Ordination

PCA for English Channel:

1) How the variables are correlated
2) Mapping out the samples in PC space

- Where those samples fall on environmental axes, how that relates to season, etc.

How about the same kind of mapping for **community composition**?

```
species.data[1,1:8]

##   Ceratium.fusus Ceratium.lineatum Nitzschia.closterium
## 1              0           0.02002                  0.2
##   Nitzschia.delicatissima Nitzschia.panduriformis Chaetoceros.danicus
## 1                   1.111                       0                   0
##   Chaetoceros.decipiens Roperia.tesselata
## 1                     0           0.03966
```

Ordination

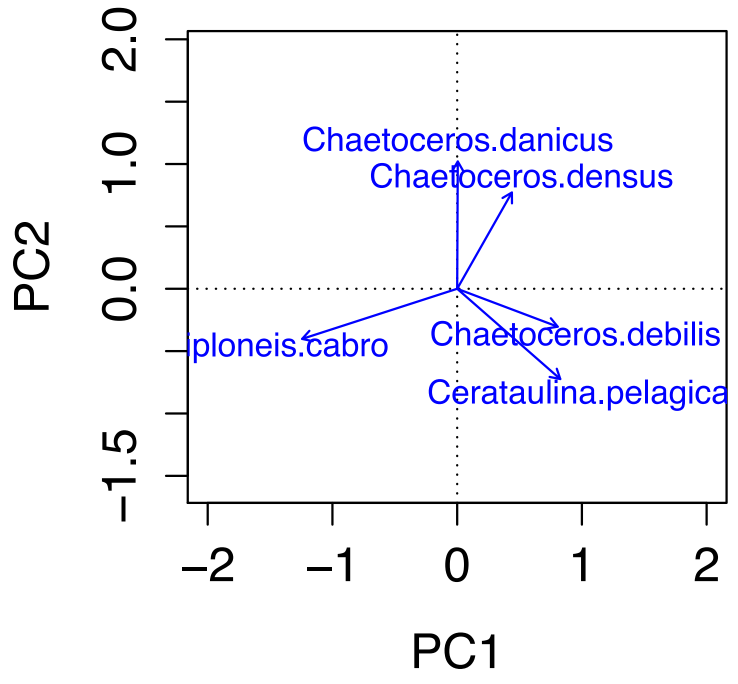
What are the major axes of variation in composition? Dimensionality?

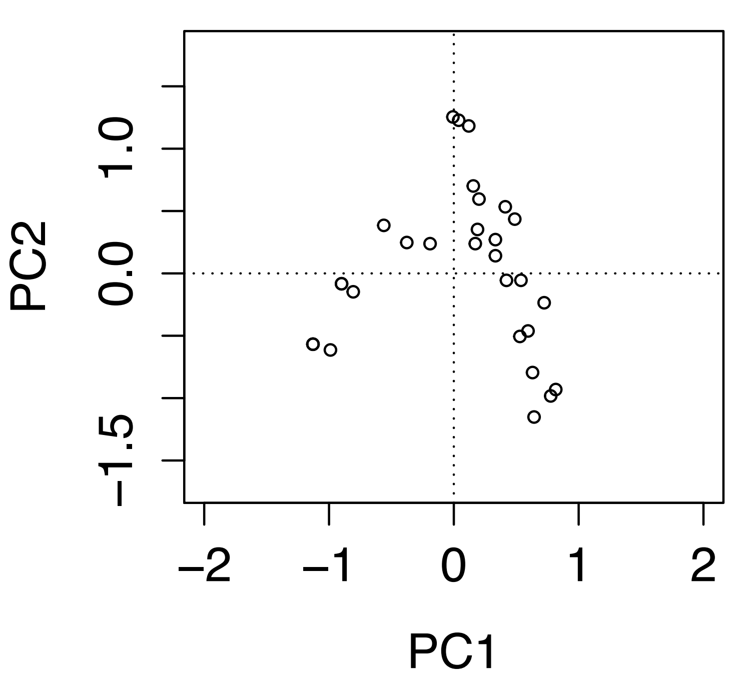
This is **ordination**: ordering community samples along gradients of composition

Can also use this to visualize if composition correlates with the environment, or with experimental treatments, etc.

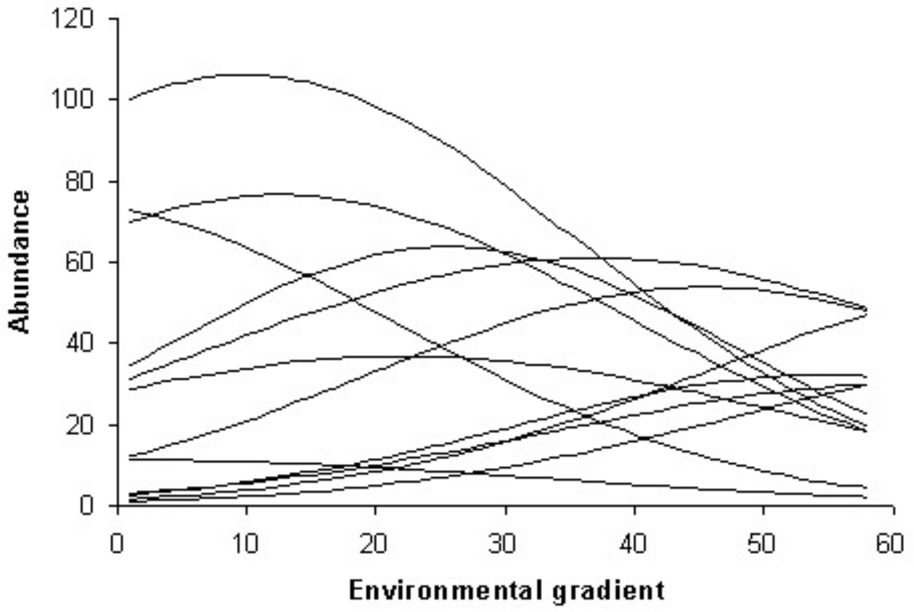
```
species.data[1,1:8]

##   Ceratium.fusus Ceratium.lineatum Nitzschia.closterium
## 1              0           0.02002                  0.2
##   Nitzschia.delicatissima Nitzschia.panduriformis Chaetoceros.danicus
## 1                   1.111                       0                   0
##   Chaetoceros.decipiens Roperia.tesselata
## 1                     0           0.03966
```



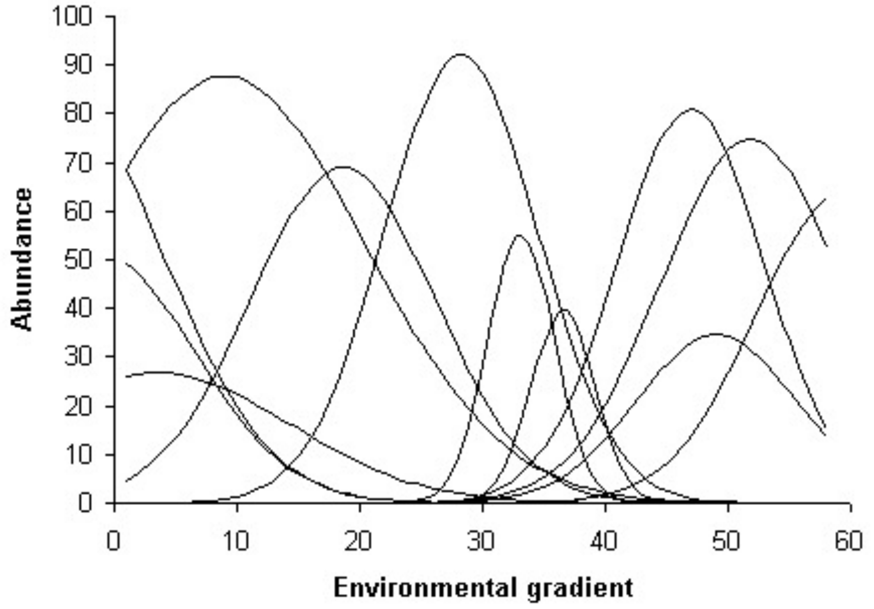


- PC1 34%, PC2 25%
- We can see how species tend to correlate along major axes
- We can map/ordinate samples in 'species space', for further analysis
- Problem: PCA is a linear method



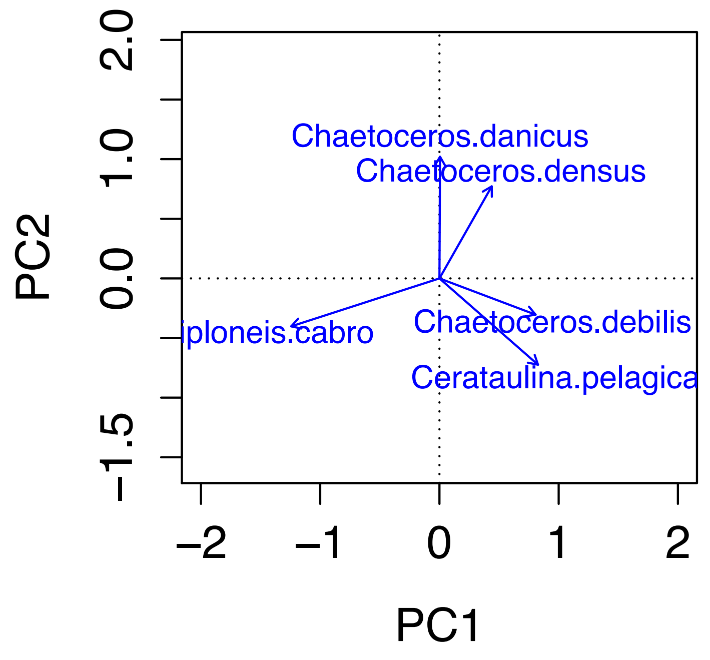
Low turnover / beta diversity

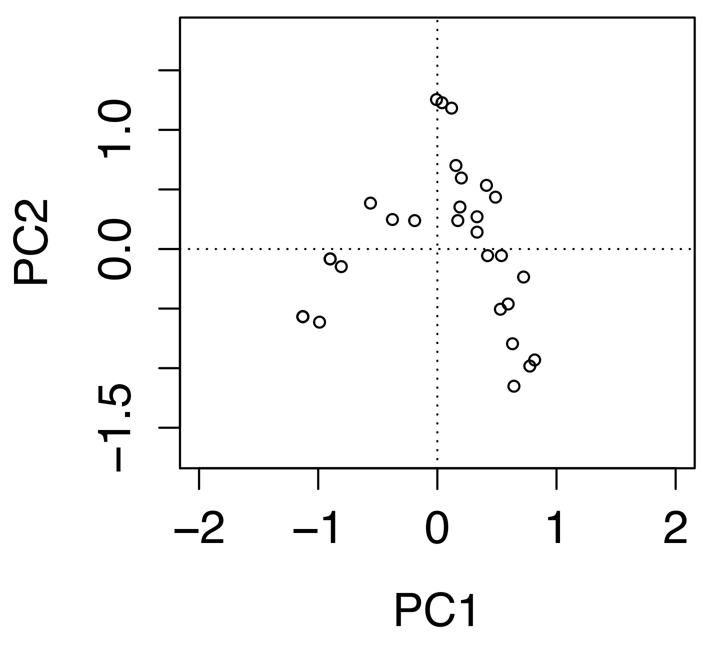
PCA might work OK

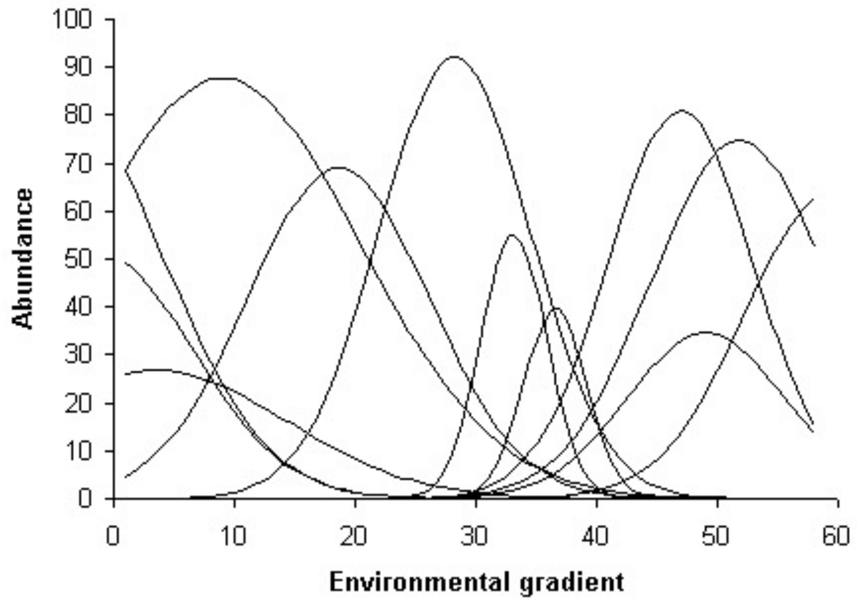


High turnover / beta diversity

PCA will do a poor job of reconstructing this







The ‘arch’ or ‘horseshoe’ effect

Samples at opposite ends of a gradient look somewhat similar, because intermediate species are absent

Community similarity / dissimilarity metrics

There are **many**

Jaccard index, for presence-absence (binary) data

$$S = \frac{J}{A + B + J}$$

J is the number of species present at both sites (or at both times)

A is the number of species present only at site A; ***B*** is the number of species only at site B

S = similarity, **D** = 1 – S = dissimilarity

Note this ignores species that are **absent from both sites**

The **'double zero'** problem: a species could be absent from two sites for two different reasons

Most community similarity metrics treat **presences as more informative than absences**

Community similarity / dissimilarity metrics

There are **many**

Bray-Curtis is a similar index, for abundance data

$$D_{jk} = \frac{\sum_i \left| X_{ij} - X_{ik} \right|}{\sum_i (X_{ij} + X_{ik})}$$

For abundance, you usually need to **transform** as well

To make species of equal importance, regardless of absolute abundance

E.g. vegan will do:
1) square root
2) Wisconsin double standardization: divide each species by its max; make each sample have the same total

Also, dropping rare species entirely is good, they just add noise

Clearly there are a lot of judgment calls, because we aren't modeling the raw data

Part of the solution is to use dissimilarity indices appropriate for communities

- Jaccard, Bray-Curtis

How do we use dissimilarities to find underlying gradients?

Dissimilarity matrix

	Site1	Site2	Site3	Site4
Site1	0	0.2	0.6	0.3
Site2		0	0.5	0.1
Site3			0	0.8
Site4				0

Note: the original abundance data is gone

- the dissimilarity matrix is the ‘data’ for the ordination methods I will now explain
- Can make with vegdist()

Part of the solution is to use dissimilarity indices appropriate for communities

- Jaccard, Bray-Curtis

How do we use dissimilarities to find underlying gradients?

Dissimilarity matrix

	Time1	Time2	Time3	Time4
Time1	0	0.2	0.6	0.3
Time2		0	0.5	0.1
Time3			0	0.8
Time4				0

Note: the original abundance data is gone

- the dissimilarity matrix is the ‘data’ for the ordination methods I will now explain
- Can make with vegdist()

Dissimilarity matrix

	Site1	Site2	Site3	Site4
Site1	0	0.2	0.6	0.3
Site2		0	0.5	0.1
Site3			0	0.8
Site4				0

Principal coordinates analysis = PCoA = (metric) multidimensional scaling

Find ordination axes so that spatial **distances** approximate **dissimilarities**

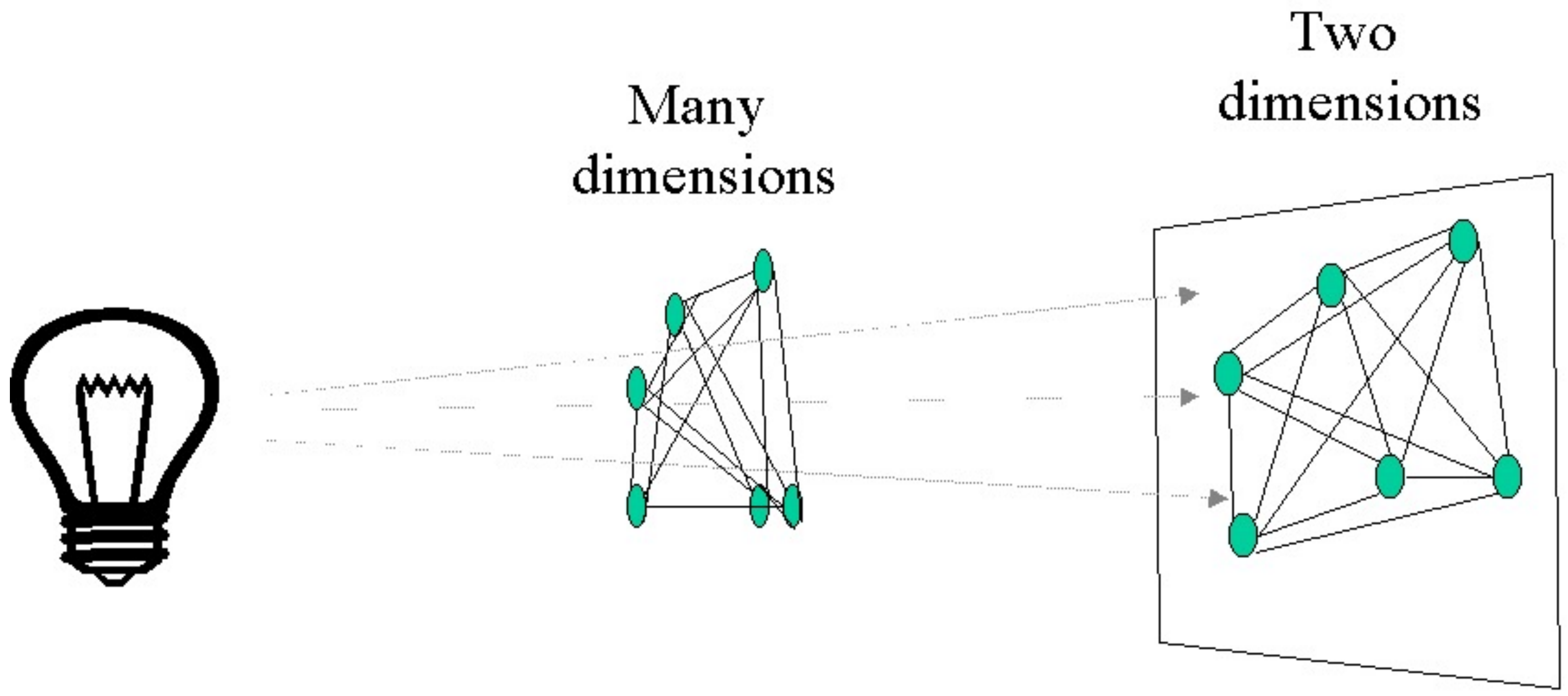
Axis 1 accounts for as much dissimilarity as possible

Axis 2 is orthogonal, accounts for as much remaining dissimilarity as possible

An **eigenanalysis** of the dissimilarity matrix

Principal coordinates analysis

The underlying math of PCoA is harder to visualize than PCA



The upshot: can use **any dissimilarity index** to map out samples along orthogonal axes

Useful for community data, and anything else where euclidean distance not great

- E.g. comparing individuals based on trait distances

Principal coordinates analysis

```
species.data.use = wisconsin(sqrt(species.data.use))

pcoa = capscale(species.data.use ~ 1, dist = "bray")
pcoa

## Call: capscale(formula = species.data.use ~ 1, distance = "bray")
##
##                  Inertia Rank
## Total               5.17
## Real Total          6.21
## Unconstrained       6.21   19
## Imaginary          -1.04   20
## Inertia is squared Bray distance
##
## Eigenvalues for unconstrained axes:
##  MDS1  MDS2  MDS3  MDS4  MDS5  MDS6  MDS7  MDS8
## 2.635 0.907 0.804 0.500 0.459 0.196 0.157 0.140
## (Showed only 8 of all 19 unconstrained eigenvalues)
```

Capscale() makes the dissimilarity matrix for you

Total inertia = summed eigenvalues = total variation

Imaginary: can get negative eigenvalues, corrections are made

Principal coordinates analysis

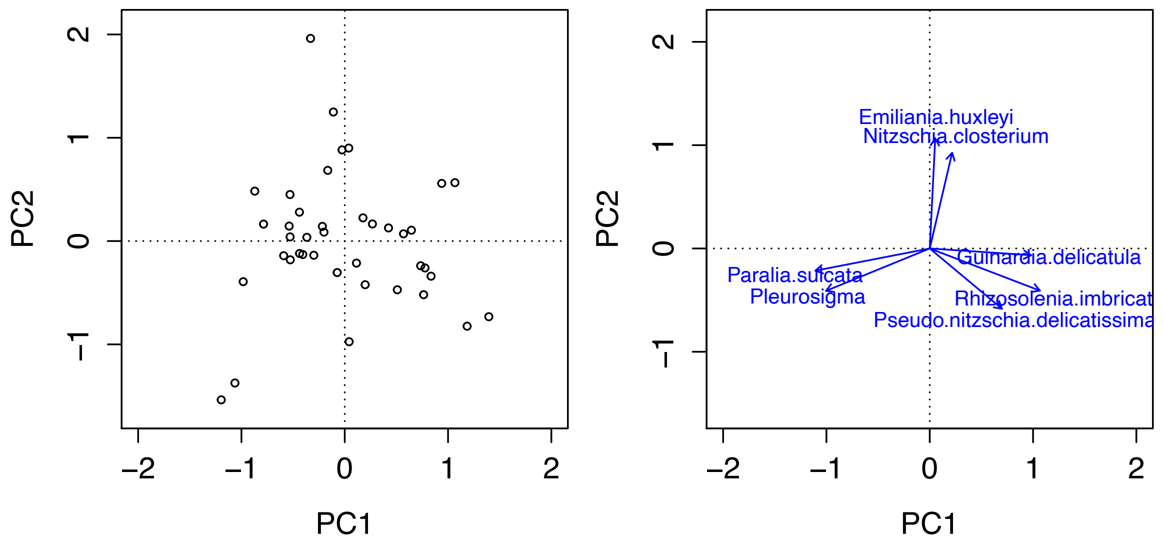
```
summary(pcoa)
##
## Eigenvalues, and their contribution to the squared Bray distance
##
## Importance of components:
##                         MDS1  MDS2  MDS3   MDS4   MDS5   MDS6   MDS7   MDS8
## Eigenvalue             2.635 0.907 0.804 0.4999 0.4588 0.1958 0.1575 0.1396
## Proportion Explained  0.424 0.146 0.130 0.0805 0.0739 0.0315 0.0254 0.0225
## Cumulative Proportion 0.424 0.571 0.700 0.7806 0.8545 0.8861 0.9114 0.9339
```

First axis explains 42% of total community dissimilarity

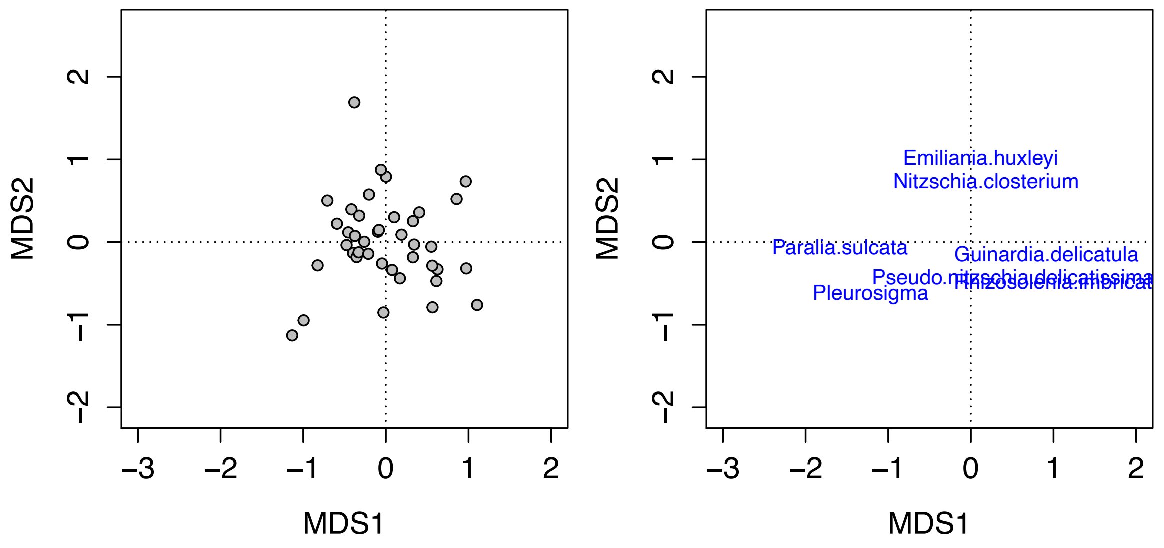
Principal coordinates analysis

```
par(mfrow = c(2,2))
biplot(pca, type = 'text', col = c('blue'), display = "species", xlim = c(-
2,2), cex = 0.5)
biplot(pca, type = c('points'), col = c('black'), display = "sites", xlim =
c(-2,2))
```

```
plot(pcoa, type = 'n', xlim = c(-3, 2))
text(pcoa, "species", col = 'blue', cex = 0.7)
plot(pcoa, type = 'n', xlim = c(-3, 2))
points(pcoa, col = 'black', bg = 'grey', pch = 21)
```



PCA



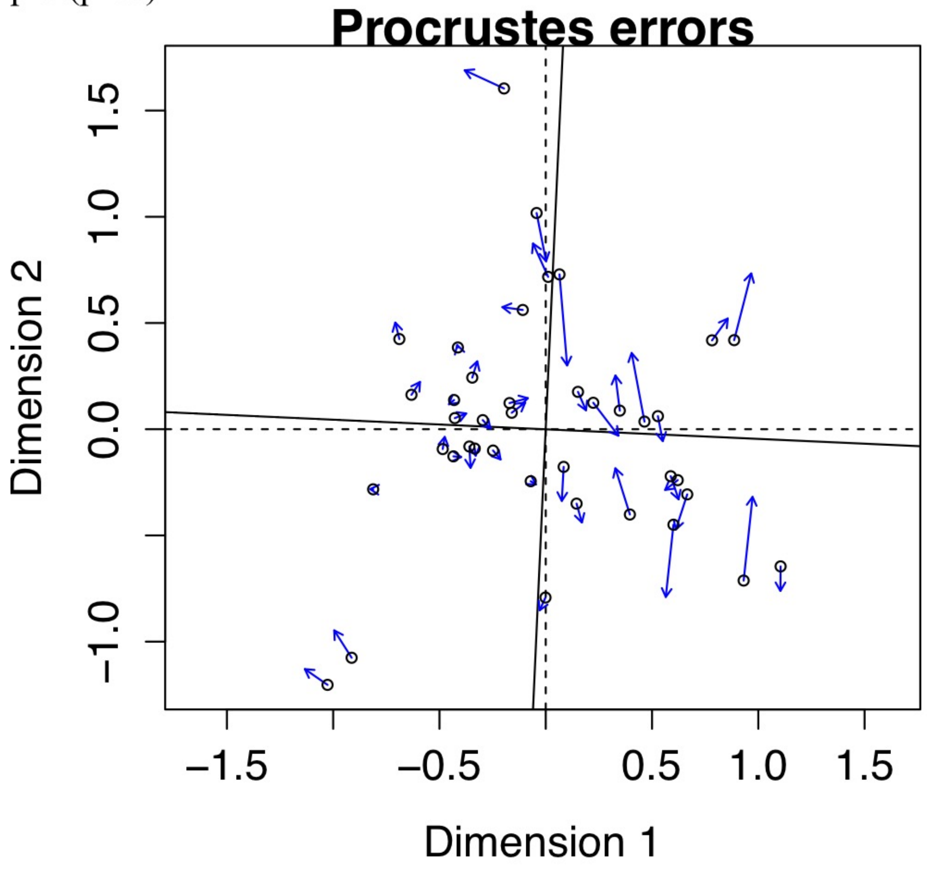
PCoA

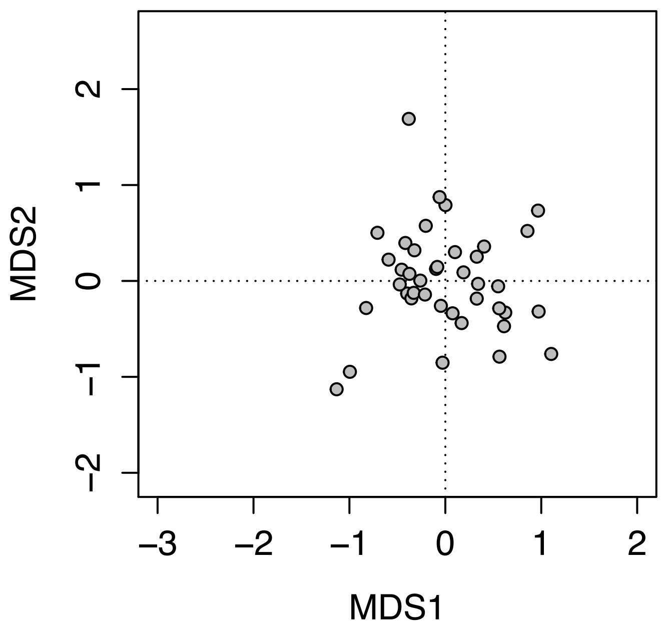
- Note: species are points (scores), not vectors for PCoA
- **Weighted average** of the sample scores where the species occur
- Samples close to a species have the highest abundance of that species

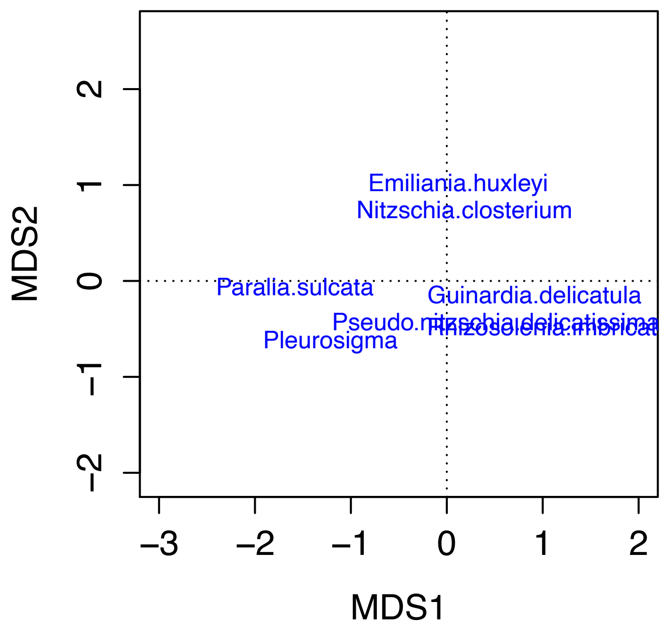
How much do the PCA and PCoA ordinations differ? Can use **procrustes** rotation

- Rotate one ordination to line up as well as possible with another
- Arrows show the difference in where the samples are located
- In this case, PCA and PCoA not that different (not always the case)

```
proc = procrustes(pcoa, pca)
plot(proc)
```



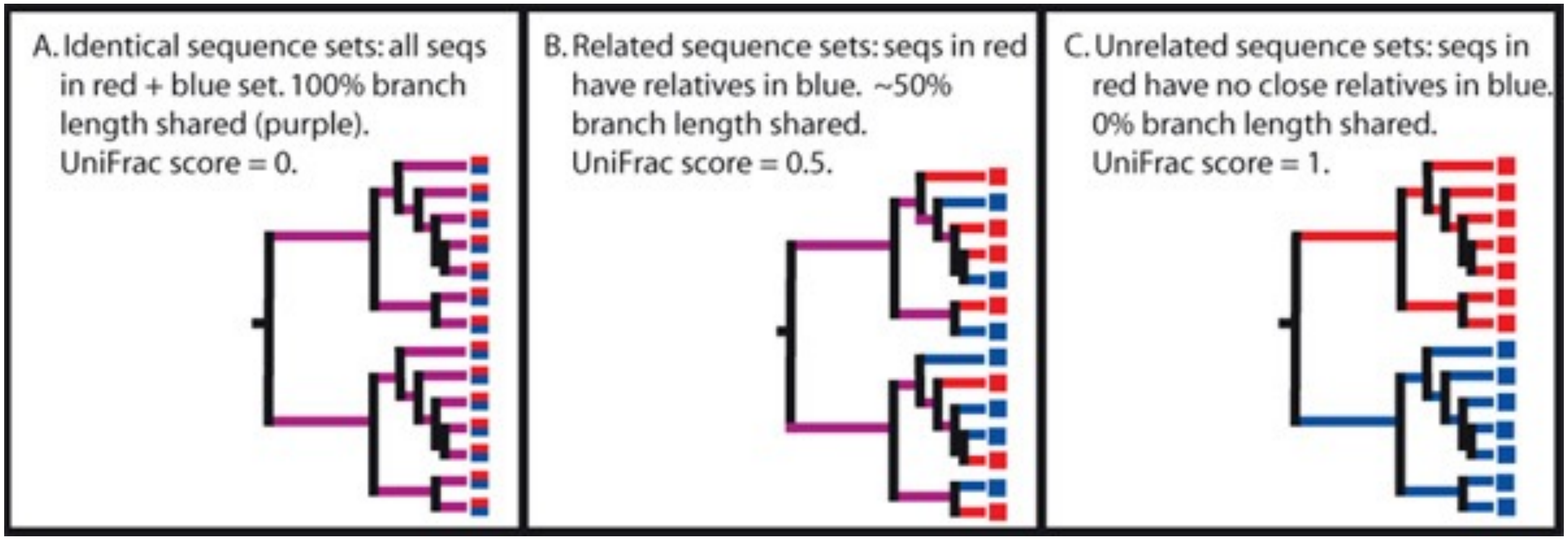




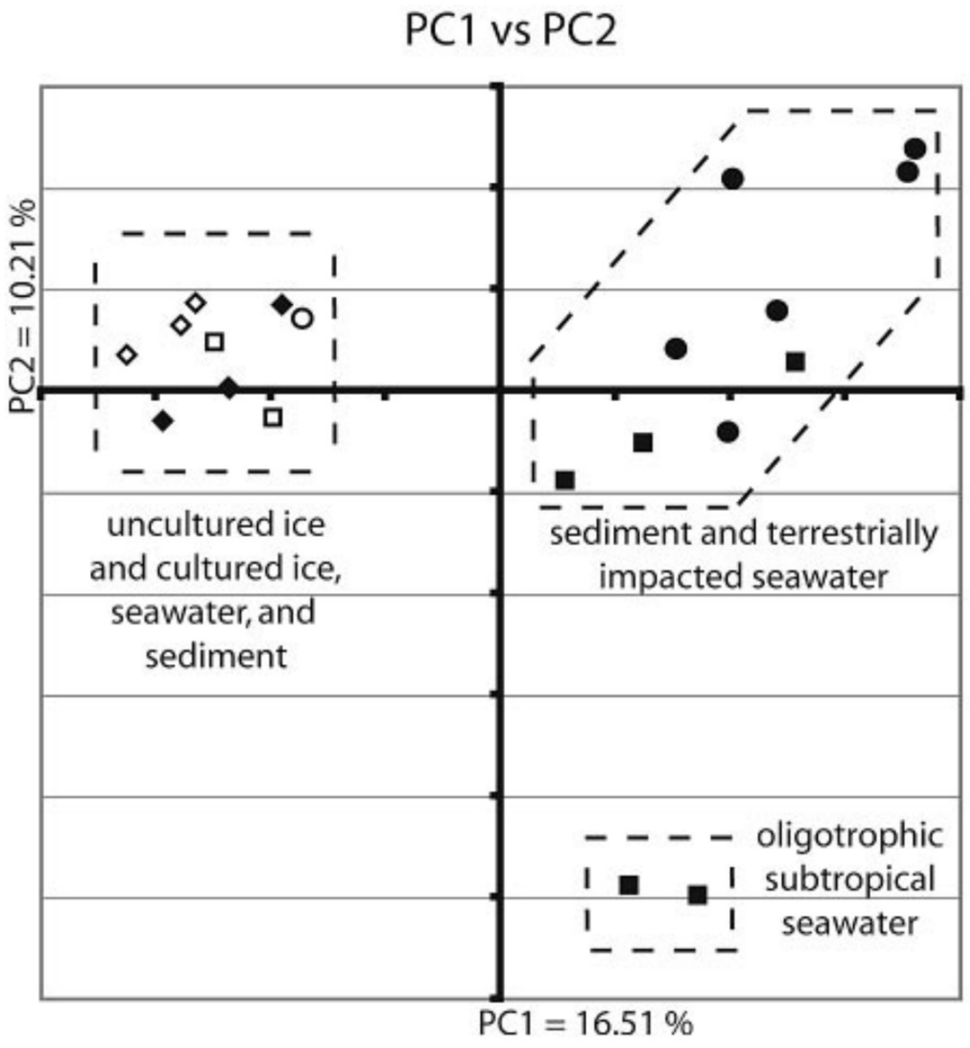
We've ordinated the data. What next?

- Do samples from different areas / times / experimental treatments differ?
- Are the axes correlated with environmental variables?
- Do related species have similar scores?
- Show these later

PCoA example: **Unifrac** for microbial genetic data


A. Identical sequence sets: all seqs in red + blue set. 100% branch length shared (purple). UniFrac score = 0.
B. Related sequence sets: seqs in red have relatives in blue. ~50% branch length shared. UniFrac score = 0.5.
C. Unrelated sequence sets: seqs in red have no close relatives in blue. 0% branch length shared. UniFrac score = 1.

PCoA example: **Unifrac** for microbial genetic data



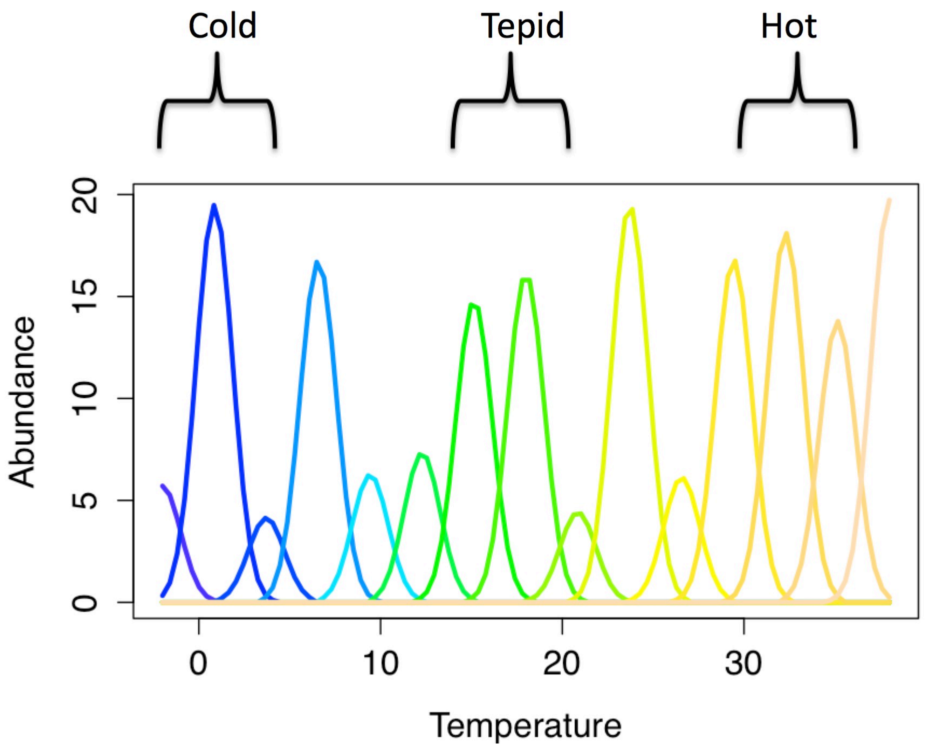
Cultured isolates are similar to each other,
And to uncultured sea ice communities

Other uncultured microbial communities are distinct

Non-metric multidimensional scaling (NMDS)

PCoA is flexible because it can use any dissimilarity index

But it still assumes that community **dissimilarity increases linearly** with distance along an underlying gradient



- Imagine we take samples at different temperatures
- As different in temperature increases, **dissimilarity saturates at 1**
- But if we want to reconstruct this from community data, the 'tepid' community needs to be closer to the 'cold' community than the 'hot' community

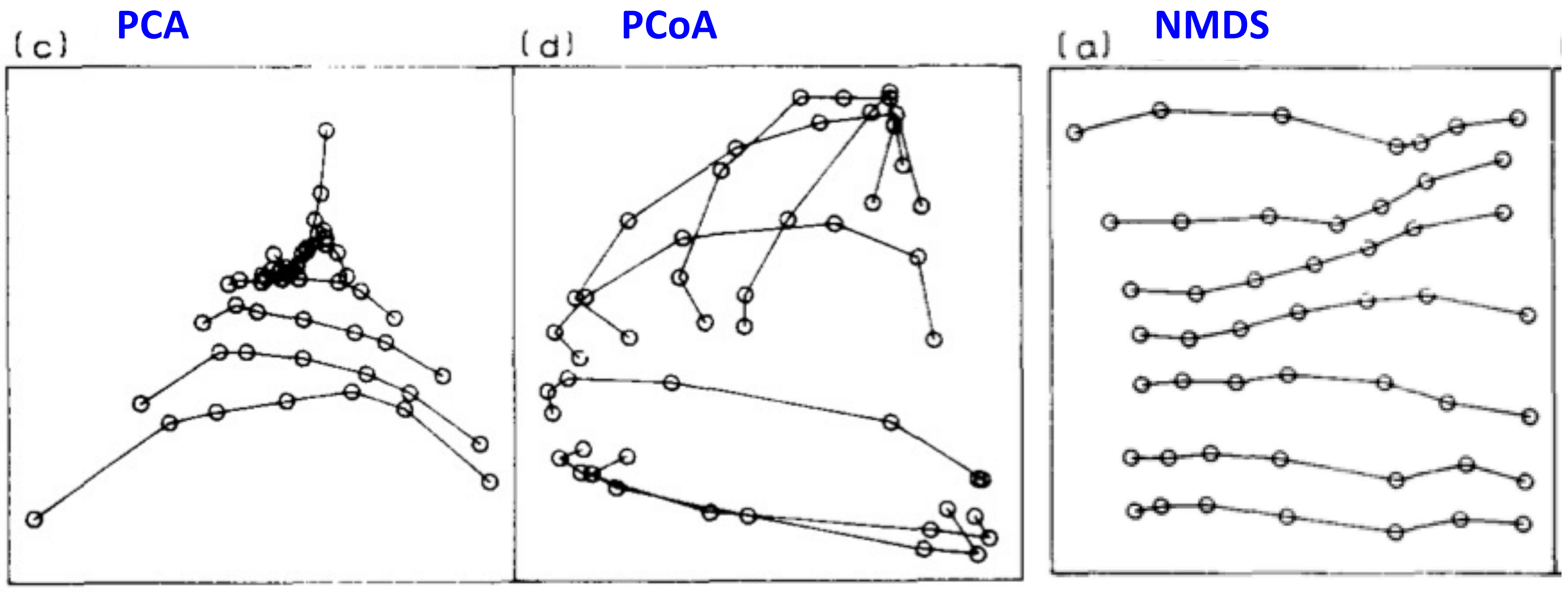
Non-metric multidimensional scaling (NMDS)

NMDS also takes a dissimilarity matrix and represents it in low-dimensional space

But it only assumes that the **ranking of distances** is correlated with the **ranking of dissimilarities**

Captures long gradients better

From a simulation of composition on a 2D grid, e.g. temperature and nitrogen (Minchin 1987):



Non-metric multidimensional scaling (NMDS)

Not an eigenanalysis method: uses an iterative algorithm

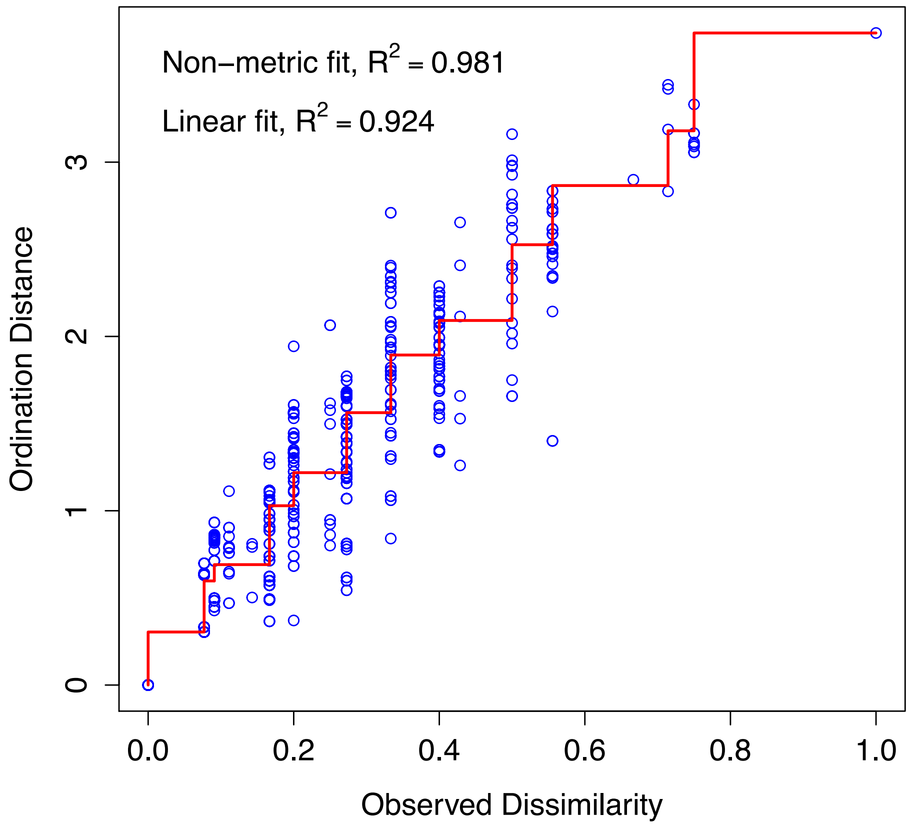
1) You pick how many dimensions (usually 2-4).

2) The samples are placed in an initial configuration (often from PCoA)

1) You pick how many dimensions (usually 2-4).

1) The samples are placed in an initial configuration (often from PCoA)

2) Compare the observed dissimilarities to the ordination distances, non-parametrically:



Relationship is **nonlinear** but **monotonic**

Calculate the spread around this fit: $Stress = \sqrt{\frac{\sum_{h,i}\left(d_{hi} - \hat{d}_{hi}\right)^2}{\sum_{h,i} d_{hi}^2}} = \sqrt{1 - R^2}$

4) Adjust the configuration of the samples in the ordination to reduce stress

5) Repeat until stress doesn’t decrease any more

metaMDS() will do this, starting from **many initial configurations**, to see if they converge on the same ordination

```r
ord = metaMDS(species.data.use, dist = "bray", trymax = 20)
```

```
ord

##
## Call:
## metaMDS(comm = species.data.use, distance = "bray", trymax = 20)
##
## global Multidimensional Scaling using monoMDS
##
## Data:     species.data.use
## Distance: bray
##
## Dimensions: 2
## Stress:     0.1488
## Stress type 1, weak ties
## Two convergent solutions found after 15 tries
## Scaling: centring, PC rotation, halfchange scaling
## Species: expanded scores based on 'species.data.use'
```

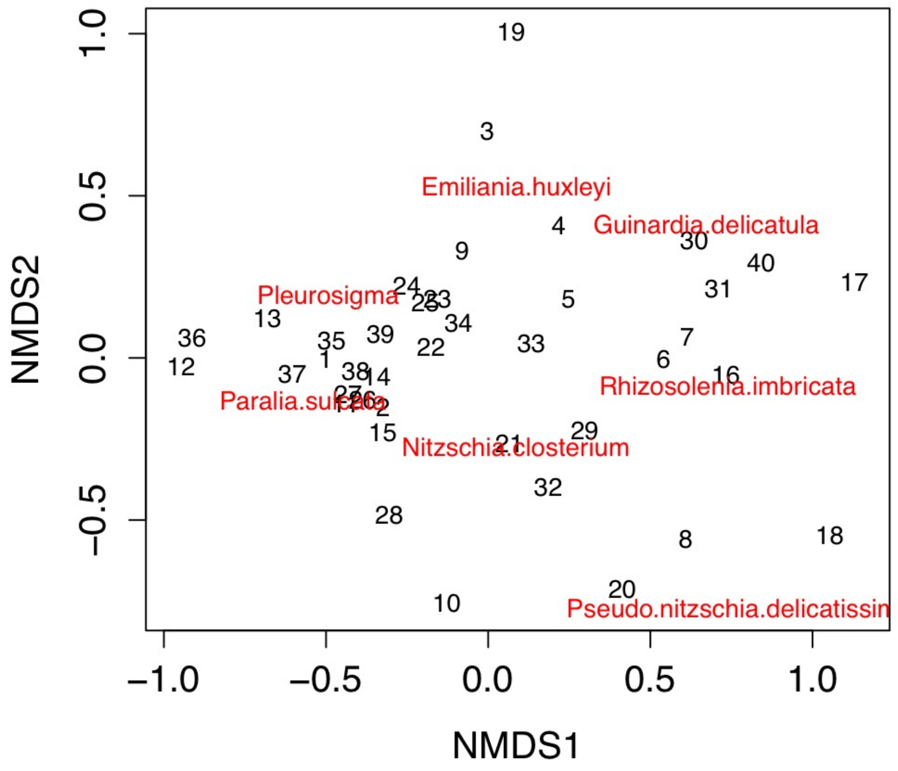
How well does it ‘fit’?

Stress > 0.3 considered very bad; **stress < 0.1** considered very good

‘weak ties’: pairs of samples with the same dissimilarity, e.g. sharing no species, are allowed to have different ordination distances

- Important for long gradients

```r
plot(ord, type = 'text')
```

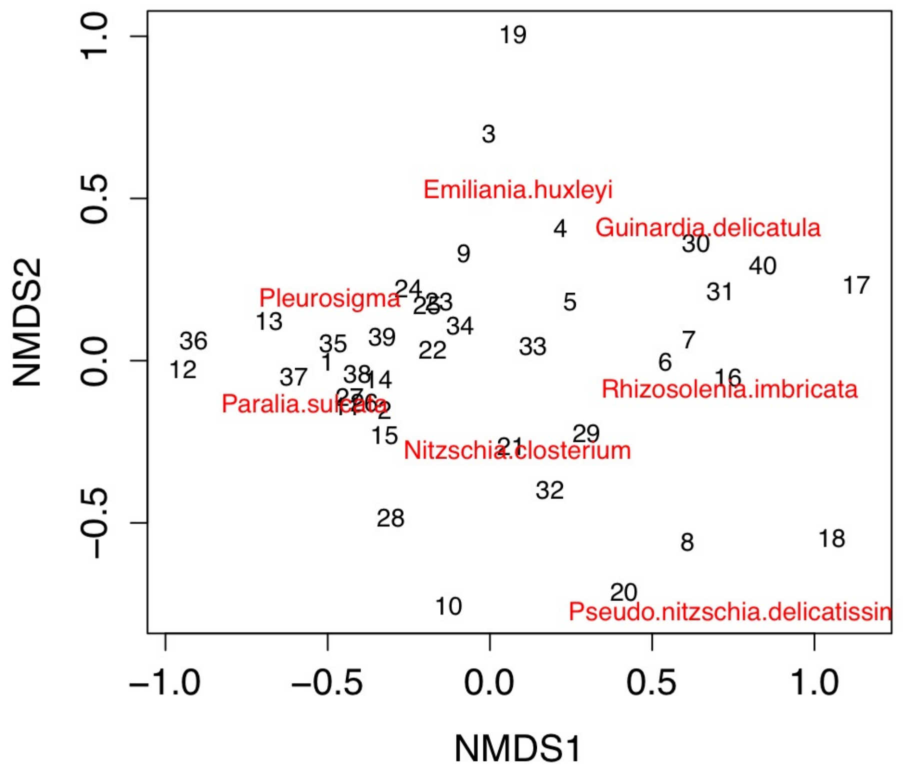


- The sample scores are what the algorithm is configuring
- The species scores are weighted averages

Note: **NMDS axes don’t mean anything**, can be rotated without changing the results

metaMDS rotates using a PCA on the scores

Scales the axes as **‘half-change’** – one unit change means a halving of similarity



Is this ordination any good?

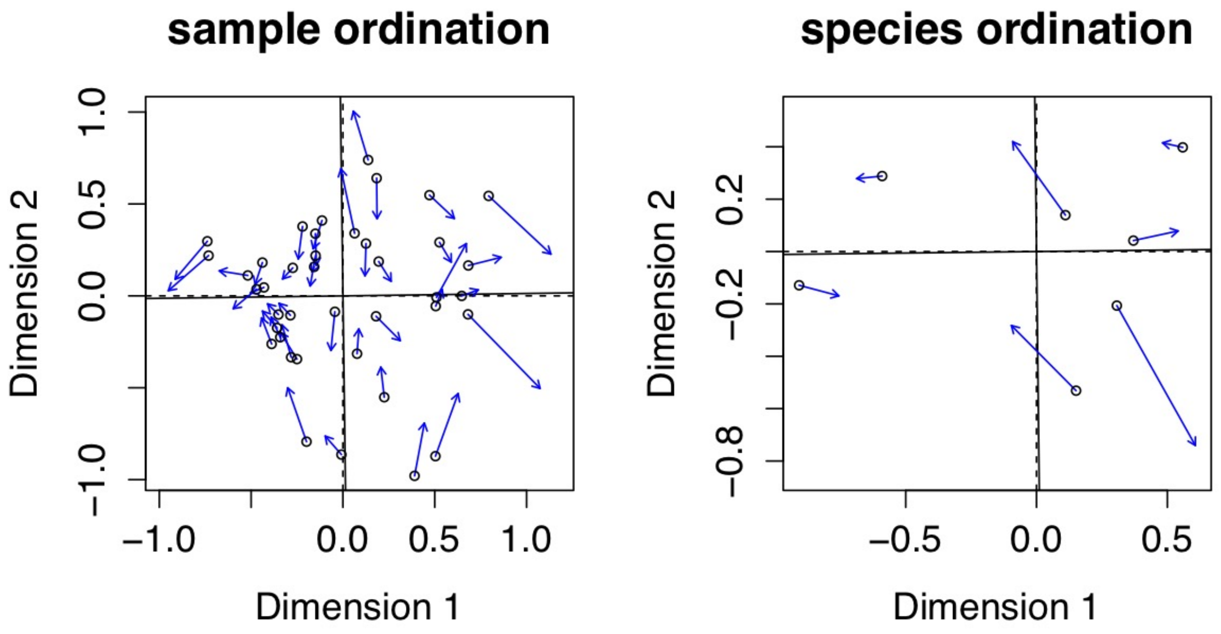
Downside of NMDS is: no % variation explained, no orthogonal axes of variation

Mostly used for visualization because simulation shows it does a better job

Can compare to another method, e.g. PCoA, see if the results are similar

```r
par(mfrow = c(1,2))
proc = procrustes(ord, pcoa)
plot(proc, main = 'sample ordination')

proc = procrustes(ord$species, summary(pcoa)$species[,1:2])
plot(proc, main = 'species ordination')
```



What is ordination good for

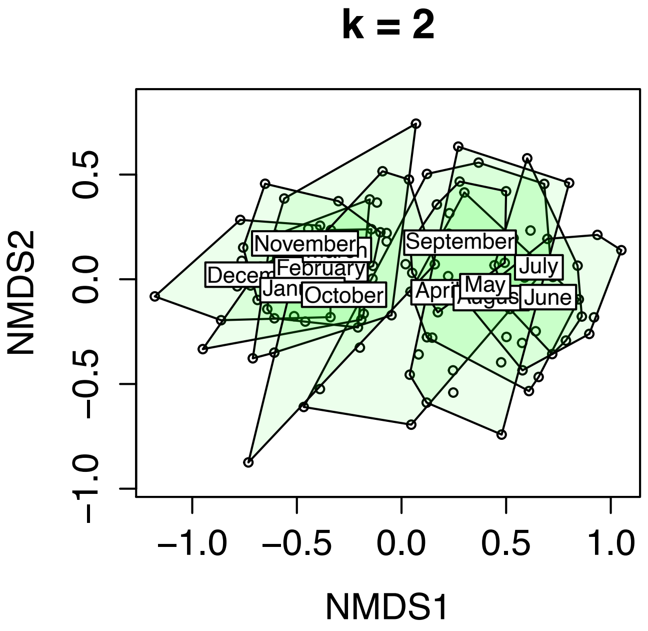
Can see if groups of data have distinct composition

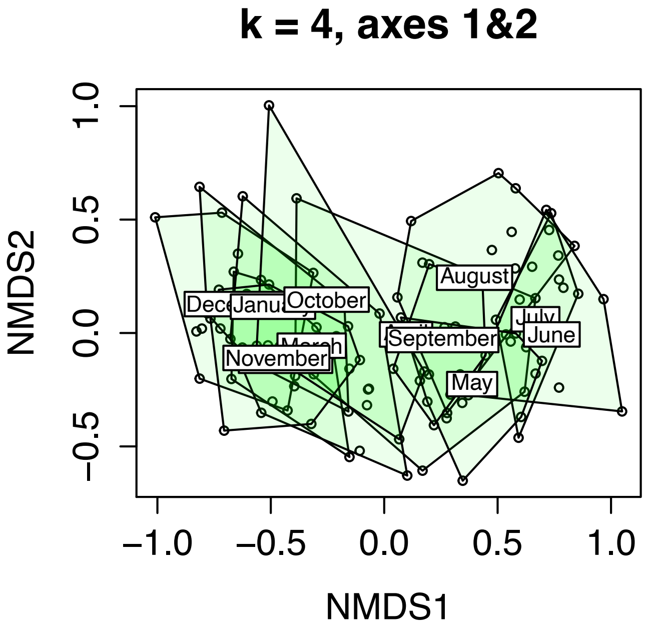
- Use English Channel data, 35 species, 120 samples (10 years)
- Look for seasonal signal, also taxonomic differences

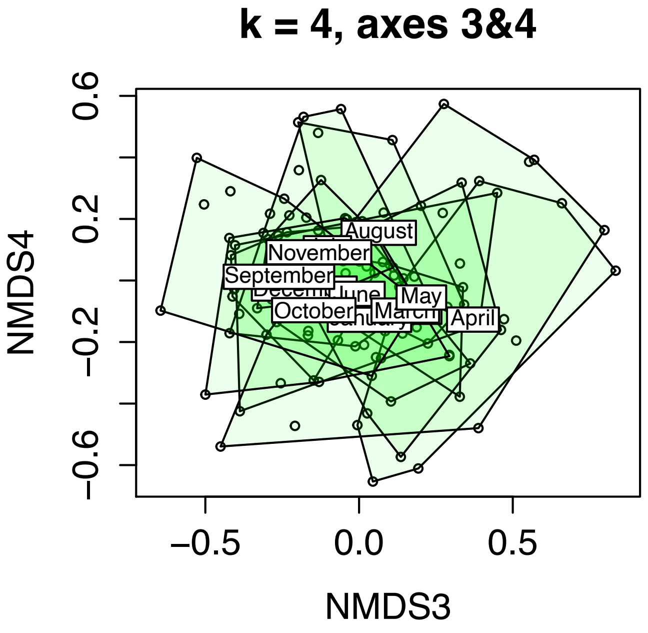
Try NMDS with 2, 3, 4 dimensions

- Got stresses of 0.2, 0.15, 0.12
- Does it matter for what I care about? Let’s compare.

```r
par(mfrow = c(1,3))
plot(ord, display = "sites", main = 'k = 2')
ordihull(ord, month.use, label = TRUE, col = 'green', border = 'black', alpha
= 20, cex = 0.6, draw = 'polygon')
plot(ord4, display = "sites", main = 'k = 4, axes 1&2')
ordihull(ord4, month.use, label = TRUE, col = 'green', border = 'black', alpha
 = 20, cex = 0.6, draw = 'polygon')
plot(ord4, display = "sites", main = 'k = 4, axes 3&4', choices = c(3,4))
ordihull(ord4, month.use, label = TRUE, col = 'green', border = 'black', alpha
 = 20, cex = 0.6, draw = 'polygon', choices = c(3,4))
```





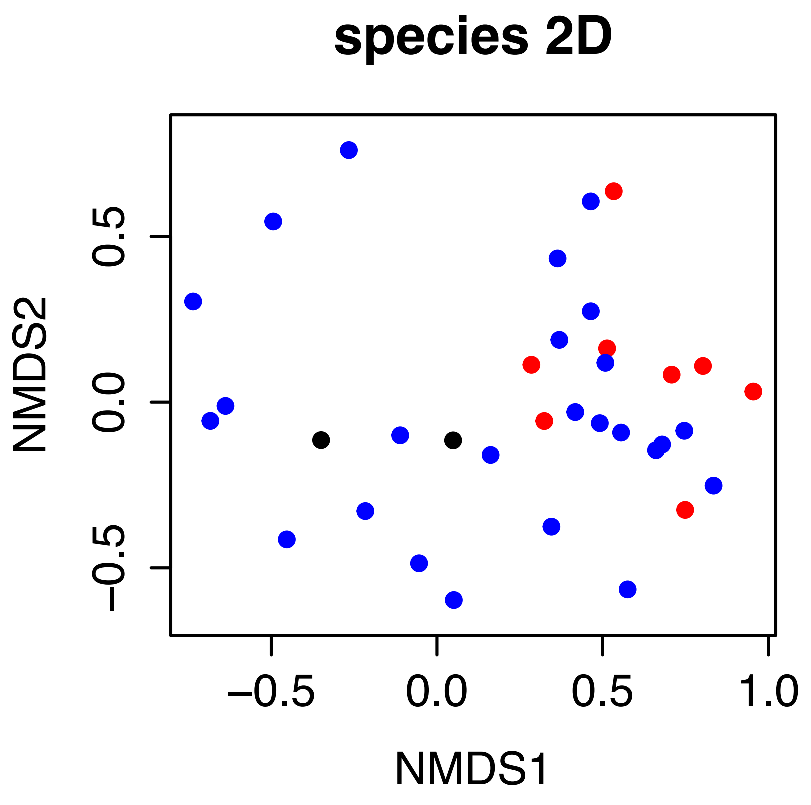


Note: can use other methods to directly ask how much variation by month

```r
plot(ord, display = "species", type = "n", main = 'species 2D')
points(ord, display = "species", col = c('blue', 'black', 'red')[taxa], pch =
19)
```

Species scores are weighted averages of the ‘site’ scores

Like ‘optima’, but really just the center of mass



Diatoms
Dinoflagellates