

# Package ‘RRPP’

March 30, 2021

**Title** Linear Model Evaluation with Randomized Residuals in a Permutation Procedure

**Version** 1.0.0

**Description** Linear model calculations are made for many random versions of data. Using residual randomization in a permutation procedure, sums of squares are calculated over many permutations to generate empirical probability distributions for evaluating model effects. This package is described by Collyer & Adams (2018) <doi:10.1111/2041-210X.13029>. Additionally, coefficients, statistics, fitted values, and residuals generated over many permutations can be used for various procedures including pairwise tests, prediction, classification, and model comparison. This package should provide most tools one could need for the analysis of high-dimensional data, especially in ecology and evolutionary biology, but certainly other fields, as well.

**Depends** R (>= 3.5.0)

**License** GPL (>= 3)

**URL** <https://github.com/mlcollyer/RRPP>

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.1

**Imports** Rcpp (>= 1.0.6), parallel, ape, ggplot2, Matrix

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0), dplyr, tibble

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Michael Collyer [aut, cre] (<<https://orcid.org/0000-0003-0238-2201>>),  
Dean Adams [aut] (<<https://orcid.org/0000-0001-9172-7894>>)

**Maintainer** Michael Collyer <[mlcollyer@gmail.com](mailto:mlcollyer@gmail.com)>

**Repository** CRAN

**Date/Publication** 2021-03-30 06:50:12 UTC

**Config/testthat/edition** 3

**R topics documented:**

RRPP-package	3
add.trajectories	4
add.tree	5
anova.lm.rppp	7
classify	8
coef.lm.rppp	9
convert2ggplot	10
fitted.lm.rppp	11
lm.rppp	12
manova.update	19
model.comparison	22
model.frame.lm.rppp	25
model.matrix.lm.rppp	25
motionpaths	26
ordinate	26
pairwise	30
PlethMorph	34
plot.lm.rppp	35
plot.model.comparison	36
plot.ordinate	36
plot.predict.lm.rppp	37
plot.trajectory.analysis	38
predict.lm.rppp	40
prep.lda	41
print.anova.lm.rppp	43
print.coef.lm.rppp	44
print.lm.rppp	44
print.model.comparison	45
print.ordinate	45
print.pairwise	46
print.predict.lm.rppp	46
print.summary.lm.rppp	47
print.summary.manova.lm.rppp	47
print.summary.ordinate	48
print.summary.pairwise	48
print.summary.trajectory.analysis	49
print.trajectory.analysis	49
Pupfish	50
PupfishHeads	50
residuals.lm.rppp	51
reveal.model.designs	51
rppp.data.frame	52
summary.anova.lm.rppp	53
summary.coef.lm.rppp	54
summary.lm.rppp	54
summary.manova.lm.rppp	55

summary.model.comparison . . . . .	55
summary.ordinate . . . . .	56
summary.pairwise . . . . .	56
summary.predict.lm.rpp . . . . .	58
summary.trajectory.analysis . . . . .	58
terms.lm.rpp . . . . .	59
trajectory.analysis . . . . .	60
vec.cor.matrix . . . . .	63

<b>Index</b>	<b>64</b>
--------------	-----------

---

RRPP-package	<i>Linear Model Evaluation with Randomized Residual Permutation Procedures</i>
--------------	--

---

## Description

Functions in this package allow one to evaluate linear models with residual randomization. The name, "RRPP", is an acronym for, "Randomization of Residuals in a Permutation Procedure." Through the various functions in this package, one can use randomization of residuals to generate empirical probability distributions for linear model effects, for high-dimensional data or distance matrices.

An especially useful option of this package is to fit models with either ordinary or generalized least squares estimation (OLS or GLS, respectively), using theoretic covariance matrices. Mixed linear effects can also be evaluated.

## Value

Key functions for this package:

<code>lm.rpp</code>	Fits linear models, using RRPP.
<code>anova.lm.rpp</code>	ANOVA on linear models, using RRPP, plus model comparisons.
<code>coef.lm.rpp</code>	Extract coefficients or perform test on coefficients, using RRPP.
<code>predict.lm.rpp</code>	Predict values from <code>lm.rpp</code> fits and generate bootstrapped confidence intervals.
<code>pairwise</code>	Perform pairwise tests, based on <code>lm.rpp</code> model fits.

## Author(s)

Michael Collyer and Dean Adams

---

add.trajectories      *Plot Function for RRPP*

---

## Description

Function adds trajectories to a principal component plot

## Usage

```
add.trajectories(  
  TP,  
  traj.pch = 21,  
  traj.col = 1,  
  traj.lty = 1,  
  traj.lwd = 1,  
  traj.cex = 1.5,  
  traj.bg = 1,  
  start.bg = 3,  
  end.bg = 2  
)
```

## Arguments

TP	plot object (from <a href="#">plot.trajectory.analysis</a> )
traj.pch	Plotting "character" for trajectory points. Can be a single value or vector of length equal to the number of trajectories. See <a href="#">par</a> and its description for pch.
traj.col	The color of trajectory lines. Can be a single value or vector of length equal to the number of trajectories. See <a href="#">par</a> and its description for col.
traj.lty	Trajectory line type. Can be a single value or vector of length equal to the number of trajectories. See <a href="#">par</a> and its description for lty.
traj.lwd	Trajectory line width. Can be a single value or vector of length equal to the number of trajectories. See <a href="#">par</a> and its description for lwd.
traj.cex	Trajectory point character expansion. Can be a single value or vector of length equal to the number of trajectories. See <a href="#">par</a> and its description for cex.
traj.bg	Trajectory point background. Can be a single value or vector of length equal to the number of trajectories. See <a href="#">par</a> and its description for bg.
start.bg	Trajectory point background, just the start points. Can be a single value or vector of length equal to the number of trajectories. See <a href="#">par</a> and its description for bg. Green start points are the default.
end.bg	Trajectory point background, just the end points. Can be a single value or vector of length equal to the number of trajectories. See <a href="#">par</a> and its description for bg. Red end points are the default.

## Details

The function adds trajectories to a plot made by [plot.trajectory.analysis](#). This function has a restricted set of plot parameters based on the number of trajectories to be added to the plot.

## Author(s)

Michael Collyer

## References

- Adams, D. C., and M. M. Cerney. 2007. Quantifying biomechanical motion using Procrustes motion analysis. *J. Biomech.* 40:437-444.
- Adams, D. C., and M. L. Collyer. 2007. The analysis of character divergence along environmental gradients and other covariates. *Evolution* 61:510-515.
- Adams, D. C., and M. L. Collyer. 2009. A general framework for the analysis of phenotypic trajectories in evolutionary studies. *Evolution* 63:1143-1154.
- Collyer, M. L., and D. C. Adams. 2007. Analysis of two-state multivariate phenotypic change in ecological studies. *Ecology* 88:683-692.
- Collyer, M. L., and D. C. Adams. 2013. Phenotypic trajectory analysis: comparison of shape change patterns in evolution and ecology. *Hystrix* 24: 75-83.
- Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. *Heredity*. 115:357-365.

## See Also

[plot.default](#) and [par](#)

---

add.tree

*Plot tool to add phylogenetic trees to ordination plots*

---

## Description

Function adds a tree based on a description of edges from a class phylo object to an existing plot made from an ordinate object.

## Usage

```
add.tree(  
  OP,  
  tree,  
  edge.col = 1,  
  edge.lty = 1,  
  edge.lwd = 1,  
  anc.pts = FALSE,  
  return.ancs = FALSE,  
  ...  
)
```

**Arguments**

OP	An object with class <code>plot.ordinate</code> .
tree	An object of class <code>phylo</code> .
edge.col	A single value or vector equal to the number of edges for edge colors.
edge.lty	A single value or vector equal to the number of edges for edge line type
edge.lwd	A single value or vector equal to the number of edges for edge line weight.
anc.pts	A logical value for whether to add points for ancestral values.
return.ancs	A logical value for whether ancestral values should be printed.
...	Arguments passed onto <code>points</code> , used only for ancestral points.

**Details**

With some `ordinate` plots, it might be desirable to add a tree connecting points in a prescribed way, which would be tedious using `points` or `lines`. This function will project a tree from an object of class `phylo` into a plot with class, `plot.ordinate`. Using an edges matrix from a `phylo` object, this function will systematically connect plot points with lines that pass through estimated ancestral character points in the same plot space. Ancestral states are estimated assuming a Brownian motion model of evolutionary divergence.

**Author(s)**

Michael Collyer

**See Also**

[lines](#) and [points](#)

**Examples**

```
# Examples use residuals from a regression of salamander morphological
# traits against body size (snout to vent length, SVL).
# Observations are species means and a phylogenetic covariance matrix
# describes the relatedness among observations.

data("PlethMorph")
Y <- as.data.frame(PlethMorph[c("TailLength", "HeadLength",
"Snout.ey", "BodyWidth",
"Forelimb", "Hindlimb")])
Y <- as.matrix(Y)
R <- lm.rrpp(Y ~ SVL, data = PlethMorph,
iter = 0, print.progress = FALSE)$LM$residuals

PCA <- ordinate(R, scale. = TRUE)
pc.plot <- plot(PCA, pch = 19, col = "blue")

add.tree(pc.plot, tree = PlethMorph$tree, anc.pts = TRUE,
pch = 19, cex = 0.5, col = "red")
```

---

`anova.lm.rppp`*ANOVA for lm.rppp model fits*

---

## Description

Computes an analysis of variance (ANOVA) table using distributions of random statistics from [lm.rppp](#). ANOVA can be performed on one model or multiple models. If the latter, the first model is considered a null model for comparison to other models. The ANOVA is functionally similar to a non-parametric likelihood ratio test for all null-full model comparisons. Residuals from the null model will be used to generate random pseudo-values via RRPP for evaluation of subsequent models. The permutation schedule from the null model will be used for random permutations. This function does not correct for improper null models. One must assure that the null model is nested within the other models. Illogical results can be generated if this is not the case.

## Usage

```
## S3 method for class 'lm.rppp'
anova(
  object,
  ...,
  effect.type = c("F", "cohenf", "SS", "MS", "Rsq"),
  error = NULL,
  print.progress = TRUE
)
```

## Arguments

<code>object</code>	Object from <a href="#">lm.rppp</a>
<code>...</code>	Additional <a href="#">lm.rppp</a> model fits or other arguments passed to <code>anova</code> .
<code>effect.type</code>	One of "F", "cohenf", "SS", "MS", "Rsq" to choose from which distribution of statistics to calculate effect sizes (Z). See <a href="#">lm.rppp</a> .
<code>error</code>	An optional character string to define MS error term for calculation of F values. See <a href="#">lm.rppp</a> for examples.
<code>print.progress</code>	A logical argument if multiple models are used and one wishes to view progress for sums of squares (SS) calculations.

## Author(s)

Michael Collyer

## Examples

```
# See examples for lm.rppp to see how anova.lm.rppp works in conjunction
# with other functions

data(Pupfish)
```

```

names(Pupfish)
Pupfish$logSize <- log(Pupfish$CS) # better to not have functions in formulas

# Single-Model ANOVA

# Note: one should increase RRPP iterations but a smaller number is used
# here for demonstration efficiency. Generally, iter = 999 will take less
# than 1s for this example with a modern computer.

fit <- lm.rrpp(coords ~ logSize + Sex*Pop, SS.type = "I",
data = Pupfish, print.progress = FALSE, iter = 499)
anova(fit)
anova(fit, effect.type = "MS")
anova(fit, effect.type = "Rsq")
anova(fit, effect.type = "cohenf")

# Multi-Model ANOVA (like a Likelihood Ratio Test)
fit.size <- lm.rrpp(coords ~ logSize, SS.type = "I", data = Pupfish,
print.progress = FALSE, iter = 499)
fit.sex <- lm.rrpp(coords ~ logSize + Sex, SS.type = "I", data = Pupfish,
print.progress = FALSE, iter = 499)
fit.pop <- lm.rrpp(coords ~ logSize + Pop, SS.type = "I", data = Pupfish,
print.progress = FALSE, iter = 499)
anova(fit.size, fit.sex, fit.pop) # compares two models to the first

# see lm.rrpp examples for mixed model ANOVA example and how to vary SS type

```

---

classify

*Deprecated functions in RRPP*

---

### Description

The following function has been deprecated in RRPP

### Usage

```
classify()
```

### Details

This function has been deprecated. Use [prep.lda](#) instead.



coef.lm.rppp

*coef for lm.rppp model fits***Description**

Computes ordinary or generalized least squares coefficients over the permutations of an `lm.rppp` model fit with predefined random permutations. For each coefficient vector, the Euclidean distance is calculated as an estimate of the amount of change in Y, the  $n \times p$  matrix of dependent variables; larger distances mean more change in location in the data space associated with a one unit change in the model design, for the parameter described. Random coefficients are based on either RRPP or FRPP, as defined by the `lm.rppp` model fit. If RRPP is used, all distributions of coefficient vector distances are based on appropriate null models as defined by SS type.

This function can be used to test the specific coefficients of an `lm.rppp` fit. The test statistics are the distances (d), which are also standardized (Z-scores). The Z-scores might be easier to compare, as the expected values for random distances can vary among coefficient vectors (Adams and Collyer 2016).

**Usage**

```
## S3 method for class 'lm.rppp'
coef(object, test = FALSE, confidence = 0.95, ...)
```

**Arguments**

<code>object</code>	Object from <code>lm.rppp</code>
<code>test</code>	Logical argument that if TRUE, performs hypothesis tests (Null hypothesis is vector distance = 0) for the observed coefficients. If FALSE, only the observed coefficients are returned.
<code>confidence</code>	The desired confidence limit to print with a table of summary statistics, if test = TRUE. Because distances are directionless, confidence limits are one-tailed.
<code>...</code>	Other arguments (currently none)

**Author(s)**

Michael Collyer

**Examples**

```
# See examples for lm.rppp to see how anova.lm.rppp works in conjunction
# with other functions

data(Pupfish)
names(Pupfish)
Pupfish$logSize <- log(Pupfish$CS)

fit <- lm.rppp(coords ~ logSize + Sex*Pop, SS.type = "I", data = Pupfish)
```

```
coef(fit)
coef(fit, test = TRUE, confidence = 0.99)
```

---

convert2ggplot

*Convert RRPP plots to ggplot objects*

---

## Description

Function attempts to coerce plot information from an RRPP plot object to an amenable ggplot object.

## Usage

```
convert2ggplot(object)
```

## Arguments

**object** A plot object produced from `plot.lm.rpp` and type equals either "PC" or "regression", `plot.predict.lm.rpp`, or `plot.ordinate`. Essentially, any RRPP plot except a series of diagnostic plots should work.

## Details

This function will attempt to use the plot arguments from an RRPP plot object to make a ggplot that can be additionally updated, as desired. Not all plot characteristics can be converted. For example, text arguments are not currently passed to `ggplot`, as the `text` function and `geom_text` arguments do not easily align. However, one can use text arguments produced by a RRPP plot object and `geom_text` to augment a ggplot object the way they like.

This function assumes no responsibility for arguments made by `ggplot`. It merely produces a ggplot object that should resemble an RRPP plot default. Any augmentation of ggplot objects can be done either by direct intervention of the ggplot produced or reformatting the initial RRPP plot produced. One should not expect direct correspondence between R base plot parameters and ggplot parameters. For example, error bars will generally appear as different widths, without an easy way to control them, changing from one format to the other.

## Author(s)

Michael Collyer

## Examples

```
### Linear Model Example

data(Pupfish)
fit <- lm.rpp(coords ~ log(CS) + Sex*Pop, SS.type = "I",
data = Pupfish, print.progress = FALSE, iter = 0)
```

```

# Predictions (holding alternative effects constant)

shapeDF <- expand.grid(Sex = levels(Pupfish$Sex),
  Pop = levels(Pupfish$Pop))
rownames(shapeDF) <- paste(shapeDF$Sex, shapeDF$Pop, sep = ".")

shapePreds <- predict(fit, shapeDF)
summary(shapePreds, PC = TRUE)

# Plot prediction

P <- plot(shapePreds, PC = TRUE, ellipse = TRUE)
convert2ggplot(P)

### Ordination Example

data("PlethMorph")

Y <- as.data.frame(PlethMorph[c("TailLength", "HeadLength",
  "Snout.eye", "BodyWidth",
  "Forelimb", "Hindlimb")])

Y <- as.matrix(Y)
R <- lm.rppp(Y ~ SVL, data = PlethMorph,
  iter = 0, print.progress = FALSE)$LM$residuals

# PCA (on correlation matrix)

PCA.ols <- ordinate(R, scale. = TRUE)
PCA.ols$rot
prcomp(R, scale. = TRUE)$rotation # should be the same

PCA.gls <- ordinate(R, scale. = TRUE,
  transform. = FALSE,
  Cov = PlethMorph$PhyCov)

P <- plot(PCA.gls)
convert2ggplot(P)

```

---

fitted.lm.rppp

*Extract fitted values*


---

## Description

Extract fitted values

**Usage**

```
## S3 method for class 'lm.rpp'  
fitted(object, ...)
```

**Arguments**

```
object      plot object (from lm.rpp)  
...        Arguments passed to other functions
```

**Author(s)**

Michael Collyer

**Examples**

```
# See examples for lm.rpp
```

---

lm.rpp	<i>Linear Model Evaluation with a Randomized Residual Permutation Procedure</i>
--------	---

---

**Description**

Function performs a linear model fit over many random permutations of data, using a randomized residual permutation procedure.

**Usage**

```
lm.rpp(  
  f1,  
  iter = 999,  
  turbo = FALSE,  
  seed = NULL,  
  int.first = FALSE,  
  RRPP = TRUE,  
  SS.type = c("I", "II", "III"),  
  data = NULL,  
  Cov = NULL,  
  print.progress = FALSE,  
  Parallel = FALSE,  
  ...  
)
```

**Arguments**

<code>f1</code>	A formula for the linear model (e.g., $y \sim x_1 + x_2$ ). Can also be a linear model fit from <code>lm</code> .
<code>iter</code>	Number of iterations for significance testing
<code>turbo</code>	A logical value that if TRUE, suppresses coefficient estimation in every random permutation. This will affect subsequent analyses that require random coefficients (see <code>coef.lm.rpp</code> ) but might be useful for large data sets for which only ANOVA is needed.
<code>seed</code>	An optional argument for setting the seed for random permutations of the resampling procedure. If left NULL (the default), the exact same P-values will be found for repeated runs of the analysis (with the same number of iterations). If <code>seed = "random"</code> , a random seed will be used, and P-values will vary. One can also specify an integer for specific seed values, which might be of interest for advanced users.
<code>int.first</code>	A logical value to indicate if interactions of first main effects should precede subsequent main effects
<code>RRPP</code>	A logical value indicating whether residual randomization should be used for significance testing
<code>SS.type</code>	A choice between type I (sequential), type II (hierarchical), or type III (marginal) sums of squares and cross-products computations.
<code>data</code>	A data frame for the function environment, see <code>rrpp.data.frame</code>
<code>Cov</code>	An optional argument for including a covariance matrix to address the non-independence of error in the estimation of coefficients (via GLS). If included, any weights are ignored.
<code>print.progress</code>	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.
<code>Parallel</code>	Either a logical value to indicate whether parallel processing should be used or a numeric value to indicate the number of cores to use in parallel processing via the <code>parallel</code> library. If TRUE, this argument invokes forking of all processor cores, except one. If FALSE, only one core is used. A numeric value directs the number of cores to use, but one core will always be spared.
<code>...</code>	Arguments typically used in <code>lm</code> , such as <code>weights</code> or <code>offset</code> , passed on to <code>rrpp.fit</code> for estimation of coefficients. If both <code>weights</code> and a covariance matrix are included, <code>weights</code> are ignored (since inverses of <code>weights</code> are the diagonal elements of weight matrix, used in lieu of a covariance matrix.)

**Details**

The function fits a linear model using ordinary least squares (OLS) or generalized least squares (GLS) estimation of coefficients over any number of random permutations of the data. A permutation procedure that randomizes vectors of residuals is employed. This procedure can randomize two types of residuals: residuals from null models or residuals from an intercept model. The latter is the same as randomizing full values, and is referred to as a full randomization permutation procedure (FRPP); the former uses the residuals from null models, which are defined by the type of sums of squares and cross-products (SSCP) sought in an analysis of variance (ANOVA), and is

referred to as a randomized residual permutation procedure (RRPP). Types I, II, and III SSCPs are supported.

Users define the SSCP type, the permutation procedure type, whether a covariance matrix is included (GLS estimation), and a few arguments related to computations. Results comprise observed linear model results (coefficients, fitted values, residuals, etc.), random sums of squares (SS) across permutation iterations, and other parameters for performing ANOVA and other hypothesis tests, using empirically-derived probability distributions.

lm.rppp emphasizes estimation of standard deviates of observed statistics as effect sizes from distributions of random outcomes. When performing ANOVA, using the `anova` function, the effect type (statistic choice) can be varied. See `anova.lm.rppp` for more details. Please recognize that the type of SS must be chosen prior to running `lm.rppp` and not when applying `anova` to the `lm.rppp` fit, as design matrices for the linear model must be created first. Therefore, `SS.type` is an argument for `lm.rppp` and `effect.type` is an argument for `anova.lm.rppp`. If MANOVA statistics are preferred, eigenvalues can be added with `manova.update` and statistics summarized with `summary.manova.lm.rppp`. See `manova.update` for examples.

The `coef.lm.rppp` function can be used to test the specific coefficients of an `lm.rppp` fit. The test statistics are the distances (d), which are also standardized (Z-scores). The Z-scores might be easier to compare, as the expected values for random distances can vary among coefficient vectors (Adams and Collyer 2016).

#### **ANOVA vs. MANOVA:**

Two SSCP matrices are calculated for each linear model effect, for every random permutation: R (Residuals or Random effects) and H, the difference between SSCPs for "full" and "reduced" models. (Full models contain and reduced models lack the effect tested; SSCPs are hypothesized to be the same under a null hypothesis, if there is no effect. The difference, H, would have a trace of 0 if the null hypothesis were true.) In RRPP, ANOVA and MANOVA correspond to two different ways to calculate statistics from R and H matrices.

ANOVA statistics are those that find the trace of R and H SSCP matrices before calculating subsequent statistics, including sums of squares (SS), mean squares (MS), and F-values. These statistics can be calculated with univariate data and provide univariate-like statistics for multivariate data. These statistics are dispersion measures only (covariances among variables do not contribute) and are the same as "distance-based" stats proposed by Goodall (1991) and Anderson (2001). MANOVA stats require multivariate data and are implicitly affected by variable covariances. For MANOVA, the inverse of R times H (`invR.H`) is first calculated for each effect, then eigenanalysis is performed on these matrix products. Multivariate statistics are calculated from the positive, real eigenvalues. In general, inferential conclusions will be similar with either approach, but effect sizes might differ.

ANOVA tables are generated by `anova.lm.rppp` on `lm.rppp` fits and MANOVA tables are generated by `summary.manova.lm.rppp`, after running `manova.update` on `lm.rppp` fits.

Currently, mixed model effects are only possible with \$ANOVA statistics, not \$MANOVA.

More detail is found in the vignette, ANOVA versus MANOVA.

#### **Notes for RRPP 0.5.0 and subsequent versions:**

The output from `lm.rppp` has changed, compared to previous versions. First, the \$LM component of output no longer includes both OLS and GLS statistics, when GLS fits are performed. Only GLS statistics (coefficients, residuals, fitted values) are provided and noted with a "gls." tag. GLS statistics can include those calculated when weights are input (similar to the `lm` argument).

Unlike previous versions, GLS and weighted LS statistics are not labeled differently, as weighted LS is one form of generalized LS estimation. Second, a new object, \$Models, is included in output, which contains the linear model fits (`lm` attributes) for all reduced and full models that are possible to estimate fits.

**Notes for RRPP 0.3.1 and subsequent versions:**

F-values via RRPP are calculated with residual SS (RSS) found uniquely for any model terms, as per Anderson and ter Braak (2003). This method uses the random pseudo-data generated by each term's null (reduced) model, meaning RSS can vary across terms. Previous versions used an intercept-only model for generating random pseudo-data. This generally has appropriate type I error rates but can have elevated type I error rates if the observed RSS is small relative to total SS. Allowing term by term unique RSS alleviates this concern.

**Value**

An object of class `lm.rppp` is a list containing the following

<code>call</code>	The matched call.
<code>LM</code>	Linear Model objects, including data (Y), coefficients, design matrix (X), sample size (n), number of dependent variables (p), dimension of data space (p.prime), QR decomposition of the design matrix, fitted values, residuals, weights, offset, model terms, data (model) frame, random coefficients (through permutations), random vector distances for coefficients (through permutations), whether OLS or GLS was performed, and the mean for OLS and/or GLS methods. Note that the data returned resemble a model frame rather than a data frame; i.e., it contains the values used in analysis, which might have been transformed according to the formula. The response variables are always labeled Y.1, Y.2, ..., in this frame.
<code>ANOVA</code>	Analysis of variance objects, including the SS type, random SS outcomes, random MS outcomes, random R-squared outcomes, random F outcomes, random Cohen's f-squared outcomes, P-values based on random F outcomes, effect sizes for random outcomes, sample size (n), number of variables (p), and degrees of freedom for model terms (df). These objects are used to construct ANOVA tables.
<code>PermInfo</code>	Permutation procedure information, including the number of permutations (perms), The method of residual randomization (perm.method), and each permutation's sampling frame (perm.schedule), which is a list of reordered sequences of 1:n, for how residuals were randomized.
<code>Models</code>	Reduced and full model fits for every possible model combination, based on terms of the entire model, plus the method of SS estimation.

**Author(s)**

Michael Collyer

**References**

Anderson MJ. 2001. A new method for non-parametric multivariate analysis of variance. *Austral Ecology* 26: 32-46.

Anderson MJ. and C.J.F. ter Braak. 2003. Permutation tests for multi-factorial analysis of variance. *Journal of Statistical Computation and Simulation* 73: 85-113.

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. *Heredity*. 115:357-365.

Adams, D.C. and M.L. Collyer. 2016. On the comparison of the strength of morphological integration across morphometric datasets. *Evolution*. 70:2623-2631.

Adams, D.C and M.L. Collyer. 2018. Multivariate phylogenetic anova: group-clade aggregation, biological challenges, and a refined permutation procedure. *Evolution*. 72:1204-1215.

### See Also

procD.lm and procD.pgls within geomorph; [lm](#) for more on linear model fits.

### Examples

```
# Examples use geometric morphometric data
# See the package, geomorph, for details about obtaining such data

data("PupfishHeads")
names(PupfishHeads)

# Head Size Analysis (Univariate)-----

# Note: one should increase RRPP iterations but a smaller number is
# used here for demonstration
# efficiency. Generally, iter = 999 will take less
# than 1s for this example with a modern computer.

fit <- lm.rrpp(log(headSize) ~ sex + locality/year, SS.type = "I",
data = PupfishHeads, print.progress = FALSE, iter = 199)
summary(fit)
anova(fit, effect.type = "F") # Maybe not most appropriate
anova(fit, effect.type = "Rsq") # Change effect type, but still not
# most appropriate

# Mixed-model approach (most appropriate, as year sampled is a random
# effect:

anova(fit, effect.type = "F", error = c("Residuals", "locality:year",
"Residuals"))

# Change to Type III SS

fit <- lm.rrpp(log(headSize) ~ sex + locality/year, SS.type = "III",
data = PupfishHeads, print.progress = FALSE, iter = 199)
summary(fit)
anova(fit, effect.type = "F", error = c("Residuals", "locality:year",
"Residuals"))

# Coefficients Test
```



```
coef(fit, test = TRUE)

# Predictions (holding alternative effects constant)

sizeDF <- data.frame(sex = c("Female", "Male"))
rownames(sizeDF) <- c("Female", "Male")
sizePreds <- predict(fit, sizeDF)
summary(sizePreds)
plot(sizePreds)

# Diagnostics plots of residuals

plot(fit)

# Body Shape Analysis (Multivariate) -----

data(Pupfish)
names(Pupfish)

# Note:

dim(Pupfish$coords) # highly multivariate!

# Note: one should increase RRPP iterations but they are
# not used at all here for a fast example.
# Generally, iter = 999 will take less
# than 1 second for this example with a modern computer.

fit <- lm.rpp(coords ~ log(CS) + Sex*Pop, SS.type = "I",
data = Pupfish, print.progress = FALSE, iter = 0)
summary(fit, formula = FALSE)
anova(fit)
coef(fit, test = TRUE)

# Predictions (holding alternative effects constant)

shapeDF <- expand.grid(Sex = levels(Pupfish$Sex),
Pop = levels(Pupfish$Pop))
rownames(shapeDF) <- paste(shapeDF$Sex, shapeDF$Pop, sep = ".")
shapeDF

shapePreds <- predict(fit, shapeDF)
summary(shapePreds)
summary(shapePreds, PC = TRUE)

# Plot prediction

plot(shapePreds, PC = TRUE)
plot(shapePreds, PC = TRUE, ellipse = TRUE)

# Diagnostics plots of residuals
```

```

plot(fit)

# PC-plot of fitted values

groups <- interaction(Pupfish$Sex, Pupfish$Pop)
plot(fit, type = "PC", pch = 19, col = as.numeric(groups))

# Regression-like plot

plot(fit, type = "regression", reg.type = "PredLine",
     predictor = log(Pupfish$CS), pch=19,
     col = as.numeric(groups))

# Body Shape Analysis (Distances) -----

D <- dist(Pupfish$coords) # inter-observation distances
length(D)
Pupfish$D <- D

# Note: one should increase RRPP iterations but they are
# not used at all here for a fast example. Generally,
# iter = 999 will take less than 1 second
# for this example with a modern computer.

fitD <- lm.rpp(D ~ log(CS) + Sex*Pop, SS.type = "I",
data = Pupfish, print.progress = FALSE, iter = 0)

# These should be the same:
summary(fitD, formula = FALSE)
summary(fit, formula = FALSE)

# GLS Example (Univariate) -----

data(PlethMorph)
fitOLS <- lm.rpp(TailLength ~ SVL, data = PlethMorph,
print.progress = FALSE, iter = 999)
fitGLS <- lm.rpp(TailLength ~ SVL, data = PlethMorph, Cov = PlethMorph$PhyCov,
print.progress = FALSE, iter = 999)

anova(fitOLS)
anova(fitGLS)

sizeDF <- data.frame(SVL = sort(PlethMorph$SVL))

# Prediction plots

# By specimen
plot(predict(fitOLS, sizeDF)) # Correlated error
plot(predict(fitGLS, sizeDF)) # Independent error

# With respect to independent variable (using abscissa)
plot(predict(fitOLS, sizeDF), abscissa = sizeDF) # Correlated error
plot(predict(fitGLS, sizeDF), abscissa = sizeDF) # Independent error

```

```

# GLS Example (Multivariate) -----

Y <- as.matrix(cbind(PlethMorph$TailLength,
PlethMorph$HeadLength,
PlethMorph$Snout.eye,
PlethMorph$BodyWidth,
PlethMorph$Forelimb,
PlethMorph$Hindlimb))
PlethMorph$Y <- Y
fitOLSm <- lm.rrpp(Y ~ SVL, data = PlethMorph,
print.progress = FALSE, iter = 199)
fitGLSm <- lm.rrpp(Y ~ SVL, data = PlethMorph,
Cov = PlethMorph$PhyCov,
print.progress = FALSE, iter = 199)

anova(fitOLSm)
anova(fitGLSm)

# Prediction plots

# By specimen
plot(predict(fitOLSm, sizeDF)) # Correlated error
plot(predict(fitGLSm, sizeDF)) # Independent error

# With respect to independent variable (using abscissa)
plot(predict(fitOLSm, sizeDF), abscissa = sizeDF) # Correlated error
plot(predict(fitGLSm, sizeDF), abscissa = sizeDF) # Independent error

```

---

manova.update

*MANOVA update for lm.rrpp model fits*


---

## Description

Function updates a `lm.rrpp` fit to add `$MANOVA`, which like `$ANOVA`, provides statistics or matrices typically associated with multivariate analysis of variance (MANOVA).

MANOVA statistics or sums of squares and cross-products (SSCP) matrices are calculated over the random permutations of a `lm.rrpp` fit. SSCP matrices are computed, as are the inverse of  $R$  time  $H$  ( $\text{inv}R.H$ ), where  $R$  is a SSCP for the residuals or random effects and  $H$  is the difference between SSCP matrices of full and reduced models (see below). From  $\text{inv}R.H$ , MANOVA statistics are calculated, including Roy's maximum root (eigenvalue), Pillai trace, Hotelling-Lawley trace, and Wilks lambda (via `summary.manova.lm.rrpp`).

The `manova.update` to add `$MANOVA` to `lm.rrpp` fits requires more computation time than the `$ANOVA` statistics that are computed automatically in `lm.rrpp`. Generally, the same inferential conclusions will be found with either approach, when observations outnumber response variables. For high-dimensional data (more variables than observations) data are projected into a Euclidean space of appropriate dimensions (rank of residual covariance matrix). One can vary the tolerance for

eigenvalue decay or specify the number of PCs, if a smaller set of PCs than the maximum is desired. This is advised if there is strong correlation among variables (the data space could be simplified to fewer dimensions), as spurious results are possible. Because distributions of MANOVA stats can be generated from the random permutations, there is no need to approximate F-values, like with parametric MANOVA. By restricting analysis to the real, positive eigenvalues calculated, all statistics can be calculated (but Wilks lambda, as a product but not a trace, might be less reliable as variable number approaches the number of observations).

#### ANOVA vs. MANOVA:

Two SSCP matrices are calculated for each linear model effect, for every random permutation: R (Residuals or Random effects) and H, the difference between SSCPs for "full" and "reduced" models. (Full models contain and reduced models lack the effect tested; SSCPs are hypothesized to be the same under a null hypothesis, if there is no effect. The difference, H, would have a trace of 0 if the null hypothesis were true.) In RRPP, ANOVA and MANOVA correspond to two different ways to calculate statistics from R and H matrices.

ANOVA statistics are those that find the trace of R and H SSCP matrices before calculating subsequent statistics, including sums of squares (SS), mean squares (MS), and F-values. These statistics can be calculated with univariate data and provide univariate-like statistics for multivariate data. These statistics are dispersion measures only (covariances among variables do not contribute) and are the same as "distance-based" stats proposed by Goodall (1991) and Anderson (2001). MANOVA stats require multivariate data and are implicitly affected by variable covariances. For MANOVA, the inverse of R times H (invR.H) is first calculated for each effect, then eigen-analysis is performed on these matrix products. Multivariate statistics are calculated from the positive, real eigenvalues. In general, inferential conclusions will be similar with either approach, but effect sizes might differ.

Two important differences between `manova.update` and `summary.manova` (for `lm` objects) are that `manova.update` does not attempt to normalize residual SSCP matrices (unnecessary for non-parametric statistical solutions) and (2) uses a generalized inverse of the residual SSCP, if needed, when the number of variables could render eigen-analysis problematic. This approach is consistent with covariance regularization methods that attempt to make covariance matrices positive-definite for calculating model likelihoods or multivariate statistics. If the number of observations far exceeds the number of response variables, observed statistics from `manova.update` and `summary.manova` will be quite similar. If the number of response variables approaches or exceeds the number of observations, `manova.update` statistics will be much more reliable.

ANOVA tables are generated by `anova.lm.rrpp` on `lm.rrpp` fits and MANOVA tables are generated by `summary.manova.lm.rrpp`, after running `manova.update` on `lm.rrpp` fits.

Currently, mixed model effects are only possible with `$ANOVA` statistics, not `$MANOVA`.

More detail is found in the vignette, ANOVA versus MANOVA.

#### Usage

```
manova.update(
  fit,
  error = NULL,
  tol = 1e-07,
  PC.no = NULL,
  print.progress = TRUE
)
```

**Arguments**

<code>fit</code>	Linear model fit from <code>lm.rppp</code>
<code>error</code>	An optional character string to define R matrices used to calculate <code>invR.H</code> . (Currently only Residuals can be used and this argument defaults to NULL. Future versions will update this argument.)
<code>tol</code>	A tolerance value for culling data dimensions to prevent spurious results. The distribution of eigenvalues for the data will be examined and if the decay becomes less than the tolerance, the data will be truncated to principal components ahead of this point. This will possibly prevent spurious results calculated from eigenvalues near 0. If <code>tol = 0</code> , all possible PC axes are used, which is likely not a problem if observations outnumber variables. If <code>tol = 0</code> and the number of variables exceeds the number of observations, the value of <code>tol</code> will be made slightly positive to prevent problems with eigen-analysis.
<code>PC.no</code>	A value that, if not NULL, can override the tolerance argument, and forces a desired number of data PCs to use for analysis. If a value larger than the possible number of PCs is chosen, the full compliment of PCs (the full data space) will be used. If a number larger than <code>tol</code> would permit is chosen, the minimum number of PCs between the <code>tol</code> argument and <code>PC.no</code> argument is returned.
<code>print.progress</code>	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.

**Value**

An object of class `lm.rppp` is updated to include class `manova.lm.rppp`, and the object, `$MANOVA`, which includes

<code>SSCP</code>	Terms and Model SSCP matrices.
<code>invR.H</code>	The inverse of the residuals SSCP times the H SSCP.
<code>eigs</code>	The eigenvalues of <code>invR.H</code> .
<code>e.rank</code>	Rank of the error (residuals) covariance matrix. Currently NULL only.
<code>PCA</code>	Principal component analysis of data, using either <code>tol</code> or <code>PC.no</code> .
<code>manova.pc.dims</code>	Resulting number of PC vectors in the analysis.
<code>e.rank</code>	Rank of the residual (error) covariance matrix, irrespective of the number of dimensions used for analysis.

**Author(s)**

Michael Collyer

**References**

- Goodall, C.R. 1991. Procrustes methods in the statistical analysis of shape. *Journal of the Royal Statistical Society B* 53:285-339.
- Anderson MJ. 2001. A new method for non-parametric multivariate analysis of variance. *Austral Ecology* 26: 32-46.

## Examples

```
# Body Shape Analysis (Multivariate) -----

data(Pupfish)

# Although not recommended as a practice, this example will use only
# three principal components of body shape for demonstration.
# A larger number of random permutations should also be used.

Pupfish$shape <- prcomp(Pupfish$coords)$x[, 1:3]

Pupfish$logSize <- log(Pupfish$CS)

fit <- lm.rpp(shape ~ logSize + Sex, SS.type = "I",
data = Pupfish, print.progress = FALSE, iter = 499)
summary(fit, formula = FALSE)
anova(fit) # ANOVA table

# MANOVA

fit.m <- manova.update(fit, print.progress = FALSE, tol = 0.001)
summary(fit.m, test = "Roy")
summary(fit.m, test = "Pillai")

fit.m$MANOVA$eigs$obs # observed eigenvalues
fit.m$MANOVA$SSCP$obs # observed SSCP
fit.m$MANOVA$invR.H$obs # observed invR.H

# Distributions of test statistics

summ.roy <- summary(fit.m, test = "Roy")
dens <- apply(summ.roy$rand.stats, 1, density)
par(mfcol = c(1, length(dens)))
for(i in 1:length(dens)) {
  plot(dens[[i]], xlab = "Roy max root", ylab = "Density",
type = "l", main = names(dens)[[i]])
  abline(v = summ.roy$rand.stats[1, i], col = "red")
}
par(mfcol = c(1,1))
```

## Description

Function calculates either log-likelihoods or traces of covariance matrices for comparison with respect to parameter penalties.

**Usage**

```

model.comparison(
  ...,
  type = c("cov.trace", "logLik"),
  tol = NULL,
  pc.no = NULL
)

```

**Arguments**

...	Any number of <code>lm.rpp</code> class objects for model fits to be compared.
type	An argument to choose between log-likelihood or covariance trace results
tol	If type = <code>logLik</code> , tol is a tolerance value between 0 and 1, indicating the magnitude below which components should be omitted (if standard deviations of components are less than the eigenvalue of the first component times the tolerance), for calculating the log-likelihood.
pc.no	If type = <code>logLik</code> , an optional value to indicate the number of principal components (maximum rank) to use for calculating the log-likelihood.

**Details**

The function calculates either log-likelihoods or traces of (residual) covariance matrices, plus parameter penalties, to assist in comparative model evaluation or selection. Because high-dimensional data often produce singular or ill-conditioned residual covariance matrices, this function does one of two things: 1) uses the trace of a covariance matrix rather than its determinant; or 2) provides a ridge-regularization (Warton, 2008) of the covariance matrix, only if it is determined that it is ill-conditioned. Regardless of implementation, covariance matrices are projected into a principal component (PC) space of appropriate dimensions.

The parameter penalty is based on that proposed by Bedrick and Tsai (1994), equal to  $2(pk + p(p + 1)/2)$ , where  $p$  is the appropriate dimension (not number of variables) of the covariance matrix. The parameter,  $k$ , is the rank of the model design matrix.

In the case that "logLik" is chosen for the argument, type, AIC scores are calculated. These scores may not perfectly match other packages or software that calculate AIC for multivariate data, if ridge regularization was used (and if other packages require  $p =$  the number of data variables). When choosing `logLik` as the type of comparison, it might be a good idea to adjust the tolerance or number of data principal components. The default (NULL) values will use all data dimensions to calculate log-likelihoods, which might cause problems if the number of variables exceeds the number of observations (producing singular residual covariance matrices). However, one should not reduce data dimensions haphazardly, as this can lead to poor estimates of log-likelihood. Furthermore, using the tolerance argument could result in different numbers of principal components used for each model to calculate log-likelihoods, which might be a concern for comparing models. If both `tol` and `pc.no` arguments are used, the solution will use the fewest PCs produced by either argument. Because the trace of a covariance matrix is not sensitive to matrix singularity, no PC adjustment is used for the `cov.trace` argument.

Users can construct their own tables from the results but this function does not attempt to summarize results, as interpreting results requires some arbitrary decisions. The `anova` function explicitly tests multiple models and can be used for nested model comparisons.

Results can also be plotted using the generic `plot` function.

Caution: For models with GLS estimation, the number of parameters used to estimate the covariance matrix is not taken into consideration. A generalized information criterion is currently in development.

### Value

An object of class `model.comparison` is a data frame with either log-likelihoods or covariance traces, plus parameter penalties. AIC scores might be include, if applicable

### Author(s)

Michael Collyer

### References

Bedrick, E.J., and C.L. Tsai. 1994. Model selection for multivariate regression in small samples. *Biometrics*, 226-231.

Warton, D.I., 2008. Penalized normal likelihood and ridge regularization of correlation and covariance matrices. *Journal of the American Statistical Association*. 103: 340-349.

### Examples

```
data(Pupfish)
Pupfish$logSize <- log(Pupfish$CS)
fit1 <- lm.rppp(coords ~ logSize, data = Pupfish, iter = 0,
print.progress = FALSE)
fit2 <- lm.rppp(coords ~ Pop, data = Pupfish, iter = 0,
print.progress = FALSE)
fit3 <- lm.rppp(coords ~ Sex, data = Pupfish, iter = 0,
print.progress = FALSE)
fit4 <- lm.rppp(coords ~ logSize + Sex, data = Pupfish, iter = 0,
print.progress = FALSE)
fit5 <- lm.rppp(coords ~ logSize + Pop, data = Pupfish, iter = 0,
print.progress = FALSE)
fit6 <- lm.rppp(coords ~ logSize + Sex * Pop, data = Pupfish, iter = 0,
print.progress = FALSE)

modComp1 <- model.comparison(fit1, fit2, fit3, fit4, fit5,
fit6, type = "cov.trace")
modComp2 <- model.comparison(fit1, fit2, fit3, fit4, fit5,
fit6, type = "logLik", tol = 0.01)

summary(modComp1)
summary(modComp2)

par(mfcol = c(1,2))
plot(modComp1)
plot(modComp2)
```



---

model.frame.lm.rpp *Extract model frame from a lm.rpp object*

---

**Description**

model.frame.lm.rpp returns the model frame constructed for an lm.rpp object.

**Usage**

```
## S3 method for class 'lm.rpp'  
model.frame(formula, ...)
```

**Arguments**

formula            Object from [lm.rpp](#)  
...                further arguments passed to or from other methods

**Author(s)**

Michael Collyer

---

model.matrix.lm.rpp *Extract the model design matrix from an lm.rpp object*

---

**Description**

model.matrix.lm.rpp returns the design matrix constructed for an lm.rpp object.

**Usage**

```
## S3 method for class 'lm.rpp'  
model.matrix(object, ...)
```

**Arguments**

object            Object from [lm.rpp](#)  
...                further arguments passed to or from other methods

**Author(s)**

Michael Collyer

---

 motionpaths

*Simulated motion paths*


---

**Description**

Simulated motion paths

**Author(s)**

Dean Adams

**References**

Adams, D. C., and M. L. Collyer. 2009. A general framework for the analysis of phenotypic trajectories in evolutionary studies. *Evolution* 63:1143-1154.

---

ordinate

*Ordination tool for data aligned to another matrix*


---

**Description**

Function performs a singular value decomposition of ordinary least squares (OLS) or generalized least squares (GLS) residuals, aligned to an alternative matrix, plus projection of data onto vectors obtained.

**Usage**

```
ordinate(
  Y,
  A = NULL,
  Cov = NULL,
  transform. = TRUE,
  scale. = FALSE,
  tol = NULL,
  rank. = NULL,
  newdata = NULL
)
```

**Arguments**

Y An n x p data matrix.

A An optional n x n symmetric matrix or an n x k data matrix, where k is the number of variables that could be associated with the p variables of Y. If NULL, an n x n identity matrix will be used.

Cov	An optional $n \times n$ covariance matrix to describe the non-independence among observations in $\mathbf{Y}$ , and provide a GLS-centering of data. Note that Cov and A can be the same, if one wishes to align GLS residuals to the same matrix used to obtain them. Note also that no explicit GLS-centering is performed on A. If this is desired, A should be GLS-centered beforehand.
transform.	An optional argument if a covariance matrix is provided to transform GLS-centered residuals, if TRUE. If FALSE, only GLS-centering is performed. Only if transform = TRUE (the default) can one expect the variances of ordinate scores in a principal component analysis to match eigenvalues.
scale.	a logical value indicating whether the variables should be scaled to have unit variance before the analysis takes place. The default is FALSE.
tol	A value indicating the magnitude below which components should be omitted. (Components are omitted if their standard deviations are less than or equal to tol times the standard deviation of the first component.) With the default null setting, no components are omitted (unless rank. is provided). Other settings for tol could be tol = sqrt(.Machine\$double.eps), which would omit essentially constant components, or tol = 0, to retain all components, even if redundant. This argument is exactly the same as in <a href="#">prcomp</a>
rank.	Optionally, a number specifying the maximal rank, i.e., maximal number of aligned components to be used. This argument can be set as alternative or in addition to tol, useful notably when the desired rank is considerably smaller than the dimensions of the matrix. This argument is exactly the same as in <a href="#">prcomp</a>
newdata	An optional data frame of values for the same variables of $\mathbf{Y}$ to be projected onto aligned components. This is only possible with OLS (transform. = FALSE).

## Details

The function performs a singular value decomposition,  $\mathbf{t}(\mathbf{A})\mathbf{Z} = \mathbf{U}\mathbf{D}\mathbf{t}(\mathbf{V})$ , where  $\mathbf{Z}$  is a matrix of residuals (obtained from  $\mathbf{Y}$ ; see below) and  $\mathbf{A}$  is an alignment matrix with the same number of rows as  $\mathbf{Z}$ . ( $\mathbf{t}$  indicates matrix transposition.)  $\mathbf{U}$  and  $\mathbf{V}$  are the matrices of left and right singular vectors, and  $\mathbf{D}$  is a diagonal matrix of singular values.  $\mathbf{V}$  are the vectors that describe maximized covariation between  $\mathbf{Y}$  and  $\mathbf{A}$ . If  $\mathbf{A} = \mathbf{I}$ , an  $n \times n$  identity matrix,  $\mathbf{V}$  are the eigen vectors (principal components) of  $\mathbf{Y}$ .

$\mathbf{Z}$  represents a centered and potentially standardized form of  $\mathbf{Y}$ . This function can center data via OLS or GLS means (the latter if a covariance matrix to describe the non-independence among observations is provided). If standardizing variables is preferred, then  $\mathbf{Z}$  both centers and scales the vectors of  $\mathbf{Y}$  by their standard deviations.

Data are projected onto aligned vectors,  $\mathbf{ZV}$ . If a GLS computation is made, the option to transform centered values (residuals) before projection is available. This is required for orthogonal projection, but from a transformed data space. Not transforming residuals maintains the Euclidean distances among observations and the OLS multivariate variance, but the projection is oblique (scores can be correlated).

The versatility of using an alignment approach is that alternative data space rotations are possible. Principal components are thus the vectors that maximize variance with respect to the data, themselves, but "components" of (co)variation can be described for any inter-matrix relationship, including phylogenetic signal, ecological signal, ontogenetic signal, size allometry, etc. More details are provided in Collyer and Adams (2021).

Much of this function is consistent with the `prcomp` function, except that centering data is not an option (it is required).

**SUMMARY STATISTICS:** For principal component plots, the traditional statistics to summarize the analysis include eigenvalues (variance by component), proportion of variance by component, and cumulative proportion of variance. When data are aligned to an alternative matrix, the statistics are less straightforward. A summary of of such an analysis (performed with `summary.ordinate`) will produce these additional statistics:

- **Singular Value** Rather than eigenvalues, the singular values from singular value decomposition of the cross-product of the scaled alignment matrix and the data.
- **Proportion of Covariance** Each component's singular value divided by the sum of singular values. The cumulative proportion is also returned. Note that these values do not explain the amount of covariance between the alignment matrix and data, but explain the distribution of the covariance. Large proportions can be misleading.
- **RV by Component** The partial RV statistic by component. Cumulative values are also returned. The sum of partial RVs is Escoffier's RV statistic, which measures the amount of covariation between the alignment matrix and data. Caution should be used in interpreting these values, which can vary with the number of observations and number of variables. However, the RV is more reliable than proportion of singular value for interpretation of the strength of linear association for aligned components. (It is most analogous to proportion of variance for principal components.)

## Value

An object of class `ordinate` is a list containing the following

<code>x</code>	Aligned component scores for all observations
<code>xn</code>	Optional projection of new data onto components.
<code>d</code>	The portion of the squared singular values attributed to the aligned components.
<code>sdev</code>	Standard deviations of <code>d</code> ; i.e., the scale of the components.
<code>rot</code>	The matrix of variable loadings, i.e. the singular vectors, <b>V</b> .
<code>center</code>	The OLS or GLS means vector used for centering.
<code>transform</code>	Whether GLS transformation was used in projection of residuals (only possible in conjunction with GLS-centering).
<code>scale</code>	The scaling used, or <code>FALSE</code> .
<code>alignment</code>	Whether data were aligned to principal axes or the name of another matrix.
<code>GLS</code>	A logical value to indicate if GLS-centering and projection was used.

## Author(s)

Michael Collyer

## References

- Collyer, M.L. and D.C. Adams. 2021. Phylogenetically-aligned Component Analysis. *Methods in Ecology and evolution*. In press.
- Revell, L. J. 2009. Size-correction and principal components for interspecific comparative studies. *Evolution*, 63:3258-3268.

**See Also**

[plot.ordinate](#), [prcomp](#), `gm.prcomp` within `geomorph`

**Examples**

```
# Examples use residuals from a regression of salamander
# morphological traits against body size (snout to vent length, SVL).
# Observations are species means and a phylogenetic covariance matrix
# describes the relatedness among observations.

data("PlethMorph")
Y <- as.data.frame(PlethMorph[c("TailLength", "HeadLength",
"Snout.eye", "BodyWidth",
"Forelimb", "Hindlimb")])
Y <- as.matrix(Y)
R <- lm.rpp(Y ~ SVL, data = PlethMorph,
iter = 0, print.progress = FALSE)$LM$residuals

# PCA (on correlation matrix)

PCA.ols <- ordinate(R, scale. = TRUE)
PCA.ols$rot
prcomp(R, scale. = TRUE)$rotation # should be the same

# phyPCA (sensu Revell, 2009)
# with projection of untransformed residuals (Collyer & Adams 2020)

PCA.gls <- ordinate(R, scale. = TRUE,
transform. = FALSE,
Cov = PlethMorph$PhyCov)

# phyPCA with transformed residuals (orthogonal projection,
# Collyer & Adams 2020)

PCA.t.gls <- ordinate(R, scale. = TRUE,
transform. = TRUE,
Cov = PlethMorph$PhyCov)

# Align to phylogenetic signal (in each case)

PaCA.ols <- ordinate(R, A = PlethMorph$PhyCov, scale. = TRUE)

PaCA.gls <- ordinate(R, A = PlethMorph$PhyCov,
scale. = TRUE,
transform. = FALSE,
Cov = PlethMorph$PhyCov)

PaCA.t.gls <- ordinate(R, A = PlethMorph$PhyCov,
scale. = TRUE,
transform. = TRUE,
Cov = PlethMorph$PhyCov)
```

```

# Summaries

summary(PCA.ols)
summary(PCA.gls)
summary(PCA.t.gls)
summary(PaCA.ols)
summary(PaCA.gls)
summary(PaCA.t.gls)

# Plots

par(mfrow = c(2,3))
plot(PCA.ols, main = "PCA OLS")
plot(PCA.gls, main = "PCA GLS")
plot(PCA.t.gls, main = "PCA t-GLS")
plot(PaCA.ols, main = "PaCA OLS")
plot(PaCA.gls, main = "PaCA GLS")
plot(PaCA.t.gls, main = "PaCA t-GLS")
par(mfrow = c(1,1))

```

---

pairwise

*Pairwise comparisons of lm.rpp fits*


---

### Description

Function generates distributions of pairwise statistics for a `lm.rpp` fit and returns important statistics for hypothesis tests.

### Usage

```

pairwise(
  fit,
  fit.null = NULL,
  groups,
  covariate = NULL,
  print.progress = FALSE
)

```

### Arguments

<code>fit</code>	A linear model fit using <code>lm.rpp</code> .
<code>fit.null</code>	An alternative linear model fit to use as a null model for RRPP, if the null model of the fit is not desired. Note, for FRPP this argument should remain <code>NULL</code> and FRPP must be established in the <code>lm.rpp</code> fit ( <code>RRPP = FALSE</code> ). If the null model is uncertain, using <code>reveal.model.designs</code> will help elucidate the inherent null model used.

groups	A factor or vector that is coercible into a factor, describing the levels of the groups for which to find LS means or slopes. Normally this factor would be part of the model fit, but it is not necessary for that to be the case in order to obtain results.
covariate	A numeric vector for which to calculate slopes for comparison. If NULL, LS means will be calculated instead of slopes. Normally this variable would be part of the model fit, but it is not necessary for that to be the case in order to obtain results.
print.progress	If a null model fit is provided, a logical value to indicate whether analytical results progress should be printed on screen. Unless large data sets are analyzed, this argument is probably not helpful.

### Details

Based on an `lm.rpp` fit, this function will find fitted values over all permutations and based on a grouping factor, calculate either least squares (LS) means or slopes, and pairwise statistics among them. Pairwise statistics have multiple flavors, related to vector attributes:

- **Distance between vectors, "dist"** Vectors for LS means or slopes originate at the origin and point to some location, having both a magnitude and direction. A distance between two vectors is the inner-product of the vector difference, i.e., the distance between their endpoints. For LS means, this distance is the difference between means. For multivariate slope vectors, this is the difference in location between estimated change for the dependent variables, per one-unit change of the covariate considered. For univariate slopes, this is the absolute difference between slopes.
- **Vector correlation, "VC"** If LS mean or slope vectors are scaled to unit size, the vector correlation is the inner-product of the scaled vectors. The arccosine (`acos`) of this value is the angle between vectors, which can be expressed in radians or degrees. Vector correlation indicates the similarity of vector orientation, independent of vector length.
- **Difference in vector lengths, "DL"** If the length of a vector is an important attribute – e.g., the amount of multivariate change per one-unit change in a covariate – then the absolute value of the difference in vector lengths is a practical statistic to compare vector lengths. Let  $d_1$  and  $d_2$  be the distances (length) of vectors. Then  $|d_1 - d_2|$  is a statistic that compares their lengths. For slope vectors, this is a comparison of rates.
- **Variance, "var"** Vectors of residuals from a linear model indicate can express the distances of observed values from fitted values. Mean squared distances of values (variance), by group, can be used to measure the amount of dispersion around estimated values for groups. Absolute differences between variances are used as test statistics to compare mean dispersion of values among groups. Variance degrees of freedom equal  $n$ , the group size, rather than  $n-1$ , as the purpose is to compare mean dispersion in the sample. (Additionally, tests with one subject in a group are possible, or at least not a hindrance to the analysis.)

The `summary.pairwise` function is used to select a test statistic for the statistics described above, as "dist", "VC", "DL", and "var", respectively. If vector correlation is tested, the `angle` type argument can be used to choose between radians and degrees.

The null model is defined via `lm.rpp`, but one can also use an alternative null model as an optional argument. In this case, residual randomization in the permutation procedure (RRPP) will be

performed using the alternative null model to generate fitted values. If full randomization of values (FRPP) is preferred, it must be established in the `lm.rpp` fit and an alternative model should not be chosen. If one is unsure about the inherent null model used if an alternative is not specified as an argument, the function `reveal.model.designs` can be used.

Observed statistics, effect sizes, P-values, and one-tailed confidence limits based on the confidence requested will be summarized with the `summary.pairwise` function. Confidence limits are inherently one-tailed as the statistics are similar to absolute values. For example, a distance is analogous to an absolute difference. Therefore, the one-tailed confidence limits are more akin to two-tailed hypothesis tests. (A comparable example is to use the absolute value of a t-statistic, in which case the distribution has a lower bound of 0.)

#### Notes for RRPP 0.6.2 and subsequent versions:

In previous versions of `pairwise`, codes `summary.pairwise` had three test types: "dist", "VC", and "var". When one chose "dist", for LS mean vectors, the statistic was the inner-product of the vector difference. For slope vectors, "dist" returned the absolute value of the difference between vector lengths, which is "DL" in 0.6.2 and subsequent versions. This update uses the same calculation, irrespective of vector types. Generally, "DL" is the same as a contrast in rates for slope vectors, but might not have much meaning for LS means. Likewise, "dist" is the distance between vector endpoints, which might make more sense for LS means than slope vectors. Nevertheless, the user has more control over these decisions with version 0.6.2 and subsequent versions.

#### Value

An object of class `pairwise` is a list containing the following

<code>LS.means</code>	LS means for groups, across permutations.
<code>slopes</code>	Slopes for groups, across permutations.
<code>means.dist</code>	Pairwise distances between means, across permutations.
<code>means.vec.cor</code>	Pairwise vector correlations between means, across permutations.
<code>means.lengths</code>	LS means vector lengths, by group, across permutations.
<code>means.diff.length</code>	Pairwise absolute differences between mean vector lengths, across permutations.
<code>slopes.dist</code>	Pairwise distances between slopes (end-points), across permutations.
<code>slopes.vec.cor</code>	Pairwise vector correlations between slope vectors, across permutations.
<code>slopes.lengths</code>	Slope vector lengths, by group, across permutations.
<code>slopes.diff.length</code>	Pairwise absolute differences between slope vector lengths, across permutations.
<code>n</code>	Sample size
<code>p</code>	Data dimensions; i.e., variable number
<code>PermInfo</code>	Information for random permutations, passed on from <code>lm.rpp</code> fit and possibly modified if an alternative null model was used.

#### Author(s)

Michael Collyer



## References

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. *Heredity*. 115:357-365.

Adams, D.C and M.L. Collyer. 2018. Multivariate phylogenetic ANOVA: group-clade aggregation, biological challenges, and a refined permutation procedure. *Evolution*. In press.

## See Also

[lm.rppp](#)

## Examples

```
# Examples use geometric morphometric data on pupfishes
# See the package, geomorph, for details about obtaining such data

# Body Shape Analysis (Multivariate) -----

data("Pupfish")

# Note:

dim(Pupfish$coords) # highly multivariate!

Pupfish$logSize <- log(Pupfish$CS)

# Note: one should use all dimensions of the data but with this
# example, there are many. Thus, only three principal components
# will be used for demonstration purposes.

Pupfish$Y <- prcomp(Pupfish$coords)$x[, 1:3]

## Pairwise comparisons of LS means

# Note: one should increase RRPP iterations but a
# smaller number is used here for demonstration
# efficiency. Generally, iter = 999 will take less
# than 1s for these examples with a modern computer.

fit1 <- lm.rppp(Y ~ logSize + Sex * Pop, SS.type = "I",
data = Pupfish, print.progress = FALSE, iter = 499)
summary(fit1, formula = FALSE)
anova(fit1)

pup.group <- interaction(Pupfish$Sex, Pupfish$Pop)
pup.group
PW1 <- pairwise(fit1, groups = pup.group)
PW1

# distances between means
summary(PW1, confidence = 0.95, test.type = "dist")
```

```

summary(PW1, confidence = 0.95, test.type = "dist", stat.table = FALSE)

# absolute difference between mean vector lengths
summary(PW1, confidence = 0.95, test.type = "DL")

# correlation between mean vectors (angles in degrees)
summary(PW1, confidence = 0.95, test.type = "VC",
  angle.type = "deg")

# Can also compare the dispersion around means
summary(PW1, confidence = 0.95, test.type = "var")

## Pairwise comparisons of slopes

fit2 <- lm.rppp(Y ~ logSize * Sex * Pop, SS.type = "I",
  data = Pupfish, print.progress = FALSE, iter = 199)
summary(fit2, formula = FALSE)
anova(fit1, fit2)

# Using a null fit that excludes all factor-covariate
# interactions, not just the last one

PW2 <- pairwise(fit2, fit.null = fit1, groups = pup.group,
  covariate = Pupfish$logSize, print.progress = FALSE)
PW2

# distances between slope vectors (end-points)
summary(PW2, confidence = 0.95, test.type = "dist")
summary(PW2, confidence = 0.95, test.type = "dist", stat.table = FALSE)

# absolute difference between slope vector lengths
summary(PW2, confidence = 0.95, test.type = "DL")

# correlation between slope vectors (and angles)
summary(PW2, confidence = 0.95, test.type = "VC",
  angle.type = "deg")

# Can also compare the dispersion around group slopes
summary(PW2, confidence = 0.95, test.type = "var")

```

---

## Description

Data for 37 species of plethodontid salamanders. Variables include snout to vent length (SVL) as species size, tail length, head length, snout to eye length, body width, forelimb length, and hind limb length, all measured in mm. A grouping variable is also included for functional guild size. The data set also includes a phylogenetic covariance matrix based on a Brownian model of evolution, to assist in generalized least squares (GLS) estimation.

**Details**

The covariance matrix was estimated with the `vcv.phylo` function of the R package, `ape`, based on the tree described in Adams and Collyer (2018).

**Author(s)**

Michael Collyer and Dean Adams

**References**

Adams, D.C and Collyer, M.L. 2018. Multivariate phylogenetic anova: group-clade aggregation, biological challenges, and a refined permutation procedure. *Evolution*, 72: 1204-1215.

---

plot.lm.rppp	<i>Plot Function for RRPP</i>
--------------	-------------------------------

---

**Description**

Plot Function for RRPP

**Usage**

```
## S3 method for class 'lm.rppp'
plot(
  x,
  type = c("diagnostics", "regression", "PC"),
  predictor = NULL,
  reg.type = c("PredLine", "RegScore"),
  ...
)
```

**Arguments**

<code>x</code>	plot object (from <code>lm.rppp</code> )
<code>type</code>	Indicates which type of plot, choosing among diagnostics, regression, or principal component plots. Diagnostic plots are similar to <code>lm</code> diagnostic plots, but for multivariate data. Regression plots plot multivariate dispersion in some fashion against predictor values. PC plots project data onto the eigenvectors of the covariance matrix for fitted values.
<code>predictor</code>	An optional vector if "regression" plot type is chosen, and is a variable likely used in <code>lm.rppp</code> . This vector is a vector of covariate values equal to the number of observations.
<code>reg.type</code>	If "regression" is chosen for plot type, this argument indicates whether prediction line (PredLine) or regression score (RegScore) plotting is performed. For explanation of prediction line, see Adams and Nistri (2010). For explanation of regression score, see Drake and Klingenberg (2008).
<code>...</code>	other arguments passed to <code>plot</code> (helpful to employ different colors or symbols for different groups). See <code>plot.default</code> and <code>par</code>

**Author(s)**

Michael Collyer

**References**

- Drake, A. G., and C. P. Klingenberg. 2008. The pace of morphological change: Historical transformation of skull shape in St Bernard dogs. *Proc. R. Soc. B.* 275:71-76.
- Adams, D. C., and A. Nistri. 2010. Ontogenetic convergence and evolution of foot morphology in European cave salamanders (Family: Plethodontidae). *BMC Evol. Biol.* 10:1-10.

---

plot.model.comparison *Plot Function for RRPP*

---

**Description**

Plot Function for RRPP

**Usage**

```
## S3 method for class 'model.comparison'
plot(x, ...)
```

**Arguments**

x plot object (from [model.comparison](#))

... other arguments passed to plot (helpful to employ different colors or symbols for different groups). See [plot.default](#) and [par](#)

**Author(s)**

Michael Collyer

---

plot.ordinate *Plot Function for RRPP*

---

**Description**

Plot Function for RRPP

**Usage**

```
## S3 method for class 'ordinate'
plot(x, axis1 = 1, axis2 = 2, flip = NULL, ...)
```

**Arguments**

x	An object of class <code>ordinate</code>
axis1	A value indicating which component should be displayed as the X-axis (default = C1)
axis2	A value indicating which component should be displayed as the Y-axis (default = C2)
flip	An argument that if not NULL can be used to flip components in the plot. The values need to match axis1 or axis2. For example, if axis1 = 3 and axis2 = 4, flip = 1 will not change either axis; flip = 3 will flip only the horizontal axis; flip = c(3, 4) will flip both axes.
...	other arguments passed to plot (helpful to employ different colors or symbols for different groups). See

**Value**

An object of class "plot.ordinate" is a list with components that can be used in other plot functions, such as the type of plot, points, a group factor, and other information depending on the plot parameters used.

**Author(s)**

Michael Collyer

---

plot.predict.lm.rppp *Plot Function for RRPP*

---

**Description**

Plot Function for RRPP

**Usage**

```
## S3 method for class 'predict.lm.rppp'
plot(x, PC = FALSE, ellipse = FALSE, abscissa = NULL, label = TRUE, ...)
```

**Arguments**

x	plot object (from <code>predict.lm.rppp</code> )
PC	A logical argument for whether the data space should be rotated to its principal components
ellipse	A logical argument to change error bars to ellipses in multivariate plots. It has no function for univariate plots or is abscissa is not NULL.

abscissa	An optional vector (numeric or factor) equal in length to predictions to use for plotting as the abscissa (x-axis), in which case predictions are the ordinate (y-axis). This might be helpful if predictions are made for a continuous independent variable. The abscissa would be the same variable used to make predictions (and can be the data.frame used for newdata in <a href="#">predict.lm.rpp</a> ).
label	A logical argument for whether points should be labeled (in multivariate plots).
...	other arguments passed to plot, arrows, points, or text (helpful to employ different colors or symbols for different groups). See <a href="#">plot.default</a> , <a href="#">arrows</a> , <a href="#">points</a> , <a href="#">par</a> , and <a href="#">text</a>

**Author(s)**

Michael Collyer

**Examples**

```
# See \link{lm.rpp} for examples.
```

---

```
plot.trajectory.analysis
      Plot Function for RRPP
```

---

**Description**

Function generates a principal component plot for trajectories

**Usage**

```
## S3 method for class 'trajectory.analysis'
plot(x, ...)
```

**Arguments**

x	plot object (from <a href="#">trajectory.analysis</a> )
...	other arguments passed to plot (helpful to employ different colors or symbols for different groups). See <a href="#">plot.default</a> and <a href="#">par</a>

**Details**

The function calculates and plots principal components of fitted values from [lm.rpp](#) that are passed onto [trajectory.analysis](#), and projects data onto them. This function is a set.up, and [add.trajectories](#) is needed to add trajectories to the plot. By having two stages of control, the plotting functions are more flexible. This function also returns plotting information that can be valuable for making individualized plots, if [add.trajectories](#) is not preferred.

**Value**

If an object is assigned, it will return:

- `pca` Principal component analysis performed using [prcomp](#).
- `pc.points` Principal component scores for all data.
- `trajectory.analysis`  
Trajectory analysis passed on.
- `trajectories` `pca` Observed trajectories projected onto principal components.

**Author(s)**

Michael Collyer

**References**

Adams, D. C., and M. M. Cerney. 2007. Quantifying biomechanical motion using Procrustes motion analysis. *J. Biomech.* 40:437-444.

Adams, D. C., and M. L. Collyer. 2007. The analysis of character divergence along environmental gradients and other covariates. *Evolution* 61:510-515.

Adams, D. C., and M. L. Collyer. 2009. A general framework for the analysis of phenotypic trajectories in evolutionary studies. *Evolution* 63:1143-1154.

Collyer, M. L., and D. C. Adams. 2007. Analysis of two-state multivariate phenotypic change in ecological studies. *Ecology* 88:683-692.

Collyer, M. L., and D. C. Adams. 2013. Phenotypic trajectory analysis: comparison of shape change patterns in evolution and ecology. *Hystrix* 24: 75-83.

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. *Heredity*. 115:357-365.

**See Also**

[plot.default](#) and [par](#)

**Examples**

```
# See \code{\link{trajectory.analysis}} for examples
```

---

predict.lm.rppp      *predict for lm.rppp model fits*

---

### Description

Computes predicted values from an `lm.rppp` model fit, using bootstrapped residuals to generate confidence intervals. (Residuals are the residuals of the `lm.rppp` fit, not its null model. The bootstrap procedure resamples residual vectors with replacement.) The bootstrap permutations use the same number of iterations and seed as used in the `lm.rppp` model fit. A `predict.lm.rppp` object can be plotted using various options. See `plot.predict.lm.rppp`.

Note that if data offsets are used (if the `offset` argument is used when fitting a `lm.rppp` model), they are ignored for estimating coefficients over iterations. Offsets are subtracted from data in `lm` and added to predicted values in `predict.lm`, effectively adjusted the intercept and then un-adjusting it for predictions. This causes problems if the `newdata` have a different number of observations than the original model fit.

### Usage

```
## S3 method for class 'lm.rppp'
predict(object, newdata = NULL, confidence = 0.95, ...)
```

### Arguments

<code>object</code>	Object from <code>lm.rppp</code> .
<code>newdata</code>	Data frame of either class <code>data.frame</code> or <code>rrpp.data.frame</code> . If null, the data frame from the <code>lm.rppp</code> fit will be used, effectively calculating all fitted values and their confidence intervals. If a numeric variable is missing from <code>newdata</code> , an attempt to average the values will be made in prediction; i.e., least squares means for factor levels can be found. All factors used in the <code>lm.rppp</code> fit should be represented in the <code>newdata</code> data frame, with appropriate factor levels.
<code>confidence</code>	The desired confidence interval level for prediction.
<code>...</code>	Other arguments (currently none)

### Author(s)

Michael Collyer

### Examples

```
# See examples for lm.rppp to see how predict.lm.rppp works in conjunction
# with other functions

data(Pupfish)

# CS is centroid (fish) size
fit <- lm.rppp(coords ~ log(CS) + Sex*Pop,
SS.type = "I", data = Pupfish, iter = 499)
```



```

# Predictions (holding alternative effects constant)

shapeDF <- expand.grid(Sex = levels(Pupfish$Sex), Pop = levels(Pupfish$Pop))
rownames(shapeDF) <- paste(shapeDF$Sex, shapeDF$Pop, sep = ".")
shapeDF

shapePreds <- predict(fit, shapeDF)
summary(shapePreds)
summary(shapePreds, PC = TRUE)

shapePreds99 <- predict(fit, shapeDF, confidence = 0.99)
summary(shapePreds99, PC = TRUE)

# Plot prediction

plot(shapePreds, PC = TRUE)
plot(shapePreds, PC = TRUE, ellipse = TRUE)
plot(shapePreds99, PC = TRUE)
plot(shapePreds99, PC = TRUE, ellipse = TRUE)

```

---

```
prep.lda
```

---

*Linear discriminant function for lm.rpp model fits*

---

## Description

Function creates arguments for `lda` or `qda` from a `lm.rpp` fit.

## Usage

```

prep.lda(
  fit,
  tol = 1e-07,
  PC.no = NULL,
  newdata = NULL,
  inherent.groups = FALSE,
  ...
)

```

## Arguments

<code>fit</code>	A linear model fit using <code>lm.rpp</code> .
<code>tol</code>	A tolerance used to decide if the matrix of data is singular. This value is passed onto both <code>lda</code> and <code>prcomp</code> , internally.
<code>PC.no</code>	An optional argument to define the specific number of principal components (PC) used in analysis. The minimum of this value or the number of PCs resulting from the <code>tol</code> argument will be used.

newdata	An optional matrix (or object coercible to a matrix) for classification. If NULL, all observed data are used.
inherent.groups	A logical argument in case one wishes to have the inherent groups in the model fit revealed. If TRUE, no other analysis will be done than to reveal the groups. This argument should always be FALSE to perform a classification analysis.
...	Arguments passed to <code>lda</code> . See <code>lda</code> for details

## Details

This function uses a `lm.rppp` fit to produce the data and the groups to use in `lda` or `qda`. There are two general purposes of this function that are challenging when using `lda`, directly. First, this function finds the inherent groups in the `lm.rppp` fit, based on factor levels. Second, this function finds pseudodata - rather than the observed data - that involve either or both a principal component projection with appropriate (or user-prescribed) dimensions and a transformation. The principal component projection incorporates GLS mean-centering, where appropriate. Transformation involves holding non-grouping model terms constant. This is accomplished by using the fitted values from the `lm.rppp` fit and the residuals of a `lm.rppp` fit with grouping factors, alone. When, the `lm.rppp` fit contains only grouping factors, this function produces raw data projected on principal components.

Regardless of variables input, data are projected onto PCs. The purpose of this function is to predict group association, and working in PC space facilitates this objective.

This is a new function and not all limits and scenarios have been tested before its release. Please report any issues or limitations or strange results to the maintainer.

### Notes for RRPP 0.5.0 and subsequent versions:

Prior to version 0.5.0, the function, `classify`, was available. This function has been deprecated. It mimicked `lda` with added features that are largely retained with `prep.lda`. However, `prep.lda` facilitates the much more diverse options available with `lda`.

## Value

A list of arguments that can be passed to `lda`. As a minimum, these arguments include `$x`, `$grouping`, and `$tol`. If `newdata` is not NULL, `$newdata`, using the same transformation and PCs as for the data, will also be included.

## Author(s)

Michael Collyer

## See Also

[lda](#), [predict.lda](#), [qda](#), [predict.qda](#)

## Examples

```
# Using the Pupfish data (see lm.rppp help for more detail)
```

```
data(Pupfish)
Pupfish$logSize <- log(Pupfish$CS)
fit <- lm.rpp(coords ~ logSize + Sex * Pop, SS.type = "I",
data = Pupfish, print.progress = FALSE, iter = 0)

prep.lda(fit, inherent.groups = TRUE) # see groups available
lda.args <- prep.lda(fit, CV = TRUE, PC.no = 6)
lda.args$x
lda.args$grouping

# not run:
# library(MASS)
# LDA <- do.call(lda, lda.args)
# LDA$posterior
# table(lda.args$grouping, LDA$class)
```

---

`print.anova.lm.rpp`    *Print/Summary Function for RRPP*

---

## **Description**

Print/Summary Function for RRPP

## **Usage**

```
## S3 method for class 'anova.lm.rpp'
print(x, ...)
```

## **Arguments**

<code>x</code>	print/summary object (from <a href="#">lm.rpp</a> )
<code>...</code>	other arguments passed to print/summary

## **Author(s)**

Michael Collyer

---

```
print.coef.lm.rpp      Print/Summary Function for RRPP
```

---

**Description**

Print/Summary Function for RRPP

**Usage**

```
## S3 method for class 'coef.lm.rpp'
print(x, ...)
```

**Arguments**

x	Object from <a href="#">coef.lm.rpp</a>
...	Other arguments passed onto coef.lm.rpp

**Author(s)**

Michael Collyer

---

```
print.lm.rpp          Print/Summary Function for RRPP
```

---

**Description**

Print/Summary Function for RRPP

**Usage**

```
## S3 method for class 'lm.rpp'
print(x, ...)
```

**Arguments**

x	print/summary object (from <a href="#">lm.rpp</a> )
...	other arguments passed to print/summary

**Author(s)**

Michael Collyer

---

print.model.comparison

*Print/Summary Function for RRPP*

---

### **Description**

Print/Summary Function for RRPP

### **Usage**

```
## S3 method for class 'model.comparison'  
print(x, ...)
```

### **Arguments**

x                    Object from [model.comparison](#)  
...                   Other arguments passed onto model.comparison

### **Author(s)**

Michael Collyer

---

print.ordinate

*Print/Summary Function for RRPP*

---

### **Description**

Print/Summary Function for RRPP

### **Usage**

```
## S3 method for class 'ordinate'  
print(x, ...)
```

### **Arguments**

x                    Object from [ordinate](#)  
...                   Other arguments passed onto print.ordinate

### **Author(s)**

Michael Collyer

---

print.pairwise      *Print/Summary Function for RRPP*

---

**Description**

Print/Summary Function for RRPP

**Usage**

```
## S3 method for class 'pairwise'  
print(x, ...)
```

**Arguments**

x                    Object from [pairwise](#)  
...                   Other arguments passed onto pairwise

**Author(s)**

Michael Collyer

---

print.predict.lm.rrpp      *Print/Summary Function for RRPP*

---

**Description**

Print/Summary Function for RRPP

**Usage**

```
## S3 method for class 'predict.lm.rrpp'  
print(x, PC = FALSE, ...)
```

**Arguments**

x                    Object from [predict.lm.rrpp](#)  
PC                   Logical argument for whether to use predicted values rotated to their PCs  
...                   Other arguments passed onto predict.lm.rrpp

**Author(s)**

Michael Collyer

---

print.summary.lm.rpp *Print/Summary Function for RRPP*

---

### Description

Print/Summary Function for RRPP

### Usage

```
## S3 method for class 'summary.lm.rpp'  
print(x, ...)
```

### Arguments

x                    print/summary object (from [summary.lm.rpp](#))  
...                   other arguments passed to print/summary

### Author(s)

Michael Collyer

---

print.summary.manova.lm.rpp  
*Print/Summary Function for RRPP*

---

### Description

Print/Summary Function for RRPP

### Usage

```
## S3 method for class 'summary.manova.lm.rpp'  
print(x, ...)
```

### Arguments

x                    Object from [summary.manova.lm.rpp](#)  
...                   Other arguments passed onto summary.manova.lm.rpp

### Author(s)

Michael Collyer

---

```
print.summary.ordinate
```

*Print/Summary Function for RRPP*

---

**Description**

Print/Summary Function for RRPP

**Usage**

```
## S3 method for class 'summary.ordinate'  
print(x, ...)
```

**Arguments**

x	Object from <a href="#">summary.ordinate</a>
...	Other arguments passed onto print.ordinate

**Author(s)**

Michael Collyer

---

```
print.summary.pairwise
```

*Print/Summary Function for RRPP*

---

**Description**

Print/Summary Function for RRPP

**Usage**

```
## S3 method for class 'summary.pairwise'  
print(x, ...)
```

**Arguments**

x	Object from <a href="#">summary.pairwise</a>
...	Other arguments passed onto summary.pairwise

**Author(s)**

Michael Collyer



---

```
print.summary.trajectory.analysis
    Print/Summary Function for RRPP
```

---

### Description

Print/Summary Function for RRPP

### Usage

```
## S3 method for class 'summary.trajectory.analysis'
print(x, ...)
```

### Arguments

x                    Object from [summary.trajectory.analysis](#)  
...                   Other arguments passed onto [summary.trajectory.analysis](#)

### Author(s)

Michael Collyer

---

```
print.trajectory.analysis
    Print/Summary Function for RRPP
```

---

### Description

Print/Summary Function for RRPP

### Usage

```
## S3 method for class 'trajectory.analysis'
print(x, ...)
```

### Arguments

x                    Object from [trajectory.analysis](#)  
...                   Other arguments passed onto

### Author(s)

Michael Collyer

---

Pupfish

*Landmarks on pupfish*

---

### Description

Landmark data from *Cyprinodon pecosensis* body shapes, with indication of Sex and Population from which fish were sampled (Marsh or Sinkhole).

### Details

These data were previously aligned with GPA. Centroid size (CS) is also provided. See the **geomorph** package for details.

### Author(s)

Michael Collyer

### References

Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. *Heredity*. 113: doi:10.1038/hdy.2014.75.

---

PupfishHeads

*Landmarks on pupfish heads*

---

### Description

Landmark data from *Cyprinodon pecosensis* head shapes, with variables for sex, month and year sampled, locality, head size, and coordinates of landmarks for head shape, per specimen. These data are a subset of a larger data set.

### Details

The variable, "coords", are data that were previously aligned with GPA. The variable, "headSize", is the Centroid size of each vector of coordinates. See the **geomorph** package for details.

### Author(s)

Michael Collyer

### References

Gilbert, M.C. 2016. Impacts of habitat fragmentation on the cranial morphology of a threatened desert fish (*Cyprinodon pecosensis*). Masters Thesis, Western Kentucky University.

---

residuals.lm.rppp      *Extract residuals*

---

**Description**

Extract residuals

**Usage**

```
## S3 method for class 'lm.rppp'  
residuals(object, ...)
```

**Arguments**

object            plot object (from [lm.rppp](#))  
...               Arguments passed to other functions

**Author(s)**

Michael Collyer

**Examples**

```
# See examples for lm.rppp
```

---

reveal.model.designs      *Reveal model designs used in lm.rppp fit*

---

**Description**

Function returns every full and reduced model for model terms used in `lm.rppp` fits. This function is useful for revealing the null and full model that would be used in the pairwise function, if a specific null model is not declared as an argument (`fit.null` in the [pairwise](#) function). It also helps to demonstrate how sums of squares and cross-products (SSCP) are calculated in `lm.rppp` permutations (iterations), from the difference between fitted values for null and full designs.

**Usage**

```
reveal.model.designs(fit)
```

**Arguments**

fit                A linear model fit from [lm.rppp](#).

**Author(s)**

Michael Collyer

## Examples

```
data(Pupfish)
fit1 <- lm.rrpp(coords~ Pop*Sex, data = Pupfish,
SS.type = "I", print.progress = FALSE, iter = 0)
fit2 <- lm.rrpp(coords~ Pop*Sex, data = Pupfish,
SS.type = "II", print.progress = FALSE, iter = 0)
fit3 <- lm.rrpp(coords~ Pop*Sex, data = Pupfish,
SS.type = "III", print.progress = FALSE, iter = 0)

reveal.model.designs(fit1)
reveal.model.designs(fit2)
reveal.model.designs(fit3)
```

---

rrpp.data.frame

*Create a data frame for lm.rrpp analysis*

---

## Description

Create a data frame for lm.rrpp analysis, when covariance or distance matrices are used

## Usage

```
rrpp.data.frame(...)
```

## Arguments

...                   Components (objects) to combine in the data frame.

## Details

This function is not much different than [data.frame](#) but is more flexible to allow distance matrices and covariance matrices to be included. Essentially, this function creates a list, much like an object of class `data.frame` is also a list. However, `rrpp.data.frame` is less concerned with coercing the list into a matrix and more concerned with matching the number of observations (n). It is wise to use this function with any `lm.rrpp` analysis so that `lm.rrpp` does not have to search the global environment for needed data.

It is assumed that multiple data sets for the same subjects are in the same order.

See [lm.rrpp](#) for examples.

## Author(s)

Michael Collyer

## Examples

```
# Why use a rpp.data.frame?
y <- matrix(rnorm(30), 10, 3)
x <- rnorm(10)
df <- data.frame(x = x, y = y)
df
rdf <- rpp.data.frame(x = x, y = y)
rdf # looks more like a list

is.list(df)
is.list(rdf)

d <- dist(y) # distance matrix as data

# One can try this but it will result in an error
# df <- data.frame(df, d = d)
rdf <- rpp.data.frame(rdf, d = d) # works

fit <- lm.rpp(d ~ x, data = rdf)
summary(fit)
```

---

summary.anova.lm.rpp *Print/Summary Function for RRPP*

---

## Description

Print/Summary Function for RRPP

## Usage

```
## S3 method for class 'anova.lm.rpp'
summary(object, ...)
```

## Arguments

object	Object from <a href="#">predict.lm.rpp</a>
...	Other arguments passed onto <a href="#">predict.lm.rpp</a>

## Author(s)

Michael Collyer

---

summary.coef.lm.rpp *Print/Summary Function for RRPP*

---

**Description**

Print/Summary Function for RRPP

**Usage**

```
## S3 method for class 'coef.lm.rpp'  
summary(object, ...)
```

**Arguments**

object	Object from <a href="#">coef.lm.rpp</a>
...	Other arguments passed onto coef.lm.rpp

**Author(s)**

Michael Collyer

---

summary.lm.rpp *Print/Summary Function for RRPP*

---

**Description**

Print/Summary Function for RRPP

**Usage**

```
## S3 method for class 'lm.rpp'  
summary(object, formula = TRUE, ...)
```

**Arguments**

object	print/summary object (from <a href="#">lm.rpp</a> )
formula	Logical argument for whether to include formula in summary table
...	other arguments passed to print/summary

**Author(s)**

Michael Collyer

---

```
summary.manova.lm.rppp
```

*Print/Summary Function for RRPP*

---

### Description

Print/Summary Function for RRPP

### Usage

```
## S3 method for class 'manova.lm.rppp'  
summary(object, test = c("Roy", "Pillai", "Hotelling-Lawley", "Wilks"), ...)
```

### Arguments

object	Object from <a href="#">lm.rppp</a> , updated with <a href="#">manova.update</a>
test	Type of multivariate test statistic to use.
...	Other arguments passed onto <code>manova.lm.rppp</code>

### Author(s)

Michael Collyer

---

```
summary.model.comparison
```

*Print/Summary Function for RRPP*

---

### Description

Print/Summary Function for RRPP

### Usage

```
## S3 method for class 'model.comparison'  
summary(object, ...)
```

### Arguments

object	Object from <a href="#">model.comparison</a>
...	Other arguments passed onto <code>model.comparison</code>

### Author(s)

Michael Collyer

---

summary.ordinate      *Print/Summary Function for RRPP*

---

**Description**

Print/Summary Function for RRPP

**Usage**

```
## S3 method for class 'ordinate'  
summary(object, ...)
```

**Arguments**

object      Object from [ordinate](#)  
...      Other arguments passed onto print.ordinate

**Author(s)**

Michael Collyer

---

summary.pairwise      *Print/Summary Function for RRPP*

---

**Description**

See [pairwise](#) for further description.

**Usage**

```
## S3 method for class 'pairwise'  
summary(  
  object,  
  stat.table = TRUE,  
  test.type = c("dist", "VC", "DL", "var"),  
  angle.type = c("rad", "deg"),  
  confidence = 0.95,  
  show.vectors = FALSE,  
  ...  
)
```



## Arguments

object	Object from <a href="#">pairwise</a>
stat.table	Logical argument for whether results should be returned in one table (if TRUE) or separate pairwise tables (if FALSE)
test.type	the type of statistic to test. See below should be used in the test.
angle.type	If test.type = "VC", whether angle results are expressed in radians or degrees.
confidence	Confidence level to use for upper confidence limit; default = 0.95 (alpha = 0.05)
show.vectors	Logical value to indicate whether vectors should be printed.
...	Other arguments passed onto pairwise

## Details

The following summarize the test that can be performed:

#'

- **Distance between vectors, "dist"** Vectors for LS means or slopes originate at the origin and point to some location, having both a magnitude and direction. A distance between two vectors is the inner-product of the vector difference, i.e., the distance between their endpoints. For LS means, this distance is the difference between means. For multivariate slope vectors, this is the difference in location between estimated change for the dependent variables, per one-unit change of the covariate considered. For univariate slopes, this is the absolute difference between slopes.
- **Vector correlation, "VC"** If LS mean or slope vectors are scaled to unit size, the vector correlation is the inner-product of the scaled vectors. The arccosine (acos) of this value is the angle between vectors, which can be expressed in radians or degrees. Vector correlation indicates the similarity of vector orientation, independent of vector length.
- **Difference in vector lengths, "DL"** If the length of a vector is an important attribute – e.g., the amount of multivariate change per one-unit change in a covariate – then the absolute value of the difference in vector lengths is a practical statistic to compare vector lengths. Let  $d_1$  and  $d_2$  be the distances (length) of vectors. Then  $|d_1 - d_2|$  is a statistic that compares their lengths.
- **Variance, "var"** Vectors of residuals from a linear model indicate can express the distances of observed values from fitted values. Mean squared distances of values (variance), by group, can be used to measure the amount of dispersion around estimated values for groups. Absolute differences between variances are used as test statistics to compare mean dispersion of values among groups. Variance degrees of freedom equal  $n$ , the group size, rather than  $n-1$ , as the purpose is to compare mean dispersion in the sample. (Additionally, tests with one subject in a group are possible, or at least not a hindrance to the analysis.)

The argument, `test.type` is used to select one of the tests above. See [pairwise](#) for examples.

### Notes for RRPP 0.6.2 and subsequent versions:

In previous versions of `pairwise`, `summary.pairwise` had three test types: "dist", "VC", and "var". When one chose "dist", for LS mean vectors, the statistic was the inner-product of the vector difference. For slope vectors, "dist" returned the absolute value of the difference between vector lengths, which is "DL" in 0.6.2 and subsequent versions. This update uses the same calculation, irrespective of vector types. Generally, "DL" is the same as a contrast in rates for slope vectors,

but might not have much meaning for LS means. Likewise, "dist" is the distance between vector endpoints, which might make more sense for LS means than slope vectors. Nevertheless, the user has more control over these decisions with version 0.6.2 and subsequent versions.

**Author(s)**

Michael Collyer

---

summary.predict.lm.rppp

*Print/Summary Function for RRPP*

---

**Description**

Print/Summary Function for RRPP

**Usage**

```
## S3 method for class 'predict.lm.rppp'  
summary(object, ...)
```

**Arguments**

object	Object from <a href="#">predict.lm.rppp</a>
...	Other arguments passed onto predict.lm.rppp

**Author(s)**

Michael Collyer

---

summary.trajectory.analysis

*Print/Summary Function for RRPP*

---

**Description**

Print/Summary Function for RRPP

**Usage**

```
## S3 method for class 'trajectory.analysis'
summary(
  object,
  stat.table = TRUE,
  attribute = c("MD", "TC", "SD"),
  angle.type = c("rad", "deg"),
  confidence = 0.95,
  show.trajectories = FALSE,
  ...
)
```

**Arguments**

object	Object from <a href="#">trajectory.analysis</a>
stat.table	Logical argument for whether results should be returned in one table (if TRUE) or separate pairwise tables (if FALSE)
attribute	Whether magnitude differences (MD, absolute difference in trajectory path lengths), trajectory correlations (TC), or trajectory shape differences (SD) are summarized.
angle.type	If attribute = "TC", whether angle results are expressed in radians or degrees.
confidence	Confidence level to use for upper confidence limit; default = 0.95 (alpha = 0.05)
show.trajectories	Logical value to indicate whether trajectories should be printed.
...	Other arguments passed onto trajectory.analysis

**Author(s)**

Michael Collyer

---

terms.lm.rppp

*Extract the terms from an lm.rppp object*

---

**Description**

terms.lm.rppp returns the terms constructed for an lm.rppp object.

**Usage**

```
## S3 method for class 'lm.rppp'
terms(x, ...)
```

**Arguments**

x	Object from <a href="#">lm.rppp</a>
...	further arguments passed to or from other methods

**Author(s)**

Michael Collyer

---

 trajectory.analysis    *Quantify and compare shape change trajectories*


---

**Description**

Function estimates attributes of multivariate trajectories

**Usage**

```
trajectory.analysis(
  fit,
  fit.null = NULL,
  groups,
  traj.pts,
  pca = TRUE,
  print.progress = FALSE
)
```

**Arguments**

<code>fit</code>	A linear model fit using <a href="#">lm.rppp</a> .
<code>fit.null</code>	An alternative linear model fit to use as a null model for RRPP, if the null model of the fit is not desired. Note, if RRPP = FALSE (FRPP rather than RRPP), then the null model has only an intercept. If the null model is uncertain, using <a href="#">reveal.model.designs</a> will help elucidate the inherent null model used.
<code>groups</code>	A factor or vector coercible to factor that defines trajectories.
<code>traj.pts</code>	Either a single value or a vector coercible to factor to define trajectory points. If only a single value, it is assumed that the data are already in the form, $y_{1p1}$ , $y_{2p1}$ , $y_{3p1}$ , ..., $y_{2p2}$ , $y_{3p2}$ , ..., $y_{jp1}$ , $y_{jp2}$ , $y_{jp3}$ , ..., $y_{jpk}$ , for $j$ variables comprising $k$ trajectory points; i.e., $\text{traj.pts} = k$ . If a factor, then a group * traj.pt factorial model is assumed, where traj.pts defines the levels for points within groups.
<code>pca</code>	A logical value to optionally project group:point means onto principal components (perform PCA on a covariance matrix of the means) This option only applies to factorial designs (traj.pts is a factor).
<code>print.progress</code>	A logical value to indicate whether a progress bar should be printed to the screen. This is helpful for long-running analyses.

## Details

The function quantifies multivariate trajectories from a set of observations, and assesses variation in attributes of the trajectories via RRPP. A trajectory is defined by a sequence of points in the data space. These trajectories can be quantified for various attributes (their size, orientation, and shape), and comparisons of these attribute enable the statistical comparison of shape change trajectories (Collyer and Adams 2007; Adams and Collyer 2007; Adams and Collyer 2009; Turner et al. 2010; Collyer and Adams 2013).

This function is a modified version of `pairwise`, retaining the least squares (LS) means as trajectory points. Analysis starts with a `lm.rrpp` fit (but a `procD.lm` fit from `geomorph` can also be used). LS means are calculated using a grouping variable. Data can be trajectories, as a start (sensu Adams and Cerney 2007), or trajectories can be calculated from data using a factorial model (in which case trajectory points are defined by factor levels).

This function produces statistics that can be summarized with the `summary.trajectory.analysis` function. The summaries are consistent with those in the `summary.pairwise` function, pertaining to trajectory attributes including, magnitude difference (MD), the difference in path lengths of trajectories; trajectory correlations (TC), better thought of as angular differences between trajectory principal axes; and if trajectories have three or more points, shape difference (SD), the square root of summed squared point differences, after scaling, centering, and rotating trajectories. The SD is the "Procrustes" distance between trajectories (Adams and Collyer 2009), much the same way as the shape difference between anatomical landmark configurations in geometric morphometrics. If attribute = "TC" is chosen for the summary, then the angle type ("rad" or "deg", can be chosen for either radians and degrees, respectively, to return angles between principal axes.)

Plotting can be performed with `plot.trajectory.analysis` and `add.trajectories`. The former plots all principal component scores for the data, and allows point-by-point control of plot parameters. The later adds trajectories points and lines, with constrained control. By saving the `plot.trajectory.analysis` object, plotting information can be retained and advanced plotting can be performed. See examples below.

## Value

An object of class "trajectory.analysis" returns a list of the following:

LS.means	LS.means from pairwise function.
trajectories	Trajectories from every permutation.
PD	Path distances of trajectories from every permutation.
MD	Magnitude differences between trajectories from every permutation.
TC	Trajectory correlations from every permutation.
SD	Trajectory shape differences from every permutation.

## Author(s)

Dean Adams and Michael Collyer

## References

- Adams, D. C., and M. M. Cerney. 2007. Quantifying biomechanical motion using Procrustes motion analysis. *J. Biomech.* 40:437-444.
- Adams, D. C., and M. L. Collyer. 2007. The analysis of character divergence along environmental gradients and other covariates. *Evolution* 61:510-515.
- Adams, D. C., and M. L. Collyer. 2009. A general framework for the analysis of phenotypic trajectories in evolutionary studies. *Evolution* 63:1143-1154.
- Collyer, M. L., and D. C. Adams. 2007. Analysis of two-state multivariate phenotypic change in ecological studies. *Ecology* 88:683-692.
- Collyer, M. L., and D. C. Adams. 2013. Phenotypic trajectory analysis: comparison of shape change patterns in evolution and ecology. *Hystrix* 24: 75-83.
- Collyer, M.L., D.J. Sekora, and D.C. Adams. 2015. A method for analysis of phenotypic change for phenotypes described by high-dimensional data. *Heredity*. 115:357-365.

## Examples

```
### Analysis of sexual dimorphism vectors (factorial approach)
data(Pupfish)
fit <- lm.rpp(coords ~ Pop * Sex, data = Pupfish, iter = 199)
reveal.model.designs(fit)
TA <- trajectory.analysis(fit, groups = Pupfish$Pop,
traj.pts = Pupfish$Sex, print.progress = FALSE)

# Magnitude difference (absolute difference between path distances)
summary(TA, attribute = "MD")

# Correlations (angles) between trajectories
summary(TA, attribute = "TC", angle.type = "deg")

# No shape differences between vectors
summary(TA, attribute = "SD")

# Retain results
TA.summary <- summary(TA, attribute = "MD")
TA.summary$summary.table

# Plot results
TP <- plot(TA, pch = as.numeric(Pupfish$Pop) + 20, bg = as.numeric(Pupfish$Sex),
cex = 0.7, col = "gray")
add.trajectories(TP, traj.pch = c(21, 22), start.bg = 1, end.bg = 2)
legend("topright", levels(Pupfish$Pop), pch = c(21, 22), pt.bg = 1)

### Analysis when data are already trajectories (motion paths)

# data are planar Cartesian coordinates (x, y) across 5 points (10 variables)
data(motionpaths)
fit <- lm.rpp(trajectories ~ groups, data = motionpaths, iter = 199)
TA <- trajectory.analysis(fit, groups = motionpaths$groups, traj.pts = 5)
```

```
# Magnitude difference (absolute difference between path distances)
summary(TA, attribute = "MD")

# Correlations (angles) between trajectories
summary(TA, attribute = "TC", angle.type = "deg")

# Shape differences between trajectories
summary(TA, attribute = "SD")

TP <- plot(TA, pch = 21, bg = as.numeric(motionpaths$groups),
cex = 0.7, col = "gray")
add.trajectories(TP, traj.pch = 21, traj.bg = 1:4)
```

---

vec.cor.matrix

*Support function for RRPP*

---

### **Description**

Calculate vector correlations for a matrix (by rows). Used for pairwise comparisons.

### **Usage**

```
vec.cor.matrix(M)
```

### **Arguments**

M                    Matrix for vector correlations.

### **Author(s)**

Michael Collyer

# Index

- \* **analysis**
  - lm.rpp, [12](#)
  - manova.update, [19](#)
  - model.comparison, [22](#)
  - ordinate, [26](#)
  - pairwise, [30](#)
  - prep.lda, [41](#)
  - reveal.model.designs, [51](#)
  - trajectory.analysis, [60](#)
- \* **datasets**
  - motionpaths, [26](#)
  - PlethMorph, [34](#)
  - Pupfish, [50](#)
- \* **graphics**
  - add.tree, [5](#)
- \* **utilities**
  - add.trajectories, [4](#)
  - anova.lm.rpp, [7](#)
  - coef.lm.rpp, [9](#)
  - convert2ggplot, [10](#)
  - fitted.lm.rpp, [11](#)
  - model.frame.lm.rpp, [25](#)
  - model.matrix.lm.rpp, [25](#)
  - plot.lm.rpp, [35](#)
  - plot.model.comparison, [36](#)
  - plot.ordinate, [36](#)
  - plot.predict.lm.rpp, [37](#)
  - plot.trajectory.analysis, [38](#)
  - predict.lm.rpp, [40](#)
  - print.anova.lm.rpp, [43](#)
  - print.coef.lm.rpp, [44](#)
  - print.lm.rpp, [44](#)
  - print.model.comparison, [45](#)
  - print.ordinate, [45](#)
  - print.pairwise, [46](#)
  - print.predict.lm.rpp, [46](#)
  - print.summary.lm.rpp, [47](#)
  - print.summary.manova.lm.rpp, [47](#)
  - print.summary.ordinate, [48](#)
  - print.summary.pairwise, [48](#)
  - print.summary.trajectory.analysis, [49](#)
  - print.trajectory.analysis, [49](#)
  - residuals.lm.rpp, [51](#)
  - rrpp.data.frame, [52](#)
  - summary.anova.lm.rpp, [53](#)
  - summary.coef.lm.rpp, [54](#)
  - summary.lm.rpp, [54](#)
  - summary.manova.lm.rpp, [55](#)
  - summary.model.comparison, [55](#)
  - summary.ordinate, [56](#)
  - summary.pairwise, [56](#)
  - summary.predict.lm.rpp, [58](#)
  - summary.trajectory.analysis, [58](#)
  - terms.lm.rpp, [59](#)
  - vec.cor.matrix, [63](#)
- \* **visualization**
  - add.trajectories, [4](#)
  - plot.lm.rpp, [35](#)
  - plot.model.comparison, [36](#)
  - plot.ordinate, [36](#)
  - plot.predict.lm.rpp, [37](#)
  - plot.trajectory.analysis, [38](#)
  - add.trajectories, [4](#), [38](#), [61](#)
  - add.tree, [5](#)
  - anova, [14](#), [23](#)
  - anova.lm.rpp, [3](#), [7](#), [14](#), [20](#)
  - arrows, [38](#)
  - classify, [8](#), [42](#)
  - coef.lm.rpp, [3](#), [9](#), [13](#), [14](#), [44](#), [54](#)
  - convert2ggplot, [10](#)
  - data.frame, [40](#), [52](#)
  - fitted.lm.rpp, [11](#)
  - ggplot, [10](#)



lda, [41](#), [42](#)  
lines, [6](#)  
lm, [13–16](#), [20](#), [35](#), [40](#)  
lm.rppp, [3](#), [7](#), [9](#), [12](#), [12](#), [19](#), [21](#), [25](#), [30](#), [31](#), [33](#),  
[35](#), [38](#), [40–44](#), [51](#), [52](#), [54](#), [55](#), [59–61](#)  
  
manova.update, [14](#), [19](#), [55](#)  
model.comparison, [22](#), [36](#), [45](#), [55](#)  
model.frame.lm.rppp, [25](#)  
model.matrix.lm.rppp, [25](#)  
motionpaths, [26](#)  
  
ordinate, [6](#), [26](#), [37](#), [45](#), [56](#)  
  
pairwise, [3](#), [30](#), [46](#), [51](#), [56](#), [57](#), [61](#)  
par, [4](#), [5](#), [35](#), [36](#), [38](#), [39](#)  
PlethMorph, [34](#)  
plot, [24](#)  
plot.default, [5](#), [35](#), [36](#), [38](#), [39](#)  
plot.lm.rppp, [10](#), [35](#)  
plot.model.comparison, [36](#)  
plot.ordinate, [6](#), [10](#), [29](#), [36](#)  
plot.predict.lm.rppp, [10](#), [37](#), [40](#)  
plot.trajectory.analysis, [4](#), [5](#), [38](#), [61](#)  
points, [6](#), [38](#)  
prcomp, [27–29](#), [39](#), [41](#)  
predict.lda, [42](#)  
predict.lm, [40](#)  
predict.lm.rppp, [3](#), [37](#), [38](#), [40](#), [40](#), [46](#), [53](#), [58](#)  
predict.qda, [42](#)  
prep.lda, [8](#), [41](#)  
print.anova.lm.rppp, [43](#)  
print.coef.lm.rppp, [44](#)  
print.lm.rppp, [44](#)  
print.model.comparison, [45](#)  
print.ordinate, [45](#)  
print.pairwise, [46](#)  
print.predict.lm.rppp, [46](#)  
print.summary.lm.rppp, [47](#)  
print.summary.manova.lm.rppp, [47](#)  
print.summary.ordinate, [48](#)  
print.summary.pairwise, [48](#)  
print.summary.trajectory.analysis, [49](#)  
print.trajectory.analysis, [49](#)  
Pupfish, [50](#)  
PupfishHeads, [50](#)  
  
qda, [41](#), [42](#)  
  
residuals.lm.rppp, [51](#)  
  
reveal.model.designs, [30](#), [32](#), [51](#), [60](#)  
RRPP (RRPP-package), [3](#)  
RRPP-package, [3](#)  
rppp.data.frame, [13](#), [40](#), [52](#)  
  
summary.anova.lm.rppp, [53](#)  
summary.coef.lm.rppp, [54](#)  
summary.lm.rppp, [47](#), [54](#)  
summary.manova, [20](#)  
summary.manova.lm.rppp, [14](#), [19](#), [20](#), [47](#), [55](#)  
summary.model.comparison, [55](#)  
summary.ordinate, [28](#), [48](#), [56](#)  
summary.pairwise, [31](#), [32](#), [48](#), [56](#), [57](#), [61](#)  
summary.predict.lm.rppp, [58](#)  
summary.trajectory.analysis, [49](#), [58](#), [61](#)  
  
terms.lm.rppp, [59](#)  
text, [10](#), [38](#)  
trajectory.analysis, [38](#), [49](#), [59](#), [60](#)  
  
vec.cor.matrix, [63](#)