multivariate_analysis.R

```
rm(list=ls(all=TRUE))

#Pinho, Bruno; ter Braak, Cajo; P. L. Melo, Felipe;
#Bauman, David; Barlow, Jos (2024).
#Data and code from Pinho et al. - Winner-Loser plant trait replacements in
#human-modified tropical forests.
#figshare. Dataset. https://doi.org/10.6084/m9.figshare.25565169

# see README.txt for further use of this data

# Libraries----
library(data.table)

## data.table 1.14.4 using 2 threads (see ?getDTthreads).  Latest news: r-
datatable.com

library(tidyr)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(vegan)

## Loading required package: permute

## Loading required package: lattice

## This is vegan 2.6-8

library(tibble)

#remotes::install_github("CajoterBraak/douconca") # or:
#install.packages("douconca") # preferred
library(douconca)#dc-CA
```

```r
# Data----
comm <- fread("Data/comm.csv")
comm <- dcast.data.table(data = comm, Region_Plot ~ Binomial_correct)

## Using 'BA' as value column. Use 'value.var' to override

## Aggregate function missing, defaulting to 'length'

comm <- column_to_rownames(comm, "Region_Plot")

env  <- fread("Data/env.csv")
env$invBA_ha <- 1/ env$BA_ha

traits <- fread("Data/traits.csv")
traits <- column_to_rownames(traits, "Binomial_correct")

#Create community matrix

# Analysis----

# an initial analysis required
# all traits are being used in this selection as formulaTraits =~.
names(traits)

## [1] "WD"   "lnSM" "LMA"  "Hmax" "DS"

out1 <- dc_CA(formulaEnv = ~lnFL2000 + Condition(Region), formulaTraits =~.,
                    response = comm, dataEnv = env, dataTraits = traits)

## Warning in dc_CA(formulaEnv = ~lnFL2000 + Condition(Region), formulaTraits
= ~., : variableCWD has missing values and is deleted from the environmental
data.

## Step 1: the CCA ordination of the transposed matrix with trait
constraints,
##         useful in itself and also yielding CWMs of the orthonormalized
traits for step 2.
## Call: cca(formula = tY ~ WD + lnSM + LMA + Hmax +
## DS, data = dataTraits)
##
## -- Model Summary --
##
##                Inertia Proportion Rank
## Total         28.23366    1.00000
## Constrained    0.47361    0.01677    6
## Unconstrained 27.76005    0.98323  270
##
## Inertia is scaled Chi-square
##
## -- Eigenvalues --
```

```
##
## Eigenvalues for constrained axes:
##    CCA1    CCA2    CCA3    CCA4    CCA5    CCA6
## 0.20958 0.07642 0.06127 0.04692 0.04579 0.03364
##
## Eigenvalues for unconstrained axes:
##    CA1    CA2    CA3    CA4    CA5    CA6    CA7
## 0.8026 0.6536 0.6198 0.6021 0.5610 0.5342 0.4894
##    CA8
## 0.4346
## (Showing 8 of 270 unconstrained eigenvalues)
##
## Step 2: the RDA ordination of CWMs of the orthonormalized traits
##         of step 1 with environmental constraints:
## Call: rda(formula = out1$CWMs_orthonormal_traits ~
## lnFL2000 + Condition(Region), data =
## out1$data$dataEnv)
##
## -- Model Summary --
##
##               Inertia Proportion Rank
## Total         0.47361    1.00000
## Conditional   0.06598    0.13930    5
## Constrained   0.04775    0.10082    1
## Unconstrained 0.35989    0.75988    6
##
## Inertia is variance
##
## -- Eigenvalues --
##
## Eigenvalues for constrained axes:
##    RDA1
## 0.04775
##
## Eigenvalues for unconstrained axes:
##     PC1     PC2     PC3     PC4     PC5     PC6
## 0.15012 0.05697 0.05161 0.03921 0.03550 0.02648
##
## mean, sd, VIF and canonical coefficients with their optimistic [!] t-
values:
##               Avg     SDS    VIF   Regr1   tval1
## RegionPB   0.0627  3.9917 1.3706  0.0114  0.4110
## RegionPGM  0.3469  7.8355 2.5118 -0.0976 -2.6019
## RegionSGD  0.1365  5.6523 1.7273  0.0040  0.1282
## RegionSTM  0.2325  6.9537 2.3089 -0.1058 -2.9433
## RegionUna  0.0849  4.5878 1.4942 -0.0221 -0.7649
## lnFL2000  -1.7400 25.7541 1.2629 -0.2456 -9.2355
##              Avg     SDS    VIF   Regr1   tval1
## WD        0.6374  0.1729 1.4905  0.5438  9.5432
## lnSM      5.6438  2.4597 1.4762  0.5540  9.7676
```

```
## LMA      82.9936 24.1529 1.0145 -0.0197 -0.4194
## Hmax     28.3753  8.7913 1.0652 -0.0711 -1.4764
## DSNonZoo  0.1693  0.3750 1.0911  0.2793  5.7275
## DSSynZoo  0.1216  0.3269 1.2521  0.0814  1.5587
##
##                weighted variance
## total                      28.234
## traits_explain              0.474
## conditionE                  0.066
## constraintsTE               0.048
## attr(,"meaning")
##                meaning
## total          "total inertia"
## traits_explain "trait-constrained inertia"
## conditionE     "trait-constrained inertia explained by the condition in
formulaEnv"
## constraintsTE  "trait-constrained inertia explained by the predictors in
formulaEnv"

out1$eigenvalues

##      dcCA1
## 0.04774874

## Fitting models according to the DAG
## (as in Fig.3), but now using dc-CA

### step 1 Forest loss-------------------------------------------------------
-----------

considered <- NULL
consider <- c("lnFL500","lnFL1000", "lnFL2000")
names(consider) <- consider
consider

##    lnFL500   lnFL1000   lnFL2000
##  "lnFL500" "lnFL1000" "lnFL2000"

fit_measures <- matrix(NA, nrow = length(consider), ncol = 2)
colnames(fit_measures) <- c("variance","pval1")
rownames(fit_measures) <- consider

cntr <- how(within = Within(type = "free"), plots = Plots(strata=
env$Region, type = "none"), nper= 1999)
set.seed(123)

test <- TRUE

for (k in seq_along(consider)){

  formulaE_FS <- as.formula(paste("~", consider[k], "+Condition(Region)" ))
```

```
  out_FS <- dc_CA(formulaE_FS,
                  dc_CA_object = out1, verbose = FALSE)

  if (test) {
    an <- anova(out_FS$RDAonEnv, permutations = cntr)
    pval <- an$`Pr(>F)`[1]} else pval <- NA
  fit_measures[k,] <- c(out_FS$inertia["constraintsTE","weighted variance"],
pval)
}


pvaladj3 <- c(p.adjust(fit_measures[1:3,"pval1"], method = "holm"), rep(NA,
nrow(fit_measures)-3))
#pvaladj <- p.adjust(fit_measures[,"pval1"], method = "holm")
fit_measures <- cbind(fit_measures,pvaladj3)
round(fit_measures, 5)

##          variance pval1 pvaladj3
## lnFL500   0.03328 5e-04   0.0015
## lnFL1000  0.04092 5e-04   0.0015
## lnFL2000  0.04775 5e-04   0.0015

# best = lnFL2000 ; all significant with and without correction for multiple
testing

fit_measuresL <- list()
fit_measuresL[[1]] <- fit_measures

considered <- c(considered, consider["lnFL2000"])
considered

##   lnFL2000
## "lnFL2000"

### step 2 Fragmentation-----------------------------------------------------
-------------
stepk <-2

considerk <-  c("lnNfrag500","lnNfrag1000", "lnNfrag2000")

fit_measures <- matrix(NA, nrow = length(considerk), ncol = 2)
colnames(fit_measures) <- c("variance","pval1")
rownames(fit_measures) <- considerk

for (k in seq_along(considerk)){

  formulaE_FS <- as.formula(paste("~", considerk[k], "+Condition(Region +",
paste(considered, collapse = "+") ,")", sep ="" ))
```

```
  out_FS <- dc_CA(formulaE_FS,
                  dc_CA_object = out1, verbose = FALSE)

  if (test) {
    an <- anova(out_FS$RDAonEnv, permutations = cntr)
    pval <- an$`Pr(>F)`[1]} else pval <- NA
  fit_measures[k,] <- c(out_FS$inertia["constraintsTE","weighted variance"],
pval)
}

pvaladj3 <- c(p.adjust(fit_measures[1:3,"pval1"], method = "holm"), rep(NA,
nrow(fit_measures)-3))
#pvaladj <- p.adjust(fit_measures[,"pval1"], method = "holm")
fit_measures <- cbind(fit_measures,pvaladj3 )
round(fit_measures, 5)

##             variance  pval1 pvaladj3
## lnNfrag500   0.00274 0.0840   0.1680
## lnNfrag1000  0.00341 0.0415   0.1245
## lnNfrag2000  0.00246 0.1345   0.1680

# best = lnNfrag1000 ; not significant, neither with nor without multiple
testing correction

fit_measuresL[[stepk]] <- fit_measures

considered <- c(considered, "lnNfrag1000")
considered

##      lnFL2000
##    "lnFL2000" "lnNfrag1000"

### step 3 Edge density-------------------------------------------------------
------------

stepk <-3
considerk <-  c("ln_edg500" , "ln_edg1000" ,"ln_edg2000")

fit_measures <- matrix(NA, nrow = length(considerk), ncol = 2)
colnames(fit_measures) <- c("variance","pval1")
rownames(fit_measures) <- considerk

for (k in seq_along(considerk)){

  formulaE_FS <- as.formula(paste("~", considerk[k], "+Condition(Region +",
paste(considered, collapse = "+") ,")", sep ="" ))

  out_FS <- dc_CA(formulaE_FS,
                  dc_CA_object = out1, verbose = FALSE)
```

```
  if (test) {
    an <- anova(out_FS$RDAonEnv, permutations = cntr)
    pval <- an$`Pr(>F)`[1]} else pval <- NA
  fit_measures[k,] <- c(out_FS$inertia["constraintsTE","weighted variance"],
pval)
}

pvaladj3 <- c(p.adjust(fit_measures[1:3,"pval1"], method = "holm"), rep(NA,
nrow(fit_measures)-3))
fit_measures <- cbind(fit_measures,pvaladj3)
round(fit_measures, 5) #

##            variance  pval1 pvaladj3
## ln_edg500   0.00323 0.0420   0.1260
## ln_edg1000  0.00165 0.2665   0.2665
## ln_edg2000  0.00281 0.0685   0.1370

# best ln_edg500

fit_measuresL[[stepk]] <- fit_measures #
fit_measuresL

## [[1]]
##            variance pval1 pvaladj3
## lnFL500  0.03328161 5e-04   0.0015
## lnFL1000 0.04092097 5e-04   0.0015
## lnFL2000 0.04774874 5e-04   0.0015
##
## [[2]]
##                variance  pval1 pvaladj3
## lnNfrag500  0.002742606 0.0840   0.1680
## lnNfrag1000 0.003408878 0.0415   0.1245
## lnNfrag2000 0.002458787 0.1345   0.1680
##
## [[3]]
##               variance  pval1 pvaladj3
## ln_edg500  0.003231004 0.0420   0.1260
## ln_edg1000 0.001647057 0.2665   0.2665
## ln_edg2000 0.002810637 0.0685   0.1370

considered <- c(considered, "ln_edg500")
considered

##      lnFL2000
##    "lnFL2000" "lnNfrag1000"   "ln_edg500"

### step 4 Local degradation-----------------------------------------------
-----------------

stepk <-4
considerk <-  c("invBA_ha")
```

```r
fit_measures <- matrix(NA, nrow = length(considerk), ncol = 2)
colnames(fit_measures) <- c("variance","pval1")
rownames(fit_measures) <- considerk

for (k in seq_along(considerk)){

  formulaE_FS <- as.formula(paste("~", considerk[k], "+Condition(Region +",
paste(considered, collapse = "+") ,")", sep ="" ))

  out_FS <- dc_CA(formulaE_FS,
                  dc_CA_object = out1, verbose = FALSE)

  if (test) {
    an <- anova(out_FS$RDAonEnv, permutations = cntr)
    pval <- an$`Pr(>F)`[1]} else pval <- NA
  fit_measures[k,] <- c(out_FS$inertia["constraintsTE","weighted variance"],
pval)
}

round(fit_measures, 5)
```

```
##          variance pval1
## invBA_ha  0.00991 5e-04
```

```r
fit_measuresL[[stepk]] <- fit_measures
fit_measuresL
```

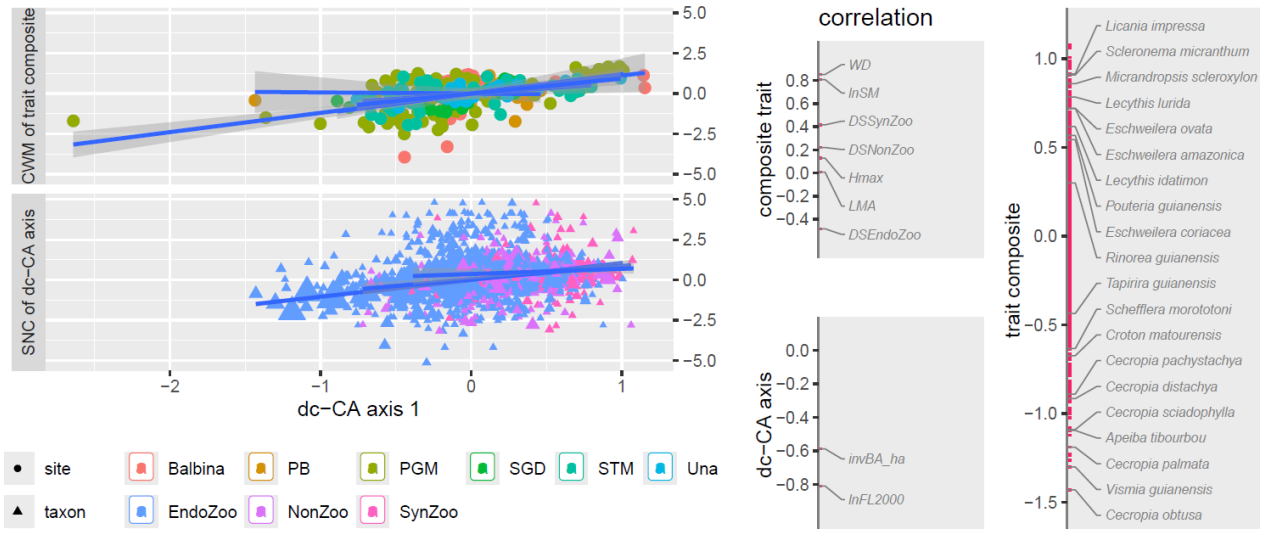
```
## [[1]]
##            variance pval1 pvaladj3
## lnFL500  0.03328161 5e-04   0.0015
## lnFL1000 0.04092097 5e-04   0.0015
## lnFL2000 0.04774874 5e-04   0.0015
##
## [[2]]
##                variance  pval1 pvaladj3
## lnNfrag500  0.002742606 0.0840   0.1680
## lnNfrag1000 0.003408878 0.0415   0.1245
## lnNfrag2000 0.002458787 0.1345   0.1680
##
## [[3]]
##               variance  pval1 pvaladj3
## ln_edg500  0.003231004 0.0420   0.1260
## ln_edg1000 0.001647057 0.2665   0.2665
## ln_edg2000 0.002810637 0.0685   0.1370
##
## [[4]]
##             variance pval1
## invBA_ha 0.009907882 5e-04
```

```r
# Final model with lnFL2000 and local degradation ------------------------


outF <- dc_CA(formulaEnv = ~lnFL2000 + invBA_ha + Condition(Region),
formulaTraits =~.,
              response = comm, dataEnv = env, dataTraits = traits)

## Warning in dc_CA(formulaEnv = ~lnFL2000 + invBA_ha + Condition(Region), :
variableCWD has missing values and is deleted from the environmental data.

## Step 1: the CCA ordination of the transposed matrix with trait
constraints,
##         useful in itself and also yielding CWMs of the orthonormalized
traits for step 2.
## Call: cca(formula = tY ~ WD + lnSM + LMA + Hmax +
## DS, data = dataTraits)
##
## -- Model Summary --
##
##                Inertia Proportion Rank
## Total         28.23366    1.00000
## Constrained    0.47361    0.01677    6
## Unconstrained 27.76005    0.98323  270
##
## Inertia is scaled Chi-square
##
## -- Eigenvalues --
##
## Eigenvalues for constrained axes:
##    CCA1    CCA2    CCA3    CCA4    CCA5    CCA6
## 0.20958 0.07642 0.06127 0.04692 0.04579 0.03364
##
## Eigenvalues for unconstrained axes:
##    CA1    CA2    CA3    CA4    CA5    CA6    CA7
## 0.8026 0.6536 0.6198 0.6021 0.5610 0.5342 0.4894
##    CA8
## 0.4346
## (Showing 8 of 270 unconstrained eigenvalues)
##
## Step 2: the RDA ordination of CWMs of the orthonormalized traits
##         of step 1 with environmental constraints:
## Call: rda(formula = out1$CWMs_orthonormal_traits ~
## lnFL2000 + invBA_ha + Condition(Region), data =
## out1$data$dataEnv)
##
## -- Model Summary --
##
##               Inertia Proportion Rank
## Total         0.47361    1.00000
## Conditional   0.06598    0.13930    5
```

```
## Constrained   0.05861    0.12376    2
## Unconstrained 0.34902    0.73694    6
##
## Inertia is variance
##
## -- Eigenvalues --
##
## Eigenvalues for constrained axes:
##    RDA1    RDA2
## 0.05733 0.00128
##
## Eigenvalues for unconstrained axes:
##     PC1     PC2     PC3     PC4     PC5     PC6
## 0.14106 0.05679 0.05154 0.03805 0.03549 0.02608
##
## mean, sd, VIF and canonical coefficients with their optimistic [!] t-
values:
##               Avg     SDS    VIF   Regr1   tval1
## RegionPB   0.0627  3.9917 1.3713  0.0140  0.5221
## RegionPGM  0.3469  7.8355 2.8648 -0.0385 -0.9949
## RegionSGD  0.1365  5.6523 1.7282  0.0009  0.0304
## RegionSTM  0.2325  6.9537 2.4016 -0.0754 -2.1278
## RegionUna  0.0849  4.5878 1.4970 -0.0273 -0.9744
## lnFL2000  -1.7400 25.7541 1.3627 -0.2137 -8.0033
## invBA_ha   0.0478  0.5157 1.2939 -0.1126 -4.3281
##                Avg     SDS    VIF   Regr1   tval1
## WD          0.6374  0.1729 1.4905  0.5245  9.4324
## lnSM        5.6438  2.4597 1.4762  0.5579 10.0812
## LMA        82.9936 24.1529 1.0145 -0.0141 -0.3073
## Hmax       28.3753  8.7913 1.0652 -0.0088 -0.1868
## DSNonZoo    0.1693  0.3750 1.0911  0.2883  6.0592
## DSSynZoo    0.1216  0.3269 1.2521  0.0949  1.8615
##
##                 weighted variance
## total                      28.234
## traits_explain              0.474
## conditionE                  0.066
## constraintsTE               0.059
## attr(,"meaning")
##                 meaning
## total           "total inertia"
## traits_explain "trait-constrained inertia"
## conditionE     "trait-constrained inertia explained by the condition in
formulaEnv"
## constraintsTE  "trait-constrained inertia explained by the predictors in
formulaEnv"
```

```r
anova(outF, by= "axis", permutations = list(999,cntr))
```

```
## $species
## Species-level permutation test using dc-CA
## Model: dc_CA(formulaEnv = ~lnFL2000 + invBA_ha + Condition(Region),
formulaTraits = ~., response = comm, dataEnv = env, dataTraits = traits)
## Residualized predictor permutation
##
##           df ChiSquare       R2        F Pr(>F)
## dcCA1      1  0.057333 0.186197 275.9672  0.001 ***
## dcCA2      1  0.001279 0.004154   6.1569  0.902
## Residual 1200  0.249303
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## $sites
##           Df ChiSquare       R2       F Pr(>F)
## dcCA1      1   0.05733 0.140647 43.2020 0.0005 ***
## dcCA2      1   0.00128 0.003138  0.9638 0.4590
## Residual 263   0.34902
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## $max
## Max test combining the community- and species- level tests
## Model: dc_CA(formulaEnv = ~lnFL2000 + invBA_ha + Condition(Region),
formulaTraits = ~., response = comm, dataEnv = env, dataTraits = traits)
##
## Taken from the species-level test:
## Residualized predictor permutation
## Permutation: free
## Number of permutations: 999
##
##           df ChiSquare       R2        F Pr(>F)
## dcCA1      1  0.057333 0.186197 275.9672  0.001 ***
## dcCA2      1  0.001279 0.004154   6.1569  0.902
## Residual 1200  0.249303
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

plot(outF, widths = c(3,1,1.2), remove_centroids = TRUE)
```




```
# Extra code showing log(invBA_ha) ---------------------------------------

# avoids an outlier

outF2 <- dc_CA(formulaEnv = ~lnFL2000 + log(invBA_ha) + Condition(Region),
formulaTraits =~.,
               response = comm, dataEnv = env, dataTraits = traits)

## Warning in dc_CA(formulaEnv = ~lnFL2000 + log(invBA_ha) +
Condition(Region), : variableCWD has missing values and is deleted from the
environmental data.

## Step 1: the CCA ordination of the transposed matrix with trait
constraints,
##         useful in itself and also yielding CWMs of the orthonormalized
traits for step 2.
## Call: cca(formula = tY ~ WD + lnSM + LMA + Hmax +
## DS, data = dataTraits)
##
## -- Model Summary --
##
##                Inertia Proportion Rank
## Total         28.23366    1.00000
## Constrained    0.47361    0.01677    6
## Unconstrained 27.76005    0.98323  270
##
## Inertia is scaled Chi-square
##
## -- Eigenvalues --
##
## Eigenvalues for constrained axes:
##    CCA1    CCA2    CCA3    CCA4    CCA5    CCA6
## 0.20958 0.07642 0.06127 0.04692 0.04579 0.03364
##
```

```
## Eigenvalues for unconstrained axes:
##    CA1    CA2    CA3    CA4    CA5    CA6    CA7
## 0.8026 0.6536 0.6198 0.6021 0.5610 0.5342 0.4894
##    CA8
## 0.4346
## (Showing 8 of 270 unconstrained eigenvalues)
##
## Step 2: the RDA ordination of CWMs of the orthonormalized traits
##         of step 1 with environmental constraints:
## Call: rda(formula = out1$CWMs_orthonormal_traits ~
## lnFL2000 + log(invBA_ha) + Condition(Region), data
## = out1$data$dataEnv)
##
## -- Model Summary --
##
##               Inertia Proportion Rank
## Total         0.47361    1.00000
## Conditional   0.06598    0.13930    5
## Constrained   0.05929    0.12519    2
## Unconstrained 0.34834    0.73551    6
##
## Inertia is variance
##
## -- Eigenvalues --
##
## Eigenvalues for constrained axes:
##    RDA1    RDA2
## 0.05755 0.00174
##
## Eigenvalues for unconstrained axes:
##     PC1     PC2     PC3     PC4     PC5     PC6
## 0.14089 0.05670 0.05161 0.03788 0.03548 0.02579
##
## mean, sd, VIF and canonical coefficients with their optimistic [!] t-
values:
##                   Avg     SDS    VIF   Regr1   tval1
## RegionPB       0.0627  3.9917 1.3830  0.0002  0.0065
## RegionPGM      0.3469  7.8355 3.1489 -0.0170 -0.4194
## RegionSGD      0.1365  5.6523 1.7552 -0.0128 -0.4243
## RegionSTM      0.2325  6.9537 2.5439 -0.0567 -1.5570
## RegionUna      0.0849  4.5878 1.5408 -0.0437 -1.5422
## lnFL2000      -1.7400 25.7541 1.4340 -0.2032 -7.4266
## log(invBA_ha) -3.1720  8.1671 1.7647 -0.1336 -4.4008
##              Avg     SDS    VIF   Regr1   tval1
## WD        0.6374  0.1729 1.4905  0.5203  8.7554
## lnSM      5.6438  2.4597 1.4762  0.5580  9.4358
## LMA      82.9936 24.1529 1.0145 -0.0291 -0.5943
## Hmax     28.3753  8.7913 1.0652  0.0012  0.0245
## DSNonZoo  0.1693  0.3750 1.0911  0.2840  5.5857
## DSSynZoo  0.1216  0.3269 1.2521  0.1021  1.8755
```

```
##
##                weighted variance
## total                      28.234
## traits_explain              0.474
## conditionE                  0.066
## constraintsTE               0.059
## attr(,"meaning")
##                meaning
## total          "total inertia"
## traits_explain "trait-constrained inertia"
## conditionE     "trait-constrained inertia explained by the condition in
formulaEnv"
## constraintsTE  "trait-constrained inertia explained by the predictors in
formulaEnv"
```

```
anova(outF2, by= "axis", permutations = list(999,cntr))
```

```
## $species
## Species-level permutation test using dc-CA
## Model: dc_CA(formulaEnv = ~lnFL2000 + log(invBA_ha) + Condition(Region),
formulaTraits = ~., response = comm, dataEnv = env, dataTraits = traits)
## Residualized predictor permutation
##
##            df ChiSquare       R2        F Pr(>F)
## dcCA1       1  0.057552 0.158522 227.3577  0.001 ***
## dcCA2       1  0.001741 0.004795   6.8765  0.821
## Residual 1200  0.303759
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## $sites
##           Df ChiSquare      R2       F Pr(>F)
## dcCA1      1   0.05755 0.14118 43.4516 0.0005 ***
## dcCA2      1   0.00174 0.00427  1.3142 0.2890
## Residual 263   0.34834
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## $max
## Max test combining the community- and species- level tests
## Model: dc_CA(formulaEnv = ~lnFL2000 + log(invBA_ha) + Condition(Region),
formulaTraits = ~., response = comm, dataEnv = env, dataTraits = traits)
##
## Taken from the species-level test:
## Residualized predictor permutation
## Permutation: free
## Number of permutations: 999
##
```

```
##            df ChiSquare       R2        F Pr(>F)
## dcCA1       1  0.057552 0.158522 227.3577  0.001 ***
## dcCA2       1  0.001741 0.004795   6.8765  0.821
## Residual 1200  0.303759
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
plot(outF2, widths = c(3,1,1.2), remove_centroids = TRUE, flip_axis = TRUE)
```