Package 'NlsyLinks'

January 13, 2013

Type Package

Title Utilities and kinship information for Behavior Genetics and Developmental research using the NLSY.

Version 1.200

Date 2012-12-18

Author Will Beasley, Joe Rodgers, David Bard, Michael Hunter, and Kelly Meredith

Maintainer Will Beasley <wibeasley@hotmail.com>

Imports methods,lavaan

Suggests ggplot2,plyr,testthat

Description Utilities and kinship information for Behavioral Genetics and Developmental research using the NLSY.

License GPL

URL https://r-forge.r-project.org/projects/nlsylinks/

LazyLoad yes

LazyData yes

Collate
'AceEstimate.R''AceLavaanGroup.R''AceUnivariate.R''CleanSemAceDataset.R''CreateAceEstimate.R''CreatePairL

R topics documented:

`NlsyLinks-package` *Utilities and kinship information for Behavior Genetics and Developmental research using the NLSY.*

Description

Utilities and kinship information for Behavior Genetics and Developmental research using the NLSY.

Details

Package:	NlsyLinks
Type:	Package
Version:	1.200
Date:	2012-12-18
License:	GPL
LazyLoad:	yes

An overview of how to use the package, including the most important functions

Note

This package considers both Gen1 and Gen2 subjects. "Gen1" refers to subjects in the original NLSY79 sample (http://www.bls.gov/nls/nlsy79.htm). "Gen2" subjects are the biological children of the Gen1 females -ie, those in the NLSY79 Children and Young Adults sample (http://www.bls.gov/nls/nlsy79ch.htm).

Author(s)

Will Beasley, Joe Rodgers, Kelly Meredith, and David Bard

Maintainer: Will Beasley <wibeasley@hotmail.com>

References

This package's development was largely supported by the NIH Grant 1R01HD65865, "NLSY Kinship Links: Reliable and Valid Sibling Identification" (PI: Joe Rodgers)

Rodgers, Joseph Lee, & Kohler, Hans-Peter (2005). Reformulating and simplifying the DF analysis model. *Behavior Genetics, 35* (2), 211-217.

Rodgers, J.L., Bard, D., Johnson, A., D'Onofrio, B., & Miller, W.B. (2008). The Cross-Generational Mother-Daughter-Aunt-Niece Design: Establishing Validity of the MDAN Design with NLSY Fertility Variables. *Behavior Genetics, 38*, 567-578.

D'Onofrio, B.M., Van Hulle, C.A., Waldman, I.D., Rodgers, J.L., Rathouz, P.J., & Lahey, B.B. (2007). Causal inferences regarding prenatal alcohol exposure and childhood externalizing problems. *Archives of General Psychiatry, 64*, 1296-1304.

Rodgers, J.L. & Doughty, D. (2000). Genetic and environmental influences on fertility expectations and outcomes using NLSY kinship data. In J.L. Rodgers, D. Rowe, & W.B. Miller (Eds.) *Genetic influences on fertility and sexuality.* Boston: Kluwer Academic Press.

Cleveland, H.H., Wiebe, R.P., van den Oord, E.J.C.G., & Rowe, D.C. (2000). Behavior problems among children from different family structures: The influence of genetic self-selection. *Child Development, 71*, 733-751.

Rodgers, J.L., Rowe, D.C., & Buster, M. (1999). Nature, nurture, and first sexual intercourse in the USA: Fitting behavioural genetic models to NLSY kinship data. *Journal of Biosocial Sciences, 31*.

Rodgers, J.L., Rowe, D.C., & Li, C. (1994). Beyond nature versus nurture: DF analysis of nonshared influences on problem behaviors. *Developmental Psychology, 30*, 374-384.

Examples

```
library(NlsyLinks) #Load the package into the current R session.
data(Links79Pair)  #Load the dataset from the NlsyLinks package.
summary(Links79Pair)  #Summarize the five variables.
hist(Links79Pair$R)  #Display a histogram of the Relatedness values.
table(Links79Pair$R)  #Create a table of the Relatedness values for the whole sample.
```

Ace *Estimates the heritability of additive traits using a single variable.*

Description

An ACE model is the foundation of most Behavior Genetic research. It estimates the additive heritability (with *a*), common environment (with *c*) and unshared heritability/environment (with *e*).

Usage

```
AceUnivariate(method=c("DeFriesFulkerMethod1","DeFriesFulkerMethod3"), dataSet, oName_1, oName_2
  rName="R", manifestScale="Continuous")

DeFriesFulkerMethod1(dataSet, oName_1, oName_2, rName="R")

DeFriesFulkerMethod3(dataSet, oName_1, oName_2, rName="R")
```

Arguments

`method`	The specific estimation technique.
`dataSet`	The `data.frame` that contains the two outcome variables and the relatedness coefficient (corresponding to `oName_1`, `oName_2`, and `rName`).

oName_1	The name of the outcome variable corresponding to the first subject in the pair. This should be a character value.
oName_2	The name of theoutcome variable corresponding to the second subject in the pair. This should be a character value.
rName	The name of the relatedness coefficient for the pair (this is typically abbreviated as R). This should be a character value.
manifestScale	Currently, only continuous manifest/outcome variables are supported.

Details

The AceUnivariate function is a wrapper that calls DeFriesFulkerMethod1 or DeFriesFulkerMethod3. Future versions will incorporate methods that use latent variable models.

Value

Currently, a list is returned with the arguments HSquared, CSquared, ESquared, and RowCount. In the future, this may be changed to an S4 class.

Author(s)

Will Beasley

References

Rodgers, Joseph Lee, & Kohler, Hans-Peter (2005). Reformulating and simplifying the DF analysis model. *Behavior Genetics, 35* (2), 211-217.

Examples

```
library(NlsyLinks) #Load the package into the current R session.
dsOutcomes <- ExtraOutcomes79
dsOutcomes$SubjectTag <- CreateSubjectTag(subjectID=dsOutcomes$SubjectID,
  generation=dsOutcomes$Generation)
dsLinks <- Links79Pair
dsLinks <- dsLinks[dsLinks$RelationshipPath=='Gen2Siblings', ] #Use only the Gen2 Siblings (ie, NLSY79-C su
dsDF <- CreatePairLinksDoubleEntered(outcomeDataset=dsOutcomes, linksPairDataset=dsLinks,
  outcomeNames=c("MathStandardized", "HeightZGenderAge", "WeightZGenderAge"))

estimatedAdultHeight <- DeFriesFulkerMethod3(
  dataSet=dsDF,
  oName_1="HeightZGenderAge_1",
  oName_2="HeightZGenderAge_2")
estimatedAdultHeight #ASquared and CSquared should be 0.606 and 0.105 for this rough analysis.

estimatedMath <- DeFriesFulkerMethod3(
  dataSet=dsDF,
  oName_1="MathStandardized_1",
  oName_2="MathStandardized_2")
estimatedMath #ASquared and CSquared should be 0.878 and 0.048.

class(GetDetails(estimatedMath))
summary(GetDetails(estimatedMath))
```

AceEstimate-class *Class* "AceEstimate"

Description

A class containing information about a single univariate ACE model.

Objects from the Class

Objects can be created by calls of the form:

```
new("AceEstimate", aSquared, cSquared, eSquared, caseCount, unity, withinBounds, details, ...)
```

Slots

ASquared: The proportion of variability due to a shared genetic influence (typically represented as a^2, or sometimes h^2).

CSquared: The proportion of variability due to shared common environmental influence.

ESquared: The proportion of variability due to unexplained/residual/error influence.

CaseCount: The number cases contributing to this estimate.

Unity: Indicates if the three components sum to 1.

WithinBounds: Indicates if the three components are each bounded by [0, 1].

Details: A `list` that contains the modeling and estimation output and details.

Methods

GetDetails `signature(object = "AceEstimate")`: ...

initialize `signature(.Object = "AceEstimate")`: ...

print `signature(x = "AceEstimate")`: ...

show `signature(object = "AceEstimate")`: ...

Note

The contents of the `Details` list depends on the underlying estimation routine. For example, when the ACE model is estimated with a DF analysis, the output is an `lm` object, because the `lm` function was used (ie, the basical general linear model). Alternatively, if the user specified the `lavaan` package should estimate that ACE model, the output is a `lavaan` object.

Examples

```
library(NlsyLinks) #Load the package into the current R session.

showClass("AceEstimate")
est <- CreateAceEstimate(.5, .2, .3, 40)
est
print(est)
```

AceLavaanGroup *A simple multiple-group ACE model with the* **lavaan** *package.*

Description

This function uses the **lavaan** package to estimate a univariate ACE model, using multiple groups. Each group has a unique value of R (i.e., the *R*elatedness coefficient).

Usage

```
AceLavaanGroup(dsClean, estimateA=TRUE, estimateC=TRUE,
  printOutput=FALSE)
```

Arguments

dsClean	The data.frame containing complete cases for the R groups to be included in the estimation.
estimateA	Should the *A* variance component be estimated? A^2 represents the proportion of variability due to a shared genetic influence.
estimateC	Should the *C* variance component be estimated? C^2 represents the proportion of variability due to a shared environmental influence.
printOutput	Indicates if the estimated parameters and fit statistics are printed to the console.

Details

The variance component for *E* is always estimated.

Value

An AceEstimate object.

Note

Currently, the variables in dsClean must be named M1, M2 and R. This may not be as restrictive as it initially seems, because dsClean is intented to be produced by CleanSemAceDataset. If this is too restrictive for your uses, we'd like to here about it (*please email wibeasley at hotmail period com*).

Author(s)

Will Beasley

References

The **lavaan** package is developed by Yves Rosseel at Ghent University. Two good starting points are the documentation (http://cran.r-project.org/web/packages/lavaan/) and his upcoming JSS paper (http://users.ugent.be/~yrosseel/lavaan/lavaanJSSpreview.pdf).

See Also

Further ACE model details are discussed in our package's vignette (in the console, type vignette("NlsyAce")).

Examples

```
library(NlsyLinks) #Load the package into the current R session.
dsLinks <- Links79PairExpanded #Start with the built-in data.frame in NlsyLinks
dsLinks <- dsLinks[dsLinks$RelationshipPath=='Gen2Siblings', ] #Use only the Gen2 Siblings (ie, NLSY79-C su

oName_1 <- "MathStandardized_1" #Stands for Outcome1
oName_2 <- "MathStandardized_2" #Stands for Outcome2

dsGroupSummary <- RGroupSummary(dsLinks, oName_1, oName_2)
dsClean <- CleanSemAceDataset(dsDirty=dsLinks, dsGroupSummary, oName_1, oName_2)

ace <- AceLavaanGroup(dsClean)
ace

#Should produce:
# [1] "Results of ACE estimation: [show]"
#     ASquared     CSquared     ESquared    CaseCount
#    0.6681874    0.1181227    0.2136900 8390.0000000

library(lavaan) #Load the package to access methods of the lavaan class.
GetDetails(ace)

#Exmaine fit stats like Chi-Squared, RMSEA, CFI, etc.
fitMeasures(GetDetails(ace)) #The function 'fitMeasures' is defined in the lavaan package.

#Examine low-level details like each group's individual parameter estimates and standard errors.
summary(GetDetails(ace))

#Extract low-level details. This may be useful when programming simulations.
inspect(GetDetails(ace), what="converged") #The lavaan package defines 'inspect'.
inspect(GetDetails(ace), what="coef")
```

`CleanSemAceDataset` *Produces a cleaned dataset that works well with when using SEM to estimate a univariate ACE model.*

Description

This function takes a 'GroupSummary' `data.frame` (which is created by the `RGroupSummary` function) and returns a `data.frame` that is used by the `Ace` function.

Usage

```
CleanSemAceDataset(dsDirty, dsGroupSummary, oName_1,
  oName_2, rName = "R")
```

Arguments

`dsDirty`	This is the `data.frame` to be cleaned.
`dsGroupSummary`	The `data.frame` containing information about which groups should be included in the analyses. It should be created by the `RGroupSummary` function.
`oName_1`	The name of the manifest variable (in `dsDirty`) for the first subject in each pair.

oName_2	The name of the manifest variable (in dsDirty) for the second subject in each pair.
rName	The name of the variable (in dsDirty) indicating the pair's relatedness coefficient.

Details

The function takes dsDirty and produces a new data.frame with the following features:

[A] Only three existing columns are retained: O1, O2, and R. They are assigned these names.

[B] A new column called GroupID is created to reflect their group membership (which is based on the R value). These valuesa re sequential integers, starting at 1. The group with the weakest R is 1. The group with the strongest R has the largest GroupID (this is typically the MZ tiwns).

[C] Any row is excluded if it has a missing data point for O1, O2, or R.

[D] The data.frame is sorted by the R value. This helps program against the multiple-group SEM API sometimes.

Value

A data.frame with one row per subject pair. The data.frame contains the following variables (which can NOT be changed by the user through optional parameters):

R	The pair's R value.
O1	The outcome variable for the first subject in each pair.
O2	The outcome variable for the second subject in each pair.
GroupID	Indicates the pair's group membership.

Author(s)

Will Beasley

Examples

```
library(NlsyLinks) #Load the package into the current R session.
dsLinks <- Links79PairExpanded #Start with the built-in data.frame in NlsyLinks
dsLinks <- dsLinks[dsLinks$RelationshipPath=='Gen2Siblings', ] #Use only NLSY79-C siblings

oName_1 <- "MathStandardized_1" #Stands for Outcome1
oName_2 <- "MathStandardized_2" #Stands for Outcome2
dsGroupSummary <- RGroupSummary(dsLinks, oName_1, oName_2)

dsClean <- CleanSemAceDataset( dsDirty=dsLinks, dsGroupSummary, oName_1, oName_2, rName="R" )
summary(dsClean)

dsClean$AbsDifference <- abs(dsClean$O1 - dsClean$O2)
plot(jitter(dsClean$R), dsClean$AbsDifference, col="gray70")
```

ColumnUtilities *A collection of functions that helps data management* data.frame*s, particularly those derived from NLS Extracts.*

Description

A collection of functions that helps data management data.frames, particularly those derived from NLS Extracts.

Usage

```
VerifyColumnExists( dataFrame, columnName )

RenameColumn( dataFrame, oldColumnName, newColumnName )

RenameNlsyColumn( dataFrame, nlsyRNumber, newColumnName )
```

Arguments

dataFrame	A data.frame object whose columns are to be verified or renamed.
columnName	The name of the column to verify is present in the data.frame.
nlsyRNumber	The name of the column to change.
oldColumnName	The name of the column to change.
newColumnName	The desired name of the column.

Details

The RNumber assigned by the NLS has a pattern. In the Nlsy79 Gen1 dataset, the names start with a 'R' or 'T' and are followed by seven digits (eg, R0000100). In the Nlsy79 Gen2 dataset, the names start with 'C' or 'Y' and are followed by seven digits (eg, C0007030, Y1994600).

In the NLS Investigator, a decimal is present in the RNumber (eg, R00001.00). When the Investigator saves the dataset as a CSV, the decimal is removed (R0000100).

Value

IMPORTANT The RenameColumn and RenameNlsyColumn functions do not use side-effects to rename the data.frame. Instead, it returns a new data.frame. In the example below, notice the assignment to ds: `ds <- RenameNlsyColumn(...)`.

The VerifyColumnExists function check that exactly one column exists with the specified columnName. If so, the index of the column is returned. If not, an exception is thrown.

Author(s)

Will Beasley

Examples

```
filePathGen2 <- file.path(path.package("NlsyLinks"), "extdata", "Gen2Birth.csv") #"F:/Projects/RDev/NlsyLi
ds <- read.csv(filePathGen2)

originalColumnNames <- c("C0000100", "C0000200", "C0005300", "C0005400",
  "C0005700", "C0328000", "C0328600", "C0328800")

newColumnNames <- c("SubjectID", "MotherID", "Race", "Gender", "Yob",
  "GestationWeeks", "BirthWeightInOunces", "BirthLengthInInches")

for( columnIndex in seq_along(originalColumNames) ) {
  ds <- RenameNlsyColumn(ds, originalColumNames[columnIndex], newColumnNames[columnIndex])
}
```

`CreateAceEstimate` *Instantiate an* `AceEstimate-class` *object.*

Description

Creates an instance of the S4 class, `AceEstimate` instantiating arguments set the parameter values estimated by the ACE model.

Usage

```
CreateAceEstimate(aSquared, cSquared, eSquared, caseCount,
  details=list(), unityTolerance = 1e-11)
```

Arguments

`aSquared`	The proportion of variability due to a shared genetic influence (typically represented as a^2, or sometimes h^2).
`cSquared`	The proportion of variability due to shared common environmental influence.
`eSquared`	The proportion of variability due to unexplained/residual/error influence.
`caseCount`	The number of cases used to estimate the model.
`unityTolerance`	Specifies how close the the sum of the ACE components should be to one, to be considered properly scaled to one.
`details`	A `list` that contains the modeling output and details.

Details

The contents of the `details` list depends on the underlying estimation routine. For example, when the ACE model is estimated with a DF analysis, the output is an `lm` object, because the `lm` function was used (ie, the basical general linear model). Alternatively, if the user specified the `lavaan` package should estimate that ACE model, the output is a `lavaan` object.

Value

An S4 object of `AceEstimate-class`.

Author(s)

Will Beasley

`CreatePairLinks` *Creates a pairs linking file.*

Description

Creates a linking file for BG designs using this file structure (e.g., DF analysis, other ACE modeling).

Usage

```
CreatePairLinksDoubleEntered(outcomeDataset, linksPairDataset, outcomeNames,
  linksNames = c("ExtendedID", "R", "RelationshipPath"), validateOutcomeDataset=TRUE)

CreatePairLinksSingleEntered(outcomeDataset, linksPairDataset, outcomeNames,
  linksNames = c("ExtendedID", "R", "RelationshipPath"), validateOutcomeDataset=TRUE)

CreatePairLinksDoubleEnteredWithNoOutcomes(linksPairDataset,
  linksNames = c("ExtendedID", "R", "RelationshipPath"))
```

Arguments

`outcomeDataset`	A data frame containing the outcome variable(s)
`linksPairDataset`	A data frame containing the `SubjectTags` of each subject in the pair and their `R` coefficient.
`outcomeNames`	The column names of the outcome variable(s)
`linksNames`	The column names desired to be prseent in the newly created data frame. `Subject1Tag` and `Subject2Tag` are included automatically.
`validateOutcomeDataset`	Indicates if characteristics of the outcomeDataset should be validated.

Details

A DF analysis requires a double-entered file that contains the `R` value for the pair, and their two outcome variable values.

`CreatePairLinksDoubleEnteredWithNoOutcomes` is intended to be a primarily a helper function for `CreateSpatialNeighbours`.

Author(s)

Will Beasley

References

For more information about a DF analysis, see: Rodgers, Joseph Lee, & Kohler, Hans-Peter (2005). Reformulating and simplifying the DF analysis model. *Behavior Genetics, 35* (2), 211-217.

Examples

```
dsSingleLinks <- data.frame(
  ExtendedID=c(1, 1, 1, 2),
  Subject1Tag=c(101, 101, 102, 201),
  Subject2Tag=c(102, 103, 103, 202),
  R=c(.5, .25, .25, .5),
  RelationshipPath=rep("Gen2Siblings", 4)
)
dsSingleOutcomes <- data.frame(
  SubjectTag=c(101, 102, 103, 201, 202),
  DV1=c(11, 12, 13, 41, 42),
  DV2=c(21, 22, 23, 51, 52))
dsDouble <- CreatePairLinksDoubleEntered(
  outcomeDataset=dsSingleOutcomes,
  linksPairDataset=dsSingleLinks,
  outcomeNames=c("DV1", "DV2"),
  validateOutcomeDataset=TRUE)
dsDouble #Show the 8 rows in the double-entered pair links
summary(dsDouble) #Summarize the variables

ValidatePairLinksAreSymmetric(dsDouble) #Should return TRUE.
```

`CreateSpatialNeighbours` *Distances between related family members, formated for spatial analysis.*

Description

This helper function formats the LinksPair datasets so it can be used in some types of spatial analyses. The **spdep** (Spatial Dependence) uses a sparse matrix (actually a `data.frame`) to represent neigbours.

Usage

```
CreateSpatialNeighbours(linksPairsDoubleEntered)
CreateSpatialNeighbours79Gen2()
```

Arguments

`linksPairsDoubleEntered`
: ## Need something here. ##

Details

There is one row per unique pair of subjects, *respecting order*. This is different than `Links79Pair` and `Links79PairExpanded`, which has one row per unique pair of subjects, *irrespective of order*.

`CreateSpatialNeighbours79Gen2` automatically creates dataset of NLSY79 Gen2 subjects (i.e., the children of the mothers in the initial NLSY79 sample.

`CreateSpatialNeighbours` accepts any paired relationships in a `data.frame`, as long as it contains the columns `Subject1Tag`, `Subject2Tag`, and `R`. See `Links79Pair` for more details.

Value

Returns an S3 spatial.neighbours object to work with functions in the **spdep** package. Subject1Tag is renamed from. Subject2Tag is renamed to. R is renamed weight. The attribute region.id specifies each unique SubjectTag. The attribue n specifies the number of unique subjects.

Note

Notice the British variant of 'neighbo*u*rs'

Author(s)

Will Beasley, David Bard

References

Bivand, R., Pebesma, E., & Gomez-Rubio, V. (2008). *Applied Spatial Data Analysis with R.* New York: Springer. (Especially Chapter 9.)

##David, can you please give me some of the articles/books that you used this with?

See Also

Links79Pair

listw2sn, sn2listw, df2sn, write.sn2gwt, write.sn2dat are some of the functions in the **spdep** package that use a spatial.neighbours object.

Examples

```
#Need something here
```

CreateSubjectTag — *Creates a* SubjectTag. *This value uniquely identifies subjects, when both generations are included in the same dataset.*

Description

A SubjectTag uniquely identify subjects. For Gen2 subjects, the SubjectTag is identical to their CID (ie, C00001.00 -the SubjectID assigned in the NLSY79-Children files). However for Gen1 subjects, the SubjectTag is their CaseID (ie, R00001.00), with "00" appended. This manipulation is necessary to identify subjects uniquely in inter-generational datasets. A Gen1 subject with an ID of 43 becomes 4300. The SubjectTags of her four children remain 4301, 4302, 4303, and 4304.

Usage

```
CreateSubjectTag(subjectID, generation)
```

Arguments

subjectID	The ID assigned by the NLSY. For Gen1 subjects, this will be their CaseID (ie, R00001.00). For Gen2 subjects, this will be their CID (ie, C00001.00).
generation	The generation of the subject. Values are either 1 or 2, representing Gen1 and Gen2. Note that this variable is not a factor (in constrast with data frames like Links79Pair). This dataset is supposed to mimick the dataset provided by the researcher, which typically will not have been converted to a factor.

Details

For a fuller explanation of SubjectTag in context, see the Links79Pair dataset documentation.

Value

A integer value under normal circumstances.

An error is thrown if the vectors subjectID and generation are different lengths.

If either input vector has NA values, the respective output element(s) will be NA too.

Author(s)

Will Beasley

See Also

Links79Pair

Examples

```
library(NlsyLinks) #Load the package into the current R session.
data(ExtraOutcomes79)  #Load the dataset from the NlsyLinks package.

#Typically these two vectors will come from a data frame.
subjectIDs <- c(71:82, 10001:10012)
generation <- c(rep(1, 12), rep(2, 12))

CreateSubjectTag(subjectIDs, generation)
#Returns 7100, ..., 8200, 10001, ..., 10012

#Use the ExtraOutcomes79 dataset, with numeric variables 'SubjectID' and 'Generation'.
ExtraOutcomes79$SubjectTag <- CreateSubjectTag(
  subjectID=ExtraOutcomes79$SubjectID,
  generation=ExtraOutcomes79$Generation
)
```

ExtraOutcomes79 *Extra outcome variables in the NLSY79*

Description

This dataset is provided primarily to facilitate documentation examples.

Currently this dataset contains only Gen2 subjects. However, it soon will include Gen1 subjects. Code written using this dataset should NOT assume it contains only Gen2 subjects. See below for an example of filtering by generation.

Usage

```
data(ExtraOutcomes79)
```

Format

A data frame with 11,495 observations on the following 6 variables. There is one row per subject.

`SubjectTag` The ID value assigned by NLS to the first subject. For Gen1 Subjects, this is their "CaseID" (ie, R00001.00). For Gen2 subjects, this is their "CID" (ie, C00001.00).

`SubjectID` The ID value assigned by NLS to the first subject. For Gen1 Subjects, this is their "CaseID" (ie, R00001.00). For Gen2 subjects, this is their "CID" (ie, C00001.00).

`Generation` The generation of the subject. Values are either 1 or 2, representing Gen1 and Gen2. Note that this variable is not a `factor` (in constrast with data frames like `Links79Pair`). This dataset is supposed to mimick the dataset provided by the researcher, which typically will not have been converted to a `factor`.

`AfqtRescaled2006Gaussified` Armed Forces Qualification Test Score for Gen1.

`HeightZGender` The subject's height, standardized by gender (see Details).

`HeightZGenderAge` The subject's height, standardized by gender and age (see Details).

`WeightZGender` The subject's weight, standardized by gender (see Details).

`WeightZGenderAge` The subject's weight, standardized by gender and age (see Details).

`Afi` Self-reported age of first intercourse (see Details).

`Afm` Self-reported age of first menstration (see Details).

`MathStandardized` Standardized PIAT Math score for Gen2 (see Details).

Author(s)

Will Beasley

Source

Gen1 information comes from the May 15, 2010 release of the NLSY79 sample. Gen2 information comes from the Sept 15, 2010 release of the NLSY79 Children and Young Adults sample. Data were extracted with the NLS Investigator (`https://www.nlsinfo.org/investigator/`).

The `SubjectTag` variable uniquely identify subjects. For Gen2 subjects, the SubjectTag is identical to their CID (ie, C00001.00 -the SubjectID assigned in the NLSY79-Children files). However for Gen1 subjects, the SubjectTag is their CaseID (ie, R00001.00), with "00" appended. This manipulation is necessary to identify subjects uniquely in inter-generational datasets. A Gen1 subject with

an ID of 43 has a SubjectTag of 4300. The SubjectTags of her four children remain 4301, 4302, 4303, and 4304.

For Gen2, an NLSY79 variable of MathStandardized is C0580100.00.

Afi, Afm, and MathStandardized values were simplied (to one value per subject) by Kelly Meredith in January 2012 using the Sept 15, 2010 release of NLSY data.

Examples

```
library(NlsyLinks) #Load the package into the current R session.
data(ExtraOutcomes79)  #Load the dataset from the NlsyLinks package.
gen2Outcomes <- subset(ExtraOutcomes79, Generation==2) #Create a dataset of only Gen2 subjects.

plot(ExtraOutcomes79)
summary(ExtraOutcomes79)

oldPar <- par(mfrow=c(3,2))
hist(ExtraOutcomes79$Generation)
hist(ExtraOutcomes79$MathStandardized)
hist(ExtraOutcomes79$HeightZGenderAge)
hist(ExtraOutcomes79$WeightZGenderAge)
hist(ExtraOutcomes79$Afi)
hist(ExtraOutcomes79$Afm)
par(oldPar)
```

GetDetails-methods *A generic function for extracting the* Details *slot of an object.*

Description

A generic function for extracting the Details slot of an AceEstimation object.

Methods

Here's a filler description.

Extracts the Details slot of an AceEstimation object.

Author(s)

signature(object = "AceEstimate") Will Beasley

GetEstimate-methods *Generic function for returning the contents from an* AceEstimate *class.*

Description

Extract the values

Methods

```
signature(object = "AceEstimate")
```

Author(s)

Will Beasley

Links79Pair *Kinship linking file for pairs of relatives in the NLSY79 and NLSY79 Children and Young Adults*

Description

This dataset specifies the relatedness coefficient (ie, 'R') between subjects in the same extended family. Each row represents a unique relationship pair. An extended family with k subjects will have $k(k\text{-}1)/2$ rows. Typically, Subject1 is older while Subject2 is younger.

The dataset contains Gen1 and Gen2 subjects. "Gen1" refers to subjects in the original NLSY79 sample (http://www.bls.gov/nls/nlsy79.htm). "Gen2" subjects are the biological children of the Gen1 females -ie, those in the NLSY79 Children and Young Adults sample (http://www.bls.gov/nls/nlsy79ch.htm).

Usage

```
data(Links79Pair)
```

Format

A data frame with 11,075 observations on the following 5 variables. There is one row per unique pair of subjects, irrespective of order.

`ExtendedID` Identity of the extended family of the pair; it corresponds to the HHID in the NLSY79. See References below.

`Subject1Tag` Identity of the pair's first subject. See Details below.

`Subject2Tag` Identity of the pair's second subject. See Details below.

`R` The pair's Relatedness coefficient. See Details below.

`RelationshipPath` Specifies the relationship category of the pair. This variable is a factor, with levels `Gen1Housemates=1`, `Gen2Siblings=2`, `Gen2Cousins=3`, `ParentChild=4`, `AuntNiece=5`.

Details

Subjects will be in the same extended family if either: [1] they are Gen1 housemates, [2] they are Gen2 siblings, [3] they are Gen2 cousins (ie, they have mothers who are Gen1 sisters in the NLSY79, [4] they are mother and child (in Gen1 and Gen2, respectively), or [5] they are auntluncle and niecelnephew (in Gen1 and Gen2, respectively).

The variables `Subject1Tag` and `Subject2Tag` uniquely identify subjects. For Gen2 subjects, the SubjectTag is identical to their CID (ie, C00001.00 -the SubjectID assigned in the NLSY79-Children files). However for Gen1 subjects, the SubjectTag is their CaseID (ie, R00001.00), with "00" appended. This manipulation is necessary to identify subjects uniquely in inter-generational datasets. A Gen1 subject with an ID of 43 has a `SubjectTag` of 4300. The SubjectTags of her four children remain 4301, 4302, 4303, and 4304.

Level 5 of `RelationshipPath` (ie, AuntNiece) is gender neutral. The relationship could be either Aunt-Niece, Aunt-Nephew, Uncle-Niece, or Uncle-Nephew. If there's a widely-accepted gender-neutral term, please tell me.

MZ twins have `R`=1. DZ twins and full-siblings have `R`=.5. Half-siblings have `R`=.25. Typical first-cousins have `R`=.125 Unrelated subjects have `R`=0 (this occasionally happens for `Gen1Housemates`). Other `R` coefficients are possible. ??Joe, do you have an earlier paper that enumerates all the obscure combinations, like half-cousins or an ambiguous AuntNiece??

Author(s)

Will Beasley

Source

Gen1 information comes from the May 15, 2010 release of the NLSY79 sample. Gen2 information comes from the Sept 15, 2010 release of the NLSY79 Children and Young Adults sample. Data were extracted with the NLS Investigator (`https://www.nlsinfo.org/investigator/`).

The internal version for the links is `Links2011V51`.

References

The NLSY79 variable HHID (ie, R00001.49) is the source for the `ExtendedID` variable. This is discussed at `http://www.nlsinfo.org/nlsy79/docs/79html/79text/hhcomp.htm`. ** We need an introduction reference/information for 'R'/Relatedness. **

See Also

The `LinksPair79` dataset contains columns necessary for a basic BG analysis. The `Links79PairExpanded` dataset contains further information that might be useful in more complicated BG analyses.

A tutorial that produces a similar dataset is `http://www.nlsinfo.org/childya/nlsdocs/tutorials/linking_mothers_and_children/linking_mothers_and_children_tutorial.html`. It provides examples in SAS, SPSS, and STATA.

The current dataset (ie, `Links79Pair`) can be saved as a CSV file (comma-separated file) and imported into in other programs and languages. In the R console, type the following two lines of code:

```
library(NlsyLinks); data(Links79Pair)

write.csv(Links79Pair, "C:/BGDirectory/Links79Pair.csv")
```

where `"C:/BGDirectory/"` is replaced by your preferred directory. Remember to use forward slashes instead of backslashes; for instance, the path `"C:\BGDirectory\Links79Pair.csv"` will be misinterpreted.

Examples

```
library(NlsyLinks) #Load the package into the current R session.
data(Links79Pair)  #Load the dataset from the NlsyLinks package.
summary(Links79Pair)  #Summarize the five variables.
hist(Links79Pair$R)  #Display a histogram of the Relatedness coefficients.
table(Links79Pair$R)  #Create a table of the Relatedness coefficients for the whole sample.

#Create a dataset of only Gen2 sibs, and display the distribution of R.
gen2Siblings <- subset(Links79Pair, RelationshipPath=='Gen2Siblings')
table(gen2Siblings$R)  #Create a table of the Relatedness coefficients for the Gen2 sibs.
```

Links79PairExpanded *Kinship linking file for pairs of relatives. It builds upon the* Links79Pair *dataset.*

Description

Please first read the documentation for Links79Pair. That dataset contains the same pairs/rows, but only a subset of the variables/columns.

Specifies the relatedness coefficient (ie, 'R') between subjects in the same extended family. Each row represents a unique relationship pair. An extended family with k subjects will have $k(k\text{-}1)/2$ rows. Typically, Subject1 is older while Subject2 is younger.

Currently this dataset contains only Gen2 siblings. However, it soon will be generalized to the five categories specified by the RelationshipPath variable. Code written using this dataset should NOT assume it contains only Gen2 sibiling pairs. See an example of filtering the relationship category in the in Links79Pair documentation.

Usage

```
data(Links79PairExpanded)
```

Format

A data frame with 11,075 observations on the following 22 variables. There is one row per unique pair of subjects, irrespective of order.

ExtendedID see the variable of the same name in Links79Pair

Subject1Tag see the variable of the same name in Links79Pair

Subject2Tag see the variable of the same name in Links79Pair

R see the variable of the same name in Links79Pair

RFull This is a superset of R. This includes all the *R* values we estimated, while R (i.e., the variable above) excludes values like *R*=0 for Gen1Housemates, and the associated relationships based on this *R* value (i.e., Gen2Cousins and AuntNieces).

RelationshipPath see the variable of the same name in Links79Pair

EverSharedHouse Indicate if the pair likely live in the same house. This is TRUE for Gen1Housemates, Gen2Siblings, and ParentChild. This is FALSE for AuntNiece and Gen2Cousins

IsMz Indicates if the pair is from the same zygote (ie, they are identical twins/triplets). This variable is a factor, with levels No=0, Yes=1, DoNotKnow=255.

`Subject1LastSurvey` The year of Subject1's most recently completed survey. This may be different that the survey's administration date.

`Subject2LastSurvey` The year of Subject2's most recently completed survey. This may be different that the survey's administration date.

`RImplicitPass1` The pair's R coefficient, using only implicit information. Interpolation was NOT used.

`RImplicit` The pair's R coefficient, using only implicit information. Interpolation was used.

`RImplicit2004` The pair's R coefficient released in our previous projects (**need reference**). This variable is provided primarily for previous users wishing to replicate previous analyses.

`RExplicitPass1` The pair's R coefficient, using only explicit information. Interpolation was NOT used.

`RExplicit` The pair's R coefficient, using only explicit information. Interpolation was used.

`RExplicitOlderSibVersion` The pair's R coefficient, according to the explicit item responses of the older sibling.

`RExplicitYoungerSibVersion` The pair's R coefficient, according to the explicit item responses of the younger sibling.

`RPass1` The pair's estimated R coefficient, using both implicit and explicit information. Interpolation was NOT used. The variable `R` is identically constructed, but it did use interpolation.

`GenerationSubject1` The generation of the first subject. Values for Gen1 and Gen2 are `1` and `2`, respectively.

`GenerationSubject2` The generation of the second subject. Values for Gen1 and Gen2 are `1` and `2`, respectively.

`Subject1ID` The ID value assigned by NLS to the first subject. For Gen1 Subjects, this is their "CaseID" (ie, R00001.00). For Gen2 subjects, this is their "CID" (ie, C00001.00).

`Subject2ID` The ID value assigned by NLS to the second subject.

`MathStandardized_1` The PIAT-Math score for Subject1. See `ExtraOutcomes79` for more information about its source.

`MathStandardized_2` The PIAT-Math score for Subject2. See `ExtraOutcomes79` for more information about its source.

`HeightZGenderAge_1` The early adult height for Subject1. See `ExtraOutcomes79` for more information about its source.

`HeightZGenderAge_2` The early adult height for Subject2. See `ExtraOutcomes79` for more information about its source.

Details

Please first read the documentation for `Links79Pair`. That dataset contains the same pairs/rows, but only a subset of the variables/columns.

The specific steps to determine the `R` coefficient will be described in an upcoming publication. The following information may influence the decisions of an applied researcher.

A distinction is made between 'Explicit' and 'Implicit' information. Explicit information comes from survey items that directly address the subject's relationships. For instance in 2006, surveys asked if the sibling pair share the same biological father (eg, Y19940.00 and T00020.00). Implicit information comes from items where the subject typically isn't aware that their responses may be used to determine genetic relatedness. For instance, if two siblings have biological fathers with the same month of death (eg, R37722.00 and R37723.00), it may be reasonable to assume they share the same biological father.

'Interpolation' is our lingo when other siblings are used to leverage insight into the current pair. For example, assume Subject 101, 102, and 103 have the same mother. Further assume 101 and 102 report they share a biological father, and that 101 and 103 share one too. Finally, assume that we don't have information about the relationship between 102 and 103. If we are comfortable with our level of uncertainty of these determinations, then we can interpolate/infer that 102 and 103 are full-siblings as well.

The math and height scores are duplicated from `ExtraOutcomes79`, but are included here to make some examples more concise and accessible.

Author(s)

Will Beasley

Source

See `Links79Pair`.

Examples

```
library(NlsyLinks) #Load the package into the current R session.
data(Links79PairExpanded)  #Load the dataset from the NlsyLinks package.
olderR <- Links79PairExpanded$RExplicitOlderSibVersion  #Declare a concise variable name.
youngerR <- Links79PairExpanded$RExplicitYoungerSibVersion  #Declare a concise variable name.

plot(jitter(olderR), jitter(youngerR))  #Scatterplot the siblings' responses.
table(youngerR, olderR)  #Table of the relationship between the siblings' responses.
ftable(youngerR, olderR, dnn=c("Younger's Version", "Older's Version")) #A formatted table.

#write.csv(Links79PairExpanded, file='F:/Projects/RDev/NlsyLinksStaging/Links79PairExpanded.csv',
#  row.names=FALSE)
```

`ReadCsvNlsy79` *Read a CSV file downloaded from the NLS Investigator*

Description

The function accepts a (file path to) CSV file and creates a `data.frame`. The `data.frame` is modified and augmented with columns to assist later routines.

Usage

```
ReadCsvNlsy79Gen2(filePath)
```

Arguments

`filePath`	A path to the CSV file. Remember to use double back-slashes in Windows, or forward slashes in Windows or *nix.

Details

The function does seven things. 1) Reads the CSV into a data.frame. 2) Checks that the NLSY variables C00001.00 and C00002.00 exist in the data.frame. 3) The NLSY variable C00001.00 is renamed SubjectID. 4) A variable named Generation is given a value of 2 for all subjects. 5) The SubjectTag variable is created. 6) The NLSY variable C00002.00 is multiplied by 100 and renamed SubjectTagOfMother. 7) The NLSY variable R00001.49 (ie, their Mother's HHID is attached to each Gen2 record).

Author(s)

Will Beasley

Examples

```
## Not run:
#filePathGen2 <- "F:/Projects/RDev/NlsyLinksStaging/Datasets/Gen2Birth.csv"
#ds <- ReadCsvNlsy79Gen2(filePath=filePathGen2)
## End(Not run)
```

RGroupSummary *Calculates summary statistics for each* R*elatedness Group in the sample.*

Description

Before and after running ACE Models, it is important to examine the characteristics of the different groups. When the ACE is estimated with an SEM using multiple groups, it is even even more important. Groups may contain too few subjects to have a well-behaved covariance matrix.

If a group's covariance matrix is not Positive Definite (or it's misbehaving in some other way), it's typically recommended to exclude that group from the SEM.

Usage

```
RGroupSummary(ds, oName_1, oName_2, rName, determinantThreshold=1e-5)
```

Arguments

ds	The data.frame containing the following variables:
oName_1	The name of the outcome variable corresponding to the first subject in the pair.
oName_2	The name of the outcome variable corresponding to the first subject in the pair.
rName	The name of the variable specifying the pair's Relatedness coefficient.
determinantThreshold	The minimum value the covariance matrix's determinant (for the group) should exceed to be considered Positive Definite.

Details

This function doesn't specific to an ACE model and groups defined by R. It could be applied to any multiple-group SEM with two manifest/outcome variables. In the future, we may generalize it beyond two manifest variables.

To get summary stats for the entire sample, create a dummy indicator variable that assigns everyone to the same group. See the second example below.

The default determinantThreshold value is nonzero, in order to forgive slight numerical inaccuracies caused by fixed-precision arithmetic.

Value

A data.frame with one row per group. The data.frame contains the following variables:

R	The group's R value. Note the name of this variable can be changed by the user, by specifying a non-default value to the rName argument.
Included	Indicates if the group should be included in a multiple-group SEM.
PairCount	The number of pairs in the group with *complete* data for R and the two outcome/manifest variables.
O1Variance	The variance (of the outcome variable) among the group's first members.
O2Variance	The variance (of the outcome variable) among the group's second members.
O1O2Covariance	The covariance (of the outcome variable) across the group's first and second members.
Correlation	The correlation (of the outcome variable) across the group's first and second members.
Determinant	The determinant of the group's covariance matrix.
PosDefinite	Indicates if the group's covariance matrix is positive definite.

Author(s)

Will Beasley and David Bard

References

Please see ZZZ (200?) for more information about SEM with multiple groups. TODO: refs for determinant & positive definite.

Examples

```
library(NlsyLinks) #Load the package into the current R session.
dsLinks <- Links79PairExpanded  #Load the dataset from the NlsyLinks package.
dsLinks <- dsLinks[dsLinks$RelationshipPath=='Gen2Siblings', ]
oName_1 <- "MathStandardized_1" #Stands for Outcome1
oName_2 <- "MathStandardized_2" #Stands for Outcome2
dsGroupSummary <- RGroupSummary(dsLinks, oName_1, oName_2)
dsGroupSummary

#Should return:
# R Included PairCount O1Variance O2Variance O1O2Covariance Correlation Determinant PosDefinite
# 1 0.250     TRUE      2719   169.1291   207.0233       40.66048   0.2172970    33360.38        TRUE
# 2 0.375     TRUE       141   167.9943   181.8788       40.67609   0.2327024    28900.07        TRUE
```

```
# 3 0.500     TRUE      5508   230.9663   233.3492      107.59822   0.4634764    42318.42        TRUE
# 4 0.750    FALSE         2   220.5000    18.0000       63.00000   1.0000000        0.00       FALSE
# 5 1.000     TRUE        22   319.1948   343.1169      277.58874   0.8387893    32465.62        TRUE

#To get summary stats for the whole sample, create one large inclusive group.
dsLinks$Dummy <- 1
(dsSampleSummary <- RGroupSummary(dsLinks, oName_1, oName_2, rName="Dummy"))

#Should return:
# Dummy Included PairCount M1Variance M2Variance M1M2Covariance Correlation Determinant PosDefinite
#1     1     TRUE      8392    216.466   229.2988       90.90266   0.4080195     41372.1        TRUE
###
### ReadCsvNlsy79
###
## Not run:
#filePathGen2 <- "F:/Projects/RDev/NlsyLinksStaging/Datasets/Gen2Birth.csv"
#ds <- ReadCsvNlsy79Gen2(filePath=filePathGen2)
## End(Not run)
```

`SubjectDetails79` *Dataset containing further details of the Gen1 and Gen2 subjects.*

Description

These variables are useful to many types of analyses (not just behavior genetics), and are provided to save users time.

Usage

```
data(SubjectDetails79)
```

Format

A data frame with 24,181 observations on the following 12 variables.

`SubjectTag` see the variable of the same name in `Links79Pair`

`ExtendedID` see the variable of the same name in `Links79Pair`

`Generation` Indicates if the subject is in generation 1 or 2.

`SiblingCountInNls` The number of the subject's siblings, including himself/herself (a singleton has a value of one). This considers only the siblings in the NLSY. For Gen1, this can exclude anyone outside the age range. For Gen2, this excludes anyone who doesn't share the same mother.

`BirthOrderInNls` Indicates the subject's birth order among the NLSY siblings.

`SimilarAgeCount` ??I forgot what this is??

`KidCountBio` The number of biological children known to the NLSY (but not necessarily interviewed by the NLSY.

`KidCountInNls` The number of children who belong to the NLSY. This is nonnull for only Gen1 subjects.

`Mob` The subject's month of birth. The exact day is not available to the public. By default, we set their birthday to the 15th day of the month.

`LastSurveyYearCompleted` ##This variable is not available yet## The year of the most recently completed survey.

`AgeAtLastSurvey` ##This variable is not available yet## The subject's age at the most recently completed survey.

`IsDead` ##This variable is not available yet## Indicates if the subject was alive for the last attempted survey.

`DeathDate` ##This variable is not available yet## The subject's month of death. The exact day is not available to the public. By default, we set their birthday to the 15th day of the month.

Author(s)

Will Beasley

Source

Gen1 information comes from the May 15, 2010 release of the NLSY79 sample. Gen2 information comes from the Sept 15, 2010 release of the NLSY79 Children and Young Adults sample. Data were extracted with the NLS Investigator (https://www.nlsinfo.org/investigator/).

Examples

```
library(NlsyLinks) #Load the package into the current R session.
data(SubjectDetails79)  #Load the dataset from the NlsyLinks package.

summary(SubjectDetails79)

oldPar <- par(mfrow=c(3,2), mar=c(2,2,1,.5), tcl=0, mgp=c(1,0,0))
hist(SubjectDetails79$SiblingCountInNls, main="",
     breaks=seq(from=0, to=max(SubjectDetails79$SiblingCountInNls, na.rm=TRUE), by=1)
)
hist(SubjectDetails79$BirthOrderInNls, main="",
     breaks=seq(from=0, to=max(SubjectDetails79$BirthOrderInNls, na.rm=TRUE), by=1)
)
hist(SubjectDetails79$SimilarAgeCount, main="",
     breaks=seq(from=0, to=max(SubjectDetails79$SimilarAgeCount, na.rm=TRUE), by=1)
)
hist(SubjectDetails79$KidCountBio, main="",
     breaks=seq(from=0, to=max(SubjectDetails79$KidCountBio, na.rm=TRUE), by=1)
)
hist(SubjectDetails79$KidCountInNls, main="",
     breaks=seq(from=0, to=max(SubjectDetails79$KidCountInNls, na.rm=TRUE), by=1)
)
#hist(SubjectDetails79$Mob, main="",
#     breaks=seq.Date(
#       from=min(SubjectDetails79$Mob, na.rm=TRUE),
#       to=max(SubjectDetails79$Mob, na.rm=TRUE),
#       by="year")
#)
par(oldPar)
```

`ValidateOutcomeDataset` *Validates the schema of datasets containing outcome variables.*

Description

The **NlsyLinks** handles a lot of the plumbing code needed to transform extracted NLSY datasets into a format that statistical routines can interpret. In some cases, a dataset of measured variables is needed, with one row per subject. This function validates the measured/outcome dataset, to ensure it posses an interpretable schema. For a specific list of the requirements, see `Details` below.

Usage

```
ValidateOutcomeDataset(dsOutcome, outcomeNames)
```

Arguments

`dsOutcome`	A data frame with the measured variables
`outcomeNames`	The column names of the measure variables that eventually will be used by a statistical procedure.

Details

The `dsOutcome` parameter must: 1) Have a non-missing value. 2) Contain at least one row. 3) Contain a column called 'SubjectTag' (case sensitive). 4) Have the SubjectTag column containing only positive numbers. 5) Have the SubjectTag column where all values are unique (ie, two rows/subjects cannot have the same value).

The `outcomeNames` parameter must: 1) Have a non-missing value 2) Contain only column names that are present in the `dsOutcome` data frame.

Value

Returns `TRUE` if the validation passes. Returns an error (and associated descriptive message) if it false.

Author(s)

Will Beasley

Examples

```
library(NlsyLinks) #Load the package into the current R session.
ds <- ExtraOutcomes79
outcomeNames <- c("MathStandardized", "WeightZGenderAge")
ValidateOutcomeDataset(dsOutcome=ds, outcomeNames=outcomeNames) #Returns TRUE.

outcomeNamesBad <- c("MathMisspelled", "WeightZGenderAge")
#ValidateOutcomeDataset(dsOutcome=ds, outcomeNames=outcomeNamesBad) #Throws error.
```

`ValidatePairLinks` *Validates the schema of a links for pairs of relatives*

Description

A helper function that verifies the linking dataset contains (A) the essential columns exist, and (B) at least one row. It is called by `CreatePairLinks`.

Usage

```
ValidatePairLinks(linksPair)
```

Arguments

`linksPair`	The `data.frame` to validate.

Details

Typical use of **NlsyLinks** will not require this function, since a valid paired links are supplied for each supported sample (ie, `Links79Pair`).

The **NlsyLinks** uses several types of linking schemas. This function validates the type where each relative subject has their own row.

The following four columns must be present: (1) `Subect1Tag`, (2) `Subect2Tag`, (3) `R`, and (4) `MultipleBirth`. They must have a `numeric` mode/datatype.

Value

Returns `TRUE` if the validation passes. Returns an error (and associated descriptive message) if it false.

Author(s)

Will Beasley

See Also

`Links79Pair`,`Links79PairExpanded`,

Examples

```
dsSingleLinks <- data.frame(
  ExtendedID=c(1, 1, 1, 2),
  Subject1Tag=c(101, 101, 102, 201),
  Subject2Tag=c(102, 103, 103, 202),
  R=c(.5, .25, .25, .5),
  RelationshipPath=rep("Gen2Siblings", 4)
)
dsSingleOutcomes <- data.frame(
  SubjectTag=c(101, 102, 103, 201, 202),
  DV1=c(11, 12, 13, 41, 42),
  DV2=c(21, 22, 23, 51, 52))
dsDouble <- CreatePairLinksDoubleEntered(
```

```
  outcomeDataset=dsSingleOutcomes,
  linksPairDataset=dsSingleLinks,
  outcomeNames=c("DV1", "DV2"),
  validateOutcomeDataset=TRUE)
dsDouble #Show the 8 rows in the double-entered pair links
summary(dsDouble) #Summarize the variables

ValidatePairLinksAreSymmetric(dsDouble) #Should return TRUE.
```

`ValidatePairLinksAreSymmetric` *Verifies that the pair relationships are symmetric.*

Description

For certain analyses, the pairs links (which can be considered a type of sparse matrix) need to be symmetric. For instance, if there is a row for Subjects 201 and 202 with R=0.5, there should be a second row for Subjects 202 and 201 with R=0.5.

This validation function is useful to some types of DF methods and some spatially-inspired methods.

Usage

```
ValidatePairLinksAreSymmetric(linksPair)
```

Arguments

`linksPair`	The `data.frame` object that should be symmetric

Value

Returns `TRUE` if symmetric. Throw an error with `stop` if asymmetric.

Author(s)

Will Beasley

See Also

`CreatePairLinksDoubleEntered`

Examples

```
dsSingleLinks <- data.frame(
  ExtendedID=c(1, 1, 1, 2),
  Subject1Tag=c(101, 101, 102, 201),
  Subject2Tag=c(102, 103, 103, 202),
  R=c(.5, .25, .25, .5),
  RelationshipPath=rep("Gen2Siblings", 4)
)
dsSingleOutcomes <- data.frame(
  SubjectTag=c(101, 102, 103, 201, 202),
  DV1=c(11, 12, 13, 41, 42),
  DV2=c(21, 22, 23, 51, 52))
```

```
dsDouble <- CreatePairLinksDoubleEntered(
  outcomeDataset=dsSingleOutcomes,
  linksPairDataset=dsSingleLinks,
  outcomeNames=c("DV1", "DV2"),
  validateOutcomeDataset=TRUE)
dsDouble #Show the 8 rows in the double-entered pair links
summary(dsDouble) #Summarize the variables

ValidatePairLinksAreSymmetric(dsDouble) #Should return TRUE.
```

Index