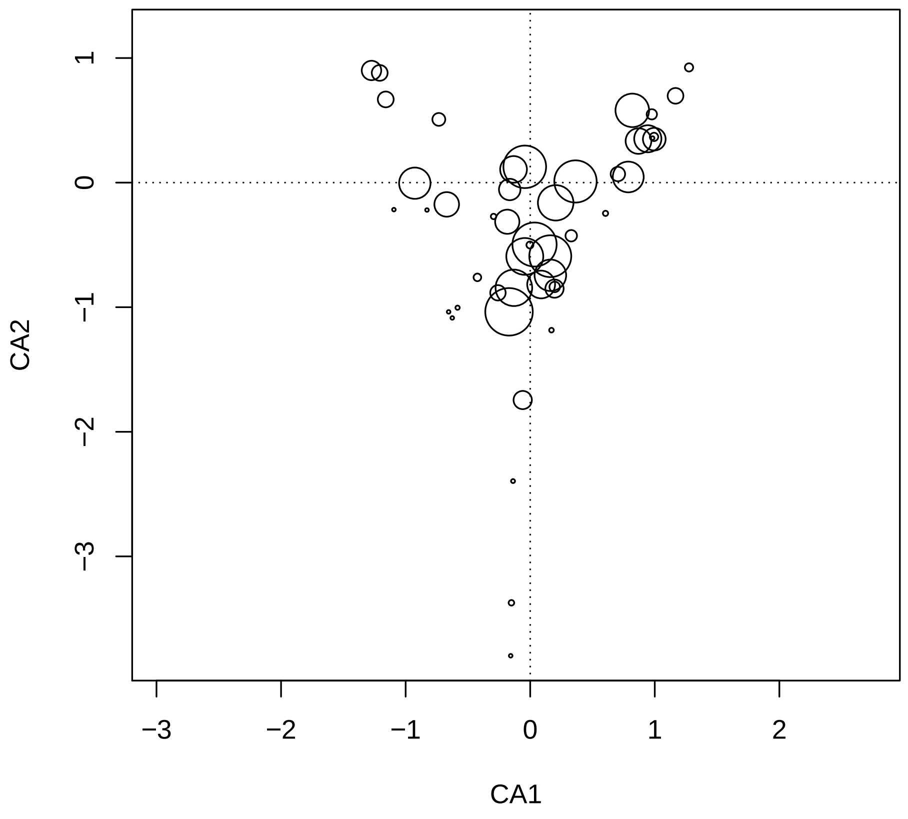

Figure 7: Correspondence Analysis biplot of the Ponds diatom data set. Points are scaled relative to the Hill's N_2 value for each species.

3 Detrended Correspondence Analysis

Correspondence analysis has been shown to be a more robust method for community ordination, where species show unimodal, rather than linear, responses to the underlying gradients. However, when analysing data sets with long gradients, CA was prone the *arch* effect, where by ordination configurations are curved within ordination space—the result of attempting to map non-linearities into Euclidean space. Whilst still interpretable, these curvatures prevent the accurate reconstruction of the underlying gradients from the CA results. Detrended Correspondence Analysis (DCA) was developed to address the following issues:

1. Single long gradients appear as arched configurations in the ordination,
2. At the ends of the gradients, sites are packed more closely together than at the centre of the space.

DCA was oringinally implemented in the DECORANA computer program. the `decorana()` function in `vegan` fits DCA models as implemented in DECORANA.

```
> pondsdca <- decorana(downweight(pondsdiat))
> pondsdca

Call:
decorana(veg = downweight(pondsdiat))
```

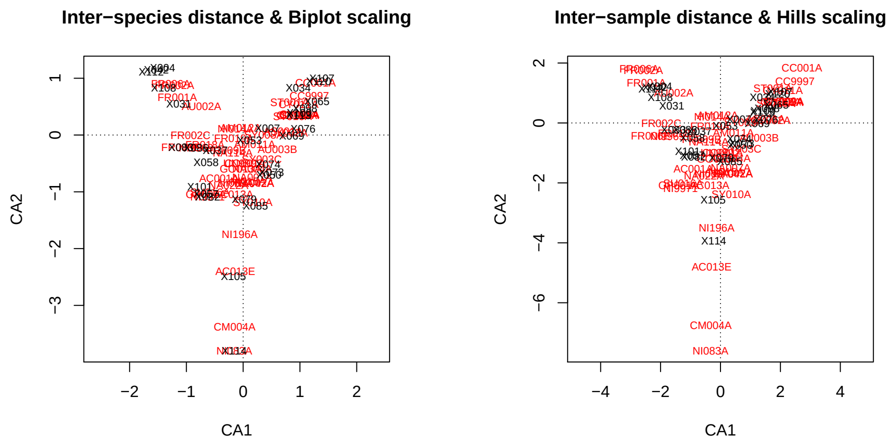

Figure 8: The effect of scaling on CA biplots of the Ponds diatom data.

```
Detrended correspondence analysis with 26 segments.
Rescaling of axes with 4 iterations.
Downweighting of rare species from fraction 1/5.

                 DCA1  DCA2  DCA3  DCA4
Eigenvalues     0.671 0.334 0.297 0.274
Decorana values 0.673 0.356 0.280 0.138
Axis lengths    3.868 2.863 2.462 2.274
```

3.1 Q and A

1. What are the gradient lengths for the four reported axes?
2. What do these values indicate about the likely species responses along these gradients?
3. What are the values for λ_{1-4}?
4. How much of the total variation is explained by the first 2 axes, and all four reported axes?

The total variance (inertia) in the DCA is the same as for CA (4.996 for the Ponds diatom data set). The eigenvalues are stored in `pondsdca$evals`. To calculate the proportions explained individually be the reported axes, divide `pondsdca$evals` by the total variance (4.996). The `cumsum()` function can be used to report the cumulative proportion of the variance explained by the four axes.

```
> pondsdca$evals / 4.996

   DCA1    DCA2    DCA3    DCA4
0.13438 0.06686 0.05952 0.05483

> cumsum(pondsdca$evals / 4.996)

  DCA1   DCA2   DCA3   DCA4
0.1344 0.2012 0.2608 0.3156
```

Again, a more thorough output is achieved using the `summary()` method of `decorana()`, and a biplot of the results can be produced using the `plot()` method.

```
> summary(pondsdca)
```

```
Call:
decorana(veg = downweight(pondsdiat))

Detrended correspondence analysis with 26 segments.
Rescaling of axes with 4 iterations.
Downweighting of rare species from fraction 1/5.

                 DCA1  DCA2  DCA3  DCA4
Eigenvalues     0.671 0.334 0.297 0.274
Decorana values 0.673 0.356 0.280 0.138
Axis lengths    3.868 2.863 2.462 2.274

Species scores:

           DCA1     DCA2     DCA3     DCA4  Weights Totals
AC001A -0.96606  2.48406  0.70099  0.09066  0.80688  37.10
AC013A -0.34394 -1.16186  0.09649  0.59911  1.00000 187.16
AC013E -0.21701 -0.25422 -0.60829 -1.37731  0.42156   7.19
....
Site scores:

         DCA1     DCA2     DCA3     DCA4 Totals
X004 -1.84448  0.59323  0.60779 -0.82675   88.9
X007  0.58677  0.03011 -0.00578  0.22698   58.5
X031 -1.43969  0.01375  0.19244 -0.03780   86.7
X034  1.22380  0.12356 -0.28863  0.24893   82.0
....
```

A biplot of the results can be produced using the `plot()` method (Figure 9).

```
> plot(pondsdca)
```

3.2 Additional indirect gradient analysis topics

This sections contains some additional exmaples of fitting some of the other indirect gradient analysis methods introduced in today's lecture. If you have time at the end of this practical then attempt some of these analyses. Don't worry if you don't have time, you can always refer back to this handout at a later date if you wish to give some of these methods a go.

3.2.1 Non-metric multidimensional scaling

Non-metric multidimensional scaling can be performed using `isoMDS()` in package `MASS`. It requires a dissimilarity matrix as an input argument. `vegdist()` can calculate these dissimilarity matrices, with the default option being Bray-Curtis dissimilarity. Function `metaMDS()` in `vegan` implements helper functions that start the NMDS iterative algorithm at *k* randomly selected start points and choose the best model fit (i.e. that reduces the stress the most). `metaMDS()` only requires a matrix of data as the function internally calculates the specified dissimilarity matrix for you. We will fit a NMDS model to the Ponds environmental data using Euclidean distances.

```
> library(MASS)
> set.seed(123456)
> euclid.dis <- vegdist(pondsenv, "euclidean")
> nmds.env <- metaMDS(pondsenv, distance = "euclidean", trymax = 50)

Run 0 stress 0.1739
Run 1 stress 0.1889
Run 2 stress 0.197
```

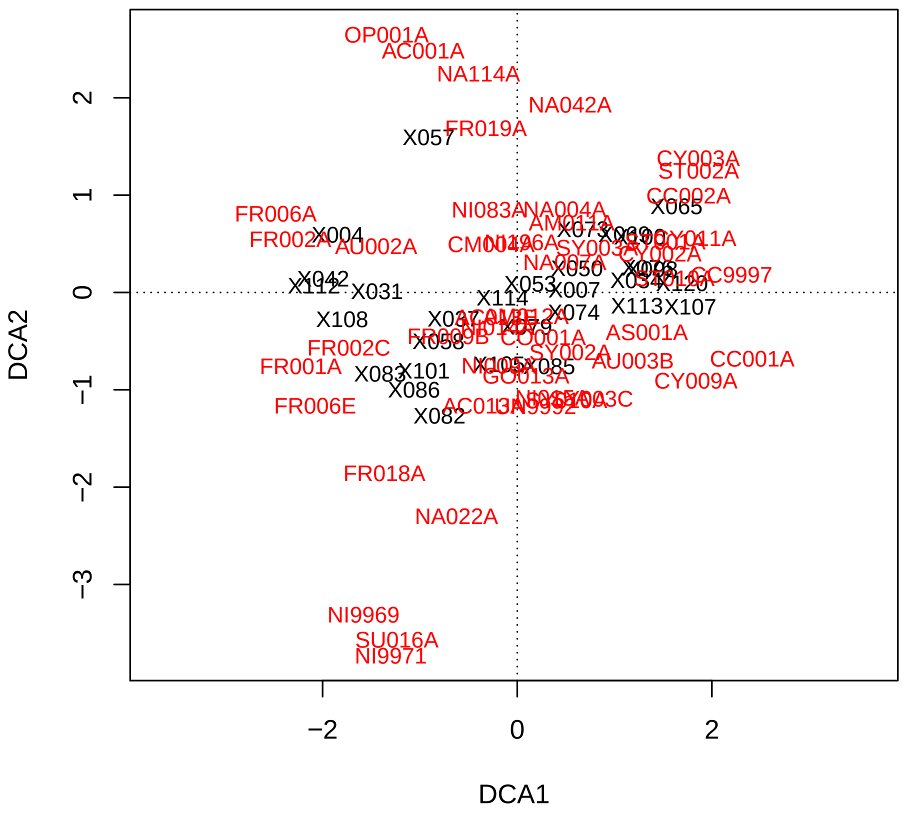

Figure 9: Detrended Correspondence Analysis biplot of the Ponds diatom data set.

```
Run 3 stress 0.197
Run 4 stress 0.1739
... New best solution
... procrustes: rmse 0.002061  max resid 0.009572
*** Solution reached
```

```
> nmds.env

Call:
metaMDS(comm = pondsenv, distance = "euclidean", trymax = 50)

global Multidimensional Scaling using monoMDS

Data:     pondsenv
Distance: euclidean

Dimensions: 2
Stress:     0.1739
Stress type 1, weak ties
Two convergent solutions found after 4 tries
Scaling: centring, PC rotation
Species: scores missing
```

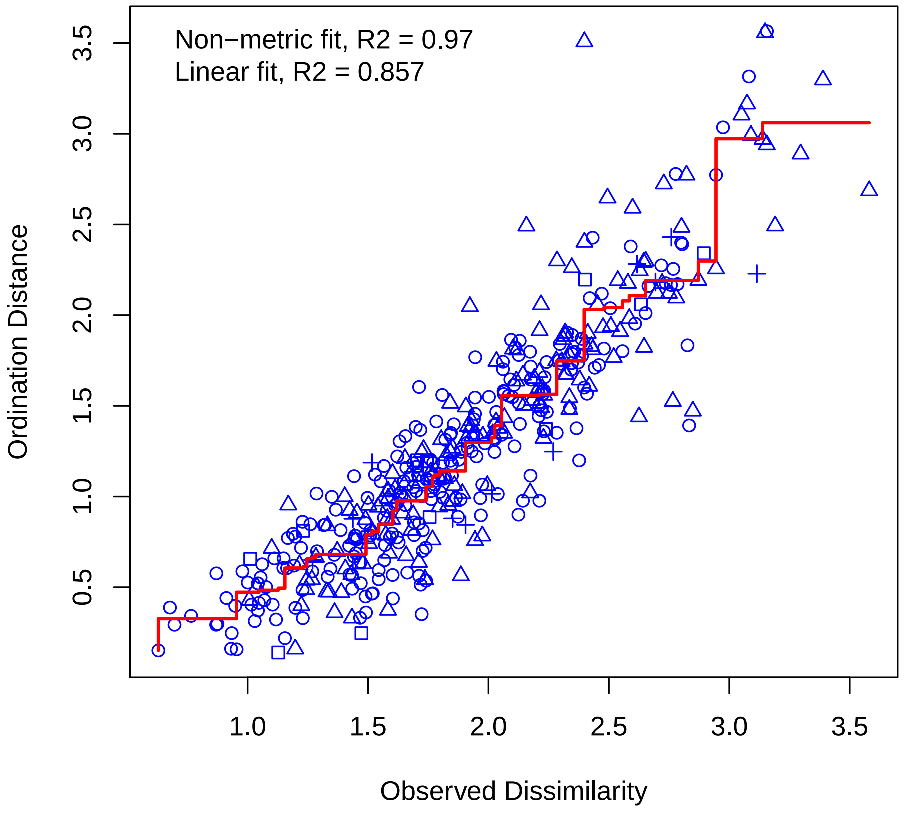

Figure 10: Shepard plot comparing the original Euclidean distancesand the NMDS ordination-based distances between ponds calculated using the environmental data.

NMDS maps the observed distances on to the ordination space in a non-linear fashion. How well this mapping is achieved can be visualised using `stressplot()`, which draws a Shepard plot and the fit of the NMDS as a stepped line. `stressplot()` also displays two correlation statistics for the goodness of the fit. The correlation based on stress is $R^2 = 1 - S^2$, and the "fit-based" correlation is the correlation between the fitted values, $\theta(d)$ and the original distances, d, which is the correlation between the stepped line and the points (Figure 10).

```
> stressplot(nmds.env, euclid.dis)
```

To draw the ordination diagram for the NMDS model, `vegan` provides a `plot()` method (Figure 11):

```
> plot(nmds.env, type = "text")
```

3.2.2 Comparing ordinations using Procrustes rotation

Two ordinations can be very similar but this similarity may be masked as a result of the two ordinations having different scalings, orientations and signs. Procrustes rotation is a good way of of comparing ordination configurations. `vegan` has function `procrustes()`, which performs Procrustes rotation.

```
> pondsenv.pro <- procrustes(nmds.env, pondspca, symmetric = TRUE)
> summary(pondsenv.pro)
```

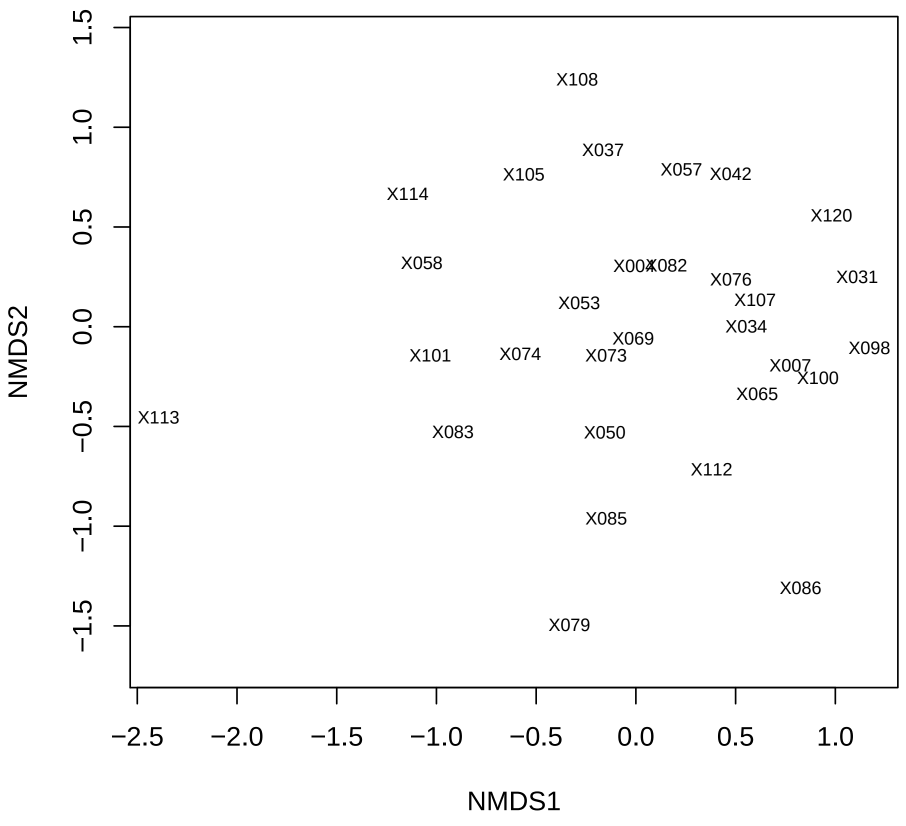

Figure 11: NMDS ordination of the Euclidean distance between ponds calculated using the environmental data.

```
Call:
procrustes(X = nmds.env, Y = pondspca, symmetric = TRUE)

Number of objects: 30    Number of dimensions: 2

Procrustes sum of squares:
 0.2447
Procrustes root mean squared error:
 0.09032
Quantiles of Procrustes errors:
     Min       1Q   Median       3Q      Max
0.007652 0.027310 0.050160 0.078327 0.357650

Rotation matrix:
        [,1]   [,2]
[1,] -0.5035 0.8640
[2,]  0.8640 0.5035

Translation of averages:
           [,1]      [,2]
```

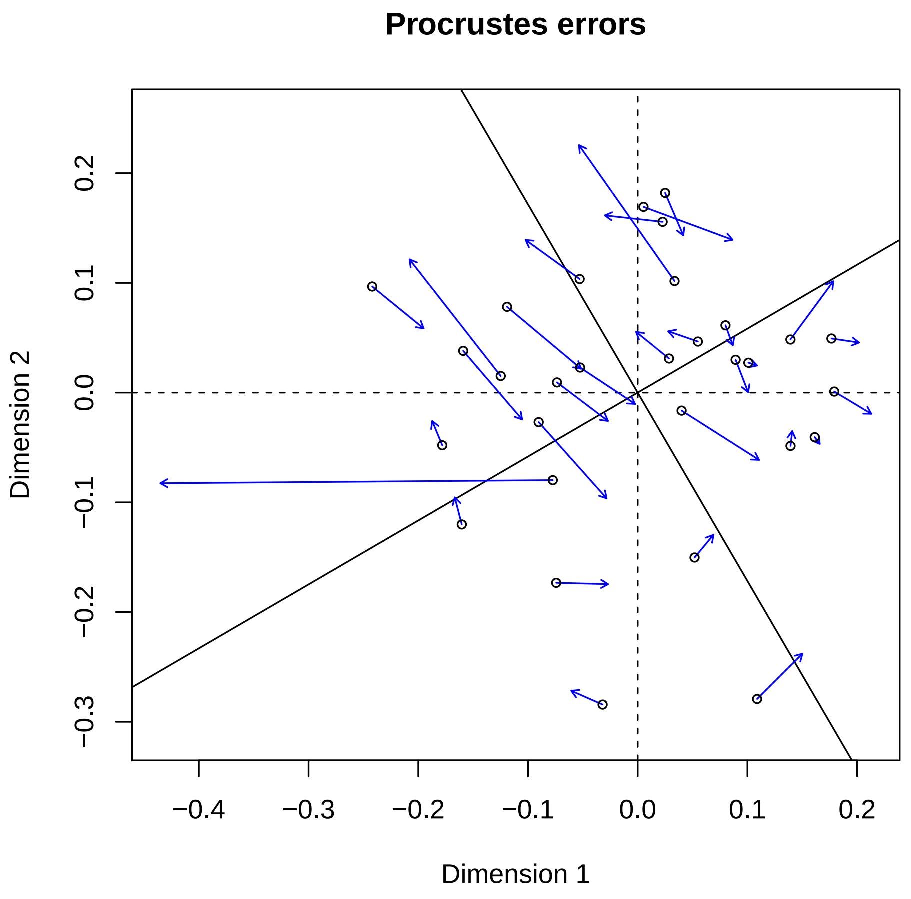

Figure 12: Procrustes superimposition plot, showing the differences between the NMDS configuration (circles) and that obtained from a a PCA (ends of the arrows) of the Ponds environmental data.

```
[1,] -4.395e-18 2.432e-18

Scaling of target:
[1] 0.8691
```

The `plot()` method for `procrustes()` can produce two kinds of plots; 1) the ordination digram showing the comparison between the two configurations (Figure 12), and 2) a residuals plot (Figure 13).

```
> par(mfrow = c(1,2))
> plot(pondsenv.pro, kind = "1")
> plot(pondsenv.pro, kind = "2")
> par(mfrow = c(1,1))
```

The PROTEST method allows you to test whether the two configurations are significantly similar to one another by means of a permutation test.

```
> set.seed(123456)
> pondsenv.prot <- protest(nmds.env, pondspca)
> pondsenv.prot

Call:
protest(X = nmds.env, Y = pondspca)
```

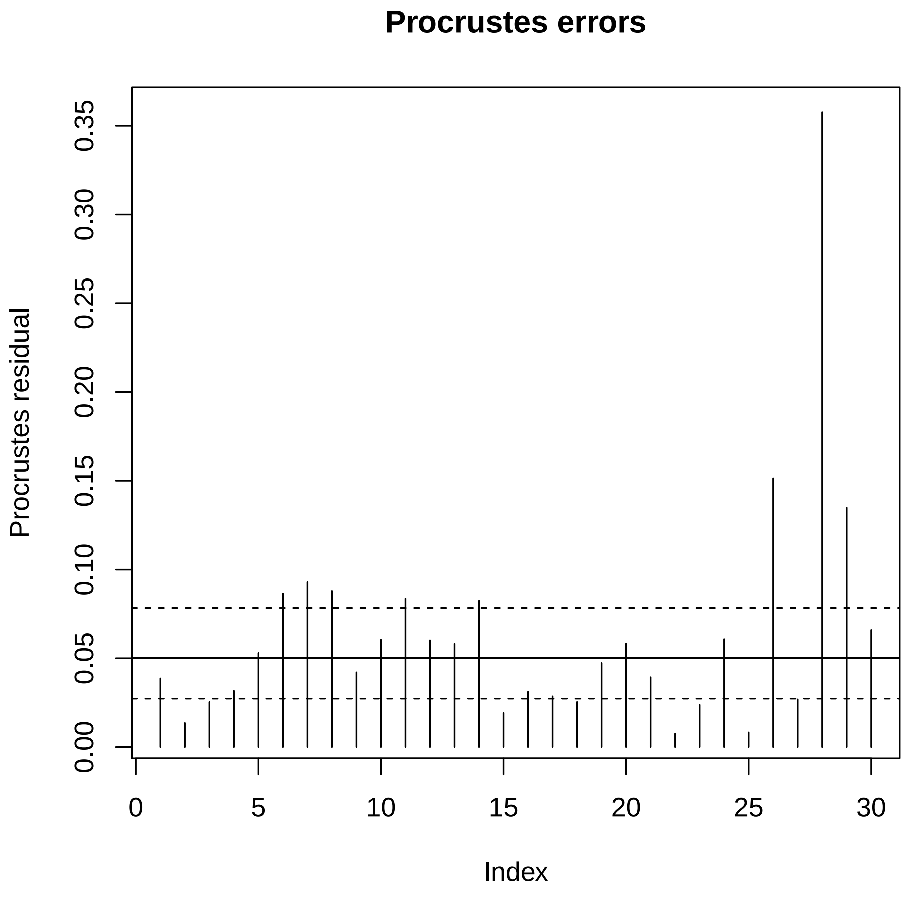

Figure 13: Procrustes rotation residuals plot, showing the differences between the NMDS configuration and that obtained from a a PCA of the Ponds environmental data.

```
Procrustes Sum of Squares (m12 squared):        0.245
Correlation in a symmetric Procrustes rotation: 0.869
Significance:  0.001
Based on 999 permutations.
```