Curso ecología de comunidades en R - clase 3

Stephanie Hereira Pacheco

Contents

Análisis multivariados en datos ecológicos

En este capítulo vamos a explorar las diferentes técnicas de análisis multivariado para explorar nuestros datos ecológicos o biológicos.

Matrices de (dis)similitud

Como lo vimos en el módulo teórico las matrices de dis(similitud) son de gran utilidad para muchos análisis en ecología de comunidades. Así que primero veamos como obtener nuestras matrices.
R posee una función predeterminada para obtenerla: **dist()**. Inicialmente llamaré al paquete vegan para cargar la data de **varespec**, esta data nos presenta 44 especies de pastos de liquenes (lichen pastures), los nombres de las columnas son nombres formados de los nombres cientificos de estas especies y los valores que nos presenta son de cobertura estimada:

```
library(vegan)
data("varespec")
head(varespec)
```

```
##    Callvulg Empenigr Rhodtome Vaccmyrt Vaccviti Pinusylv Descflex Betupube
## 18     0.55    11.13     0.00     0.00    17.80     0.07     0.00        0
## 15     0.67     0.17     0.00     0.35    12.13     0.12     0.00        0
## 24     0.10     1.55     0.00     0.00    13.47     0.25     0.00        0
## 27     0.00    15.13     2.42     5.92    15.97     0.00     3.70        0
## 23     0.00    12.68     0.00     0.00    23.73     0.03     0.00        0
## 19     0.00     8.92     0.00     2.42    10.28     0.12     0.02        0
##    Vacculig Diphcomp Dicrsp Dicrfusc Dicrpoly Hylosple Pleuschr Polypili
## 18     1.60     2.07   0.00     1.62     0.00      0.0     4.67     0.02
## 15     0.00     0.00   0.33    10.92     0.02      0.0    37.75     0.02
## 24     0.00     0.00  23.43     0.00     1.68      0.0    32.92     0.00
## 27     1.12     0.00   0.00     3.63     0.00      6.7    58.07     0.00
## 23     0.00     0.00   0.00     3.42     0.02      0.0    19.42     0.02
```

```
## 19     0.00     0.00   0.00     0.32     0.02      0.0    21.03     0.02
##    Polyjuni Polycomm Pohlnuta Ptilcili Barbhatc Cladarbu Cladrang Cladstel
## 18     0.13     0.00     0.13     0.12     0.00    21.73    21.47     3.50
## 15     0.23     0.00     0.03     0.02     0.00    12.05     8.13     0.18
## 24     0.23     0.00     0.32     0.03     0.00     3.58     5.52     0.07
## 27     0.00     0.13     0.02     0.08     0.08     1.42     7.63     2.55
## 23     2.12     0.00     0.17     1.80     0.02     9.08     9.22     0.05
## 19     1.58     0.18     0.07     0.27     0.02     7.23     4.95    22.08
##    Cladunci Cladcocc Cladcorn Cladgrac Cladfimb Cladcris Cladchlo Cladbotr
## 18     0.30     0.18     0.23     0.25     0.25     0.23     0.00     0.00
## 15     2.65     0.13     0.18     0.23     0.25     1.23     0.00     0.00
## 24     8.93     0.00     0.20     0.48     0.00     0.07     0.10     0.02
## 27     0.15     0.00     0.38     0.12     0.10     0.03     0.00     0.02
## 23     0.73     0.08     1.42     0.50     0.17     1.78     0.05     0.05
## 19     0.25     0.10     0.25     0.18     0.10     0.12     0.05     0.02
##    Cladamau Cladsp Cetreric Cetrisla Flavniva Nepharct Stersp Peltapht Icmaeric
## 18     0.08   0.02     0.02     0.00     0.12     0.02   0.62     0.02        0
## 15     0.00   0.00     0.15     0.03     0.00     0.00   0.85     0.00        0
## 24     0.00   0.00     0.78     0.12     0.00     0.00   0.03     0.00        0
## 27     0.00   0.02     0.00     0.00     0.00     0.00   0.00     0.07        0
## 23     0.00   0.00     0.00     0.00     0.02     0.00   1.58     0.33        0
## 19     0.00   0.00     0.00     0.00     0.02     0.00   0.28     0.00        0
##    Cladcerv Claddefo Cladphyl
## 18        0     0.25        0
## 15        0     1.00        0
## 24        0     0.33        0
## 27        0     0.15        0
## 23        0     1.97        0
## 19        0     0.37        0
```

Ahora sí apliquemos la función **dist()** para obtener una matriz euclidiana:

```
eucl_dist<- dist(varespec, method = "euclidean")
eucl_dist
```

```
##          18       15       24       27       23       19       22
## 15  40.37368
## 24  46.27477 28.35874
## 27  59.87344 30.49724 39.77855
## 23  24.54328 26.60815 33.21226 41.78866
## 19  34.27126 31.61316 36.60304 44.23517 27.04775
## 22  49.33512 30.86003 45.82774 48.59480 38.64358 43.91427
## 16  35.98314 27.72034 40.70025 48.96734 31.22161 35.84254 19.69141
## 28  76.28558 42.45458 50.01474 25.42484 59.74367 58.06714 60.04516
## 13  29.72944 47.05861 53.53636 67.42766 42.33547 40.18469 54.19887
## 14  35.94866 38.79924 42.90444 61.39094 35.65736 38.80321 41.17716
## 20  22.06805 27.36560 31.18323 46.93335 14.46833 25.17124 39.18976
## 25  40.47882 21.53302 20.39499 37.83977 25.72712 30.43738 32.24576
## 7   28.69034 57.88555 64.55017 77.62356 50.38028 56.44443 67.09804
## 5   43.55582 65.97081 70.81557 83.18682 60.33552 64.61900 74.62079
## 6   27.15780 57.20421 63.97989 77.36307 49.50250 51.52029 66.54232
## 3   56.62383 71.76913 74.05306 84.19067 65.99354 47.35542 76.83357
## 4   33.67680 50.83691 54.76934 69.60520 45.05405 31.23740 58.31114
```

```
## 2   75.42232  87.14308  87.93849  94.99337  80.91850  60.42866  90.50360
## 9   85.79913  92.53898  92.84378  99.39486  86.92633  65.93488  95.29205
## 12  63.00428  73.25901  74.22795  81.14001  65.82505  45.60175  77.42516
## 10  84.67200  93.84384  94.17626 100.62635  87.63274  65.95702  96.11021
## 11  36.56964  44.70114  49.65825  58.21754  41.58233  28.70565  56.55456
## 21  35.04495  49.20403  48.88209  59.78971  29.57417  32.95802  51.05770
##           16        28        13        14        20        25         7
## 15
## 24
## 27
## 23
## 19
## 22
## 16
## 28  62.35616
## 13  40.42538  79.77486
## 14  33.73697  72.94486  41.24095
## 20  27.33743  61.86287  36.41482  28.13798
## 25  27.68331  50.77494  49.05372  37.54970  25.77439
## 7   51.94699  91.36889  36.54075  53.07489  45.61214  61.06352
## 5   57.88412  95.79736  46.06488  64.89809  53.94732  69.19542  27.47562
## 6   52.39714  91.37006  33.08080  50.18062  45.40277  60.50051  15.63836
## 3   66.10597  96.59351  52.62834  68.53040  61.16084  71.35649  63.04649
## 4   44.97883  81.67954  29.97348  44.39767  37.63058  50.55367  43.49105
## 2   82.58188 107.13462  71.88607  84.86208  77.91306  85.62402  84.74813
## 9   91.75562 109.88646  83.44949  91.20159  85.79285  90.75482  99.86172
## 12  70.79713  94.42697  62.31453  71.74607  64.26202  71.46658  77.04733
## 10  90.82292 111.91465  81.64771  90.61187  85.77934  91.41398  97.21518
## 11  45.30842  71.82501  36.11344  51.32046  37.83895  48.36004  47.19558
## 21  44.81175  74.35370  48.95056  44.52420  30.49493  43.42614  59.54120
##           5         6         3         4         2         9        12
## 15
## 24
## 27
## 23
## 19
## 22
## 16
## 28
## 13
## 14
## 20
## 25
## 7
## 5
## 6   34.59352
## 3   63.76947  54.71236
## 4   49.64549  36.75185  31.91485
## 2   84.95918  75.75794  22.87043  51.71322
## 9  103.64324  89.82991  42.68294  63.99890  26.40679
## 12  80.75521  67.71033  23.25221  43.02526  18.25976  25.62686
## 10 100.35825  87.29694  37.76951  61.62167  18.76653  12.87040  22.96952
## 11  48.94759  41.11796  31.22807  25.14522  48.71641  61.41074  39.12502
## 21  67.68046  56.84553  60.17585  44.72281  71.96865  76.91780  57.74893
```

```
##           10        11
## 15
## 24
## 27
## 23
## 19
## 22
## 16
## 28
## 13
## 14
## 20
## 25
## 7
## 5
## 6
## 3
## 4
## 2
## 9
## 12
## 10
## 11  60.66839
## 21  77.20776  46.18480
```

Dentro de los métodos que podemos escoger con esta función encontramos: "euclidean", "maximum", "manhattan", "canberra", "binary" o "minkowski". Como vemos la mayoria de estas distancias no son tan aplicadas en ecologia sino en otros tipos de datos como los económicos.

El paquete **vegan()** tiene también su función para obtener las matrices de distancia y es **vegdist()**, siguiendo el ejemplo anterior:

```
eucl_dist<- vegdist(varespec, method = "euclidean")
eucl_dist
```

```
##           18        15        24        27        23        19        22
## 15  40.37368
## 24  46.27477  28.35874
## 27  59.87344  30.49724  39.77855
## 23  24.54328  26.60815  33.21226  41.78866
## 19  34.27126  31.61316  36.60304  44.23517  27.04775
## 22  49.33512  30.86003  45.82774  48.59480  38.64358  43.91427
## 16  35.98314  27.72034  40.70025  48.96734  31.22161  35.84254  19.69141
## 28  76.28558  42.45458  50.01474  25.42484  59.74367  58.06714  60.04516
## 13  29.72944  47.05861  53.53636  67.42766  42.33547  40.18469  54.19887
## 14  35.94866  38.79924  42.90444  61.39094  35.65736  38.80321  41.17716
## 20  22.06805  27.36560  31.18323  46.93335  14.46833  25.17124  39.18976
## 25  40.47882  21.53302  20.39499  37.83977  25.72712  30.43738  32.24576
## 7   28.69034  57.88555  64.55017  77.62356  50.38028  56.44443  67.09804
## 5   43.55582  65.97081  70.81557  83.18682  60.33552  64.61900  74.62079
## 6   27.15780  57.20421  63.97989  77.36307  49.50250  51.52029  66.54232
## 3   56.62383  71.76913  74.05306  84.19067  65.99354  47.35542  76.83357
## 4   33.67680  50.83691  54.76934  69.60520  45.05405  31.23740  58.31114
## 2   75.42232  87.14308  87.93849  94.99337  80.91850  60.42866  90.50360
## 9   85.79913  92.53898  92.84378  99.39486  86.92633  65.93488  95.29205
## 12  63.00428  73.25901  74.22795  81.14001  65.82505  45.60175  77.42516
```

```
## 10  84.67200  93.84384  94.17626 100.62635  87.63274  65.95702  96.11021
## 11  36.56964  44.70114  49.65825  58.21754  41.58233  28.70565  56.55456
## 21  35.04495  49.20403  48.88209  59.78971  29.57417  32.95802  51.05770
##           16        28        13        14        20        25         7
## 15
## 24
## 27
## 23
## 19
## 22
## 16
## 28  62.35616
## 13  40.42538  79.77486
## 14  33.73697  72.94486  41.24095
## 20  27.33743  61.86287  36.41482  28.13798
## 25  27.68331  50.77494  49.05372  37.54970  25.77439
## 7   51.94699  91.36889  36.54075  53.07489  45.61214  61.06352
## 5   57.88412  95.79736  46.06488  64.89809  53.94732  69.19542  27.47562
## 6   52.39714  91.37006  33.08080  50.18062  45.40277  60.50051  15.63836
## 3   66.10597  96.59351  52.62834  68.53040  61.16084  71.35649  63.04649
## 4   44.97883  81.67954  29.97348  44.39767  37.63058  50.55367  43.49105
## 2   82.58188 107.13462  71.88607  84.86208  77.91306  85.62402  84.74813
## 9   91.75562 109.88646  83.44949  91.20159  85.79285  90.75482  99.86172
## 12  70.79713  94.42697  62.31453  71.74607  64.26202  71.46658  77.04733
## 10  90.82292 111.91465  81.64771  90.61187  85.77934  91.41398  97.21518
## 11  45.30842  71.82501  36.11344  51.32046  37.83895  48.36004  47.19558
## 21  44.81175  74.35370  48.95056  44.52420  30.49493  43.42614  59.54120
##           5         6         3         4         2         9        12
## 15
## 24
## 27
## 23
## 19
## 22
## 16
## 28
## 13
## 14
## 20
## 25
## 7
## 5
## 6   34.59352
## 3   63.76947  54.71236
## 4   49.64549  36.75185  31.91485
## 2   84.95918  75.75794  22.87043  51.71322
## 9  103.64324  89.82991  42.68294  63.99890  26.40679
## 12  80.75521  67.71033  23.25221  43.02526  18.25976  25.62686
## 10 100.35825  87.29694  37.76951  61.62167  18.76653  12.87040  22.96952
## 11  48.94759  41.11796  31.22807  25.14522  48.71641  61.41074  39.12502
## 21  67.68046  56.84553  60.17585  44.72281  71.96865  76.91780  57.74893
##           10        11
## 15
## 24
```

```
## 27
## 23
## 19
## 22
## 16
## 28
## 13
## 14
## 20
## 25
## 7
## 5
## 6
## 3
## 4
## 2
## 9
## 12
## 10
## 11  60.66839
## 21  77.20776  46.18480
```

La ventaja de esta función es que posee métodos como “manhattan”,“canberra”, “clark”, “bray”, “kulczynski”, “jaccard”, “gower”, “altGower”, “morisita”, “horn”, “mountford”, “raup”, “binomial”, “chao”, “cao”, “mahalanobis”, “chisq” ó “chord” que suelen ser distancias más conocidas y aplicables en ecología de comunidades.

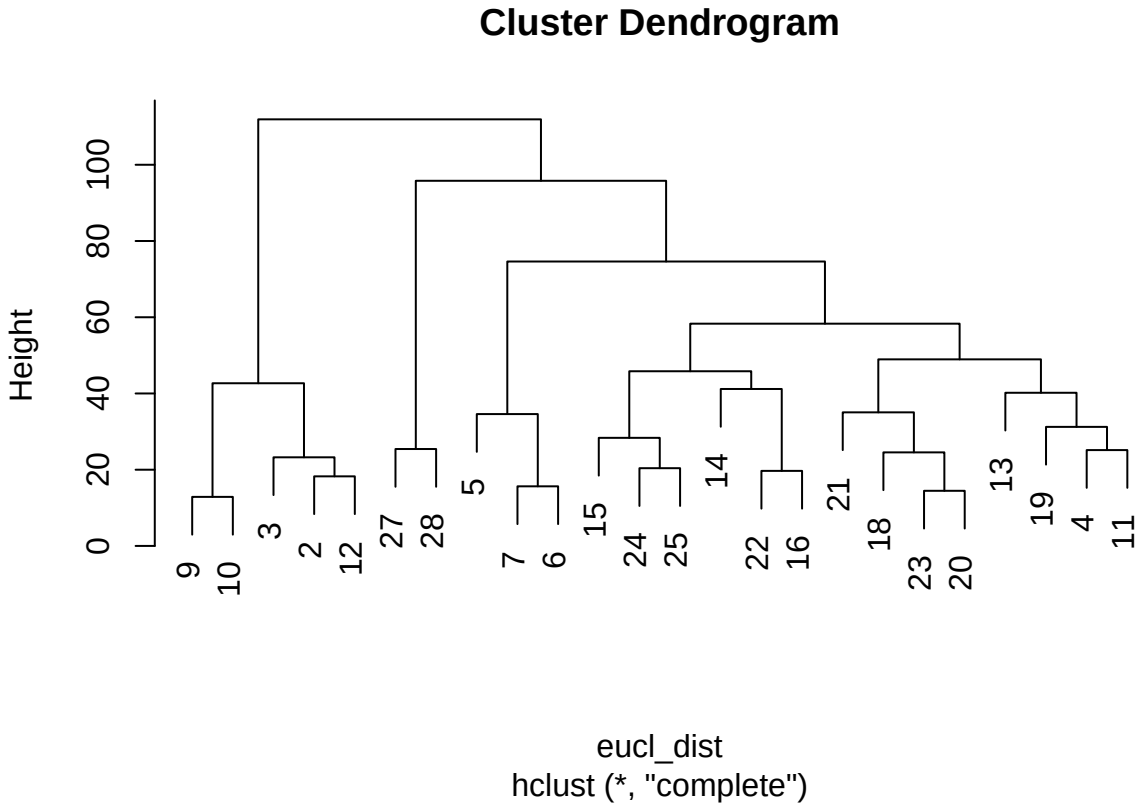
Análisis de agrupamiento

El análisis de clusters o agrupamiento se aplica sobre una matriz de distancia previamente obtenida. Como vimos en el módulo teórico hay varios métodos para realizar este clustering, tales como: simple (single), completo (complete) y promedio (average). La función **hclust()** de R nos permite explorar todas estas formas y otras más como ward.D2, mendian y centroid, comúnmente usados en análisis de datos biológicos. Apliquemosla sobre nuestra matriz previament generada:

```
clusters<-hclust(eucl_dist, method = "complete")
```

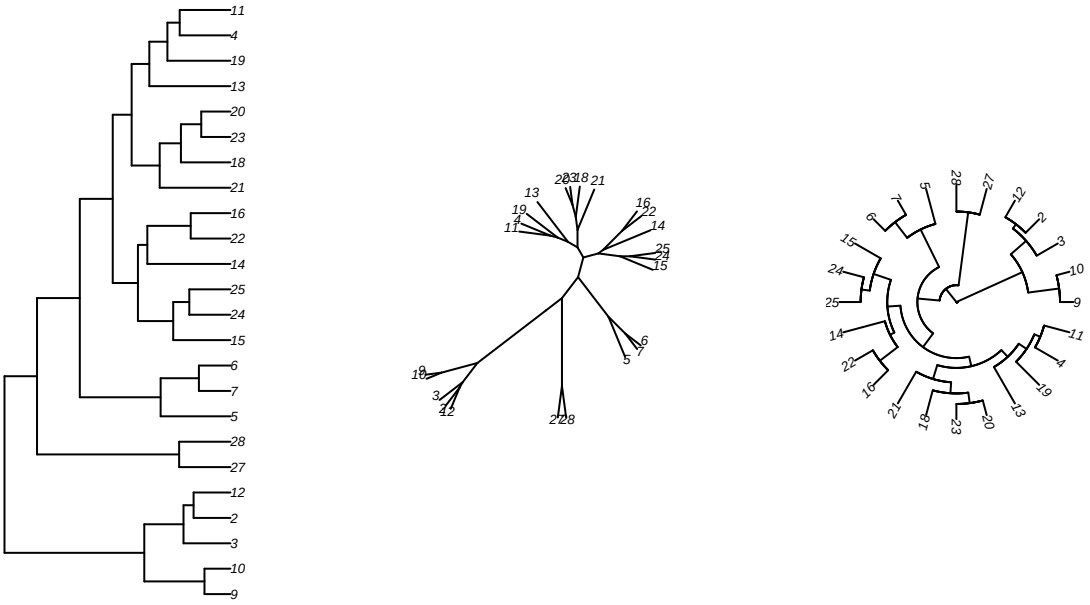
Para visualizar nuestros clusters o mejor llamado dendograma usamos la función **plot()**:

```
plot(clusters)
```



Con el paquete **ape()** podemos formatear de diferentes formas nuestro dendograma:

```
library("ape")
par( mfrow= c(1,3) )
plot(as.phylo(clusters), cex = 0.6)
plot(as.phylo(clusters), cex = 0.6, type = "unrooted")
plot(as.phylo(clusters), type = "fan")
```



Medida de la distorsión

Si nos interesa evaluar la bondad o qué tan bien nuestro dendograma explica nuestros datos entonces debemos medirlo a través de una matriz cofenética. En primer lugar se calcula la matriz cofenética con la función **cophenetic()** , que resulta de obtener una nueva matriz a partir del dendrograma y luego se calcula la correlación de Pearson entre la matriz cofenética y la matriz original.

```
mat.clusters<- cophenetic(clusters)
cor(mat.clusters, eucl_dist, method = "pearson")
```

```
## [1] 0.8052116
```

En este caso el coeficiente nos dio 0.8 lo cual está bien, entre más cercano a 1 nos dice qué tan bien correlacionados están y que mi dendograma explica en buena medida mis datos.

Análisis de ordenación canónica

Hay diversos tipos métodos de ordenación con los que podemos explorar nuestros datos ecológicos. R presenta diversas funciones para obtenerlas:

Método de Ordenación	Función	Paquete
PCA	princomp	stats
	prcomp	stats
	PCA	FactoMineR
	rda	vegan
PCoA	cmdscale	stats
	pcoa	ape
	wcmdscale	vegan
NMDS	**metaMDS**	vegan
	isoMDS	MASS
CA	CA	FactoMineR
	corresp	MASS
	cca	vegan

PCA

La función en vegan para PCA es rda(), que técnicamente significa Análisis de redundancia. No entraré en RDA (que al fin no tratamos previamente), pero cuando se ejecuta esta función sobre una matriz de especies sin ninguna variable ambiental, hace un PCA. En este ejemplo trabajaremos con otra data de ejemplo llamada **dune**, que son datos de vegetación de praderas de dunas, dunas, tienen valores de clase de cobertura de 30 especies en 20 sitios:

```
data("dune")
data("dune.env")
```

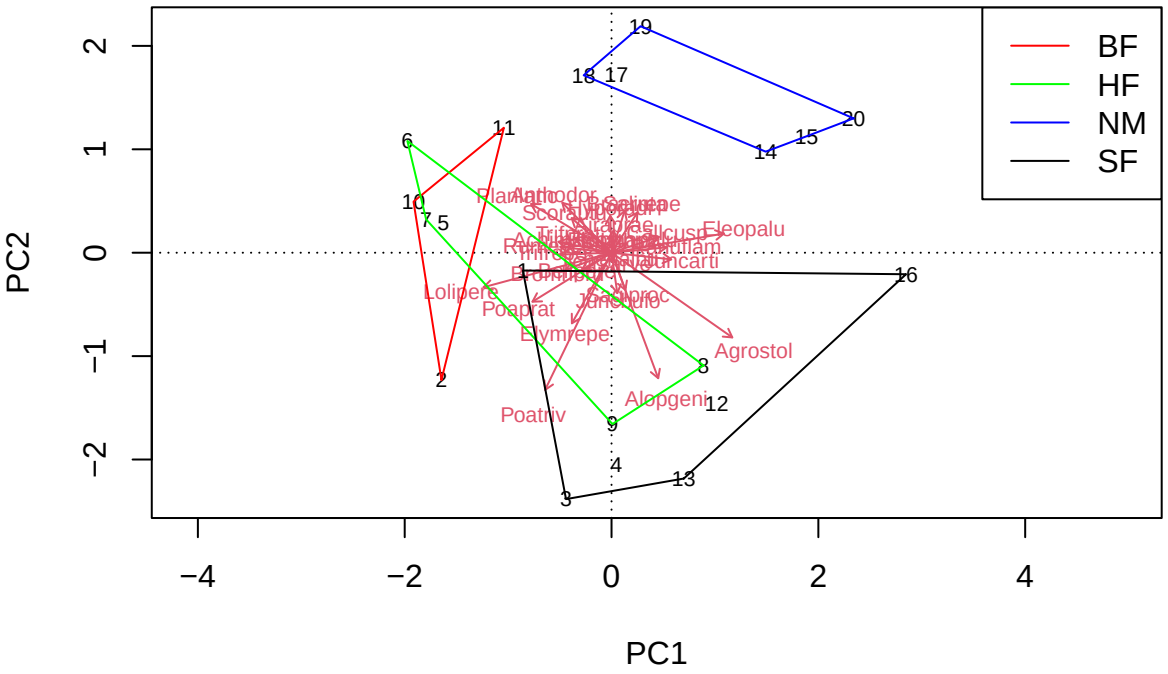
Ahora corremos el PCA:

```
dune_pca <- rda(dune)
sum_dune_pca <- summary(dune_pca)
head(sum_dune_pca)
```

```
##
## Call:
## rda(X = dune)
##
## Partitioning of variance:
##               Inertia Proportion
## Total           84.12          1
## Unconstrained   84.12          1
##
## Eigenvalues, and their contribution to the variance
##
## Importance of components:
##                             PC1     PC2     PC3     PC4    PC5     PC6     PC7
## Eigenvalue            24.7953 18.1466 7.62913 7.15277 5.6950 4.33331 3.19936
## Proportion Explained   0.2947  0.2157 0.09069 0.08503 0.0677 0.05151 0.03803
## Cumulative Proportion  0.2947  0.5105 0.60115 0.68618 0.7539 0.80539 0.84342
##                             PC8    PC9    PC10    PC11    PC12    PC13     PC14
## Eigenvalue            2.78186 2.4820 1.85377 1.74712 1.31358 0.99051 0.637794
## Proportion Explained  0.03307 0.0295 0.02204 0.02077 0.01561 0.01177 0.007582
## Cumulative Proportion 0.87649 0.9060 0.92803 0.94880 0.96441 0.97619 0.983768
##                             PC15     PC16     PC17     PC18     PC19
## Eigenvalue            0.550827 0.350584 0.199556 0.148798 0.115753
## Proportion Explained  0.006548 0.004167 0.002372 0.001769 0.001376
## Cumulative Proportion 0.990316 0.994483 0.996855 0.998624 1.000000
##
## Scaling 2 for species and site scores
## * Species are scaled proportional to eigenvalues
## * Sites are unscaled: weighted dispersion equal on all dimensions
## * General scaling constant of scores:  6.322924
##
##
## Species scores
##
##               PC1     PC2       PC3      PC4      PC5      PC6
## Achimill -0.60379  0.1239  0.008464  0.15957  0.40871  0.12786
## Agrostol  1.37395 -0.9640  0.166905  0.26647 -0.08765  0.04737
## Airaprae  0.02342  0.2508 -0.194768 -0.32604  0.05574 -0.07962
## Alopgeni  0.53123 -1.4278 -0.505241 -0.04288 -0.44293  0.27857
## Anthodor -0.55914  0.5676 -0.476205  0.01578  0.34408 -0.13578
## Bellpere -0.33356 -0.1888  0.140638 -0.08418  0.12541  0.13477
## ....
##
##
## Site scores (weighted sums of species scores)
##
##           PC1     PC2     PC3     PC4     PC5      PC6
## 1    -0.85678 -0.1724  2.6079 -1.1296  0.4507 -2.49113
## 2    -1.64477 -1.2299  0.8867 -0.9859  2.0346  1.81057
## 3    -0.44010 -2.3827  0.9297 -0.4601 -1.0278 -0.05183
## 4     0.04795 -2.0463  1.2737 -0.9742 -0.6421 -0.72074
## 5    -1.62445  0.2900 -1.5927  1.5398  1.8601 -2.21191
## 6    -1.97427  1.0802 -1.1501  3.3534 -1.5203  0.03127
## ....
```

Para visualizarlo usamos la función **biplot()**

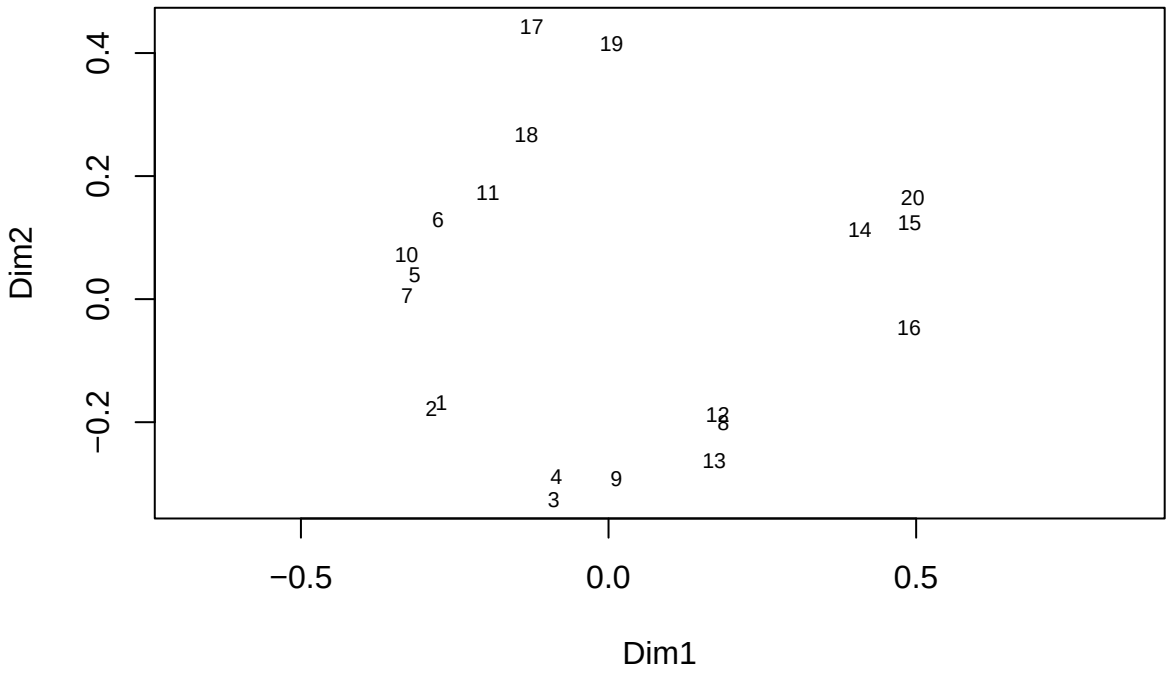
```
biplot(dune_pca)
ordihull(dune_pca, groups = dune.env$Management, col = c("red", "green", "blue", "black"))
man_names<- levels(dune.env$Management)
legend("topright",col = c("red", "green", "blue", "black"), lty = 1, legend = man_names )
```



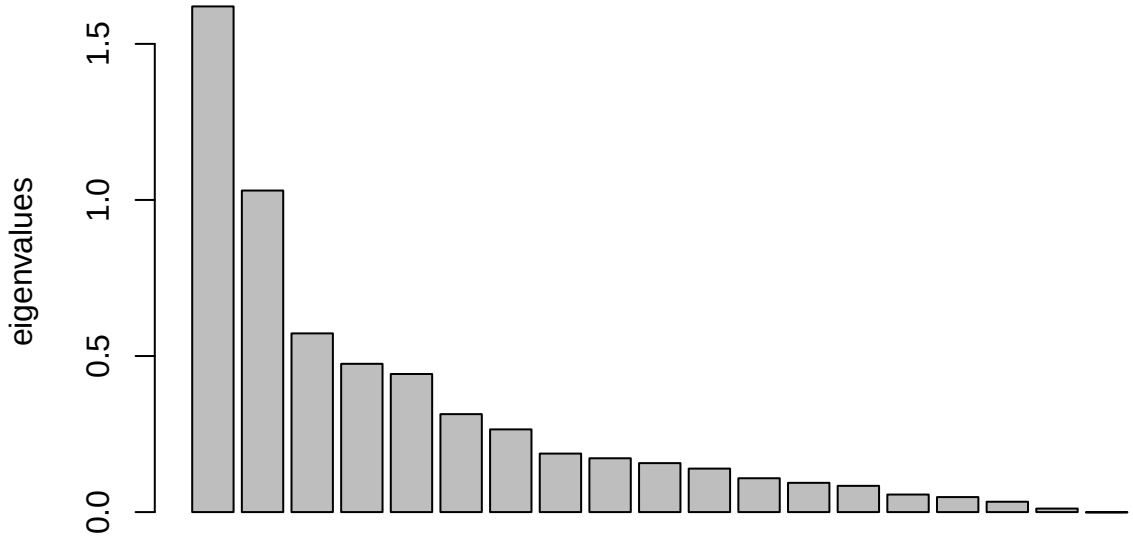
PCoA

Usaremos la misma matriz para calcular PCoA y dibujar el diagrama de ordenación. Recuerden que la ventaje de este método es poder uscar cualquier tipo de matriz de distancia que deseemos:

```
d <- vegdist(dune, method = "jaccard")
ord <- wcmdscale(d, eig = TRUE)
ordiplot(ord, display = 'sites', type = 'text')
```




```
barplot (ord$eig, las = 3, ylab = 'eigenvalues')
```



NMDS

El objetivo de NMDS es colapsar la información de múltiples dimensiones (p. ej., de múltiples comunidades, sitios, etc.) en solo unas pocas, para que puedan visualizarse e interpretarse.

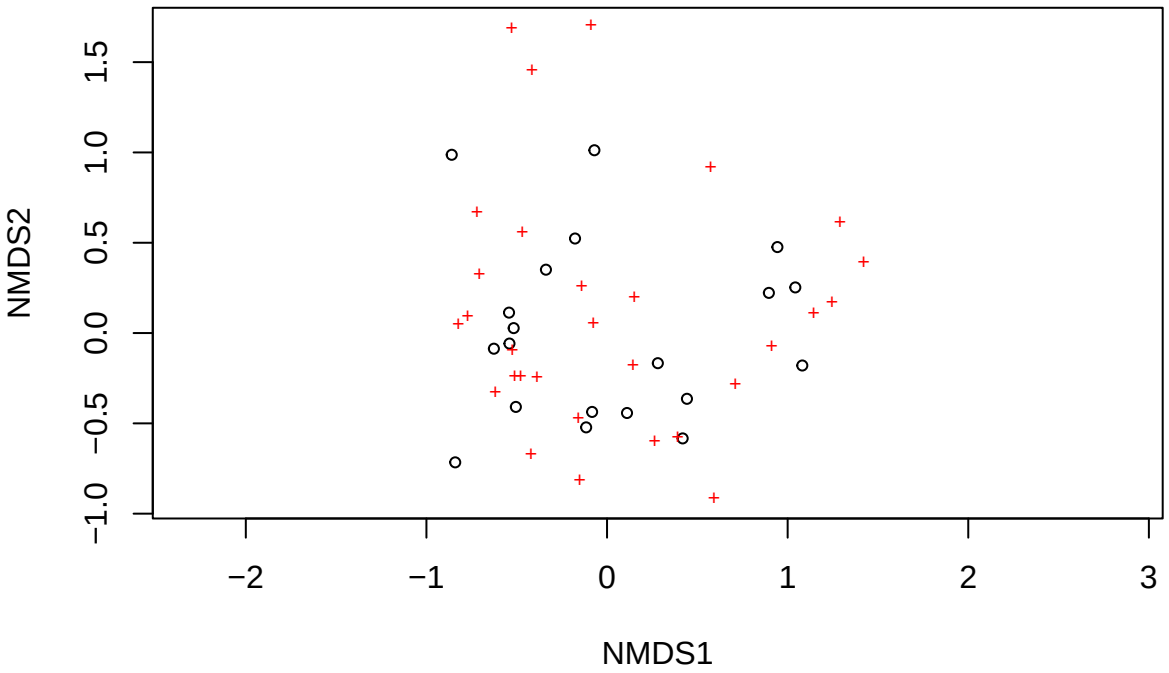
```
nmds <- metaMDS(dune, distance = "bray", k = 2)
```

```
## Run 0 stress 0.1192678
## Run 1 stress 0.1192679
## ... Procrustes: rmse 0.000197919  max resid 0.0006069313
## ... Similar to previous best
```

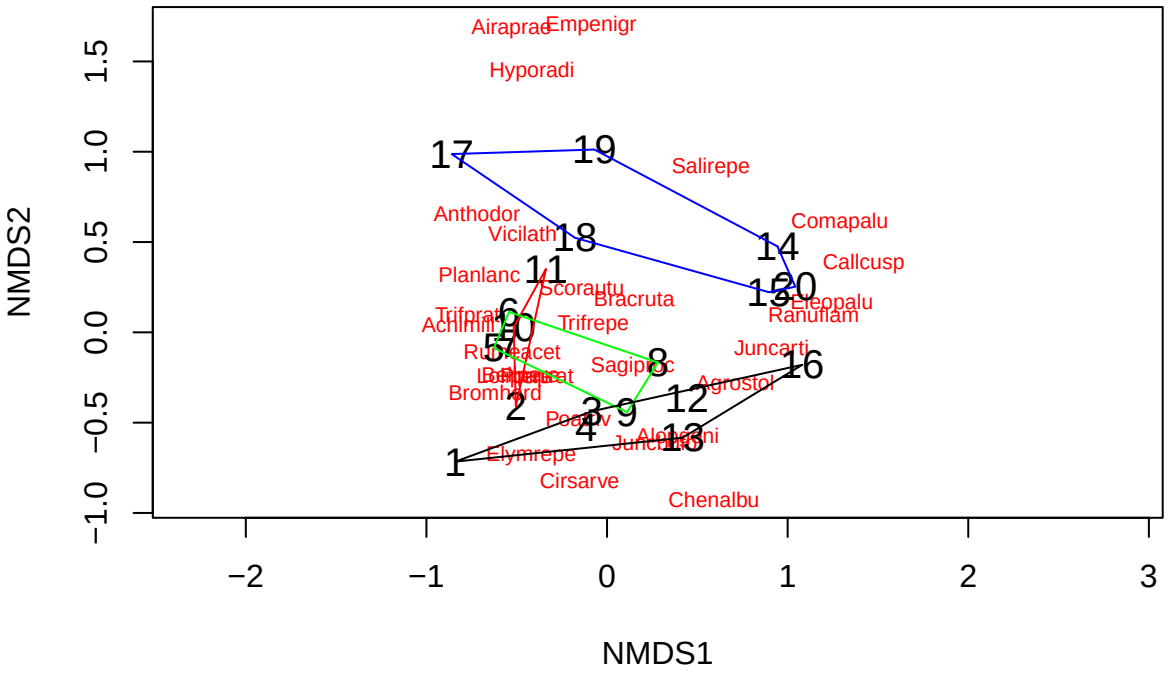
```
## Run 2 stress 0.1192678
## ... Procrustes: rmse 4.400121e-05  max resid 0.0001343421
## ... Similar to previous best
## Run 3 stress 0.1183186
## ... New best solution
## ... Procrustes: rmse 0.02026943  max resid 0.0649581
## Run 4 stress 0.1192678
## Run 5 stress 0.1183186
## ... Procrustes: rmse 1.258191e-05  max resid 3.83231e-05
## ... Similar to previous best
## Run 6 stress 0.1183186
## ... New best solution
## ... Procrustes: rmse 7.592599e-06  max resid 2.537232e-05
## ... Similar to previous best
## Run 7 stress 0.1192678
## Run 8 stress 0.1192678
## Run 9 stress 0.1192678
## Run 10 stress 0.1192678
## Run 11 stress 0.1939203
## Run 12 stress 0.1183186
## ... Procrustes: rmse 1.974282e-05  max resid 6.260737e-05
## ... Similar to previous best
## Run 13 stress 0.1192679
## Run 14 stress 0.1886532
## Run 15 stress 0.1183186
## ... Procrustes: rmse 4.51532e-06  max resid 1.555241e-05
## ... Similar to previous best
## Run 16 stress 0.1886532
## Run 17 stress 0.2035424
## Run 18 stress 0.1192678
## Run 19 stress 0.1192678
## Run 20 stress 0.1192678
## *** Solution reached
```

Visualizando:

```
plot(nmds)
```




```
ordiplot(nmds,type="n")
orditorp(nmds,display="species",col="red",air=0.01)
orditorp(nmds,display="sites",cex=1.25,air=0.01)
ordihull(nmds, groups = dune.env$Management, col = c("red", "green", "blue", "black"))
```



De esta manera visualizamos nuestros datos o coordenadas:

```
head(scores(nmds, display = "species"))
```

```
##                NMDS1       NMDS2
## Achimill -0.8228039  0.04328404
## Agrostol  0.7109617 -0.28924281
```

```
## Airaprae -0.5281813  1.67987096
## Alopgeni  0.3909695 -0.58595828
## Anthodor -0.7202217  0.65914392
## Bellpere -0.4783837 -0.24446837
```

```
head(scores(nmds, display = "sites"))
```

```
##         NMDS1       NMDS2
## 1 -0.84052967 -0.71584228
## 2 -0.50485823 -0.40893732
## 3 -0.08267057 -0.43667741
## 4 -0.11562376 -0.52223869
## 5 -0.62654756 -0.08669609
## 6 -0.54270176  0.11315529
```

CCA

También conocido como análisis de correspondencia canónica.

Los métodos de ordenación anteriores son ordenaciones "sin restricciones", lo que significa que la ordenación se realiza solo considerando los recuentos de especies (counts).

La ordenación restringida es apropiada para lo que es común en los datos ecológicos: una matriz de comunidades y otra matriz de características ambientales o fisicoquímicas.

Con la ordenación restringida, podemos preguntarnos cómo se relacionan las variables ambientales con la composición de la comunidad.

Veamoslo con un ejemplo:

```
data("varespec") #especies
data("varechem") #fisicoquímicos
vares_cca <- cca(varespec ~ N+P+K+Ca+Mg+S+Al+Fe+Mn+Zn+Mo+Baresoil+Humdepth+pH , data=varechem)
summary(vares_cca)
```

```
##
## Call:
## cca(formula = varespec ~ N + P + K + Ca + Mg + S + Al + Fe +      Mn + Zn + Mo + Baresoil + Humdepth
##
## Partitioning of scaled Chi-square:
##               Inertia Proportion
## Total          2.0832      1.000
## Constrained    1.4415      0.692
## Unconstrained  0.6417      0.308
##
## Eigenvalues, and their contribution to the scaled Chi-square
##
## Importance of components:
##                           CCA1   CCA2    CCA3    CCA4    CCA5    CCA6    CCA7
## Eigenvalue            0.4389 0.2918 0.16285 0.14213 0.11795 0.08903 0.07029
## Proportion Explained  0.2107 0.1401 0.07817 0.06823 0.05662 0.04274 0.03374
## Cumulative Proportion 0.2107 0.3507 0.42890 0.49713 0.55375 0.59649 0.63023
##                           CCA8    CCA9    CCA10    CCA11    CCA12    CCA13
## Eigenvalue            0.05836 0.03114 0.013294 0.008364 0.006538 0.006156
## Proportion Explained  0.02801 0.01495 0.006382 0.004015 0.003139 0.002955
## Cumulative Proportion 0.65825 0.67319 0.679576 0.683592 0.686730 0.689685
```

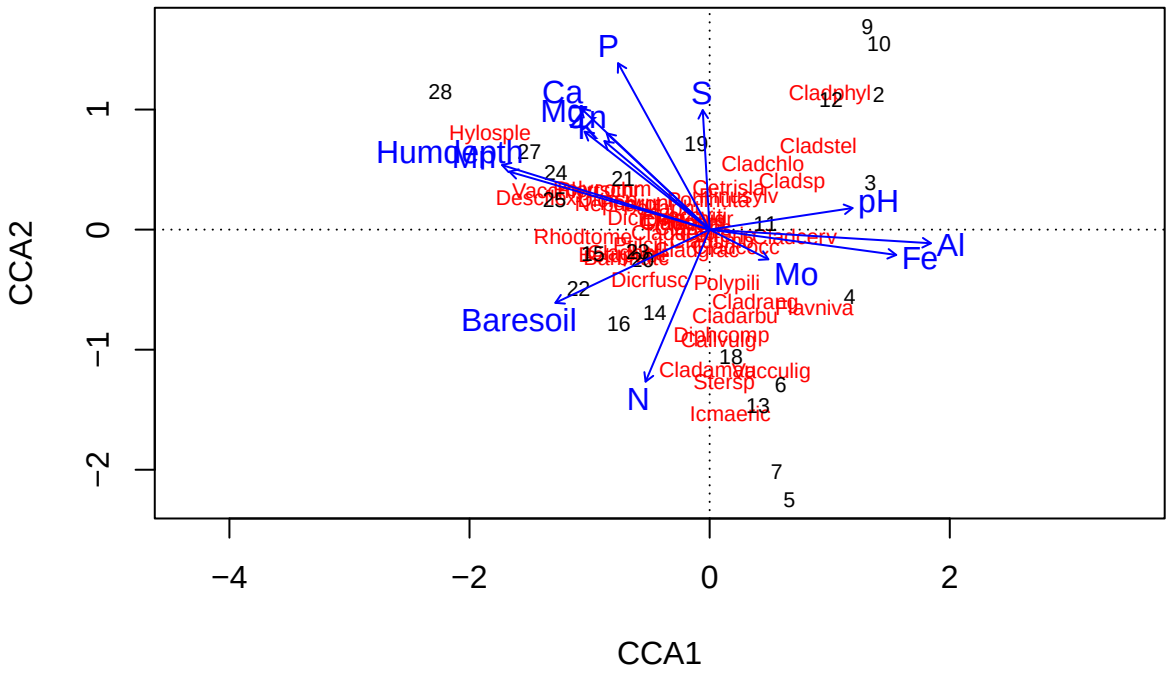
```
##                          CCA14     CA1     CA2     CA3     CA4     CA5     CA6
## Eigenvalue            0.004733 0.19776 0.14193 0.10117 0.07079 0.05330 0.03330
## Proportion Explained  0.002272 0.09493 0.06813 0.04857 0.03398 0.02559 0.01598
## Cumulative Proportion 0.691958 0.78689 0.85502 0.90359 0.93757 0.96315 0.97914
##                            CA7      CA8      CA9
## Eigenvalue            0.018868 0.015104 0.009488
## Proportion Explained  0.009057 0.007251 0.004554
## Cumulative Proportion 0.988195 0.995446 1.000000
##
## Accumulated constrained eigenvalues
## Importance of components:
##                          CCA1   CCA2   CCA3   CCA4    CCA5    CCA6    CCA7
## Eigenvalue            0.4389 0.2918 0.1628 0.1421 0.11795 0.08903 0.07029
## Proportion Explained  0.3045 0.2024 0.1130 0.0986 0.08183 0.06176 0.04877
## Cumulative Proportion 0.3045 0.5069 0.6198 0.7184 0.80027 0.86203 0.91080
##                          CCA8    CCA9    CCA10    CCA11    CCA12    CCA13
## Eigenvalue            0.05836 0.03114 0.013294 0.008364 0.006538 0.006156
## Proportion Explained  0.04049 0.02160 0.009223 0.005803 0.004536 0.004271
## Cumulative Proportion 0.95128 0.97288 0.982107 0.987910 0.992446 0.996716
##                          CCA14
## Eigenvalue            0.004733
## Proportion Explained  0.003284
## Cumulative Proportion 1.000000
##
## Scaling 2 for species and site scores
## * Species are scaled proportional to eigenvalues
## * Sites are unscaled: weighted dispersion equal on all dimensions
##
##
## Species scores
##
##               CCA1     CCA2      CCA3      CCA4      CCA5      CCA6
## Callvulg  0.075347 -0.93581  1.677742  0.695507  1.077518 -0.345001
## Empenigr -0.181340  0.07610  0.036462 -0.427727 -0.138153  0.010517
## Rhodtome -1.053549 -0.06026  0.077428 -0.938897 -0.213938 -0.518031
## Vaccmyrt -1.277428  0.30759  0.303704 -0.092088 -0.568820 -0.613023
## Vaccviti -0.152563  0.12054 -0.053031 -0.362279  0.083942  0.008938
## Pinusylv  0.242956  0.26432  0.223265 -0.273806  0.292102 -0.063335
## Descflex -1.443872  0.27019 -0.162082  0.606576 -0.476067  0.382590
## Betupube -0.711004 -0.22681 -0.083007 -2.408417 -0.216212 -1.671857
## Vacculig  0.513817 -1.18831 -0.377748  0.177035 -0.958084  0.311138
## Diphcomp  0.099310 -0.89289 -0.419273 -0.532348 -0.270745  0.622270
## Dicrsp   -0.849964  0.23153 -1.751924  0.260810  1.522412  0.390210
## Dicrfusc -0.499460 -0.41539  0.824743 -0.258156  0.112149  0.638702
## Dicrpoly -0.527090  0.08050 -0.812083 -1.201383  0.768689 -1.025365
## Hylosple -1.828026  0.79385  0.049816  1.358093 -0.916528 -0.223338
## Pleuschr -0.924978  0.33684 -0.009146  0.308091 -0.065518  0.018741
## Polypili  0.144172 -0.45586 -0.515356 -0.281796 -0.052660  0.050659
## Polyjuni -0.606869  0.21021 -0.352109 -0.336004 -0.612858  0.351629
## Polycomm -0.894165  0.32063 -0.234919 -1.076106 -0.408823 -0.776736
## Pohlnuta -0.009508  0.25268 -0.140571 -0.351201  0.424031 -0.096811
## Ptilcili -0.576115 -0.12234 -0.058593 -2.109265 -0.166198 -1.507591
## Barbhatc -0.694092 -0.22970 -0.118360 -2.574980 -0.172821 -2.054320
## Cladarbu  0.211517 -0.71201 -0.026366  0.052216 -0.040564 -0.078262
```

```
## Cladrang  0.381030 -0.61678 -0.243893  0.105921 -0.163536  0.032637
## Cladstel  0.906486  0.70213  0.082949  0.067771 -0.016579  0.027407
## Cladunci -0.230671  0.06372 -0.013810 -0.391170  0.910527 -0.146092
## Cladcocc  0.219419 -0.13619  0.128350 -0.077450  0.033754  0.125028
## Cladcorn -0.225404  0.07008 -0.090524 -0.258643 -0.109501  0.170706
## Cladgrac -0.108836 -0.18599 -0.159664 -0.201023  0.241156 -0.021594
## Cladfimb  0.020022 -0.09179  0.192626 -0.262413 -0.035959 -0.034780
## Cladcris -0.137056  0.01609  0.422960 -0.423861  0.138016 -0.129810
## Cladchlo  0.443621  0.55305 -0.278345 -0.576292  0.169030 -0.224882
## Cladbotr -0.680481 -0.19013  0.195105 -1.330144  0.218169 -1.262258
## Cladamau -0.015996 -1.16331 -0.728763 -0.498887 -0.350481  0.714608
## Cladsp    0.686166  0.39137  0.307091  0.279524  0.604150  0.124850
## Cetreric  0.064619 -0.03889 -0.427516  0.118844  0.945590 -0.173838
## Cetrisla  0.159171  0.35076 -0.049161 -0.884501  0.166607 -0.689545
## Flavniva  0.872373 -0.64645 -0.465365  1.961193  0.368671 -2.332045
## Nepharct -0.762768  0.19877 -0.558560 -0.057976 -1.137069  0.744096
## Stersp    0.121697 -1.28229 -0.963619 -0.003712 -0.369284  0.417103
## Peltapht -0.397796  0.16843  0.049634 -0.338986 -0.263955  0.194009
## Icmaeric  0.172805 -1.53313 -0.429975 -0.154452 -0.413750  0.319003
## Cladcerv  0.708032 -0.05882 -0.316283  1.225539  0.004871 -1.044377
## Claddefo -0.301412 -0.02090  0.243431 -0.564576  0.292677 -0.188788
## Cladphyl  1.002262  1.12620  0.016613 -0.101195  0.094379  0.145598
##
##
## Site scores (weighted averages of species scores)
##
##       CCA1     CCA2      CCA3     CCA4     CCA5     CCA6
## 18  0.1785 -1.05988 -0.408835 -0.60721 -0.56492  0.24175
## 15 -0.9702 -0.19714  0.421046  0.30324  0.15171  0.80394
## 24 -1.2798  0.47645 -2.946863  0.39292  3.95433  0.76592
## 27 -1.5009  0.65216  0.085837  0.76207 -1.23251 -0.09756
## 23 -0.5981 -0.18404 -0.135611 -1.16425 -0.30249  0.07033
## 19 -0.1103  0.71431  0.016591 -0.07773 -0.55210 -0.08258
## 22 -1.0921 -0.49026  2.120668 -0.43014  0.26010  1.87287
## 16 -0.7558 -0.78712  1.652152 -0.15892  0.47523  1.73677
## 28 -2.2421  1.15075  0.248921  1.88204 -1.80814 -1.19935
## 13  0.4035 -1.46904  2.240249  1.21956  1.85549 -0.91541
## 14 -0.4563 -0.69333  1.089571 -1.04519  2.70161  0.15628
## 20 -0.5583 -0.25296 -0.336340 -0.36433  0.27453  0.10923
## 25 -1.2922  0.25087 -1.456542 -0.02698  0.96227  2.19508
## 7   0.5576 -2.01700 -0.923568  0.14954 -1.34406  0.19237
## 5   0.6651 -2.24847 -1.631533  0.44110 -1.23074  0.53544
## 6   0.5920 -1.29165 -0.470112 -0.08331 -0.28830 -0.18265
## 3   1.3379  0.39399 -0.212551  0.26020 -0.61477  0.30075
## 4   1.1675 -0.55997 -0.207980  2.14490  0.35776 -3.17436
## 2   1.4091  1.12669  0.011297  0.04175 -0.40173  0.27311
## 9   1.3130  1.69016  0.238808 -0.13429  0.00160  0.04923
## 12  1.0115  1.08413  0.085287 -0.24485 -0.12365  0.18392
## 10  1.4105  1.54744  0.232569 -0.16699 -0.15736  0.16768
## 11  0.4651  0.05411 -0.146473  0.25902 -0.08197 -0.03886
## 21 -0.7191  0.42952  0.009702 -3.83149 -0.83861 -4.06109
##
##
## Site constraints (linear combinations of constraining variables)
```

```
##
##        CCA1     CCA2     CCA3     CCA4      CCA5    CCA6
## 18 -0.42308 -1.32466 -0.49215 -0.94489 -0.048464  0.9398
## 15 -0.19026  0.49687  0.45454 -0.52951 -0.076603 -0.7899
## 24 -0.86328  0.25213 -2.76035  0.56993  3.292710  0.2629
## 27 -1.69805  0.48669 -0.56351  1.07358 -0.614147  0.4988
## 23 -0.79557  0.10723  0.25751 -0.90419 -0.287557  0.4387
## 19 -0.67702  1.00130  0.03344 -1.00351 -0.141279 -0.9383
## 22 -0.81881 -0.67147  1.51674 -0.05858  0.566703  2.2159
## 16 -0.14877 -1.16222  1.02373 -0.44751 -0.154699 -0.2515
## 28 -2.07190  1.09778  0.49758  1.88707 -1.394002 -0.6375
## 13  0.16534 -1.35508  2.60193  1.25142  1.760111 -0.5461
## 14 -0.14069  0.20118  0.77762 -0.87922  0.676806 -0.3838
## 20 -0.68566  0.08107 -0.20421 -1.11529  1.112185 -0.7635
## 25 -0.90562  0.29517 -0.55183 -0.07379 -1.131782  0.8128
## 7   1.38453 -1.92877 -0.80045  0.36440 -1.653585 -0.1187
## 5   0.09709 -2.02095 -1.57794  0.03999 -0.441247  0.9902
## 6   0.41866 -0.56908 -0.32436  0.06603 -0.058116  0.3371
## 3   0.95649  0.12458 -0.51056  0.15157 -1.065096 -0.1616
## 4   0.85641 -0.79366 -0.46982  2.32495  0.468453 -2.8417
## 2   1.53650  0.92994  0.09664  0.25941 -0.009995  0.7130
## 9   1.53381  1.60412 -0.01520 -0.11658  0.698700  0.6643
## 12  0.44751  0.23990  0.93887 -0.28191  0.128819  0.3828
## 10  1.11107  1.59354 -0.04164  0.11005 -0.461130  0.2664
## 11  0.59050  0.36592 -0.04552 -0.14145 -0.070919 -0.3881
## 21 -0.68681 -0.23299 -0.17348 -2.78317 -0.205599 -2.1817
##
##
## Biplot scores for constraining variables
##
##                CCA1     CCA2      CCA3     CCA4      CCA5      CCA6
## N         -0.22290 -0.52891  0.006729  0.17735 -0.253216  0.102014
## P         -0.31866  0.57886 -0.162001  0.47947  0.184099 -0.121835
## K         -0.36612  0.30794  0.359824  0.47942  0.325444 -0.196637
## Ca        -0.44764  0.42176 -0.037765  0.09827  0.307969  0.043545
## Mg        -0.43499  0.34051 -0.142169  0.10790  0.497841 -0.005758
## S         -0.02406  0.41570  0.148384  0.44446  0.597063 -0.166296
## Al         0.76978 -0.04747  0.037610  0.39098  0.160905 -0.336554
## Fe         0.64909 -0.08811 -0.042067  0.26297 -0.069806 -0.111345
## Mn        -0.72232  0.22460  0.113052  0.29152 -0.138680  0.180471
## Zn        -0.35810  0.33493 -0.277916  0.34572  0.619191 -0.001195
## Mo         0.20413 -0.10334 -0.157007  0.32424  0.516439 -0.313525
## Baresoil -0.53675 -0.25477  0.136910 -0.52055  0.166621 -0.352409
## Humdepth -0.69673  0.20163  0.271625 -0.13574 -0.003252 -0.051350
## pH        0.49716  0.07509 -0.326341  0.02092 -0.145569 -0.059091
```

Ahora, visualicemos los resultados en un biplot:

```
plot(vares_cca)
```



Si queremos saver cuales son las variables con más peso en el análisis ocupamos la función **envfit()**.

```
envfit(vares_cca  ~ N+P+K+Ca+Mg+S+Al+Fe+Mn+Zn+Mo+Baresoil+Humdepth+pH ,
       data=varechem )
```

```
##
## ***VECTORS
##
##              CCA1     CCA2     r2 Pr(>r)
## N        -0.36715 -0.93016 0.2781  0.048 *
## P        -0.49393  0.86950 0.3861  0.011 *
## K        -0.76057  0.64926 0.2028  0.098 .
## Ca       -0.72500  0.68875 0.3355  0.013 *
## Mg       -0.78169  0.62367 0.2703  0.038 *
## S        -0.08664  0.99624 0.1505  0.209
## Al        0.99584 -0.09111 0.5130  0.004 **
## Fe        0.98659 -0.16322 0.3716  0.012 *
## Mn       -0.94705  0.32108 0.5003  0.002 **
## Zn       -0.72738  0.68624 0.2133  0.088 .
## Mo        0.88340 -0.46863 0.0462  0.643
## Baresoil -0.91064 -0.41319 0.2960  0.040 *
## Humdepth -0.95304  0.30286 0.4594  0.004 **
## pH        0.99256  0.12175 0.2151  0.103
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## Permutation: free
## Number of permutations: 999
```

perMANOVA

Análisis de varianza multivariante permutacional (perMANOVA).

La evaluación de las diferencias en la composición de la comunidad se realiza con el análisis de varianza multivariado permutacional.

Estas pruebas se realizan sobre distancias, es decir, evalúan las diferencias entre comunidades en función de la disimilitud.

```
dune_perm <- adonis(dune ~ Management+Use+Moisture, data = dune.env, method = "euclidean")
dune_perm2 <- adonis(dist(dune) ~ Management+Use+Moisture, data = dune.env)

dune_perm;dune_perm2
```

```
##
## Call:
## adonis(formula = dune ~ Management + Use + Moisture, data = dune.env,      method = "euclidean")
##
## Permutation: free
## Number of permutations: 999
##
## Terms added sequentially (first to last)
##
##            Df SumsOfSqs MeanSqs F.Model      R2 Pr(>F)
## Management  3    555.38 185.128  3.2580 0.34747  0.001 ***
## Use         2    136.15  68.077  1.1981 0.08518  0.278
## Moisture    3    281.76  93.919  1.6528 0.17628  0.053 .
## Residuals  11    625.05  56.823         0.39106
## Total      19   1598.35                 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Call:
## adonis(formula = dist(dune) ~ Management + Use + Moisture, data = dune.env)
##
## Permutation: free
## Number of permutations: 999
##
## Terms added sequentially (first to last)
##
##            Df SumsOfSqs MeanSqs F.Model      R2 Pr(>F)
## Management  3    555.38 185.128  3.2580 0.34747  0.001 ***
## Use         2    136.15  68.077  1.1981 0.08518  0.284
## Moisture    3    281.76  93.919  1.6528 0.17628  0.044 *
## Residuals  11    625.05  56.823         0.39106
## Total      19   1598.35                 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Como vemos el input puede ser una matriz de distancia o una data de comunidades pero indicandole la distancia que queremos ocupar.