Package 'dgmb'

October 22, 2015

Type Package

Title Simulating Data for PLS Mode B Structural Models

Version 1.2

Date 2015-10-02

Author Alba Martinez-Ruiz <amartine@ucsc.cl> and Claudia Martinez-Araneda <cmartinez@ucsc.cl>

Maintainer Alba Martinez-Ruiz <amartine@ucsc.cl>

Depends R (>= 3.1.0), tcltk, MASS, abind

Description

A set of functions have been implemented to generate random data to perform Monte Carlo simulations on structural models with formative constructs and interaction and nonlinear effects (Two-Step PLS Mode B structural models). The setup of the true model considers a simple structure with three formative exogenous constructs related to one formative endogenous construct. The routines take into account the interaction and nonlinear effects of the exogenous constructs on the endogenous construct.

SystemRequirements tktable, BWidget

License GPL (>= 2)

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2015-10-22 15:23:16

R topics documented:

---

dgmb-package *Simulating Data for PLS Mode B Structural Models*

Description

A set of functions have been implemented to generate random data to perform Monte Carlo simulations on structural models with formative constructs (PLS-Mode-B models). The setup of the true model considers a simple structure with three formative exogenous constructs related to one formative endogenous construct. The routines take into account the interaction and nonlinear effects of the exogenous constructs on the endogenous construct.

A graphical user interface allows to fix the simulation parameters such as the number of replications, the sample size of each data set and the number of indicators for each block of variables. Using binary matrices, it is possible to set the structural relationships between constructs as well as nonlinear and interaction effects. A set of permissible outer weights and path coefficients is available to run the procedure and to obtain the data sets.

Details

Type:	Package
Title:	Simulating Data for PLS Mode B Structural Models
Version:	1.2
Date:	2015-10-02
Depends:	abind, tcltk, MASS
License:	GPL <>=2
LazyLoad:	yes

Author(s)

Alba Martinez-Ruiz <amartine@ucsc.cl>, Claudia Martinez-Araneda <cmartinez@ucsc.cl> Maintainer: Alba Martinez-Ruiz <amartine@ucsc.cl>

References

Chin, W. W., Marcolin, B. L., and Newsted, P. R. 2003. A partial least squares latent variable modeling approach for measuring interaction effects: Results from a Monte Carlo simulation study and an electronic-mail emotion/adoption study. *Information Systems Research*, **14**(**2**), 189-217.

Dalgaard's Peter (2002) *Rnews: The Newsletter of the R Project*, Vol. 3.

Hanafi, M. 2007. PLS path modeling: Computation of latent variables with the estimation mode B. *Computational Statistics*, **22**, 275-292.

Martinez-Ruiz, A., Aluja-Banet, T. (2013) Two-step PLS path modeling mode B: Nonlinear and interaction effects between formative constructs. In *New Perspectives in Partial Least Squares and Related Methods*, eds H. Abdi, W. Chin, V. Esposito Vinzi, G. Russolillo, and L. Trinchera, Springer Proceedings in Mathematics and Statistics, volume **56**, pp. 187-199.

R Development Core Team (2011).*R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria.* ISBN 3-900051-07-0, http://www.r-project.org/

Tenenhaus, M., Esposito Vinzi, V., Chatelin, Y. M., and Lauro, C. 2005. PLS path modeling. *Computational Statistics & Data Analysis*, **48**, 159-205.

See Also

dgmbGui

Examples

```
dgmbGui()
help(dgmbGui)
```

---

dgmbGui *Graphical user interface (GUI) for random data generation*

Description

A function to implement a graphical user interface (GUI) to fix simulation parameters.

Usage

```
dgmbGui()
```

Details

The following main window is displayed in R:

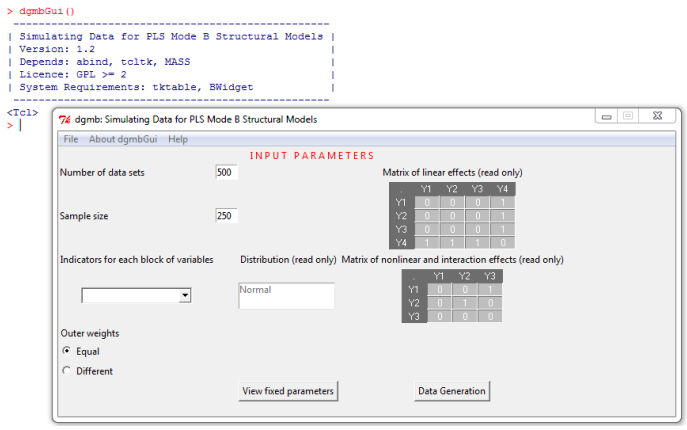

Figure 1 Graphical user interface for random data generation

Functions are implemented to generate data for a specific model (Figure 2). The true model considers three formative exogenous constructs and one formative endogenous construct. Linear effects are specified in a binary matrix as well as nonlinear and interaction effects. The interface allows one to set the number of data sets or Monte Carlo replications and the sample size of each data set; it is advisable to consider a sample size equal to or higher than 100 observations. All the measurement models have to have the same number of manifest variables. Four options can be used: 2, 4, 6, and 8 indicators per construct.

For each true model, two sets of true outer weights are available. The value of the outer effects may be equal in all measurement models or a set of different permissible values may be considered. Manifest variables are generated as random normal data.

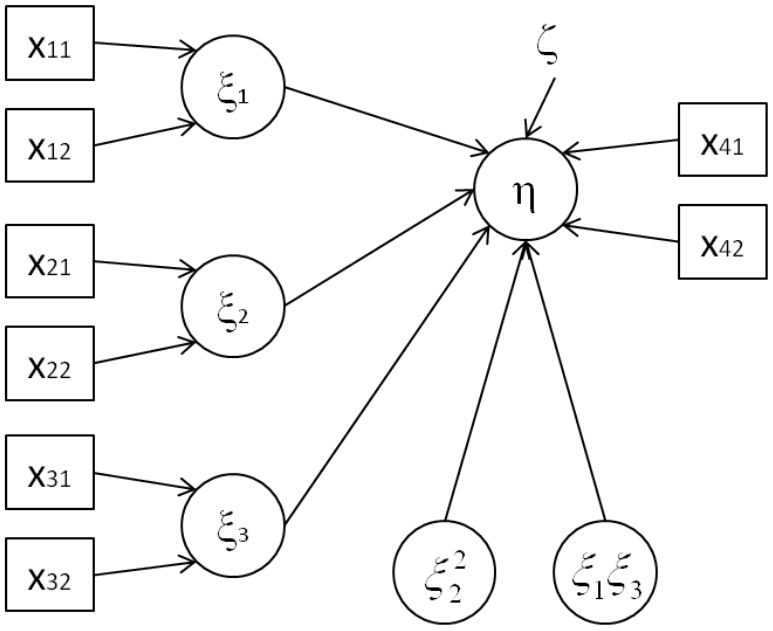

Figure 2 True structural and measurement models

Value

Two files:

`text file`	A `.txt` file in ASCII format readable from R-project with `dget`.
`csv file`	A `.csv` file.

Author(s)

Alba Martinez-Ruiz <amartine@ucsc.cl>, Claudia Martinez-Araneda <cmartinez@ucsc.cl>

References

Chin, W. W., Marcolin, B. L., and Newsted, P. R. 2003. A partial least squares latent variable modeling approach for measuring interaction effects: Results from a Monte Carlo simulation study and an electronic-mail emotion/adoption study. *Information Systems Research*, **14**(**2**), 189-217.

Dalgaard's Peter (2002) *Rnews: The Newsletter of the R Project*, Vol. 3.

Hanafi, M. 2007. PLS path modeling: Computation of latent variables with the estimation mode B. *Computational Statistics*, **22**, 275-292.

Martinez-Ruiz, A., Aluja-Banet, T. (2013) Two-step PLS path modeling mode B: Nonlinear and interaction effects between formative constructs. In *New Perspectives in Partial Least Squares and Related Methods*, eds H. Abdi, W. Chin, V. Esposito Vinzi, G. Russolillo, and L. Trinchera, Springer Proceedings in Mathematics and Statistics, volume **56**, pp. 187-199.

R Development Core Team (2011).*R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria.* ISBN 3-900051-07-0, http://www.r-project.org/

Tenenhaus, M., Esposito Vinzi, V., Chatelin, Y. M., and Lauro, C. 2005. PLS path modeling. *Computational Statistics & Data Analysis*, **48**, 159-205.

See Also

dgmb, IntPar, ViewParameters, ToProcess, <tcltk> package

Examples

```
help(dgmb)
dgmbGui()
```

EnLVs *Calculating endogenous constructs*

Description

A function to calculate endogenous constructs.

Usage

```
EnLVs(N, n, ben, bet, elv, y.ex.tot)
```

Arguments

N	The number of data sets.
n	The sample size of each data set.
ben	The number of endogenous constructs per model (default = 1).
bet	A vector with the true values of path coefficients (linear, nonlinear and interaction effects).
elv	An array with N matrices of dimension n times ben with the disturbance terms of the endogenous constructs.
y.ex.tot	An array with N matrices of dimension n times ncol(y.ex)+ncol(a.nle)+ncol(a.ie) with the scores of exogenous constructs (linear, nonlinear, and interaction effects).

Value

y.en	An array with N matrices of dimension n times ben with the scores of endogenous constructs.

Author(s)

Alba Martinez-Ruiz <amartine@ucsc.cl>

See Also

EnMVs, XexXen

Examples

```
N <- 500
n <- 250
## Not run:
yen <- EnLVs(N,n,intpar$ben,path.coef,err$elv,yexcor$y.ex.tot)
attributes(yen)
## End(Not run)
```

EnMVs *Calculating manifest variables of endogenous constructs*

Description

A function to calculate manifest variables of endogenous constructs.

Usage

```
EnMVs(N, n, ind.en, weien, y.en)
```

Arguments

N	The number of data sets.
n	The sample size of each data set.
ind.en	The total number of manifest variables of endogenous constructs.
weien	A matrix with the true values of outer weights. Each row gives the true values for a measurement model.
y.en	An array with N matrices of dimension n times ben with the scores of endogenous constructs.

Value

x.en	An array with N matrices of dimension n times ind.en with the scores of manifest variables.

Author(s)

Alba Martinez-Ruiz <amartine@ucsc.cl>

See Also

XexXen

Examples

```
N <- 500
n <- 250
## Not run:
xen <- EnMVs(N,n,intpar$ind.en,wei.en,yen$y.en)
attributes(xen)
## End(Not run)
```

ErrEnLV *Calculating disturbance terms*

Description

A function to calculate the disturbance terms of the endogenous construct.

Usage

```
ErrEnLV(N, n, ben, bet, y.ex.cor)
```

Arguments

N	The number of data sets.
n	The sample size of each data set.
ben	The number of endogenous constructs per model (default = 1).
bet	A vector with the true values of path coefficients (linear, nonlinear and interaction effects).
y.ex.cor	An array with N matrices of dimension ncol(y.ex)+ncol(a.nle)+ncol(a.ie) times ncol(y.ex)+ncol(a.nle)+ncol(a.ie) with the correlations between exogenous constructs (linear, nonlinear, and interaction effects).

Value

A list with the following components:

std.error	A vector with the standard deviation of disturbance terms of the endogenous constructs. The vector length is equal to the number of replications.
elv	An array with N matrices of dimension n times ben with the disturbance terms of the endogenous constructs.

Author(s)

Alba Martinez-Ruiz <amartine@ucsc.cl>

See Also

EnLVs, EnMVs, XexXen

Examples

```
N <- 500
n <- 250
## Not run:
err <- ErrEnLV(N,n,intpar$ben,path.coef,yexcor$y.ex.cor)
attributes(err)
## End(Not run)
```

ExoLVs *Calculating exogenous constructs*

Description

A function to calculate exogenous constructs.

Usage

```
ExoLVs(N, n, bv, bex, x.ex, weiex)
```

Arguments

N	The number of data sets.
n	The sample size of each data set.
bv	A vector with the number of manifest variables per construct.
bex	The number of exogenous constructs.
x.ex	An array of N matrices of dimension n times ind.ex with the scores of manifest variables.
weiex	A matrix with the true values of outer weights. Each row gives the true values for a measurement model.

Value

y.ex	An array with N matrices of dimension n times bex with the scores of exogenous constructs.

Author(s)

Alba Martinez-Ruiz <amartine@ucsc.cl>

See Also

NIEffects, ExoLVsCor, ErrEnLV, EnLVs, EnMVs, XexXen

Examples

```
N <- 500
n <- 250
bv <- c(2,2,2,2)
## Not run:
yex <- ExoLVs(N,n,bv,intpar$bex,xex$x.ex,wei.ex)
attributes(yex)
## End(Not run)
```

`ExoLVsCor` *Calculating correlations between exogenous constructs*

Description

A function to calculate correlations between exogenous constructs.

Usage

```
ExoLVsCor(N, n, bex, rie, y.ex, a.nle, a.ie)
```

Arguments

`N`	The number of data sets.
`n`	The sample size of each data set.
`bex`	The number of exogenous constructs.
`rie`	A binary matrix specifying nonlinear and interaction effects on the endogenous construct.
`y.ex`	An array with `N` matrices of dimension `n` times `bex` with the scores of exogenous constructs.
`a.nle`	An array with `N` matrices of dimension `n` times `ncol(rie)` with the scores of nonlinear effects.
`a.ie`	An array with `N` matrices of dimension `n` times `ncol(rie)` with the scores of interaction effects.

Value

A list with the following components:

`y.ex.tot`	An array with `N` matrices of dimension `n` times `ncol(y.ex)+ncol(a.nle)+ncol(a.ie)` with the scores of exogenous constructs (linear, nonlinear, and interaction effects).
`y.ex.cor`	An array with `N` matrices of dimension `ncol(y.ex)+ncol(a.nle)+ncol(a.ie)` times `ncol(y.ex)+ncol(a.nle)+ncol(a.ie)` with the correlations between exogenous constructs (linear, nonlinear, and interaction effects).

Author(s)

Alba Martinez-Ruiz <amartine@ucsc.cl>

See Also

ErrEnLV, EnLVs, EnMVs, XexXen

Examples

```
N <- 500
n <- 250
## Not run:
yexcor <- ExoLVsCor(N,n,intpar$bex,intpar$rie,yex$y.ex,nlie$a.nle,nlie$a.ie)
attributes(yexcor)
## End(Not run)
```

ExoMVs *Calculating exogenous manifest variables*

Description

A function to calculate manifest variables of exogenous constructs.

Usage

```
ExoMVs(N, n, ind.ex)
```

Arguments

N	The number of data sets.
n	The sample size of each data set.
ind.ex	The total number of manifest variables of exogenous constructs.

Value

x.ex	An array with N matrices of dimension n times ind.ex with the scores of manifest variables.

Author(s)

Alba Martinez-Ruiz <amartine@ucsc.cl>

See Also

IntPar, ExoLVs, NIEffects, ExoLVsCor, ErrEnLV, EnLVs, EnMVs, XexXen

Examples

```
N <- 500
n <- 250
## Not run:
xex <- ExoMVs(N,n,intpar$ind.ex)
attributes(xex)
## End(Not run)
```

IntPar *Calculating internal parameters*

Description

A function to compute internal parameters.

Usage

```
IntPar(rs, rie, modo, bv)
```

Arguments

rs	A binary matrix specifying the structural relationships between constructs.
rie	A binary matrix specifying nonlinear and interaction effects on the endogenous construct.
modo	A vector with the mode of each block of variables. "F" for formative measurement models or Mode B in PLS structural models.
bv	A vector with the number of manifest variables per construct.

Value

A list with the following components:

nat	A vector with the nature of each construct, exogenous ("ex") or endogenous ("en").
bex	The number of exogenous constructs.
ben	The number of endogenous constructs.
ind.ex	The total number of manifest variables of exogenous constructs.
ind.en	The total number of manifest variables of endogenous constructs.

Author(s)

Alba Martinez-Ruiz <amartine@ucsc.cl>

See Also

dgmb, dgmbGui, ExoMVs, ExoLVs, NIEffects, ExoLVsCor, ErrEnLV, EnLVs, EnMVs, XexXen

Examples

```
r.s <- matrix(c(0,0,0,1,
                0,0,0,1,
                0,0,0,1,
                1,1,1,0),4,4,byrow=TRUE)

r.ie <- matrix(c(0,0,1,
                 0,1,0,
                 1,0,0),3,3,byrow=TRUE)

modo <- c("F","F","F","F")

bv <- c(2,2,2,2)

intpar <- IntPar(r.s,r.ie,modo,bv)

attributes(intpar)
```

`NIEffects` *Calculating nonlinear and interaction effects*

Description

A function to calculate nonlinear and interaction effects.

Usage

```
NIEffects(N, n, y.ex, rie)
```

Arguments

`N`	The number of data sets.
`n`	The sample size of each data set.
`y.ex`	An array with `N` matrices of dimension `n` times `bex` with the scores of exogenous constructs.
`rie`	A binary matrix specifying nonlinear and interaction effects on the endogenous construct.

Value

A list with the following components:

`a.nle`	An array with `N` matrices of dimension `n` times `ncol(rie)` with the scores of nonlinear effects.
`a.ie`	An array with `N` matrices of dimension `n` times `ncol(rie)` with the scores of interaction effects.

Author(s)

Alba Martinez-Ruiz <amartine@ucsc.cl>

See Also

ExoLVsCor, ErrEnLV, EnLVs, EnMVs, XexXen

Examples

```
N <- 500
n <- 250
## Not run:
nlie <- NIEffects(N,n,yex$y.ex,intpar$rie)
attributes(nlie)
## End(Not run)
```

ToProcess *Processing input parameters*

Description

A function to process the input parameters. The function calls functions to generate random data for PLS-Mode-B models.

Usage

```
ToProcess(N, n, parMode, pI, eq)
```

Arguments

N	The number of data sets.
n	The sample size of each data set.
parMode	The mode of each block of variables (default = "F")
pI	The number of indicators per construct: {2,4,6,8}.
eq	A boolean variable. TRUE for equal outer weights; FALSE for different combinations of outer weights.

Value

Two files:

text file	A .txt file in ASCII format readable from R-project with dget.
csv file	A .csv file.

Author(s)

Claudia Martinez-Araneda <cmartinez@ucsc.cl>

References

Dalgaard's Peter (2002) *Rnews: The Newsletter of the R Project*, Vol. 3.

See Also

IntPar, ExoMVs, EnMVs, ExoLVs, EnLVs, ErrEnLV, ExoLVsCor, NIEffects, XexXen

Examples

```
ToProcess <- function(N,n,parMode,pI,eq){

require (MASS)
cat(paste("\nInputs " , N," ",n ," ",  parMode,"  ",pI, "  ",eq,"\n"))
if ( is.na(N) || is.na(n))
cat ("N and n values must be an integer greater than zero (try again)")
else
   if (is.numeric(N)==FALSE || is.numeric(n)==FALSE)
   cat ("N and n values must be an integer greater than zero (try again)")
   else
   if (length(pI)==0)
   cat ("Must select an indicador for each block variables")
   else
   {
   #Input retrieve
   #-------------------------------------------------#
   if (is.na(pI)) pI<- 2
   if (eq==1) eq<- TRUE
   else eq<-FALSE

   #Setting parameters to functions
   #-------------------------------------------------#
   r.s <- matrix(c(0,0,0,1,
                   0,0,0,1,
                   0,0,0,1,
                   1,1,1,0),4,4,byrow=TRUE)

   r.ie <- matrix(c(0,0,1,
                    0,1,0,
                    0,0,0),3,3,byrow=TRUE)

   modo <- c(parMode,parMode,parMode,parMode)
   bv <- c(pI,pI,pI,pI)
   path.coef <- c(0.5,0.4,0.3,0,0.3,0,0.3,0,0)
   #-------------------------------------------------#
   if (eq==TRUE) #equal
   {
    if (pI==2){ #Case 1: two indicators per construct, all outer weights equal
    wei.ex <- matrix (c(0.63,0.63,
                        0.63,0.63,
                        0.63,0.63), 3,2,byrow=TRUE)#cambio 07.11
    wei.en <- matrix(c(0.63,0.63),1,2,byrow=TRUE)
    }
```

```
    if (pI==4){#Case 2: four indicators per construct, all outer weights equal
    wei.ex <- matrix (c(0.42,0.42,0.42,0.42,
                        0.42,0.42,0.42,0.42,
                        0.42,0.42,0.42,0.42), 3,4,byrow=TRUE)
    wei.en <- matrix(c(0.42,0.42,0.42,0.42),1,4,byrow=TRUE)
    }
    if (pI==6){#Case 3: six indicators per construct, all outer weights equal
    wei.ex <- matrix (c(0.35,0.35,0.35,0.35,0.35,0.35,
                        0.35,0.35,0.35,0.35,0.35,0.35,
                        0.35,0.35,0.35,0.35,0.35,0.35),
                        3,6,byrow=TRUE)
    wei.en <- matrix(c(0.35,0.35,0.35,0.35,0.35,0.35),1,6,byrow=TRUE)
    }
    if (pI==8){#Case 4: eight indicators per construct, all outer weights equal
    wei.ex <- matrix (c(0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,
                        0.3,0.3,0.3,0.3,0.3,0.3, 0.3,0.3,
                        0.3,0.3,0.3,0.3,0.3,0.3, 0.3,0.3),
                        3,8,byrow=TRUE)
    wei.en <- matrix(c(0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3),1,8,byrow=TRUE)
    }

   }else{#Different

    if (pI==2) {#Case 1: two indicators per construct, all differents
    wei.ex <- matrix(c(0.8,0.4,
                       0.4,0.8,
                       0.1,0.9),3,2,byrow=TRUE)
    wei.en <- matrix(c(0.4,0.8),1,2,byrow=TRUE)
    }
    if (pI==4){#Case 2: four indicators per construct, all differents
    wei.ex <- matrix(c(0.2,0.3,0.5,0.7,
                       0.2,0.4,0.6,0.5,
                       0.3,0.5,0.7,0.2),3,4,byrow=TRUE)
    wei.en <- matrix(c(0.2,0.3,0.5,0.5),1,4,byrow=TRUE)

    }
    if (pI==6){#Case 3: six indicators per construct, all differents
     wei.ex <- matrix(c(0.5,0.3,0.4,0.3,0.5,0.1,
                        0.2,0.4,0.6,0.4,0.2,0.3,
                        0.3,0.6,0.2,0.3,0.4,0.2),
                        3,6,byrow=TRUE)
     wei.en <- matrix(c(0.5,0.3,0.4,0.3,0.5,0.1),1,6,byrow=TRUE)
     }
    if (pI==8){#Case 4: eight indicators per construct, all differents
     wei.ex <- matrix(c(0.3,0.3,0.4,0.3,0.4,0.3,0.2,0.3,
                        0.3,0.3,0.4,0.3,0.2,0.3,0.4,0.2,
                        0.4,0.5,0.4,0.3,0.2,0.1,0.3,0.2),
                        3,8,byrow=TRUE)
     wei.en <- matrix(c(0.3,0.3,0.4,0.3,0.4,0.3,0.2,0.3),1,8,byrow=TRUE)
    }
}

#Call to functions simulators
```

```
#-------------------------------------------------------------------#
intpar <- IntPar(r.s,r.ie,modo,bv)
xex <- ExoMVs(N,n,intpar$ind.ex)
yex <- ExoLVs(N,n,bv,intpar$bex,xex$x.ex,wei.ex)
nlie <- NIEffects(N,n,yex$y.ex,intpar$rie)
yexcor <- ExoLVsCor(N,n,intpar$bex,intpar$rie,yex$y.ex,nlie$a.nle,nlie$a.ie)
err <- ErrEnLV(N,n,intpar$ben,path.coef,yexcor$y.ex.cor)

if (as.numeric(err$vis) == 0)
 {
 yen <- EnLVs(N,n,intpar$ben,path.coef,err$elv,yexcor$y.ex.tot)
 xen <- EnMVs(N,n,intpar$ind.en,wei.en,yen$y.en)
 x <- XexXen(N,n,intpar$ind.ex,intpar$ind.en,xex$x.ex,xen$x.en)
 x <- x$x

#Output to text file and csv file
#-------------------------------------------------------------------#
outputfile <- tclvalue(tkgetSaveFile(initialdir="~"))
   if (outputfile != "")
   {
    dput(x, paste(outputfile,".txt"), control="all")
    write.csv(x, file = paste(outputfile,".csv"))
    cat ("Generating data...")
    msg<-paste("csv and txt data were successfully saved ")
    }else msg<-paste("The data were not successfully saved")
  tkmessageBox(title="Data Output",message=msg)
  }
 }
}#Fin ToProcess
```

`ViewHelp` *Viewing help files in pdf format*

Description

A function to view the help files.

Usage

```
ViewHelp()
```

Value

A file:

`pdf file` A help file including the description of the main functions of the dgmb-package.

Author(s)

Claudia Martinez-Araneda <cmartinez@ucsc.cl>

References

Dalgaard's Peter (2002) *Rnews: The Newsletter of the R Project*, Vol. 3.

See Also

<tcltk> package

Examples

```
ViewHelp <- function()
{
helpfile<-paste(find.package("dgmb",verbose = verbose))
helpfile<-substring(helpfile,1,nchar(helpfile))
helpfile<-paste(helpfile,"/docs/dgmb-manual.pdf")
helpfile<-sub("dgmb ","dgmb", helpfile)
cat ("Viewing dgmb help...otherwise, you should write help(dgmb)
  in R console\n")
browseURL(helpfile)
}
```

ViewParameters *A dialog box to display parameters*

Description

A function to display a dialog box with simulation parameters.

Usage

```
ViewParameters(tkl, tkr)
```

Arguments

tkl	An instance of tkcombobox
tkr	An instance of tkradiobutton

Author(s)

Claudia Martinez-Araneda <cmartinez@ucsc.cl>

References

Dalgaard's Peter (2002) *Rnews: The Newsletter of the R Project*, Vol. 3.

Examples

```
ViewParameters <- function(tkl,tkr){

 rb<- tclvalue(tkr)
 ll<- as.numeric(tclvalue(tcl(tkl,"getvalue")))
 msg<-paste("")

 if  (rb == 1) #all outer weights equal
 {
 if (as.integer(ll)==0)
  {
         msg<-paste(msg,"path.coef <- c(0.5,0.4,0.3,0,0.3,0,0.3,0,0)\n")
         msg<-paste(msg,"wei.ex <- matrix (c(0.63,0.63,\n\t\t
                                0.63,0.63,\n\t\t
                                0.63,0.63),3,2,byrow=TRUE)\n")
        msg<-paste(msg,"wei.en <- matrix(c(0.63,0.63),1,2,byrow=TRUE),3,2,byrow=TRUE)")
  }
 if (as.integer(ll)==1)
  {
         msg<-paste(msg,"path. coef <- c(0.5,0.4,0.3,0,0.3,0,0.3,0,0)\n")
         msg<-paste(msg,"wei.ex <- matrix (c(0.42,0.42,0.42,0.42,\n\t\t
                                0.42,0.42,0.42,0.42,\n\t\t
                                0.42,0.42,0.42,0.42),3,4,byrow=TRUE)\n")
         msg<-paste(msg,"wei.en <- matrix(c(0.42,0.42,0.42,0.42),1,4,byrow=TRUE)\n")
  }
 if (as.integer(ll)==2)
  {
         msg<-paste(msg,"path. coef <- c(0.5,0.4,0.3,0,0.3,0,0.3,0,0)\n")
         msg<-paste(msg,"wei.ex <- matrix (c(0.35,0.35,0.35,0.35,0.35,0.35,\n\t\t
                                0.35,0.35,0.35,0.35,0.35,0.35,\n\t\t
                                0.35,0.35,0.35,0.35,0.35,0.35),
                                3,6,byrow=TRUE),3,2,byrow=TRUE)\n")
         msg<-paste(msg,"wei.en <- matrix(c(0.35,0.35,0.35,0.35,0.35,0.35),
                                1,6,byrow=TRUE)")
  }
 if (as.integer(ll)==3)
  {
         msg<-paste(msg,"path. coef <- c(0.5,0.4,0.3,0,0.3,0,0.3,0,0)\n")
         msg<-paste(msg,"wei.ex <- matrix (c(0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,\n\t\t
                                0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3,\n\t\t
                                0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3),
                                3,8,byrow=TRUE)\n")
         msg<-paste(msg,"wei.en <- matrix(c(0.3,0.3,0.3,0.3,0.3,0.3,0.3,0.3),
                                1,8,byrow=TRUE)")
  }
 }else{#all outer weights different
    if (as.integer(ll)==0)
    {
         msg<-paste(msg,"path.coef <- c(0.5,0.4,0.3,0,0.3,0,0.3,0,0)\n")
         msg<-paste(msg,"wei.ex <- matrix(c(0.8,0.4,\n\t\t
                                0.4,0.8,\n\t\t0.1,0.9),3,2,byrow=TRUE)\n")
         msg<-paste(msg,"wei.en <- matrix(c(0.4,0.8),1,2,byrow=TRUE)")
```

```
  }
  if (as.integer(ll)==1)
  {
        msg<-paste(msg,"path. coef <- c(0.5,0.4,0.3,0,0.3,0,0.3,0,0)\n")
        msg<-paste(msg,"wei.ex <- matrix(c(0.2,0.3,0.5,0.7\n\t\t
                                0.2,0.4,0.6,0.5\n\t\t
                                0.3,0.5,0.7,0.2),3,4,byrow=TRUE)\n")
        msg<-paste(msg,"wei.en <- matrix(c(0.2,0.3,0.5,0.5),1,4,byrow=TRUE)\n")
   }
  if (as.integer(ll)==2)
   {
        msg<-paste(msg,"path. coef <- c(0.5,0.4,0.3,0,0.3,0,0.3,0,0)\n")
        msg<-paste(msg,"wei.ex <- matrix(c(0.5,0.3,0.4,0.3,0.5,0.1,\n\t\t
                                0.2,0.4,0.6,0.4,0.2,0.3,\n\t\t
                                0.3,0.6,0.2,0.3,0.4,0.2),3,6,byrow=TRUE)\n")
        msg<-paste(msg,"wei.en <- matrix(c(0.5,0.3,0.4,0.3,0.5,0.1),1,6,byrow=TRUE)")
   }
if (as.integer(ll)==3)  {

        msg<-paste(msg,"path. coef <- c(0.5,0.4,0.3,0,0.3,0,0.3,0,0)\n")
        msg<-paste(msg,"wei.ex <- matrix(c(0.3,0.3,0.4,0.3,0.4,0.3,0.2,0.3,\n\t\t
                                0.3,0.3,0.4,0.3,0.2,0.3,0.4,0.2,\n\t\t
                                0.4,0.5,0.4,0.3,0.2,0.1,0.3,0.2),
                                3,8,byrow=TRUE)\n")
        msg<-paste(msg,"wei.en <- matrix(c(0.3,0.3,0.4,0.3,0.4,0.3,0.2,0.3),
                                1,8,byrow=TRUE)")
  }
}
tkmessageBox(title="Fixed Parameters", message=paste(msg, "\n Distribution = Normal"),
             icon="info", type="ok")

}#End ViewParameters
```

XexXen *Simulated data*

Description

A function to create an array with simulated data.

Usage

```
XexXen(N, n, ind.ex, ind.en, x.ex, x.en)
```

Arguments

N	The number of data sets.
n	The sample size of each data set.
ind.ex	The total number of manifest variables of exogenous constructs.

ind.en	The total number of manifest variables of endogenous constructs.
x.ex	An array with N matrices of dimension n times ind.ex with the scores of manifest variables.
x.en	An array with N matrices of dimension n times ind.en with the scores of manifest variables.

Value

x	An array with N matrices of dimension n times ind.ex + ind.en with the scores of manifest variables.

Author(s)

Alba Martinez-Ruiz <amartine@ucsc.cl>

Examples

```
N <- 500
n <- 250
## Not run:
x <- XexXen(N,n,intpar$ind.ex,intpar$ind.en,xex$x.ex,xen$x.en)
attributes(x)
x <- x$x
## End(Not run)
```

Index