Ordination graphs with vegan, BiodiversityR and ggplot2

Article · November 2020

CITATIONS
0

READS
44

1 author:

Roeland Kindt
Consultative Group on International Agricultural Research
953 PUBLICATIONS **31,591** CITATIONS

SEE PROFILE

Ordination graphs with vegan, BiodiversityR and ggplot2

Roeland Kindt

20/11/2020

Contents

Packages needed

```
library(BiodiversityR) # also loads vegan
library(ggplot2)
library(readxl)
library(ggsci)
library(ggrepel)
library(ggforce)
```

Introduction

The main objective of this document is to give some examples of how data from **ordination**, such **non metric multidimensional scaling** or **redundancy analysis** that were obtained via **vegan** and **BiodiversityR**, can be plotted via **ggplot2**. The key intermediate steps to allow plotting with ggplot2 is to get data in the 'long' (tidy) format that is used in ggplot2, which can be achieved with some functions in BiodiversityR that should work with most ordination results.

I expect that researchers and students will modify the scripts given here to create their own graphs. This obviously requires some working knowledge of the ggplot2 package, for which I will not give details here on specific functions, arguments and choices that I selected (as an introduction to ggplot2, I would especially recommend a two-part online workshop given by Thomas Lin Pedersen).

I will not give details on specific vegan functions to generate ordination results and neither will I dive deeply in the theory of ordination analysis. More details and key references are available from Chapter 10 - Analysis of ecological distance by ordination of the **Tree Diversity Analysis** book that I wrote with Ric Coe. This manual on community ecology and biodiversity analysis can be downloaded for free and is also the recommended citation for BiodiversityR. For a thorough insight in the different methods of ordination analysis, I highly recommend Numerical Ecology by the Pierre and Louis Legendre.

Another method by which somebody may become familiar with plotting ordination diagrams via ggplot2 is from the Graphical User Interface of BiodiversityR, available from function **BiodiversityRGUI**.

In the three examples below, the same work flow is used of:

- Step 1. Obtain an ordination graph via function **ordiplot**
- Step 2. Extract data for plotting via functions such as **sites.long** and **axis.long**
- Step 3. Use the extracted data to generate a plot with ggplot2

The **theme** for ggplot2 plotting will be modified as follows, resulting in a more empty plotting canvas than the default for ggplot2.

```
BioR.theme <- theme(
        panel.background = element_blank(),
        panel.border = element_blank(),
        panel.grid = element_blank(),
        axis.line = element_line("gray25"),
        text = element_text(size = 12),
        axis.text = element_text(size = 10, colour = "gray25"),
        axis.title = element_text(size = 14, colour = "gray25"),
        legend.title = element_text(size = 14),
        legend.text = element_text(size = 14),
        legend.key = element_blank())
```

Example 1: Dune meadow data

Typical for community ecology analysis in vegan and BiodiversityR, example data are included in two data sets, the **community data set** and the **environmental data set**. It is expected that the rows of these data sets correspond to the same sample sites.

The data in this example are data that were used as case study in Data Analysis in Community and Landscape Ecology and also in the Tree Diversity Analysis manual.

Data set **dune** is a community data set, where variables (columns) typically correspond to different species and data represents abundance of each species. Species names were abbreviated to eight characters, with for example Agrostol representing *Agrostis stolonifera*.

```
data(dune)
str(dune)
```

```
## 'data.frame':    20 obs. of  30 variables:
##  $ Achimill: num  1 3 0 0 2 2 2 0 0 4 ...
##  $ Agrostol: num  0 0 4 8 0 0 0 4 3 0 ...
##  $ Airaprae: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Alopgeni: num  0 2 7 2 0 0 0 5 3 0 ...
##  $ Anthodor: num  0 0 0 0 4 3 2 0 0 4 ...
##  $ Bellpere: num  0 3 2 2 2 0 0 0 0 2 ...
##  $ Bromhord: num  0 4 0 3 2 0 2 0 0 4 ...
##  $ Chenalbu: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Cirsarve: num  0 0 0 2 0 0 0 0 0 0 ...
##  $ Comapalu: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Eleopalu: num  0 0 0 0 0 0 0 4 0 0 ...
##  $ Elymrepe: num  4 4 4 4 4 0 0 0 6 0 ...
##  $ Empenigr: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Hyporadi: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Juncarti: num  0 0 0 0 0 0 0 4 4 0 ...
##  $ Juncbufo: num  0 0 0 0 0 0 2 0 4 0 ...
##  $ Lolipere: num  7 5 6 5 2 6 6 4 2 6 ...
##  $ Planlanc: num  0 0 0 0 5 5 5 0 0 3 ...
##  $ Poaprat : num  4 4 5 4 2 3 4 4 4 4 ...
##  $ Poatriv : num  2 7 6 5 6 4 5 4 5 4 ...
##  $ Ranuflam: num  0 0 0 0 0 0 0 2 0 0 ...
##  $ Rumeacet: num  0 0 0 0 5 6 3 0 2 0 ...
##  $ Sagiproc: num  0 0 0 5 0 0 0 2 2 0 ...
##  $ Salirepe: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Scorautu: num  0 5 2 2 3 3 3 3 2 3 ...
##  $ Trifprat: num  0 0 0 0 2 5 2 0 0 0 ...
##  $ Trifrepe: num  0 5 2 1 2 5 2 2 3 6 ...
##  $ Vicilath: num  0 0 0 0 0 0 0 0 0 1 ...
##  $ Bracruta: num  0 0 2 2 2 6 2 2 2 2 ...
##  $ Callcusp: num  0 0 0 0 0 0 0 0 0 0 ...
```

Data set **dune.env** is an environmental data set, where variables (columns) correspond to different descriptors (typically continuous and categorical variables) of the sample sites. One of the variables is **Management**, a categorical variable that describes different management categories, coded as BF (an abbreviation for biological farming), HF (hobby farming), NM (nature conservation management) and SF (standard farming).

```
data(dune.env)
summary(dune.env)
```

```
##        A1         Moisture Management       Use    Manure
##  Min.   : 2.800   1:7      BF:3       Hayfield:7   0:6
##  1st Qu.: 3.500   2:4      HF:5       Haypastu:8   1:3
##  Median : 4.200   4:2      NM:6       Pasture :5   2:4
##  Mean   : 4.850   5:7      SF:6                    3:4
##  3rd Qu.: 5.725                                    4:3
##  Max.   :11.500
```

For some plotting methods, it is necessary that the environmental data set is attached:

```
attach(dune.env)
```

Step 1

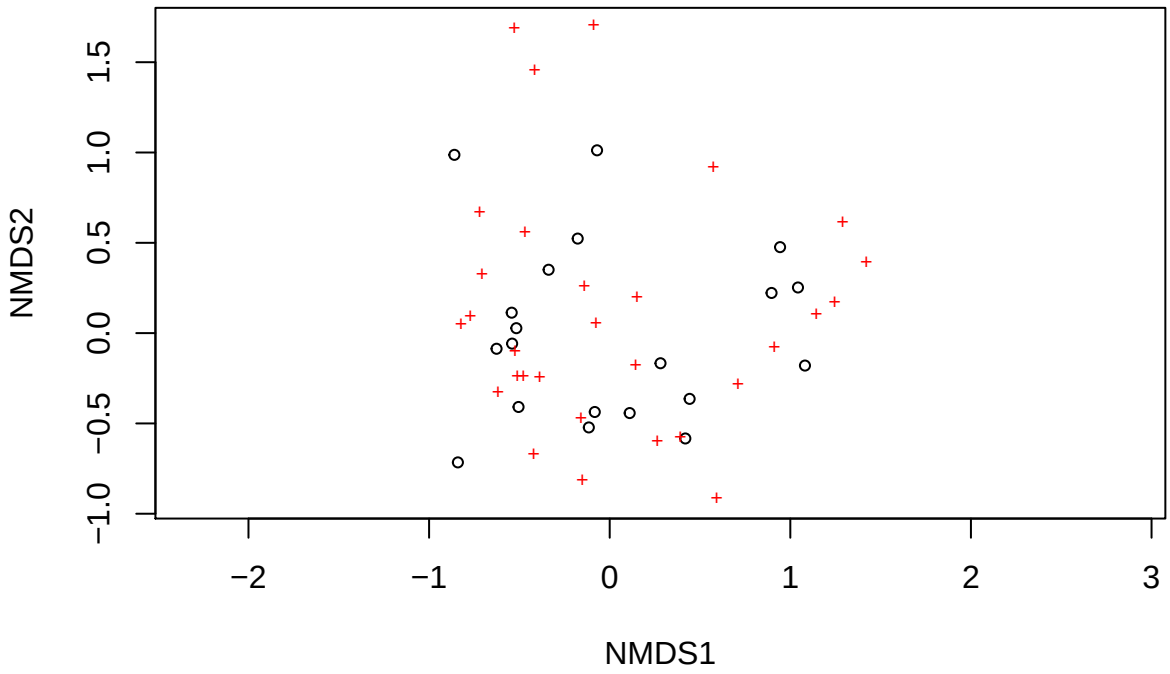
Within the first step, a ordination diagram is generated via the **ordiplot** function. For this example, the ordination method is non metric multidimensional scaling, available via function **metaMDS**.

```
# script generated by the BiodiversityR GUI from the unconstrained ordination menu
Ordination.model1 <- metaMDS(dune, distance='bray', k=2, trymax=1,
  autotransform=TRUE, noshare=0.1, expand=TRUE, trace=1, plot=FALSE)
```

```
## Run 0 stress 0.1192678
## Run 1 stress 0.1183186
## ... New best solution
## ... Procrustes: rmse 0.02027065  max resid 0.06495793
## Run 2 stress 0.119268
## Run 3 stress 0.1183186
## ... Procrustes: rmse 2.21056e-05  max resid 6.524274e-05
## ... Similar to previous best
## Run 4 stress 0.1183187
## ... Procrustes: rmse 5.866521e-05  max resid 0.0001571926
## ... Similar to previous best
## Run 5 stress 0.1886532
## Run 6 stress 0.1192679
## Run 7 stress 0.119268
## Run 8 stress 0.1192683
## Run 9 stress 0.1183186
## ... New best solution
## ... Procrustes: rmse 1.062683e-05  max resid 2.563018e-05
## ... Similar to previous best
## Run 10 stress 0.1183186
## ... Procrustes: rmse 8.16368e-05  max resid 0.0002678326
## ... Similar to previous best
## Run 11 stress 0.1183186
## ... New best solution
## ... Procrustes: rmse 1.348606e-05  max resid 4.411437e-05
## ... Similar to previous best
## Run 12 stress 0.1183186
## ... Procrustes: rmse 9.183005e-06  max resid 2.801707e-05
## ... Similar to previous best
## Run 13 stress 0.1183186
## ... Procrustes: rmse 8.300514e-05  max resid 0.0002688047
## ... Similar to previous best
## Run 14 stress 0.1183186
## ... Procrustes: rmse 9.794096e-06  max resid 2.845689e-05
## ... Similar to previous best
## Run 15 stress 0.1183186
## ... Procrustes: rmse 4.119442e-06  max resid 7.762614e-06
## ... Similar to previous best
## Run 16 stress 0.1886532
## Run 17 stress 0.1192678
## Run 18 stress 0.1192682
## Run 19 stress 0.2810094
## Run 20 stress 0.1183186
## ... Procrustes: rmse 2.45072e-05  max resid 7.571321e-05
## ... Similar to previous best
## *** Solution reached
```

This ordination result can now be plotted via **ordiplot**.

```
plot1 <- ordiplot(Ordination.model1, choices=c(1,2))
```



Step 2

To plot data via ggplot2, we extract information on the locations of sites (circles in the ordiplot) via function **sites.long**. Information on characteristics of the sites is added via the argument of **env.data**.

```
sites.long1 <- sites.long(plot1, env.data=dune.env)
head(sites.long1)
```

```
##    A1 Moisture Management      Use Manure       axis1       axis2 labels
## 1 2.8        1         SF Haypastu      4 -0.84053743 -0.71583558      1
## 2 3.5        1         BF Haypastu      2 -0.50485672 -0.40893563      2
## 3 4.3        2         SF Haypastu      4 -0.08266982 -0.43667771      3
## 4 4.2        2         SF Haypastu      4 -0.11562478 -0.52223931      4
## 5 6.3        1         HF Hayfield      2 -0.62654539 -0.08669526      5
## 6 4.3        1         HF Haypastu      2 -0.54269205  0.11315912      6
```

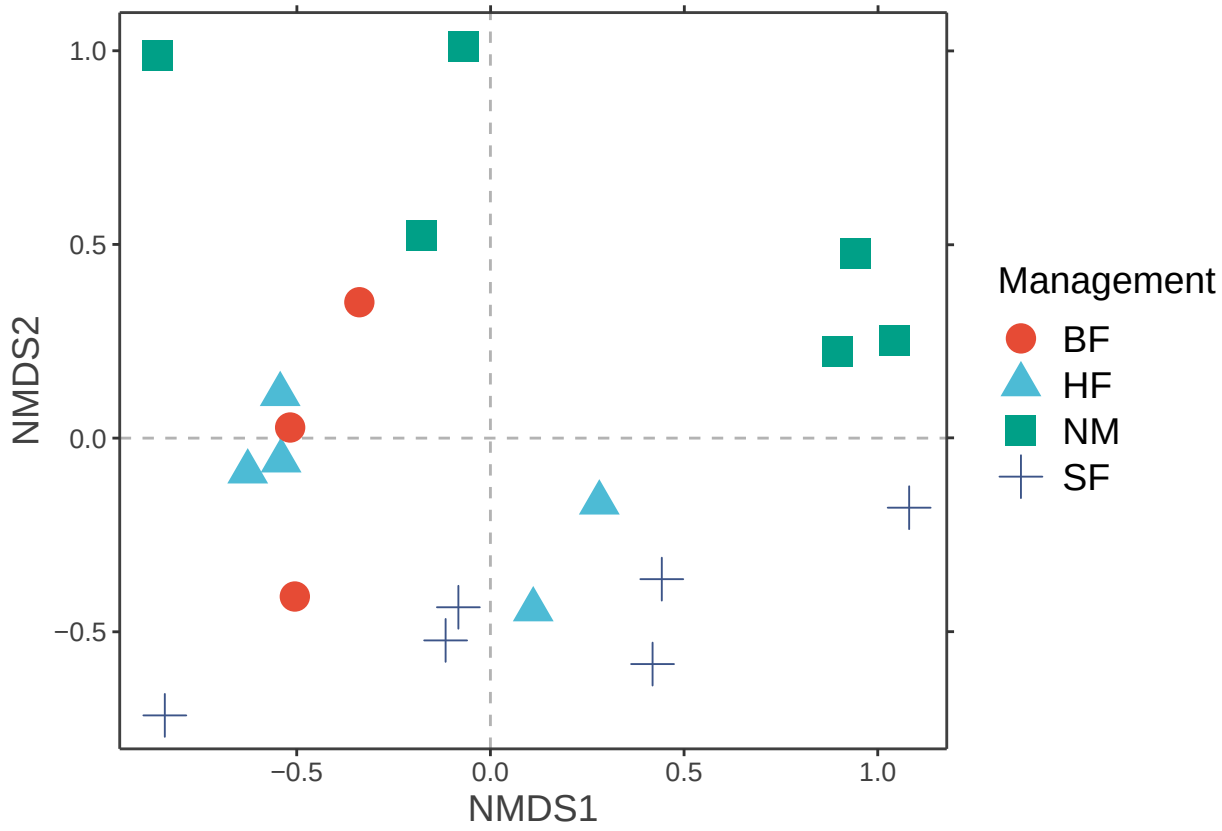
Step 3

Now we can generate the plot, where we differentiate among different categories of Management...

```
plotgg1 <- ggplot() +
    geom_vline(xintercept = c(0), color = "grey70", linetype = 2) +
    geom_hline(yintercept = c(0), color = "grey70", linetype = 2) +
    xlab("NMDS1") +
```

```
    ylab("NMDS2") +
    scale_x_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    scale_y_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    geom_point(data=sites.long1,
               aes(x=axis1, y=axis2, colour=Management, shape=Management),
               size=5) +
    BioR.theme +
    ggsci::scale_colour_npg() +
    coord_fixed(ratio=1)

plotgg1
```



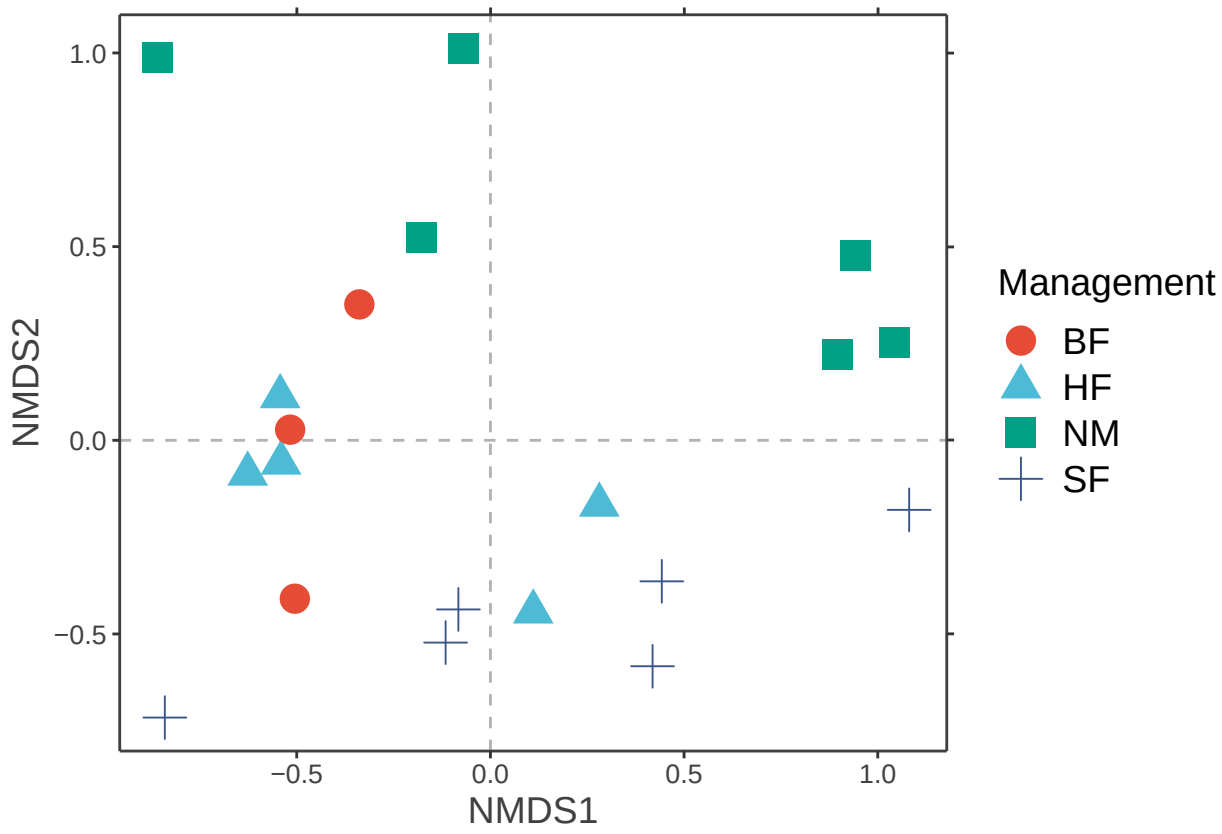
. . . and see the results.

The functions that extract the 'long' data from the ordiplot can also be used directly when using a ggplot object, combining steps 2 and 3 as below.

```
plotgg1 <- ggplot() +
    geom_vline(xintercept = c(0), color = "grey70", linetype = 2) +
    geom_hline(yintercept = c(0), color = "grey70", linetype = 2) +
    xlab("NMDS1") +
    ylab("NMDS2") +
    scale_x_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    scale_y_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    geom_point(data=sites.long(plot1, env.data=dune.env),
               aes(x=axis1, y=axis2, colour=Management, shape=Management),
               size=5) +
```

```
    BioR.theme +
    ggsci::scale_colour_npg() +
    coord_fixed(ratio=1)

plotgg1
```



Example 2: Dune meadow data with constrained ordination

For the second example, we used the contrained ordination method of **redundancy analysis**.

Step 1

The ordiplot object is obtained from the result of via function **rda**. The ordination is done with a community data set that is transformed by the Hellinger method as recommended in this article.

```
# script generated by the BiodiversityR GUI from the constrained ordination menu
dune.Hellinger <- disttransform(dune, method='hellinger')
Ordination.model2 <- rda(dune.Hellinger ~ Management, data=dune.env, scaling="species")
summary(Ordination.model2)
```

```
##
## Call:
## rda(formula = dune.Hellinger ~ Management, data = dune.env, scaling = "species")
##
## Partitioning of variance:
##               Inertia Proportion
```

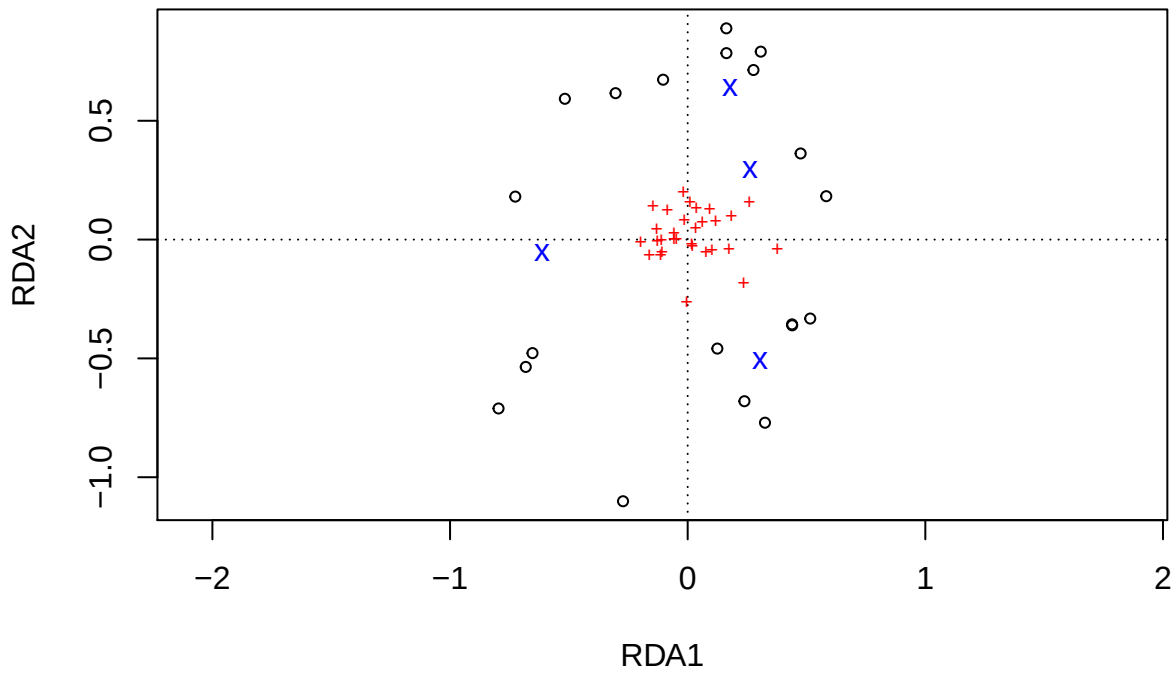
```
## Total          0.5559     1.0000
## Constrained    0.1667     0.2999
## Unconstrained  0.3892     0.7001
##
## Eigenvalues, and their contribution to the variance
##
## Importance of components:
##                            RDA1    RDA2    RDA3    PC1     PC2     PC3     PC4
## Eigenvalue             0.09377 0.05304 0.01988 0.1279 0.05597 0.04351 0.03963
## Proportion Explained   0.16869 0.09542 0.03575 0.2300 0.10069 0.07827 0.07129
## Cumulative Proportion 0.16869 0.26411 0.29986 0.5299 0.63054 0.70881 0.78010
##                            PC5     PC6     PC7     PC8     PC9     PC10     PC11
## Eigenvalue             0.03080 0.02120 0.01623 0.01374 0.01138 0.009469 0.007651
## Proportion Explained   0.05541 0.03814 0.02919 0.02471 0.02047 0.017034 0.013764
## Cumulative Proportion 0.83551 0.87365 0.90284 0.92755 0.94802 0.965056 0.978820
##                            PC12     PC13     PC14     PC15      PC16
## Eigenvalue             0.003957 0.003005 0.002485 0.001670 0.0006571
## Proportion Explained   0.007117 0.005406 0.004470 0.003004 0.0011821
## Cumulative Proportion 0.985937 0.991344 0.995814 0.998818 1.0000000
##
## Accumulated constrained eigenvalues
## Importance of components:
##                            RDA1    RDA2    RDA3
## Eigenvalue             0.09377 0.05304 0.01988
## Proportion Explained   0.56255 0.31822 0.11923
## Cumulative Proportion 0.56255 0.88077 1.00000
##
## 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:  1.80276
##
##
## Species scores
##
##               RDA1      RDA2      RDA3       PC1       PC2       PC3
## Achimill  0.036568  0.127978  0.016399 -0.188968 -0.070111  0.175399
## Agrostol -0.003429 -0.270611 -0.018285  0.361207  0.005321 -0.014204
## Airaprae -0.127487 -0.011618 -0.005281 -0.120126  0.103133  0.060671
## Alopgeni  0.235229 -0.190348  0.027102  0.163865  0.129547 -0.078901
## Anthodor -0.085534  0.115742 -0.079991 -0.240230  0.117203  0.201272
## Bellpere  0.033345  0.041954  0.083723 -0.081623 -0.079661 -0.076104
## Bromhord  0.093316  0.120369  0.065335 -0.058227 -0.047198  0.043097
## Chenalbu  0.016343 -0.027339  0.008563  0.006896  0.028380  0.004920
## Cirsarve  0.019792 -0.033109  0.010370 -0.009773 -0.008401 -0.025099
## Comapalu -0.110016 -0.010026 -0.004558  0.091463 -0.045993  0.026311
## Eleopalu -0.160847 -0.069849 -0.041472  0.352544 -0.115164  0.126956
## Elymrepe  0.173955 -0.047695 -0.002020 -0.108520 -0.206622 -0.115376
## Empenigr -0.047886 -0.004364 -0.001984 -0.029817  0.061163 -0.018612
## Hyporadi -0.130851  0.036162  0.047182 -0.127285  0.148136  0.027482
## Juncarti -0.057086 -0.003074 -0.101556  0.265930 -0.061713 -0.009618
## Juncbufo  0.102986 -0.052188 -0.062882  0.033316  0.152206 -0.038314
## Lolipere  0.260211  0.153503  0.052002 -0.192751 -0.234269 -0.146056
## Planlanc -0.018339  0.192788 -0.064491 -0.222661  0.025888  0.091459
```

```
## Poaprat   0.183546  0.093202  0.019068 -0.196708 -0.136129 -0.115068
## Poatriv   0.377233 -0.046485 -0.039312 -0.019700 -0.010085  0.010877
## Ranuflam -0.113470 -0.073249 -0.022781  0.249877 -0.046445  0.073742
## Rumeacet  0.118187  0.070051 -0.198589 -0.056786  0.065922  0.042193
## Sagiproc  0.076848 -0.060054  0.017557  0.035039  0.222856 -0.152162
## Salirepe -0.197203 -0.017971 -0.008170 -0.014209  0.015031 -0.104243
## Scorautu -0.146131  0.134240  0.015763 -0.059562  0.117768 -0.088453
## Trifprat  0.061485  0.069094 -0.135080 -0.066713 -0.001921  0.069306
## Trifrepe  0.010076  0.151446  0.031054  0.039480  0.082043 -0.081712
## Vicilath -0.014220  0.076123  0.084100 -0.026628  0.009338 -0.057100
## Bracruta -0.057834  0.021502 -0.054847  0.083143  0.070717 -0.115855
## Callcusp -0.107307 -0.059711  0.009214  0.169330 -0.068364  0.107835
##
##
## Site scores (weighted sums of species scores)
##
##       RDA1    RDA2     RDA3      PC1      PC2       PC3
## 1   0.5831  0.1826  0.52222 -0.59008 -0.95148 -0.002365
## 2   0.4752  0.3624  0.90879  0.05773 -0.27927 -0.053465
## 3   0.5157 -0.3323  0.54177 -0.17847 -0.28527 -0.368932
## 4   0.4399 -0.3608  0.65164 -0.15066 -0.12951 -0.386916
## 5   0.2767  0.7135 -0.68027 -0.32785 -0.13648  0.331094
## 6   0.1630  0.7846 -0.99354 -0.24794  0.06413  0.280023
## 7   0.3075  0.7908 -0.69144 -0.29554  0.01116  0.283797
## 8   0.1248 -0.4586 -0.03533  0.58291 -0.02785 -0.296149
## 9   0.4391 -0.3568 -0.47994  0.28843  0.08904 -0.598766
## 10  0.1626  0.8889  0.52513 -0.10281 -0.02239  0.398688
## 11 -0.1034  0.6728  0.63968  0.04508  0.30166 -0.345223
## 12  0.2387 -0.6802 -0.39191  0.13764  0.90201 -0.091365
## 13  0.3253 -0.7706  0.08451  0.12875  0.52983  0.091846
## 14 -0.6531 -0.4778  0.18077  0.45837 -0.26343  0.275022
## 15 -0.6814 -0.5359 -0.46471  0.55931 -0.24900  0.020745
## 16 -0.2720 -1.1011 -0.44903  0.65282 -0.06558  0.757733
## 17 -0.5168  0.5924 -0.02957 -0.74414  0.25120  0.742874
## 18 -0.3035  0.6161  0.46554 -0.42677 -0.21642 -0.831729
## 19 -0.7255  0.1808  0.15665 -0.38151  0.78259 -0.238142
## 20 -0.7960 -0.7107 -0.46098  0.53475 -0.30493  0.031229
##
##
## Site constraints (linear combinations of constraining variables)
##
##       RDA1     RDA2     RDA3      PC1      PC2       PC3
## 1   0.3051 -0.51040  0.15987 -0.59008 -0.95148 -0.002365
## 2   0.1781  0.64135  0.69120  0.05773 -0.27927 -0.053465
## 3   0.3051 -0.51040  0.15987 -0.17847 -0.28527 -0.368932
## 4   0.3051 -0.51040  0.15987 -0.15066 -0.12951 -0.386916
## 5   0.2622  0.29468 -0.57610 -0.32785 -0.13648  0.331094
## 6   0.2622  0.29468 -0.57610 -0.24794  0.06413  0.280023
## 7   0.2622  0.29468 -0.57610 -0.29554  0.01116  0.283797
## 8   0.2622  0.29468 -0.57610  0.58291 -0.02785 -0.296149
## 9   0.2622  0.29468 -0.57610  0.28843  0.08904 -0.598766
## 10  0.1781  0.64135  0.69120 -0.10281 -0.02239  0.398688
## 11  0.1781  0.64135  0.69120  0.04508  0.30166 -0.345223
## 12  0.3051 -0.51040  0.15987  0.13764  0.90201 -0.091365
```

```
## 13  0.3051 -0.51040  0.15987  0.12875  0.52983  0.091846
## 14 -0.6127 -0.05584 -0.02538  0.45837 -0.26343  0.275022
## 15 -0.6127 -0.05584 -0.02538  0.55931 -0.24900  0.020745
## 16  0.3051 -0.51040  0.15987  0.65282 -0.06558  0.757733
## 17 -0.6127 -0.05584 -0.02538 -0.74414  0.25120  0.742874
## 18 -0.6127 -0.05584 -0.02538 -0.42677 -0.21642 -0.831729
## 19 -0.6127 -0.05584 -0.02538 -0.38151  0.78259 -0.238142
## 20 -0.6127 -0.05584 -0.02538  0.53475 -0.30493  0.031229
##
##
## Biplot scores for constraining variables
##
##                  RDA1     RDA2     RDA3 PC1 PC2 PC3
## ManagementHF  0.3756  0.42205 -0.82512   0   0   0
## ManagementNM -0.9950 -0.09068 -0.04122   0   0   0
## ManagementSF  0.4955 -0.82890  0.25963   0   0   0
##
##
## Centroids for factor constraints
##
##                  RDA1     RDA2     RDA3 PC1 PC2 PC3
## ManagementBF  0.1781  0.64135  0.69120   0   0   0
## ManagementHF  0.2622  0.29468 -0.57610   0   0   0
## ManagementNM -0.6127 -0.05584 -0.02538   0   0   0
## ManagementSF  0.3051 -0.51040  0.15987   0   0   0
```

This ordination result can now be plotted via **ordiplot**.

```
plot2 <- ordiplot(Ordination.model2, choices=c(1,2))
```



Step 2

To plot data via ggplot2, information on the locations of sites (circles in the ordiplot) is obtained via function **sites.long**. Information on species is extracted by function **species.long**.

```
sites.long2 <- sites.long(plot2, env.data=dune.env)
head(sites.long1)
```

```
##    A1 Moisture Management      Use Manure       axis1       axis2 labels
## 1 2.8        1         SF Haypastu      4 -0.84053743 -0.71583558      1
## 2 3.5        1         BF Haypastu      2 -0.50485672 -0.40893563      2
## 3 4.3        2         SF Haypastu      4 -0.08266982 -0.43667771      3
## 4 4.2        2         SF Haypastu      4 -0.11562478 -0.52223931      4
## 5 6.3        1         HF Hayfield      2 -0.62654539 -0.08669526      5
## 6 4.3        1         HF Haypastu      2 -0.54269205  0.11315912      6
```

```
species.long2 <- species.long(plot2)
species.long2
```

```
##                axis1        axis2   labels
## Achimill  0.036568078  0.127977891 Achimill
## Agrostol -0.003429466 -0.270610691 Agrostol
## Airaprae -0.127487366 -0.011618161 Airaprae
## Alopgeni  0.235229086 -0.190347650 Alopgeni
## Anthodor -0.085534098  0.115742475 Anthodor
## Bellpere  0.033345323  0.041954014 Bellpere
## Bromhord  0.093315942  0.120369228 Bromhord
```

```
## Chenalbu  0.016342543 -0.027338897 Chenalbu
## Cirsarve  0.019791802 -0.033109049 Cirsarve
## Comapalu -0.110015792 -0.010025944 Comapalu
## Eleopalu -0.160846983 -0.069849205 Eleopalu
## Elymrepe  0.173954575 -0.047694621 Elymrepe
## Empenigr -0.047885538 -0.004363898 Empenigr
## Hyporadi -0.130850618  0.036162085 Hyporadi
## Juncarti -0.057085751 -0.003074036 Juncarti
## Juncbufo  0.102986248 -0.052188302 Juncbufo
## Lolipere  0.260211309  0.153502553 Lolipere
## Planlanc -0.018338806  0.192788162 Planlanc
## Poaprat   0.183546184  0.093202170  Poaprat
## Poatriv   0.377232554 -0.046484976  Poatriv
## Ranuflam -0.113470344 -0.073248720 Ranuflam
## Rumeacet  0.118187238  0.070051484 Rumeacet
## Sagiproc  0.076847673 -0.060053832 Sagiproc
## Salirepe -0.197203102 -0.017971486 Salirepe
## Scorautu -0.146131145  0.134240387 Scorautu
## Trifprat  0.061485089  0.069093660 Trifprat
## Trifrepe  0.010076281  0.151445912 Trifrepe
## Vicilath -0.014220226  0.076123468 Vicilath
## Bracruta -0.057833775  0.021501638 Bracruta
## Callcusp -0.107306687 -0.059711006 Callcusp
```

Information on the labeling of the axes is obtained with function **axis.long**. This information is extracted from the ordination model and not the ordiplot, hence it is important to select the same axes via argument 'choices'. The extracted information includes information on the percentage of explained variation. I suggest that you cross-check with the summary of the redundancy analysis, where information on proportion explained is given.

```
axis.long2 <- axis.long(Ordination.model2, choices=c(1, 2))
axis.long2
```

```
##   axis     ggplot        label
## 1    1 xlab.label RDA1 (16.9%)
## 2    2 ylab.label  RDA2 (9.5%)
```

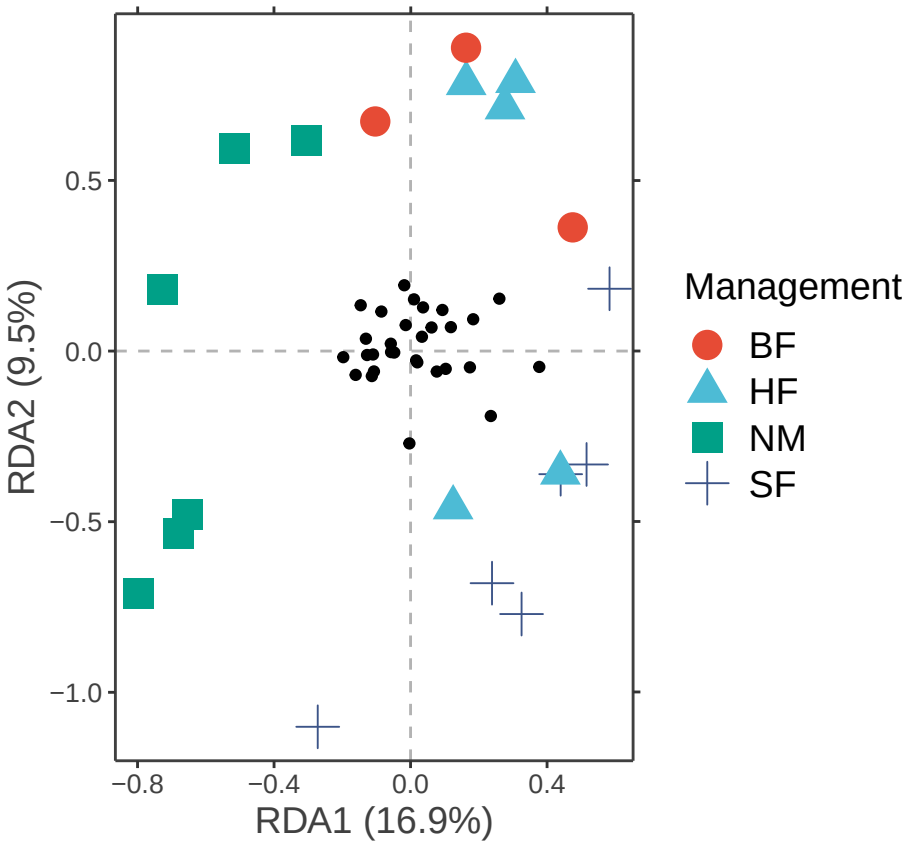
Step 3

Now we can generate the plot, where we add information on sites and species similar to the ordiplot.

```
plotgg2 <- ggplot() +
    geom_vline(xintercept = c(0), color = "grey70", linetype = 2) +
    geom_hline(yintercept = c(0), color = "grey70", linetype = 2) +
    xlab(axis.long2[1, "label"]) +
    ylab(axis.long2[2, "label"]) +
    scale_x_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    scale_y_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    geom_point(data=sites.long2,
               aes(x=axis1, y=axis2, colour=Management, shape=Management),
               size=5) +
    geom_point(data=species.long2,
               aes(x=axis1, y=axis2)) +
    BioR.theme +
    ggsci::scale_colour_npg() +
    coord_fixed(ratio=1)
```

And see the result...

```
plotgg2
```



With an ordination via redundancy analysis, the positions of species should be interpreted as arrows. It is straightforward to show the positions of the species as arrows. However, with a large number of species, it is better to reduce the number to those that explain more of the variation among the sites. This information is obtained with the **envfit** function of vegan.

```
spec.envfit <- envfit(plot2, env=dune.Hellinger)
spec.data.envfit <- data.frame(r=spec.envfit$vectors$r, p=spec.envfit$vectors$pvals)
species.long2 <- species.long(plot2, spec.data=spec.data.envfit)
species.long2
```

```
##                   r     p        axis1        axis2   labels
## Achimill 0.51120821 0.004  0.036568078  0.127977891 Achimill
## Agrostol 0.88043167 0.001 -0.003429466 -0.270610691 Agrostol
## Airaprae 0.28434077 0.035 -0.127487366 -0.011618161 Airaprae
## Alopgeni 0.73537212 0.001  0.235229086 -0.190347650 Alopgeni
## Anthodor 0.47733652 0.009 -0.085534098  0.115742475 Anthodor
## Bellpere 0.20842931 0.125  0.033345323  0.041954014 Bellpere
## Bromhord 0.31316708 0.049  0.093315942  0.120369228 Bromhord
## Chenalbu 0.12490789 0.291  0.016342543 -0.027338897 Chenalbu
## Cirsarve 0.07819748 0.793  0.019791802 -0.033109049 Cirsarve
## Comapalu 0.27920171 0.040 -0.110015792 -0.010025944 Comapalu
## Eleopalu 0.63294392 0.001 -0.160846983 -0.069849205 Eleopalu
## Elymrepe 0.43568538 0.011  0.173954575 -0.047694621 Elymrepe
## Empenigr 0.15033609 0.155 -0.047885538 -0.004363898 Empenigr
```

```
## Hyporadi 0.34152966 0.013 -0.130850618  0.036162085 Hyporadi
## Juncarti 0.39888741 0.013 -0.057085751 -0.003074036 Juncarti
## Juncbufo 0.23314927 0.107  0.102986248 -0.052188302 Juncbufo
## Lolipere 0.64836700 0.001  0.260211309  0.153502553 Lolipere
## Planlanc 0.72258576 0.001 -0.018338806  0.192788162 Planlanc
## Poaprat  0.68186496 0.001  0.183546184  0.093202170  Poaprat
## Poatriv  0.84781457 0.001  0.377232554 -0.046484976  Poatriv
## Ranuflam 0.67206044 0.002 -0.113470344 -0.073248720 Ranuflam
## Rumeacet 0.18554697 0.167  0.118187238  0.070051484 Rumeacet
## Sagiproc 0.12671017 0.301  0.076847673 -0.060053832 Sagiproc
## Salirepe 0.32856158 0.015 -0.197203102 -0.017971486 Salirepe
## Scorautu 0.41239233 0.009 -0.146131145  0.134240387 Scorautu
## Trifprat 0.27418926 0.047  0.061485089  0.069093660 Trifprat
## Trifrepe 0.06136616 0.575  0.010076281  0.151445912 Trifrepe
## Vicilath 0.24521625 0.088 -0.014220226  0.076123468 Vicilath
## Bracruta 0.10673228 0.384 -0.057833775  0.021501638 Bracruta
## Callcusp 0.45329066 0.006 -0.107306687 -0.059711006 Callcusp
```

Species are selected that explain at least 60% of variation.

```
species.long3 <- species.long2[species.long2$r >= 0.6, ]
species.long3
```

```
##                  r     p        axis1       axis2   labels
## Agrostol 0.8804317 0.001 -0.003429466 -0.27061069 Agrostol
## Alopgeni 0.7353721 0.001  0.235229086 -0.19034765 Alopgeni
## Eleopalu 0.6329439 0.001 -0.160846983 -0.06984921 Eleopalu
## Lolipere 0.6483670 0.001  0.260211309  0.15350255 Lolipere
## Planlanc 0.7225858 0.001 -0.018338806  0.19278816 Planlanc
## Poaprat  0.6818650 0.001  0.183546184  0.09320217  Poaprat
## Poatriv  0.8478146 0.001  0.377232554 -0.04648498  Poatriv
## Ranuflam 0.6720604 0.002 -0.113470344 -0.07324872 Ranuflam
```

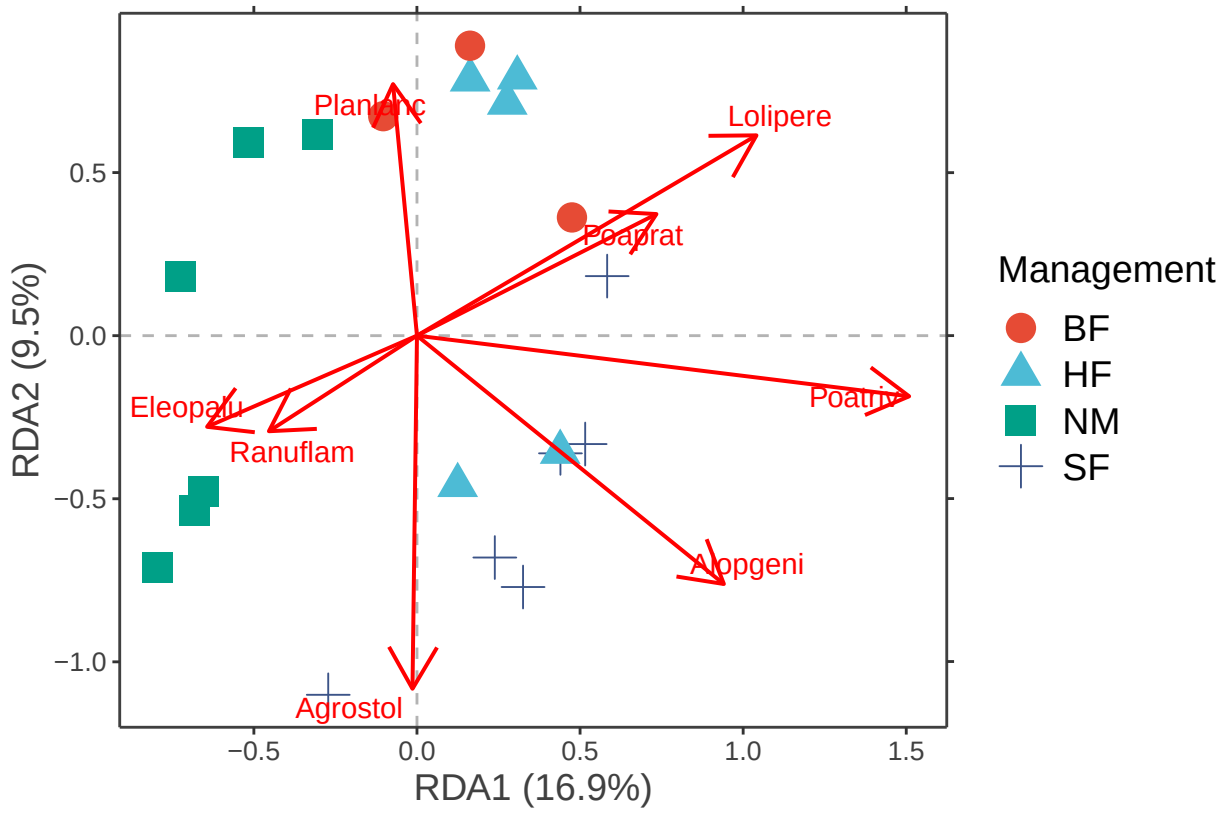
Now the information of the selected species can be added to the ordination diagram.

```
plotgg2 <- ggplot() +
    geom_vline(xintercept = c(0), color = "grey70", linetype = 2) +
    geom_hline(yintercept = c(0), color = "grey70", linetype = 2) +
    xlab(axis.long2[1, "label"]) +
    ylab(axis.long2[2, "label"]) +
    scale_x_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    scale_y_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    geom_point(data=sites.long2,
               aes(x=axis1, y=axis2, colour=Management, shape=Management),
               size=5) +
    geom_segment(data=species.long3,
                 aes(x=0, y=0, xend=axis1*4, yend=axis2*4),
                 colour="red", size=0.7, arrow=arrow()) +
    geom_text_repel(data=species.long3,
                    aes(x=axis1*4, y=axis2*4, label=labels),
                    colour="red") +
    BioR.theme +
    ggsci::scale_colour_npg() +
    coord_fixed(ratio=1)
```

... which gives the following ordination diagram (length of vectors were multiplied, which does not change

their interpretation):

```
plotgg2
```



Example 3: adding ordispider diagrams

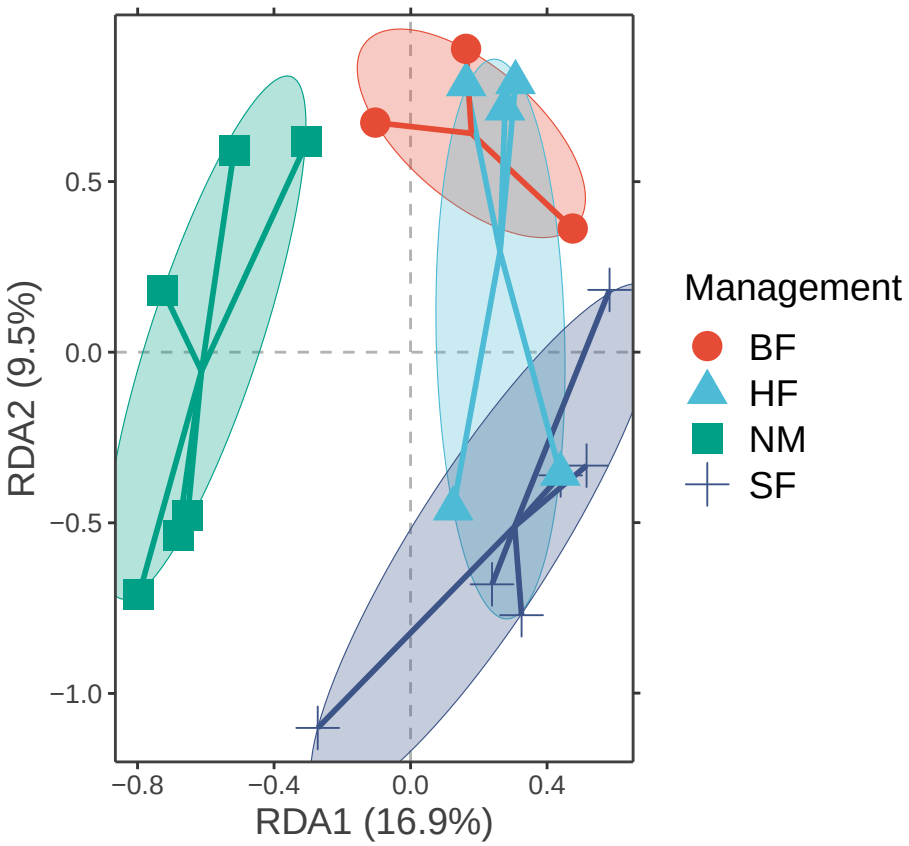
Function **centroids.long** obtains data on the centroids of each grouping. It is possible to directly add their results as 'ordispider diagrams' in ggplot. Here also another layer is added of **superellipses** obtained from the ggforce::geom_mark_ellipse function.

```
plotgg3 <- ggplot() +
    geom_vline(xintercept = c(0), color = "grey70", linetype = 2) +
    geom_hline(yintercept = c(0), color = "grey70", linetype = 2) +
    xlab(axis.long2[1, "label"]) +
    ylab(axis.long2[2, "label"]) +
    scale_x_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    scale_y_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    geom_mark_ellipse(data=sites.long2,
                      aes(x=axis1, y=axis2, colour=Management,
                          fill=after_scale(alpha(colour, 0.2))),
                      expand=0, size=0.2, show.legend=FALSE) +
    geom_segment(data=centroids.long(sites.long2, grouping=Management),
                 aes(x=axis1c, y=axis2c, xend=axis1, yend=axis2, colour=Management),
                 size=1, show.legend=FALSE) +
    geom_point(data=sites.long2,
               aes(x=axis1, y=axis2, colour=Management, shape=Management),
```

```
                size=5) +
    BioR.theme +
    ggsci::scale_colour_npg() +
    coord_fixed(ratio=1)
```

... which gives this result

```
plotgg3
```



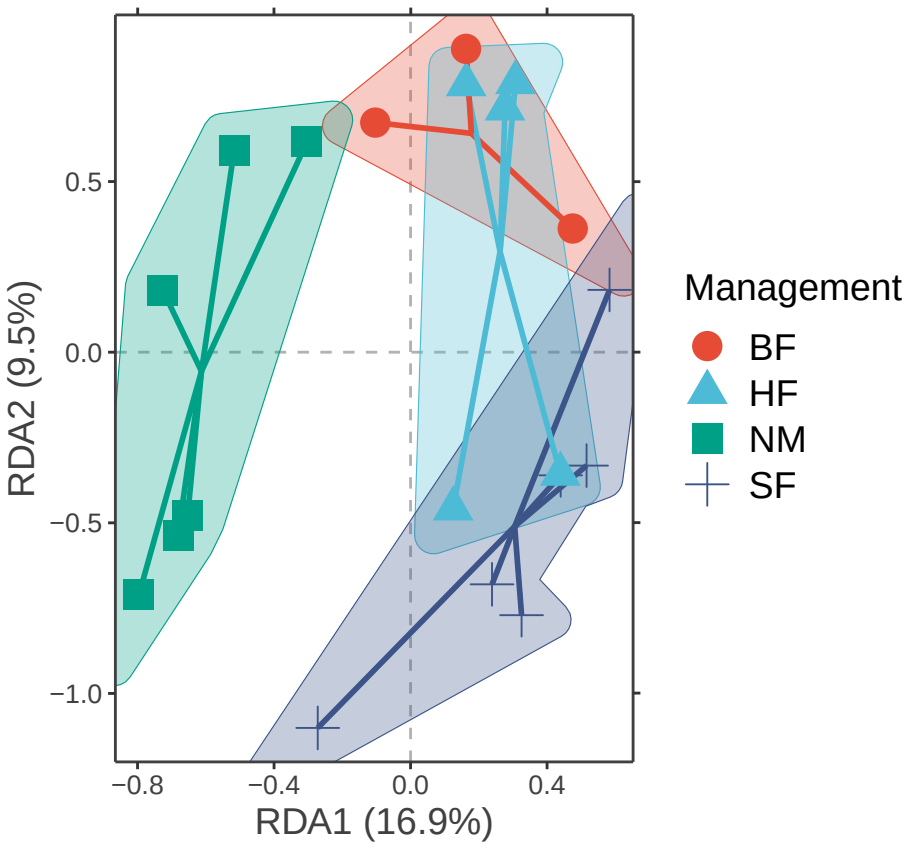
Instead of superellipses, add a concave hull via the ggforce package

```
plotgg3 <- ggplot() +
    geom_vline(xintercept = c(0), color = "grey70", linetype = 2) +
    geom_hline(yintercept = c(0), color = "grey70", linetype = 2) +
    xlab(axis.long2[1, "label"]) +
    ylab(axis.long2[2, "label"]) +
    scale_x_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    scale_y_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    geom_mark_hull(data=sites.long2,
                   aes(x=axis1, y=axis2, colour=Management,
                       fill=after_scale(alpha(colour, 0.2))),
                   concavity=0.1, size=0.2, show.legend=FALSE) +
    geom_segment(data=centroids.long(sites.long2, grouping=Management),
                 aes(x=axis1c, y=axis2c, xend=axis1, yend=axis2, colour=Management),
                 size=1, show.legend=FALSE) +
    geom_point(data=sites.long2,
               aes(x=axis1, y=axis2, colour=Management, shape=Management),
```

```
                size=5) +
    BioR.theme +
    ggsci::scale_colour_npg() +
    coord_fixed(ratio=1)
```

... which gives this result

```
plotgg3
```

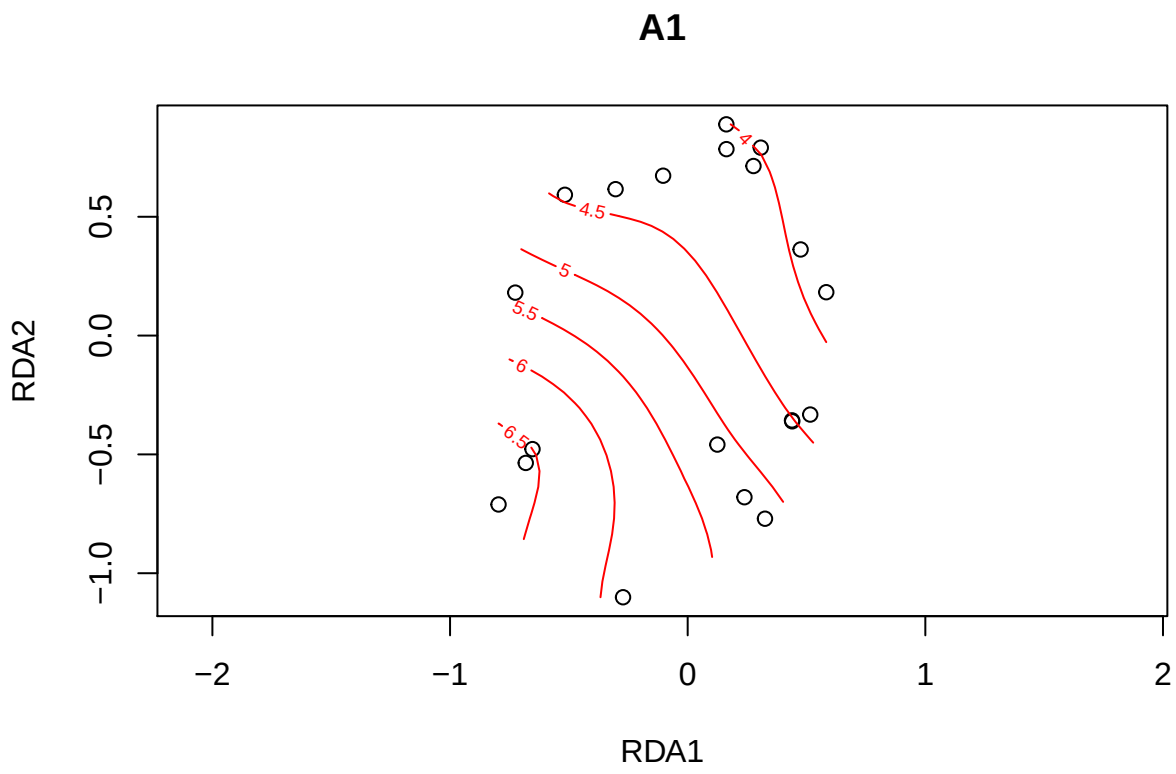


Example 4: adding ordisurf diagrams

Where the previous example examined patterns for a categorical explanatory variable, here we will explore patterns for the continuous variable A1, documenting the thickness of the A1 horizon.

The vegan package includes a method of adding a smooth surface to an ordination diagram. This method is implemented in function **ordisurf**.

```
A1.surface <- ordisurf(plot2, y=A1)
```



Function **ordisurfgrid.long** extracts the data to be plotted with ggplot2

```
A1.grid <- ordisurfgrid.long(A1.surface)
```

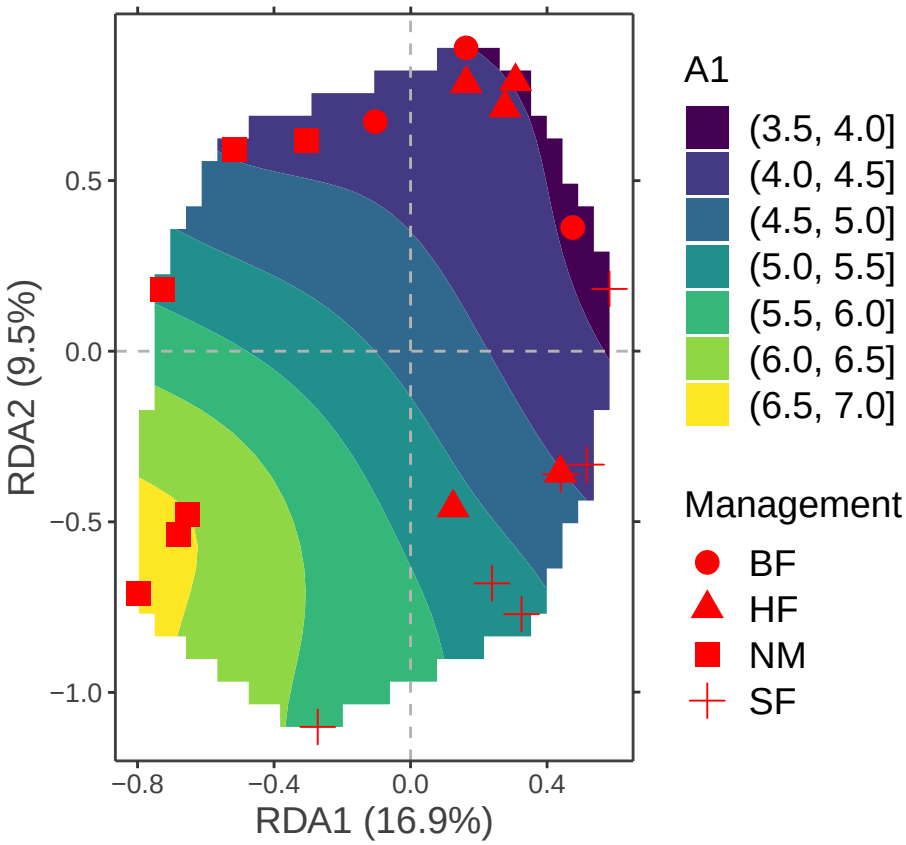
Preparing the plot

```
plotgg4 <- ggplot() +
    geom_contour_filled(data=A1.grid,
                        aes(x=x, y=y, z=z)) +
    geom_vline(xintercept = c(0), color = "grey70", linetype = 2) +
    geom_hline(yintercept = c(0), color = "grey70", linetype = 2) +
    xlab(axis.long2[1, "label"]) +
    ylab(axis.long2[2, "label"]) +
    scale_x_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    scale_y_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    geom_point(data=sites.long2,
               aes(x=axis1, y=axis2, shape=Management),
               colour="red", size=4) +
    BioR.theme +
    scale_fill_viridis_d() +
    labs(fill="A1") +
    coord_fixed(ratio=1)
```

... and seeing the plot.

```
plotgg4
```

```
## Warning: Removed 209 rows containing non-finite values (stat_contour_filled).
```



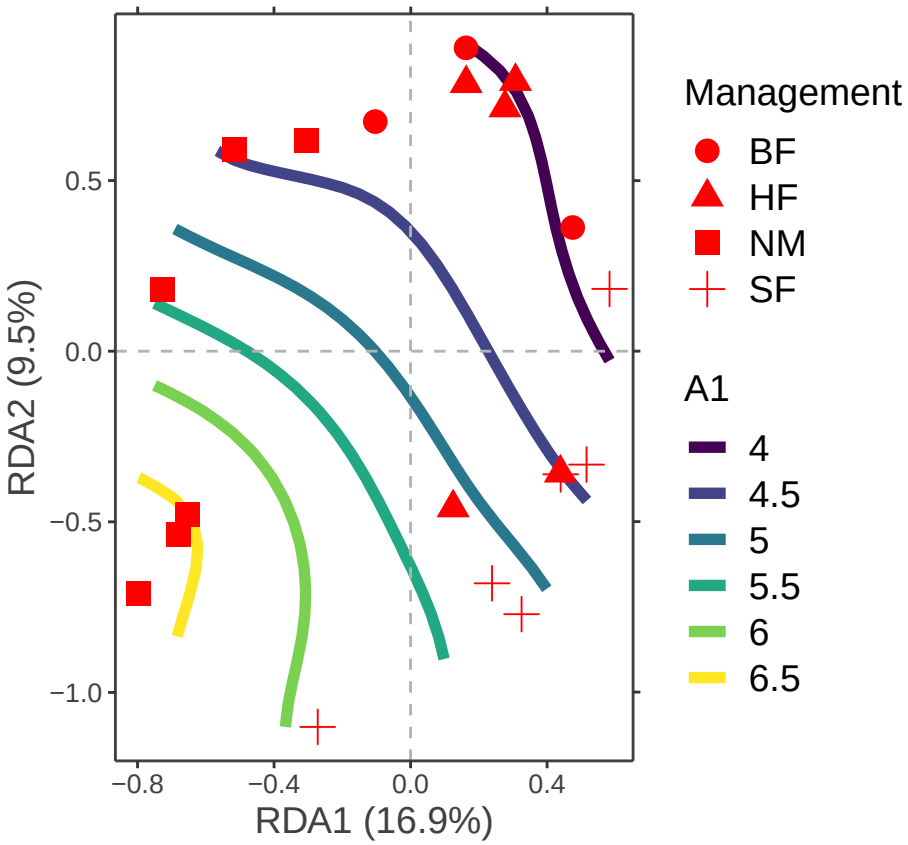
As an alternative method, colour the contour lines:

```
plotgg4 <- ggplot() +
    geom_contour(data=A1.grid,
                 aes(x=x, y=y, z=z, colour=factor(after_stat(level))),
                 size=2) +
    geom_vline(xintercept = c(0), color = "grey70", linetype = 2) +
    geom_hline(yintercept = c(0), color = "grey70", linetype = 2) +
    xlab(axis.long2[1, "label"]) +
    ylab(axis.long2[2, "label"]) +
    scale_x_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    scale_y_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    geom_point(data=sites.long2,
               aes(x=axis1, y=axis2, shape=Management),
               colour="red", size=4) +
    BioR.theme +
    scale_colour_viridis_d() +
    labs(colour="A1") +
    coord_fixed(ratio=1)
```

... which has as result

```
plotgg4
```

```
## Warning: Removed 209 rows containing non-finite values (stat_contour).
```



Example 5: Tree inventories from Panama

The second data sets document tree species composition and site characteristics for 1-ha plots sampled in Panama. The data set can be be downloaded from the supplementary data provided by Condit et al. 2002. Beta-diversity in tropical forest trees. *Science* 295: 666-669.

As I had some problems with accessing the downloaded Excel file directly with readxl, I saved the file in the .xlsx format on a local directory in my computer.

```
Condit.down <- "C:\\BiodiversityR_test\\Condit.xlsx"
```

The following script reads and formats the community data set.

```
Condit.1 <- as.data.frame(readxl::read_excel(Condit.down, sheet=1))
spec.columns <- as.character(Condit.1[, 1])
Condit.spec <- as.data.frame(t(Condit.1[, -1]))
names(Condit.spec) <- spec.columns
```

The community data set has 100 sites (rows) and 802 species (columns):

```
dim(Condit.spec)
```

```
## [1] 100 802
```

Next the environmental data set is read:

```
Condit.2 <- as.data.frame(readxl::read_excel(Condit.down, sheet=2))
```

```
## New names:
```

```
## * `` -> ...8
```

```
site.rows <- as.character(Condit.2[, 1])
Condit.env <- Condit.2[, 2:7]
rownames(Condit.env) <- site.rows
Condit.env$Age.cat <- factor(Condit.env$age, levels=c("1", "2", "3"))
```

In the script above, I created a new variables of **Age.cat**, a categorical variable corresponding to age category.

```
summary(Condit.env)
```

```
##     EW coord         NS coord           precip          elev
##  Min.   :600714   Min.   : 962862   Min.   :1887   Min.   : 10.0
##  1st Qu.:625929   1st Qu.:1011569   1st Qu.:2516   1st Qu.:120.0
##  Median :626404   Median :1011819   Median :2530   Median :120.0
##  Mean   :631536   Mean   :1013061   Mean   :2581   Mean   :150.4
##  3rd Qu.:637892   3rd Qu.:1012908   3rd Qu.:2530   3rd Qu.:120.0
##  Max.   :688165   Max.   :1045987   Max.   :4002   Max.   :830.0
##                                     NA's   :2      NA's   :2
##       age         geology          Age.cat
##  Min.   :1.00   Length:100         1   :18
##  1st Qu.:2.00   Class :character   2   :15
##  Median :3.00   Mode  :character   3   :65
##  Mean   :2.48                      NA's: 2
##  3rd Qu.:3.00
##  Max.   :3.00
##  NA's   :2
```

Some further changes were made to the data sets, first to ensure that they had the same rownames (I checked first that the order of sites was the same).

```
check.datasets(Condit.spec, Condit.env)
```

```
## Warning: rownames for community and environmental datasets are different
```

```
# same rownames
rownames(Condit.spec) <- rownames(Condit.env)
```

A next change was to remove cases with missing values.

```
Condit.env <- Condit.env[complete.cases(Condit.env), ]
Condit.spec <- same.sites(Condit.spec, Condit.env)
check.datasets(Condit.spec, Condit.env)
```

```
## OK
```

And a final change was to only use one (plot B27) of the 1-ha subplots of a 50-ha plot, in order to have a more even spatial distribution among the sites.

```
rows.include <- c(28, 51:98)
rownames(Condit.spec)[rows.include]
```

```
##  [1] "B27" "p1"  "p2"  "p3"  "p4"  "p5"  "p6"  "p7"  "p8"  "p9"  "p10" "p11"
## [13] "p12" "p13" "p14" "p15" "p16" "p17" "p18" "p19" "p20" "p21" "p22" "p23"
## [25] "p24" "p25" "p26" "p27" "p28" "p29" "p30" "p31" "p32" "p33" "p34" "p35"
## [37] "p36" "p37" "p38" "p39" "C1"  "C2"  "C3"  "C4"  "S0"  "S1"  "S2"  "S3"
## [49] "S4"
```

```
Condit.spec2 <- Condit.spec[rows.include, ]
Condit.env2 <- Condit.env[rows.include, ]
```

```
check.datasets(Condit.spec2, Condit.env2)
```

```
## OK
```

Step 1

Now that we have a community and environmental data set in the proper format, we can proceed with step 1.

```
# script generated by the BiodiversityR GUI from the constrained ordination menu
Condit.Hellinger <- disttransform(Condit.spec2, method='hellinger')
Ordination.model3 <- rda(Condit.Hellinger ~ Age.cat + precip + elev, data=Condit.env2, scaling="species
Ordination.model3
```

```
## Call: rda(formula = Condit.Hellinger ~ Age.cat + precip + elev, data =
## Condit.env2, scaling = "species")
##
##               Inertia Proportion Rank
## Total          0.7512     1.0000
## Constrained    0.1654     0.2202    4
## Unconstrained  0.5858     0.7798   44
## Inertia is variance
##
## Eigenvalues for constrained axes:
##    RDA1    RDA2    RDA3    RDA4
## 0.08270 0.04215 0.02281 0.01773
##
## Eigenvalues for unconstrained axes:
##     PC1     PC2     PC3     PC4     PC5     PC6     PC7     PC8
## 0.10237 0.05800 0.03250 0.03085 0.02598 0.02295 0.02238 0.01996
## (Showing 8 of 44 unconstrained eigenvalues)
```

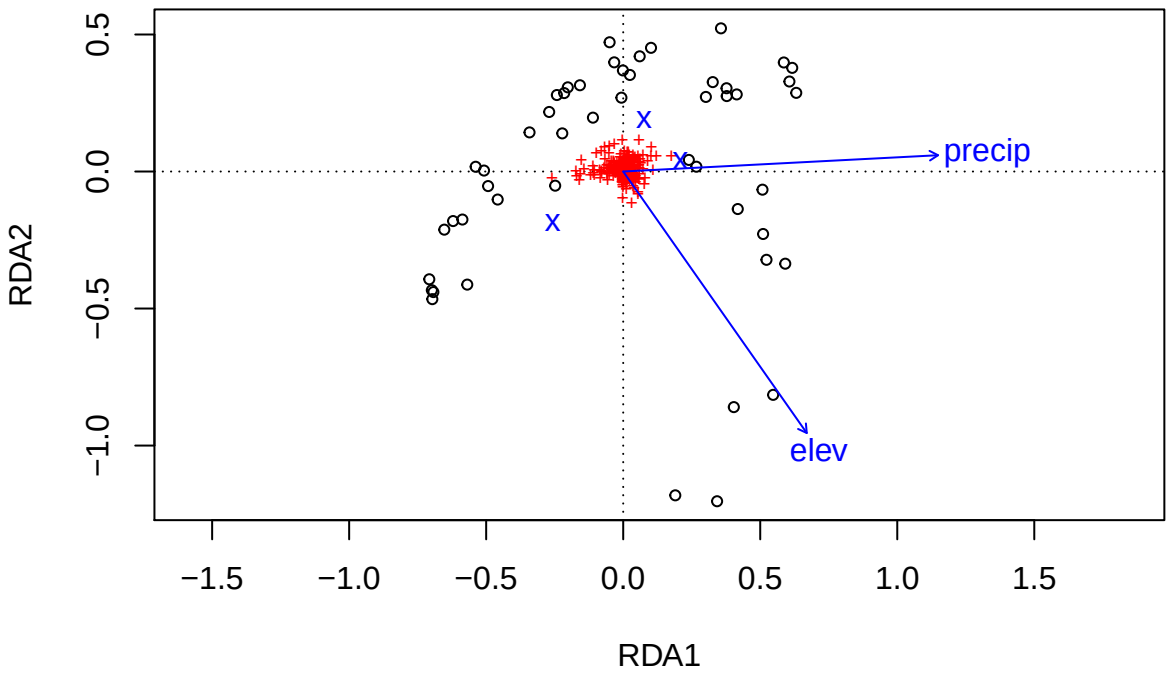
```
# summary(Ordination.model3) # do not give summary as scores given for 802 species

attach(Condit.env2) # variables needed in some plotting methods
```

```
## The following object is masked from package:datasets:
##
##     precip
```

The ordiplot is now:

```
plot3 <- ordiplot(Ordination.model3, choices=c(1,2))
```



Step 2

In step 2, extract the various scores for plotting.

```
sites.long3 <- sites.long(plot3, env.data=Condit.env2)

spec.envfit <- envfit(plot3, env=Condit.Hellinger, permutations=99)
spec.data.envfit <- data.frame(r=spec.envfit$vectors$r, p=spec.envfit$vectors$pvals)
species.long2 <- species.long(plot3, spec.data=spec.data.envfit)
species.long3 <- species.long2[species.long2$r >= 0.6, ]
species.long3$labels <- make.cepnames(species.long3$labels)
species.long3
```

```
##                                r    p        axis1         axis2   labels
## Anacardium.excelsum       0.6577745 0.01 -0.259960115 -0.0291946997 Anacexce
## Antirhea.trichantha       0.6008900 0.01 -0.159841002 -0.0382757823 Antitric
## Brosimum.alicastrum       0.6235811 0.01 -0.105072774  0.0009667776 Brosalic
## Cavanillesia.platanifolia 0.6082452 0.01 -0.108585267 -0.0106381153 Cavaplat
## Protium.tenuifolium       0.6905969 0.01 -0.152769757  0.0355642627 Prottenu
## Attalea.butyracea         0.6642871 0.01 -0.169694174 -0.0224727150 Attabuty
## Socratea.exorrhiza        0.6073032 0.01  0.175183951  0.0497468126 Socrexor
## Virola.sebifera           0.6085578 0.01 -0.002468489  0.1107599845 Virosebi
```

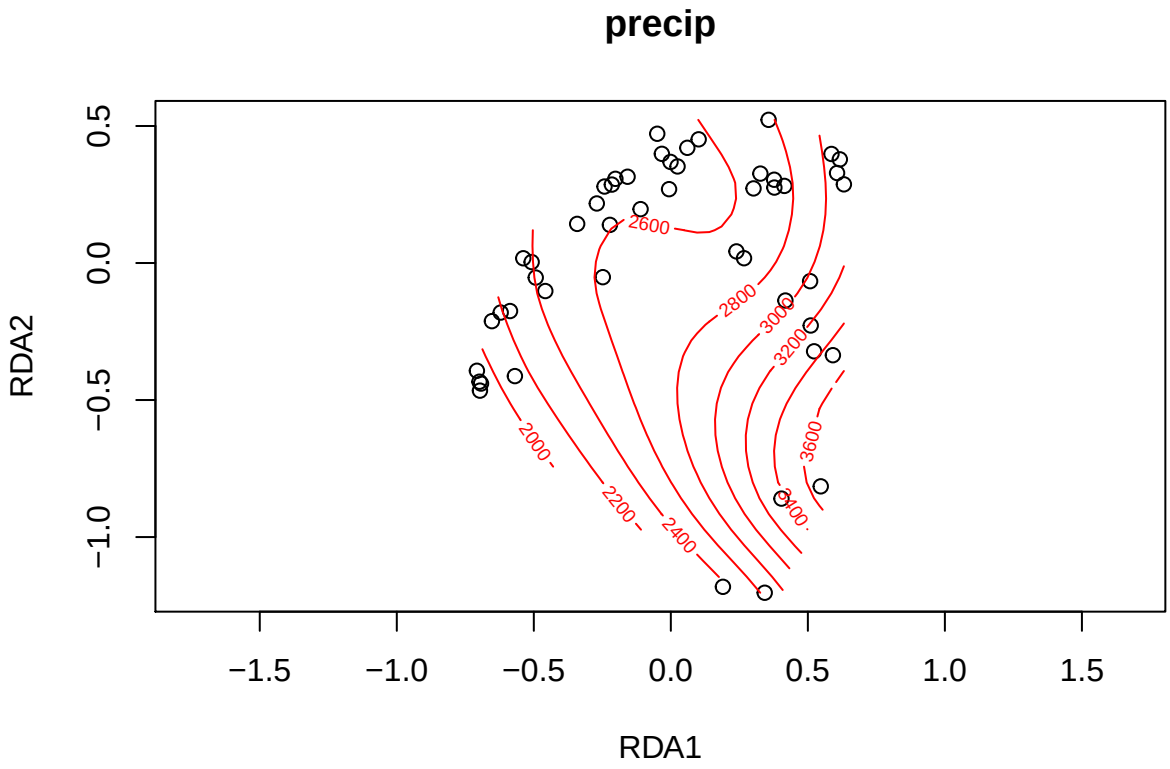
```
vectors.envfit <- envfit(plot3, env=Condit.env2)
vectors.long3 <- vectorfit.long(vectors.envfit)
vectors.long3
```

```
##            vector      axis1       axis2         r     p
## EW coord EW coord -0.6571952 -0.75372040 0.1505758 0.020
## NS coord NS coord  0.8230097  0.56802731 0.4038393 0.001
## precip     precip  0.9973844 -0.07228006 0.6146344 0.001
## elev         elev  0.5693957 -0.82206361 0.7641714 0.001
## age           age  0.9132501  0.40739948 0.2390186 0.004
```

```
axis.long3 <- axis.long(Ordination.model3, choices=c(1, 2))
axis.long3
```

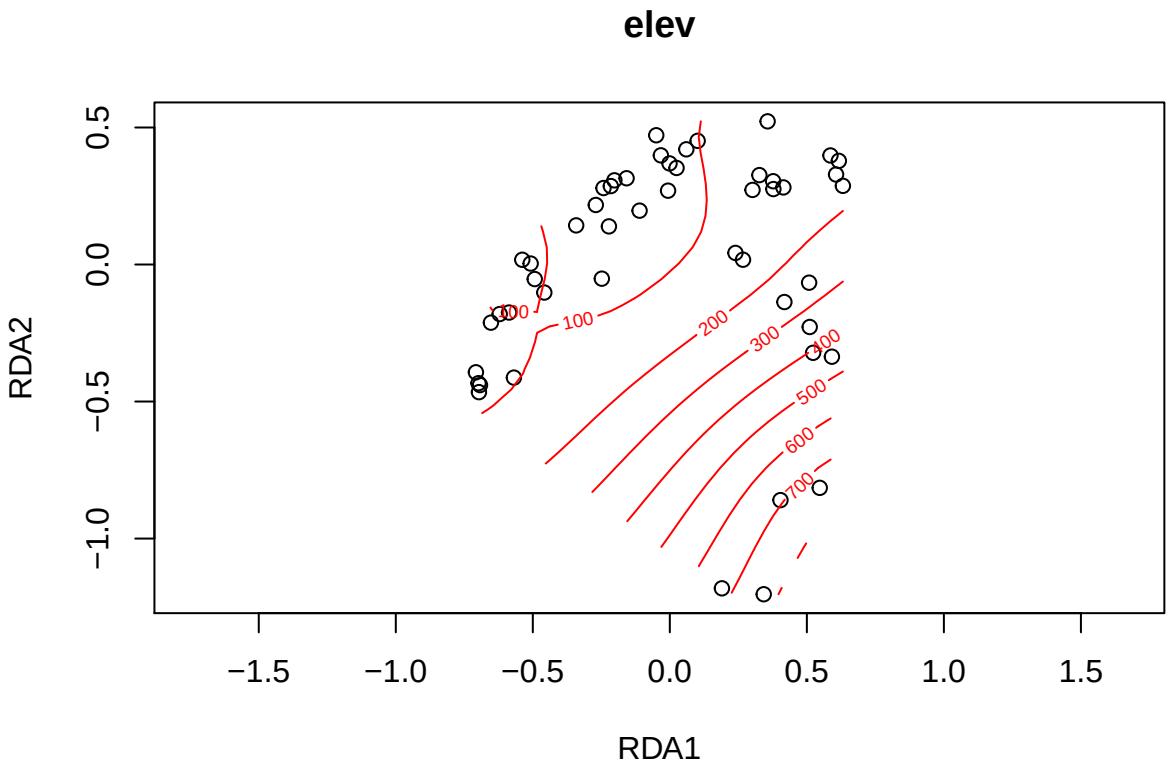
```
##   axis     ggplot       label
## 1    1 xlab.label  RDA1 (11%)
## 2    2 ylab.label RDA2 (5.6%)
```

```
precip.surface <- ordisurf(plot3, y=precip)
```




```
precip.grid <- ordisurfgrid.long(precip.surface)

elev.surface <- ordisurf(plot3, y=elev)
```




```
elev.grid <- ordisurfgrid.long(elev.surface)
```

Step 3

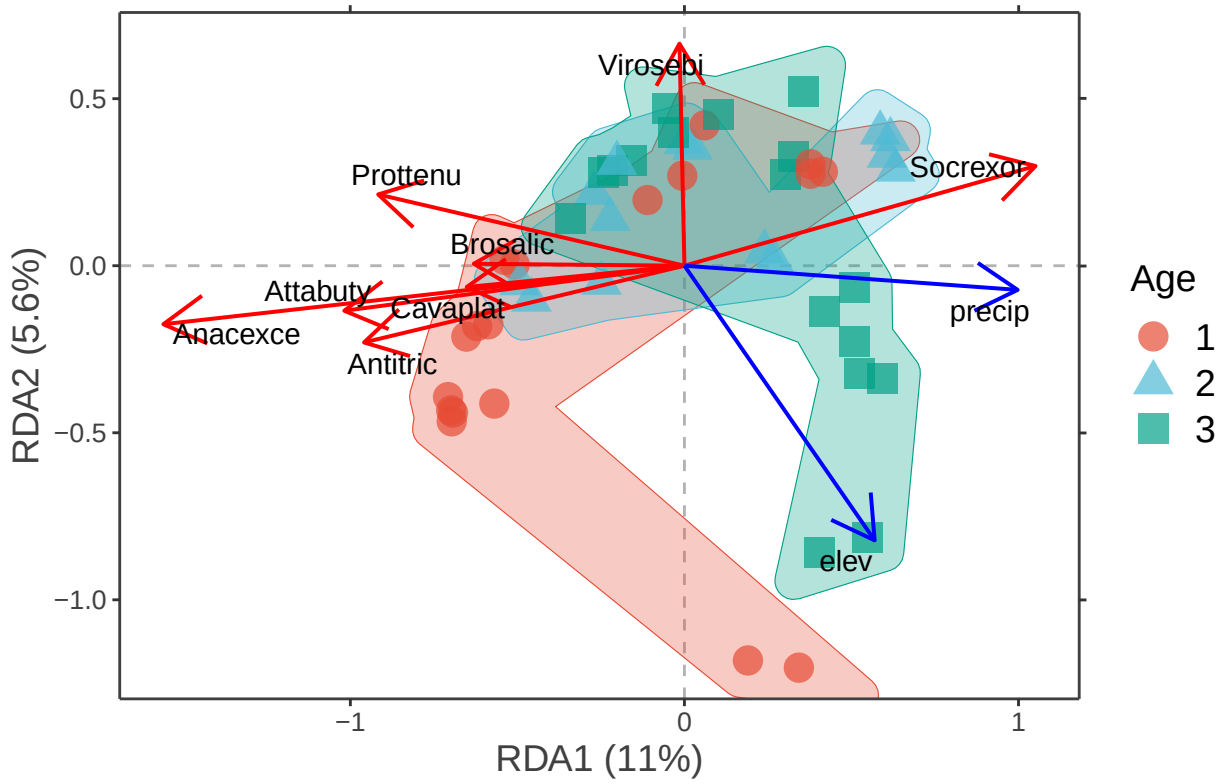
Now it is possible to generate ordination graphs, using similar scripts as above.

```
plotgg5 <- ggplot() +
    geom_vline(xintercept = c(0), color = "grey70", linetype = 2) +
    geom_hline(yintercept = c(0), color = "grey70", linetype = 2) +
    xlab(axis.long3[1, "label"]) +
    ylab(axis.long3[2, "label"]) +
    scale_x_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    scale_y_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    geom_mark_hull(data=sites.long3,
                   aes(x=axis1, y=axis2, colour=Age.cat,
                       fill=after_scale(alpha(colour, 0.2))),
                   concavity=0.1, size=0.2, show.legend=FALSE) +
    geom_point(data=sites.long3,
               aes(x=axis1, y=axis2, colour=Age.cat, shape=Age.cat),
               alpha=0.7, size=5) +
    geom_segment(data=species.long3,
                 aes(x=0, y=0, xend=axis1*6, yend=axis2*6),
                 colour="red", size=0.7, arrow=arrow()) +
    geom_text_repel(data=species.long3,
                    aes(x=axis1*6, y=axis2*6, label=labels),
                    colour="black") +
```

```
    geom_segment(data=subset(vectors.long3, vector %in% c("precip", "elev")),
                 aes(x=0, y=0, xend=axis1*1, yend=axis2*1),
                 colour="blue", size=0.7, arrow=arrow()) +
    geom_text_repel(data=subset(vectors.long3, vector %in% c("precip", "elev")),
                    aes(x=axis1*1, y=axis2*1, label=vector),
                    colour="black") +
    BioR.theme +
    ggsci::scale_colour_npg() +
    labs(shape="Age", colour="Age", fill="Age") +
    coord_fixed(ratio=1)
```

This is now the resulting plot, where also vectors were added for continuous explanatory variables.

```
plotgg5
```



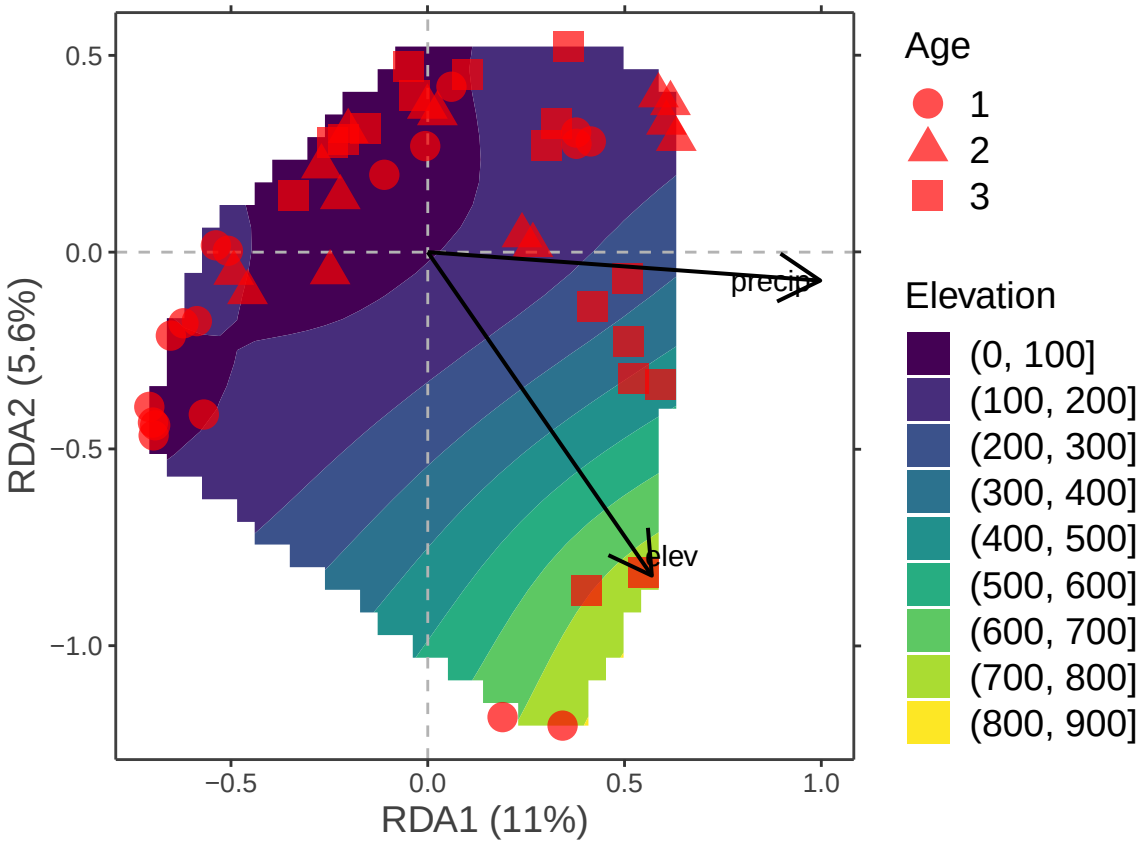
The smoothed surface for elevation is obtained by this script:

```
plotgg5 <- ggplot() +
    geom_contour_filled(data=elev.grid,
                        aes(x=x, y=y, z=z)) +
    geom_vline(xintercept = c(0), color = "grey70", linetype = 2) +
    geom_hline(yintercept = c(0), color = "grey70", linetype = 2) +
    xlab(axis.long3[1, "label"]) +
    ylab(axis.long3[2, "label"]) +
    scale_x_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    scale_y_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    geom_point(data=sites.long3,
```

```
              aes(x=axis1, y=axis2, shape=Age.cat),
              colour="red", alpha=0.7, size=5) +
    geom_segment(data=subset(vectors.long3, vector %in% c("precip", "elev")),
                 aes(x=0, y=0, xend=axis1*1, yend=axis2*1),
                 colour="black", size=0.7, arrow=arrow()) +
    geom_text_repel(data=subset(vectors.long3, vector %in% c("precip", "elev")),
                    aes(x=axis1*1, y=axis2*1, label=vector),
                    colour="black") +
    BioR.theme +
    scale_fill_viridis_d() +
    labs(shape="Age", fill="Elevation") +
    coord_fixed(ratio=1)

plotgg5
```

```
## Warning: Removed 251 rows containing non-finite values (stat_contour_filled).
```



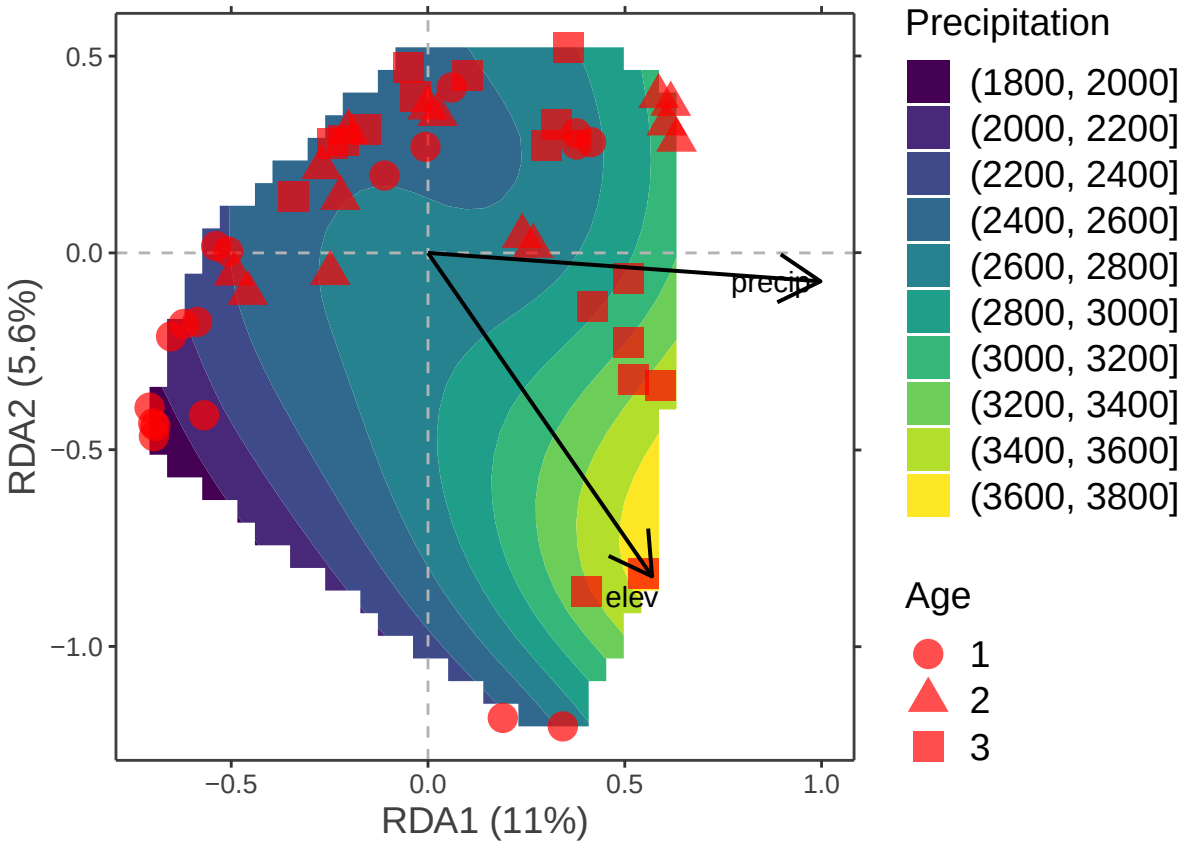
... and the smoothed surface for Precipitation by this script:

```
plotgg5 <- ggplot() +
    geom_contour_filled(data=precip.grid,
                        aes(x=x, y=y, z=z)) +
    geom_vline(xintercept = c(0), color = "grey70", linetype = 2) +
    geom_hline(yintercept = c(0), color = "grey70", linetype = 2) +
    xlab(axis.long3[1, "label"]) +
    ylab(axis.long3[2, "label"]) +
    scale_x_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
```

```
    scale_y_continuous(sec.axis = dup_axis(labels=NULL, name=NULL)) +
    geom_point(data=sites.long3,
               aes(x=axis1, y=axis2, shape=Age.cat),
               colour="red", alpha=0.7, size=5) +
    geom_segment(data=subset(vectors.long3, vector %in% c("precip", "elev")),
                 aes(x=0, y=0, xend=axis1*1, yend=axis2*1),
                 colour="black", size=0.7, arrow=arrow()) +
    geom_text_repel(data=subset(vectors.long3, vector %in% c("precip", "elev")),
                    aes(x=axis1*1, y=axis2*1, label=vector),
                    colour="black") +
    BioR.theme +
    scale_fill_viridis_d() +
    labs(shape="Age", fill="Precipitation") +
    coord_fixed(ratio=1)

plotgg5
```

```
## Warning: Removed 251 rows containing non-finite values (stat_contour_filled).
```



Some Other Examples

In my experience, the key to plot ordination results with ggplot2 is to get the data in the long format, whereas the layout and composition of the graphs can be altered via ggplot2. This can be achieved with the functions that I showed above such as sites.long and species.long (and achieved even more easily via **BiodiversityRGUI**). But alternative pathways work as well of course, as in the following examples:

- Vegan cheat sheet from Ann Bui
- NMDS Ordination from Rebekah Grieger
- Ecological Diversity from Ro Allen

Session Information

```
sessionInfo()
```

```
## R version 4.0.2 (2020-06-22)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 18363)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United Kingdom.1252
## [2] LC_CTYPE=English_United Kingdom.1252
## [3] LC_MONETARY=English_United Kingdom.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United Kingdom.1252
##
## attached base packages:
## [1] tcltk     stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] ggforce_0.3.2        ggrepel_0.8.2        ggsci_2.9
##  [4] readxl_1.3.1         ggplot2_3.3.2        BiodiversityR_2.12-2
##  [7] rgl_0.100.54         vegan3d_1.1-2        vegan_2.5-6
## [10] lattice_0.20-41      permute_0.9-5
##
## loaded via a namespace (and not attached):
##   [1] minqa_1.2.4             colorspace_1.4-1        ellipsis_0.3.1
##   [4] class_7.3-17            rio_0.5.16              htmlTable_2.0.0
##   [7] base64enc_0.1-3         rstudioapi_0.11         farver_2.0.3
##  [10] splines_4.0.2           knitr_1.28              effects_4.1-4
##  [13] polyclip_1.10-0         Formula_1.2-3           jsonlite_1.6.1
##  [16] nloptr_1.2.2.1          cluster_2.1.0           Rcmdr_2.6-2
##  [19] png_0.1-7               shiny_1.4.0.2           compiler_4.0.2
##  [22] backports_1.1.7         Matrix_1.2-18           fastmap_1.0.1
##  [25] survey_4.0              tweenr_1.0.1            later_1.1.0.1
##  [28] tcltk2_1.2-11           acepack_1.4.1           htmltools_0.5.0
##  [31] tools_4.0.2             gtable_0.3.0            glue_1.4.1
##  [34] dplyr_1.0.2             V8_3.4.0                Rcpp_1.0.4.6
##  [37] carData_3.0-4           cellranger_1.1.0        vctrs_0.3.4
##  [40] nlme_3.1-148            crosstalk_1.1.0.1       xfun_0.15
##  [43] stringr_1.4.0           openxlsx_4.1.5          lme4_1.1-23
##  [46] mime_0.9                miniUI_0.1.1.1          lifecycle_0.2.0
##  [49] RcmdrMisc_2.7-0         statmod_1.4.34          MASS_7.3-51.6
##  [52] zoo_1.8-8               scales_1.1.1            hms_0.5.3
##  [55] promises_1.1.1          parallel_4.0.2          sandwich_2.5-1
##  [58] RColorBrewer_1.1-2      yaml_2.2.1              curl_4.3
```

```
##  [61] gridExtra_2.3           rpart_4.1-15            latticeExtra_0.6-29
##  [64] stringi_1.4.6           nortest_1.0-4           e1071_1.7-3
##  [67] checkmate_2.0.0         boot_1.3-25             zip_2.0.4
##  [70] manipulateWidget_0.10.1 rlang_0.4.8             pkgconfig_2.0.3
##  [73] evaluate_0.14           purrr_0.3.4             labeling_0.3
##  [76] htmlwidgets_1.5.1       tidyselect_1.1.0        magrittr_1.5
##  [79] R6_2.4.1                generics_0.1.0          Hmisc_4.4-0
##  [82] DBI_1.1.0               pillar_1.4.4            haven_2.3.1
##  [85] foreign_0.8-80          withr_2.2.0             mgcv_1.8-31
##  [88] survival_3.1-12         scatterplot3d_0.3-41    abind_1.4-5
##  [91] nnet_7.3-14             tibble_3.0.1            crayon_1.3.4
##  [94] car_3.0-8               relimp_1.0-5            rmarkdown_2.3
##  [97] jpeg_0.1-8.1            isoband_0.2.1           grid_4.0.2
## [100] data.table_1.12.8       forcats_0.5.0           digest_0.6.25
## [103] webshot_0.5.2           xtable_1.8-4            httpuv_1.5.4
## [106] munsell_0.5.0           viridisLite_0.3.0       concaveman_1.1.0
## [109] mitools_2.4
```