Package 'NormData'

August 21, 2026

Type Package

Title Derivation of Regression-Based Normative Data

Version 1.2

Description

Normative data are often used to estimate the relative position of a raw test score in the population. This package allows for deriving regression-based normative data. It includes functions that enable the fitting of regression models for the mean and residual (or variance) structures, test the model assumptions, derive the normative data in the form of normative tables or automatic scoring sheets, and estimate confidence intervals for the norms. This package accompanies the book Van der Elst, W. (2024). Regression-based normative data for psychological assessment. A hands-on approach using R. Springer Nature.

Depends R (>= 3.5.0)

Imports car, doBy, MASS, lmtest, dplyr, sandwich, openxlsx, methods

License GPL (>= 2)

Repository CRAN

NeedsCompilation no

Author Wim Van der Elst [aut, cre] (ORCID:
<https://orcid.org/0000-0003-4315-7406>)

Maintainer Wim Van der Elst <Wim.vanderelst@gmail.com>

Date/Publication 2026-08-21 07:20:10 UTC

Contents

`Bootstrap.Stage.2.NormScore`

Bootstraps a confidence interval for a percentile rank

Description

The function `Stage.2.NormScore()` can be used to convert a raw test score of a tested person Y_0 into a percentile rank $\hat{\pi}_0$ (taking into account specified values of the independent variables). The function `Bootstrap.Stage.2.NormScore()` can be used to obtain a confidence interval (CI) around the point estimate of the percentile rank $\hat{\pi}_0$. A non-parametric bootstrap is used to compute a confidence interval (CI) around the estimated percentile rank (for details, see Chapter 8 in Van der Elst, 2023).

Usage

```
Bootstrap.Stage.2.NormScore(Stage.2.NormScore,
CI=.99, Number.Bootstraps=2000, Seed=123,
Rounded=FALSE, Show.Fitted.Boot=FALSE, verbose=TRUE)
```

Arguments

`Stage.2.NormScore`	A fitted object of class `Stage.2.NormScore`.
`CI`	The desired CI around the percentile rank for the raw test score at hand. Default `CI=.99`.
`Number.Bootstraps`	The number of bootstrap samples that are taken. Default `Number.Bootstraps=2000`.
`Seed`	The seed to be used in the bootstrap (for repoducibility). Default `Seed = 123`.
`Rounded`	Logical. Should the percentile rank be rounded to a whole number? Default `Rounded=FALSE`.
`Show.Fitted.Boot`	Logical. Should the fitted Stage 1 models for the bootstrap samples be printed? Default `Show.Fitted.Boot=FALSE`.
`verbose`	A logical value indicating whether verbose output should be generated.

Details

For details, see Chapter 8 in Van der Elst (2023).

Value

An object of class `Stage.2.NormScore` with components,

`CI.Percentile`	The bootstrapped CI around the estimated percentile rank.
`CI`	The CI used.
`All.Percentiles`	All bootstrapped percentile ranks for the raw test score at hand.
`Assume.Homoscedasticity`	Logical. Was homoscedasticity assumed in the normative conversion? For details, see `Stage.2.NormScore`.
`Assume.Normality`	Logical. Was normality assumed in the normative conversion? For details, see `Stage.2.NormScore`.
`Stage.2.NormScore`	The fitted `Stage.2.NormScore` object used in the function call.
`Percentile.Point.Estimate`	The point estimate for the percentile rank (based on the original dataset).

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

Stage.2.NormScore

Examples

```
 # Time-intensive part
# Replicate the bootstrap results that were obtained in
# Case study 1 of Chapter 8 in Van der Elst (2023)
# ------------------------------------------------------
library(NormData) # load the NormData package
data(GCSE)        # load the GCSE dataset

# Fit the Stage 1 model
Model.1.GCSE <- Stage.1(Dataset=GCSE,
  Model=Science.Exam~Gender)

# Stage 2: Convert a science exam score = 30 obtained by a
# female into a percentile rank (point estimate)
Normed_Score <- Stage.2.NormScore(Stage.1.Model=Model.1.GCSE,
  Score=list(Science.Exam=30, Gender="F"), Rounded = FALSE)
summary(Normed_Score)

# Derive the 99pc CI around the point estimate
# using a bootstrap procedure
Bootstrap_Normed_Score <- Bootstrap.Stage.2.NormScore(
  Stage.2.NormScore=Normed_Score)

summary(Bootstrap_Normed_Score)

plot(Bootstrap_Normed_Score)


# Replicate the bootstrap results that were obtained in
# Case study 2 of Chapter 8 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset

# Make the new variable Age.C (= Age centered) that is
# needed to fit the final Stage 1 model,
# and add it to the Substitution dataset
Substitution$Age.C <- Substitution$Age - 50

# Fit the final Stage 1 model
Substitution.Model.9 <- Stage.1(Dataset=Substitution,
  Alpha=0.005, Model=LDST~Age.C+LE, Order.Poly.Var=1)
summary(Substitution.Model.9)

# Convert an LDST score = 40 obtained by a
# 20-year-old test participant with LE=Low
# into a percentile rank (point estimate)
Normed_Score <- Stage.2.NormScore(
```

```
   Stage.1.Model=Substitution.Model.9,
   Score=list(LDST=40, Age.C=20-50, LE = "Low"),
   Rounded = FALSE)

# Derive the 99pc CI around the point estimate
# using a bootstrap
Bootstrap_Normed_Score <- Bootstrap.Stage.2.NormScore(
   Stage.2.NormScore = Normed_Score)
summary(Bootstrap_Normed_Score)
plot(Bootstrap_Normed_Score)
```

`Bootstrap.Stage.2.NormTable` *Bootstraps confidence intervals for a normative table*

Description

The function `Stage.2.NormTable()` is used to derive a normative table that shows the percentile ranks $\hat{\pi}_0$ that correspond to a wide range of raw test scores Y_0 (stratified by the relevant independent variables). The function `Bootstrap.Stage.2.NormTable()` can be used to obtain confidence intervals (CIs) around the point estimates of the percentile ranks $\hat{\pi}_0$ in the normative table. A non-parametric bootstrap is used to compute these CIs (for details, see Chapter 8 in Van der Elst, 2023).

Usage

```
Bootstrap.Stage.2.NormTable(Stage.2.NormTable,
CI=.99, Number.Bootstraps=2000, Seed=123,
Rounded=FALSE, Show.Fitted.Boot=FALSE, verbose=TRUE)
```

Arguments

`Stage.2.NormTable`	A fitted object of class `Stage.2.NormTable`.
`CI`	The desired CI around the percentile ranks. Default `CI=.99`.
`Number.Bootstraps`	The number of bootstrap samples that are taken. Default `Number.Bootstraps=2000`.
`Seed`	The seed to be used in the bootstrap (for repoducibility). Default `Seed = 123`.
`Rounded`	Logical. Should the percentile ranks that are shown in the normative table be rounded to a whole number? Default `Rounded=FALSE`.
`Show.Fitted.Boot`	Logical. Should the fitted Stage 1 models for the bootstrap samples be printed? Default `Show.Fitted.Boot=FALSE`.
`verbose`	A logical value indicating whether verbose output should be generated.

Details

For details, see Chapter 8 in Van der Elst (2023).

Value

An object of class `Stage.2.NormTable` with components,

`NormTable.With.CI`
: The normative table with the bootstrapped CI.

`CI`
: The CI used.

`Assume.Homoscedasticity`
: Logical. Was homoscedasticity assumed in the normative conversion? For details, see `Stage.2.NormTable`.

`Assume.Normality`
: Logical. Was normality assumed in the in the normative conversion? For details, see `Stage.2.NormTable`.

`NormTable.With.CI.Min`
: A table with the lower bounds of the CIs.

`NormTable.With.CI.Max`
: A table with the upper bounds of the CIs.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

`Stage.2.NormTable`

Examples

```
 # Time-intensive part
# Replicate the bootstrap results that were obtained in
# Case study 1 of Chapter 8 in Van der Elst (2023)
# ------------------------------------------------------
library(NormData) # load the NormData package
data(GCSE)        # load the GCSE dataset

# Fit the Stage 1 model
Model.1.GCSE <- Stage.1(Dataset=GCSE,
  Model=Science.Exam~Gender)

# Normative table with CIs
NormTable.GCSE <- Stage.2.NormTable(
  Stage.1.Model=Model.1.GCSE,
  Test.Scores=seq(from=10, to=85, by=5),
  Grid.Norm.Table=data.frame(Gender=c("F", "M")),
  Rounded = FALSE)
summary(NormTable.GCSE)
```

```
# Bootstrap the CIs
Bootstrap_NormTable.GCSE <- Bootstrap.Stage.2.NormTable(
  Stage.2.NormTable = NormTable.GCSE)
summary(Bootstrap_NormTable.GCSE)


# Replicate the bootstrap results that were obtained in
# Case study 2 of Chapter 8 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset

# Make the new variable Age.C (= Age centered) that is
# needed to fit the final Stage 1 model,
# and add it to the Substitution dataset
Substitution$Age.C <- Substitution$Age - 50

# Fit the final Stage 1 model
Substitution.Model.9 <- Stage.1(Dataset=Substitution,
  Alpha=0.005, Model=LDST~Age.C+LE, Order.Poly.Var=1)

summary(Substitution.Model.9)

# Make the normative table
NormTable.LDST <- Stage.2.NormTable(
Stage.1.Model=Substitution.Model.9,
  Test.Scores=seq(from=25, to=40, by=5),
  Grid.Norm.Table=expand.grid(
  Age.C=seq(from=-30, to=30, by = 1),
  LE=c("Low", "Average", "High")), Rounded = FALSE)

# Bootstrap the CIs
Bootstrap_NormTable.LDST <- Bootstrap.Stage.2.NormTable(
  Stage.2.NormTable = NormTable.LDST)

summary(Bootstrap_NormTable.LDST)
```

`Check.Assum` *Check assumptions for a fitted Stage 1 model*

Description

Helper function to check the validity of the homoscedasticity and normality assumptions for a fitted Stage 1 model

Usage

```
Check.Assum(Stage.1.Model)
```

Arguments

`Stage.1.Model` The fitted `Stage.1` model.

Details

For details, see Van der Elst (2023).

Value

An object of class `Check.Assum` with component,

`Assume.Homo.S2` Is the homoscedasticity assumption valid?

`Assume.Normality.S2`
Is the normality assumption valid?

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

`Stage.1`

Examples

```
data("Substitution")
# Fit a model with a linear mean prediction function
Fit <- Stage.1(Dataset = Substitution, Model = LDST~Age)
Check.Assum(Fit)
   # Output shows that the homoscedasticity and normality
   # assumptions are both violated
```

CheckFit *Check the fit of the mean structure of a regression model*

Description

The function `CheckFit()` allows for evaluating the fit of the mean structure of a regression model by comparing sample means and model-predicted means. If the model fits the data well, there should be a good agreement between the sample means and the predicted mean test scores in the relevant subgroups. When the model only contains (binary and/or non-binary) qualitative independent variables, the subgroups correspond to all possible combinations of the different levels of the qualitative variables. When there are quantitative independent variables in the model, these have to be discretized first.

Usage

```
CheckFit(Stage.1.Model, Means, CI=.99, Digits=6)
```

Arguments

`Stage.1.Model`	The fitted `Stage.1` model.
`Means`	A formula in the form of `Test.Score~Independent.Var1+Independent.Var2+...`. The mean, SD, and N will be provided for all combinations of the independent variable values levels. Note that all indpendent variables should be factors (i.e., non -quantitative).
`CI`	The required confidence limits. Default `CI=.99`, i.e. the 99 percent CI.
`Digits`	The number of digits used when showing the results. Default `Digits=6`.

Details

For details, see Van der Elst (2023).

Value

An object of class `CheckFit` with component,

`Results.Observed`
: A table with the means, SDs, and N for the observed test score, for each combination of independent variable levels.

`Results.Predicted`
: A table with the mean predicted test scores, for each combination of independent variable levels.

`Miss`	The number of missing observations in the dataset.
`Dataset`	The dataset used in the analysis.
`Model`	The specified model for the mean.
`CI`	The requested CI around the mean.
`N`	The sample size of the specified dataset.
`Stage.1.Model`	The fitted `Stage.1.Model` used in the analysis.
`Saturated`	Is the fitted `Stage.1.Model` a saturated model?

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

`Stage.1`, `plot.CheckFit`

Examples

```
# Replicate the fit plot that was obtained in
# Case study 1 of Chapter 7 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset
head(Substitution)  # have a look at the first datalines in
# the Substitution dataset

# Final Stage 1 model
Substitution$Age.C <- Substitution$Age - 50
  # Add Age_Group (that discretizes the quantitative variable Age
  # into 6 groups with a span of 10 years in the dataset for use
  # by the CheckFit() function later on)
Substitution$Age_Group <- cut(Substitution$Age,
   breaks=seq(from=20, to=80, by=10))
Substitution.Model.9 <- Stage.1(Dataset=Substitution,
   Alpha=0.005, Model=LDST~Age.C+LE, Order.Poly.Var=1)

# Examine fit
Fit.LDST <- CheckFit(Stage.1.Model=Substitution.Model.9,
  Means=LDST~Age_Group+LE)
summary(Fit.LDST)
plot(Fit.LDST)


# Replicate the fit plot that was obtained in
# Case study 2 of Chapter 7 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
data(VLT)           # load the VLT dataset
head(VLT)           # have a look at the first datalines in
                    # the VLT dataset

# Fit the final Stage 1 model
VLT$Age.C <- VLT$Age - 50
VLT$Age.C2 <- (VLT$Age - 50)**2
  # Add Age_Group (that discretizes the quantitative variable Age
  # into 6 groups with a span of 10 years in the dataset for use
  # by the CheckFit() function later on)
VLT$Age_Group <- cut(VLT$Age, breaks=seq(from=20, to=80, by=10))

VLT.Model.4 <- Stage.1(Dataset = VLT, Alpha = .005,
  Model = Total.Recall ~ Age.C+Age.C2+Gender+LE+Age.C:Gender)

# Examine fit using fit plots for the Age Group by
# LE by Gender subgroups
Fit.Means.Total.Recall <- CheckFit(Stage.1.Model=VLT.Model.4,
  Means=Total.Recall~Age_Group+LE+Gender)

summary(Fit.Means.Total.Recall)
plot(Fit.Means.Total.Recall)
```

Coding *Check the coding of a variable*

Description

This function checks the coding of a variable, e.g., the dummy-coding scheme that will be used for binary or qualitative variables.

Usage

```
Coding(x, verbose=TRUE)
```

Arguments

x	The variable to be evaluated.
verbose	A logical value indicating whether verbose output should be generated.

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

Examples

```
data(Substitution)
Coding(Substitution$LE)
```

Densities *Plot densities*

Description

Plot densities for an outcome for different subgroups.

Usage

```
Densities(Dataset, Test.Score, IV, Color=TRUE,
Size.Legend=1, xlab="Test score", main, ...)
```

Arguments

`Dataset`	The name of the dataset.
`Test.Score`	The name of the outcome variable (e.g., a raw test score).
`IV`	The name of the stratification variable, that defines for which subgroups density plots should be provided. If `IV` is not specified, a single density is shown (no subgroups).
`Color`	Logical. Should densities for different subgroups be depicted in color? Default `Color=TRUE`.
`Size.Legend`	The size of the legend in the plot. Default `Size.Legend=1`.
`xlab`	The label on the X-axis. Default `xlab="Test score"`.
`main`	The title of the plot.
`...`	Other arguments to be passed to the `plot(function)`, e.g. `xlim=c(0, 100)`.

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

Examples

```
# Plot Gender-specific densities of the raw science exam
# scores in the GCSE dataset
data(GCSE)
Densities(Dataset = GCSE, Test.Score = Science.Exam, IV=Gender)

# Plot LE-specific densities of the residuals of a model
# where the Openness scale score is regressed on LE
data(Personality)
Fit <- Stage.1(Dataset = Personality, Model = Openness~LE)
summary(Fit)
Data.With.Residuals <- data.frame(Personality,
  Fit$HomoNorm$Residuals)
Densities(Dataset = Data.With.Residuals,
  Test.Score = Fit.HomoNorm.Residuals, IV = LE)
```

`ExploreData` *Explore data*

Description

This function provides summary statistics of a test score (i.e., the mean, SD, N, standard error of the mean, and CI of the mean), stratified by the independent variable(s) of interest. The independent variables should be factors (i.e., binary or non-binary qualitiative variables).

Usage

```
ExploreData(Dataset, Model, CI=.99, Digits=6)
```

Arguments

`Dataset`	A dataset.
`Model`	A formula in the form of `Test.Score~IV.1+IV.2+...`. Summary statistics (i.e., the mean, SD, N, standard error of the mean, and CI of the mean) are provided for all combinations of the levels of the IVs (independent variables). Note that all IVs should be factors (i.e., binary or non-binary qualitative variables).
`CI`	The CI for the mean. Default `CI=.99`, i.e. the 99 CI.
`Digits`	The number of digits used when showing the results. Default `Digits=6`.

Details

For details, see Van der Elst (2023).

Value

An object of class `ExploreData` with component,

`Results`	A table with the summary statistics.
`Miss`	The number of missing observations in the dataset.
`Dataset`	The dataset used in the analysis.
`Model`	The specified model.
`CI`	The requested CI around the mean.
`N`	The sample size of the specified dataset.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

Examples

```
# Replicate the exploratory analyses that were conducted
# in Case study 1 of Chapter 5 in Van der Elst (2023)
# ------------------------------------------------------
library(NormData) # load the NormData package

data(Personality) # load the Personality dataset
Explore_Openness <- ExploreData(Dataset=Personality,
  Model=Openness~LE)
summary(Explore_Openness)
plot(Explore_Openness,
  main="Mean Openness scale scores and 99pc CIs")


# Replicate the exploratory analyses that were conducted
# in Case study 1 of Chapter 7 in Van der Elst (2023)
# ------------------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset
head(Substitution)  # have a look at the first datalines in
                    # the Substitution dataset

# First make a new variable Age_Group, that discretizes the
# quantitative variable Age into 6 groups with a span of 10 years
Substitution$Age_Group <- cut(Substitution$Age,
   breaks=seq(from=20, to=80, by=10))

# Compute descriptives of the LDST score for different Age Group
# by LE combinations
Explore.LDST.Age.LE <- ExploreData(Dataset=Substitution,
   Model=LDST~Age_Group+LE)
summary(Explore.LDST.Age.LE)

# Make a plot of the results.
plot(Explore.LDST.Age.LE,
   main="Mean (99pc CI) LDST scores by Age group and LE")

# Compute descriptives of the LDST score for different
# Age Group by Gender combinations
Explore.LDST.Age.Gender <- ExploreData(Dataset=Substitution,
  Model=LDST~Age_Group+Gender)

# Plot the results
plot(Explore.LDST.Age.Gender,
  main="Mean (99pc CI) LDST scores by Age group and Gender")

# Compute descriptives of the LDST score for different
# LE by Gender combinations
Explore.LDST.LE.Gender <-
  ExploreData(Dataset=Substitution, Model=LDST~LE+Gender)

# Plot the results
```

```
plot(Explore.LDST.LE.Gender,
  main="Mean (99pc CI) LDST scores by LE and Gender")

# Compute summary statistics of the LDST score in the
# Age Group by LE by Gender combinations
Explore.LDST <- ExploreData(Dataset=Substitution,
   Model=LDST~Age_Group+LE+Gender)

# Plot the results
plot(Explore.LDST)
```

`Fluency` *Verbal fluency data*

Description

This dataset contains the scores of the Fruits Verbal Fluency Test. The $N = 1241$ test participants were instructed to generate as many words as possible that belong to the category 'fruits' (e.g., apple, orange, banana, etc.) within 60 seconds. These are simulated data based on the results described in Rivera *et al.* (2019).

Usage

```
data(Fluency)
```

Format

A `data.frame` with 1241 observations on 3 variables.

`Id` The Id number of the test participant.

`Country` The country where the test participant lives, coded as a factor.

`Fruits` The number of correctly generated fruit names. Higher score is better.

References

Rivera *et al.* (2019). Normative Data For Verbal Fluency in Healthy Latin American Adults: Letter M, and Fruits and Occupations Categories. *Neuropsychology, 33,* 287-300.

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

`Fract.Poly` *Fit fractional polynomials*

Description

Fit a fractional polynomial model with m terms of the form X^p, where the exponents p are selected from a small predefined set S of both integer and non-integer values. This function can be useful to model the mean or variance prediction function in a more flexible way than by using linear, quadratic or cubic polynomials.

Usage

```
Fract.Poly(IV, Outcome,
S=c(-3, -2.5, -2.0, -1.5, -1, -0.5, 0.5, 1, 1.5, 2, 2.5, 3),
Max.M=3)
```

Arguments

`IV`	The Independent Variable to be considered in the model.
`Outcome`	The outcome to be considered in the model.
`S`	The set S from which each power p^m is selected. Default `S={-3, -2.5, -2.0, -1.5, -1, -0.5, 0.5, 1, 1.5, 2, 2.5, 3}`.
`Max.M`	The maximum order M to be considered for the fractional polynomial. This value can be 5 at most. When $M = 5$, then fractional polynomials of order 1 to 5 are considered. Default `Max.M=3`.

Value

`All.Results`	The results (powers and AIC values) of the fractional polynomials.
`Lowest.AIC`	Table with the fractional polynomial model that has the lowest AIC.
`Best.Model`	The best fitted model (`lm` object).
`IV`	The IV tha was considered in the model.
`Outcome`	The outcome that was considered in the model.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

Examples

```
data(VLT)
# Fit fractional polynomials of orders 1 to 2
FP <- Fract.Poly(IV = VLT$Age, Outcome = VLT$Total.Recall,
  Max.M=2)
FP$Lowest.AIC
FP$Best.Model
# Model with lowest AIC: 127.689 + (-190.731 * (Age**(-0.5))) +
#  (-7.586 * (Age**(0.5)))

# Make plot
plot(x=VLT$Age, y=VLT$Total.Recall, col="grey")
  # add best fitted fractional polynomial
Age.Vals.Plot <- 20:80
Pred.Vals <- 127.689 + (-190.731 * (Age.Vals.Plot**(-0.5))) +
   (-7.586 * (Age.Vals.Plot**(0.5)))
lines(x=Age.Vals.Plot, y=Pred.Vals, lwd=2, col="red", lty=2)
legend("topright", lwd=2, col="red", lty=2,
  legend="Mean Prediction Function, Fractional Polynomial")
```

GCSE *GCSE exam score*

Description

Thiis dataset contains the scores on a written science exam (General Certificate of Secondary Education; GCSE) that is taken by $N = 1905$ students in 73 schools in England. The exam is taken at the end of compulsory schooling, when students are typically 16 years old. The actual score maximum is 160, but here a rescaled score (with max value 100) is provided. The data originally come from the package `mlmRev`, dataset `Gcsemv`.

Usage

```
data(GCSE)
```

Format

A `data.frame` with 1905 observations on 3 variables.

`Id` The Id number of the student.

`Gender` The gender of the student, coded as M = male and F = female.

`Science.Exam` The science exam score.

GLT *Conduct the General Linear Test (GLT) procedure*

Description

The function GLT fits two nested linear regression models (that are referred to as the unrestricted and the restricted models), and evaluates whether or not the fit of both models differs significantly.

Usage

```
GLT(Dataset, Unrestricted.Model, Restricted.Model, Alpha=0.05,
Alpha.Homosc=0.05, Assume.Homoscedasticity=NULL)
```

Arguments

Dataset	A data.frame that should consist of one line per test participant. Each line should contain (at least) one test score and one independent variable.
Unrestricted.Model	The unrestricted regression model to be fitted. A formula should be provided using the syntaxis of the lm function (for help, see ?lm). For example, Test.Score~Gender will fit a linear regression model in which Gender is regressed on Test.Score. Test.Score~Gender+Age+Gender:Age will regress Test.Score on Gender, Age, and their interaction.
Restricted.Model	The restricted regression model to be fitted.
Alpha	The significance level that should be used in the GLT procedure. Default Alpha=0.05.
Alpha.Homosc	The significance level to conduct the homoscedasticity test. If the unrestricted model only contains qualitative independent variables, the Levene test is used. If the model contains at least one quantitative independent variables, the Breusch-Pagan test is used. If the homoscedasticity assumption is violated, a heteroscedasticity-robust F* test is provided. Default Alpha.Homosc=0.05.
Assume.Homoscedasticity	Logical. The NormData package 'decides' whether the homoscedasticity assumption is valid based on the Levene (or Breusch-Pagan) test. The Assume.Homoscedasticity= TRUE/FALSE argument can be used to overrule this decision process and 'force' the NormData package to assume or not assume homoscedasticity.

Details

For details, see Van der Elst (2023).

Value

An object of class GLT with components,

F.Test.Stat.Results
: The result of the GLT procedure, i.e., the SSEs and DFs the fitted unrestricted and restricted models, and the F^* test-statistic.

Fit.Unrestricted.Model
: The fitted unrestricted model.

Fit.Restricted.Model
: The fitted restricted model.

Alpha
: The significance level that was used.

p.val.homoscedasticity
: The p-value that was used in the homoscedasticity test for the unrestricted model.

F.Test.Hetero.Robust
: The result of the heteroscedasticity-robust F* test. For details, see the waldtest function of the lmtest package (see ?waldtest).

Alpha.Homoscedasticity
: The significance level that was used to conduct the homoscedasticity test. Default Alpha.Homoscedasticity=0.05.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

Examples

```
# Replicate the GLT results that were obtained in
# Case study 1 of Chapter 5 in Van der Elst (2023)
# ------------------------------------------------
data(Personality)

GLT.Openness <- GLT(Dataset=Personality,
     Unrestricted.Model=Openness~LE, Restricted.Model=Openness~1)
summary(GLT.Openness)

# Replicate the GLT results that were obtained in
# Case study 2 of Chapter 5 in Van der Elst (2023)
# ------------------------------------------------
data(Fluency)

GLT.Fruits <- GLT(Dataset=Fluency,
     Unrestricted.Model=Fruits~LE, Restricted.Model=Fruits~1)
summary(GLT.Fruits)
```

ICC *Intra class correlation*

Description

The function ICC computes the intra class correlation. The ICC corresponds to the proportion of the total variance in the residuals that is accounted for by the clustering variable at hand (Kutner *et al.*, 2005).

Usage

```
ICC(Cluster, Test.Score, Dataset, CI = 0.95)
```

Arguments

`Cluster`	The name of the clustering variable in the dataset.
`Test.Score`	The name of the outcome variable in the dataset (e.g., a test score).
`Dataset`	A dataset.
`CI`	The required confidence limits around the ICC. Default `CI=.95`, i.e. the 95 CI.

Details

This function is a modification of the `ICCest` function from the `ICC` package (v2.3.0), with minimal changes. For details of the original function, see https://cran.r-project.org/web/packages/ICC/ICC.pdf. The author of the original function is Matthew Wolak.

Value

An object of class `ICC` with component,

`ICC`	The intra class correlation coefficient.
`LowerCI`	The lower bound of the CI around the ICC.
`UpperCI`	The upper bound of the CI around the ICC.
`Num.Clusters`	The number of clusters in the dataset.
`Mean.Cluster.Size`	The mean number of observations per cluster.
`Data`	The dataset used in the analysis (observations with missing values are excluded).
`N.Dataset`	The sample size of the full dataset.
`N.Removed`	The number of observations that are removed due to missingness.
`alpha`	The specified α-level for the CI, i.e., α = 1 - CI.
`Labels.Cluster`	The labels of the clustering variable.

Author(s)

Original function: Matthew Wolak (with some small modifications by Wim Van der Elst)

References

https://cran.r-project.org/web/packages/ICC/ICC.pdf

Kutner, M. H., Nachtsheim, C. J., Neter, J., and Li, W. (2005). *Applied linear statistical models* (5th edition). New York: McGraw Hill.

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

plot.ICC

Examples

```
# Compute ICC in Substitution dataset, using Test.Administrator as
# clustering unit
data(Substitution)

# Add administrator to the dataset (just randomly allocate labels
# as Test.Administrator, so ICC should be approx. 0)
Substitution$Test.Adminstrator <- NA
Substitution$Test.Adminstrator <- sample(LETTERS[1:10],
  replace = TRUE, size = length(Substitution$Test.Adminstrator))
Substitution$Test.Adminstrator <-
  as.factor(Substitution$Test.Adminstrator)

ICC_LDST <- ICC(Cluster = Test.Adminstrator, Test.Score = LDST, Data = Substitution)

# Explore results
summary(ICC_LDST)
plot(ICC_LDST)
```

Levels *Explore data*

Description

Gives the levels of a variable.

Usage

```
Levels(x)
```

Arguments

x	A variable for which the different levels should be printed.

Details

For details, see Van der Elst (2023).

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

Examples

```
data(Substitution)
Levels(Substitution$Gender)
```

`Personality` *Data of the Openness scale of a personality test*

Description

These are the data of the Openness subscale of International Personality Item Pool (ipip.ori.org). This subscale consists of 5 items: 1 = *I am full of ideas*, 2 = *I avoid difficult reading material*, 3 = *I carry the conversation to a higher level*, 4 = *I spend time reflecting on things*, and 5 = *I will not probe deeply into a subject*. Each item is scored on a 6-point response scale with answer categories 1 = very inaccurate, 2 = moderately inaccurate, 3 = slightly inaccurate, 4 = slightly accurate, 5 = moderately accurate, and 6 = very accurate. The Openness scale score corresponds to the sum of the individual item scores, with items 2 and 5 being reverse scored. The raw Openness scale score ranges between 5 and 30. A higher score is indicative of higher levels of curiosity, intellectualism, imagination, and aesthetic interests (McCrae, 1994).

The data were collected as part of the Synthetic Apeture Personality Assessment (SAPA http://sapa-project.org) web-based personality assessment project.

Usage

```
data(Personality)
```

Format

A `data.frame` with 2137 observations on 3 variables.

`Id` The Id number of the participant.

`LE` The Level of Education (LE) of the participant, coded as 1 = less than high school, 2 = finished high school, 3 = some college but did not graduate, 4 = college graduate, and 5 = graduate degree.

`Openness` Level of Openness.

References

McCrae, R. R. (1994). Openness to Experience: expanding the boundaries of factor V. *European Journal of Personality, 8,* 251-272.

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

`plot Bootstrap.Stage.2.NormScore`
Plot the bootstrap distribution and the percentile bootstrap CI

Description

This function plots the bootstrap distribution and the percentile bootstrap CI for a test score based on a `Bootstrap.Stage.2.NormScore` object. A non-parametric bootstrap is used to compute a confidence interval (CI) around the estimated percentile rank (for details, see Chapter 8 in Van der Elst, 2023).

Usage

```
## S3 method for class 'Bootstrap.Stage.2.NormScore'
plot(x,
cex.axis=1, cex.main=1, cex.lab=1, ...)
```

Arguments

`x`	A fitted object of class `Bootstrap.Stage.2.NormScore`.
`cex.axis`	The magnification to be used for axis annotation.
`cex.main`	The magnification to be used for the main label.
`cex.lab`	The magnification to be used for X and Y labels.
`...`	Other arguments to be passed to the `plot()` function.

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

Bootstrap.Stage.2.NormScore

Examples

```
  # Time-intensive part
# Replicate the bootstrap results that were obtained in
# Case study 1 of Chapter 8 in Van der Elst (2023)
# ------------------------------------------------------
library(NormData) # load the NormData package
data(GCSE)        # load the GCSE dataset

# Fit the Stage 1 model
Model.1.GCSE <- Stage.1(Dataset=GCSE,
  Model=Science.Exam~Gender)

# Stage 2: Convert a science exam score = 30 obtained by a
# female into a percentile rank (point estimate)
Normed_Score <- Stage.2.NormScore(Stage.1.Model=Model.1.GCSE,
  Score=list(Science.Exam=30, Gender="F"), Rounded = FALSE)
summary(Normed_Score)

# Derive the 99pc CI around the point estimate
# using a bootstrap procedure
Bootstrap_Normed_Score <- Bootstrap.Stage.2.NormScore(
  Stage.2.NormScore=Normed_Score)

summary(Bootstrap_Normed_Score)

plot(Bootstrap_Normed_Score)


# Replicate the bootstrap results that were obtained in
# Case study 2 of Chapter 8 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset

# Make the new variable Age.C (= Age centered) that is
# needed to fit the final Stage 1 model,
# and add it to the Substitution dataset
Substitution$Age.C <- Substitution$Age - 50

# Fit the final Stage 1 model
Substitution.Model.9 <- Stage.1(Dataset=Substitution,
  Alpha=0.005, Model=LDST~Age.C+LE, Order.Poly.Var=1)
summary(Substitution.Model.9)

# Convert an LDST score = 40 obtained by a
# 20-year-old test participant with LE=Low
# into a percentile rank (point estimate)
Normed_Score <- Stage.2.NormScore(
   Stage.1.Model=Substitution.Model.9,
   Score=list(LDST=40, Age.C=20-50, LE = "Low"),
   Rounded = FALSE)
```

```
# Derive the 99pc CI around the point estimate
# using a bootstrap
Bootstrap_Normed_Score <- Bootstrap.Stage.2.NormScore(
   Stage.2.NormScore = Normed_Score)
summary(Bootstrap_Normed_Score)
plot(Bootstrap_Normed_Score)
```

`plot CheckFit` *Evaluate the fit of the mean structure of a fitted Stage 1 model.*

Description

The function `CheckFit()` allows for evaluating the fit of the mean structure of a regression model by comparing sample means and model-predicted means. This function plots the sample means (with CIs) and the means of the model-predicted values. If the model fits the data well, there should be a good agreement between the sample means and the predicted mean test scores in the relevant subgroups. When the model only contains (binary and/or non-binary) qualitative independent variables, the subgroups correspond to all possible combinations of the different levels of the qualitative variables. When there are quantitative independent variables in the model, these have to be discretized first.

Usage

```
## S3 method for class 'CheckFit'
plot(x, Color, pch, lty,
Width.CI.Lines=.125, Size.symbol = 1,
No.Overlap.X.Axis=TRUE, xlab, ylab="Test score",
main = " ", Legend.text.size=1, Connect.Means,
cex.axis=1, cex.main=1.5, cex.lab=1.5, ...)
```

Arguments

`x`	A fitted object of class `CheckFit`.
`Color`	The colors to be used for the means. If not specified, the default colors are used.
`pch`	The symbols to be used for the means. If not specified, dots are used.
`lty`	The line types to be used for the means. If not specified, solid lines are used.
`Width.CI.Lines`	The width of the horizontal lines that are used to depict the CI around the mean. Default `Width.CI.Lines=0.125`.
`Size.symbol`	The size of the symbol used to depict the mean test score. Default `Size.symbol=1`.
`No.Overlap.X.Axis`	Logical. When a plot is constructed using two IVs (i.e., 2 or more lines of the mean and CIs in the plot), it is possible that the plot is unclear because the different means and CIs can no longer be distinguished. To avoid this, the levels of IV1 (plotted on the X-axis) can be assigned slightly different values for each level of IV2. For example, the mean for the subcategory males in age

	range [20; 40] will be shown at value X=0.9 (rather than 1) and the mean for the subcategory females in age range [20; 40] will be shown at value X=1.1 (rather than 1). In this way, the different means and CIs can be more clearly distinguished. Default `No.Overlap.X.Axis=TRUE`.
`xlab`	The label that should be added to the X-axis.
`ylab`	The label that should be added to the Y-axis. Default `ylab="Test score"`.
`main`	The title of the plot. Default `main=" "`.
`Legend.text.size`	The size of the text of the label for IV2. Default `Legend.text.size=1`.
`Connect.Means`	Logical. Should the symbols depicting the mean test scores be connected? If not specified, `Connect.Means = TRUE` is used if the model contains numeric independent variables and `Connect.Means = FALSE` otherwise.
`cex.axis`	The size of the labels on the X- and Y-axis. Default `cex.axis=1`.
`cex.main`	The magnification to be used for the main label.
`cex.lab`	The magnification to be used for X and Y labels.
`...`	Extra graphical parameters to be passed to `plot()`.

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

`Stage.1`, `plot.CheckFit`

Examples

```
# Replicate the fit plot that was obtained in
# Case study 1 of Chapter 7 in Van der Elst (2023)
# -----------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset
head(Substitution)  # have a look at the first datalines in
# the Substitution dataset

# Final Stage 1 model
Substitution$Age.C <- Substitution$Age - 50
  # Add Age_Group (that discretizes the quantitative variable Age
  # into 6 groups with a span of 10 years in the dataset for use
```

```
  # by the CheckFit() function later on)
Substitution$Age_Group <- cut(Substitution$Age,
   breaks=seq(from=20, to=80, by=10))
Substitution.Model.9 <- Stage.1(Dataset=Substitution,
   Alpha=0.005, Model=LDST~Age.C+LE, Order.Poly.Var=1)

# Examine fit
Fit.LDST <- CheckFit(Stage.1.Model=Substitution.Model.9,
  Means=LDST~Age_Group+LE)
summary(Fit.LDST)
plot(Fit.LDST)


# Replicate the fit plot that was obtained in
# Case study 2 of Chapter 7 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
data(VLT)           # load the VLT dataset
head(VLT)           # have a look at the first datalines in
                    # the VLT dataset

# Fit the final Stage 1 model
VLT$Age.C <- VLT$Age - 50
VLT$Age.C2 <- (VLT$Age - 50)**2
  # Add Age_Group (that discretizes the quantitative variable Age
  # into 6 groups with a span of 10 years in the dataset for use
  # by the CheckFit() function later on)
VLT$Age_Group <- cut(VLT$Age, breaks=seq(from=20, to=80, by=10))

VLT.Model.4 <- Stage.1(Dataset = VLT, Alpha = .005,
  Model = Total.Recall ~ Age.C+Age.C2+Gender+LE+Age.C:Gender)

# Examine fit using fit plots for the Age Group by
# LE by Gender subgroups
Fit.Means.Total.Recall <- CheckFit(Stage.1.Model=VLT.Model.4,
  Means=Total.Recall~Age_Group+LE+Gender)

summary(Fit.Means.Total.Recall)
plot(Fit.Means.Total.Recall)
```

`plot ExploreData` *Plot means and CIs for test scores.*

Description

Plot the means (and CIs) for the test scores, stratified by the independent variable(s) of interest. The independent variables should be factors (i.e., binary or non-binary qualitiative variables).

Usage

```
## S3 method for class 'ExploreData'
plot(x, Width.CI.Lines=.125, Size.symbol = 1,
No.Overlap.X.Axis=TRUE, xlab, ylab="Test score", main,
Color, pch, lty, Black.white=FALSE, Legend.text.size=1,
Connect.Means = TRUE, Error.Bars = "CI",
cex.axis=1, cex.main=1, cex.lab=1, ...)
```

Arguments

x	A fitted object of class ExploreData.
Width.CI.Lines	The width of the horizontal lines that are used to depict the CI around the mean. Default Width.CI.Lines=0.125.
Size.symbol	The size of the symbol used to depict the mean test score. Default Size.symbol=1.
No.Overlap.X.Axis	Logical. When a plot is constructed using multiple IVs (specified in the Model= argument of the ExploreData() function), it is possible that the plot becomes unclear because the different means (and CIs) largely overlap. To avoid this, the levels of IV1 (plotted on the X-axis) can be slightly shifted for each level of IV2. For example, if IV1=Age group and IX2=Gender, the mean for the subcategory males in age range [20; 40] will be shown at value 0.9 on the X-axis (rather than 1) and the mean for the subcategory females in age range [20; 40] will be shown at value 1.1 (rather than 1), and similarly for all levels of IV1. In this way, the different means and CIs can be more clearly distinguished. Default No.Overlap.X.Axis=TRUE.
xlab	The label that should be added to the X-axis.
ylab	The label that should be added to the Y-axis. Default ylab="Test score".
main	The title of the plot.
Color	The colors that should be used for the means. If not specified, the default colors are used.
pch	The symbols to be used for the means. If not specified, dots are used.
lty	The line types to be used for the means. If not specified, solid lines are used (i.e., lty=1.
Black.white	Logical. Should the plot be in black and white (rather than in color)? Default Black.white=FALSE.
Legend.text.size	The size of the text of the label for IV2. Default Legend.text.size=1.
Connect.Means	Logical. Should the symbols depicting the mean test scores be connected? Default Connect.Means = TRUE.
Error.Bars	The type of error bars around the means that should be added in the plot: confidence intervals (Error.Bars = "CI"), standard errors (Error.Bars = "SE"), standard deviations (Error.Bars = "SD") or no error bars (Error.Bars = "None"). Default Error.Bars = "CI".
cex.axis	The magnification to be used for axis annotation.

`cex.main`	The magnification to be used for the main label.
`cex.lab`	The magnification to be used for X and Y labels.
`...`	Extra graphical parameters to be passed to `plot()`.

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

ExploreData

Examples

```
# Replicate the exploratory analyses that were conducted
# in Case study 1 of Chapter 5 in Van der Elst (2023)
# ------------------------------------------------------
library(NormData) # load the NormData package

data(Personality) # load the Personality dataset
Explore_Openness <- ExploreData(Dataset=Personality,
  Model=Openness~LE)
summary(Explore_Openness)
plot(Explore_Openness,
  main="Mean Openness scale scores and 99pc CIs")


# Replicate the exploratory analyses that were conducted
# in Case study 1 of Chapter 7 in Van der Elst (2023)
# ------------------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset
head(Substitution)  # have a look at the first datalines in
                    # the Substitution dataset

# First make a new variable Age_Group, that discretizes the
# quantitative variable Age into 6 groups with a span of 10 years
Substitution$Age_Group <- cut(Substitution$Age,
   breaks=seq(from=20, to=80, by=10))

# Compute descriptives of the LDST score for different Age Group
# by LE combinations
```

```
Explore.LDST.Age.LE <- ExploreData(Dataset=Substitution,
   Model=LDST~Age_Group+LE)
summary(Explore.LDST.Age.LE)

# Make a plot of the results.
plot(Explore.LDST.Age.LE,
   main="Mean (99pc CI) LDST scores by Age group and LE")

# Compute descriptives of the LDST score for different
# Age Group by Gender combinations
Explore.LDST.Age.Gender <- ExploreData(Dataset=Substitution,
  Model=LDST~Age_Group+Gender)

# Plot the results
plot(Explore.LDST.Age.Gender,
  main="Mean (99pc CI) LDST scores by Age group and Gender")

# Compute descriptives of the LDST score for different
# LE by Gender combinations
Explore.LDST.LE.Gender <-
  ExploreData(Dataset=Substitution, Model=LDST~LE+Gender)

# Plot the results
plot(Explore.LDST.LE.Gender,
  main="Mean (99pc CI) LDST scores by LE and Gender")

# Compute summary statistics of the LDST score in the
# Age Group by LE by Gender combinations
Explore.LDST <- ExploreData(Dataset=Substitution,
   Model=LDST~Age_Group+LE+Gender)

# Plot the results
plot(Explore.LDST)
```

---

plot ICC *Graphical depiction of the ICC.*

Description

The ICC corresponds to the proportion of the total variance in the residuals that is accounted for by the clustering variable at hand (Kutner *et al.*, 2005). This function visualizes the extent ot which there is clustering in the dataset.

Usage

```
## S3 method for class 'ICC'
plot(x, X.Lab="Cluster", Y.Lab="Test score",
Main="", Add.Jitter=0.2, Size.Points=1, Size.Labels=1,
Add.Mean.Per.Cluster=TRUE, Col.Mean.Symbol="red", Seed=123,
...)
```

Arguments

x	A fitted object of class ICC.
X.Lab	The label that should be added to the X-axis. X.Lab="Cluster".
Y.Lab	The label that should be added to the Y-axis. Y.Lab="Test score".
Main	The title of the plot. Default Main=" ", i.e., no title.
Add.Jitter	The amount of jitter (random noise) that should be added in the horizontal direction (predicted scores, X-axis) of the plot. Adding a bit of jitter is useful to show the inidividual data points more clearly. The specified value Add.Jitter= in the function call determines the amount of jitter (range of values) that is added. For example, when Add.Jitter=0.2, a random value between -0.2 and 0.2 (sampled from a uniform) is added to the X-axis. Default Add.Jitter=0.2.
Size.Points	The size of the points in the plot. Default Size.Points=1.
Size.Labels	The size of the Labels of the X-axis in the plot. Default Size.Labels=1.
Add.Mean.Per.Cluster	Logical. Should the means per cluster be shown? Default Add.Mean.Per.Cluster=TRUE.
Col.Mean.Symbol	The color of the symbol that is used to indicate the mean (for each of the clusters). Default Col.Mean.Symbol="red".
Seed	The random seed that is used to add jitter. Default Seed=123.
...	Other arguments to be passed to the plot function.

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Kutner, M. H., Nachtsheim, C. J., Neter, J., and Li, W. (2005). *Applied linear statistical models* (5th edition). New York: McGraw Hill.

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R*. Springer Nature.

See Also

ICC

Examples

```
# Compute ICC in Substitution dataset, using Test.Administrator as
# clustering unit
data(Substitution)

# Add administrator to the dataset (just randomly allocate labels
# as Test.Administrator, so ICC should be approx. 0)
Substitution$Test.Adminstrator <- NA
Substitution$Test.Adminstrator <- sample(LETTERS[1:10],
  replace = TRUE, size = length(Substitution$Test.Adminstrator))
Substitution$Test.Adminstrator <-
  as.factor(Substitution$Test.Adminstrator)

ICC_LDST <- ICC(Cluster = Test.Adminstrator, Test.Score = LDST, Data = Substitution)

# Explore results
summary(ICC_LDST)
plot(ICC_LDST)

# Make points in the plot a bit larger and reduce
# the size of labels on the X-axis (initials test administrators)
plot(ICC_LDST, Size.Labels = .5, Size.Points=.5)
```

---

plot Stage.1 *Check the model assumptions for a fitted Stage 1 model graphically.*

Description

This function provides several plots that are useful to evaluate model assumptions. When the `plot()` function is applied to a fitted `Stage.1` object, three panels are generated. These panels show plots that can be used (i) to evaluate the homoscedasticity assumption, (ii) to evaluate the normality assumption, and (iii) to evaluate the presence of outliers.

Usage

```
## S3 method for class 'Stage.1'
plot(x, Homoscedasticity=TRUE, Normality=TRUE,
Outliers=TRUE, Assume.Homoscedasticity, Add.Jitter=0, Seed=123,
Confidence.QQ.Normality=.99, Plots.Together=TRUE,
Y.Lim.ResVarFunction, Group.Spec.Densities.Delta=FALSE, Main.Homosced.1,
Main.Homosced.2, Main.Norm.1, Main.Norm.2, Main.Norm.3, Main.Outliers,
cex.axis.homo=1, cex.main.homo=1, cex.lab.homo=1,
cex.axis.norm=1.6, cex.main.norm=1.5, cex.lab.norm=1.5,
cex.axis.outl=1, cex.main.outl=1, cex.lab.outl=1,
Color="red", Loess.Span=0.75, verbose=TRUE, ...)
```

Arguments

`x` A fitted object of class `Stage.1`.

`Homoscedasticity`
Logical. Should plots to evaluate homoscedasticity be shown?
Default `Homoscedasticity=TRUE`.

`Normality` Logical. Should plots to evaluate the normality assumption be shown? The normality plots are based on the standardized residuals in the normative dataset, which are computed as explained in the `Assume.Homoscedasticity=` argument documentation below. Default `Normality=TRUE`.

`Outliers` Logical. Should plots to evaluate outliers be shown? The outlier plot is based on the standardized residuals in the normative dataset, which are computed as explained in the `Assume.Homoscedasticity=` argument documentation below. Default `Outliers=TRUE`.

`Assume.Homoscedasticity`
By default, the standardized residuals $\widehat{\delta}_i$ that are shown in the normality and outlier plots are computed based on the overall residual standard error when the homoscedasticity assumption is valid (i.e., as $\widehat{\delta}_i = \frac{\widehat{\varepsilon}_i}{\widehat{\sigma}^2_\varepsilon}$, with $\widehat{\sigma}^2_\varepsilon$ corresponding to the overall residual standard error), or based on prediction-specific residual standard errors when the homoscedasticity assumption is invalid (i.e., as $\widehat{\delta}_i = \frac{\widehat{\varepsilon}_i}{\widehat{\sigma}^2_{\varepsilon_i}}$, with $\widehat{\sigma}^2_{\varepsilon_i}$ corresponding to e.g., a cubic polynomial variance prediction function $\widehat{\sigma}^2_{\varepsilon_i} = \widehat{\gamma}_0 + \widehat{\gamma}_1\,\widehat{Y} + \widehat{\gamma}_2\,\widehat{Y}^2 + \gamma_3\,\widehat{Y}^3$ when the mean structure of the model contains quantitiative independent variables). The default behaviour of the `plot()` function can be overruled using the `Assume.Homoscedasticity` argument. For example, when adding the argument `Assume.Homoscedasticity=TRUE` to the function call, the standardized residuals that are plotted will be computed based on the overall residual standard error (irrespective of the result of the Levene or Breusch-Pagan test).

`Add.Jitter` The amount of jitter (random noise) that should be added to the X-axis of the homoscedasticity plots (which show the model-predicted mean values). Adding a bit of jitter is useful to show the data more clearly (especially when there are only a few unique predicted values, e.g., when a binary or non-binary qualitative independent variable is considered in the mean structure of the model), i.e., to avoid overlapping data points. The specified value `Add.Jitter=` in the function call determines the amount of jitter (range of values) that is added. For example, when `Add.Jitter=0.1`, a random value between -0.1 and 0.1 (sampled from a uniform) is added to the predicted values in the homoscedasticity plots (shown on the X-axis). Default `Add.Jitter=0`, i.e., no jitter added to the predicted values in the homoscedasticity plots.

`Seed` The seed that is used when adding jitter. Default `Seed=123`.

`Confidence.QQ.Normality`
Specifies the desired confidence-level for the confidence band arond the line of perfect agreement/normality in the QQ-plot that is used to evaluate normality. Default `Confidence.QQ.Normality=0.95`. Use `Confidence.QQ.Normality=FALSE` if no confidence band is needed.

`Plots.Together` The different homoscedasticity and normality plots are grouped together in a panel by default. For example, the three normality plots are shown together in

one panel. If it is preferred to have the different plots in separate panels (rather than grouped to- gether), the argument `Plots.Together=FALSE` can be used. Default `Plots.Together=TRUE`.

`Y.Lim.ResVarFunction`
: The min, max limits of the Y-axis that should be used for the variance function plot. By default, the limit of the Y-axis is set between 0 and the maximum value of estimated variances multiplied by 2. This can be changed using the `Y.Lim.ResVarFunction` argument. For example, adding the argument `Y.Lim.ResVarFunction=c(0, 500)` sets the range of the Y-axis of the variance function plot from 0 to 500.

`Group.Spec.Densities.Delta`
: Logical. Should a plot with the group-specific densities of the standardized residuals be shown? Default `Group.Spec.Densities.Delta=FALSE`.

`Main.Homosced.1`
: The title of the first panel of the homoscedasticity plot (i.e., the scatterplot of the residuals against the predicted scores).

`Main.Homosced.2`
: The title of second panel of the homoscedasticity plot (i.e., the variance function plot).

`Main.Norm.1` The title of the first panel of the normality plot (i.e., the histogram of the standardized residuals).

`Main.Norm.2` The title of the second panel of the normality plot (i.e., the density of the standardized residuals and standard normal distribution).

`Main.Norm.3` The title of the third panel of the normality plot (i.e., the QQ-plot).

`Main.Outliers` The title of the outlier plot.

`cex.axis.homo` The magnification to be used for axis annotation of the homoscedasticity plots.

`cex.main.homo` The magnification to be used for the main label of the homoscedasticity plots.

`cex.lab.homo` The magnification to be used for the X- and Y-axis labels of the homoscedasticity plots.

`cex.axis.norm` The magnification to be used for axis annotation of the normality plots.

`cex.main.norm` The magnification to be used for the main label of the normality plots.

`cex.lab.norm` The magnification to be used for X and Y labels of the normality plots.

`cex.axis.outl` The magnification to be used for axis annotation of the outlier plot.

`cex.main.outl` The magnification to be used for the main label of the outlier plot.

`cex.lab.outl` The magnification to be used for X- and Y-axis labels of the outlier plot.

`Color` The color to be used for the Empirical Variance Function (EVF) and the standard normal distribution in the variance function plot and the normality plot that show the densities of the standardized residuals and the normal distribution, respectively. Default `Color="red"`.

`Loess.Span` The parameter α that determines the degree of smoothing of the EVF that is shown in the variance function plot. Default `Loess.Span=0.75`.

`verbose` A logical value indicating whether verbose output should be generated.

`...` Other arguments to be passed.

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

Examples

```
# Replicate the Stage 1 results that were obtained in
# Case study 1 of Chapter 4 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
data(GCSE)          # load the GCSE dataset

# Conduct the Stage 1 analysis
Model.1.GCSE <- Stage.1(Dataset=GCSE,
  Model=Science.Exam~Gender)

summary(Model.1.GCSE)
plot(Model.1.GCSE, Add.Jitter = .2)

# Use blue color for EVF and density normal distribution
plot(Model.1.GCSE, Add.Jitter = .2, Color="blue")

# Change the title of the variance function plot into
# "Variance function plot, residuals Science exam"
plot(Model.1.GCSE, Add.Jitter = .2,
  Main.Homosced.2 = "Variance function plot, residuals Science exam")

# Use a 95 percent CI around the line of perfect agreement in the
# QQ plot of normality
plot(Model.1.GCSE, Add.Jitter = .2,
     Confidence.QQ.Normality = .9)


# Replicate the Stage 1 results that were obtained in
# Case study 1 of Chapter 7 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset

# Add the variable Age.C (= Age centered) to the Substitution dataset
Substitution$Age.C <- Substitution$Age - 50

# Fit the final Stage 1 model
Substitution.Model.9 <- Stage.1(Dataset=Substitution,
```

```
   Alpha=0.005, Model=LDST~Age.C+LE,
   Order.Poly.Var=1) # Order.Poly.Var=1 specifies a linear polynomial
                     # for the variance prediction function

# Final Stage 1 model
summary(Substitution.Model.9)
plot(Substitution.Model.9)

# Request a variance function plot that assumes that
# the homoscedasticity assumption is valid
plot(Substitution.Model.9, Assume.Homoscedasticity = TRUE)
```

`plot Stage.2.NormScore`

Plot the results for a `Stage.2.NormScore` *object.*

Description

The function `Stage.2.NormScore()` is used to convert the raw test score of a tested person Y_0 into a percentile rank $\hat{\pi}_0$ (taking into account specified values of the independent variables). This function plots the results graphically. In particular, the density of the standard normal distribution is shown (when the normality assumption is valid for the fitted Stage 1 model), or the density of the standardized residuals in the normative sample (when the noormality assumption is not shown). The AUC between $-\infty$ and the tested person's standarized test score $\widehat{\delta}_i$ is shaded in grey, which visualizes the percentile rank that corresponds to the raw test score.

Usage

```
## S3 method for class 'Stage.2.NormScore'
plot(x, Main=" ", Both.CDFs=FALSE, xlim,
cex.axis=1, cex.main=1, cex.lab=1, ...)
```

Arguments

`x`	A fitted object of class `Stage.2.NormScore`.
`Main`	The title of the plot. Default `Main=" "`.
`Both.CDFs`	Should both the densities of the standard normal distribution and of the standardized residuals $\widehat{\delta}_i$ in the normative sample be shown in one plot? Default `Both.CDFs=FALSE`.
`xlim`	The limits for the X-axis. Default `xlim=c(-4,4)`.
`cex.axis`	The magnification to be used for axis annotation.
`cex.main`	The magnification to be used for the main label.
`cex.lab`	The magnification to be used for X and Y labels.
`...`	Extra graphical parameters to be passed to `plot()`.

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

Stage.2.NormScore

Examples

```
# Replicate the normative conversion that was obtained in
# Case study 1 of Chapter 3 in Van der Elst (2023)
# (science exam score = 30 obtained by a female)
# ------------------------------------------------------
library(NormData) # load the NormData package
data(GCSE)        # load the GCSE dataset

# Fit the Stage 1 model
Model.1.GCSE <- Stage.1(Dataset=GCSE,
  Model=Science.Exam~Gender)

# Stage 2: Convert a science exam score = 30 obtained by a
# female into a percentile rank (point estimate)
Normed_Score <- Stage.2.NormScore(Stage.1.Model=Model.1.GCSE,
  Score=list(Science.Exam=30, Gender="F"))

summary(Normed_Score)
plot(Normed_Score)


# Replicate the normative conversion that was obtained in
# Case study 1 of Chapter 7 in Van der Elst (2023)
# (LDST score = 40 obtained by a 20-year-old
# test participant with LE=Low)
# ------------------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset

# Make the new variable Age.C (= Age centered) that is
# needed to fit the final Stage 1 model,
# and add it to the Substitution dataset
Substitution$Age.C <- Substitution$Age - 50

# Fit the final Stage 1 model
```

```
Substitution.Model.9 <- Stage.1(Dataset=Substitution,
  Alpha=0.005, Model=LDST~Age.C+LE, Order.Poly.Var=1)
summary(Substitution.Model.9)

# Convert an LDST score = 40 obtained by a
# 20-year-old test participant with LE=Low
# into a percentile rank (point estimate)
Normed_Score <- Stage.2.NormScore(
   Stage.1.Model=Substitution.Model.9,
   Score=list(LDST=40, Age.C=20-50, LE = "Low"))

summary(Normed_Score)
plot(Normed_Score)
```

`plot Tukey.HSD` *Plot the results of Tukey's Honest Significance Difference test.*

Description

This function plots the results of Tukey's Honest Significance Difference (HSD; Tukey, 1949) test that allows for making post hoc comparisons of the group means. Tukey's HSD can only be conducted when the mean structure of the Stage 1 model only contains qualitative independent variables (i.e., when the fitted regression model is essentially an ANOVA).

Usage

```
## S3 method for class 'Tukey.HSD'
plot(x, ...)
```

Arguments

`x`	A fitted object of class `Tukey.HSD`.
`...`	Extra graphical parameters to be passed to `plot()`.

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Tukey, J. (1949). Comparing individual means in the Analysis of Variance. *Biometrics, 5,* 99-114.

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

Tukey.HSD

Examples

```
data(Personality)
Model.Openness <- Stage.1(Dataset = Personality, Model = Openness ~ LE)
# conduct post hoc comparisons for the levels of education
Tukey.Openness <- Tukey.HSD(Model.Openness)
summary(Tukey.Openness)
plot(Tukey.Openness)

# conduct post hoc comparisons for the levels of education by education combinations
data(Substitution)
Model.Substitution <- Stage.1(Dataset = Substitution, Model = LDST ~ LE*Gender)
Tukey.Substitution <- Tukey.HSD(Model.Substitution)
summary(Tukey.Substitution)
plot(Tukey.Substitution)
```

Plot.Scatterplot.Matrix

Explore data

Description

The function `Plot.Scatterplot.Matrix()` makes a scatterplot matrix of the specified variables.

Usage

```
Plot.Scatterplot.Matrix(Dataset, Variables,
Add.Jitter=0.1, Seed=123, ...)
```

Arguments

`Dataset`	The name of the dataset.
`Variables`	The names of the variables that should be shown in the scatterplot matrix.
`Add.Jitter`	The amount of jitter (random noise) that should be added to the variables in the scatterplot matrix. Adding a bit of jitter is useful to show the inidividual data points more clearly, especially if several qualitative variables are added in the plot. The specified value `Add.Jitter=` in the function call determines the amount of jitter (range of values) that is added. For example, when `Add.Jitter=0.1`, a random value between -0.1 and 0.1 (sampled from a uniform distribution) is added to the datapoints. Default `Add.Jitter=0.1`.
`Seed`	The seed that is used when adding jitter. Default `Seed=123`.
`...`	Extra graphical parameters to be passed to `plot()`.

Details

For details, see Van der Elst (2023).

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

Examples

```
data(Substitution)

# Make a scatterplot matrix with the variables LDST,
# Age, Gender and LE in the Substitution dataset
Plot.Scatterplot.Matrix(Dataset = Substitution,
Variables = c("LDST", "Age", "Gender", "LE"))
```

PlotFittedPoly *Explore data*

Description

The function PlotFittedPoly fits polynomials of a specified order to the data.

Usage

```
PlotFittedPoly(Dataset, Test.Score, IV, Center.Value.IV=0,
Order.Polynomial=3, Confidence.Band.Poly=FALSE, Alpha=.01,
EMF = TRUE, Confidence.Band.EMF=TRUE,
xlab, ylab, Color = "red", Black.white=FALSE,
Legend.Location="topright", Legend.text.size=1,
Add.Jitter=0, Seed=123, cex.axis=1, cex.main=1,
cex.lab=1, Loess.Span=0.75, ...)
```

Arguments

Dataset	The name of the dataset.
Test.Score	The name of the test score.
IV	The name of the independent variable.
Center.Value.IV	The constant that is subtracted from the independent variable. Default Center.Value.IV=0.
Order.Polynomial	The order of the polynomials to be fitted. By default, Order.Polynomial=3 and thus a cubic polynomial is fitted. If no polynomial has to be plotted, the argument Order.Polynomial="None" can be used.
Confidence.Band.Poly	Logical. Should a confidence band around the prediction function of the polynomial model be added to the plot? Default Confidence.Band.Poly=FALSE.
Alpha	The Alpha-level of the confidence band(s) for the polynomial and/or loess models. Default Alpha=0.01 and thus a 99% confidence band is fitted.
EMF	Logical. Should the EMF be added to the plot? Default EMF=TRUE.
Confidence.Band.EMF	Logical. Should a confidence band around the prediction function of the loess model be added to the plot? Default Confidence.Band.EMF=TRUE.
xlab	The label that should be added to the X-axis. Default xlab="IV"
ylab	The label that should be added to the Y-axis. Default ylab="Test score".
Color	The color to be used for the fitted EMF. Default Color = "red".
Black.white	Logical. Should the plot be in black and white (rather than in color)? Default Black.white=FALSE.
Legend.Location	The location of the legend. Default Legend.Location="topright". If no legend is needed, the argument Legend.Location="None" can be used.
Legend.text.size	The size of the text of the label for IV2. Default Legend.text.size=1.
Add.Jitter	The amount of jitter (random noise) that should be added to the test score. Adding a bit of jitter is useful to show the data more clearly, i.e., to avoid overlapping data points. The specified value Add.Jitter= in the function call determines the amount of jitter (range of values) that is added. For example, when Add.Jitter=0.1, a random value between -0.1 and 0.1 (sampled from a uniform) is added to the test scores. Default Add.Jitter=0, i.e., no jitter added to the predicted values in the homoscedasticity plot.
Seed	The seed that is used when adding jitter. Default Seed=123.
cex.axis	The magnification to be used for axis annotation.
cex.main	The magnification to be used for the main label.
cex.lab	The magnification to be used for X and Y labels.
Loess.Span	The parameter α that determines the degree of smoothing of the Empirical Variance Function. Default Loess.Span=0.75.
...	Extra graphical parameters to be passed to plot().

Details

For details, see Van der Elst (2023).

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

Examples

```
data(Substitution)

# plot of linear, quadratic and cubic polynomials relating age
# to the LDST test score
PlotFittedPoly(Dataset = Substitution, Test.Score = LDST, IV = Age,
Order.Polynomial = 1, Center.Value.IV = 50)

PlotFittedPoly(Dataset = Substitution, Test.Score = LDST, IV = Age,
Order.Polynomial = 2, Center.Value.IV = 50)

PlotFittedPoly(Dataset = Substitution, Test.Score = LDST, IV = Age,
Order.Polynomial = 3, Center.Value.IV = 50)
```

Sandwich *Sandwich estimators for standard errors*

Description

The `Sandwich()` function can be used to obtain heteroscedasticity-consistent standard errors of the regression parameters of a fitted Stage 1 model. These are used to account for heteroscedasticity.

Usage

```
Sandwich(Stage.1.Model, Type="HC0")
```

Arguments

`Stage.1.Model`	The fitted stage 1 model for which heteroscedasticity-consistent standard errors (sandwich estimators) for the standard errors of the regression parameters has to be provided.
`Type`	The type of the heteroscedasticity-consistent estimator that is used. By default, White's (White, 1980) estimator is used (i.e., `Type="HC0"`) but other estimators are available. For details, see the `vcovHC` function of the `sandwich` package.

Value

`Sandwich`	The fitted Stage 1 model with sandwich estimators.
`Alpha`	The significance level that is used for inference. Default `Alpha=0.05`.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

White, H. (1980). A heteroscedasticity-consistent covariance matrix and a direct test for heteroscedasticity. Econometrica, 48, 817-838.

See Also

Stage.1

Examples

```
data(GCSE)
Model.1.GCSE <- Stage.1(Dataset = GCSE, Model = Science.Exam~Gender)
Sandwich(Stage.1.Model = Model.1.GCSE)
```

`Stage.1` *Stage 1 of the regression-based normative analysis*

Description

The function `Stage.1` fits a regression model with the specified mean and residual variance components, and conducts several model checks (homoscedasticity, normality, absence of outliers, and multicollinearity) that are useful in a setting where regression-based normative data have to be established.

Usage

```
Stage.1(Dataset, Model, Order.Poly.Var=3,
Alpha=0.05, Alpha.Homosc=0.05, Alpha.Norm = .05,
Assume.Homoscedasticity=NULL,
Test.Assumptions=TRUE, Outlier.Cut.Off=4,
Show.VIF=TRUE, GVIF.Threshold=10, Sandwich.Type="HC0",
Alpha.CI.Group.Spec.SD.Resid=0.01)
```

Arguments

`Dataset`	A `data.frame` that should consist of one line per test participant (the so-called 'wide' data-format). Each line should contain (at least) one test score and one independent variable.
`Model`	The regression model to be fitted (mean structure). A formula should be provided using the syntaxis of the `lm` function (for help, see `?lm`). For example, `Test.Score~Gender` will fit a linear regression model in which `Gender` (the independent variable) is regressed on `Test.Score`. `Test.Score~Gender+Age+Gender:Age` will regress `Test.Score` on `Gender`, `Age`, and the interaction term. `Test.Score~1` will fit an intercept-only model.
`Order.Poly.Var`	If the homoscedasticity assumption is violated and the mean structure of the fitted model contains at least one quantitative variable, a polynomial variance prediction function is fitted. The argument `Order.Poly.Var=` determines the order of the polynomial, e.g., `Order.Poly.Var=1`, `Order.Poly.Var=2`, `Order.Poly.Var=3` for linear, quadratic and cubic polynomials, respectively. By default, `Order.Poly.Var = 3`.
`Alpha`	The significance level to be used when conducting inference for the mean structure of the model. Default `Alpha=0.05`.
`Alpha.Homosc`	The significance level to be used to evaluate the homoscedasticity assumption based on the Levene test (when all independent variables in the model are qualitative) or the Breusch-Pagan test (when at least one of the independent variables is quantitative). Default `Alpha.Homosc=0.05`.
`Alpha.Norm`	The significance level to be used to test the normality assumption for the standardized errors using the Shapiro-Wilk test. The normality assumption is evaluated based on the standardized residuals in the normative dataset, which are computed as explained in the `Assume.Homoscedasticity=` argument documentation below. Default `Alpha.Shapiro=0.05`.
`Assume.Homoscedasticity`	Logical. The `NormData` package 'decides' whether the homoscedasticity assumption is valid based on the Levene or Breusch-Pagan tests (for models that only include qualitative independent variables versus models that include at least one quantitative independent variable, respectively). The `Assume.Homoscedasticity=TRUE/FALSE` argument can be used to overrule this decision process and 'force' the `NormData` package to assume or not assume homoscedasticity. When the argument `Assume.Homoscedasticity=TRUE` is used, the argument `Alpha.Homosc=0` is automatically used in the `Stage.1()` function call and thus the homoscedasticity

assumption will never be rejected (because the p-value of the Levene or Breusch-Pagan test-statistics will always be larger than the specified $\alpha = 0$). When `Assume.Homoscedasticity=FALSE` is used, the argument `Alpha.Homosc=1` is automatically used thus the homoscedasticity assumption will always be rejected (because the p-value of the Levene or Breusch-Pagan test-statistics will always be smaller than the specified $\alpha = 1$).

By default, the standardized residuals $\widehat{\delta}_i$ that are shown in the normality and outlier output sections of the results (and the plots, see `plot Stage.1`) are computed based on the overall residual standard error when the homoscedasticity assumption is valid (i.e., as $\widehat{\delta}_i = \frac{\widehat{\varepsilon}_i}{\widehat{\sigma}^2_\varepsilon}$, with $\widehat{\sigma}^2_\varepsilon$ corresponding to the overall residual standard error), or based on prediction-specific residual standard errors when the homoscedasticity assumption is invalid (i.e., as $\widehat{\delta}_i = \frac{\widehat{\varepsilon}_i}{\widehat{\sigma}^2_{\varepsilon_i}}$, with $\widehat{\sigma}^2_{\varepsilon_i}$ corresponding to e.g., a cubic polynomial variance prediction function $\widehat{\sigma}^2_{\varepsilon_i} = \widehat{\gamma}_0 + \widehat{\gamma}_1\,\widehat{Y} + \widehat{\gamma}_2\,\widehat{Y}^2 + \gamma_3\,\widehat{Y}^3$ when the mean structure of the model contains quantitiative independent variables).

`Test.Assumptions`

Logical. Should the model assumptions be evaluated for the specified model? Default `Test.Assumptions=TRUE`.

`Outlier.Cut.Off`

Outliers are evaluated based on the standardized residuals, which are computed as explained in the `Assume.Homoscedasticity=` argument documentation. The `Outlier.Cut.Off=` argument specifies the absolute value that is used as a threshold to detect outliers. Default `Outlier.Cut.Off=4`, so test scores with standardized residuals < -4 or > 4 are flagged as outliers.

`Show.VIF` Logical. Should the generalized VIF (Fox and Monette, 1992) be shown when the function `summary()` is applied to the fitted object? Default `Show.VIF=TRUE`. If all names of the independent variables in the fitted Stage 1 model contain the string 'Age' (e.g., Age, Age.2 and Age.3), a higher-order polynomial model for the mean structure is being fitted. For such models, multicollinearity diagnostics are essentially irrelevant (see Van der Elst, 2023) and in such cases the generalized VIF is not printed in the summary output. The generalized VIF is also not shown whenn there is only one independent variable in the model (because multicollinearity relates to the linear association of two or more independent variables).

`GVIF.Threshold` The threshold value to be used to detect multicollinearity based on the generalized VIF. Default `GVIF.Threshold=10`.

`Sandwich.Type` When the homoscedasticity assumption is violated, so-called sandwich estimators (or heteroscedasticity-consistent estimators) for the standard errors of the regression parameters are used. For example, the sandwich estimator for the standard error of $\widehat{\beta}_1$ in a simple linear regression model corresponds to $\widehat{\sigma}_{\beta_1} = \sqrt{\frac{\sum_{i=1}^{N}\left(\left(X_i - \widehat{\mu}_{X_i}\right)^2 \widehat{\varepsilon}_i^2\right)}{\left(\sum_{i=1}^{N}(X_i - \widehat{\mu}_{X_i})^2\right)^2}}$. For multiple linear regression models, the sandwich estimators for the different independent variables $\widehat{\sigma}_{\beta_0}$, $\widehat{\sigma}_{\beta_1}$, ...correspond to the square roots of the diagonal elements of $\widehat{\boldsymbol{\Sigma}}_\beta =$

$\left(\boldsymbol{X}^{'}\boldsymbol{X}\right)^{-1}\left(\boldsymbol{X}^{'}\begin{bmatrix}\widehat{\varepsilon}_1^2 & 0 & \dots & 0\\ 0 & \widehat{\varepsilon}_2^2 & \dots & 0\\ \vdots & \vdots & \ddots & 0\\ 0 & 0 & 0 & \widehat{\varepsilon}_N^2\end{bmatrix}\boldsymbol{X}\right)\left(\boldsymbol{X}^{'}\boldsymbol{X}\right)^{-1}$. The sandwich-estimators that are shown in the above expressions are referred to as the Heteroscedasticity-Consistent 0 estimator (or HC0 estimator), which is the first sandwich-estimator that was proposed in the literature. The HC0 sandwich-estimator is justified based on asymptotic theory, and its application thus requires large sample sizes. For smaller sample sizes of $N < 250$, the use of the HC3 estimator is recommended because the HC0 sandwich-estimator tends to be negatively biased (Long and Erwin, 2000). By default, the HC0 estimator is used. The argument `Sandwich.Type=` can be used to request another type of the heteroscedasticity-consistent estimator. For details on these estimators, see the `vcovHC` function of the `sandwich` package. If $N < 250$ and the homoscedasticity assumption is violated, a note will be given that the use of the HC3-estimator is recommended. To this end, the argument `Sandwich.Type="HC3"` can be added in the `Stage.1()` function call.

`Alpha.CI.Group.Spec.SD.Resid`
: The α-level to be used for the CIs around the prediction-specific residual standard errors (when the homoscedasticity assumption is invalid and the model only contains qualitative independent variable). These CIs are used in the variance function plot. Default `Alpha.CI.Group.Spec.SD.Resid=0.01`.

Details

For details, see Van der Elst (2023).

Value

An object of class `Stage.1` with components,

`HomoNorm`	The fitted regression model assuming homoscedasticity and normality.
`NoHomoNorm`	The fitted regression model assuming no homoscedasticity and normality.
`HomoNoNorm`	The fitted regression model assuming homoscedasticity and no normality.
`NoHomoNoNorm`	The fitted regression model assuming no homoscedasticity and no normality.
`Predicted`	The predicted test scores based on the fitted model.
`Sandwich.Type`	The requested sandwich estimator.
`Order.Poly.Var`	The order of the polynomial variance prediction function.

Author(s)

Wim Van der Elst

References

Fox, J. and Monette, G. (1992). Generalized collinearity diagnostics. *JASA, 87,* 178-183.

Long, J. S. and Ervin, L. H. (2000). Using Heteroscedasticity Consistent Standard Errors in the Linear Regression Model. *The American Statistician, 54,* 217-224.

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

plot Stage.1, Stage.2.AutoScore, Stage.2.NormScore, Stage.2.NormTable

Examples

```
# Replicate the Stage 1 results that were obtained in
# Case study 1 of Chapter 4 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
data(GCSE)          # load the GCSE dataset

# Conduct the Stage 1 analysis
Model.1.GCSE <- Stage.1(Dataset=GCSE,
    Model=Science.Exam~Gender)

summary(Model.1.GCSE)
plot(Model.1.GCSE)


# Replicate the Stage 1 results that were obtained in
# Case study 1 of Chapter 7 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset

# Add the variable Age.C (= Age centered) and its
# quadratic and cubic terms to the Substitution dataset
Substitution$Age.C <- Substitution$Age - 50
Substitution$Age.C2 <- (Substitution$Age - 50)**2
Substitution$Age.C3 <- (Substitution$Age - 50)**3

# Fit the full Stage 1 model
Substitution.Model.1 <- Stage.1(Dataset=Substitution,
   Model=LDST~Age.C+Age.C2+Age.C3+Gender+LE+Age.C:LE+
   Gender:LE+Age.C:Gender, Alpha=0.005)
summary(Substitution.Model.1)

# Fit the model in which the non-significant Age.C:Gender
# interaction term is removed
Substitution.Model.2 <- Stage.1(Dataset=Substitution,
    Alpha=0.005,
    Model=LDST~Age.C+Age.C2+Age.C3+Gender+LE+
    Age.C:LE+Gender:LE)
summary(Substitution.Model.2)

# Evaluate the significance of the Gender:LE interaction term
# GLT is used because the interaction involves multiple regression
# parameters
GLT.1 <- GLT(Dataset=Substitution, Alpha=0.005,
```

```
   Unrestricted.Model=LDST~Age.C+Age.C2+Age.C3+
      Gender+LE+Age.C:LE+Gender:LE,
   Restricted.Model=LDST~Age.C+Age.C2+Age.C3+
      Gender+LE+Age.C:LE)
summary(GLT.1)

# Fit the model in which the non-significant Gender:LE
# interaction term is removed
Substitution.Model.3 <- Stage.1(Dataset=Substitution,
    Alpha=0.005,
    Model=LDST~Age.C+Age.C2+Age.C3+Gender+LE+Age.C:LE)
summary(Substitution.Model.3)

# Evaluate the significance of the Age:LE interaction
# using the General Linear Test framework
GLT.2 <- GLT(Dataset=Substitution,
    Unrestricted.Model=LDST~Age.C+Age.C2+Age.C3+Gender+LE+Age.C:LE,
    Restricted.Model=LDST~Age.C+Age.C2+Age.C3+Gender+LE, Alpha=0.005)
summary(GLT.2)

# Fit the model in which the non-significant Age_c:LE
# interaction term is removed
Substitution.Model.4 <- Stage.1(Dataset=Substitution,
   Alpha=0.005, Model=LDST~Age.C+Age.C2+Age.C3+Gender+LE)
summary(Substitution.Model.4)

# Fit the model in which the non-significant Age.C3 term is removed
Substitution.Model.5 <- Stage.1(Dataset=Substitution,
   Alpha=0.005, Model=LDST~Age.C+Age.C2+Gender+LE)
summary(Substitution.Model.5)

# Fit the model in which the non-significant Age.C2 term is removed
Substitution.Model.6 <- Stage.1(Dataset=Substitution,
   Alpha=0.005, Model=LDST~Age.C+Gender+LE)
summary(Substitution.Model.6)

# Fit the model in which the non-significant main effect of Gender
# is removed
Substitution.Model.7 <- Stage.1(Dataset=Substitution,
  Alpha=0.005, Model=LDST~Age.C+LE)
summary(Substitution.Model.7)
plot(Substitution.Model.7, Normality = FALSE, Outliers = FALSE)

# Check the significance of LE using the GLT framework
GLT.3 <- GLT(Dataset=Substitution, Alpha=0.005,
    Unrestricted.Model=LDST~Age.C+LE,
    Restricted.Model=LDST~Age.C)
summary(GLT.3)

# Residual variance function. Substitution.Model.7 uses
# a cubic polynomial variance prediction function.
# Remove cubic Pred.Y term from Substitution.Model.7, so
# fit quadratic variance prediction function
```

```
Substitution.Model.8 <- Stage.1(Dataset=Substitution,
    Alpha=0.005, Model=LDST~Age.C+LE,
    Order.Poly.Var=2)  # Order.Poly.Var=2 specifies a quadratic polynomial
                       # for the variiance prediction function
summary(Substitution.Model.8)
plot(Substitution.Model.8, Normality = FALSE, Outliers = FALSE)

# Remove quadratic Pred.Y term, so fit linear variance
# prediction function
Substitution.Model.9 <- Stage.1(Dataset=Substitution,
    Alpha=0.005, Model=LDST~Age.C+LE,
    Order.Poly.Var=1) # Order.Poly.Var=1 specifies a linear polynomial
                      # for the variiance prediction function

# Final Stage 1 model
summary(Substitution.Model.9)
plot(Substitution.Model.9)
```

`Stage.2.AutoScore` *Make an automatic scoring sheet*

Description

This function is useful to construct an automatic scoring sheet that implements the Stage 2 normative conversion approach in a spreadsheet. In particular, a spreadsheet will be created with three tabs that should be copy-pasted to the different sections of the `Model details` tab of the template file. For details, see Van der Elst (2023).

Usage

```
Stage.2.AutoScore(Stage.1.Model, Assume.Homoscedasticity,
  Assume.Normality, Folder, NameFile="NormSheet.xlsx",
  verbose=TRUE)
```

Arguments

`Stage.1.Model` A fitted object of class `Stage.1` that should be written to the Excel sheet (i.e., the final Stage 1 model).

`Assume.Homoscedasticity`
Logical. Should homoscedasticity be assumed? By default, homoscedasticity is assumed when the p-value of the Levene or Breusch-Pagan test for the fitted Stage 1 model is above the specified α-level in the `Stage.1()` function call. When homoscedasticity is assumed, an overall residual standard error is written to the spreadsheet. When homoscedasticity is not assumed, prediction-specific residual standard errors are written to the spreadsheet. The default decision procedure can be overruled by means of the arguments `Assume.Homoscedasticity=TRUE` or `Assume.Homoscedasticity=FALSE`.

`Assume.Normality`
: Logical. Should normality of the standardized errors be assumed? By default, normality is assumed when the p-value of the Shapiro-Wilk test for the fitted Stage 1 model is above the specified α-level in the `Stage.1()` function call. When normality is assumed, the CDF of the standard normal distribution is written to the spreadsheet. When normality is not assumed, the CDF of the standardized residuals in the normative sample is written to the speeadsheet. The default decision procedure can be overruled by means of the arguments argument `Assume.Normality=TRUE` or `Assume.Normality=FALSE`.

`Folder`
: The folder where the spreadsheet file should be saved.

`NameFile`
: The name of the file in which the normative tables should be saved. Default `NameFile="NormTable.xlsx"`

`verbose`
: A logical value indicating whether verbose output should be generated.

Details

For details, see Van der Elst (2023).

Value

An object of class `Stage.2.AutoScore` with components,

`Mean.Structure`
: The mean prediction function.

`Residual.Structure`
: The variance prediction function.

`Percentiles.Delta`
: A table of the standardized residuals and their corresponding estimated percentile ranks (based on the CDF of the standard normal distribution or the CDF of the standardized residuals in the normative sample, see above).

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

`Stage.1`, `Stage.2.NormTable`, `Stage.2.AutoScore`

Examples

```
# Replicate the Stage 1 results that were obtained in
# Case study 1 of Chapter 4 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
```

```
data(GCSE)          # load the GCSE dataset

# Conduct the Stage 1 analysis
Model.1.GCSE <- Stage.1(Dataset=GCSE,
  Model=Science.Exam~Gender)

summary(Model.1.GCSE)
plot(Model.1.GCSE, Add.Jitter = .2)

# Write the results to a spreadsheet file
Stage.2.AutoScore(Stage.1.Model=Model.1.GCSE,
  Folder=tempdir(),   # Replace tempdir() by the desired folder
  NameFile="GCSE.Output.xlsx")

# Copy-paste the information in GCSE.Output.xlsx to the
# template file, as detailed in Van der Elst (2023)


# Replicate the Stage 1 results that were obtained in
# Case study 1 of Chapter 7 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset

# Add the variable Age.C (= Age centered) to the Substitution dataset
Substitution$Age.C <- Substitution$Age - 50

# Fit the final Stage 1 model
Substitution.Model.9 <- Stage.1(Dataset=Substitution,
   Alpha=0.005, Model=LDST~Age.C+LE, Order.Poly.Var=1)

# Final Stage 1 model
summary(Substitution.Model.9)
plot(Substitution.Model.9)

# Write the results to a spreadsheet file
Stage.2.AutoScore(Stage.1.Model=Substitution.Model.9,
   Folder=tempdir(),  # Replace tempdir() by the desired folder
   NameFile="LDST.Output.xlsx")

# Copy-paste the information in LDST.Output.xlsx to the
# template file, as detailed in Van der Elst (2023)
```

`Stage.2.NormScore` *Convert a raw score to a percentile rank*

Description

The function `Stage.2.NormScore()` can be used to convert the raw test score of a tested person Y_0 into a percentile rank $\hat{\pi}_0$ (taking into account specified values of the independent variables).

Usage

```
Stage.2.NormScore(Stage.1.Model, Assume.Homoscedasticity,
Assume.Normality, Score, Rounded=TRUE)
```

Arguments

`Stage.1.Model`	A fitted object of class `Stage.1` that should be used to conduct the normative conversions.
`Assume.Homoscedasticity`	Logical. Should homoscedasticity be assumed in conducting the normative conversion? By default, homoscedasticity is assumed when the p-value of the Levene or Breusch-Pagan test for the fitted Stage 1 model is above the specified α-level in the `Stage.1()` function call. When homoscedasticity is assumed, an overall residual standard error is used in the normative conversions. When homoscedasticity is not assumed, prediction-specific residual standard errors used. The default decision procedure can be overruled by means of the arguments argument `Assume.Homoscedasticity=TRUE` or `Assume.Homoscedasticity=FALSE`.
`Assume.Normality`	Logical. Should normality of the standardized errors be assumed in conducting the normative conversion? By default, normality is assumed when the p-value of the Shapiro-Wilk test for the fitted Stage 1 model is above the specified α-level in the `Stage.1()` function call. When normality is assumed, the Y_0 to $\hat{\pi}_0$ conversion is based on the CDF of the standard normal distribution. When normality is not assumed, this conversion is based on the CDF of the standardized residuals in the normative sample. The default decision procedure can be overruled by means of the arguments argument `Assume.Normality=TRUE` or `Assume.Normality=FALSE`.
`Score`	A `list` that contains the test score Y_0 to be converted into a percentile rank and the values for the relevant independent variable(s). For example, the argument `Score=list(Science.Exam=30, Gender="F")` specifies that a female student obtained a raw Science Exam score Y_0. Observe that quotes are used to refer to a female student (i.e., "F"). This is done because the string `F` (without quotes) is shorthand notation for the logical indicator `FALSE` in R. If no quotes are used, an error will be generated that a logical indicator was provided where a factor level was expected. To avoid such issues, it is recommended to always use quotes to refer to the levels of a factor. In the `Score=...` argument, the test score should always be specified first followed by the independent variable. Notice that both the name of the independent variable and the coding scheme that is specified in the `Score=...` argument should correspond to the name of the independent variable and the original coding scheme that was used in the `Stage.1()` function call. For example, if the variable name `Gender` original coding scheme `F` and `M` was used in the `Stage.1()` function call, the same should be done in the `Stage.2.NormScore()` call. Thus `Score=list(Science.Exam=30, Gender="F")` should be used, and not e.g., `Score=list(Science.Exam=30, GenderM=0)`.
`Rounded`	Logical. Should the percentile rank be rounded to a whole number? Default `Rounded=TRUE`.

Details

For details, see Van der Elst (2023).

Value

An object of class `Stage.2.NormScore` with components,

`Fitted.Model` A fitted object of class `Stage.1()` that was used to convert the raw test score Y_0 into a percentile rank $\hat{\pi}_0$.

`Results` A data frame that contains the observed test score, residuals, percentile rank, ...

`Assume.Homoscedasticity` The homoscedasticity assumption that was made in the normative conversion.

`Assume.Normality` The normality assumption that was made in the normative conversion.

`Score` The test score and the value(s) of the independent variable(s) that were used in the computations.

`Stage.1.Model` The `Stage.1.Model` model used in the analysis.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

`Stage.2.NormTable`, `Stage.2.AutoScore`, `Bootstrap.Stage.2.NormScore`

Examples

```
# Replicate the normative conversion that was obtained in
# Case study 1 of Chapter 3 in Van der Elst (2023)
# (science exam score = 30 obtained by a female)
# ------------------------------------------------------
library(NormData) # load the NormData package
data(GCSE)        # load the GCSE dataset

# Fit the Stage 1 model
Model.1.GCSE <- Stage.1(Dataset=GCSE,
  Model=Science.Exam~Gender)

# Stage 2: Convert a science exam score = 30 obtained by a
# female into a percentile rank (point estimate)
Normed_Score <- Stage.2.NormScore(Stage.1.Model=Model.1.GCSE,
  Score=list(Science.Exam=30, Gender="F"))
```

```
summary(Normed_Score)
plot(Normed_Score)


# Replicate the normative conversion that was obtained in
# Case study 1 of Chapter 7 in Van der Elst (2023)
# (LDST score = 40 obtained by a 20-year-old
# test participant with LE=Low)
# ------------------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset

# Make the new variable Age.C (= Age centered) that is
# needed to fit the final Stage 1 model,
# and add it to the Substitution dataset
Substitution$Age.C <- Substitution$Age - 50

# Fit the final Stage 1 model
Substitution.Model.9 <- Stage.1(Dataset=Substitution,
  Alpha=0.005, Model=LDST~Age.C+LE, Order.Poly.Var=1)
summary(Substitution.Model.9)

# Convert an LDST score = 40 obtained by a
# 20-year-old test participant with LE=Low
# into a percentile rank (point estimate)
Normed_Score <- Stage.2.NormScore(
  Stage.1.Model=Substitution.Model.9,
  Score=list(LDST=40, Age.C=20-50, LE = "Low"))

summary(Normed_Score)
plot(Normed_Score)
```

`Stage.2.NormTable` *Derive a normative table*

Description

This function allows for deriving a normative table that shows percentile ranks $\hat{\pi}_0$ that correspond to a wide range of raw test scores Y_0 (stratified by the relevant independent variables).

Usage

```
Stage.2.NormTable(Stage.1.Model, Assume.Homoscedasticity,
Assume.Normality, Grid.Norm.Table, Test.Scores, Digits=6,
Rounded=TRUE)
```

Arguments

`Stage.1.Model`	A fitted object of class `Stage.1` that should be used to derive the normative table.

`Assume.Homoscedasticity`

Logical. Should homoscedasticity be assumed when deriving the normative table? By default, homoscedasticity is assumed when the p-value of the Levene or Breusch-Pagan test for the fitted Stage 1 model is above the specified α-level in the `Stage.1()` function call. When homoscedasticity is assumed, an overall residual standard error is used in the derivation of the normative table. When homoscedasticity is not assumed, prediction-specific residual standard errors used. The default decision procedure can be overruled by means of the arguments argument `Assume.Homoscedasticity=TRUE` or `Assume.Homoscedasticity=FALSE`.

`Assume.Normality`

Logical. Should normality of the standardized errors be assumed when deriving the normative table? By default, normality is assumed when the p-value of the Shapiro-Wilk test for the fitted Stage 1 model is above the specified α-level in the `Stage.1()` function call. When normality is assumed, the Y_0 to $\hat{\pi}_0$ conversions in the normative table are based on the CDF of the standard normal distribution. When normality is not assumed, these conversions are based on the CDF of the standardized residuals in the normative sample. The default decision procedure can be overruled by means of the arguments argument `Assume.Normality=TRUE` or `Assume.Normality=FALSE`.

`Grid.Norm.Table`

A `data.frame` that specifies the name of the independent variable(s) (e.g., `Gender`) and the levels (e.g., `"F"` and `"M"`) or values (e.g., `Age.C=seq(from=20, to=80, by=1)-50`)) for which the estimated percentile ranks should be tabulated. Both the name of the independent variable and the coding scheme that is specified in the `Grid.Norm.Table=...` argument should exactly match the name of the independent variable and the original coding scheme that was used in the `Stage.1()` function call. For example, if the variable name `Gender` with original coding scheme `F` and `M` was used in the `Stage.1()` function call, the same should be done in the `Stage.2.NormTable()` function call. So `Grid.Norm.Table= data.frame(Gender=c("F", "M"))` should be used, and not e.g., `Grid.Norm.Table= data.frame(GenderM=c(0,1))`. Observe that quotes are used to refer to a female student (i.e., "F"). This is done because the string `F` (without quotes) is shorthand notation for the logical indicator `FALSE` in R. If no quotes are used, an error will be generated that a logical indicator was provided where a factor level was expected.

When multiple independent variables are considered, the `data.frame` can be constructed using the `expand.grid()` function. For example, `Grid.Norm.Table= expand.grid(Age.C=seq(from=-30, to=30, by=1), LE=c("Low", "Average", "High"))` specifies that the normative table should be stratified for both Age centered (with score range -30 to 30) and LE.

`Test.Scores` A vector that specifies the raw test scores that should be shown in the normative table.

`Rounded` Logical. Should the percentile ranks that are shown in the normative table be rounded to a whole number? Default `Rounded=TRUE`.

`Digits` The number of digits that need to be shown in the normative table for the predicted means and residual standard errors. Default `Digits=6`.

Details

For details, see Van der Elst (2023).

Value

An object of class `Stage.2.NormTable` with components,

`Norm.Table` The normative table.

`Group.Specific.SD.Resid`
Logical. Where prediction-specific SDs of the residuals used?

`Empirical.Dist.Delta`
Logical. Was the CDF of the standardized residuals used to convert the raw test scores into percentile ranks?

`N.Analysis` The sample size of the analyzed dataset.

`Test.Scores` A vector of raw test scores for which percentile ranks were requested.

`Assume.Homoscedasticity`
Is homoscedasticity assumed in the computation of the normative data?

`Assume.Normality`
Is normality assumed in the computation of the normative data?

`Stage.1.Model` The `Stage.1.Model` model that was used to do the computations.

`Grid.Norm.Table`
The specified `Grid.Norm.Table` in the function call.

`Digits.Percentile`
The number of digits after the decimal point that were requested for the percentile ranks.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

`Stage.2.NormScore`, `Stage.2.AutoScore`, `Bootstrap.Stage.2.NormScore`

Examples

```
# Replicate the normative table that was obtained in
# Case study 1 of Chapter 3 in Van der Elst (2023)
# ------------------------------------------------
library(NormData) # load the NormData package
data(GCSE)        # load the GCSE dataset

# Fit the Stage 1 model
```

```
Model.1.GCSE <- Stage.1(Dataset=GCSE,
    Model=Science.Exam~Gender)

# Make a normative table for raw Science Exam scores = 10,
# 11, ... 85, stratified by Gender
NormTable.GCSE <- Stage.2.NormTable(Stage.1.Model=Model.1.GCSE,
  Test.Scores=c(10:85),
  Grid.Norm.Table=data.frame(Gender=c("F", "M")))

summary(NormTable.GCSE)


# Replicate the normative table that was obtained in
# Case study 1 of Chapter 7 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset

# Make the new variable Age.C (= Age centered) that is
# needed to fit the final Stage 1 model,
# and add it to the Substitution dataset
Substitution$Age.C <- Substitution$Age - 50

# Fit the final Stage 1 model
Substitution.Model.9 <- Stage.1(Dataset=Substitution,
    Alpha=0.005, Model=LDST~Age.C+LE, Order.Poly.Var=1)

# Make a normative table for LDST scores = 10, 12, ... 56,
# stratified by Age and LE
NormTable.LDST <- Stage.2.NormTable(
  Stage.1.Model=Substitution.Model.9,
  Test.Scores=seq(from=10, to=56, by=2),
  Grid.Norm.Table=expand.grid(Age.C=seq(from=-30, to=30, by=1),
  LE=c("Low", "Average", "High")))
```

STAS *State-Trait Anger Scale (STAS)*

Description

This dataset contains the scores of the Trait Anger scale of the STAS. The test participants were 316 first-year psychology students from a university in the Dutch speaking part of Belgium. Participation was a partial fulfillment of the requirement to participate in research. The sample consists of 73 males and 243 females, reflecting the gender proportion among psychology students. The average age was 18.4 years. The data originally come from the package `psychotools`, dataset `VerbalAgression`.
For more info, see `https://cran.r-project.org/web/packages/psychotools/psychotools.pdf`.

Usage

```
data(STAS)
```

Format

A `data.frame` with 316 observations on 3 variables.

`Id` The Id number of the student.

`Gender` The gender of the student, coded as a factor.

`Anger` The Trait Anger scale score of the STAS.

`Substitution` *Substitution test data*

Description

Substitution tests are speed-dependent tasks that require the participant to match particular signs (symbols, digits, or letters) to other signs within a specified time period. The LDST is an adaptation of earlier substitution tests, such as the Digit Symbol Substitution Test (DSST; Wechsler, 1981) and the Symbol Digit Modalities Test (SDMT; Smith, 1982). The LDST differs from other substitution tests in that the key consists of 'over-learned' signs, i.e., letters and digits. These are simulated data that are based on the results described in Van der Elst *et al.* (2006) (see Table 2).

Usage

```
data(Substitution)
```

Format

A `data.frame` with 1765 observations on 5 variables.

`Id` The Id number of the participant.

`Age` The age of the participant, in years.

`Gender` The gender of the participant, coded as a factor with levels `Male` and `Female`.

`LE` The Level of Education of the test participant, coded as a factor with levels `Low`, `Average` and `High`.

`LDST` The test score on the LDST (written version), i.e., the number of correct substitutions made in 60 seconds. A higher score reflects better performance.

TMAS *TMAS data*

Description

This dataset contains the scores of the Taylor Manifest Anxiety Scale (TMAS; Taylor, 1953), administered online. A total of 523 test participants completed the questionnaire. The TMAS scale score ranges between 0 and 50, with lower scores corresponding to higher levels of anxiety.

Usage

```
data(TMAS)
```

Format

A `data.frame` with 523 observations on 3 variables.

`Id` The Id number of the test participant.

`Gender` The gender of the test participant, coded as a factor.

`Score` The TMAS score. A higher value is iindicative for less anxiety.

References

Taylor, J. (1953). A personality scale of manifest anxiety. *The Journal of Abnormal and Social Psychology, 48(2),* 285-290.

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

Tukey.HSD *Conducts Tukey's Honest Significance Difference test*

Description

This function conducts Tukey's Honest Significance Difference (HSD; Tukey, 1949) test that allows for making post hoc comparisons of the group means. Tukey's HSD can only be conducted when the mean structure of the Stage 1 model only contains qualitative independent variables (i.e., when the fitted regression model is essentially an ANOVA).

Usage

```
Tukey.HSD(Stage.1.Model, ...)
```

Arguments

`Stage.1.Model`	A fitted stage one model that only contains qualitative variables.
`...`	Arguments to be passed to the plot function of the Tukey HSD procedure.

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Tukey, J. (1949). Comparing individual means in the Analysis of Variance. *Biometrics, 5,* 99-114.

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

plot.Tukey.HSD

Examples

```
data(Personality)
Model.Openness <- Stage.1(Dataset = Personality, Model = Openness ~ LE)
# conduct post hoc comparisons for the levels of education
Tukey.Openness <- Tukey.HSD(Model.Openness)
summary(Tukey.Openness)
plot(Tukey.Openness)

# conduct post hoc comparisons for the levels of education by education combinations
data(Substitution)
Model.Substitution <- Stage.1(Dataset = Substitution, Model = LDST ~ LE*Gender)
Tukey.Substitution <- Tukey.HSD(Model.Substitution)
summary(Tukey.Substitution)
plot(Tukey.Substitution)
```

VLT *Verbal Learning Test data*

Description

This dataset contains the Total Recall scores of the Verbal Learning Test (VLT). A total of 1460 test-participants participated in the study. These are simulated data based on the results described in Van der Elst *et al.* (2005).

Usage

```
data(VLT)
```

Format

A `data.frame` with 1460 observations on 5 variables.

`Id` The Id number of the test participant.

`Age` The age of the test participant (in years).

`Gender` The gender of the test participant, coded as a factor.

`LE` The level of education of the test participant.

`Total.Recall` The Total Recall score. A higher score is indicative for better verbal memory ability.

References

Van der Elst *et al.* (2005). Rey's Verbal Learning Test: Normative data for 1,855 healthy participants aged 24-81 years and the influence of age, sex, education, and mode of presentation. *Journal of the International Neuropsychological Society, 11,* 290-302.

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

`WriteNormTable` *Write a normative table from R to a .txt/.csv/.xlsx file*

Description

The function `Stage.2.NormTable()` allows for deriving a normative table that shows percentile ranks $\hat{\pi}_0$ that correspond to a wide range of raw test scores Y_0 (stratified by the relevant independent variables). The raw R output format that is provided by the `Stage.2.NormTable()` function is not always convenient, especially when a large number of test scores are tabulated and the table is spread out over several lines. The function `WriteNormTable()` can be used to export the normative table to a `.txt`, `.csv` or `.xlsx` file. Such a file can then be opened in a spreadsheet (such as Google Sheets or LibreOffice), where the normative table can be put in a more user-friendly format.

Usage

```
WriteNormTable(NormTable, Folder, NameFile="NormTable.xlsx",
verbose=TRUE)
```

Arguments

`NormTable`	An object of class `Stage.2.NormTable` that contains the normative table that has to be exported.
`Folder`	The folder where the file with the normative table should be saved.
`NameFile`	The name of the file to which the normative table should be written. Only the extensions `.txt`, `.csv` or `.xlsx` can be used. If unspecified, the argument `NameFile="NormTable.xlsx"` is used by default. The `.txt` and `.csv` files use a space as the delimiter.
`verbose`	A logical value indicating whether verbose output should be generated.

Value

No return value, called for side effects.

Author(s)

Wim Van der Elst

References

Van der Elst, W. (2024). *Regression-based normative data for psychological assessment: A hands-on approach using R.* Springer Nature.

See Also

Stage.2.NormTable

Examples

```
# Replicate the normative table that was obtained in
# Case study 1 of Chapter 3 in Van der Elst (2023)
# ------------------------------------------------------
library(NormData) # load the NormData package
data(GCSE)        # load the GCSE dataset

# Fit the Stage 1 model
Model.1.GCSE <- Stage.1(Dataset=GCSE,
    Model=Science.Exam~Gender)

# Make a normative table for raw Science Exam scores = 10,
# 11, ... 85, stratified by Gender
NormTable.GCSE <- Stage.2.NormTable(Stage.1.Model=Model.1.GCSE,
    Test.Scores=c(10:85),
    Grid.Norm.Table=data.frame(Gender=c("F", "M")))
summary(NormTable.GCSE)

# Write the normative table to the user's computer
WriteNormTable(NormTable=NormTable.GCSE,
     NameFile="NormTable.GCSE.xlsx",
     Folder=tempdir()) # Replace tempdir() by the desired folder


# Replicate the normative table that was obtained in
# Case study 1 of Chapter 7 in Van der Elst (2023)
# ------------------------------------------------
library(NormData)   # load the NormData package
data(Substitution)  # load the Substitution dataset

# Make the new variable Age.C (= Age centered) that is
# needed to fit the final Stage 1 model,
# and add it to the Substitution dataset
Substitution$Age.C <- Substitution$Age - 50
```

```
# Fit the final Stage 1 model
Substitution.Model.9 <- Stage.1(Dataset=Substitution,
   Alpha=0.005, Model=LDST~Age.C+LE, Order.Poly.Var=1)

# Make a normative table for LDST scores = 10, 12, ... 56,
# stratified by Age and LE
NormTable.LDST <- Stage.2.NormTable(
  Stage.1.Model=Substitution.Model.9,
  Test.Scores=seq(from=10, to=56, by=2),
  Grid.Norm.Table=expand.grid(Age.C=seq(from=-30, to=30, by=1),
     LE=c("Low", "Average", "High")))

# Write the normative table to the user's computer
WriteNormTable(NormTable=NormTable.LDST,
  NameFile="NormTable.LDST.xlsx",
  Folder=tempdir()) # Replace tempdir() by the desired folder
```

Index