

Package ‘ez’

January 2, 2012

Version 3.0-0

Date 2011-02-05

Title Easy analysis and visualization of factorial experiments.

Author Michael A. Lawrence <Mike.Lawrence@dal.ca>

Maintainer Michael A. Lawrence <Mike.Lawrence@dal.ca>

URL <http://groups.google.com/group/ez4r> (discussion list),<https://github.com/mike-lawrence/ez/issues> (bug reports and feature requests)

Depends R (>= 2.12-1), car (>= 2.0-8), reshape2 (>= 1.1), plyr (>= 1.4), ggplot2 (>= 0.8-9), stringr (>= 0.4), lme4 (>= 0.999375-37), Matrix (>= 0.999375-46)

Description This package facilitates easy analysis of factorial experiments, including purely within-Ss designs (a.k.a. ‘‘repeated measures’’), purely between-Ss designs, and mixed within-and-between-Ss designs. The functions in this package aim to provide simple, intuitive and consistent specification of data analysis and visualization. Visualization functions also include design visualization for pre-analysis data auditing, and correlation matrix visualization. Finally, this package includes functions for non-parametric analysis, including permutation tests and bootstrap resampling. The bootstrap function obtains predictions either by cell means or by more advanced/powerful mixed effects models, yielding predictions and confidence intervals that may be easily visualized at any level of the experiment’s design.

License GPL (>= 2)

LazyLoad yes

Repository CRAN

Date/Publication 2011-02-06 09:19:22

R topics documented:

ez-package	2
ANT	4
ANT2	5
ezANOVA	5
ezBoot	9
ezBootPlot	11
ezCor	13
ezDesign	16
ezMixed	18
ezPerm	21
ezPlot	23
ezPrecis	28
ezPredict	30
ezResample	31
ezStats	33
progress_time	36
progress_timeCI	37
Index	38

 ez-package

Easy analysis and visualization of factorial experiments.

Description

This package facilitates easy analysis of factorial experiments, including purely within-Ss designs (a.k.a. "repeated measures"), purely between-Ss designs, and mixed within-and-between-Ss designs. The functions in this package aim to provide simple, intuitive and consistent specification of data analysis and visualization. Visualization functions also include design visualization for pre-analysis data auditing, and correlation matrix visualization. Finally, this package includes functions for non-parametric analysis, including permutation tests and bootstrap resampling. The bootstrap function obtains predictions either by cell means or by more advanced/powerful mixed effects models, yielding predictions and confidence intervals that may be easily visualized at any level of the experiment's design.

Details

Package: ez
 Type: Package
 Version: 3.0-0
 Date: 2011-02-05
 License: GPL-3
 LazyLoad: yes

This package contains several useful functions:

- [ezANOVA](#) Provides simple interface to ANOVA, including assumption checks.
- [ezBoot](#) Computes bootstrap resampled cell means or lmer predictions
- [ezBootPlot](#) When supplied the results from a call to [ezBoot](#), plots predictions with bootstrapped confidence intervals.
- [ezCor](#) Function to plot a correlation matrix with scatterplots, linear fits, and univariate density plots
- [ezDesign](#) Function to plot a visual representation of the balance of data given a specified experimental design. Useful for diagnosing missing data issues.
- [ezMixed](#) Provides assessment of fixed effects in a mixed effects modelling context.
- [ezPerm](#) Provides simple interface to the Permutation test.
- [ezPlot](#) Uses the `ggplot2` graphing package to generate plots for any given user-requested effect, by default producing error bars that facilitate visual post-hoc multiple comparisons.
- [ezPrecis](#) Provides a summary of a given data frame.
- [ezPredict](#) Computes predicted values from the fixed effects of a mixed effects model.
- [ezResample](#) Resamples data, useful when bootstrapping.
- [ezStats](#) Provides between-Ss descriptive statistics for any given user-requested effect.
- [progress_time](#) A progress bar that estimates time remaining.
- [progress_timeCI](#) A progress bar that estimates time remaining, including a bootstrapped confidence interval (useful when operations vary in duration). This package also contains two data sets:
- [ANT](#) Simulated data from the Attention Network Test
- [ANT2](#) Messy version of the ANT data set

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

ANT

ANT data

Description

Simulated data from the Attention Network Test (see reference below), consisting of 2 within-Ss variables ("cue" and "flank"), 1 between-Ss variable ("group") and 2 dependent variables (response time, "rt", and whether an error was made, "error")

Usage

```
data(ANT)
```

Format

A data frame with 5760 observations on the following 10 variables.

subnum a factor with levels 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

group a factor with levels Control Treatment

block a numeric vector

trial a numeric vector

cue a factor with levels None Center Double Spatial

flank a factor with levels Neutral Congruent Incongruent

location a factor with levels down up

direction a factor with levels left right

rt a numeric vector

error a numeric vector

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

References

J Fan, BD McCandliss, T Sommer, A Raz, MI Posner (2002). Testing the efficiency and independence of attentional networks. *Journal of Cognitive Neuroscience*, **14**, 340-347.

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
data(ANT)
head(ANT)
ezPrecis(ANT)
```

ANT2

Messy ANT data

Description

A "messy" version of the ANT data set (see [ANT](#)). In this version of the data, subnum #7 is missing data from the last half of the experiment, subnum #14 made all errors in the incongruent cells, and subnum #12 mistakenly reversed their responses.

Usage

```
data(ANT2)
```

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
data(ANT2)
head(ANT2)
ezPrecis(ANT2)
```

ezANOVA

Function to perform a factorial ANOVA

Description

This function provides easy analysis of data from factorial experiments, including purely within-Ss designs (a.k.a. "repeated measures"), purely between-Ss designs, and mixed within-and-between-Ss designs, yielding ANOVA results and assumption checks.

Usage

```
ezANOVA(
  data
  , dv
  , wid
  , within = NULL
  , between = NULL
  , observed = NULL
  , diff = NULL
  , reverse_diff = FALSE
  , type = 2
  , white.adjust = FALSE
  , detailed = FALSE
  , return_aov = FALSE
)
```

Arguments

<code>data</code>	Data frame containing the data to be analyzed.
<code>dv</code>	.() object specifying the column in <code>data</code> that contains the dependent variable. Values in this column must be numeric.
<code>wid</code>	.() object specifying the column in <code>data</code> that contains the variable specifying the case/Ss identifier.
<code>within</code>	Optional .() object specifying one or more columns in <code>data</code> that contain predictor variables that are manipulated (or observed) within-Ss.
<code>between</code>	Optional .() object specifying one or more columns in <code>data</code> that contain predictor variables that are manipulated (or observed) between-Ss.
<code>observed</code>	Optional .() object specifying one or more columns in <code>data</code> that are already specified in either <code>within</code> or <code>between</code> that contain predictor variables that are observed variables (i.e. not manipulated). The presence of observed variables affects the computation of the generalized eta-squared measure of effect size reported by ezANOVA .
<code>diff</code>	Optional .() object specifying a 2-level within-Ss variable to collapse to a difference score.
<code>reverse_diff</code>	Logical. If TRUE, triggers reversal of the difference collapse requested by <code>diff</code> .
<code>type</code>	Numeric value (either 1, 2 or 3) specifying the Sums of Squares "type" to employ when data are unbalanced (eg. when group sizes differ). <code>type = 2</code> is the default because this will yield identical ANOVA results as <code>type = 1</code> when data are balanced but <code>type = 2</code> will additionally yield various assumption tests where appropriate. When data are unbalanced, users are warned that they should give special consideration to the value of <code>type</code> . <code>type=3</code> will emulate the approach taken by popular commercial statistics packages like SAS and SPSS, but users are warned that this approach is not without criticism.
<code>white.adjust</code>	Only affects behaviour if the design contains only between-Ss predictor variables. If not FALSE, the value is passed as the <code>white.adjust</code> argument to Anova ,

	which provides heteroscedasticity correction. See Anova for details on possible values.
detailed	Logical. If TRUE, returns extra information (sums of squares, raw likelihood ratios, AIC & BIC values used in computing corrected likelihood ratios, etc.).
return_aov	Logical. If TRUE, computes and returns an aov object corresponding to the requested ANOVA (useful for computing post-hoc contrasts).

Details

While `within` and `between` are both optional, at least one column of data must be provided to either `within` or `between`. Prior to running, `dv` is collapsed to a mean for each cell defined by the combination of `wid` and any variables supplied to `within` and/or `between` and/or `diff`.

Value

A list containing one or more of the following components:

ANOVA	A data frame containing the ANOVA results.
Mauchly's Test for Sphericity	If any within-Ss variables with >2 levels are present, a data frame containing the results of Mauchly's test for Sphericity. Only reported for effects >2 levels because sphericity necessarily holds for effects with only 2 levels.
Sphericity Corrections	If any within-Ss variables are present, a data frame containing the Greenhouse-Geisser & Huynh-Feldt epsilon values, and corresponding corrected p-values.
Levene's Test for Homogeneity	If the design is purely between-Ss, a data frame containing the results of Levene's test for Homogeneity of variance. Note that Huynh-Feldt corrected p-values where the Huynh-Feldt epsilon >1 will use 1 as the correction epsilon.
aov	An aov object corresponding to the requested ANOVA.

Note

Some column names in the output data frames are abbreviated to conserve space:

DFn	Degrees of Freedom in the numerator (a.k.a. DFeffect).
DFd	Degrees of Freedom in the denominator (a.k.a. DFerror).
SSn	Sum of Squares in the numerator (a.k.a. SSeffect).
SSd	Sum of Squares in the denominator (a.k.a. SSerror).
F	F-value.
p	p-value (probability of the null hypothesis given the data).
p<.05	Highlights p-values less than the traditional alpha level of .05.
ges	Generalized Eta-Squared measure of effect size (see in references below: Bakeman, 2005).
GGe	Greenhouse-Geisser epsilon.
p[GGe]	p-value after correction using Greenhouse-Geisser epsilon.
p[GGe]<.05	Highlights p-values (after correction using Greenhouse-Geisser epsilon) less than the traditional alpha level of .05.
HFe	Huynh-Feldt epsilon.
p[HFe]	p-value after correction using Huynh-Feldt epsilon.

p[HFe]<.05 Highlights p-values (after correction using Huynh-Feldt epsilon) less than the traditional alpha level of .05.
 W Mauchly's W statistic

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

References

- Bakeman, R. (2005). Recommended effect size statistics for repeated measures designs. Behavior Research Methods, 37 (3), 379-384.
- Glover S, Dixon P. (2004) Likelihood ratios: a simple and flexible statistic for empirical psychologists. Psychonomic bulletin & review, 11 (5), 791-806.

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
#Read in the ANT data (see ?ANT).
data(ANT)
head(ANT)
ezPrecis(ANT)

#Run an ANOVA on the mean correct RT data.
rt_anova = ezANOVA(
  data = ANT[ANT$error==0,]
  , dv = .(rt)
  , wid = .(subnum)
  , within = .(cue,flank)
  , between = .(group)
)

#Show the ANOVA & assumption tests.
print(rt_anova)

#Run an ANOVA on the mean_rt data, ignoring group.
rt_anova2 = ezANOVA(
  data = ANT[ANT$error==0,]
  , dv = .(rt)
  , wid = .(subnum)
  , within = .(cue,flank)
)

#Show the ANOVA & assumption tests.
print(rt_anova2)
```

```

#Run a purely between-Ss ANOVA on the mean_rt data.
rt_anova3 = ezANOVA(
  data = ANT[ANT$error==0,]
  , dv = .(rt)
  , wid = .(subnum)
  , between = .(group)
)

#Show the ANOVA & assumption tests.
print(rt_anova3)

```

ezBoot	<i>Function to compute bootstrap resampled predictions for each cell in a specified experimental design.</i>
--------	--

Description

This function is used to compute bootstrap resampled predictions for each cell in a specified experimental design, using either cell means or mixed effects modelling to obtain predictions. The results can be visualized using [ezBootPlot](#).

Usage

```

ezBoot(
  data
  , dv
  , wid
  , within = NULL
  , between = NULL
  , resample_within = TRUE
  , iterations = 1e3
  , lmer = TRUE
  , family = 'gaussian'
  , alarm = TRUE
)

```

Arguments

data	Data frame containing the data to be analyzed.
dv	.() object specifying the column in data that contains the dependent variable. Values in this column must be numeric.
wid	.() object specifying the column in data that contains the variable specifying the case/Ss identifier.
within	Optional .() object specifying one or more columns in data that contain predictor variables that are manipulated (or observed) within-Ss.

<code>between</code>	Optional <code>.</code> (<code>.</code>) object specifying one or more columns in data that contain predictor variables that are manipulated (or observed) between-Ss.
<code>resample_within</code>	Logical value specifying whether to resample within each cell of the design within each wid unit. If there is only one observation per such cells, then this should be set to <code>FALSE</code> to avoid useless computation.
<code>iterations</code>	Numeric value specifying the number of bootstrap iterations to complete.
<code>lmer</code>	Logical. If <code>TRUE</code> (default), predictions are obtained via mixed effects modelling; if <code>FALSE</code> predictions are obtained via cell means.
<code>family</code>	When obtaining predictions via mixed effects modelling (i.e. when <code>lmer=TRUE</code>), you must specify the residuals family. While the bootstrap is in theory non-parametric, it may be more powerful if you specify a family that might reasonably be expected to match your data. For example, if the data are binary outcomes (eg. accuracy), then use the "binomial" family. See <code>lmer</code> .
<code>alarm</code>	Logical. If <code>TRUE</code> (default), call the <code>alarm</code> function when <code>ezBoot</code> completes.

Details

While `within` and `between` are both optional, at least one column of data must be provided to either `within` or `between`. Any numeric or character variables in data that are specified as either `wid`, `within` or `between` will be converted to a factor with a warning. Prior to running, `dv` is collapsed to a mean for each cell defined by the combination of `wid`, `within` or `between`.

Value

A list containing either two or three components:

<code>fit</code>	If predictions are obtained by mixed effects modelling, an <code>link[lme4]{lmer}</code> object consisting of the original mixed effects model
<code>cells</code>	A data frame containing predictions for each cell of the design.
<code>boots</code>	A data frame containing predictions for each cell of the design from each iteration of the bootstrap procedure.

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```

#Read in the ANT data (see ?ANT).
data(ANT)
head(ANT)
ezPrecis(ANT)

#Run ezBoot on the accurate RT data
rt = ezBoot(
  data = ANT
  , dv = .(rt)
  , wid = .(subnum)
  , within = .(cue,flank)
  , between = .(group)
  , iterations = 1e1 #1e3 or higher is best for publication
)

#plot the full design
from_ezBootPlot = ezBootPlot(
  from_ezBoot = rt
  , x = .(flank)
  , split = .(cue)
  , col = .(group)
)
print(from_ezBootPlot$plot)

#plot the effect of group across the flank*cue design
from_ezBootPlot = ezBootPlot(
  from_ezBoot = rt
  , x = .(flank)
  , split = .(cue)
  , diff = .(group)
)
print(from_ezBootPlot$plot)

#plot the flank*cue design, averaging across group
from_ezBootPlot = ezBootPlot(
  from_ezBoot = rt
  , x = .(flank)
  , split = .(cue)
)
print(from_ezBootPlot$plot)

```

ezBootPlot

Function to plot bootstrapped predictions and confidence intervals.

Description

This function provides easy visualization of any given user-requested effect from the bootstrap predictions computed by [ezBoot](#).

Usage

```
ezBootPlot(
  from_ezBoot
  , confidence = .95
  , x
  , split = NULL
  , row = NULL
  , col = NULL
  , do_lines = TRUE
  , bar_width = NULL
  , to_numeric = NULL
  , x_lab = NULL
  , y_lab = NULL
  , split_lab = NULL
  , levels = NULL
  , diff = NULL
  , reverse_diff = FALSE
  , row_y_free = FALSE
  , alarm = TRUE
)
```

Arguments

from_ezBoot	An list object resulting from a call to ezBoot .
confidence	Numeric vector of one or more confidence levels to use for plotting error bars. If plotting multiple confidence regions, it is suggested that an equal number of different values are supplied to the bar_width argument for differentiation.
x	.() object specifying the variable to plot on the x-axis.
split	Optional .() object specifying a variable by which to split the data into different shapes/colors (and line types, if do_lines==TRUE).
row	Optional .() object specifying a variable by which to split the data into rows.
col	Optional .() object specifying a variable by which to split the data into columns.
do_lines	Logical. If TRUE, lines will be plotted connecting groups of points.
bar_width	Optional numeric value specifying custom widths for the error bar hat. Must either have a length of 1, or the same length as confidence.
to_numeric	Optional .() object specifying any variables that need to be converted to the numeric class before plotting.
x_lab	Optional character string specifying the x-axis label.
y_lab	Optional character string specifying the y-axis label.
split_lab	Optional character string specifying the key label.
levels	Optional named list where each item name matches a factored column in data that needs either reordering of levels, renaming of levels, or both. Each item should be a list containing named elements new_order or new_names or both.
diff	Optional .() object specifying a 2-level variable to collapse to a difference score.

reverse_diff	Logical. If TRUE, triggers reversal of the difference collapse requested by diff.
row_y_free	Logical. If TRUE, then rows will permit different y-axis scales.
alarm	Logical. If TRUE (default), call the <code>alarm</code> function when <code>ezBootPlot</code> completes.

Value

A list with 4 components:

plot	A printable/modifiable ggplot2 object.
cells	A data frame containing predictions for each cell in the requested design.
boots	A data frame containing each iteration of bootstrap predictions, collased to the requested design.
boot_stats	A data frame containing the "lo" and "hi" bounds of the 95 percent bootstrap confidence interval of each cell in the requested design.

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
#see examples in ezBoot
```

ezCor	<i>Function to plot a correlation matrix with scatterplots, linear fits, and univariate density plots</i>
-------	---

Description

This function provides simultaneous visualization of a correlation matrix, scatter-plot with linear fits, and univariate density plots for multiple variables.

Usage

```
ezCor(
  data
  , r_size_lims = c(10,30)
  , point_alpha = .5
  , density_height = 1
  , density_adjust = 1
  , density_colour = 'white'
  , label_size = 10
  , label_colour = 'black'
  , label_alpha = .5
  , lm_colour = 'red'
  , ci_colour = 'green'
  , ci_alpha = .5
  , test_alpha = .05
)
```

Arguments

<code>data</code>	Data frame containing named columns of data only.
<code>r_size_lims</code>	Minimum and maximum size of the text reporting the correlation coefficients. Minimum is mapped to coefficients of 0 and maximum is mapped to coefficients of 1, with the mapping proportional to r^2 .
<code>point_alpha</code>	Transparency of the data points (1 = opaque).
<code>density_adjust</code>	Adjusts the bandwidth of the univariate density estimator. See "adjust" parameter in density
<code>density_height</code>	Proportion of the facet height taken up by the density plots.
<code>density_colour</code>	Colour of the density plot.
<code>label_size</code>	Size of the variable labels on the diagonal.
<code>label_colour</code>	Colour of the variable labels on the diagonal.
<code>label_alpha</code>	Transparency of the variable labels on the diagonal (1 = opaque).
<code>lm_colour</code>	Colour of the fitted line.
<code>ci_colour</code>	Colour of the confidence interval surrounding the fitted line.
<code>ci_alpha</code>	Transparency of the confidence interval surrounding the fitted line (1 = opaque).
<code>test_alpha</code>	Type-I error rate requested for coloring of the "significant" correlation coefficients.

Value

A ggplot2 object.

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
#####
# Set up some fake data
#####
library(MASS)
N=100

#first pair of variables
variance1=1
variance2=2
mean1=10
mean2=20
rho = .8
Sigma=matrix(
  c(
    variance1
    , sqrt(variance1*variance2)*rho
    , sqrt(variance1*variance2)*rho
    , variance2
  )
  , 2
  , 2
)
pair1=mvnrm(N,c(mean1,mean2),Sigma,empirical=TRUE)

#second pair of variables
variance1=10
variance2=20
mean1=100
mean2=200
rho = -.4
Sigma=matrix(
  c(
    variance1
    , sqrt(variance1*variance2)*rho
    , sqrt(variance1*variance2)*rho
    , variance2
  )
  , 2
  , 2
)
pair2=mvnrm(N,c(mean1,mean2),Sigma,empirical=TRUE)

my_data=data.frame(cbind(pair1,pair2))

#####
# Now plot
```

```
#####
p = ezCor(
  data = my_data
)
print(p)

#you can modify the default colours of the
##correlation coefficients as follows
p = p + scale_colour_manual(values = c('red', 'blue'))
print(p)
#see the following for alternatives:
# http://had.co.nz/ggplot2/scale_manual.html
# http://had.co.nz/ggplot2/scale_hue.html
# http://had.co.nz/ggplot2/scale_brewer.html
```

 ezDesign

Function to plot the balance of data in an experimental design

Description

This function provides easy visualization of the balance of data in a data set given a specified experimental design. This function is useful for identifying missing data and other issues (see examples).

Usage

```
ezDesign(
  data
  , x
  , y
  , row = NULL
  , col = NULL
  , cell_border_size = 10
)
```

Arguments

<code>data</code>	Data frame containing the data to be visualized.
<code>x</code>	.() object specifying the column in data that contains the variable to plot on the x-axis.
<code>y</code>	.() object specifying the column in data that contains the variable to plot on the y-axis.
<code>row</code>	Optional .() object specifying a variable by which to split the data into rows.
<code>col</code>	Optional .() object specifying a variable by which to split the data into columns.
<code>cell_border_size</code>	Numeric value specifying the size of the border separating cells (0 specifies no border)

Details

The function works by counting the number of rows in data in each cell of the design specified by the factorial combination of x, y, row, col variables.

Value

A printable/modifiable ggplot2 object.

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
#Read in the ANT2 data (see ?ANT2).
data(ANT2)
head(ANT2)
ezPrecis(ANT2)

ezDesign(
  data = ANT2
  , x = .(trial)
  , y = .(subnum)
  , row = .(block)
  , col = .(group)
)
#subnum #7 is missing data from the last half of the experiment

ezDesign(
  data = ANT2
  , x = .(flank)
  , y = .(subnum)
  , row = .(cue)
)
#again, subnum#7 has half the data as the rest

#now look at error rates, which affect the number of RTs we can use
ezDesign(
  data = ANT2[ANT2$error==0,]
  , x = .(flank)
  , y = .(subnum)
  , row = .(cue)
)
#again, subnum#7 stands out because they have half the data as the rest
```

```
#also, subnum#14 has no data in any incongruent cells, suggesting that
##they made all errors in this condition
#finally, subnum#12 has virtually no data, suggesting that they mistakenly
##swapped responses
```

ezMixed	<i>Provides assessment of fixed effects in an mixed effects modelling context.</i>
---------	--

Description

This function provide assessment of fixed effects and their interactions via mixed effects modelling, including automated assessment of non-linearity via polynomials. Models are built using [lmer](#) and permit specification of multiple random effects.

Usage

```
ezMixed(
  data
  , dv
  , random
  , fixed
  , fixed_poly = NULL
  , fixed_poly_max = NULL
  , family = gaussian
  , alarm = TRUE
  , results_as_progress = FALSE
  , highest = 0
  , return_models = FALSE
  , highest_first = TRUE
)
```

Arguments

data	Data frame containing the data to be analyzed.
dv	.() object specifying the column in data that contains the dependent variable. Values in this column must be numeric.
random	.() object specifying one or more columns in data that contain random effects.
fixed	.() object specifying one or more columns in data that contain fixed effects.
fixed_poly	Optional .() object specifying one or more columns in data that are already specified in fixed and contain fixed effects to be fit with polynomials.
fixed_poly_max	Optional numeric vector with the same length as fixed_poly specifying the maximum polynomial degree for each corresponding variable supplied to fixed_poly. Otherwise, the default maximum degree is 5.

family	a description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. See family .
alarm	Logical. If TRUE (default), call the alarm function when ezMixed completes.
results_as_progress	Logical. Default is FALSE, but if TRUE, results are printed as they are computed (useful for very long computations).
highest	Integer specifying the highest order interaction between the fixed effects to test. The default value, 0, will test to the highest possible order.
return_models	Logical. If TRUE, the returned list object will also include each lmer model (can become memory intensive for complex models and/or large data sets).
highest_first	Logical. Default is TRUE, which computes the models for the highest order interaction first.

Details

Computation is achieved via [lmer](#). Assessment of each effect of interest necessitates building two models: (1) a "unrestricted" model that contains the effect of interest plus any lower order effects and (2) a "restricted" model that contains only the lower order effects (thus "restricting" the effect of interest to zero). These are then compared by means of a likelihood ratio, which needs to be corrected to account for the additional complexity of the unrestricted model relative to the restricted model. This correction can be achieved by either the Akaike Information Criterion (AIC) or the Bayesian Information Criterion (BIC), two equally well-defined but differently motivated approaches to accounting for model complexity. Generally, the BIC imposes a stronger penalty for complexity, yielding corrected likelihood ratios that are more likely to favor the restricted model. Users are encouraged to refer to Kuha (2004, listed in references below) for discussion of the different motivations of AIC and BIC to help decide which suits their application best.

Both complexity-corrected variants of the likelihood ratio returned by [ezMixed](#) are transformed to the log-base-10 scale (also known as the Bel scale), which has the following convenient properties:

- (1) The Bel scale permits easy representation of both very large and very small likelihood ratios.
- (2) The Bel scale represents equivalent evidence between the restricted and unrestricted models by a value of 0.
- (3) The Bel scale represents ratios favoring the restricted model symmetrically to those favoring the unrestricted model. That is, say one effect obtains a likelihood ratio of 10, and another effect obtains a likelihood ratio of .1; both ratios indicate the same degree of imbalance of evidence (10:1 and 1:10) and on the Bel scale they faithfully represent this symmetry as values 1 and -1, respectively.
- (4) While any logarithmic transform achieves the above benefits, the Bel scale is particularly useful because integer differences on the Bel scale represent changes by an order of magnitude on the raw likelihood ratio scale (eg. 1 Bel corresponds to a likelihood ratio of 1, 2 Bels corresponds to a likelihood ratio of 100, etc.).

Value

A list with the following elements:

summary	A data frame summarizing the results, including whether warnings or errors occurred during the assessment of each effect, raw natural-log likelihood of the unrestricted and restricted models (RLnLu and RLnLr, respectively), degrees of freedom of the unrestricted and restricted models (DFu and DFr, respectively), and log-base-10 likelihood ratios corrected via AIC and BIC (L10LRa and L10LRb, respectively)
formulae	A list of lists, each named for an effect and containing two elements named "unrestricted" and "restricted", which in turn contain the right-hand-side formulae used to fit the unrestricted and restricted models, respectively.
errors	A list similar to formulae, but instead storing errors encountered in fitting each model.
warnings	A list similar to formulae, but instead storing warnings encountered in fitting each model.
models	(If requested by setting return_models=TRUE) A list similar to formulae but instead storing each fitted model.

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

References

- Kuha, J. (2004). AIC and BIC : Comparisons of Assumptions and Performance. Sociological Methods & Research, 33, p188.

See Also

[lmer](#), [ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
#Read in the ANT data (see ?ANT).
data(ANT)
head(ANT)
ezPrecis(ANT)

#Run ezMixed on the accurate RT data
rt = ezMixed(
  data = ANT[ANT$error==0,]
  , dv = .(rt)
  , random = .(subnum)
  , fixed = .(cue,flank,group)
)
```

```

print(rt$summary)

## Not run:
#Run ezMixed on the error rate data
er = ezMixed(
  data = ANT
  , dv = .(error)
  , random = .(subnum)
  , fixed = .(cue,flank,group)
  , family = 'binomial'
)
print(er$summary)

## End(Not run)

```

ezPerm

Function to perform a factorial permutation test

Description

This function provides easy non-parametric permutation test analysis of data from factorial experiments, including purely within-Ss designs (a.k.a. "repeated measures"), purely between-Ss designs, and mixed within-and-between-Ss designs.

Usage

```

ezPerm(
  data
  , dv
  , wid
  , within = NULL
  , between = NULL
  , perms
  , alarm = TRUE
)

```

Arguments

data	Data frame containing the data to be analyzed.
dv	.() object specifying the column in data that contains the dependent variable. Values in this column must be numeric.
wid	.() object specifying the column in data that contains the variable specifying the case/Ss identifier.
within	Optional .() object specifying one or more columns in data that contain predictor variables that are manipulated within-Ss.
between	Optional .() object specifying one or more columns in data that contain predictor variables that are manipulated between-Ss.

perms An integer > 0 specifying the number of permutations to compute.
 alarm Logical. If TRUE (default), call the `alarm` function when `ezPerm` completes.

Details

While `within` and `between` are both optional, at least one column of data must be provided to either `within` or `between`. Any numeric or character variables in data that are specified as either `wid`, `within` or `between` will be converted to a factor with a warning. The expected standard deviation of p-values is approximately $\sqrt{\text{true_p} \times (1 - \text{true_p}) / \text{perms}}$; significance tests using an alpha of .05 should therefore employ at least $1e3$ permutations. As the permutation test is computationally intensive, it is advisable to pre-test smaller values of `perms` and extrapolate to estimate the total test duration before attempting a full run. To facilitate such extrapolation, test duration is provided in the output after running a permutation test.

Value

A data frame containing the permutation test results.

Warning

`ezPerm()` is a work in progress. Under the current implementation, only main effects may be trusted.

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
#Read in the ANT data (see ?ANT).
data(ANT)
head(ANT)
ezPrecis(ANT)

#Compute some useful statistics per cell.
cell_stats = ddply(
  .data = ANT
  , .variables = .( subnum , group , cue , flank )
  , .fun <- function(x){
    #Compute error rate as percent.
    error_rate = mean(x$error)*100
    #Compute mean RT (only accurate trials).
    mean_rt = mean(x$rt[x$error==0])
    #Compute SD RT (only accurate trials).
```

```

        sd_rt = sd(x$rt[x$error==0])
        return(c(error_rate=error_rate,mean_rt=mean_rt,sd_rt=sd_rt))
    }
)

#Compute the grand mean RT per Ss.
gmrt = ddply(
  .data = cell_stats
  , .variables = .( subnum , group )
  , .fun <- function(x){
    y = mean(x$mean_rt)
    return(c(y=y))
  }
)

#Run a purely between-Ss ANOVA on the mean_rt data.
mean_rt_perm = ezPerm(
  data = gmrt
  , dv = .(y)
  , wid = .(subnum)
  , between = .(group)
  , perms = 1e1 #1e3 or higher is best for publication
)

#Show the Permutation test.
print(mean_rt_perm)

```

 ezPlot

Function to plot data from a factorial experiment

Description

This function provides easy visualization of any given user-requested effect from factorial experiments, including purely within-Ss designs (a.k.a. "repeated measures"), purely between-Ss designs, and mixed within-and-between-Ss designs. By default, Fisher's Least Significant Difference is computed to provide error bars that facilitate visual post-hoc multiple comparisons (see Warning section below).

Usage

```

ezPlot(
  data
  , dv
  , wid
  , within = NULL
  , between = NULL
  , between_full = NULL
  , x
  , do_lines = TRUE
)

```

```

    , doBars = TRUE
    , barWidth = NULL
    , barSize = NULL
    , split = NULL
    , row = NULL
    , col = NULL
    , toNumeric = NULL
    , xLab = NULL
    , yLab = NULL
    , splitLab = NULL
    , levels = NULL
    , diff = NULL
    , reverseDiff = FALSE
  , type = 2
    , dvLevs = NULL
    , dvLabs = NULL
    , rowYFree = FALSE
)

```

Arguments

<code>data</code>	Data frame containing the data to be analyzed. OR, if multiple values are specified in <code>dv</code> , a list with as many element as values specified in <code>dv</code> , each element specifying a data frame for each <code>dv</code> in sequence.
<code>dv</code>	.() object specifying the column in <code>data</code> that contains the dependent variable. Values in this column should be of the numeric class. Multiple values will yield a plot with <code>dv</code> mapped to row.
<code>wid</code>	.() object specifying the column in <code>data</code> that contains the variable specifying the case/Ss identifier. Values in this column will be converted to factor class if necessary.
<code>within</code>	Optional .() object specifying the column(s) in <code>data</code> that contains predictor variables that are manipulated within-Ss. Values in this column will be converted to factor class if necessary.
<code>between</code>	Optional .() object specifying the column(s) in <code>data</code> that contains predictor variables that are manipulated between-Ss. Values in this column will be converted to factor class if necessary.
<code>between_full</code>	Same as <code>between</code> , but must specify the full set of between-Ss variables if <code>between</code> specifies only a subset of the design.
<code>x</code>	.() object specifying the variable to plot on the x-axis.
<code>do_lines</code>	Logical. If TRUE, lines will be plotted connecting groups of points.
<code>do_bars</code>	Logical. If TRUE, error bars will be plotted.
<code>bar_width</code>	Optional numeric value specifying custom widths for the error bar hat.
<code>bar_size</code>	Optional numeric value or vector specifying custom size of the error bars.
<code>split</code>	Optional .() object specifying a variable by which to split the data into different shapes/colors (and line types, if <code>do_lines==TRUE</code>).

row	Optional <code>.</code> () object specifying a variable by which to split the data into rows.
col	Optional <code>.</code> () object specifying a variable by which to split the data into columns.
to_numeric	Optional <code>.</code> () object specifying any variables that need to be converted to the numeric class before plotting.
x_lab	Optional character string specifying the x-axis label.
y_lab	Optional character string specifying the y-axis label.
split_lab	Optional character string specifying the key label.
levels	Optional named list where each item name matches a factored column in data that needs either reordering of levels, renaming of levels, or both. Each item should be a list containing named elements <code>new_order</code> or <code>new_names</code> or both.
diff	Optional <code>.</code> () object specifying a 2-level within-Ss variable to collapse to a difference score.
reverse_diff	Logical. If TRUE, triggers reversal of the difference collapse requested by <code>diff</code> .
type	Numeric value (either 1, 2 or 3) specifying the Sums of Squares "type" to employ when data are unbalanced (eg. when group sizes differ). See ezANOVA for details.
dv_levs	Optional character vector specifying the factor ordering of multiple values specified in <code>dv</code> .
dv_labs	Optional character vector specifying new factor labels for each of the multiple values specified in <code>dv</code> .
row_y_free	Logical. If TRUE, then rows will permit different y-axis scales.

Details

While `within` and `between` are both optional, at least one column of data must be provided to either `within` or `between`. Any numeric or character variables in data that are specified as either `wid`, `within` or `between` will be converted to a factor with a warning. Fisher's Least Significant Difference is computed as $\sqrt{2} * qt(.975, DFd) * \sqrt{MSd/N}$, where N is taken as the mean N per group in cases of unbalanced designs.

Value

A printable/modifiable `ggplot2` object.

Warning

The default error bars are Fisher's Least Significant Difference for the plotted effect, facilitating visual post-hoc multiple comparisons. Note however that in the context of mixed within-and-between-Ss designs, these bars can only be used for within-Ss comparisons.

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
#Read in the ANT data (see ?ANT).
data(ANT)
head(ANT)
ezPrecis(ANT)

#Run an ANOVA on the mean correct RT data.
mean_rt_anova = ezANOVA(
  data = ANT[ANT$error==0,]
  , dv = .(rt)
  , wid = .(subnum)
  , within = .(cue,flank)
  , between = .(group)
)

#Show the ANOVA & assumption tests.
print(mean_rt_anova)

#Plot the main effect of group.
group_plot = ezPlot(
  data = ANT[ANT$error==0,]
  , dv = .(rt)
  , wid = .(subnum)
  , between = .(group)
  , x = .(group)
  , do_lines = FALSE
  , x_lab = 'Group'
  , y_lab = 'RT (ms)'
)

#Show the plot.
print(group_plot)

#Re-plot the main effect of group, using the levels
##argument to re-arrange/rename levels of group
group_plot = ezPlot(
  data = ANT[ANT$error==0,]
  , dv = .(rt)
  , wid = .(subnum)
  , between = .(group)
  , x = .(group)
  , do_lines = FALSE
  , x_lab = 'Group'
  , y_lab = 'RT (ms)'
```

```

    , levels = list(
      group = list(
        new_order = c('Treatment', 'Control')
        , new_names = c('Treatment\nGroup', 'Control\nGroup')
      )
    )
  )
)

#Show the plot.
print(group_plot)

#Plot the cue*flank interaction.
cue_by_flank_plot = ezPlot(
  data = ANT[ANT$error==0,]
  , dv = .(rt)
  , wid = .(subnum)
  , within = .(cue,flank)
  , x = .(flank)
  , split = .(cue)
  , x_lab = 'Flanker'
  , y_lab = 'RT (ms)'
  , split_lab = 'Cue'
)

#Show the plot.
print(cue_by_flank_plot)

#Plot the cue*flank interaction by collapsing the cue effect to
##the difference between None & Double
cue_by_flank_plot2 = ezPlot(
  data = ANT[ ANT$error==0 & (ANT$cue %in% c('None', 'Double')) ,]
  , dv = .(rt)
  , wid = .(subnum)
  , within = .(flank)
  , diff = .(cue)
  , reverse_diff = TRUE
  , x = .(flank)
  , x_lab = 'Flanker'
  , y_lab = 'RT Effect (None - Double, ms)'
)

#Show the plot.
print(cue_by_flank_plot2)

#Plot the group*cue*flank interaction.
group_by_cue_by_flank_plot = ezPlot(
  data = ANT[ANT$error==0,]
  , dv = .(rt)
  , wid = .(subnum)

```

```

    , within = .(cue,flank)
    , between = .(group)
    , x = .(flank)
    , split = .(cue)
    , col = .(group)
    , x_lab = 'Flanker'
    , y_lab = 'RT (ms)'
    , split_lab = 'Cue'
)

#Show the plot.
print(group_by_cue_by_flank_plot)

#Plot the group*cue*flank interaction in both error rate and mean RT.
group_by_cue_by_flank_plot_both = ezPlot(
  data = list(
    ANT
    , ANT[ANT$error==0,]
  )
  , dv = .(error,rt)
  , wid = .(subnum)
  , within = .(cue,flank)
  , between = .(group)
  , x = .(flank)
  , split = .(cue)
  , col = .(group)
  , x_lab = 'Flanker'
  , split_lab = 'Cue'
  , dv_labs = c('ER (%)', 'RT (ms)')
  , row_y_free = TRUE
)

#Show the plot.
print(group_by_cue_by_flank_plot_both)

```

 ezPrecis

Function to obtain a structure summary of a given data frame.

Description

This function provides a structure summary of a given data frame.

Usage

```

ezPrecis(
  data
  , transpose = TRUE
)

```

Arguments

data	Data frame containing the data to be analyzed.
transpose	Logical. If TRUE (default), triggers tranposition of the resulting summary data frame (useful when there are many columns in the original data frame, leading the untransposed summary data frame to wrap).

Details

This function was inspired by the `whatis()` function from the `YaleToolkit` package.

Value

A data frame containing the descriptive information about each column in the specified data frame:

type	This row indicates the type of data R thinks is in each column. Recall that when R imports data to a data frame, each column is given a label that indicates what type of information is in that column (character, numeric, or a factor data).
missing	This row reports a count of the number of missing values in each column.
unique	This row reports a count of the number of unique values in each column.
min	This row reports the minimum value found in each column. If the column data is numeric this is straightforward. If the column data is factored, the first level is reported. If the column data is character, the alphabetically first string is reported.
max	This row reports the maximum value found in each column. If the column data is numeric this is straightforward. If the column data is factored, the last level is reported. If the column data is character, the alphabetically last string is reported.

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
#Read in the ANT2 data (see ?ANT2).
data(ANT2)
head(ANT2)

#Show a summary of the ANT2 data.
ezPrecis(ANT2)
```

ezPredict	<i>Computes predicted values from the fixed effects of a mixed effects model</i>
-----------	--

Description

This function computes the predicted values from the fixed effects of a mixed effects model.

Usage

```
ezPredict(  
  fit  
  , to_predict = NULL  
  , numeric_res = 0  
)
```

Arguments

fit	Fitted <code>lmer</code> object.
to_predict	Optional data frame containing the fixed effects design to predict. If absent, the function will assume that the full design from the provided fitted model is requested.
numeric_res	Integer value specifying the sampling resolution of any numeric fixed effect. Has no effect if non-NULL value supplied to <code>to_predict</code> . If <code>to_predict</code> is null and a numeric fixed effect is encountered in the fitted model, then predictions will be obtained at this many evenly spaced intervals between the minimum and maximum values in the original fitted data. The default value, 0, obtains predictions for each unique value found in the original data frame.

Value

A data frame containing the prediction value (and estimated variance of this value) for each cell in the fixed effects design.

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
#Read in the ANT data (see ?ANT).
data(ANT)
head(ANT)

#fit a mixed effects model to the error rate data
er_fit = lmer(
  formula = error ~ cue*flank*group + (1|subnum)
  , family = binomial
  , data = ANT
)

#obtain the predictions from the model
er_preds = ezPredict(
  fit = er_fit
)

#compute 95% CI for each prediction
er_preds$lo = er_preds$value - qnorm(.975)*sqrt(er_preds$var)
er_preds$hi = er_preds$value + qnorm(.975)*sqrt(er_preds$var)

#visualize the predictions
ggplot(
  data = er_preds
  , mapping = aes(
    x = flank
    , y = value
    , ymin = lo
    , ymax = hi
  )
)+
  geom_point(
    alpha = .75
  )+
  geom_line(
    alpha = .5
  )+
  geom_errorbar(
    alpha = .5
  )+
  facet_grid(
    cue ~ group
  )+
  labs(
    y = 'Error Rate (log odds)'
  )
)
```

Description

This function resamples data (useful when bootstrapping and used by [ezBoot](#)).

Usage

```
ezResample(
  data
  , dv
  , wid
  , within = NULL
  , between = NULL
  , resample_within = FALSE
)
```

Arguments

<code>data</code>	Data frame containing the data to be analyzed.
<code>dv</code>	.() object specifying the column in <code>data</code> that contains the dependent variable. Values in this column must be numeric.
<code>wid</code>	.() object specifying the column in <code>data</code> that contains the variable specifying the case/Ss identifier.
<code>within</code>	Optional .() object specifying one or more columns in <code>data</code> that contain predictor variables that are manipulated (or observed) within-Ss.
<code>between</code>	Optional .() object specifying one or more columns in <code>data</code> that contain predictor variables that are manipulated (or observed) between-Ss.
<code>resample_within</code>	Logical. If TRUE, and if there are multiple observations per subject within each cell of the design specified by the factorial combination of variables supplied to <code>within</code> and <code>between</code> , then these observations-within-cells are resampled with replacement.

Value

A data frame consisting of the resampled data

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```

#Read in the ANT data (see ?ANT).
data(ANT)
head(ANT)
ezPrecis(ANT)

#Bootstrap the within-cell variances
var_boots = ldply(
  .data = 1:1e1 #1e3 or higher should be used for publication
  , .fun = function(x){
    this_resample = ezResample(
      data = ANT[ANT$error==0,]
      , dv = .(rt)
      , wid = .(subnum)
      , within = .(cue,flank)
      , between = .(group)
    )
    cell_vars = ddply(
      .data = this_resample
      , .variables = .(subnum,cue,flank,group)
      , .fun = function(x){
        to_return = data.frame(
          value = var(x$rt)
        )
        return(to_return)
      }
    )
    mean_cell_vars = ddply(
      .data = cell_vars
      , .variables = .(cue,flank,group)
      , .fun = function(x){
        to_return = data.frame(
          value = mean(x$value)
        )
        return(to_return)
      }
    )
    mean_cell_vars$iteration = x
    return(mean_cell_vars)
  }
  , .progress = 'timeCI'
)

```

Description

This function provides easy computation of descriptive statistics (between-Ss means, between-Ss SD, Fisher's Least Significant Difference) for data from factorial experiments, including purely within-Ss designs (a.k.a. "repeated measures"), purely between-Ss designs, and mixed within-and-between-Ss designs.

Usage

```
ezStats(
  data
  , dv
  , wid
  , within = NULL
  , between = NULL
  , between_full = NULL
  , diff = NULL
  , reverse_diff = FALSE
  , type = 2
)
```

Arguments

<code>data</code>	Data frame containing the data to be analyzed.
<code>dv</code>	.() object specifying the column in <code>data</code> that contains the dependent variable. Values in this column should be of the numeric class.
<code>wid</code>	.() object specifying the column in <code>data</code> that contains the variable specifying the case/Ss identifier. Values in this column will be converted to factor class if necessary.
<code>within</code>	Optional .() object specifying the column(s) in <code>data</code> that contains predictor variables that are manipulated within-Ss. Values in this column will be converted to factor class if necessary.
<code>between</code>	Optional .() object specifying the column(s) in <code>data</code> that contains predictor variables that are manipulated between-Ss. Values in this column will be converted to factor class if necessary.
<code>between_full</code>	Same as <code>between</code> , but must specify the full set of between-Ss variables if <code>between</code> specifies only a subset of the design.
<code>diff</code>	Optional .() object specifying a 2-level within-Ss variable to collapse to a difference score.
<code>reverse_diff</code>	Logical. If TRUE, triggers reversal of the difference collapse requested by <code>diff</code> .
<code>type</code>	Numeric value (either 1, 2 or 3) specifying the Sums of Squares "type" to employ when data are unbalanced (eg. when group sizes differ). See ezANOVA for details.

Details

While `within` and `between` are both optional, at least one column of data must be provided to either `within` or `between`. Any numeric or character variables in `data` that are specified as either

wid, within or between will be converted to a factor with a warning. Fisher's Least Significant Difference is computed as $\sqrt{2} * qt(.975, DFd) * \sqrt{MSd/N}$, where N is taken as the mean N per group in cases of unbalanced designs.

Value

A data frame containing the descriptive statistics for the requested effect. N = number of Ss per cell. Mean = between-Ss mean. SD = between-Ss SD. FLSD = Fisher's Least Significant Difference.

Warning

The descriptives include Fisher's Least Significant Difference for the requested effect. In the context of purely within-Ss or purely between-Ss this value may be used for post-hoc multiple comparisons. Note however that in the context of mixed within-and-between-Ss designs, this value can only be used for within-Ss comparisons.

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
#Read in the ANT data (see ?ANT).
data(ANT)
head(ANT)
ezPrecis(ANT)

#Run an ANOVA on the mean correct RT data.
mean_rt_anova = ezANOVA(
  data = ANT[ANT$error==0,]
  , dv = .(rt)
  , wid = .(subnum)
  , within = .(cue,flank)
  , between = .(group)
)

#Show the ANOVA & assumption tests.
print(mean_rt_anova)

#Compute descriptives for the main effect of group.
group_descriptives = ezStats(
  data = ANT[ANT$error==0,]
  , dv = .(rt)
```

```
    , wid = .(subnum)
    , between = .(group)
  )

#Show the descriptives.
print(group_descriptives)
```

progress_time

A progress bar with time remaining estimation

Description

This function is intended to be used with the various plyr functions. See examples below.

Usage

```
progress_time()
```

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
## Not run:
l_ply(
  .data = 1:1e3
  , .fun = function(x){
    Sys.sleep(1)
  }
  , .progress = 'time'
)

## End(Not run)
```

progress_timeCI	<i>A progress bar with time remaining estimation with bootstrapped confidence intervals.</i>
-----------------	--

Description

This function is intended to be used with the various plyr functions. Bootstrap resampling obtains 10

Usage

```
progress_timeCI()
```

Author(s)

Michael A. Lawrence <Mike.Lawrence@dal.ca> To report bugs or request features, please visit: <https://github.com/mike-lawrence/ez/issues> To keep up to date on developments related to this package, join the discussion group at: <http://groups.google.com/group/ez4r>

See Also

[ANT](#), [ANT2](#), [ezANOVA](#), [ezBoot](#), [ezBootPlot](#), [ezCor](#), [ezDesign](#), [ezMixed](#), [link{ezMixedRel}](#), [ezPerm](#), [ezPlot](#), [ezPrecis](#), [ezPredict](#), [ezResample](#), [ezStats](#), [progress_time](#), [progress_timeCI](#)

Examples

```
## Not run:  
l_ply(  
  .data = 1:1e3  
  , .fun = function(x){  
    Sys.sleep(rexp(1,1))  
  }  
  , .progress = 'timeCI'  
)  
  
## End(Not run)
```

Index

*Topic **datasets**

ANT, [4](#)

ANT2, [5](#)

*Topic **package**

ez-package, [2](#)

alarm, [10](#), [13](#), [19](#), [22](#)

Anova, [6](#), [7](#)

ANT, [3](#), [4](#), [4](#), [5](#), [8](#), [10](#), [13](#), [15](#), [17](#), [20](#), [22](#), [26](#), [29](#), [30](#), [32](#), [35–37](#)

ANT2, [3](#), [4](#), [5](#), [5](#), [8](#), [10](#), [13](#), [15](#), [17](#), [20](#), [22](#), [26](#), [29](#), [30](#), [32](#), [35–37](#)

density, [14](#)

ez (ez-package), [2](#)

ez-package, [2](#)

ezANOVA, [3](#), [4](#), [5](#), [5](#), [6](#), [8](#), [10](#), [13](#), [15](#), [17](#), [20](#), [22](#), [25](#), [26](#), [29](#), [30](#), [32](#), [34–37](#)

ezBoot, [3–5](#), [8](#), [9](#), [10–13](#), [15](#), [17](#), [20](#), [22](#), [26](#), [29](#), [30](#), [32](#), [35–37](#)

ezBootPlot, [3–5](#), [8–10](#), [11](#), [13](#), [15](#), [17](#), [20](#), [22](#), [26](#), [29](#), [30](#), [32](#), [35–37](#)

ezCor, [3–5](#), [8](#), [10](#), [13](#), [13](#), [15](#), [17](#), [20](#), [22](#), [26](#), [29](#), [30](#), [32](#), [35–37](#)

ezDesign, [3–5](#), [8](#), [10](#), [13](#), [15](#), [16](#), [17](#), [20](#), [22](#), [26](#), [29](#), [30](#), [32](#), [35–37](#)

ezMixed, [3–5](#), [8](#), [10](#), [13](#), [15](#), [17](#), [18](#), [19](#), [20](#), [22](#), [26](#), [29](#), [30](#), [32](#), [35–37](#)

ezPerm, [3–5](#), [8](#), [10](#), [13](#), [15](#), [17](#), [20](#), [21](#), [22](#), [26](#), [29](#), [30](#), [32](#), [35–37](#)

ezPlot, [3–5](#), [8](#), [10](#), [13](#), [15](#), [17](#), [20](#), [22](#), [23](#), [26](#), [29](#), [30](#), [32](#), [35–37](#)

ezPrecis, [3–5](#), [8](#), [10](#), [13](#), [15](#), [17](#), [20](#), [22](#), [26](#), [28](#), [29](#), [30](#), [32](#), [35–37](#)

ezPredict, [3–5](#), [8](#), [10](#), [13](#), [15](#), [17](#), [20](#), [22](#), [26](#), [29](#), [30](#), [30](#), [32](#), [35–37](#)

ezResample, [3–5](#), [8](#), [10](#), [13](#), [15](#), [17](#), [20](#), [22](#), [26](#), [29](#), [30](#), [31](#), [32](#), [35–37](#)

ezStats, [3–5](#), [8](#), [10](#), [13](#), [15](#), [17](#), [20](#), [22](#), [26](#), [29](#), [30](#), [32](#), [33](#), [35–37](#)

family, [19](#)

lmer, [10](#), [18–20](#), [30](#)

progress_time, [3–5](#), [8](#), [10](#), [13](#), [15](#), [17](#), [20](#), [22](#), [26](#), [29](#), [30](#), [32](#), [35](#), [36](#), [36](#), [37](#)

progress_timeCI, [3–5](#), [8](#), [10](#), [13](#), [15](#), [17](#), [20](#), [22](#), [26](#), [29](#), [30](#), [32](#), [35](#), [36](#), [37](#), [37](#)