Turtles all the way down? Psychometric approaches to the reduction problem
Kievit, R.A.

Citation for published version (APA):
Appendix B

R code for Package Simpsons

Description
This package detects instances of Simpson's Paradox in datasets of bivariate continuous data. It examines subpopulations in the data, either user-defined or by means of cluster analysis, to test whether a regression at the level of the group is in the opposite direction at the level of subpopulations.

Type    Package
Title    Detecting Simpson’s Paradox
Version  0.1.0
Date     2012-08-17
Author   Rogier Kievit & Sacha Epskamp
Depends R (> = 2.14.0), mclust
Maintainer Rogier Kievit <rogierkievit@gmail.com>
License  GPL-2
ByteCompile yes
Repository CRAN
Date/Publication 2012-08-23 14:41:48
NeedsCompilation no

References

Usage
Simpsons(X, Y, clusterid, clustervars, data, nreps)

Arguments
X The first continuous variable to be used in the regression analysis.
Turtles all the way down?

Y: The second continuous variable to be used in the regression analysis.

clusterid: If you have a vector describing group membership, such as gender, you can specify it here. This will then be used to test for possible instances of Simpson’s Paradox. If left empty, a cluster analysis will attempt to discover clusters in the data. See example 1.

Clustervars: By default, the cluster analysis will be carried out on the X and Y variables. If you want to define the clusters on the basis of a different set of variables, such as a questionnaire, you can specify them using this command. See example 3.

data: Describes the data matrix. Should be a dataframe.

nreps: Specifies the number of permutations run for each cluster in the permutation significance test. The default is 5000. Each repetition is stored in the matrix 'permutationtest'.

Details
This package detects instances of Simpson's Paradox in datasets. That is, it tests whether some bivariate relationship found at the level of the whole dataset is consistent (in direction and strength) for possible subpopulations. It examines whether there is evidence for more than one cluster in the data in the data using cluster analysis, either user-defined or by means of cluster analysis. Then, it plots the data, using a different colour for every cluster, plots the regression lines for each cluster, and estimates the regression of X on Y for each cluster. Finally, it tests whether the regression at the level of the whole dataset is different from the regression at the level of the subclusters.

Because clusters in the data are part of the whole dataset, and therefore create a dependency, a permutation test is used to test for significant differences. For each cluster, the cluster labels are permuted within the whole dataset, the regression is run within the cluster and the whole dataset, and the difference between these two betas is stored as 1 repetition of the null distribution and stored in the object 'permutationtest'. A regression is considered significantly different from the group if the difference in beta estimate exceeds the lower or upper 2.5 percent of the permuted null distribution. If this is the case, a warning is issued as follows: "Warning: Beta regression estimate in cluster X is significantly different compared to the group!". If the sign of the regression within a cluster is different (positive or negative) than the sign for the group and the beta estimate deviates significantly, a warning states "Sign reversal: Simpson's Paradox! Cluster X is significantly different and in the opposite direction compared to the group!"

Output
A list of class "Simpson" with the following elements:
Examples
#This section contains three examples of the types of analyses you can run
#using the 'Simpsons' function, illustrating the commands and the types of
#output.

#Example 1
Here, we want to estimate the relationship between 'Coffee' #and
'Neuroticism', taking into account possible gender differences.
#As we have measured gender, we supply this information using the
#clusterid' command.
#This means that the function runs the analysis both for #the dataset as a
whole and within the two subgroups.
#It then checks whether the subgroups deviate significantly #from the
regression at the level of the group.

#Simulating 100 males
coffeem=rnorm(1,1,15)
neuroticismm=(coffeem*.8)+rnorm(1,15,8)
clusterid=rep(1,1)
males=cbind(coffeem,neuroticismm,clusterid)

#Simulating 100 females
coffeef=rnorm(1,1,15)
neuroticismf=16 +((coffeef-.8)+rnorm(1,15,8))
clusterid=rep(2,1)
females=cbind(coffeef,neuroticismf,clusterid)
data=data.frame(rbind(males,females))
colnames(data) <- c("Coffee","Neuroticism","gender")

#'normal' data analysis: Plot & regression
plot(data[,1:2])
a=lm(data[,1]~data[,2])
Turtles all the way down?

abline(a)

summary(a) #A normal regression shows no effect

#Running a Simpsons Paradox analysis, using gender as known clustering variable
example1=Simpsons(Coffee,Neuroticism,clusterid=gender, data=data)
# Analyse the relationship between coffee and neuroticism for both males # and females.
example1

Example 2
Here we estimate the relationship between 'Coffee' and 'Neuroticism'.
#As opposed to example 1, we have not measured any possible clustering identifiers
#such as gender, so we want to estimate whether there is evidence for #clustering based #only on the data we measured: Coffee and Neuroticism.

generating data
Coffee1=rnorm(1,1,15)
Neuroticism1=(Coffee1*.8)+rnorm(1,15,8)
g1=cbind(Coffee1, Neuroticism1)
Coffee2=rnorm(1,17,15)
Neuroticism2=(3-(Coffee2*.8)+rnorm(1,15,8))
g2=cbind(Coffee2, Neuroticism2)
Coffee3=rnorm(1,14,15)
Neuroticism3=(2-(Coffee3*.8)+rnorm(1,15,8))
g3=cbind(Coffee3, Neuroticism3)
data2=data.frame(rbind(g1,g2,g3))
colnames(data2) <- c("Coffee","Neuroticism")

#'normal' data analysis: Plot & regression
plot(data2)
b=lm(data2[,1]~data2[,2])
summary(b)
abline(b)

# Running the analysis tool identifies three clusters, and warns that the relationship between alcohol and coffee is in the opposite direction in two of the subclusters.
example2=Simpsons(Coffee,Neuroticism,data=data2)
example2

Example 3

#In this final example, we want again want to analyse the relationship # between 'Alcohol' and 'Mood'. However, this time #we have reason to believe that responses to a questionnaire #will fall into clusters of response types. Therefore, we want to # estimate the clusters in the data on the basis of a different set
# of variables. In this case, we have simulate three types of responses
# to a questionnaire of nine questions, with continuous responses
# ranging between 1 and 7. We then first estimate the clusters on
# the basis of the questionnaire, and then examine the relationship # between
'Alcohol' and 'Mood' based on these detected clusters.

#group 1

signal=matrix(rnorm(3 ,7,1),1 ,3)
noise=matrix(rnorm(6 ,3.5,1),1 ,6)
g1=cbind(signal,noise)

#group 2

signal=matrix(rnorm(3 ,1,1),1 ,3)
noise=matrix(rnorm(6 ,3.5,1),1 ,6)
g2=cbind(noise, signal)

#group 3

signal=matrix(rnorm(3 ,7,1),1 ,3)
noise1=matrix(rnorm(3 ,3.5,1),1 ,3)
noise2=matrix(rnorm(3 ,3.5,1),1 ,3)
g3=cbind(noise1,signal,noise2)

questionnaire=rbind(g1,g2,g3)
colnames(questionnaire)=c('q1','q2','q3','q4','q5','q6','q7','q8','q9')

Alc1=rnorm(1 ,1 ,8)
Mood1=(Alc1*.4)+rnorm(1 ,3,4)
A=cbind(Alc1, Mood1)
Alc2=rnorm(1 ,15,8)
Mood2=(Alc2*-.4)+rnorm(1 ,3,4)
B=cbind(Alc2,Mood2)
Alc3=rnorm(1 ,2 ,8)
Mood3=(Alc3*.8)+rnorm(1 ,3,4)
C=cbind(Alc3,Mood3)
data=data.frame(rbind(A,B,C))
colnames(data) <- c("Alcohol","Mood")
alldata=cbind(questionnaire,data)
alldata=as.data.frame(alldata)

#Run Simpsons Paradox detection algorithm, clustering on the basis of the
questionnaire
example3=Simpsons(Alcohol,Mood,clustervars=c("q1","q2","q3","q4",
'q5'"q6'"q7'"q8'"q9"),data=alldata)
example3
function (X, Y, clusterid, clustervars, data, nreps = 5000)
{
    if (missing(X) | missing(Y))
        stop("'X' and 'Y' must be specified")
    EstCluster <- missing(clusterid)
    ClustGiven <- !missing(clustervars)
    mc <- match.call()
    namex = as.character(mc["X"])
    namey = as.character(mc["Y"])
    mclustanalysis = NULL
    if (!missing(data)) {
        if (!is.data.frame(data))
            stop("'data' not a data frame")
        if (is.null(data[[namex]]))
            stop(paste("Variable", namey, "does not exist in dataframe"))
        if (is.null(data[[namey]]))
            stop(paste("Variable", namex, "does not exist in dataframe"))
        alldata <- data.frame(X = data[[namex]], Y = data[[namey]])
        if (!EstCluster) {
            if (is.null(data[[as.character(mc["clusterid"])]]))
                stop(paste("Variable", as.character(mc["clusterid"])),
                     "does not exist in dataframe")
            alldata$clusterid <- data[[as.character(mc["clusterid"])]]
        }
        else {
            alldata <- data.frame(X = X, Y = Y)
            if (!EstCluster) {
                alldata$clusterid <- clusterid
            }
        }
    }
    if (EstCluster) {
        if (ClustGiven) {
            if (!missing(data)) {
                varNames <- clustervars
                for (var in varNames) {
                    if (is.null(data[[var]]))
                        stop(paste("Variable", var, "does not exist in dataframe"))
                }
                ClustData <- data[varNames]
            }
            else {
                ClustData <- as.data.frame(do.call("cbind", clustervars))
            }
            mclustanalysis <- suppressWarnings(Mclust(ClustData))
        }
        else {
            mclustanalysis <- suppressWarnings(Mclust(alldata[2("X",

Turtles all the way down?

Source Code for package ‘Simpsons’
Psychometric approaches to the reduction problem

```
Nclusters <- mclustanalysis$G
Alldata$clusterid <- as.vector(mclustanalysis$classification)

Allclusters <- list()
Nclusters = max(Alldata$clusterid)
Npercluster <- matrix()
Allint <- matrix()
Allbeta <- matrix()
pvalues <- matrix()
for (i in 1:Nclusters) {
  x = subset(Alldata, clusterid == i)
  assign(paste("C", i, sep = ""), x)
  Allclusters[[i]] = x
  Npercluster[i] = nrow(x)
  reg = lm(x[, 2] ~ x[, 1])
  Allint[i] = reg$coefficients[1]
  Allbeta[i] = reg$coefficients[2]
  pvals = summary(reg)
  pvals = pvals$coefficients
  pvalues[i] = pvals[8]
}
plot(Alldata[, 1], Alldata[, 2], col = Alldata[, 3] + 1,
     pch = Alldata[, 3], xlab = namex, ylab = namey)
Allbeta = as.numeric(Allbeta)
Allint = as.numeric(Allint)

groupreg = lm(Alldata[, 2] ~ alldata[, 1])
groupint = groupreg$coefficients[1]
groupbeta = groupreg$coefficients[2]
pvalsgr = summary(groupreg)
pvalsgr = pvalsgr$coefficients
pvaluesgr = pvalsgr[8]
for (j in 1:Nclusters) {
  a = Allint[j]
  b = Allbeta[j]
  abline(a, b, lty = j, col = j + 1)
}
abline(groupint, groupbeta, lwd = 3, col = 1)
xpd <- par("xpd")
par(xpd = TRUE)
legend(mean(par("usr")[, c(1, 2)]), par("usr")[, 4], paste("cluster",
      1:Nclusters), col = 1:Nclusters + 1, pch = 1:Nclusters,
      bty = "n", horiz = TRUE, xjust = 0.5, yjust = 0, cex = 1)
par(xpd = xpd)
totaln = nrow(Alldata)
permutationtest = matrix(NA, nrow = nreps, ncol = Nclusters)
lowerboundary = matrix(NA, ncol = Nclusters)
upperboundary = matrix(NA, ncol = Nclusters)
if (Nclusters > 1)
  message(paste(Nclusters, "clusters detected"))
```
if (Nclusters == 1)
message(paste(Nclusters, "cluster detected"))
for (i in 1:Nclusters) {
clustdata = subset(alldata, clusterid == i)
clustern = nrow(clustdata)
if (Nclusters > 1) {
messgmessage(paste("Permuting cluster", i))
for (j in 1:nreps) {
permid = rep(0:1, c(totaln - clustern, clustern))
permid = sample(permid, size = totaln, replace = F)
permalldata = cbind(alldata, permid)
permclust = subset(permalldata, permid == 1)
clusterbeta = lm(permclust[, 2] ~ permclust[, 1])
clustbeta = clusterbeta$coefficients[2]
groupbeta = lm(permalldata[, 2] ~ permalldata[, 1])
nulldif = clustbeta - groupbeta
permutationtest[i, j] = nulldif
setTxtProgressBar(pb, j)
}
close(pb)
lowerboundary[i] = quantile(permutationtest[, i], 0.025)
upperboundary[i] = quantile(permutationtest[, i], 0.975)
}
groupsign = sign(groupbeta)
custersign = sign(Allbeta)
for (i in 1:Nclusters) {
group = groupbeta
clustbeta = Allbeta[i]
significance = ((group - clustbeta) < lowerboundary[i] ||
(group - clustbeta) > upperboundary[i])
csign = clustersign[i]
if (!is.na(significance)) {
if (groupsign != csign & significance) {
message(paste("Sign reversal: Simpson's Paradox! Cluster ",
i, " is significantly different and in the opposite direction compared to the
group!",
sep = " "))
}
else if (significance) {
message(paste("Warning: Beta regression estimate in Cluster ",
i, " is significantly different compared to the group!",
sep = " "))
}
else {

}
message(paste("No evidence for Simpson's Paradox: Cluster",
i, "is not significantly different compared to the group regression",
sep = " "))
}
else {
message("One cluster detected: No evidence for Simpson's Paradox")
}
}
results = list(Nclusters = Nclusters, clustersize = tapply(alldata$clusterid,
alldata$clusterid, length), alldata = alldata, Allbeta = Allbeta,
Allint = Allint, permutationtest = permutationtest, namex = namex,
namey = namey, pvalues = pvalues, groupbeta = groupbeta,
groupint = groupint, groupint = groupint, pvaluesgr = pvaluesgr,
totaln = totaln, mclustanalysis = mclustanalysis)
class(results) <- "Simpson"
return(results)

<bytecode: 0x000000000fe9e660>
<environment: namespace:Simpsons>