CBCS
http://agrobiol.sggw.waw.pl/cbcs

Communications in Biometry and Crop Science
Vol. 5, No. 1, 2010, pp. 34–40

International Journal of the Faculty of Agriculture and Biology,
Warsaw University of Life Sciences, Poland

SOFTWARE TRICKS AND TIPS

A simple R function for inspecting multivariate data

Marcin Kozak[1]*, Agnieszka Wnuk[1], Wojtek J. Krzanowski[2]

[1] Department of Experimental Design and Bioinformatics, Warsaw University of Life Sciences, Nowoursynowska 159, 02-776 Warsaw, Poland.
[2] School of Engineering, Mathematics and Physical Sciences, University of Exeter, UK and Department of Mathematics, Imperial College of Science, Technology and Medicine, UK.
* Corresponding author: Marcin Kozak, E-mail: nyggus@gmail.com

CITATION: Kozak, M., Wnuk, A., Krzanowski, W.J. (2010). A simple R function for inspecting multivariate data. *Communications in Biometry and Crop Science* 5 (1), 34–40.

Received: 11 March 2010, Accepted: 18 June 2010, Published online: 29 June 2010

ABSTRACT
The paper offers R code for a simple representation of multivariate data sets that provides a broad picture of any patterns in the data and highlights anomalous observations. The code can be used for both grouped and ungrouped data.

Key Words: *multivariate statistics; software.*

INTRODUCTION

In crop science, and in agricultural sciences in general, multivariate data sets—with n rows and *p* columns—are very common. Before applying any formal techniques of multivariate analysis (see, for example, Krzanowski 2000), and particularly when *n* is not too large, researchers may wish to inspect their data in order to spot any untypical or outlier observations, patterns or structures and the like. Here we propose a simple R code for such an inspection, making use of the `symnum()` function (R Development Core Team 2010). The algorithm for the representation is simple:

1. Transform (scale) values for each variable to zero mean and unit standard deviation (= standard normal deviate) by subtracting the mean and dividing by the sample standard deviation.
2. Truncate the scaled values, retaining only the integer part, i.e. the distance to the mean in standard deviation units.
3. Order the rows in the dataset by the sum of either the absolute or the signed values of the truncated scaled variables. (Optionally, the original ordering can be left.)
4. Form a table made up of the truncated values if they are smaller than –1 or bigger than 1, otherwise leaving a blank space in the cell.

In the third step, one can choose whether the absolute or the signed truncated scaled values are to be used. In the latter scenario, it is assumed that the sign counts, which would be the case for example when one attempts to spot cases with values higher or lower than the mean for all the variables. In the last step, we obtain a table in which each cell gives an immediate picture of the distance of its value from the mean of the relevant variable. Those values that are close to the mean (so are in the range of mean ± one standard deviation for that variable) are not presented at all (i.e. a blank space is inserted). In that way one can immediately spot untypical values in terms either of single values or whole rows.

DESCRIPTION OF THE FUNCTION

The following function will run this algorithm:

```
symb.mv <- function(x, grouping.variables = NULL,
    cols.for.order = 1:ncol(x), ord.method = "sum.abs", cutoff = NULL,
    ...) {
    if (nrow(x) < 2) stop("The matrix must have at least two cases")
       rn <- rownames(x)
    x <- trunc(scale(x))
    if (is.null(cutoff) == F) {
              sums <- rowSums(abs(x))
              o <- which(sums >= cutoff)
              if (length(o) < 2) {
               info <- ifelse(length(o) == 0, "No cases", "Only one case")
              info <- paste(info, "left after applying this cutoff value\n")
              cat(info)
              if (length(o) == 1) {
                     if (is.null(grouping.variables) == F)
                            x.print <- c(as.character(x[o,]),
                             as.character(grouping.variables[o, ]))
              print(x.print, quote = F)}
              stop("Please change the cutoff value")
              }
              x <- x[o,]
       }

       if (is.null(cols.for.order)) {o <- 1:nrow(x) }
              else if (length(cols.for.order) == 1) {
        if (ord.method == "sum.abs")
            o <- rev(order(abs(x[, cols.for.order])))
                           else o <- rev(order(x[, cols.for.order]))
    }
    else if (ord.method == "sum.abs")
        {o <- rev(order(rowSums(abs(x[, cols.for.order]), na.rm = TRUE)))}
              else o <- rev(order(rowSums(x[, cols.for.order], na.rm = TRUE)))
    x <- x[o, ]
    rownames(x) <- rn[o]
       x[x < -20] <- -20; x[x > 20] <- 20
    sr <- symnum(x, cutpoints = c((-21:-1) + 0.001, (1:21) -
        0.001), symbols = c(-20:-1, "", 1:20), legend = F, ...)
    srr <- data.frame(matrix(as.character(sr), byrow = F, ncol = ncol(sr)))
    if (is.null(grouping.variables) == F)
        srr <- cbind(srr, grouping.variables[o, ])
    colnames(srr) <- c(attributes(sr)$dimnames[[2]],
                            colnames(grouping.variables))
    rownames(srr) <- rownames(x)
    print(srr)
    if (any(srr == "?"))
        cat("NOTE: \"?\" indicates a missing value.\n")
    invisible(srr)
}
```

The argument `cols.for.order` enables one to choose which variables (by column numbers) should be used for ordering the rows; the default is all the variables in the data frame. It can be set to `NULL`, in which case the original ordering will be applied. The argument `grouping.variables` enables one to incorporate in the result any grouping variable(s). It is not used in any way in the algorithm, but it is included in the resulting table in order to draw attention to any *a-priori* grouping structure in the data, for example as a prelude to canonical variate analysis. The default setting `NULL` means that no such variable is incorporated. Table 1 explains all the arguments.

APPLYING THE FUNCTION

As an example, consider the famous iris data of Anderson (1935) popularized by Fisher (1936). The dataset is part of the base R installation and can be activated by:

```
> data(iris)
```

The simplest call to the symb.mv function, then, will be:

```
> symb.mv(iris[,1:4]) # the 5th column contains species names
```

which will produce (only a portion of the output is shown below)

```
     S.L S.W P.L P.W
132   2   1   1   1
119   2  -1   1   1
118   2   1   1   1
16        3  -1  -1
136   2       1   1
123   2       1   1
. . . . . . . . . . .
```

The result of the function could also be assigned to an object

```
> iris.symb <- symb.mv(iris[,1:4])
```

We can now print this object to a text file (in the current directory, which can be changed with the `setwd()` function):

```
> write.table(iris.symb, "iris_symb.txt", quote = F, sep = "\t")
```

or display portions of the table:

```
> iris.symb[1:10,]  # producing the first ten rows
> iris.symb[rownames(iris.symb) == "123",] # producing the 123rd row from the
original frame
> iris.symb[rownames(iris.symb) %in% c("1", "50", "123"),] # producing rows 1,
50 and 123 from the original frame
```

In the case of missing values, a warning is produced that indicates that the question mark is used (this is the default of the `symnum` function). Unspecified arguments to be passed to the `symnum` function (type `?symnum` to learn about them) can also be provided thanks to the "..." argument to the `symb.mv` function (refer to Table 1), as here:

```
> symb.mv(iris[,1:4], abbr.colnames = F)
```

after which the column names will not be abbreviated:

```
     Sepal.Length Sepal.Width Petal.Length Petal.Width
132             2           1            1           1
119             2          -1            1           1
118             2           1            1           1
16                          3           -1          -1
136             2                        1           1
123             2                        1           1
. . . . . . . . . . . . . .
```

We already know that the Iris data are grouped by Iris species, but this information was ignored above. We can here make use of the `grouping.variables` argument:

```
> symb.mv(iris[,1:4], grouping.variables = data.frame(Species =
          iris$Species))
```

resulting in:

```
      S.L S.W P.L P.W    Species
132   2   1   1   1  virginica
119   2  -1   1   1  virginica
118   2   1   1   1  virginica
16        3  -1  -1     setosa
136   2       1   1  virginica
123   2       1   1  virginica
  . . . . . . . . . . .
```

To display only the most interesting values, for which the sum of absolute truncated values are bigger than 5, change the `cutoff` argument:

```
> symb.mv(iris[, 1:4], cutoff = 5,
          grouping.variables = data.frame(Species = iris$Species))
```

with the result:

```
  S.L S.W P.L P.W Species
4   2   1   1   1  setosa
3   2  -1   1   1  setosa
2   2   1   1   1  setosa
1       3  -1  -1  setosa
```

Note that the `grouping.variables` argument requires a data frame, not a matrix or vector. We could also add the information about the species into the table by

```
> rownames(iris) <- paste(substr(iris$Species, 1, 2), rownames(iris), sep="_")
> symb.mv(iris[,1:4])
```

which yields:

```
         S.L S.W P.L P.W
vi_132 2   1   1   1
vi_119 2  -1   1   1
vi_118 2   1   1   1
se_16      3  -1  -1
vi_136 2       1   1
vi_123 2       1   1
. . . . . . . . . . .
```

We can treat the species independently:

```
> symb.mv(iris[iris$Species == "versicolor", 1:4])
> symb.mv(iris[iris$Species == "setosa", 1:4])
> symb.mv(iris[iris$Species == "virginica", 1:4])
```

We can pick out cases with the highest values of the four traits:

```
> data(iris)
> symb.mv(iris[, 1:4], ord.method = "sum")
```

with the result:

```
      S.L S.W P.L P.W
132   2   1   1   1
118   2   1   1   1
136   2       1   1
123   2       1   1
110   1   1   1   1
106   2       1   1
. . . . . . . . . . .
```

or with the widest and longest sepals:

```
> symb.mv(iris[, 1:4], cols.for.order = 1:2, ord.method = "sum")

132   2   1   1   1
118   2   1   1   1
16        3  -1  -1
136   2       1   1
123   2       1   1
110   1   1   1   1
. . . . . . . . . . .
```

Finally, the following code will produce a table in html format, which in certain situations can be easier to read than the corresponding table in R console. This needs the `hwriter` package (Pau 2010) to be installed.

```
symb.mv.www <- function(x, grouping.variables = NULL,
    cols.for.order = 1:ncol(x), ord.method = "sum.abs",
    css = "http:// agrobiol.sggw.waw.pl/~cbcs/articles/5_1_6/css_MK_CBCS.css",.)
{
    a <- symb.mv(x, grouping.variables, cols.for.order, ord.method,
        ...)
    aa <- as.matrix(a)
    rownames(aa) <- rownames(a)
    colnames(aa) <- colnames(a)
    p <- openPage("symb.html",
           title = "Symbolic representation of\nmultivariate data",
           link.css = css)
    if (any(aa == "?"))
        hwrite("\"?\" indicates a missing value", p, br = T)
    hwrite("<br><br>", p)
    hwrite(aa, p, border = 0)
    if (any(aa == "?"))
        hwrite("\"?\" indicates a missing value", p, br = T)
    closePage(p, splash = F)
    if (interactive())
        try(browseURL(file.path(paste(getwd(), "/symb.html",
            sep = ""))))
    else print(paste("Open the URL from", paste(getwd(), "/symb.html",
        sep = "")))
}
```

Run this simply by:

```
> library(hwriter)
> symb.mv.www(iris[,1:4], grouping.variables = data.frame(Sp =
          abbreviate(iris$Species, 2)))
```

The result can be seen in http://agrobiol.sggw.waw.pl/~cbcs/articles/5_1_6/example.html. More examples of the use of the functions, also describing the usefulness of the other arguments of the symb.mv function (Table 1), are presented at the web page accompanying this paper at http://agrobiol.sggw.waw.pl/~cbcs/articles/5_1_6/index.html, where functions and more examples are provided. To use the functions it suffices to type

```
> source("http://agrobiol.sggw.waw.pl/~cbcs/articles/5_1_6/functions_www.R")
```

which will download the functions into R, making them ready for use within the R console (although the `hwriter` package still needs to be loaded if one wants to use the `symb.mv.www` function).

A summary description of the two functions is presented in Tables 1 and 2. The code was tested in R 2.8.1 and 2.10.1 under Windows XP and Windows 7, and in R 2.10.1 under Linux Fedora 11.

Table 1. Description of the `symb.mv` function.

```
symb.mv

Description

This function supports inspection of multivariate data sets by means of symbolic
representation. The variables are scaled to have mean of 0 and variance of 1, and
these values are truncated. Original values are then replaced by truncated ones,
and those between -1 and 1 are replaced by a blank space. Ordering of rows is
carried out by the sum of either the absolute or the signed values of the
truncated scaled variables.

Usage

symb.mv <- function(x, grouping.variables = NULL,
    cols.for.order = 1:ncol(x), ord.method = "sum.abs", cutoff = NULL, ...)

Arguments

x                   a data frame or matrix of values to be presented symbolically;
                    x must have at least two rows
grouping.variables  an optional data frame of grouping variables; the default is
                    NULL, in which case no grouping variable is presented
cols.for.order      a vector of indices of columns in x to be used for ordering;
                    if omitted, all columns are used; if NULL, the original
                    ordering is used
ord.method          a way of ordering of rows based on selected columns with two
                    possible values: "sum.abs" does that by absolute scaled values
                    (the default), while "sum" by original scaled values
cutoff              a cutoff value, limiting the cases to be included in the
                    result table; only cases with the sum of absolute values of
                    the truncated data equal to or greater than cutoff will be
                    included; if there are only one or two corresponding cases, a
                    warning is produced
                    other (optional) arguments passed to the symnum() function
Value

The table is printed within the R console, and a data frame with the results is
returned.

Examples

data(iris)
symb.mv(x = iris[,1:4])
symb.mv(x = iris[,1:4], grouping.variables = data.frame(Species =
iris$Species))

# Refer to http://agrobiol.sggw.waw.pl/~cbcs/articles/5_1_6/demo_www.R for more
examples
```

Table 2. Description of the `symb.mv.www` function.

```
symb.mv.www

Description

This function supports the symb.mv function by returning a web page with the
resulting table. It requires the hwriter package. It requires the symb.mv function
to be downloaded into the R console.

Usage

symb.mv.www <- function(x, grouping.variables = NULL,
    cols.for.order = 1:ncol(x), ord.method = "sum.abs",
    css = "http:// agrobiol.sggw.waw.pl/~cbcs/articles/5_1_6/css_MK_CBCS.css",
...)

Arguments

x, grouping.variables,
cols.for.order,
ord.method                arguments passed to symb.mv function
css                       css file
                          other (optional) arguments passed to the symnum()
                          function
Value

The table is printed within the R console and to the URL.

Examples

data(iris)
symb.mv.www(x = iris[,1:4])
symb.mv.www(x = iris[,1:4],
              grouping.variables = data.frame(Species = iris$Species))

# Refer to http://agrobiol.sggw.waw.pl/~cbcs/articles/5_1_6/demo_www.R for more
examples
```

REFERENCES

Anderson, E. (1935). The irises of the Gaspe Peninsula. *Bulletin of the American Iris Society* 59, 2–5.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics* 7, Part II, 179–188.

Krzanowski, W.J. (2000). *Principles of Multivariate Analysis; A User's Perspective (Revised Edition)*. Oxford: Oxford University Press.

Pau, G. (2010). *hwriter: HTML Writer – Outputs R objects in HTML format. R package version 1.1.* http://www.ebi.ac.uk/~gpau/hwriter/

R Development Core Team (2010). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.