Christoph´s mixed R functions - Version 2021-09-08

This document contains an assembly of useful small snippets of R code, compiled and updated continuously.

 # Change axis labels and sizes in effects plots (John Fox´s "effects" library); the example below is for a multinomial model fit with nnet):

```
library(RColorBrewer)
mycolors=colorRampPalette(brewer.pal(12,"Paired"))(16)

plot(allEffects(m1),
  axes=list(
    y=list(cex=1.5,
        style="stacked",
        lab=list(cex=2,label="Relative abundance")
      ),
    x=list(cex=1.5,
      location.ord=list(
      lab=list(cex=2,label="Location")
      ))
    ),
  lines=list(col=mycolors),main="",
  lattice=list(key.args=list(columns=3))
)
```

 # change graphical output in effects plots (effects package):

```
trellis.device()
trellis.par.set(add.line=list(lwd=2))

e1=effect("myeffect",model12)

plot(e1,xlab=list(cex=2,label="mylabel"),ylab=list("mylabel",cex=2),
main="",lwd=2)

p1=plot(e1,xlab=list(cex=2,label="myeffect"),ylab=list("mylabel",cex=2),
main="")

update(p1,scales=list(x=list(cex=1.5),y=list(cex=1.5)))


effects:::plot.eff
lattice:::llines.default
getAnywhere("effect.llines")


trellis.device()
trellis.par.set(add.line=list(lwd=2))

p1=plot(effect("myeffect",model12),xlab=list(cex=2,label="mylabel"),ylab=li
st("myeffect",cex=2),main="",lwd=2)
update(p1,scales=list(x=list(cex=1.5),y=list(cex=1.5)),par.strip.text=list(
cex=2))

trellis.par.get()
```

 # Log transformation when data assume negative values

```
log.modulus=function(x,k)sign(x)*log(abs(x)+k) # k is an arbitrary constant
to avoid taking logs of zero# based on John & Draper (1980) J Roy Stat Soc
C 29: 190-197#
```

 # Empirical logit transformation

```
logit.e=function(p,e){
log((p+e)/(1-p+e))}

invlogit.e=function(x,e)(exp(x) + e*exp(x) - e)/(1+exp(x))
```

This function has been used in the following paper: Scherber Cet al. (2013) Ecology and Evolution 3:1449-1460.

 # Ranging and standardization functions

```
ranging=function(x)(x-min(x,na.rm=T))/(max(x,na.rm=T)-min(x,na.rm=T))
standardization=function(x)(x-mean(x,na.rm=T))/sd(x,na.rm=T)
```

These functions were used in the following paper: Scherber C. et al (2010) Nature 468, 553-556.

 # untransform a logarithmic y axis

```
x0=1:1000
x=log(x0)
y=log(x)


minx=min(x)
maxx=max(x)
prettylabels=pretty(exp(seq(minx,maxx,by=0.1)))

plot(y~x,type="n",axes=FALSE)
points(y~x,pch=16)
axis(1,at=seq(minx,maxx,length=length(prettylabels)),labels=prettylabels)
```

#Basic graphics settings (for publication-quality graphs)

```
par(
fin=c(6,4),pch=16,las=T,tck=0.02,cex.lab=1.5,
cex.axis=1.5,
mar=c(4,4,4,2),mgp=c(3,0.5,0),lwd=2)


mt=matrix(seq(1:3),nrow=3,ncol=3)
mt[,c(1,3)]=0

layout(mt,widths=c(0.1,0.3,0.1))
```

```
layout.show(3)
```

 # Non-linear mixed effects models with fixed and random effects

```
model3=nlme(response~SSmicmen(explanatory,Vm,K),data,
random=Vm+K~1,groups=~FACTOR,
fixed=list(Vm~Type,K~FACTOR), start=c(60,0,100,0))
```

 # Calculate Shannon diversity index

```
shannondiv=function(x)
{
ifelse((length(x)>0),

{df=data.frame(table(x))
ab.sum=sum(df$Freq,na.rm=T)
p.i=df$Freq/ab.sum
lnpi=log(p.i)
c(-sum(p.i*lnpi,na.rm=T))
}
,c(0))
}
```

 # Replace missing values by the mean (impute mean)

```
require(plyr)
impute.mean <- function(x) replace(x, is.na(x), mean(x, na.rm = TRUE))
```

 # Load multiple libraries

```
cc=c("soilDB","soilwater","soilphysics","soilprofile")

lapply(cc, require, character.only=T)
```

 # Shipley´s d-separation test for structural equation models

```
install.packages("gRbase")
library(gRbase)

source("http://bioconductor.org/biocLite.R")
biocLite("RBGL")
biocLite("ggm")

library(ggm)

myDAG=DAG(x5~x1+x2+x3+x4,x6~x5)
EDGE=edgematrix(myDAG)
ADJ=adjMatrix(EDGE)
plotGraph(ADJ)

dSep(myDAG, first="x1", second="x6", cond = "x5")
dSep(myDAG, first="x2", second="x6", cond = "x5")
```

```
dSep(myDAG, first=c("x1","x2","x3","x4"), second="x6", cond = "x5")
```

 # xyplot with error bars

```
panel.loc_and_scale <- function(x, y, fun1 = mean, fun2 = sd,
horizontal = TRUE,
lwd = add.line$lwd, lty = add.line$lty,
col, col.line = add.line$col, ...) {
x <- as.numeric(x)
y <- as.numeric(y)
add.line <- trellis.par.get( "add.line")
if (!missing(col)) {
if (missing(col.line))
col.line <- col
}
if (horizontal) {
vals <- unique(sort(y))
yy <- seq_along(vals)
xx1 <- xx2 <- numeric(length(yy))
for (i in yy) {
xx1[i] <- fun1(x[y == vals[i]])
xx2[i] <- fun2(x[y == vals[i]])
}
panel.arrows( x0 = xx1 - xx2, y0 = vals[yy],
x1 = xx1 + xx2, y1 = vals[yy],
code = 3, angle = 90, length = 0.1,
col = col.line, lty = lty, lwd = lwd)
panel.points( x = xx1, y = vals[yy], col = col, pch = 18)

}
else {
vals <- unique(sort(x))
xx <- seq_along(vals)
yy1 <- yy2 <- numeric(length(xx))
for (i in xx) {
yy1[i] <- fun1(y[x == vals[i]])
yy2[i] <- fun2(y[x == vals[i]])
}
panel.arrows( x0 = vals[xx], y0 = yy1 - yy2,
x1 = vals[xx], y1 = yy1 + yy2,
code = 3, angle = 90, length = 0.1,
col = col.line, lty = lty, lwd = lwd)
panel.points( x = vals[xx], y = yy1, col = col, pch = 18)
}
}


stripplot(yield ~ site, barley, groups = year, # fun1 = median, fun2 = IQR,
panel = function(x, y, groups, subscripts, ...) {
panel.grid(h = -1, v = 0)
panel.stripplot(x, y, ..., jitter.data = TRUE,
groups = groups, subscripts = subscripts)
panel.superpose(x, y, ..., panel.groups = panel.loc_and_scale,
groups = groups, subscripts = subscripts)
panel.superpose(x, y, ..., panel.groups = panel.average,
groups = groups, subscripts = subscripts)
},
auto.key = list(points = FALSE, lines = TRUE, columns = 2))
```

 # perform file manipulations with R

(sometimes even better than Windows Explorer!)

```
file.create(...)
file.exists(...)
file.remove(...)
file.rename(from, to)
file.append(file1, file2)
file.copy(from, to, overwrite = FALSE)
file.symlink(from, to)
dir.create(path, showWarnings = TRUE, recursive = FALSE)
```

#Add a regression line when using coplot (y~x|z)

```
coplot(log(biomass+1)~logdiv|leafs03+treat,
panel=function(x,y,...) {
#panel.smooth(x,y,span=.8,iter=5,...)
points(x,y)
abline(lm(y ~ x), col="blue")
} )
```

#Extract regression coefficients and display formula in a graph

```
regression1=lm(y~x)yachs=summary(regression1)$coefficients[1]
yachs.fehl=summary(regression1)$coefficients[3]
steig=summary(regression1)$coefficients[2]
steig.fehl=summary(regression1)$coefficients[4]
rsq=summary(regression1)$adj.r.squaredyachs=round(yachs,1)
yachs.fehl=round(yachs.fehl,1)
steig=round(steig,2)
steig.fehl=round(steig.fehl,2)
rsq=round(rsq,2)gleichung=paste("y=",yachs,"±",yachs.fehl,"+",steig,"±",ste
ig.fehl,"x"," r²=",rsq)
text(locator(1),gleichung)
```

Divert output to file:

```
ink("F:\\test.txt")
summary.lm(gb0)
sink()
```

Extract a random sample from a dataframe

```
#first, create a vector of length N that has as many elements as there
#are rows in the dataframe:

z=1:10#create an artificial dataframe:
factor=gl(2,5)
df=data.frame(z,factor)#now sample from the rows of the dataframe:
x=sample(z,4,replace=F)#extract the sampled rows:
df[sort(x),]
```

```
#######
#now apply this to a diversity-level dataframe:DL=gl(3,4)

subplot=rep(1:4,3)
diversity.data=data.frame(DL,subplot)
df <- data.frame( DL=gl(3,4), subplot=rep(1:4,3) )
df$index <- 1:nrow(df)
ind <- tapply( df$index, df$DL, function(x) sample(x,2) )
df[ unlist(ind), ] ## alternatively:tmp <- lapply( split(df, df$DL),
function(m) m[sample(1:nrow(m),2),] )
do.call("rbind", tmp)

#explains the usage of function(m):

lapply(split(df,df$DL),function(m) mean(m[,2]))
```

#Some useful graphics parameter and layout settings

```
par(fin=c(6,4),pch=16,las=T,tck=0.02,cex.lab=1.5,
cex.axis=1.5,mar=c(4,4,4,2),mgp=c(3,0.5,0),lwd=2)mt=matrix(seq(1:3),nrow=3,
ncol=3)
mt[,c(1,3)]=0layout(mt,widths=c(0.1,0.3,0.1))
layout.show(3)
```

#Import from Microsoft Excel (prior to Excel 2007):

```
z<-odbcConnectExcel("C:\\sample.xls")
frame<-sqlFetch(z,"sheet1") #or any other name of the Excel sheet
close(z)
frame
```

#Back-transform from logit scale (generalized linear model parameter estimates):

```
invlogit<-function(x)(exp(x)/(1+exp(x)))

#Levene test for homogeneity of variances:
levene.test <- function(y, group) {
meds <- tapply(y, group, median, na.rm=TRUE)
resp <- abs(y - meds[group])
table <- anova(lm(resp ~ group, na.action=na.omit))
row.names(table)[2] <- " "
cat("Levene's Test for Homogeneity of Variance\n\n")
table[,c(1,4,5)]
}
```

#Some useful pdf and graphics parameter settings

```
pdf("file.pdf",width=5,height=2.17,pagecentre=T,pointsize=10,fonts=c("Helve
tica"),

encoding="WinAnsi")par(pch=16,las=T,tck=0.02,mar=c(4,6,4,2),mgp=c(2.5,0.2,0
))
```

#reshape example

```
df=data.frame(
plot=gl(4,25),
block=gl(2,50),
conc1=sample(1:10,100,replace=T),
conc2=sample(1:10,100,replace=T),
conc3=sample(1:10,100,replace=T),
conc4=sample(1:10,100,replace=T))reshape(df,timevar="conc",varying=list(nam
es(df)[3:6]),
direction="long")
```

#Summarize data using mean plus-minus #standard error of the mean:

```
#summarize, Version 5.0
# produces mean +- standard errors for a given set of variables
# structure: summarize(a,b,b2,c)
# a any variable for which means & se shall be calculated
# b,b2 two factors upon which summaries for a shall be based
# c desired precision (number of decimal places)
# written by C.Scherber 6th May 2006
# last edited 14th April 2009 (to handle also missing values)


summarize2<-function(a,b,b2,c){
b<-as.factor(b)
b2<-as.factor(b2)
overall<-mean(a,na.rm=T)
se.overall<-sqrt( var(a,na.rm=T)/length(a) )

means1<-tapply(a,list(b,b2),function(x)mean(x,na.rm=T)) vars<-
tapply(a,list(b,b2),function(x)var(x,na.rm=T))  lengths<-
tapply(a,list(b,b2),function(x)length(x)) serrors1<-c(sqrt(vars/lengths))
means2<-round(tapply(a,b,function(x)mean(x,na.rm=T)),c) vars2<-
tapply(a,b,function(x)var(x,na.rm=T))     lengths2<-tapply(a,b,length)
serrors2<-round(c(sqrt(vars2/lengths2)),c) means3<-
round(tapply(a,b2,function(x)mean(x,na.rm=T)),c) vars3<-
tapply(a,b2,function(x)var(x,na.rm=T))    lengths3<-tapply(a,b2,length)
serrors3<-round(c(sqrt(vars3/lengths3)),2)

cat("Overall mean",round(overall,c),"+-",round(se.overall,c),"\n")
means.se1<-(matrix(c(means2,serrors2),nrow=length(levels(b)),ncol=2,
dimnames=list(levels(b),c("mean","std.error")))) means.se2<-
(matrix(c(means3,serrors3),nrow=length(levels(b2)),ncol=2,
dimnames=list(levels(b2),c("mean","std.error")))) means<-
(matrix(means1,nrow=length(levels(b)),ncol=length(levels(b2)),
dimnames=list(levels(b),levels(b2)))) serrors<-
(matrix(serrors1,nrow=length(levels(b)),ncol=length(levels(b2)),
dimnames=list(levels(b),levels(b2))))

return(means.se1,means.se2,means=round(means,c),serrors=round(serrors,c))
}
```

#Date conversions:

```
date1<-as.character(date)
date2<-
paste(substr(date1,1,2),substr(date1,4,8),"20",substr(date1,9,10),sep="")
z <- strptime(date2, "%d %b %Y")
as.Date(z)#use %m instead of %b if month is a numeric value
```

 # a complex xyplot example with multiple lines

```
#per panel and square-root scale on the y axis:

##This graph is based on a dataset collected in 2004, where the number of
##flower stalks of Common Sorrel (Rumex acetosa) is used as the response
##variable. Plants were grown in N=81 different artificial plant
communities
##that differed in the number and functional identity of plant species
##present.##The explanatory variables are:

## - presence of legumes
## - presence of grasses
## - insecticide-treated or not and
## - the number of plant species present.##The dataset was published in
2006 in Journal of Ecology.

library(lattice)#set up the data frame:

sqrtax=c(1.14017543,1.04880885,1,1.22474487,2.30217289,0.8660254,1.22474487
,2.12132034,

0.83666003,2.06155281,2.54950976,1.41421356,0.91287093,0.70710678,0.7071067
8,0.91287093,

1.22474487,1.6583124,1,1.41421356,1.44913767,0.8660254,1.14017543,1.6832508
2,0.8660254,

1.22474487,2.12132034,1.22474487,1.04880885,1.04880885,2.38746728,1.1180339
9,1.73205081,

1.97484177,1.32287566,1,1.22474487,1.58113883,1.5,1.58113883,1.47196014,1.6
583124,

1.68325082,1.93649167,1.22474487,1.3540064,0.8660254,0.70710678,1.58113883,
3.04138127,

1.04880885,1.44913767,0.8660254,0.70710678,0.8660254,2.42899156,2.41522946,
1.51657509,

1.04880885,0.8660254,1,0.70710678,3.01662063,0.91287093,1.04880885,2.121320
34,0.83666003,

2.39791576,1.22474487,1.14017543,1,1.32287566,1.5,2.02484567,1.08012345,2.5
0998008,

1.58113883,1.70293864,2.79880927,1.14017543,1.5,0.70710678,0.70710678,1.581
13883,

1.41421356,0.70710678,1.3540064,1.41421356,0.83666003,1.73205081,2.06155281
,1.5,
```

```
1.22474487,1.14017543,0.8660254,1.22474487,1.04880885,1.32287566,1.04880885
,0.70710678,

1.37840488,1.08012345,1.08012345,0.91287093,1.22474487,1.87082869,1.3784048
8,1,1.30384048,

1.64316767,1,1.58113883,2.19848433,1.30384048,1,1.87082869,1.22474487,1.224
74487,

1.22474487,1.5,1.08012345,0.70710678,1.58113883,1.22474487,1.22474487,0.836
66003,

0.70710678,1.44913767,3.56370594,0.70710678,1.22474487,1.3540064,1.11803399
,1.64316767,

1.87082869,1.22474487,0.8660254,0.83666003,1.11803399,0.83666003,2.54950976
,1.11803399,

1.04880885,2.07364414,0.9486833,2.38746728,1.14017543,1.47196014,0.70710678
,2.21359436,

0.70710678,1.70293864,1,3.75499667,1.30384048,1.22474487,2.54950976,1.22474
487)
sowndiv=c(16,8,8,4,2,16,2,1,16,8,4,8,1,2,2,1,4,16,4,60,4,2,60,1,1,4,2,4,16,
8,1,8,1,4,8,

16,2,2,8,16,1,2,4,8,1,8,2,16,4,1,4,60,16,1,2,8,2,16,4,16,60,16,1,4,8,4,8,1,
8,4,1,1,2,

2,8,2,16,16,2,4,16,8,8,4,2,16,2,1,16,8,4,8,1,2,2,1,4,16,4,60,4,2,60,1,4,2,4
,16,8,1,8,1,4,

8,2,2,8,16,1,2,4,8,8,1,8,2,16,4,1,4,60,16,2,8,2,16,4,16,60,16,1,4,8,4,8,1,8
,4,1,1,2,2,8,2,16,16,2,4)
GRASS=factor(c("Grasses","Grasses","Grasses","Grasses","No
Grasses","Grasses","No Grasses","No Grasses","No Grasses","No Grasses","No
Grasses","Grasses","No Grasses","Grasses","Grasses","No
Grasses","Grasses","No
Grasses","Grasses","Grasses","Grasses","Grasses","Grasses","No
Grasses","Grasses","No Grasses","No Grasses","No Grasses","Grasses","No
Grasses","No Grasses","Grasses","No Grasses","No Grasses","No
Grasses","Grasses","No Grasses","No Grasses","No Grasses","Grasses","No
Grasses","Grasses","Grasses","Grasses","Grasses","Grasses","Grasses","Grass
es","Grasses","No Grasses","Grasses","Grasses","No Grasses","No
Grasses","Grasses","No Grasses","No
Grasses","Grasses","Grasses","Grasses","Grasses","Grasses","No
Grasses","Grasses","No Grasses","No Grasses","Grasses","No
Grasses","Grasses","Grasses","Grasses","No Grasses","No Grasses","No
Grasses","Grasses","No Grasses","Grasses","No Grasses","No Grasses","No
Grasses","Grasses","Grasses","Grasses","Grasses","No Grasses","Grasses","No
Grasses","Grasses","No Grasses","No Grasses","No Grasses","Grasses","No
Grasses","Grasses","Grasses","No Grasses","Grasses","No
Grasses","Grasses","Grasses","Grasses","Grasses","Grasses","No Grasses","No
Grasses","No Grasses","No Grasses","Grasses","No Grasses","No
Grasses","Grasses","No Grasses","No Grasses","No Grasses","No Grasses","No
Grasses","No Grasses","Grasses","No
Grasses","Grasses","Grasses","Grasses","Grasses","Grasses","Grasses","Grass
es","Grasses","Grasses","No Grasses","Grasses","Grasses","No
Grasses","Grasses","No Grasses","No
Grasses","Grasses","Grasses","Grasses","Grasses","Grasses","No
Grasses","Grasses","No Grasses","No Grasses","Grasses","No
Grasses","Grasses","Grasses","Grasses","No Grasses","No Grasses","No
```

```
Grasses","Grasses","No Grasses","Grasses","No Grasses","No Grasses","No
Grasses"))
LEG=factor(c("Legumes","No Legumes","Legumes","Legumes","Legumes","No
Legumes","No Legumes","Legumes","No
Legumes","Legumes","Legumes","Legumes","No Legumes","No Legumes","No
Legumes","No Legumes","No Legumes","Legumes","No
Legumes","Legumes","Legumes","No Legumes","Legumes","No Legumes","No
Legumes","Legumes","Legumes","No Legumes","No Legumes","No Legumes","No
Legumes","Legumes","Legumes","Legumes","Legumes","Legumes","No
Legumes","Legumes","Legumes","Legumes","No Legumes","No
Legumes","Legumes","Legumes","No Legumes","Legumes","No Legumes","No
Legumes","No Legumes","Legumes","No Legumes","Legumes","Legumes","No
Legumes","No
Legumes","Legumes","Legumes","Legumes","Legumes","Legumes","Legumes","No
Legumes","No Legumes","Legumes","No Legumes","Legumes","No
Legumes","Legumes","No Legumes","Legumes","No Legumes","No Legumes","No
Legumes","Legumes","Legumes","No Legumes","Legumes","Legumes","Legumes","No
Legumes","Legumes","No Legumes","Legumes","Legumes","Legumes","No
Legumes","No Legumes","No Legumes","No
Legumes","Legumes","Legumes","Legumes","No Legumes","No Legumes","No
Legumes","No Legumes","No Legumes","Legumes","No
Legumes","Legumes","Legumes","No Legumes","Legumes","No
Legumes","Legumes","Legumes","No Legumes","No Legumes","No Legumes","No
Legumes","Legumes","Legumes","Legumes","Legumes","No
Legumes","Legumes","Legumes","Legumes","No Legumes","No
Legumes","Legumes","No Legumes","Legumes","No Legumes","Legumes","No
Legumes","No Legumes","No Legumes","Legumes","No
Legumes","Legumes","Legumes","No
Legumes","Legumes","Legumes","Legumes","Legumes","Legumes","Legumes","No
Legumes","No Legumes","Legumes","No Legumes","Legumes","No
Legumes","Legumes","No Legumes","Legumes","No Legumes","No Legumes","No
Legumes","Legumes","Legumes","No Legumes","Legumes","Legumes","Legumes","No
Legumes"))
treatment=factor(c(rep("Insecticide",78),rep("Control",80)))#change trellis
settings (colors, line types etc.):

chris.theme <- function() {
par <- col.whitebg()
par$strip.background$col<-c("white","grey")
par$add.text$col <- "mediumblue"
par$add.text$font <- 2
par$background$col <- "white"
par$superpose.line$lty <- c(1,4)
par$superpose.line$col[1:2] <- c("mediumblue","violetred")
par$superpose.symbol$pch[1:2] <- c(16,18)
par$superpose.symbol$col[1:2] <- c("mediumblue","violetred")
par
}

trellis.device(theme="chris.theme")

#Create the xyplot:

xyplot(sqrtax~sowndiv|treatment+GRASS,
groups=LEG,subscripts=T,
xlab=list("Number of plant species",cex=1.5),
ylab=list("Number of main axes",cex=1.5),
ylim=c(0,4),layout=c(2, 2),
aspect="fill",index.cond=list(c(2,1),c(1,2)),
key=list(space="top",border=TRUE,
columns=1,
transparent=FALSE,
points=list(pch=c(16,18),col=c("mediumblue","violetred"),cex=1),
```

```
lines=list(col=c("mediumblue","violetred"), lty=c(1,4), lwd=c(1.5,1.5)),
text=list(levels(LEG), cex=1.0)),
par.strip.text=list(cex=1.0),
strip = function(..., strip.names)
strip.default(...,strip.names=c(FALSE, TRUE),style = 1),
scales=list(alternating=c(3,3),tck=c(-1,-1),
x=list(at=c(1, 2, 4, 8, 16, 60), labels=c(1,2,4,8,16,60),log=TRUE,cex=1),
y=list(at=sqrt(seq(0,15,by=1)+0.5),labels=c(0,1,"",3,"","",6,"","",9,"","",
12,"","",15)),cex=1), panel=function(x, y, subscripts, groups){
panel.superpose(x, y, subscripts, groups,cex=1.0)
which <- groups[subscripts]
panel.abline(lm(y[which=="No Legumes"] ~x [which=="No Legumes"]),
lty=1, lwd=1.5)
panel.abline(lm(y[which=="Legumes"] ~ x[which=="Legumes"]),
lty=4, lwd=1.5)
})
```