Variable selection (constrained ordination)

Theory R functions **Examples**

Example 1: tb-RDA with forward selection: Which chemical variables measured in fen water are important for vegetation of Carpathian wetlands?

We will use data about species composition of vascular plants in Carpathian wetlands, and ask which chemical variables, measured in fen water running through these wetlands, are the most important to explain the compositional changes.

First, prepare the data:

```
# Carpathian wetlands - import data
vasc <- read.delim
('https://raw.githubusercontent.com/zdealveindy/anadat-r/master/data/vasc_plants.tx
t', row.names = 1)
chem <- read.delim
('https://raw.githubusercontent.com/zdealveindy/anadat-
r/master/data/chemistry.txt', row.names = 1)

# transform data using Hellinger transformation to prepare them for tb-RDA:
library (vegan)
vasc.hell <- decostand (vasc, 'hell')

# the last variable in the 'chem' dataset is 'slope', which is not a chemical
variable - remove it:
chem1 <- chem[,-15]
```

Here, we will use transformation-based RDA ordination (tb-RDA). First, we need to test the global model (including all explanatory variables from which you plan to do selection) to see whether it is significant; if it is not, the forward selection should not be done.

```
tb_rda.all <- rda (vasc.hell ~ ., chem1)
anova (tb_rda.all)  # 0.001 *** - it is significant
adjR2.tbrda <- RsquareAdj (tb_rda.all)$adj.r.squared # 0.2011048 - adjusted R2
explained by all 14 variables is 20.1%
```

Option a) Use of forward.sel function

The function `forward.sel` from the package `adespatial` is an elaborated forward selection approach based on linear constrained ordination (RDA). If you want to calculate CCA, use `ordiR2step` or `ordistep` from `vegan` instead (see below). Along with the testing significance of the selected variable, this function also includes other stopping rules, published by Blanchet et al. (2008). The method based only on significance is prone to being inflated, which may result in models with too many variables; Blanchet et al. (2008) came up with another set of stopping rules along with the significance one. First, the adjusted R^2 of the model including all variables (from which the selection is made) is calculated. While selecting variables during the selection process, the selection stops if (1) the next selected variables is not significant (does not significantly improve the explained variation), and (2) if the R^2_{adj} of the model including this variable exceeds the R^2_{adj} of the global model.

```
# install.packages ('adespatial')  # run this code if you haven't install the
adespatial package before
library (adespatial)
sel.fs <- forward.sel (Y = vasc.hell, X = chem1, adjR2thresh = adjR2.tbrda)  #
NOTE: forward.sel calculates only RDA (or tb-RDA in case of Hell. transformed data)
```

```
Testing variable 1
Testing variable 2
Testing variable 3
Testing variable 4
Testing variable 5
Testing variable 6
Testing variable 7
Testing variable 8
Procedure stopped (alpha criteria): pvalue for variable 8 is 0.118000 (> 0.050000)
```

See the table with selected variables saved in `sel.fs` object:

```
sel.fs
```

```
  variables order         R2     R2Cum  AdjR2Cum         F  pval
1        Ca     1 0.13898260 0.1389826 0.1263206 10.976337 0.001
2   conduct    14 0.03248342 0.1714660 0.1467337  2.626796 0.001
3        Si     6 0.02722659 0.1986926 0.1622696  2.242529 0.001
4       NH3    10 0.02398868 0.2226813 0.1748463  2.005953 0.002
5       NO3     9 0.02112004 0.2438013 0.1847233  1.787470 0.001
6        Mg     2 0.01914995 0.2629513 0.1927562  1.636862 0.003
7        pH    13 0.01744989 0.2804012 0.1991562  1.503467 0.024
```

Or, if you want to see only the names of selected variables, subset it from the `sel.fs` object:

```
sel.fs$variables
```

```
[1] "Ca"      "conduct" "Si"      "NH3"     "NO3"     "Mg"      "pH"
```

Since there are potentially many tests of significance in the forward selection procedure, it is better to apply a correction for multiple testing. The most common correction (but rather conservative) is Bonferroni, which multiplies each calculated *P*-value by the overall number of tests. Less conservative corrections include Holm's and the False Discovery Rate (FDR). In forward selection, the overall number of tests equals the number of variables from which the selection is made (i.e., not the number of selected variables), and this should be considered while applying the adjustment.

The function available for adjusting the *P*-values is `p.adjust`, which has the argument n indicating the number of all possible tests (use the argument and include the correct number of all possible tests!).

The *P*-values from the forward selection above are in the object `sel.fs$pval`, and the adjustment can be done in the following way (using Holm's correction):

```
n.tests <- ncol (chem1)  # number of tests equals the number of all variables from
which are being selected
pval.adj <- p.adjust (sel.fs$pval, method = 'holm', n = n.tests)
sel.fs$pval.adj <- pval.adj
sel.fs
```

```
  variables order         R2     R2Cum  AdjR2Cum         F  pval pval.adj
```

```
1        Ca     1 0.13898260 0.1389826 0.1263206 10.976337 0.001    0.014
2   conduct    14 0.03248342 0.1714660 0.1467337  2.626796 0.001    0.014
3        Si     6 0.02722659 0.1986926 0.1622696  2.242529 0.001    0.014
4       NH3    10 0.02398868 0.2226813 0.1748463  2.005953 0.003    0.033
5       NO3     9 0.02112004 0.2438013 0.1847233  1.787470 0.005    0.050
6        Mg     2 0.01914995 0.2629513 0.1927562  1.636862 0.014    0.126
7        pH    13 0.01744989 0.2804012 0.1991562  1.503467 0.020    0.160
```

After the correction, the last two variables (Mg and pH) are no longer significant. Even the first selected variable has the significance $P < 0.05$. The reason is the number of permutations used in the `forward.sel` function, which by default equals 999 (argument `nperm`). The lowest *P*-value which can be obtained with 999 permutations is 1/(999+1) = 0.001, and the multiple testing correction will considerably increase this value (to 0.014 in case of 14 variables). The workaround is to increase the number of permutations to obtain lower *P*-values, which, even after correction, could still fall below the 0.001 threshold (here, using 49,999 permutations to achieve a P-value of 1/50000 = 0.00002).

```
sel.fs <- forward.sel (vasc.hell, chem1, nperm = 49999)
pval.adj <- p.adjust (sel.fs$pval, method = 'holm', n = ncol (chem1))
sel.fs$pval.adj <- pval.adj
sel.fs
```

```
  variables order         R2     R2Cum  AdjR2Cum         F    pval pval.adj
1        Ca     1 0.13898260 0.1389826 0.1263206 10.976337 0.00002  0.00028
2   conduct    14 0.03248342 0.1714660 0.1467337  2.626796 0.00004  0.00052
3        Si     6 0.02722659 0.1986926 0.1622696  2.242529 0.00016  0.00192
4       NH3    10 0.02398868 0.2226813 0.1748463  2.005953 0.00080  0.00880
5       NO3     9 0.02112004 0.2438013 0.1847233  1.787470 0.00304  0.03040
6        Mg     2 0.01914995 0.2629513 0.1927562  1.636862 0.00824  0.07416
7        pH    13 0.01744989 0.2804012 0.1991562  1.503467 0.02094  0.16752
```

Since the `forward.sel` function is not optimized, the calculation with 49,999 permutations takes some time, but the reward is the *P*-values (after correction) below 0.001 in case of the first two variables.

You can see that the `forward.sel` function, after the p-value correction, selected five environmental variables, which together explain 18.5% of the variance (adjusted R2, the column `AdjR2Cum` in the table above showing cumulative adjusted R2 of all variables). This is just slightly less than 20.1%, which is explained by the global model with 14 variables (see the result of `RsquareAdj (rda.all)$adj.r.squared` above).

In this case, the results of the `forward.sel` function are the same as the `ordiR2step` and `ordistep` functions below (options b and c), with seven included environmental variables (if calculated without correction for multiple testing).

Option b) Use of ordiR2step function

The `vegan`'s function `ordiR2step` does a similar job as `forward.sel` from `adespatial`, but its use is a bit more complex. On the other side, the method is more general - it allows the use of also CCA or db-RDA methods, while `forward.sel` is based purely on linear constrained ordination (RDA, and tb-RDA in the case that the species composition data are pre-transformed by Hellinger transformation). In the default setting the criteria for including the variable is based on both significance of the newly selected variables, and the comparison of adjusted variation (R^2_{adj}) explained by the selected variables to R^2_{adj} explained by the global model (with all variables); if the new variable is not significant *or* the R^2_{adj} of the model including this new variable would exceed the R^2_{adj} of the global model, the selection will be stopped.

To use the `ordiR2step` function, you need to specify two models - the “empty” model containing only the intercept, and the “full” model including all variables (aka global model). Also, the argument `direction` allows

for selecting the selection direction (`forward`, `backward`, `both`, with the default to `both`):

```
tb_rda.vasc.0 <- rda (vasc.hell ~ 1, data = chem1) # model containing only species
matrix and intercept
tb_rda.vasc.all <- rda (vasc.hell ~ ., data = chem1) # model including all
variables from matrix chem1 (the dot after tilda (~) means "include all from data")
```

The function `ordiR2step` then uses the empty model and the scope (variables) from the full model to proceed with the selection. Set `direction = 'forward' '' to ensure that forward instead of stepwise selection is done, and increase the number of permutations to 999 (instead of the default 499) to make results comparable to ''forward.sel` (which by default is based on 999 permutations):

```
sel.osR2 <- ordiR2step (tb_rda.vasc.0, scope = formula (tb_rda.vasc.all), R2scope =
adjR2.tbrda, direction = 'forward', permutations = 999)
```

```
Step: R2.adj= 0
Call: vasc.hell ~ 1

                 R2.adjusted
<All variables>  0.201104784
+ Ca             0.126320581
+ conduct        0.109418731
+ Mg             0.106926067
+ Corg           0.076260217
+ pH             0.074169805
+ Na             0.043030978
+ NH3            0.031875777
+ Si             0.022863900
+ SO4            0.018929801
+ K              0.015918981
+ NO3            0.008112568
+ Fe             0.004863012
+ PO4            0.002933349
<none>           0.000000000
+ Cl            -0.002347612

     Df     AIC      F Pr(>F)
+ Ca  1 -47.149 10.976  0.001 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Step: R2.adj= 0.1263206
Call: vasc.hell ~ Ca

                R2.adjusted
<All variables>   0.2011048
+ conduct         0.1467337
+ NH3             0.1417066
+ Na              0.1403837
+ Si              0.1403827
+ pH              0.1403289
+ Mg              0.1386000
+ Corg            0.1375824
+ NO3             0.1340217
```

```
+ K               0.1302515
+ Fe              0.1299960
+ SO4             0.1266017
<none>            0.1263206
+ Cl              0.1254155
+ PO4             0.1235944

          Df     AIC      F Pr(>F)
+ conduct  1 -47.841 2.6268  0.001 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

...
...
...

Step: R2.adj= 0.1991562
Call: vasc.hell ~ Ca + conduct + Si + NH3 + NO3 + Mg + pH

                R2.adjusted
+ Corg            0.2023822
+ Fe              0.2015249
+ K               0.2014550
+ SO4             0.2012822
<All variables>   0.2011048
<none>            0.1991562
+ Na              0.1988753
+ PO4             0.1974424
+ Cl              0.1974018
```

```
sel.osR2
```

```
Call: rda(formula = vasc.hell ~ Ca + conduct + Si + NH3 + NO3 + Mg + pH, data =
chem1)

              Inertia Proportion Rank
Total          0.5674     1.0000
Constrained    0.1591     0.2804    7
Unconstrained  0.4083     0.7196   62
Inertia is variance

Eigenvalues for constrained axes:
   RDA1    RDA2    RDA3    RDA4    RDA5    RDA6    RDA7
0.09590 0.01904 0.01205 0.01027 0.00871 0.00718 0.00595

Eigenvalues for unconstrained axes:
    PC1     PC2     PC3     PC4     PC5     PC6     PC7     PC8
0.03601 0.03279 0.02383 0.01908 0.01687 0.01593 0.01504 0.01358
(Showed only 8 of all 62 unconstrained eigenvalues)
```

The summary table is part of the `ordiR2step` function output, stored in the component `$anova`:

```
sel.osR2$anova
```

```
                 R2.adj Df     AIC       F Pr(>F)
+ Ca            0.12632  1 -47.149 10.9763  0.001 ***
```

```
+ conduct       0.14673  1 -47.841  2.6268  0.001 ***
+ Si            0.16227  1 -48.180  2.2425  0.002 **
+ NH3           0.17485  1 -48.308  2.0060  0.001 ***
+ NO3           0.18472  1 -48.236  1.7875  0.005 **
+ Mg            0.19276  1 -48.032  1.6369  0.011 *
+ pH            0.19916  1 -47.709  1.5035  0.028 *
<All variables> 0.20111
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

The result is exactly the same as that of the `forward.sel` function and also as the results of the `ordistep` method based on P-values and AIC (below). P-values can be adjusted for multiple testing (e.g., by Holm's correction) as follows:

```
sel.osR2_adj <- sel.osR2
sel.osR2_adj$anova$`Pr(>F)` <- p.adjust (sel.osR2$anova$`Pr(>F)`, method = 'holm',
n = ncol (chem1))
sel.osR2_adj$anova
```

```
                R2.adj Df     AIC       F Pr(>F)
+ Ca            0.12632  1 -47.149 10.9763  0.014 *
+ conduct       0.14673  1 -47.841  2.6268  0.014 *
+ Si            0.16227  1 -48.180  2.2425  0.022 *
+ NH3           0.17485  1 -48.308  2.0060  0.014 *
+ NO3           0.18472  1 -48.236  1.7875  0.050 *
+ Mg            0.19276  1 -48.032  1.6369  0.099 .
+ pH            0.19916  1 -47.709  1.5035  0.224
<All variables> 0.20111
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

You may consider increasing the number of permutations (argument `permutations` in `ordiR2step` to reach lower P-values; in the `forward.sel` example above, we increased the permutations to 49,999 to make sure that even after adjustment of P-values, the most significant one could be smaller than 0.001.

Option c) Use of ordistep function

Function `ordistep` performs step-wise selection of environmental variables based on two criteria: if their inclusion into the model leads to a significant increase in explained variance (the same as in `forward.sel` and `ordiR2step`), and if the AIC of the new model is lower than the AIC of the simpler model. In contrast to previous functions, the function does not consider as a criterion whether the adjusted R^2 of the model exceeds the adjusted R^2 of the global model. The use of `ordistep` function has the same logic as `ordiR2step`.

```
rda.vasc.0 <- rda (vasc.hell ~ 1, data = chem1) # model containing only species
matrix and intercept
rda.vasc.all <- rda (vasc.hell ~ ., data = chem1) # model including all variables
from matrix chem1 (the dot after tilda (~) means ALL!)
sel.os <- ordistep (rda.vasc.0, scope = formula (rda.vasc.all), direction =
'forward')
```

```
Start: vasc.hell ~ 1

          Df     AIC       F Pr(>F)
+ Ca       1 -47.149 10.9763  0.005 **
```

```
+ conduct  1 -45.808  9.4775  0.005 **
+ Mg       1 -45.612  9.2612  0.005 **
+ Corg     1 -43.249  6.6964  0.005 **
+ pH       1 -43.091  6.5277  0.005 **
+ Na       1 -40.775  4.1026  0.005 **
+ NH3      1 -39.964  3.2718  0.005 **
+ Si       1 -39.316  2.6145  0.005 **
+ SO4      1 -39.034  2.3314  0.005 **
+ K        1 -38.820  2.1162  0.010 **
+ NO3      1 -38.267  1.5643  0.060 .
+ Fe       1 -38.038  1.3372  0.105
+ PO4      1 -37.902  1.2030  0.150
+ Cl       1 -37.532  0.8384  0.700
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Step: vasc.hell ~ Ca

          Df     AIC      F Pr(>F)
+ conduct  1 -47.841 2.6268  0.005 **
+ NH3      1 -47.430 2.2190  0.005 **
+ pH       1 -47.318 2.1081  0.005 **
+ Na       1 -47.322 2.1125  0.010 **
+ Si       1 -47.322 2.1124  0.010 **
+ Mg       1 -47.177 1.9694  0.010 **
+ Corg     1 -47.095 1.8880  0.010 **
+ NO3      1 -46.806 1.6047  0.035 *
+ K        1 -46.502 1.3073  0.110
+ Fe       1 -46.482 1.2873  0.115
+ SO4      1 -46.209 1.0219  0.435
+ Cl       1 -46.114 0.9296  0.590
+ PO4      1 -45.968 0.7885  0.890
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Step: vasc.hell ~ Ca + conduct

...
...
...

       Df     AIC      F Pr(>F)
+ pH    1 -47.709 1.5035  0.040 *
+ SO4   1 -47.424 1.2453  0.095 .
+ Corg  1 -47.462 1.2795  0.100 .
+ Fe    1 -47.378 1.2040  0.115
+ K     1 -47.401 1.2247  0.135
+ Na    1 -47.110 0.9623  0.580
+ Cl    1 -47.044 0.9036  0.650
+ PO4   1 -46.979 0.8450  0.775
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Step: vasc.hell ~ Ca + conduct + Si + NH3 + NO3 + Mg + pH

       Df     AIC      F Pr(>F)
```

```
+ Corg  1 -47.130 1.2508  0.140
+ Fe    1 -47.054 1.1839  0.150
+ K     1 -47.048 1.1785  0.175
+ SO4   1 -47.033 1.1650  0.185
+ Na    1 -46.823 0.9783  0.595
+ PO4   1 -46.697 0.8676  0.715
+ Cl    1 -46.694 0.8645  0.735
```

Results are stored in `sel.os` object:

```
sel.os
```

```
Call: rda(formula = vasc.hell ~ Ca + conduct + Si + NH3 + NO3 + Mg + pH, data =
chem1)

              Inertia Proportion Rank
Total          0.5674     1.0000
Constrained    0.1591     0.2804    7
Unconstrained  0.4083     0.7196   62
Inertia is variance

Eigenvalues for constrained axes:
   RDA1    RDA2    RDA3    RDA4    RDA5    RDA6    RDA7
0.09590 0.01904 0.01205 0.01027 0.00871 0.00718 0.00595

Eigenvalues for unconstrained axes:
    PC1     PC2     PC3     PC4     PC5     PC6     PC7     PC8
0.03601 0.03279 0.02383 0.01908 0.01687 0.01593 0.01504 0.01358
(Showed only 8 of all 62 unconstrained eigenvalues)
```

Selected variables are Ca, `conduct`, Si, NH3, NO3, Mg and pH.

Example 2: Which soil chemical variables are important for the vegetation of tropical forests? (Barro Colorado Island)

We will use the dataset from the forest dynamics plot in tropical rainforest on Barro Colorado Island (Panama), namely data about woody species composition (dataset `BCI` in library `vegan`) and chemical variables from soil samples (`BCI.soil`). We will use forward selection to select the subset of soil variables that best explain species composition, and we will plot the resulting ordination diagram. Since the forest dynamics plot is a regular grid of 10 columns and 5 rows, we also need to choose an appropriate randomization schema when testing the explained variation.

Read the data about the composition of woody species (dataset `BCI` in `vegan`) and upload also data about soil variables (from the dataset description). Since species composition data are represented as the number of individuals of a given species in the subplot (100 m x 100 m), we log-transform them before analysis to reduce the influence of dominant species. We also calculate DCA on the species composition data to see how heterogeneous they are.

```
library (vegan)
data (BCI)
BCI.soil <- read.delim
('https://raw.githubusercontent.com/zdealveindy/anadat-r/master/data/bci.soil.txt')
BCI.spe.log <- log1p (BCI) # log-transform the species data in BCI
BCI.soil2 <- BCI.soil[,-1:-2]
```

```
decorana (BCI.spe.log)  # The length of DCA1 is 1.37 = homogeneous dataset!
```

Based on the length of the first DCA axis, we conclude that the dataset is homogeneous and can use the linear constrained ordination method (RDA). Let's define a global model (including all explanatory variables we have), and test its significance. Only if it is significant can we do forward selection.

When doing the permutation model in the function `anova`, we need to specify how to permute the samples. Since subplots are sampled in a contiguous regular grid with equal distances between neighbours in the same row or column, they are spatially dependent (closer samples are more likely to be similar to each other than distant ones), and we need to preserve this dependence when permuting the subplots. We will choose the toroidal shift permutation for the grid (see theory), with the appropriate grid size (10 columns, 5 rows; see the schema below). We will use this restricted permutation for all testing.

y-axis										
	450	5	10	15	20	25	30	35	40	45	50
	350	4	9	14	19	24	29	34	39	44	49
	250	3	8	13	18	23	28	33	38	43	48
	150	2	7	12	17	22	27	32	37	42	47
	50	1	6	11	16	21	26	31	36	41	46
		50	150	250	350	450	550	650	750	850	950
		x-axis									

```
rda_all <- rda (BCI.spe.log ~ ., BCI.soil2)
anova (rda_all, permutations = how (within = Within (type = 'grid', ncol = 10, nrow
= 5,
                                                     mirror = TRUE), complete =
TRUE))
# Permutation test for rda under reduced model
# Permutation: grid mirrored 5 rows 10 columns
# Number of permutations: 199
#
# Model: rda(formula = BCI.spe.log ~ Al + B + Ca + Cu + Fe + K + Mg + Mn + P + Zn +
N + N.min. + pH, data = BCI.soil2)
#          Df Variance      F Pr(>F)
# Model    13   24.838 2.3429  0.005 **
# Residual 36   29.358
# ---
# Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


adjR2 <- RsquareAdj (rda_all)$adj.r.squared
adjR2  # 0.2626876
```

To continue with forward selection with restricted permutation, we need to use the function `ordiR2step`, since `forward.sel` from `library (adespatial)` does not support restricted permutations. That also means that in addition to the full model `rda_all`, we also need to define the null model (with no predictor), `rda_0`:

```
rda_0 <- rda (BCI.spe.log ~ 1, data = BCI.soil2)  # includes only intercept
```

Function `ordiR2step` starts from a null model and continues with relevant variables. Important is the argument `scope`, which indicates the list of variables from which the selection is done; the scope can be taken from the full model using the function `formula`. The argument `R2scope` should include the adjusted R^2 of the general (whole) model as another stopping criterion of the selection.

```
rda_sel <- ordiR2step (rda_0, scope = formula (rda_all), direction = 'forward',
```

```
R2scope = adjR2,
                        permutations = how (within = Within (type = 'grid', ncol =
10, nrow = 5,
                                                             mirror = TRUE),
complete = TRUE))
rda_sel
# Call: rda(formula = BCI.spe.log ~ pH + P + Zn + Cu + Fe + Mn, data = BCI.soil2)
#
# Inertia Proportion Rank
# Total         54.1967     1.0000
# Constrained   18.1174     0.3343    6
# Unconstrained 36.0793     0.6657   43
# Inertia is variance
#
# Eigenvalues for constrained axes:
#   RDA1  RDA2  RDA3  RDA4  RDA5  RDA6
# 5.816 5.601 2.771 1.590 1.360 0.981
#
# Eigenvalues for unconstrained axes:
#   PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8
# 3.190 2.750 2.150 1.833 1.477 1.386 1.359 1.240
# (Showing 8 of 43 unconstrained eigenvalues)

rda_sel$anova
#                   R2.adj Df    AIC      F Pr(>F)
# + pH            0.073892  1 197.75 4.9096  0.005 **
# + P             0.132305  1 195.44 4.2314  0.005 **
# + Zn            0.176470  1 193.75 3.5206  0.005 **
# + Cu            0.210740  1 192.53 2.9973  0.005 **
# + Fe            0.229719  1 192.19 2.1087  0.005 **
# + Mn            0.241399  1 192.28 1.6775  0.010 **
# <All variables> 0.262688
# ---
# Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

The anova table above contains the column `R2.adj` with adjusted R^2 explained by that variable (for pH) or with adjusted R^2explained by given variable in addition to all previous ones included in the model (e.g. for P it is variation explained by P in addition to the variation already explained by pH).

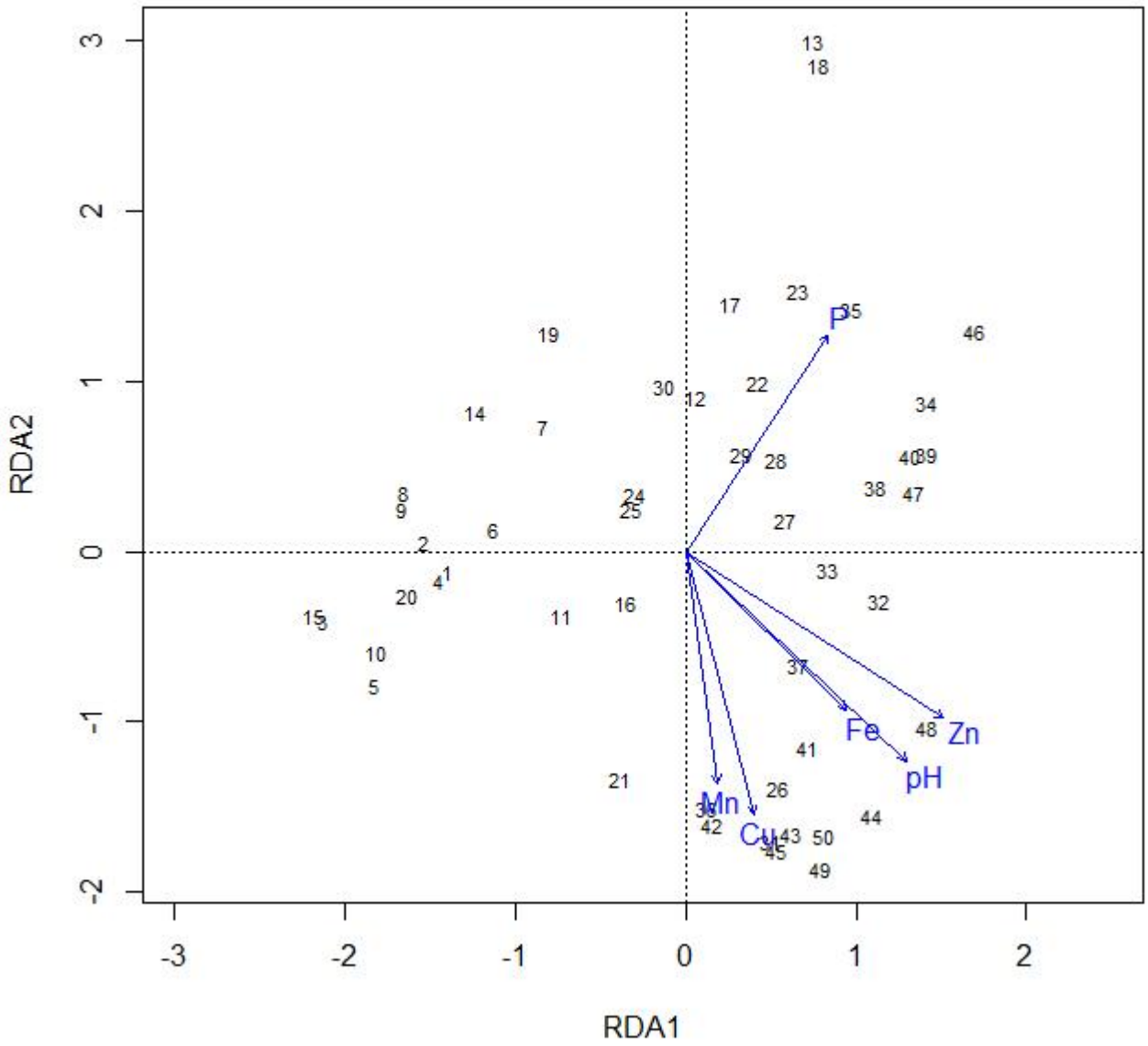
The problem with the table is an inflated Type I error rate for the P-values, due to multiple testing. We need to adjust them (here by False Discovery Rate, FDR, correction):

```
rda_sel$anova$`Pr(>F)` <- p.adjust (rda_sel$anova$`Pr(>F)`, method = 'fdr', n =
ncol (BCI.soil2))
rda_sel$anova
# R2.adj Df    AIC      F  Pr(>F)
# + pH            0.073892  1 197.75 4.9096 0.01300 *
# + P             0.132305  1 195.44 4.2314 0.01300 *
# + Zn            0.176470  1 193.75 3.5206 0.01300 *
# + Cu            0.210740  1 192.53 2.9973 0.01300 *
# + Fe            0.229719  1 192.19 2.1087 0.01300 *
# + Mn            0.241399  1 192.28 1.6775 0.02167 *
# <All variables> 0.262688
# ---
# Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

After the adjustment, all results became less significant (P < 0.05). This is the cost of the grid-based randomization schema with 10 x 5 = 50 subplots: it returns (after allowing for mirroring) only 50 x 4 = 200 permutations (including the original, non-permuted data), and the minimal P-value is then 1/200 = 0.005. After FDR adjustment for 13 permutation tests (since there are 13 variables to select from), the P-values considerably increased (0.005 after adjustment becomes 0.013). But still, all six selected variables remained significant at the level of 0.05.

Finally, we can draw the ordination diagram (using `rda_sel` ordination object created by the forward selection process):

```
ordiplot (rda_sel, display = c('sites', 'bp'), type = 't')
```



Truly interesting is that there seem to be two rather strong and almost perpendicular environmental gradients - pH (correlated also with Zn, Fe, Mn, and Cu) vs P. Also, if we check the eigenvalues of the first two ordination axes (RDA1 and RDA2 in `RDA_sel`), both of them explain a high proportion of variation (variation 5.8 and 5.6, respectively, out of total variation of 54.2). I am not familiar with the locality of BCI, but I know that there is a small depression in the western part of the plot, which sometimes gets flooded; I guess this is where the values of P are lower (samples 13 and 18 in the ordination diagram); for the distribution of pH in the plot, check dataset

description).

From:
https://anadat-r.davidzeleny.net/ - **Analysis of community ecology data in R**

Permanent link:
https://anadat-r.davidzeleny.net/doku.php/en:forward_sel_examples

Last update: **2026/04/30 09:02**