Table of Contents

Section: Ordination analysis

RDA, tb-RDA, CCA & db-RDA (constrained ordination)

Theory R functions **Examples** Exercise

Example 1: Vltava river valley dataset (tb-RDA)

In this example, we will apply constrained ordination (tb-RDA) on Vltava river valley dataset. We will ask how much variance in species composition can be explained by two variables, soil pH and soil depth. Both are important factors for plant growth, and moreover, in the study area, they are somewhat correlated (shallower soils have lower pH since the prevailing geological substrate is acid).

First, upload the Vltava river valley data:

```
vltava.spe <- read.delim
('https://raw.githubusercontent.com/zdealveindy/anadat-r/master/data/vltava-
spe.txt', row.names = 1)
vltava.env <- read.delim
('https://raw.githubusercontent.com/zdealveindy/anadat-r/master/data/vltava-
env.txt')

spe <- vltava.spe  # rename variables to make them shorter
env <- vltava.env[, c('pH', 'SOILDPT')]  # select only two explanatory
variables
```

Upload library vegan and calculate tb-RDA based on Hellinger pre-transformed species composition data. Note that since the original data represent estimates of percentage cover, it is better to log transform these values first before Hellinger transformation is done (using function `log1p`, which calculates *log (x+1)* to avoid *log (0)*):

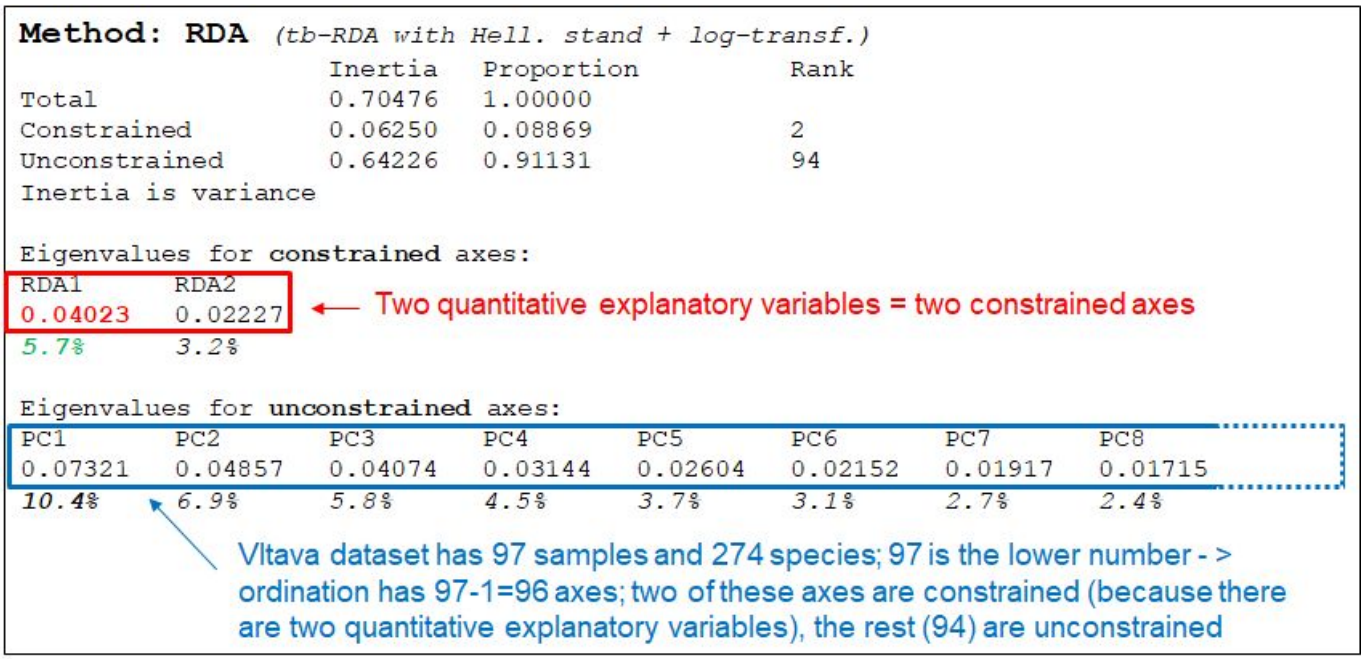
```
library (vegan)
spe.log <- log1p (spe)  # species data are in percentage scale which is
strongly rightskewed, better to transform them
spe.hell <- decostand (spe.log, 'hell')  # we are planning to do tb-RDA,
this is Hellinger pre-transformation
tbRDA <- rda (spe.hell ~ pH + SOILDPT, data = env)  # calculate tb-RDA with
two explanatory variables
tbRDA
```

The result printed by `rda` function is the following:

```
Call: rda(formula = spe.hell ~ pH + SOILDPT, data = env)

              Inertia Proportion Rank
Total         0.70476    1.00000
Constrained   0.06250    0.08869    2
Unconstrained 0.64226    0.91131   94
```

```
Inertia is variance

Eigenvalues for constrained axes:
   RDA1    RDA2
0.04023 0.02227

Eigenvalues for unconstrained axes:
    PC1     PC2     PC3     PC4     PC5     PC6     PC7     PC8
0.07321 0.04857 0.04074 0.03144 0.02604 0.02152 0.01917 0.01715
(Showed only 8 of all 94 unconstrained eigenvalues)
```

and, the same with comments:



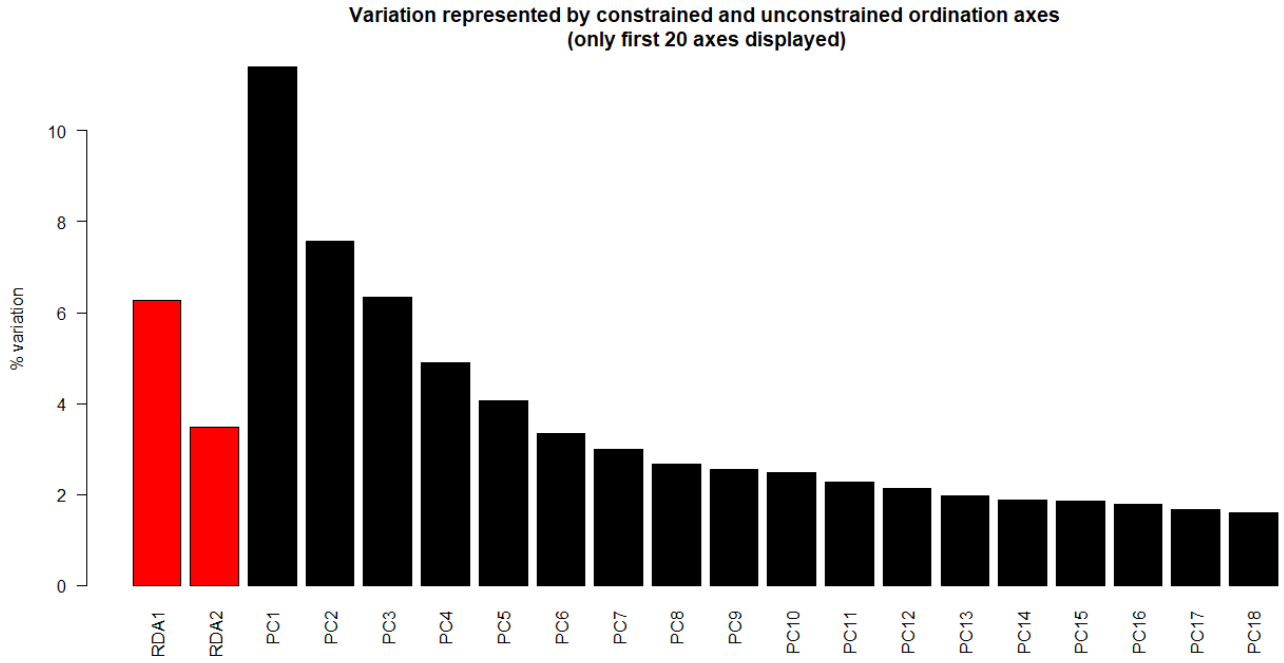
The two variables explain 8.9% of the variance (the row `Constrained` and column `Proportion` in the table above, can be calculated also as the sum of eigenvalues for the constrained axes divided by total variance (inertia): (0.04023+0.02227) /0.70476=0.08869. The first constrained axis (RDA1) explains 0.04023/0.70476=5.7% of the variance, while the second (RDA2) explains 0.02227/0.70476=3.2%. Note that the first unconstrained axis (PC1) represents 0.07321/0.70476=10.4% of total variance, which is more than both explanatory variables together; the first two unconstrained explain (0.07321+0.04857)/0.70476=17.3%. This means that the dataset may be structured by some strong environmental variable(s) different from pH and soil depth (we will check this below).

The relationship between the variation represented by individual (constrained and unconstrained) ordination axes can be displayed using the barplot on eigenvalues:

```
constrained_eig <- tbRDA$CCA$eig/tbRDA$CA$tot.chi*100
unconstrained_eig <- tbRDA$CA$eig/tbRDA$CA$tot.chi*100
barplot (c(constrained_eig, unconstrained_eig), col = c(rep ('red', length
(constrained_eig)), rep ('black', length (unconstrained_eig))), las = 2,
ylab = '% variation')
```

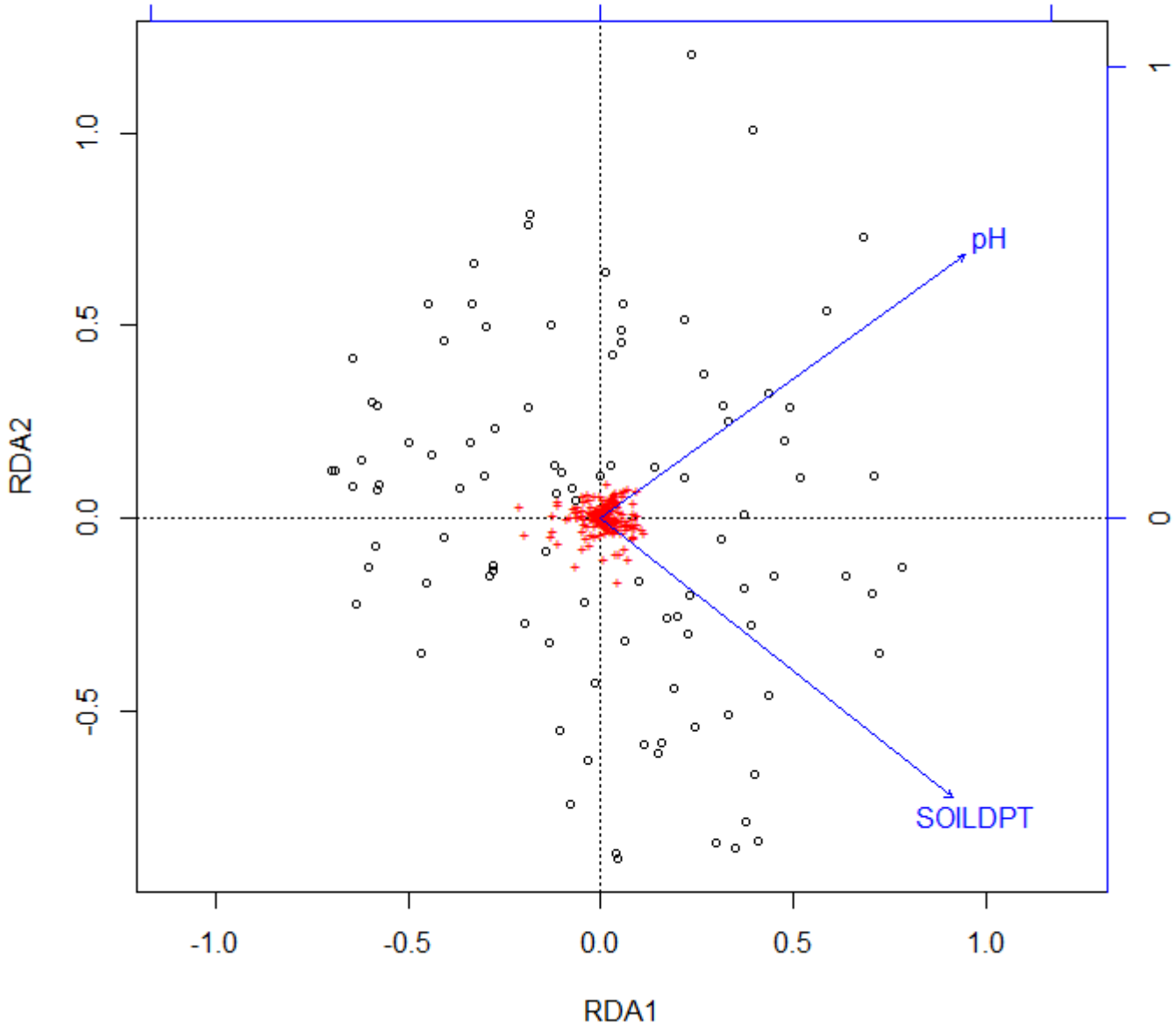
(note that all information about the eigenvalues and total inertia is in the object calculated by

vegan's ordination function (`rda` in this case, stored in the list `tbRDA`), you just need to search a bit inside to find it - consider using the function `str` to check the structure of tbRDA first).



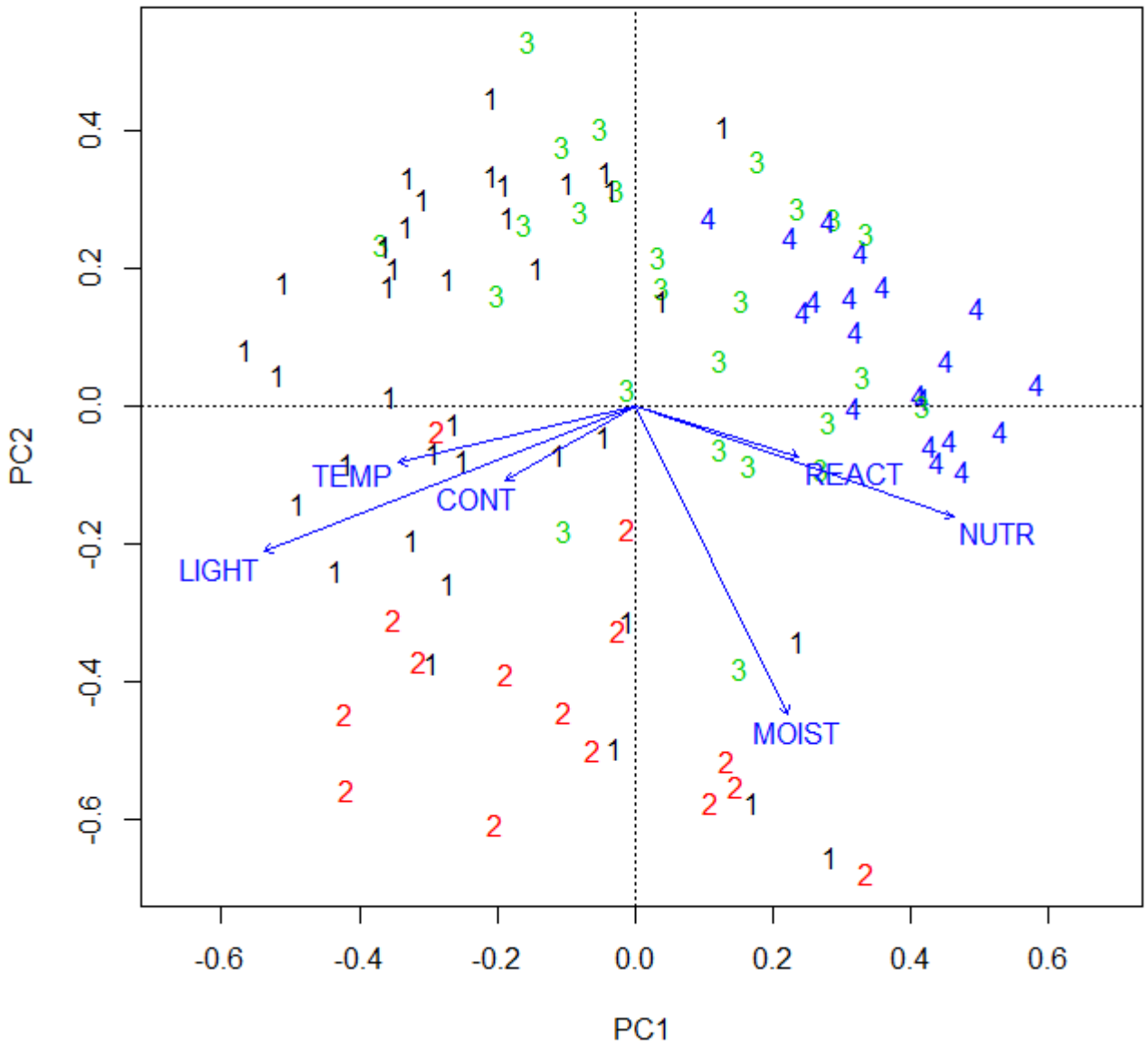
Let's see the ordination diagram:

```
ordiplot (tbRDA)
```



What may be those environmental variables associated with unconstrained axes? The `vltava.env` dataset contains a number of other measured variables which we may fit as supplementary to the first and second unconstrained axis to see which of them is most related to which of them. But here we will do an alternative thing: we will use mean Ellenberg indicator values (mEIV) calculated for each plot based on the species composition and tabulated Ellenberg species indicator values (ecological optima of species along several main environmental gradients). This approach will illustrate the situation as if in the field we measured only soil pH and depth (relatively easily obtained variables), and we use these indirect estimates to get an idea about which other factors may be important.

```
ordiplot (tbRDA, choices = c(3,4), type = 'n')
points (tbRDA, choices = c(3,4), display = 'sites', pch = as.character
(vltava.env$GROUP), col = vltava.env$GROUP)
ef <- envfit (tbRDA, vltava.env[,23:28], choices = c(3,4), permutations = 0)
plot (ef)
```




```
ef
```

```
***VECTORS

           PC1      PC2     r2
LIGHT -0.93135 -0.36411 0.6282
TEMP  -0.97246 -0.23305 0.2352
CONT  -0.86643 -0.49929 0.0885
MOIST  0.44495 -0.89556 0.4706
REACT  0.95614 -0.29291 0.1166
NUTR   0.94383 -0.33044 0.4519
```

The highest R2 of regression with the first two axes have light and moisture, with light associated mostly with the first axis and the moisture mostly with the second. Seems that these two ecological factors, not related to soil pH and soil depth, are important for vegetation but not measured; light

passing through the canopy of the forest has strong effect on the species composition of the herb understory (herbs makes most of the species in this analysis, since temperate forest is rather poor for woody species), and moisture also (flooded alluvial forests at the bottom parts of the valley, veg. type 2, have very different species composition from dry open forests on the upper parts of the valley slopes).

Note one more thing: when applying the function `envfit` on mean Ellenberg indicator values, I did not test for the significance (I set the argument `permutation = 0`). This has a meaning: both mean Ellenberg indicator values and sample scores on ordination axes are calculated from the same matrix of species composition, and to directly test their relationship would be wrong (they are not independent, and we get high probability to get significant result even if the species Ellenberg indicator values are randomly generated). Check the section Analysis of species attributes (e.g. traits or species indicator values) for detail explanation on how to solve this Fix Me!.

Example 2: How different are cookies from pastries and pizzas (CCA)?

This example is using Difference between cookies, pastries and pizzas dataset, which I found on reddit.com, posted by author everest4ever. As the post on reddit.com goes, the author attended the Christmas party at his office with "Christmass cookie competition", which sparked a "huge debate about what are eligible entries for the cookie competition (e.g. are mini-pizzas cookies?)". The author decided to approach the discussion rigorously and did the following: "I scraped 1931 recipes from the Food Network that contain the keywords cookies (my group of interest), pastry, or pizza (two control groups). Next, I extracted the ingredient list and pooled similar ingredients together (e.g. *salt*, *seasalt*, *Kosher salt*), coming up with a total of 133 unique ingredients. I ended up with a 1931×133 matrix, where each row is one recipe, and each column is whether this recipe contains a certain ingredient (0 or 1)". The author did PCA analysis on the data accompanied by some clustering and predictions, just to prove that "NO IAN AND JOSEPH YOUR FUCKING EGG TARTS AREN'T COOKIES, NO MATTER HOW GOOD THEY WERE!!". I think we can also use this dataset for a simple constrained ordination exercise. First import the data:

```
recipes.ingr <- read.delim
('https://raw.githubusercontent.com/zdealveindy/anadat-r/master/data/cookie_
dataset_everest4ever_composition.txt', row.names = 1)
recipes.type <- read.delim
('https://raw.githubusercontent.com/zdealveindy/anadat-r/master/data/cookie_
dataset_everest4ever_type.txt', row.names = 1)
```

Data represent a matrix of presence/absence of different ingredients in individual recipes (each row of `recipes.ingr` matrix is a recipe, each column one ingredients). To know which recipe is classified how, we need a variable `type_of_food` in the data frame `recipes.type`.

To get familiar with data, let's first calculate DCA:

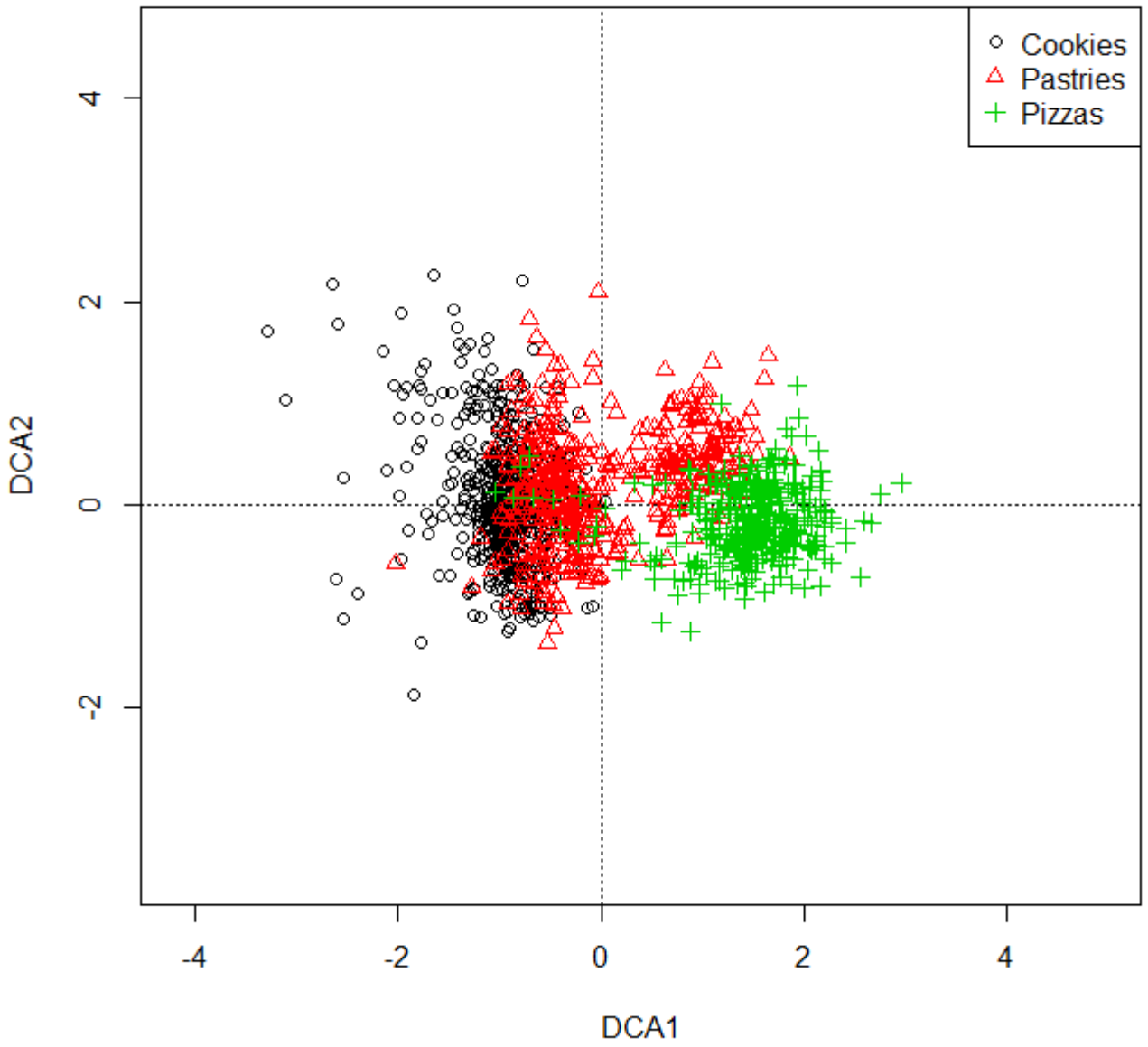
```
library (vegan)
DCA <- decorana (recipes.ingr)
DCA
```

```
Call:
decorana(veg = recipes.ingr)
```

```
Detrended correspondence analysis with 26 segments.
Rescaling of axes with 4 iterations.

                  DCA1   DCA2   DCA3   DCA4
Eigenvalues     0.5633 0.2598 0.2224 0.2185
Decorana values 0.5918 0.2622 0.2352 0.2138
Axis lengths    6.2276 4.1302 5.6396 3.5449
```

The output shows that the length of the first axis is 6.2 S.D. units, so unimodal ordination methods is advisable. The ordination diagram which displays recipes of cookies, pastries and pizzas by different symbols and colours, is more informative:

```
type_num <- as.numeric (recipes.type$type_of_food)

ordiplot (DCA, type = 'n')
points (DCA, display = 'sites', col = type_num, pch = type_num)
legend ('topright', col = 1:3, pch = 1:3, legend = levels
(recipes.type$type_of_food))
```



(the variable `type_num` contains numerical values 1, 2 and 3 in place of Cookies, Pastries and Pizzas from the original `type_of_food` variable in `recipes.type` data frame, so as we can use these values as colors and symbols in ordination diagram).

It seems that pizzas are somewhat different from the rest (although part of pastries is close), while pastries and cookies form a cloud with big overlap. Let's try to ask the following question: can the classification of a recipe into cookies/pastries/pizzas (done largely subjectively by authors of that recipes based on their opinion how each category item should look like) explain the difference in "ingredients composition" of individual recipes? This is task for constrained ordination. Since the first DCA axis is long, we use CCA for it, with recipes dependent variable and assignment into the type as explanatory. Note that explanatory variable is categorical with three levels (`pastry`, `cookie`, `pizza`):

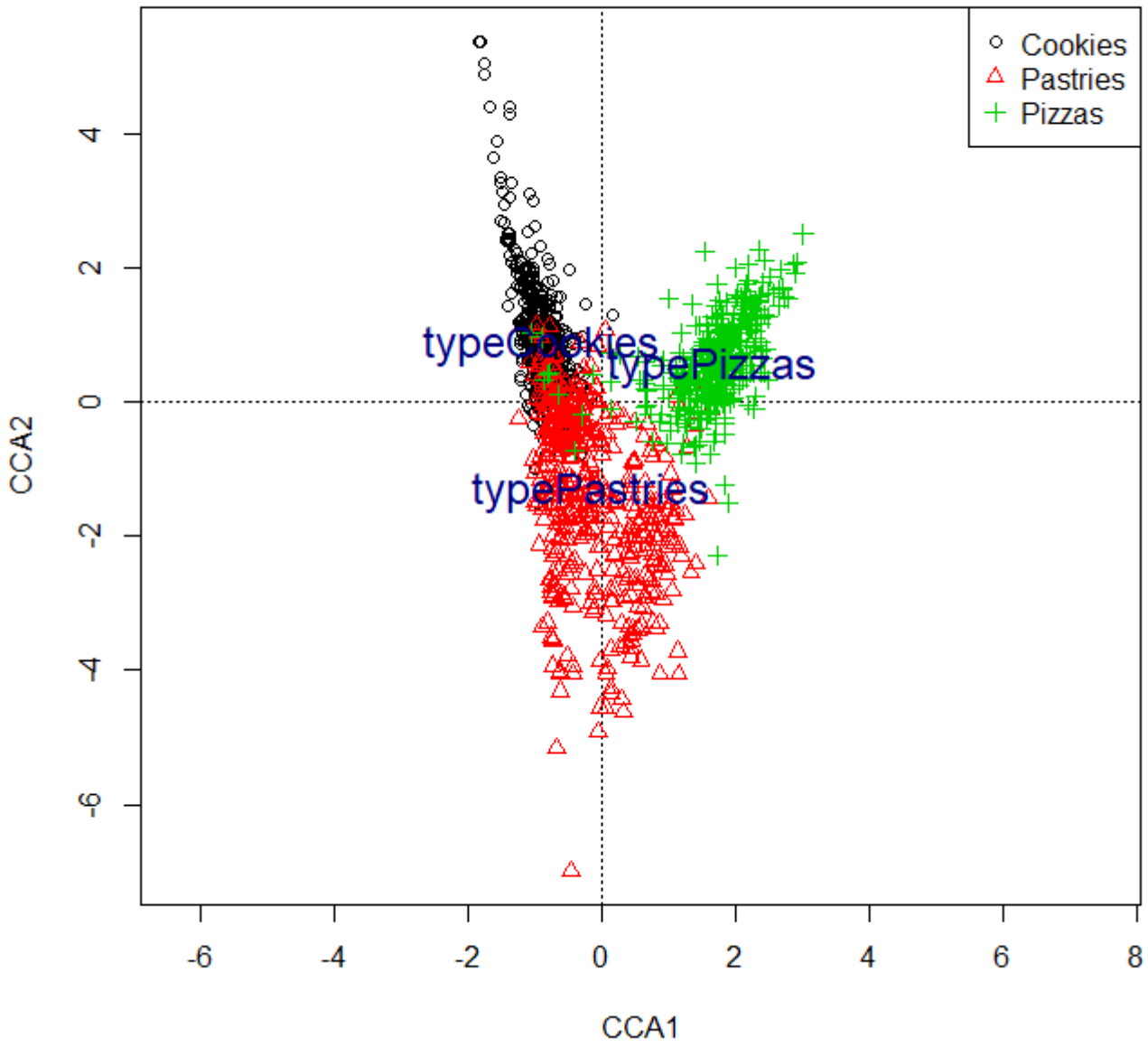
```
type <- recipes.type$type_of_food
CCA <- cca (recipes.ingr ~ type)
CCA
```

```
Call: cca(formula = recipes.ingr ~ type)

              Inertia Proportion Rank
Total         14.28961    1.00000
Constrained    0.60311    0.04221    2
Unconstrained 13.68650    0.95779  132
Inertia is scaled Chi-square

Eigenvalues for constrained axes:
  CCA1   CCA2
0.4649 0.1382

Eigenvalues for unconstrained axes:
    CA1     CA2     CA3     CA4     CA5     CA6     CA7     CA8
0.30447 0.28409 0.26249 0.25278 0.23942 0.21521 0.21069 0.20674
(Showed only 8 of all 132 unconstrained eigenvalues)
```

(note that I saved the column `type_of_food` from the data frame `recipes.type` into a variable `type`, not really because I want to simplify the `cca` (it would need to be `CCA <- cca (recipes.ingr ~ type_of_food, data = recipes.type)`, but that is still fine), but because this will make it simple to display individual factor levels onto ordination diagram (the levels are displayed as the Name_of_variableName_of_category, which would be too long with the original names).

We got two constrained axes (explanatory variable is qualitative with three factor levels -> number of contrained axes = number of levels - 1), the first exlaining more than 3 time more than the second (eig_{CCA1} = 0.4649, eig_{CCA2} = 0.1382, which means that the first axis represents 0.4649/14.28961 = 3.3 % of variance (eigenvalue/total inertia), while the second 0.1382/14.28961 = 1.0%. Ordination diagram shows that the first axis is mostly separating pizzas (right) and cookies+pastries (left), while the second axis is mostly separating cookies (up) from pastries (bottom):

```
ordiplot (CCA, display = c('si', 'cn'), type = 'n')
points (CCA, display = 'si', col = type_num, pch = type_num)
text (CCA, display = 'cn', col = 'navy', cex = 1.5)
legend ('topright', col = 1:3, pch = 1:3, legend = levels
(recipes.type$type_of_food))
```



Few more things. First, we should ask whether the CCA ordination is significant and whether it is worth to interpret it:

```
anova (CCA)
```

```
Permutation test for cca under reduced model
Permutation: free
Number of permutations: 999

Model: cca(formula = recipes.ingr ~ type)
          Df ChiSquare      F Pr(>F)
Model      2    0.6031 42.479  0.001 ***
Residual 1928   13.6865
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

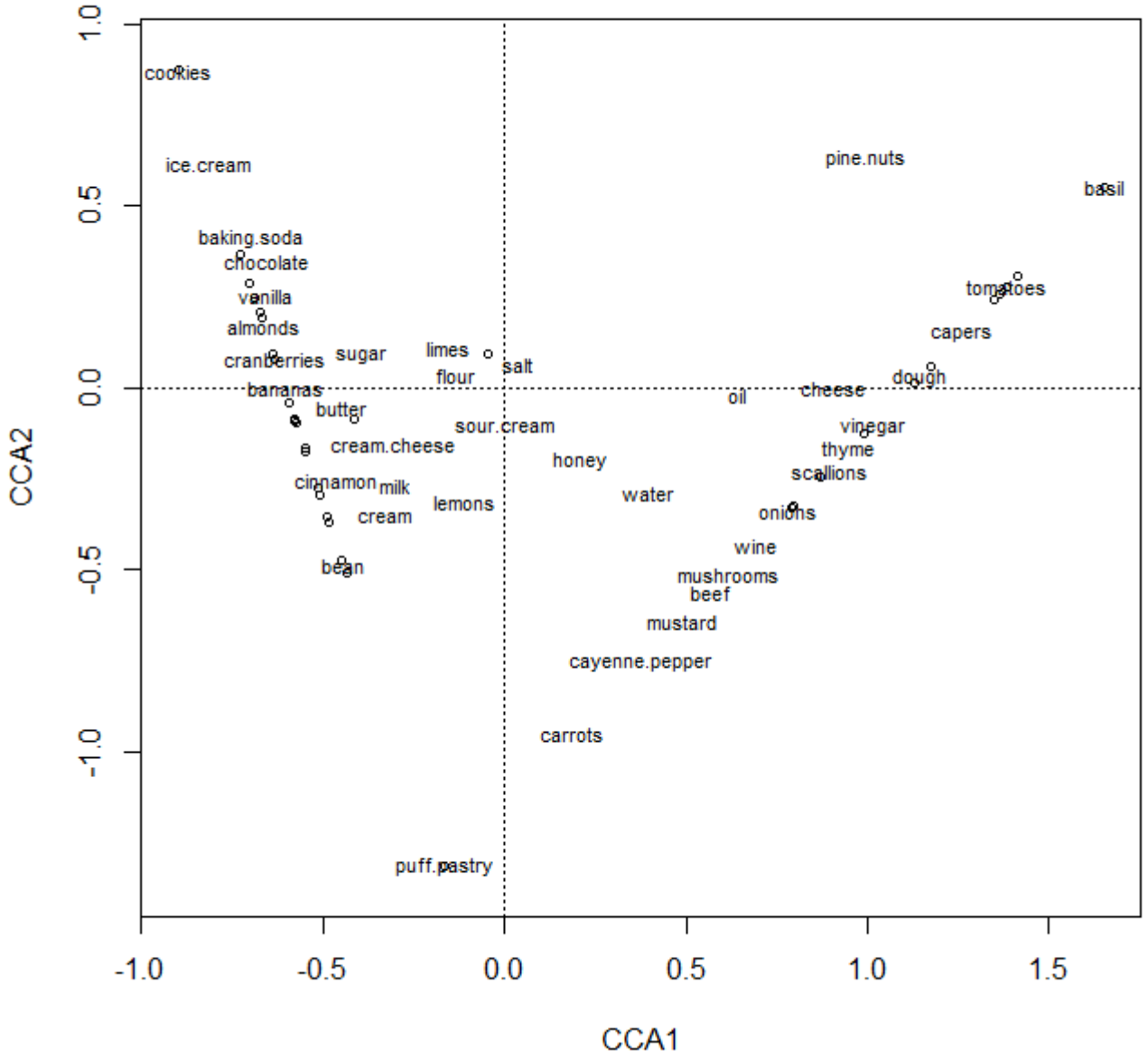
Indeed, it is. And how about individual axes, are they both significant? We will use the argument by = "axis" in the function anova - see this explanation what it means:

```
anova (CCA, by = 'axis')
```

```
Permutation test for cca under reduced model
Forward tests for axes
Permutation: free
Number of permutations: 999

Model: cca(formula = recipes.ingr ~ type)
           Df ChiSquare      F Pr(>F)
CCA1        1    0.4649 65.493  0.001 ***
CCA2        1    0.1382 19.465  0.001 ***
Residual 1928   13.6865
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Both axes are significant, which means that even the distinction between cookies and pastries along the second axis is important. So I would say, without further testing, that not only pizza is (quite obviously) different from cookies, but also much more ambiguous category pastries are different regarding ingredients they use. Btw, let's see these ingredients (display species in CCA ordination diagram):

```
ordiplot (CCA, display = c('sp', 'cn'), type = 'n')
orditorp (CCA, display = 'sp', priority = colSums (recipes.ingr))
```



Note that I did not display all 133 ingredients (“species”); otherwise the diagram gets too cluttered. I used the low-level graphical function `orditorp` which is adding only some labels and draws others as symbols. It has argument `priority` (the species with the highest priority will be more likely plotted as text, with lower as symbols, if there is not enough space); the priority here is the overall frequency of ingredence in the dataset (`colSums` applied on the `recipes.ingr` data frame). The diagram shows clear triangle, with each corner representing one type of food. There is a gradient of ingredients connecting pizzas with pastries and pastries with cookies, but almost no ingredients connecting pizzas and cookies (except `pine nuts`, which can perhaps make it in both pizzas and cookies - but I have no idea). Some ingredients are shared among all three (obviously *water* and seems that also *honey*), some are only for that type of food (*basil* for pizza, *puff pastry* for pastries and *cookies*(??) and *ice cream* for cookies. Please, see the diagram and guess which item in your opinion should be where (*carrots* in pastry? not sure...).