Table of Contents

Definition of the function envfit.iv

Reference: Zelený & Schafers (2012)

envfit.iv

Appendix S2 of the paper Zelený & Schaffers (2012) – Function for projection of mean Ellenberg indicator values onto an ordination, with modified permutation test.

Note: this function has been included in the package `weimea` as `envfit_cwm`, with improved functionality. Consider installing `weimea` and trying it!

Description

The function fits weighted mean of species attributes (e.g. Ellenberg indicator values), calculated for samples, onto an ordination. It employs permutation test, based on a null model randomizing species attributes prior to calculation of their sample mean weighted by their abundance. This test is invented to correct the bias in the relationship between mean species attributes (e.g. mean Ellenberg indicator values) and sample scores along ordination axes introduced by the fact that weighted mean of species attributes inherits compositional similarity among samples.

Usage

```
## Default S3 method: envfit.iv (ord, veg, spec.iv, permutations = 999,
choices=c(1,2), display = “sites”, w = weights(ord), na.rm = FALSE, ...)
```

Arguments

`ord`	An ordination object or other structure from which the ordination scores can be extracted (including a data frame or matrix of scores).
`veg`	Matrix-like object of community data (samples in rows, species in columns). It should be transformed into presence-absence data (e.g. using function decostand (veg, method = ‘pa’) from vegan package) if the species values are not intended to be weighted by species abundances.
`spec.iv`	Data frame, matrix or vector of species indicator values (species in rows, variables in columns). The order of the species in rows MUST COMPLY with the order of the species in columns of community matrix veg (no warning is returned if this is not true, however, the result will be wrong in such case).
`permutations`	Number of permutations for assessing the significance of the fit. If set to 0, the significance test is not calculated.
`choices`	Ordination axes, for which the fit is calculated.

display	Ordinary site scores ('sites') or linear combination scores ('lc') in constrained analysis.
w	Weights used in fitting (concerns mainly cca and decorana results which have nonconstant weights).
na.rm	Remove points with missing values in ordination scores or environmental variables.
...	Parameters passed to scores.

Details

The function envfit.iv is derived from the function envfit (library vegan), which fits the environmental vectors or factors onto an ordination and tests their significance. The modification is in the permutation schema used to test the fit significance: while the original envfit function builds the null distribution by permuting the values of environmental vector, modified envfit.iv function permutes the assignment of species to species indicator values, from which the weighted average for sample is calculated. This modifies the null hypothesis to be tested, from the original 'no statistical relationship between mean indicator values and sample ordination scores' hypothesis to a new 'no statistical relationship between information from external ecological data and sample ordination scores' hypothesis.

The permutation model requires information about the original vegetation matrix used to calculate the weighted mean of species indicator values, and also original species indicator values.

The generic functions plot and scores for the class envfit can be used to plot the vectors of mean indicator values onto an ordination diagram and to assess the fitting results.

Value

Identical to those of envfit.

Author(s)

David Zelený (zeleny@ntu.edu.tw); the script is almost entirely based on the original functions envfit and vectorfit from vegan package, written by Jari Oksanen. Update to the function envfit.iv for latest version of vegan was provided by Sebastian Utermann.

See Also

envfit, vectorfit and plot.envfit from library vegan

Examples

```
# Import example data first (suppose that you set up the working directory
```

```
to the folder where you have the example datasets vltava-spe.csv and vltava-
eiv.csv, e.g. using the function setwd):
vltava.spe <- read.csv ('vltava-spe.csv', row.names = 1) #species data
(presence-absence)
vltava.species.eiv <- read.csv ('vltava-spec.eiv.csv', row.names = 1) #
species Ellenberg indicator values

# Calculate mean EIVs:
vltava.mean.eiv <- apply (vltava.species.eiv, 2, FUN = function (x)
colSums(t(vltava.spe)*x, na.rm = T)/rowSums (vltava.spe[,!is.na(x)], na.rm =
T))

# Load required vegan library:
library (vegan)

# Calculate Detrended correspondance analysis (species data are presence-
absence, so there is no need to transform):
dca <- decorana (vltava.spe)

# Calculate the fit of mean Ellenberg indicator values with significances
based on the original permutation model:
fit.orig <- envfit (dca, vltava.mean.eiv, permutations = 999)

fit.orig
# ***VECTORS
#
#            DCA1      DCA2     r2 Pr(>r)
# Light  0.477462  0.878653 0.6004  0.001 ***
# Temp   0.349901  0.936787 0.4709  0.001 ***
# Cont   0.725976  0.687720 0.1479  0.004 **
# Moist -0.924542  0.381079 0.8971  0.001 ***
# Nutr  -0.997826  0.065911 0.8314  0.001 ***
# React -0.652896  0.757448 0.4287  0.001 ***
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# P values based on 999 permutations.


# Calculate the fit of mean Ellenberg indicator values with modified
permutation model (this expects that you defined the envfit.iv function):
fit.modif <- envfit.iv (ord = dca, veg = vltava.spe, spec.iv =
vltava.species.eiv, permutations = 999)

fit.modif
# ***VECTORS
#
#            DCA1      DCA2     r2 Pr(>r)
# Light  0.477462  0.878653 0.6004  0.003 **
# Temp   0.349901  0.936787 0.4709  0.017 *
# Cont   0.725976  0.687720 0.1479  0.432
# Moist -0.924542  0.381079 0.8971  0.001 ***
```

```
# Nutr  -0.997826  0.065911 0.8314  0.001 ***
# React -0.652896  0.757448 0.4287  0.030 *
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
# P values based on 999 permutations.

# Plot DCA ordination diagram with passively projected Ellenberg indicator
values:
plot (dca, display = 'sites', type = 'p')
plot (fit.orig, p.max = .05)
plot (fit.modif, p.max = 0.05, col = 'red')
```

Definition of the function (version updated on 6/2019 by Sebastian Utermann)

```
envfit.iv <- function (ord, veg, spec.iv, permutations = 999, choices = c(1,
2), display = "sites", w = weights(ord), na.rm = FALSE, ...)
{
  weights.default <- function(object, ...) NULL
  vectorfit.iv <-
    function (X, veg, spec.iv, permutations, w, ...)
  {
    apply.FUN <- function (x)
    {
      veg.temp <- veg [,!is.na (x)]
      x.temp <- x[!is.na (x)]
      colSums (t(veg.temp)*x.temp)/rowSums (veg.temp)
    }
    apply.FUN.sample <- function (x)
    {
      veg.temp <- veg [,!is.na (x)]
      x.temp <- x[!is.na (x)]
      colSums (t(veg.temp)*sample (x.temp))/rowSums (veg.temp)
    }
    P <- apply (spec.iv, 2, FUN = apply.FUN)
    X <- as.matrix(X)
    if (missing(w) || is.null(w))
      w <- 1
    if (length(w) == 1)
      w <- rep(w, nrow(X))
    Xw <- .Call("do_wcentre", X, w)
    dim(Xw) <- dim(X)
    Pw <- .Call("do_wcentre", P, w)
    dim(Pw) <- dim(P)
    colnames(Pw) <- colnames(P)
    nc <- ncol(X)
    Q <- qr(Xw)
    H <- qr.fitted(Q, Pw)
```

```
    heads <- qr.coef(Q, Pw)
    r <- diag(cor(H, Pw)^2)
    heads <- decostand(heads, "norm", 2)
    heads <- t(heads)
    if (is.null(colnames(X)))
      colnames(heads) <- paste("Dim", 1:nc, sep = "")
    else colnames(heads) <- colnames(X)
    if (permutations) {
      nr <- nrow(X)
      permstore <- matrix(nrow = permutations, ncol = ncol(P))
      for (i in 1:permutations) {
        take <- apply (spec.iv, 2, FUN = apply.FUN.sample)
        take <- .Call("do_wcentre", take, w)
        dim(take) <- dim(P)
        Hperm <- qr.fitted(Q, take)
        permstore[i, ] <- diag(cor(Hperm, take))^2
      }
      permstore <- sweep(permstore, 2, r, ">")
      pvals <- (apply(permstore, 2, sum) + 1)/(permutations +
                                               1)
    }
    else pvals <- NULL
    sol <- list(arrows = heads, r = r, permutations = permutations,
                pvals = pvals)
    class(sol) <- "vectorfit"
    sol
    }

    w <- eval(w)
    vectors <- NULL
    factors <- NULL
    seed <- NULL
    X <- scores(ord, display = display, choices = choices, ...)
    keep <- complete.cases(X)
    if (any(!keep)) {
        if (!na.rm)
            stop("missing values in data: consider na.rm = TRUE")
        X <- X[keep, , drop = FALSE]
        na.action <- structure(seq_along(keep)[!keep], class = "omit")
    }
    vectors <- vectorfit.iv(X, veg, spec.iv, permutations, choices,
        w = w, ...)
    sol <- list(vectors = vectors, factors = factors)
    if (!is.null(na.action))
        sol$na.action <- na.action
    class(sol) <- "envfit"
    sol
}
```

Definition of the function (original version published in Zelený & Scheffers 2012)

```
envfit.iv <- function (ord, veg, spec.iv, permutations = 999, choices = c(1,
2), display = "sites", w = weights(ord), na.rm = FALSE, ...)
{
    weights.default <- function(object, ...) NULL
    vectorfit.iv <- function (X, veg, spec.iv, permutations, w, ...)
    {
    apply.FUN <- function (x)
    {
    veg.temp <- veg [,!is.na (x)]
    x.temp <- x[!is.na (x)]
    colSums (t(veg.temp)*x.temp)/rowSums (veg.temp)
    }

    apply.FUN.sample <- function (x)
    {
    veg.temp <- veg [,!is.na (x)]
    x.temp <- x[!is.na (x)]
    colSums (t(veg.temp)*sample (x.temp))/rowSums (veg.temp)
    }
    P <- apply (spec.iv, 2, FUN = apply.FUN)
    X <- as.matrix(X)
    if (missing(w) || is.null(w))
        w <- 1
    if (length(w) == 1)
        w <- rep(w, nrow(X))
    Xw <- .C("wcentre", x = as.double(X), as.double(w), as.integer(nrow(X)),
        as.integer(ncol(X)), PACKAGE = "vegan")$x
    dim(Xw) <- dim(X)
    Pw <- .C("wcentre", x = as.double(P), as.double(w), as.integer(nrow(P)),
        as.integer(ncol(P)), PACKAGE = "vegan")$x
    dim(Pw) <- dim(P)
    colnames(Pw) <- colnames(P)
    nc <- ncol(X)
    Q <- qr(Xw)
    H <- qr.fitted(Q, Pw)
    heads <- qr.coef(Q, Pw)
    r <- diag(cor(H, Pw)^2)
    heads <- decostand(heads, "norm", 2)
    heads <- t(heads)
    if (is.null(colnames(X)))
        colnames(heads) <- paste("Dim", 1:nc, sep = "")
    else colnames(heads) <- colnames(X)
    if (permutations) {
        nr <- nrow(X)
        permstore <- matrix(nrow = permutations, ncol = ncol(P))
        for (i in 1:permutations) {
            take <- apply (spec.iv, 2, FUN = apply.FUN.sample)
            take <- .C("wcentre", x = as.double(take), as.double(w),
                as.integer(nrow(take)), as.integer(ncol(take)),
                PACKAGE = "vegan")$x
```

```
            dim(take) <- dim(P)
            Hperm <- qr.fitted(Q, take)
            permstore[i, ] <- diag(cor(Hperm, take))^2
        }
        permstore <- sweep(permstore, 2, r, ">")
        pvals <- (apply(permstore, 2, sum) + 1)/(permutations +
            1)
    }
    else pvals <- NULL
    sol <- list(arrows = heads, r = r, permutations = permutations,
        pvals = pvals)
    class(sol) <- "vectorfit"
    sol
    }
    w <- eval(w)
    vectors <- NULL
    factors <- NULL
    seed <- NULL
    X <- scores(ord, display = display, choices = choices, ...)
    keep <- complete.cases(X)
    if (any(!keep)) {
        if (!na.rm)
            stop("missing values in data: consider na.rm = TRUE")
        X <- X[keep, , drop = FALSE]
        na.action <- structure(seq_along(keep)[!keep], class = "omit")
    }
    vectors <- vectorfit.iv(X, veg, spec.iv, permutations, choices,
        w = w, ...)
    sol <- list(vectors = vectors, factors = factors)
    if (!is.null(na.action))
        sol$na.action <- na.action
    class(sol) <- "envfit"
    sol
}
```