Package ‘rr2’

October 14, 2022

Type Package
Title R2s for Regression Models
Version 1.0.2
Description Three methods to calculate R2 for models with correlated errors, including Phylogenetic GLS, Phylogenetic Logistic Regression, Linear Mixed Models (LMMs), and Generalized Linear Mixed Models (GLMMs). See details in Ives 2018 <doi:10.1093/sysbio/syy060>.
License GPL-3
Depends R (>= 3.0), stats
Encoding UTF-8
LazyData true
Imports lme4, phylolm, ape, utils, Matrix, nlme
RoxygenNote 6.1.1
Suggests testthat
URL https://github.com/arives/rr2
BugReports https://github.com/arives/rr2/issues
NeedsCompilation no
Author Anthony Ives [aut, cre], Daijiang Li [ctb]
Maintainer Anthony Ives <arives@wisc.edu>
Repository CRAN
Date/Publication 2019-05-09 20:30:03 UTC

R topics documented:

binaryPGLMM ................................................................. 2
inv.logit ................................................................. 3
partialR2 ................................................................. 3
partialR2adj ............................................................... 4
binaryPGLMM

Phylogenetic GLM for binary data

Description

Fitting phylogenetic generalized linear models for binary data (0 and 1).

Usage

binaryPGLMM(formula, data = list(), phy, s2.init = 0.1, 
B.init = NULL, tol.pql = 10^-6, maxit.pql = 200, 
maxit.reml = 100)

Arguments

- formula: Regression formula.
- data: Data frame to fit the model with.
- phy: Phylogenetic tree of type phylo with branch lengths.
- s2.init: Initial variance values for random terms, default is 0.1.
- B.init: Initial coefficient values for fixed terms, if not provided, will use those from lm.
- tol.pql: Tolerance value, default is 10^-6.
- maxit.pql: The number of iterations, default is 200.
- maxit.reml: The number of iterations for optim, default is 100.

Value

A large list with class as binaryPGLMM.
**inv.logit**

*Inv logit function*

**Description**

Convert numeric values between 0 and 1.

**Usage**

inv.logit(x)

**Arguments**

- x: A numeric vector.

---

**partialR2**

*Partial R2*

**Description**

Get partial R2 by comparing a model and its reduced model.

**Usage**

partialR2(mod, mod.r)

**Arguments**

- mod: A linear regression model.
- mod.r: A reduced model based on mod.

**Value**

R2 value between 0 and 1.
**R2adj**

*Adjusted partial R2*

**Description**

Get adjusted partial R2 by comparing a model and its reduced model.

**Usage**

```r
partialR2adj(mod, df.f = summary(mod)$df[1], mod.r, df.r = summary(mod.r)$df[1])
```

**Arguments**

- `mod`: A linear regression model.
- `df.f`: Degree of freedom of the `mod`.
- `mod.r`: A reduced model based on `mod`.
- `df.r`: Degree of freedom of the reduced `mod.r`.

**Value**

A list of both R2 and adjusted R2, the latter is not necessary to be between 0 and 1.

---

**R2**

*Calculate R2.lik, R2.resid, and R2.pred*

**Description**

This is a wrapper for calculating three R2s – R2.lik, R2.resid, and R2.pred – for LMMs and GLMMs, and phylogenetic LMMs (PLMMs) and GLMMs (PGLMMs). Note that the individual functions `R2.lik()`, `R2.resid()`, and `R2.pred()` can be called separately. This is preferrable if you are only interested in one R2; for example, for `phylolm()` called from 'R2' you need to specify `phy` (phylo object for the phylogeny), while `R2.lik()` does not require this.

**Usage**

```r
R2(mod = NULL, mod.r = NULL, phy = NULL, sigma2_d = c("s2w", "NS", "rNS"), lik = TRUE, resid = TRUE, pred = TRUE)
```
Arguments

- **mod**: A regression model with one of the following classes: 'lm', 'glm', lmerMod', glmerMod', 'phylolm', 'gls', binaryPGLMM', or 'communityPGLMM'.
- **mod.r**: A reduced model; if not provided, the total R² will be given by setting 'mod.r' to the model corresponding to 'mod' with the intercept as the only predictor.
- **phy**: The phylogeny for phylogenetic models (as a 'phylo' object), which is not required to be specified for R2.lik() of non-phylogenetic models.
- **sigma2_d**: Distribution-specific variance $\sigma^2_d$ (see Details) used in R2.resid(). For binomial GLMs, GLMMs and PGLMMs with logit link functions, options are c('s2w', 'NS', 'rNS'). For binomial GLMs, GLMMs and PGLMMs with probit link functions, options are c('s2w', 'NS'). Other families use 's2w'.
- **lik**: Whether to calculate R2.lik; default is TRUE.
- **resid**: Whether to calculate R2.resid; default is TRUE.
- **pred**: Whether to calculate R2.pred; default is TRUE.

Details

Details about the methods are provided under the separate functions for R2.lik(), R2.resid(), and R2.pred(). There are also many worked examples.

Value

A vector, with all three R²s by default.

Author(s)

Daijiang Li and Anthony R. Ives

References


Ives A.R. 2018. R²s for Correlated Data: Phylogenetic Models, LMMs, and GLMMs. Systematic Biology. DOI:10.1093/sysbio/syy060

See Also

MuMIn, lme4, ape, phylolm, pez

Examples

```r
library(ape)
library(phylolm)
library(lme4)
library(nlme)

# LMM with two fixed and two random effects
```
p1 <- 10
nsample <- 10
n <- p1 * nsample

d <- data.frame(x1 = 0, x2 = 0, y = 0, u1 = rep(1:p1, each = nsample),
                u2 = rep(1:p1, times = nsample))
d$u1 <- as.factor(d$u1)
d$u2 <- as.factor(d$u2)

b1 <- 1
b2 <- -1
sd1 <- 1.5

d$x1 <- rnorm(n = n)
d$x2 <- rnorm(n = n)
d$y <- b1 * d$x1 + b2 * d$x2 + rep(rnorm(n = p1, sd = sd1), each = nsample) +
     rep(rnorm(n = p1, sd = sd1), times = nsample) + rnorm(n = n)

z.f <- lmer(y ~ x1 + x2 + (1 | u1) + (1 | u2), data = d, REML = FALSE)
z.x <- lmer(y ~ x1 + (1 | u1) + (1 | u2), data = d, REML = FALSE)
z.v <- lmer(y ~ 1 + (1 | u2), data = d, REML = FALSE)
z.0 <- lm(y ~ 1, data = d)

R2(z.f, z.x)
R2(z.f, z.v)
R2(z.f)

# These give different results for R2.resid.

p1 <- 10
nsample <- 10
n <- p1 * nsample

d <- data.frame(x = 0, y = 0, u = rep(1:p1, each = nsample))
d$u <- as.factor(d$u)

b1 <- 1
sd1 <- 1.5

d$x <- rnorm(n = n)
prob <- inv.logit(b1 * d$x + rep(rnorm(n = p1, sd = sd1), each = nsample))
d$y <- rbinom(n = n, size = 1, prob = prob)

z.f <- glmer(y ~ x + (1 | u), data = d, family = 'binomial')
z.x <- glmer(y ~ 1 + (1 | u), data = d, family = 'binomial')
z.v <- glm(y ~ x, data = d, family = 'binomial')

R2(z.f, z.x)
R2(z.f, z.v)
R2(z.f)

# These give different results for R2.resid.
R2(z.f, sigma2_d = 's2w')
R2(z.f, sigma2_d = 'NS')
R2(z.f, sigma2_d = 'rNS')

# PGLS with a single fixed effect

n <- 100
D <- data.frame(x = array(0, dim = n), y = 0)
b1 <- 1.5
signal <- 0.7

phy <- compute.brlen(rtree(n = n), method = 'Grafen', power = 1)
phy.x <- compute.brlen(phy, method = 'Grafen', power = .0001)

# Generate random data
x <- rTraitCont(phy.x, model = 'BM', sigma = 1)
e <- signal^0.5 * rTraitCont(phy, model = 'BM', sigma = 1) + (1-signal)^0.5 * rnorm(n = n)
D$x <- x[match(names(e), names(x))]
D$y <- b1 * x + e
rownames(D) <- phy$tip.label

z.x <- phylolm(y ~ 1, phy = phy, data = D, model = 'lambda')
z.f <- phylolm(y ~ x, phy = phy, data = D, model = 'lambda')
z.v <- lm(y ~ x, data = D)

R2(z.f, z.x, phy = phy)
R2(z.f, z.v, phy = phy)
R2(z.f, phy = phy)

# This also works for models fit with gls() in {nlme}
z.x <- gls(y ~ 1, data = D, correlation = corPagel(1, phy), method = 'ML')
z.f <- gls(y ~ x, data = D, correlation = corPagel(1, phy), method = 'ML')
z.v <- lm(y ~ x, data = D)

R2(z.f, z.x)
R2(z.f, z.v)
R2(z.f)

# But note that you need to define weights for gls() with non-ultrametric trees;
# if not, you will get a error from R2.resid, "Matrix is not block-diagonal"

phy.nu <- rtree(n = n)

# Generate random data
e <- signal^0.5 * rTraitCont(phy.nu, model = 'BM', sigma = 1) + (1-signal)^0.5 * rnorm(n = n)
D$x <- x[match(names(e), names(x))]
D$y <- b1 * x + e
rownames(D) <- phy.nu$tip.label

weights <- diag(vcv.phylo(phy.nu))
z.x <- gls(y ~ 1, data = D,
          correlation = corPagel(1, phy.nu),
          weights = weights)

R2(z.f, z.x)
weights=varFixed(~weights), method = "ML")
z.f <- gls(y ~ x, data = d, 
correlation = corPagel(1, phy.nu),
weights=varFixed(~weights), method = "ML")
z.v <- lm(y ~ x, data = d)

R2(z.f, z.x)
R2(z.f, z.v)
R2(z.f)

########################################
# PGLMM with one fixed effect

n <- 100
b1 <- 1.5
signal <- 2

phy <- compute.brlen(rtree(n = n), method = 'Grafen', power = 1)
phy.x <- compute.brlen(phy, method = 'Grafen', power = .0001)

# Generate random data
x <- rnorm(n)
d <- data.frame(x = x, y = 0)
e <- signal * rTraitCont(phy, model = 'BM', sigma = 1)
e <- e[match(phy$tip.label, names(e))]
d$s$ <- rbinom(n = n, size = 1, prob = inv.logit(b1 * d$s + e))
rownames(d) <- phy$tip.label

# Use the function binaryPGLMM() from the rr2 package rather than ape.
z.f <- rr2::binaryPGLMM(y ~ x, data = d, phy = phy)
z.x <- rr2::binaryPGLMM(y ~ 1, data = d, phy = phy)
z.v <- glm(y ~ x, data = d, family = 'binomial')

# R2.lik is not produced, because binaryPGLMM() does not generate a likelihood.
R2(z.f, z.x, phy = phy)
R2(z.f, z.v, phy = phy)
R2(z.f, phy = phy)

R2.lik  Calculate R2.lik

Description
Calculate partial and total R2s for LMM, GLMM, PGLS, and PGLMM using R2.lik, an R2 based on the likelihood of observing the data.

Usage
R2.lik(mod = NULL, mod.r = NULL)
Arguments

mod

A regression model with one of the following classes: 'lm', 'glm', 'lmerMod', 'glmerMod', 'phylolm', 'phyloglm', 'gls', or 'communityPGLMM'.

mod.r

A reduced model; if not provided, the total R2 will be given by setting 'mod.r' to the model corresponding to 'mod' with the intercept as the only predictor.

Details

\( R2.lik() \) works with classes 'lm', 'glm', 'lmerMod', 'glmerMod', 'phylolm', 'phyloglm', and 'communityPGLMM' (family = 'gaussian' only). It is implemented as

\[
\text{partial} R^2 = 1 - \exp\left(-\frac{2}{n} \times (\log \text{Lik}(\text{mod.f}) - \log \text{Lik}(\text{mod.r}))\right)
\]

where 'mod.f' and 'mod.r' are the full and reduced models, respectively. The total R2 is given when 'mod.r' is the model corresponding to mod.f that contains only the intercept. For GLMMs and PGLMMs, \( R2.lik() \) is standardized to have a maximum of one following Nagelkerke (1991).

Note that \( \text{phyloglm()} \) can have difficulties in finding solutions when there is no phylogenetic signal. Therefore, when the estimate of alpha is >50, indicating no phylogenetic signal, the model is refit with the corresponding GLM.

\( R2.lik() \) is also computed for LMMs and GLMMs in the \text{MuMIn} package.

Value

\( R2.lik \) value.

Author(s)

Anthony R. Ives

References


Ives A.R. 2018. R2s for Correlated Data: Phylogenetic Models, LMMs, and GLMMs. Systematic Biology. DOI:10.1093/sysbio/syy060


See Also

\text{MuMIn}, \text{lme4}, \text{ape}, \text{phylolm}, \text{pez}
Examples

```r
library(ape)
library(phylolm)
library(lme4)
library(nlme)

#########################
# LMM with two fixed and two random effects
p1 <- 10
nsample <- 10
n <- p1 * nsample

d <- data.frame(x1 = 0, x2 = 0, y = 0, u1 = rep(1:p1, each = nsample),
                 u2 = rep(1:p1, times = nsample))
d$u1 <- as.factor(d$u1)
d$u2 <- as.factor(d$u2)

b1 <- 1
b2 <- -1
sd1 <- 1.5
d$x1 <- rnorm(n = n)
d$x2 <- rnorm(n = n)
d$y <- b1 * d$x1 + b2 * d$x2 + rep(rnorm(n = p1, sd = sd1), each = nsample) +
     rep(rnorm(n = p1, sd = sd1), times = nsample) + rnorm(n = n)

z.f <- lmer(y ~ x1 + x2 + (1 | u1) + (1 | u2), data = d, REML = FALSE)
z.x <- lmer(y ~ x1 + (1 | u1) + (1 | u2), data = d, REML = FALSE)
z.v <- lmer(y ~ 1 + (1 | u2), data = d, REML = FALSE)
z.0 <- lm(y ~ 1, data = d)

R2.lik(z.f, z.x)
R2.lik(z.f, z.v)
R2.lik(z.f)

# These give the same results.
R2.lik(z.f, z.0)
R2.lik(z.f)

#########################
# GLMM with one fixed and one random effect
p1 <- 10
nsample <- 10
n <- p1 * nsample

d <- data.frame(x = 0, y = 0, u = rep(1:p1, each = nsample))
d$u <- as.factor(d$u)

b1 <- 1
sd1 <- 1.5
```
R2.lik

```r
# Generate random data
n <- 100
d <- data.frame(x = array(0, dim = n), y = 0)
b1 <- 1.5
signal <- 0.7

phy <- compute.brlen(rtree(n = n), method = "Grafen", power = 1)
phy.x <- compute.brlen(phy, method = "Grafen", power = .0001)

# Generate random data
x <- rTraitCont(phy.x, model = "BM", sigma = 1)
e <- signal^0.5 * rTraitCont(phy, model = "BM", sigma = 1) + (1-signal)^0.5 * rnorm(n = n)
d$x <- x[match(names(e), names(x))]
d$y <- b1 * x + e
rownames(d) <- phy$tip.label

z.x <- phylolm(y ~ 1, phy = phy, data = d, model = "lambda")
lam.x <- round(z.x$optpar, digits = 4)
z.f <- phylolm(y ~ x, phy = phy, data = d, model = "lambda")
z.v <- lm(y ~ x, data = d)
R2.lik(z.f, z.x)
R2.lik(z.f, z.v)
R2.lik(z.f)

# This also works for models fit with gls() in {nlme}
z.x <- gls(y ~ 1, data = d, correlation = corPagel(1, phy), method = "ML")
z.f <- gls(y ~ x, data = d, correlation = corPagel(1, phy), method = "ML")
z.v <- lm(y ~ x, data = d)
R2.lik(z.f, z.x)
R2.lik(z.f, z.v)
R2.lik(z.f)
```

# PGLMM with one fixed effect

```r
n <- 100
b1 <- 1.5
```
signal <- 2
phy <- compute.brlen(rtree(n = n), method = 'Grafen', power = 1)
phy.x <- compute.brlen(phy, method = 'Grafen', power = .0001)

# Generate random data
x <- rnorm(n)
d <- data.frame(x = x, y = 0)
e <- signal * rTraitCont(phy, model = 'BM', sigma = 1)
e <- e[match(phy$tip.label, names(e))]

d$y <- rbinom(n = n, size = 1, prob = inv.logit(b1 * d$x + e))
rownames(d) <- phy$tip.label

z.f <- phyloglm(y ~ x, data = d, start.alpha = 1, phy = phy)
z.x <- phyloglm(y ~ 1, data = d, phy = phy, start.alpha = min(20, z.f$alpha))
z.v <- glm(y ~ x, data = d, family = 'binomial')

R2.lik(z.f, z.x)
R2.lik(z.f, z.v)
R2.lik(z.f)

---

**R2.pred**

**Calculate R2.pred**

**Description**

Calculate partial and total R2s for LMM, GLMM, PGLS, and PGLMM using R2.pred, an R2 based on the variance of the difference between the observed and predicted values of a fitted model.

**Usage**

R2.pred(mod = NULL, mod.r = NULL, phy = NULL)

**Arguments**

- **mod**: A regression model with one of the following classes: 'lm', 'glm', 'lmerMod', 'glmerMod', 'phylolm', 'gls', 'binaryPGLMM', or 'communityPGLMM'.
- **mod.r**: A reduced model; if not provided, the total R2 will be given by setting 'mod.r' to the model corresponding to 'mod' with the intercept as the only predictor.
- **phy**: The phylogeny for phylogenetic models (as a 'phylo' object), which must be specified for models of class 'phylolm'.
Details

R2.pred works with classes 'lm', 'glm', 'lmerMod', 'glmerMod', 'phylolm', 'phyloglm', 'gls', binaryPGLMM', and 'communityPGLMM' (family = gaussian and binomial).

LMM (lmerMod), GLMM (glmerMod), PGLMM (binaryPGLMM and communityPGLMM):

\[
partial R^2 = 1 - \frac{\text{var}(y - y.fitted.f)}{\text{var}(y - y.fitted.r)}
\]

where \( y \) are the observed data, and \( y.fitted.f \) and \( y.fitted.r \) are the fitted (predicted) values from the full and reduced models. For GLMMs and PGLMMs, the values of \( y.fitted \) are in the space of the raw data (as opposed to the 'Normal' or 'latent' space). When the reduced model 'mod.r' is not specified, the total \( R^2 \) is computing using the reduced model with only the intercept.

Note that the version of binaryPGLMM() in the package ape is replaced by a version contained within rr2 that outputs all of the required information for the calculation of R2.resid.

PGLS (phylolm and gls):

For PGLS, the total \( R^2\.pred \) is computed by removing each datum one at a time, predicting its value from the fitted model, repeating this for all data points, and then calculating the variance of the difference between observed and fitted values. The predictions are calculated as

\[
\text{res.predicted}[j] = V[j, -j] \text{solve}(V[-j, -j]) \text{res}[-j]
\]

where \( \text{res}[-j] \) is a vector of residuals with datum \( j \) removed, \( V[-j, -j] \) is the phylogenetic covariance matrix with row and column \( j \) removed, and \( V[j, -j] \) is row \( j \) of covariance matrix \( V \) with element \( j \) removed. The partial \( R^2\.pred \) is calculated from the total \( R^2\.pred \) from full and reduced models as

\[
partial R^2 = 1 - \frac{(1 - R^2\.pred.f)}{(1 - R^2\.pred.r)}
\]

Note that phylolm() can have difficulties in finding solutions when there is no phylogenetic signal; when the estimate indicates no phylogenetic signal, you should refit the model with the corresponding LM.

LM (lm) and GLM (glm):

For compatibility and generating reduced models, rr2 will compute \( R^2\.pred \) for LM and GLM that correspond to LMM/PGLS and GLMM/PGLMM.

Value

R2.pred value.

Author(s)

Anthony R. Ives

References


Ives A.R. 2018. R2s for Correlated Data: Phylogenetic Models, LMMs, and GLMMs. Systematic Biology. DOI:10.1093/sysbio/syy060
See Also

MuMIn, lme4, ape, phylolm, pez

Examples

library(ape)
library(phylolm)
library(lme4)
library(nlme)

#################
# LMM with two fixed and two random effects
p1 <- 10
nsample <- 10
n <- p1 * nsample

d <- data.frame(x1 = 0, x2 = 0, y = 0, u1 = rep(1:p1, each = nsample),
                u2 = rep(1:p1, times = nsample))
d$u1 <- as.factor(d$u1)
d$u2 <- as.factor(d$u2)

b1 <- 1
b2 <- -1
sd1 <- 1.5

d$x1 <- rnorm(n = n)
d$x2 <- rnorm(n = n)
d$y <- b1 * d$x1 + b2 * d$x2 + rep(rnorm(n = p1, sd = sd1), each = nsample) +
       rep(rnorm(n = p1, sd = sd1), times = nsample) + rnorm(n = n)

z.f <- lmer(y ~ x1 + x2 + (1 | u1) + (1 | u2), data = d, REML = FALSE)
z.x <- lmer(y ~ x1 + (1 | u1) + (1 | u2), data = d, REML = FALSE)
z.v <- lmer(y ~ 1 + (1 | u2), data = d, REML = FALSE)
z.0 <- lm(y ~ 1, data = d)

R2.pred(z.f, z.x)
R2.pred(z.f, z.v)
R2.pred(z.f)

#################
# GLMM with one fixed and one random effect
p1 <- 10
nsample <- 10
n <- p1 * nsample

d <- data.frame(x = 0, y = 0, u = rep(1:p1, each = nsample))
d$u <- as.factor(d$u)

b1 <- 1
sd1 <- 1.5
d$x <- rnorm(n = n)
prob <- inv.logit(b1 * d$x + rep(rnorm(n = p1, sd = sd1), each = nsample))
d$y <- rbinom(n = n, size = 1, prob = prob)

z.f <- glmer(y ~ x + (1 | u), data = d, family = 'binomial')
z.x <- glmer(y ~ 1 + (1 | u), data = d, family = 'binomial')
z.v <- glm(y ~ x, data = d, family = 'binomial')

R2.pred(z.f, z.x)
R2.pred(z.f, z.v)
R2.pred(z.f)

############################
# PGLS with a single fixed effect

n <- 100
d <- data.frame(x = array(0, dim = n), y = 0)

b1 <- 1.5
signal <- 0.7

phy <- compute.brlen(rtree(n = n), method = 'Grafen', power = 1)
phy.x <- compute.brlen(phy, method = 'Grafen', power = .0001)

# Generate random data
x <- rTraitCont(phy.x, model = 'BM', sigma = 1)
e <- signal^0.5 * rTraitCont(phy, model = 'BM', sigma = 1) + (1-signal)^0.5 * rnorm(n = n)
d$x <- x[match(names(e), names(x))]
d$y <- b1 * x + e
rownames(d) <- phy$tip.label

z.x <- gls(y ~ 1, data = d, correlation = corPagel(1, phy), method = "ML")
z.f <- gls(y ~ x, data = d, correlation = corPagel(1, phy), method = "ML")
z.v <- lm(y ~ x, data = d)

R2.pred(z.f, z.x)
R2.pred(z.f, z.v)
R2.pred(z.f)

############################
# PGLMM with one fixed effect

n <- 100
b1 <- 1.5
signal <- 2

phy <- compute.brlen(rtree(n = n), method = 'Grafen', power = 1)
phy.x <- compute.brlen(phy, method = 'Grafen', power = .0001)

# Generate random data
x <- rnorm(n)
d <- data.frame(x = x, y = 0)
```r
# Load necessary libraries
library(ape)
library(rr2)

# Create data
d <- data.frame(x = runif(10), y = runif(10))
rownames(d) <- 1:10

# Define model
m <- formula(y ~ x)

# Fit model
fit <- lme4::glmer(m, data = d)

# Calculate R2
r2 <- R2.resid(fit)
```

---

**R2.resid**

Calculate partial and total R2s for LMM, GLMM, PGLS, and PGLMM using R2.resid, an extension of ordinary least-squares (OLS) R2s. For LMMs and GLMMs, R2.resid is related to the method proposed by Nakagawa and Schielzeth (2013).

**Usage**

```r
R2.resid(mod = NULL, mod.r = NULL, phy = NULL, sigma2_d = c("s2w", "NS", "rNS"))
```

**Arguments**

- **mod**: A regression model with one of the following classes: 'lm', 'glm', 'lmerMod', 'glmerMod', 'phylolm', 'gls', or 'binaryPGLMM'. For 'glmerMod', only family = c('binomial', 'poisson') are supported.
- **mod.r**: A reduced model; if not provided, the total R2 will be given by setting 'mod.r' to the model corresponding to 'mod' with the intercept as the only predictor.
- **phy**: The phylogeny for phylogenetic models (as a 'phylo' object), which must be specified for models of class 'phylolm'.
- **sigma2_d**: Distribution-specific variance \( \sigma_d^2 \) (see Details). For binomial GLMs, GLMMs and PGLMMs with logit link functions, options are c("s2w", "NS", "rNS"). For binomial GLMs, GLMMs and PGLMMs with probit link functions, options are c("s2w", "NS"). Other families use 's2w'.

**Description**

Calculate partial and total R2s for LMM, GLMM, PGLS, and PGLMM using R2.resid, an extension of ordinary least-squares (OLS) R2s. For LMMs and GLMMs, R2.resid is related to the method proposed by Nakagawa and Schielzeth (2013).
Details


LMM (lmerMod):

\[
\text{partial } R^2 = 1 - \frac{\sigma_{e,f}^2}{\sigma_{e,r}^2}
\]

\[
\text{total } R^2 = 1 - \frac{\sigma_{e,f}^2}{\text{var}(y)}
\]

where \(\sigma_{e,f}^2\) and \(\sigma_{e,r}^2\) are the estimated residual variances from the full and reduced LMM, and \(\text{var}(y)\) is the total variance of the response (dependent) variable.

GLMM (glmerMod):

\[
\text{total } R^2 = 1 - \frac{\sigma_d^2}{\sigma_x^2 + \sigma_b^2 + \sigma_d^2}
\]

where \(\sigma_x^2\) and \(\sigma_b^2\) are the estimated variances associated with the fixed and random effects. \(\sigma_d^2\) is a term that scales the implied ‘residual variance’ of the GLMM (see Ives 2018, Appendix 1). The default used for \(\sigma_d^2\) is \(\sigma_w^2\) which is computed from the iterative weights of the GLMM. Specifically,

\[
\sigma_w^2 = \text{var}(g'(\mu) * (y - \mu))
\]

where \(g'(\cdot)\) is the derivative of the link function, and \((y - \mu)\) is the difference between the data \(y\) and their predicted values \(\mu\). This is the default option specified by sigma2_d = ‘s2w’. For binomial models with a logit link function, sigma2_d = ‘NS’ gives the scaling \(\sigma_d^2 = \pi^2 / 3\), and sigma2_d = ‘rNS’ gives \(\sigma_d^2 = 0.8768809 * \pi^2 / 3\). For binomial models with a probit link function, sigma2_d = ‘NS’ gives the scaling \(\sigma_d^2 = 1\). In general option sigma2_d = ‘s2w’ will give values lower than sigma2_d = ‘NS’ and ‘rNS’, but the values will be closer to R2.lik() and R2.pred(). For other forms of sigma2_d from Nakagawa and Schielzeth (2013) and Nakagawa et al. (2017), see the MuMIn package.

Partial R2s are given by the standard formula

\[
\text{partial } R^2 = 1 - \frac{(1 - R^2_f)}{(1 - R^2_r)}
\]

where \(R^2_f\) and \(R^2_r\) are the total R2s for full and reduced models, respectively.

PGLS (phylolm):

\[
\text{partial } R^2 = 1 - c.f * \frac{\sigma_f^2}{(c.r * \sigma_r^2)}
\]

where \(\sigma_f^2\) and \(\sigma_r^2\) are the variances estimated for the PGLS full and reduced models, and \(c.f\) and \(c.r\) are the scaling values for full and reduce models that equal the total sum of phylogenetic branch length estimates. Note that the phylogeny needs to be specified in R2.resid.

Note that phylolm() can have difficulties in finding solutions when there is no phylogenetic signal; when the estimate indicates no phylogenetic signal, you should refit the model with the corresponding LM.
PGLMM (binaryPGLMM):
The binary PGLMM is computed in the same way as the binomial GLMM, with options sigma_d = c('s2w', 'NS', 'rNS'). The estimated variance of the random effect associated with the phylogeny, \( \sigma^2_b \), is multiplied by the diagonal elements of the phylogenetic covariance matrix. For binary models, this covariance matrix should be standardized so that all diagonal elements are the same (a contemporaneous or ultrametric phylogenetic tree) (Ives and Garland 2014). In case this is not done, however, the code takes the geometric average of the diagonal elements.

Note that the version of binaryPGLMM() in the package ape is replaced by a version contained within rr2 that outputs all of the required information for the calculation of R2.resid()

LM (lm) and GLM (glm):
For compatibility and generating reduced models, rr2 will compute R2.resid() for LM and GLM that correspond to LMM/PGLS and GLMM/PGLMM.

Value
R2.resid value.

Author(s)
Anthony R. Ives

References
Ives A.R. 2018. R2s for Correlated Data: Phylogenetic Models, LMMs, and GLMMs. Systematic Biology. DOI:10.1093/sysbio/syy060

See Also
MuMIn, lme4, ape, phylolm, pez

Examples
library(ape)
library(phylolm)
library(lme4)
library(nlme)
### LMM with two fixed and two random effects

```r
p1 <- 10
nsample <- 10
n <- p1 * nsample
d <- data.frame(x1 = 0, x2 = 0, y = 0, u1 = rep(1:p1, each = nsample), u2 = rep(1:p1, times = nsample))
d$u1 <- as.factor(d$u1)
d$u2 <- as.factor(d$u2)
b1 <- 1
b2 <- -1
sd1 <- 1.5
d$x1 <- rnorm(n = n)
d$x2 <- rnorm(n = n)
d$y <- b1 * d$x1 + b2 * d$x2 + rep(rnorm(n = p1, sd = sd1), each = nsample) + rep(rnorm(n = p1, sd = sd1), times = nsample) + rnorm(n = n)
z.f <- lmer(y ~ x1 + x2 + (1 | u1) + (1 | u2), data = d, REML = FALSE)
z.x <- lmer(y ~ x1 + (1 | u1) + (1 | u2), data = d, REML = FALSE)
z.v <- lmer(y ~ 1 + (1 | u2), data = d, REML = FALSE)
z.0 <- lm(y ~ 1, data = d)
R2.resid(z.f, z.x)
R2.resid(z.f, z.v)
R2.resid(z.f)
```

### GLMM with one fixed and one random effect

```r
p1 <- 10
nsample <- 10
n <- p1 * nsample
d <- data.frame(x = 0, y = 0, u = rep(1:p1, each = nsample))
d$u <- as.factor(d$u)
b1 <- 1
sd1 <- 1.5
d$x <- rnorm(n = n)
prob <- inv.logit(b1 * d$x + rep(rnorm(n = p1, sd = sd1), each = nsample))
d$y <- rbinom(n = n, size = 1, prob = prob)
z.f <- glmer(y ~ x + (1 | u), data = d, family = binomial)
z.x <- glmer(y ~ 1 + (1 | u), data = d, family = binomial)
z.v <- glm(y ~ x, data = d, family = binomial)
R2.resid(z.f, z.x)
R2.resid(z.f, z.v)
```
# PGLS with a single fixed effect

```r
define the number of data points
n <- 100

define the dataset
锣 <- data.frame(x = array(0, dim = n), y = 0)

define the fixed effect
b1 <- 1.5

define the signal
signal <- 0.7

compute branch lengths
phy <- compute.brlen(rtree(n = n), method = 'Grafen', power = 1)
phy.x <- compute.brlen(phy, method = 'Grafen', power = 0.0001)

# Generate random data
x <- rTraitCont(phy.x, model = 'BM', sigma = 1)
e <- signal^0.5 * rTraitCont(phy, model = 'BM', sigma = 1) + (1-signal)^0.5 * rnorm(n = n)
d$x <- x[match(names(e), names(x))]
d$y <- b1 * x + e
rownames(d) <- phy$tip.label

# Model fitting
z.x <- phylolm(y ~ 1, phy = phy, data = d, model = 'lambda')
lam.x <- round(z.x$optpar, digits = 4)
z.f <- phylolm(y ~ x, phy = phy, data = d, model = 'lambda')
z.v <- lm(y ~ x, data = d)

R2.resid(z.f, z.x, phy = phy)
R2.resid(z.f, z.v, phy = phy)
R2.resid(z.f, phy = phy)

# This also works for models fit with gls() in {nlme}
z.x <- gls(y ~ 1, data = d, correlation = corPagel(1, phy), method = "ML")
z.f <- gls(y ~ x, data = d, correlation = corPagel(1, phy), method = "ML")
z.v <- lm(y ~ x, data = d)

R2.resid(z.f, z.x)
R2.resid(z.f, z.v)
R2.resid(z.f)

# But note that you need to define weights for gls() with non-ultrametric trees;
# if not, you will get an error "Matrix is not block-diagonal"

phy.nu <- rtree(n = n)

# Generate random data
e <- signal^0.5 * rTraitCont(phy.nu, model = 'BM', sigma = 1) + (1-signal)^0.5 * rnorm(n = n)
d$x <- x[match(names(e), names(x))]
d$y <- b1 * x + e
rownames(d) <- phy.nu$tip.label

weights <- diag(vcv.phylo(phy.nu))
z.x <- gls(y ~ 1, data = d, correlation = corPagel(1, phy.nu),

```
weights=varFixed(~weights), method = "ML")

z.f <- gls(y ~ x, data = d,
  correlation = corPagel(1, phy.nu),
  weights=varFixed(~weights), method = "ML")

z.v <- lm(y ~ x, data = d)

R2.resid(z.f, z.x)
R2.resid(z.f, z.v)
R2.resid(z.f)

#################
# PGLMM with one fixed effect

n <- 100
b1 <- 1.5
signal <- 2

phy <- compute.brlen(rtree(n = n), method = 'Grafen', power = 1)
phy.x <- compute.brlen(phy, method = 'Grafen', power = .0001)

# Generate random data
x <- rnorm(n)
d <- data.frame(x = x, y = 0)

e <- signal * rTraitCont(phy, model = 'BM', sigma = 1)
e <- e[match(phy$tip.label, names(e))]

d$y <- rbinom(n = n, size = 1, prob = inv.logit(b1 * d$x + e))
rownames(d) <- phy$tip.label

# Use the function binaryPGLMM() from the rr2 package rather than ape.
z.f <- rr2::binaryPGLMM(y ~ x, data = d, phy = phy)
z.x <- rr2::binaryPGLMM(y ~ 1, data = d, phy = phy)
z.v <- glm(y ~ x, data = d, family = 'binomial')

R2.resid(z.f, z.x, phy = phy)
R2.resid(z.f, z.v, phy = phy)
R2.resid(z.f, phy = phy)

---

**rr2**

**rr2: An R package to calculate R2s for regression models**

---

**Description**

The rr2 package provides methods to calculate R2 for models with correlated errors, including Phylogenetic GLS, Phylogenetic Logistic Regression, LMMs, GLMM, and PGLMM.
transf_phy

Transform a phylogeny based on a phylolm model

Description
Using a fitted phylolm model to transform branch lengths of a phylogeny

Usage
transf_phy(phylolmMod, phy)

Arguments
- phylolmMod: A fitted phylolm model.
- phy: A phylogeny with class 'phylo'.

Value
A transformed phylogeny.
Index

binaryPGLMM, 2
inv.logit, 3
partialR2, 3
partialR2adj, 4
R2, 4
R2.lik, 8
R2.pred, 12
R2.resid, 16
rr2, 21
rr2-package (rr2), 21
transf_phy, 22