Table of Contents

Unconstrained ordination

Ordination diagrams

Important component of unconstrained ordination analysis is visualization of results (this doesn't apply so much in case of constrained ordination, when we are more interested in explained variance and results of significance test). R is notoriously not good in drawing ordination diagrams with complex information, because these usually needs manual adjustment, which is not easy in R. Generally, if you need ordination diagram with interpretation focused on individual species or samples (with appropriate labels), R may not produce satisfying results - you may consider using CANOCO 4.5 with CanoDraw for Windows or CANOCO 5[1)]. However, R can still draw nice ordination diagrams, if the focus is on overall pattern of samples, grouping of samples into clusters or projecting linear or surface response of environment onto ordination space (even three dimensional) - few examples are below.

R functions

From library vegan:

- **`ordiplot`** - high-level plotting function, draws complete ordination diagrams (`plot` applied on result of ordination does the same).
- **`orditorp`** - adds labels onto existing ordination diagram, so as they are readable (they don't overlap). Has arguments `priority` (which species/sites will be given priority to draw the text instead of only point) and `select` (logical vector - which species/sites should be drawn).
- **`ordilabel`** - adds the labels which look like stickers; also has `priority` and `select` arguments (see `orditorp` function above).
- **`points`** - adds points to the ordination diagram (low-level plotting function, adds points to ordination diagram created by `ordiplot` or `plot`). Has argument `select` - logical vector indicating which elements should be displayed (the same for `text` below).
- **`text`** - similar to `points` above, adds the text labels for sites/species.
- **`ordipointlabel`** - creates new ordination diagram and adds both points and labels (for species or sites) in a way to minimize their overlap (uses iteratively optimizing algorithm and can be rather slow). Can be stored in an object further editable interactively by the function `orditkplot`.
- **`orditkplot`** - produces editable and clickable ordination diagrams, which can be exported, saved into R and also reedited again. Check blogpost of Gavin Simpson to see how to use it.
- **`ordispider`** - creates spiderplot by connecting individual members of the group with the group centroid.
- **`ordihull`** - draws envelope (convexhull) around the group of samples (possibly also color-filled polygon).
- **`ordiellipse`** - similar to envelopes - clouds of points within the group are encircled by ellipse.
- **`ordiarrows`** - draws arrows connecting the groups of samples (visualizing e.g. development of composition in time or in space).

From library vegan3d[2)]:

- **ordirgl** - draws 3D ordination diagram, using functionality of `library (rgl)`.
- **orglspider** - adds spiders to 3D ordination diagram created by `ordirgl`.

Custom functions:

- **ordicenter** - adds labels to centroids of groups onto ordination diagrams. Similar to `ordispider` with argument `label = TRUE`, but does not draw spider plot. Arguments and use are analogous to `ordispider` and `ordihull`. Definition here.
- **orglhull** - adds 3D convexhulls wrapping groups of samples (possibly transparent). Custom function not in `vegan` or `vegan3d`, requires `library (geometry)`. Definition here.

Example of use

The following plotting functions are using results of DCA (detrended correspondence analysis) of Vltava data:

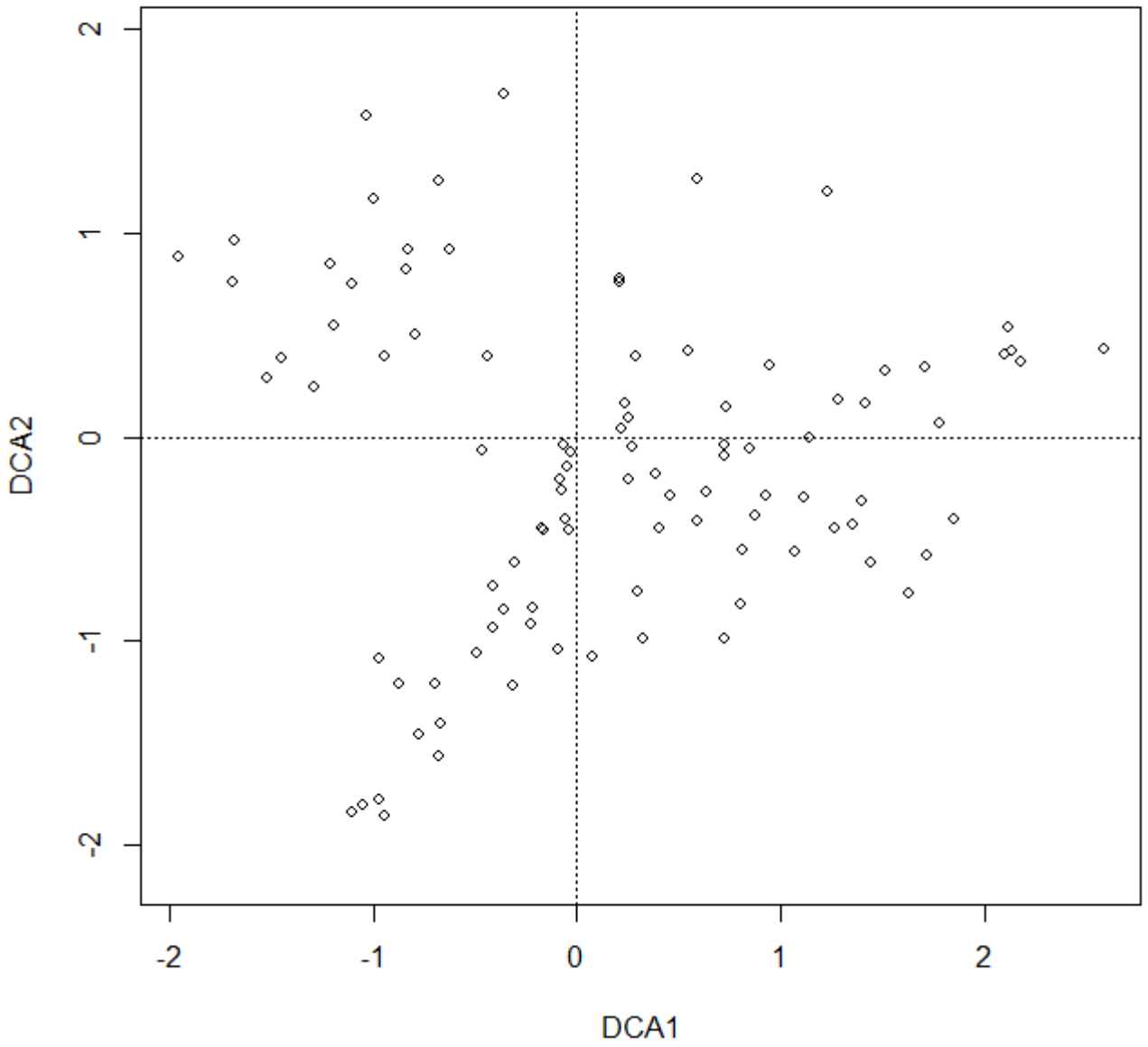
```
veg.data <- read.delim
('http://www.davidzeleny.net/anadat-r/data-download/vltava-spe.txt',
row.names = 1)
env.data <- read.delim
('http://www.davidzeleny.net/anadat-r/data-download/vltava-env.txt')

library (vegan)
DCA <- decorana (veg = log1p (veg.data))
```

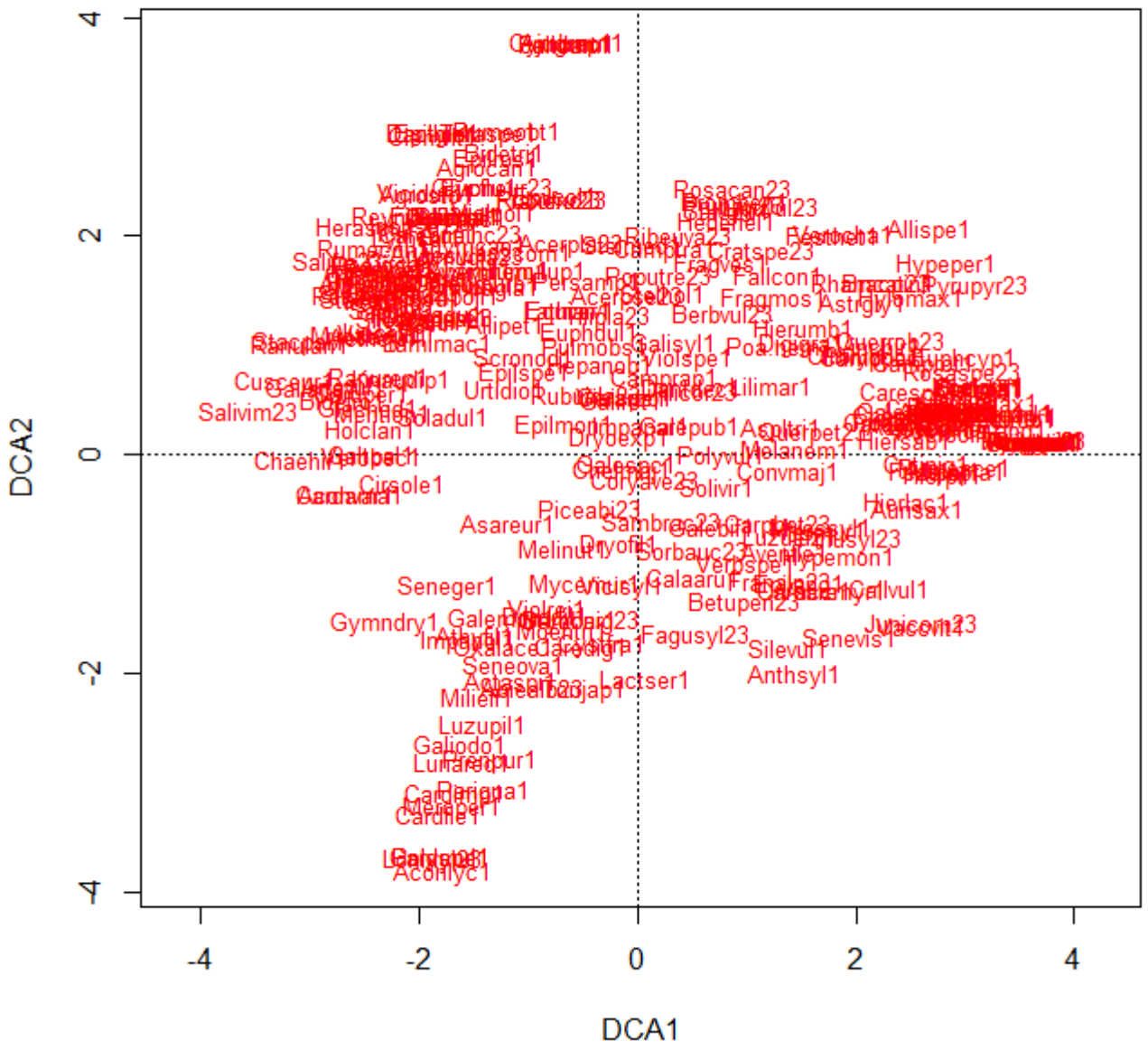
ordiplot (library vegan)

Draws ordination diagrams.

```
ordiplot (DCA, display = 'sites', type = 'p')
```



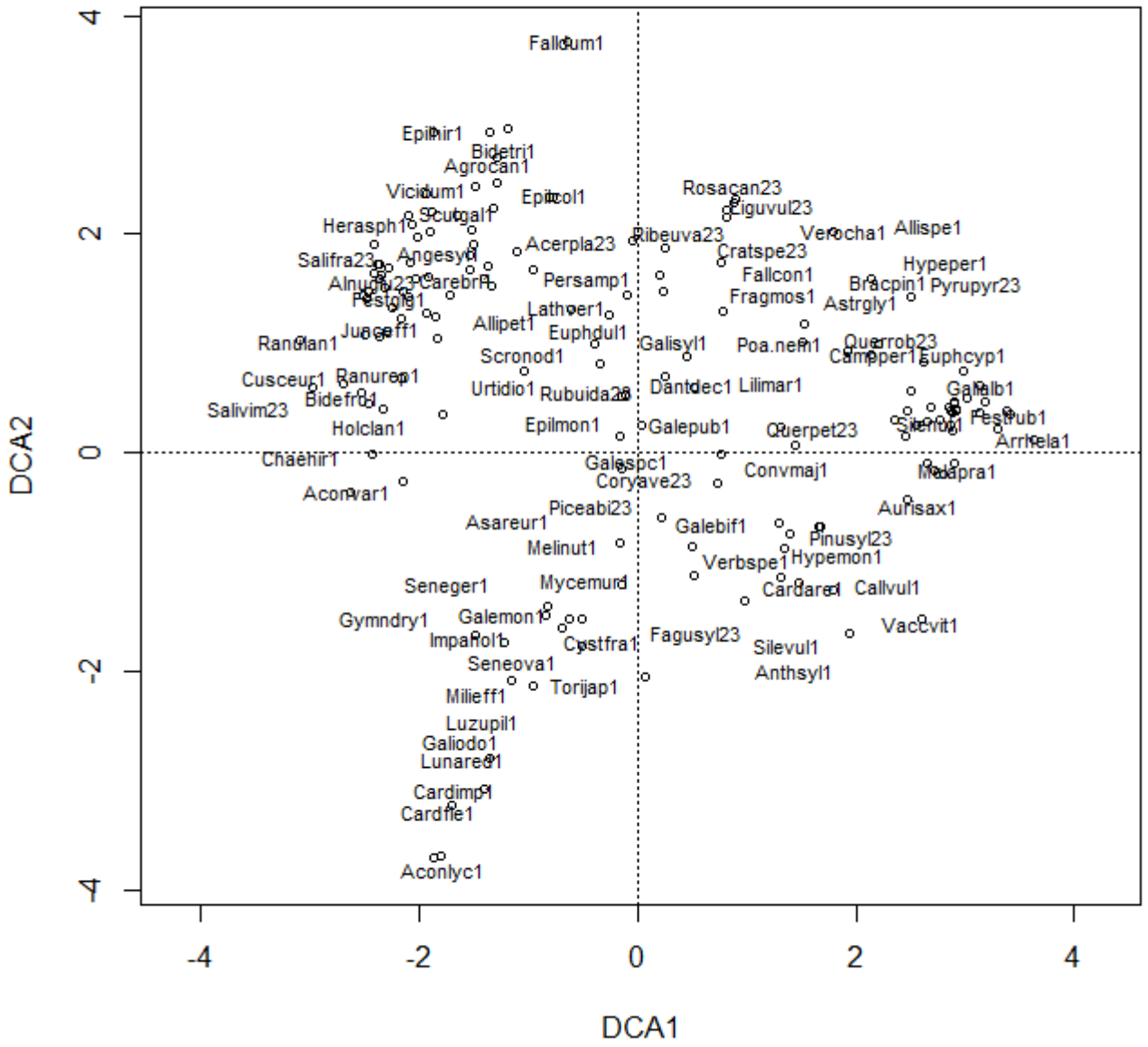

```
ordiplot (DCA, display = 'species', type = 't')
```



orditorp (library vegan)

Adds the labels onto an ordination diagram, so as they can be read (and don't overlap). First, draw empty diagram using function `ordiplot` and argument `type = 'n'`. Then add the labels using function `orditorp`.

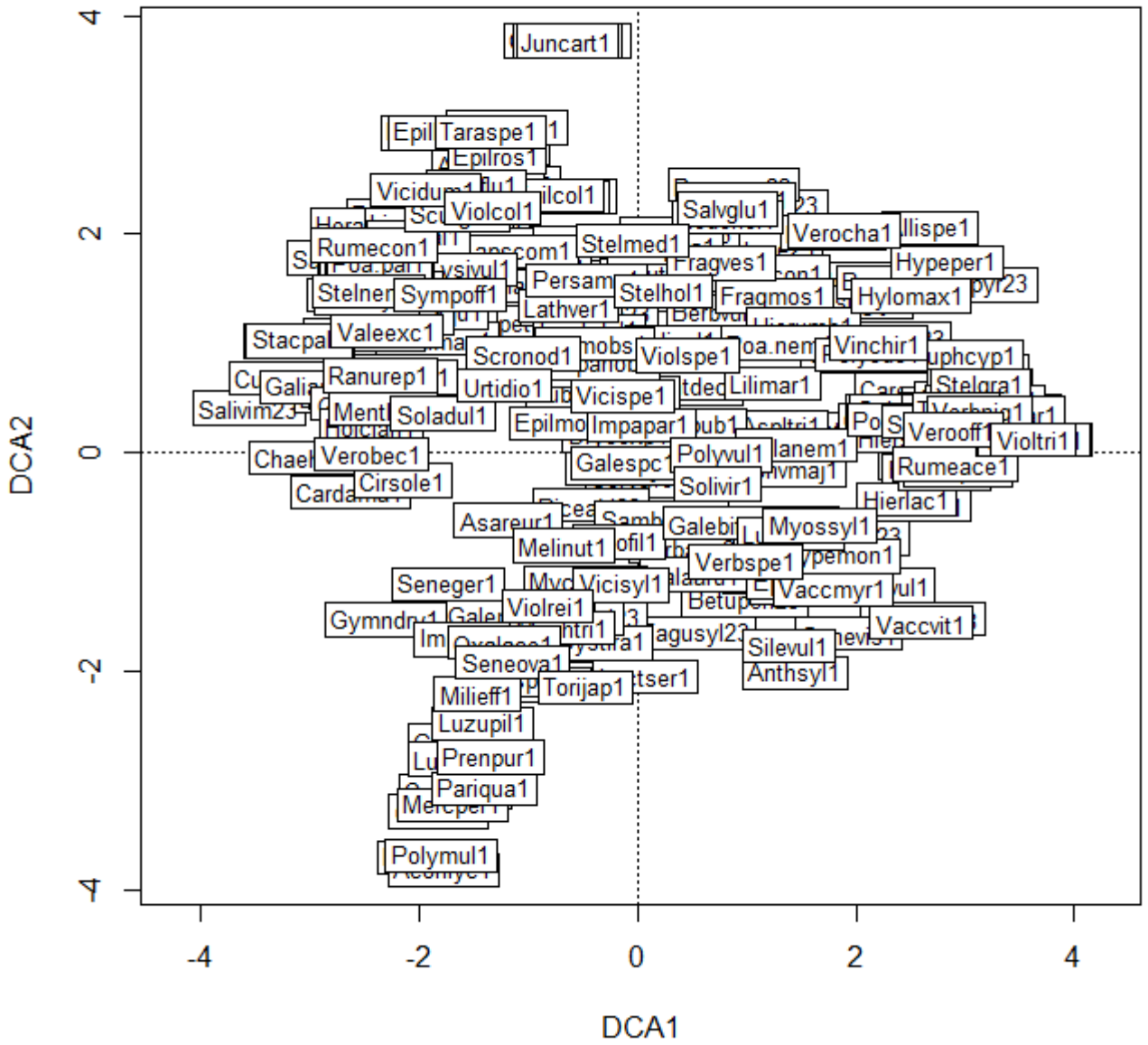
```
ordiplot (DCA, display = 'sp', type = 'n')
orditorp (DCA, display = 'sp')
```



ordilabel (library vegan)

Adds the labels which looks like stickers:

```
ordiplot (DCA, display = 'sp', type = 'n')
ordilabel (DCA, display = 'sp')
```

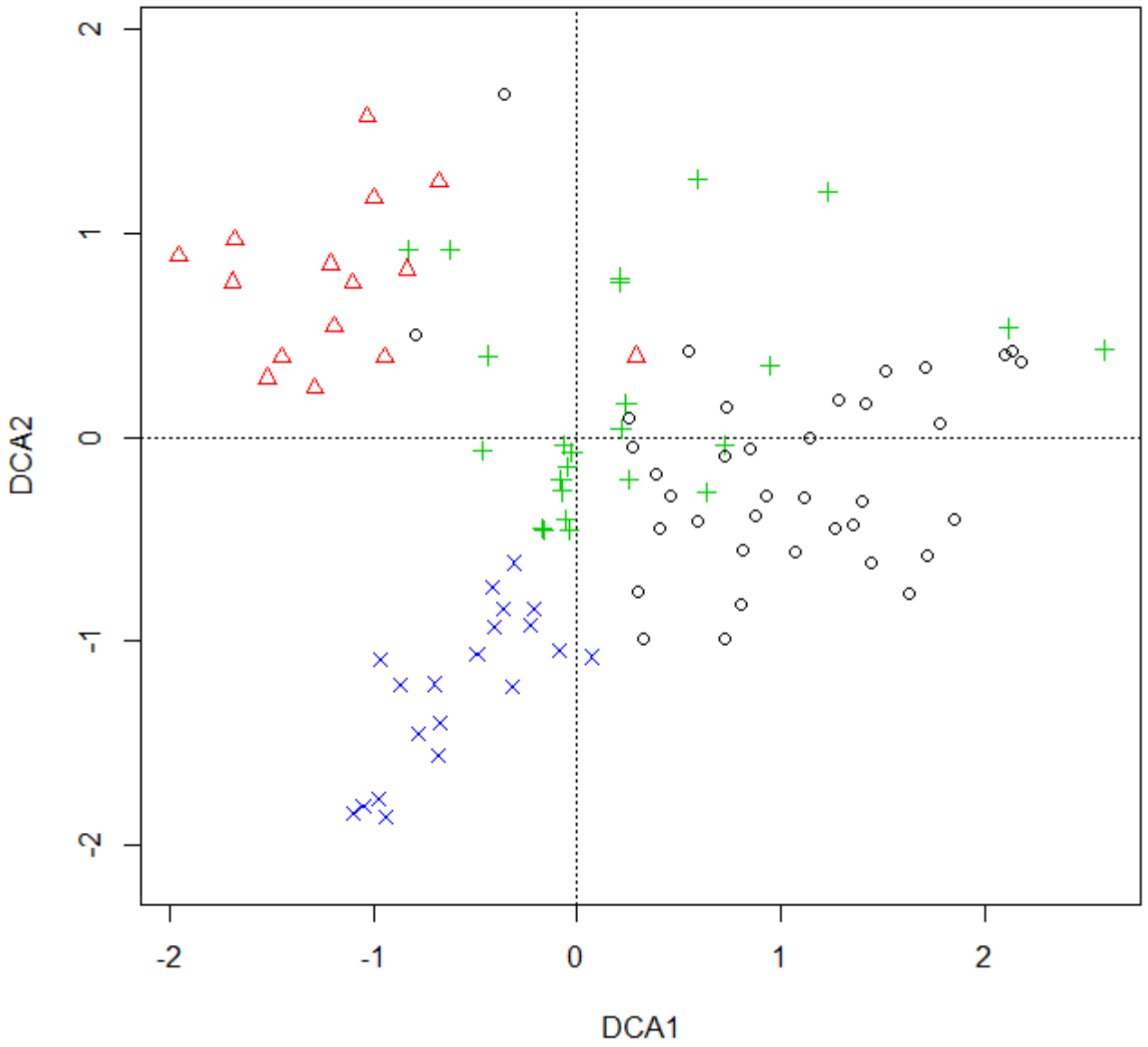


Other options: `identify.ordiplot` (adding the labels onto diagram by clicking the mouse), `orditklabel` (interactive diagrams in tcltk interface, but still not too elegant).

points (library vegan)

Adds points to ordination diagram - you can control their color, symbol, size etc. The following examples use the classification of samples (done by cluster analysis) into four groups, which is stored in variable GROUP in `env.data` data frame. First, draw empty ordination diagram (argument `type = 'n'` in function `ordiplot`) and add colourful points according to `env.data$GROUP` variable using function `points`:

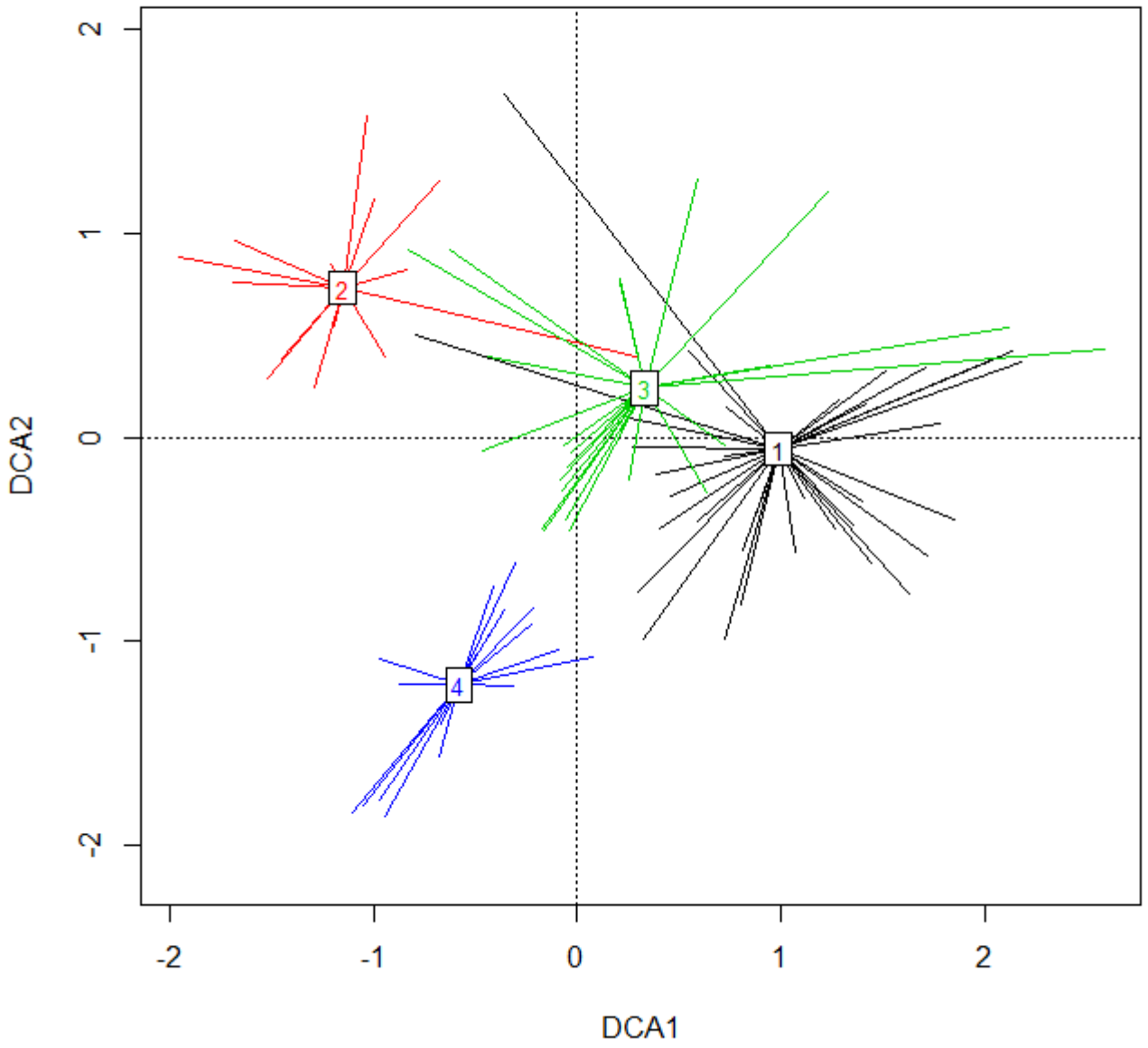
```
ordiplot (DCA, display = 'si', type = 'n')
points (DCA, col = env.data$GROUP, pch = env.data$GROUP )
```



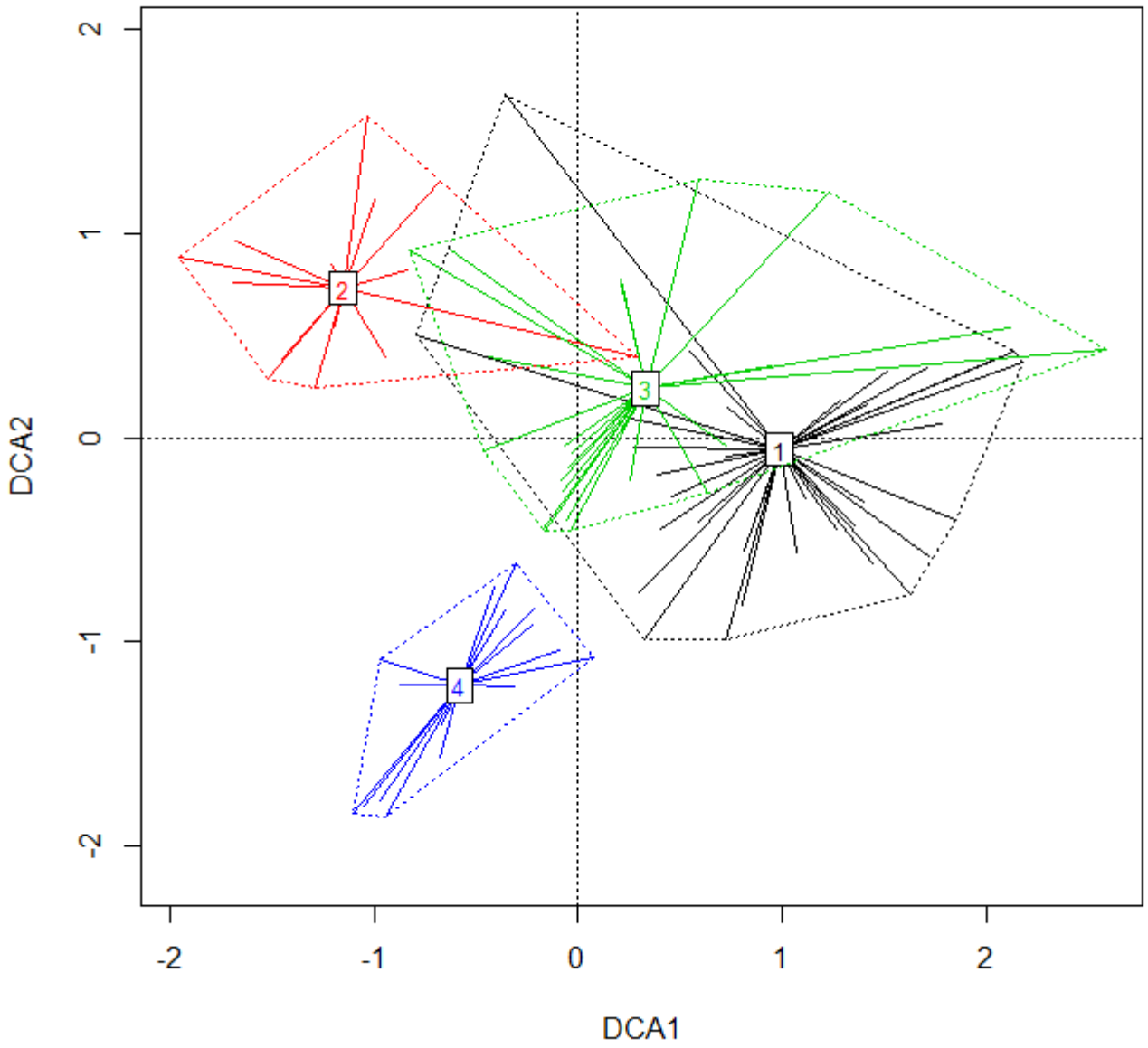
ordispider and ordihull (library vegan)

Function `ordihull` adds envelopes around groups of samples, while function `ordispider` adds so called spiders, connecting each sample with the centroid of particular group. Argument `label = T` draws the labels into the centroid of each spider:

```
ordiplot (DCA, display = 'si', type = 'n')
for (i in seq (1, 4)) ordispider (DCA, groups = env.data$GROUP, show.groups
= i, col = i, label = T)
```

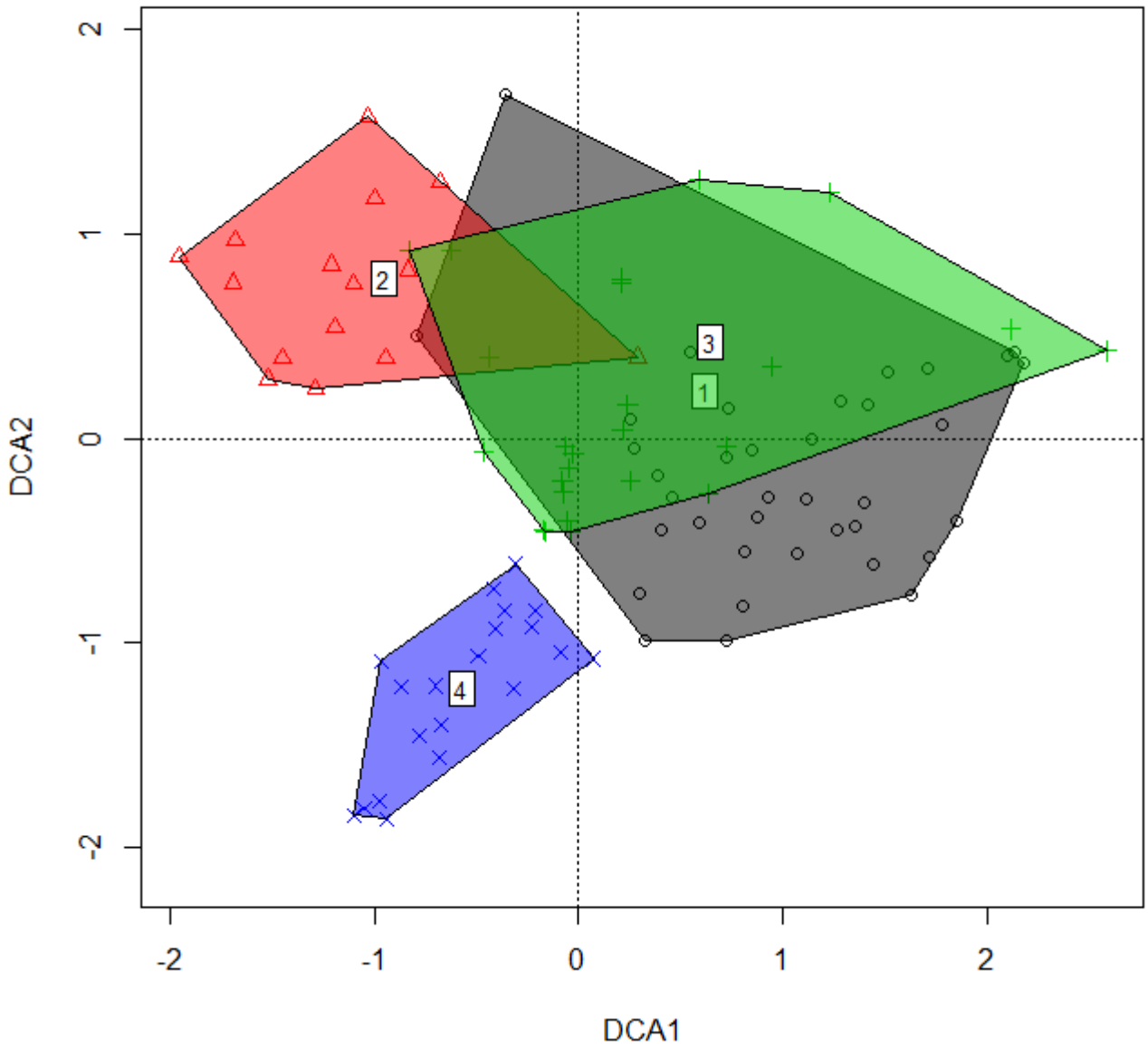



```
for (i in seq (1, 4)) ordihull (DCA, groups = env.data$GROUP, show.groups =
i, col = i, lty = 'dotted')
```



Function `ordihull` can draw not only envelopes, but also transparent polygons, by modifying the argument `draw = "polygon"` and optionally also transparency by argument `alpha` (default `alpha = 127` means that polygons will be semitransparent):

```
ordiplot (DCA, display = 'si', type = 'n')
points (DCA, col = env.data$GROUP, pch = env.data$GROUP)
for (i in unique (env.data$GROUP)) ordihull (DCA, groups = env.data$GROUP,
show.group = i, col = i, draw = 'polygon', label = T)
```



Note that setting `label = TRUE` also draws group labels, but this time not in the centroids of all the plots, but in centroids of plots which are part of envelope margin (that's why their position differs from spiderplot figure above)

ordicenter (custom function)

Adds labels into the group centroids. This can be usefull if you want to draw only the group centroids, not the envelopes or spiderplots (using `ordihull` or `ordispider`). Custom function (definition here), with code heavily borrowing from `ordispider` function in `vegan`, with the same arguments - check `?ordispider` for more details.

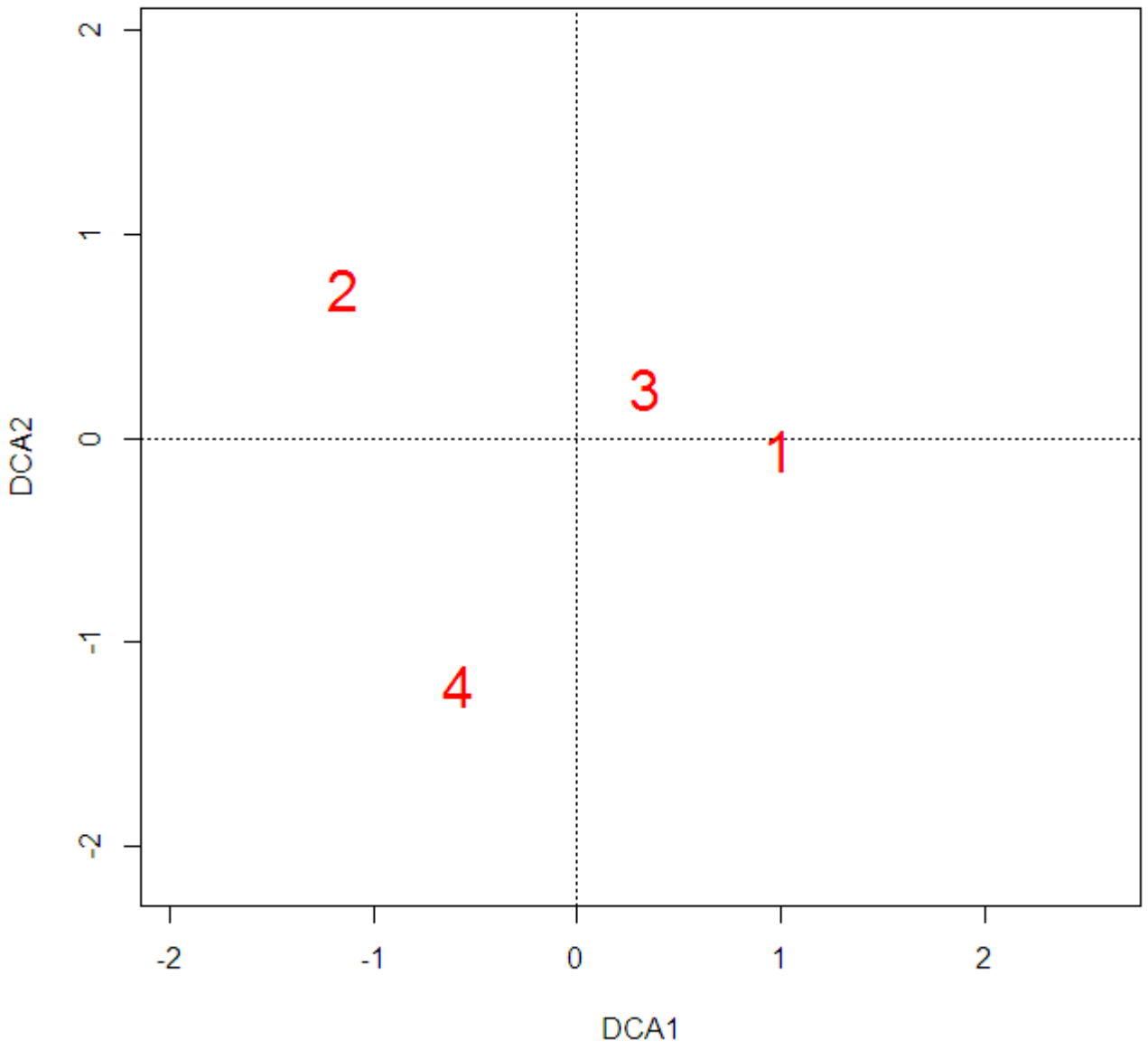
The function can be *sourced* from here in the following way:

```
source
```

```
('http://www.davidzeleny.net/anadat-r/doku.php/en:customized_functions:ordicenter?do=export_code&codeblock=0')
```

Draw plain ordination diagram with group numbers as centroids:

```
ordiplot (DCA, display = 'si', type = 'n')
ordicenter (DCA, groups = env.col = 'red', cex = 2)
```

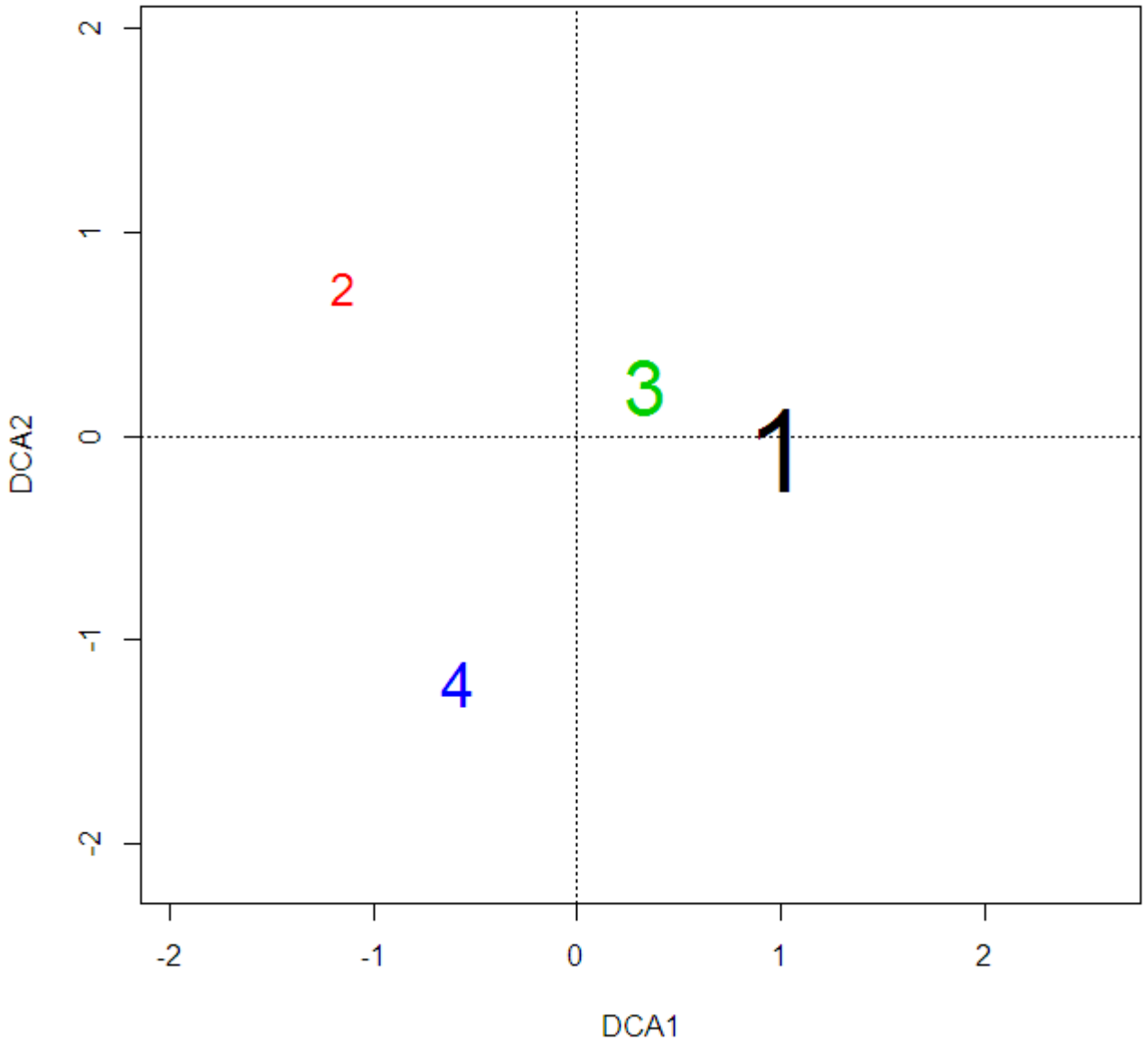


And a bit more advanced visualization - the size of the label is proportional to number of samples in the group, and labels differ by color:

```
ordiplot (DCA, display = 'si', type = 'n')
scaling.parameter <- as.vector (table (env.data$GROUP))/max (as.vector
(table (env.data$GROUP)))
for (i in 1:length (unique (env.data$GROUP)))
```

```
  ordicenter (DCA, groups = env.data$GROUP, show.groups = i, col = i, cex =
4*scaling.parameter[i])
```

(Note: the `scaling.parameter` simply calculates number of samples within each group (using function `table`, with output transormed into vector), and standardize these values between 0 and 1 by dividing them by the size of largest group. This scaling is in the next step (function `ordicenter`) used to multiply the maximum size (here 4) of the label in the argument `cex` to resize the centroid label.)



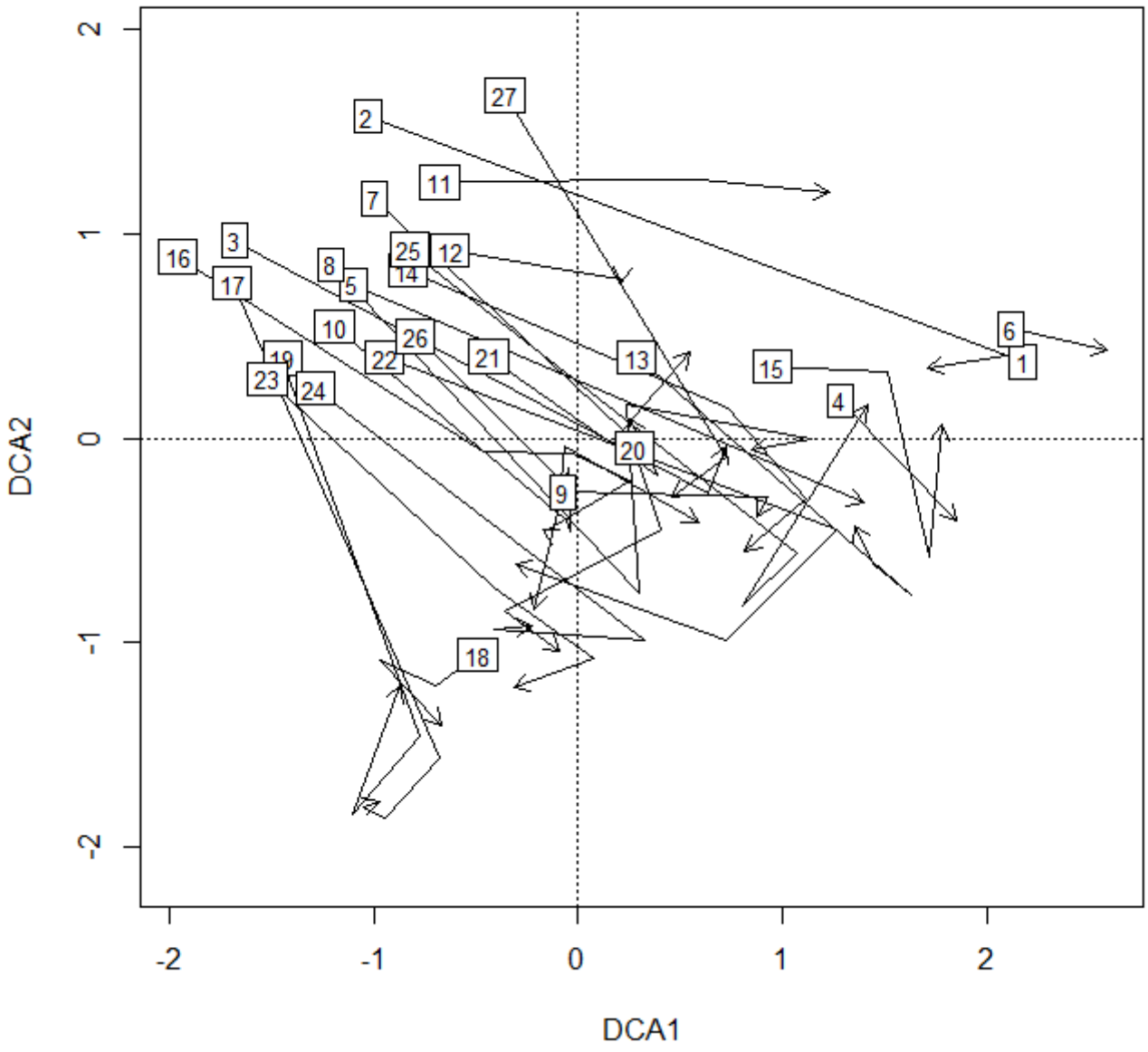
ordiarrows (library vegan)

This functions adds the arrows connecting the samples in certain order within a group. In case of Vltava data, this can be used to visualize the direction, in which changes the composition of vegetation along the transect going from the bottom of the valley (wet alluvial forest) toward the upper part of the valley (in either dry habitats on southern slopes or mesic habitats on northern

slopes).

```
ordiplot (DCA, display = 'si', type = 'n')
ordiarrows (DCA, groups = env.data$TRANSECT, order.by = env.data$ELEVATION,
startmark = 1, label = TRUE, length = .1)
```

Note the meaning of the arguments (there are several other coding options how to specify which samples belongs to which arrow and how to sort the order of samples within the groups):`groups` specifies samples belonging to the same group, which will be drawn by one arrow; `order` specifies the variable used to define the order of samples how they will be drawn within the arrow[3]; `label` defines whether the label with the group number should be drawn at the beginning of the arrow, and `length` is a specific argument which modified the behaviour of `arrows`, the underling function actually drawing the arrows (namely it makes the length of the arrow head shorter).

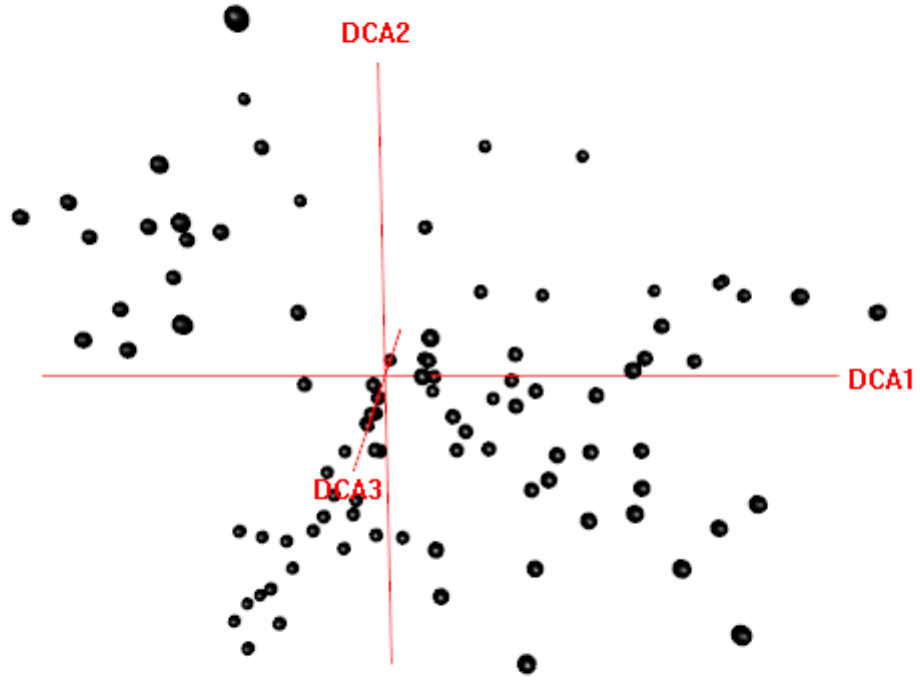


ordirgl (libraries vegan3d and rgl)

This function creates three dimensional ordination diagrams, which can be rotated by clicking left mouse button and moving, and zoomed in and out by the mouse wheel.

```
library (vegan3d)
ordirgl (DCA)
```

Fix Me![4)]

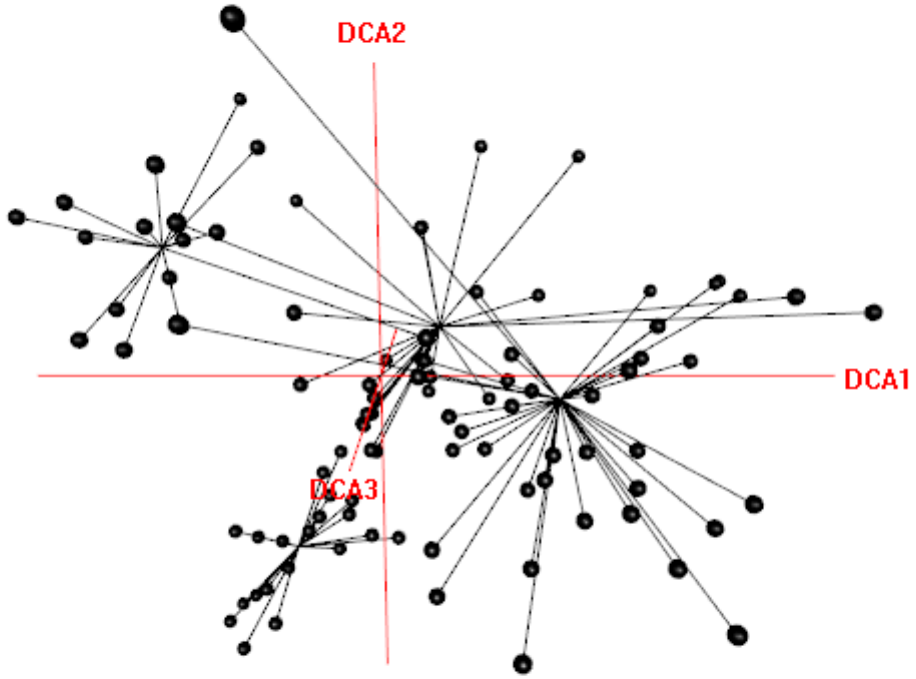


orglspider (library vegan3d and rgl)

3D spider plot.

```
orglspider (DCA, groups = env.data$GROUP)
```

Fix Me![5)]



orglhull (needs to be defined, requires library geometry)

Install library `geometry` first, if not yet done:

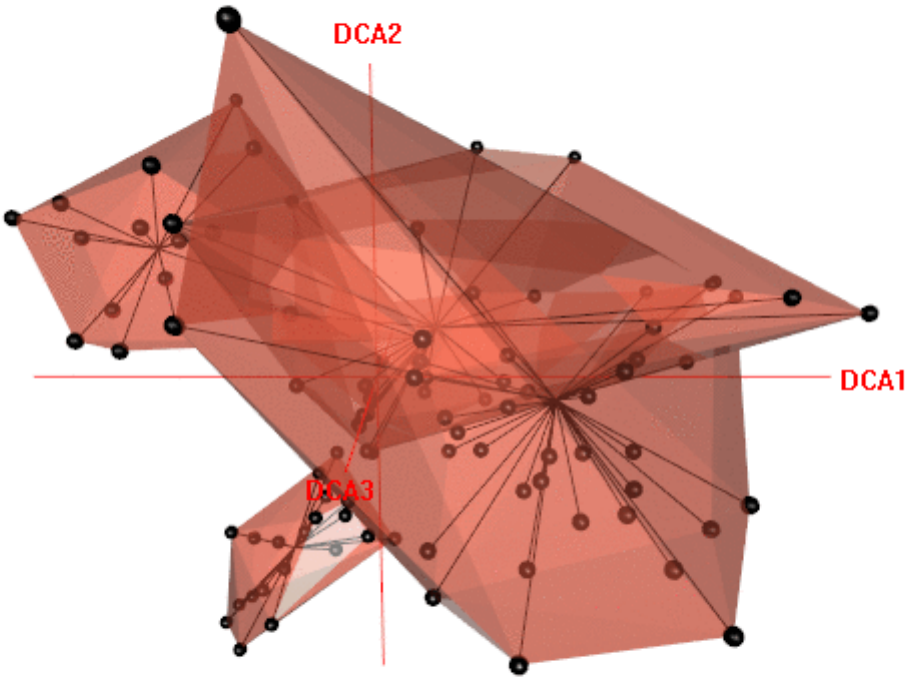
```
install.packages ('geometry')
```

The definition of the function (author D. Zelen??, fix for R version 2.15.0 introduced by Paolo Piras) can be found here: orglhull.

And here is how to use it:

```
orglhull (DCA, groups = env.data$GROUP)
```

Fix Me![6)]



1)

But you will need to calculate the whole analysis in CANOCO.

2)

Note that functions for drawing 3D ordination diagrams have been moved from vegan to a new package vegan3d - the reason according to this message from Jari Oksanen is that these function caused troubles with running or even installing vegan on some platforms, so to separate them was good for stability of vegan.

3)

In the specific case of this data it is important, since the samples in the table are sorted according to the order they have been collected in the field; I started in the upper part of the valley and go down, making the vegetation plots; than, once at the bottom, I have chosen locality of another transect and started to climb up, making the vegetation plots. Hence once the plots starts at the upper part of the transect, another time at the bottom...

4), 5), 6)

This figure is using the previous vltava dataset.