Package 'galamm'

October 31, 2026

Title Generalized Additive Latent and Mixed Models

Version 0.4.2

Description Estimates generalized additive latent and mixed models using maximum marginal likelihood, as defined in Sorensen et al. (2023) <doi:10.1007/s11336-023-09910-z>, which is an extension of Rabe-Hesketh and Skrondal (2004)'s unifying framework for multilevel latent variable modeling <doi:10.1007/BF02295939>. Efficient computation is done using sparse matrix methods, Laplace approximation, and automatic differentiation. The framework includes generalized multilevel models with heteroscedastic residuals, mixed response types, factor loadings, smoothing splines, crossed random effects, and combinations thereof. Syntax for model formulation is close to 'lme4' (Bates et al. (2015) <doi:10.18637/jss.v067.i01>) and 'PLmixed' (Rockwood and Jeon (2019) <doi:10.1080/00273171.2018.1516541>).

License GPL (>= 3)

URL https://docs.ropensci.org/galamm/, https://github.com/ropensci/galamm

BugReports https://github.com/ropensci/galamm/issues

Encoding UTF-8

Imports gratia, lattice, lme4, Matrix, memoise, methods, mgcv, nlme, Rcpp, Rdpack, reformulas, stats

Depends R (>= 3.5.0)

LinkingTo Rcpp, RcppEigen

LazyData true

Suggests covr, gamm4, knitr, PLmixed, rlang, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr, rmarkdown

RdMacros Rdpack

NeedsCompilation yes

SystemRequirements C++17

Config/roxygen2/version 8.1.1

Author Øystein Sørensen [aut, cre] (ORCID:
<https://orcid.org/0000-0003-0724-3542>),
Douglas Bates [ctb],
Ben Bolker [ctb],
Martin Maechler [ctb],
Allan Leal [ctb],
Fabian Scheipl [ctb],
Steven Walker [ctb],
Simon Wood [ctb]

Maintainer Øystein Sørensen <oystein.sorensen@psykologi.uio.no>

Repository CRAN

Date/Publication 2026-10-31 08:20:21 UTC

Contents

anova.galamm . . . 3
appraise.galamm . . . 4
coef.galamm . . . 5
cognition . . . 6
confint.galamm . . . 7
derivatives.galamm . . . 8
deviance.galamm . . . 9
diet . . . 10
draw.galamm . . . 11
epilep . . . 12
extract_optim_parameters.galamm . . . 13
factor_loadings.galamm . . . 14
family.galamm . . . 15
fitted.galamm . . . 16
fixef . . . 17
formula.galamm . . . 18
galamm . . . 19
galammObject . . . 24
galamm_control . . . 25
gfam . . . 26
hsced . . . 27
latent_covariates . . . 28
latent_covariates_long . . . 28
lifespan . . . 29
logLik.galamm . . . 30
model.frame.galamm . . . 31
mresp . . . 32
mresp_hsced . . . 32

nobs.galamm 33
plot.galamm 34
plot_smooth.galamm 36
predict.galamm 37
print.galamm 39
print.summary.galamm 40
print.VarCorr.galamm 41
qqmath.galamm 42
ranef.galamm 43
residuals.galamm 44
response 45
s 46
sigma.galamm 47
summary.galamm 49
t2 50
VarCorr 51
vcov.galamm 52

Index **54**

`anova.galamm` *Compare likelihoods of galamm objects*

Description

Anova function for comparing different GALAMMs fitted on the same data.

Usage

```
## S3 method for class 'galamm'
anova(object, ...)
```

Arguments

`object`	An object of class `galamm` returned from `galamm`.
`...`	Other fitted models of class `galamm`. Currently, if no models are provided in this argument, no table will be returned.

Value

A table with model comparison metric.

Author(s)

Some of the source code for this function is adapted from `lme4:::anova.merMod`, with authors Douglas M. Bates, Martin Maechler, Ben Bolker, and Steve Walker.

References

Bates DM, Mächler M, Bolker B, Walker S (2015). "Fitting Linear Mixed-Effects Models Using Lme4." *Journal of Statistical Software*, **67**(1), 1–48. ISSN 1548-7660. doi:10.18637/jss.v067.i01.

See Also

`summary.galamm()` for the summary method and `anova()` for the generic function.

Other summary functions: `draw.galamm()`, `plot_smooth.galamm()`, `print.galamm()`, `print.summary.galamm()`, `summary.galamm()`

Examples

```
# Poisson GLMM
count_mod <- galamm(
  formula = y ~ lbas * treat + lage + v4 + (1 | subj),
  data = epilep, family = poisson
)

# Model without interaction
count_mod0 <- galamm(
  formula = y ~ lbas + treat + lage + v4 + (1 | subj),
  data = epilep, family = poisson
)

# Model comparison
anova(count_mod, count_mod0)
```

`appraise.galamm` *Gratia style model diagnostic plots*

Description

This function uses `gratia::appraise` to make model diagnostic plots. See `gratia::appraise()` for details. When `model` is not of class `galamm`, it is forwarded to `gratia::appraise()`.

Usage

```
## S3 method for class 'galamm'
appraise(model, ...)
```

Arguments

`model`	An object of class `galamm` returned from `galamm`.
`...`	Other arguments passed on to `gratia::appraise()`.

Value

A ggplot object.

See Also

Other details of model fit: VarCorr, coef.galamm(), confint.galamm(), derivatives.galamm(), deviance.galamm(), factor_loadings.galamm(), family.galamm(), fitted.galamm(), fixef, formula.galamm(), llikAIC(), logLik.galamm(), model.frame.galamm(), nobs.galamm(), predict.galamm(), print.VarCorr.galamm(), ranef.galamm(), residuals.galamm(), response(), sigma.galamm(), vcov.galamm()

Examples

```
dat <- subset(cognition, domain == 1 & item == "11")
dat$y <- dat$y[, 1]
mod <- galamm(y ~ s(x) + (1 | id), data = dat)
appraise(mod)
```

---

coef.galamm *Extract galamm coefficients*

Description

Currently, this function only returns the fixed effects.

Usage

```
## S3 method for class 'galamm'
coef(object, ...)
```

Arguments

object	An object of class galamm, from galamm.
...	Optional arguments passed on to other methods. Currently not used.

Value

A matrix with the requested coefficients.

See Also

fixef.galamm() for fixed effects, ranef.galamm() for random effects, and coef() for the generic function.

Other details of model fit: VarCorr, appraise.galamm(), confint.galamm(), derivatives.galamm(), deviance.galamm(), factor_loadings.galamm(), family.galamm(), fitted.galamm(), fixef, formula.galamm(), llikAIC(), logLik.galamm(), model.frame.galamm(), nobs.galamm(), predict.galamm(), print.VarCorr.galamm(), ranef.galamm(), residuals.galamm(), response(), sigma.galamm(), vcov.galamm()

Examples

```
# Poisson GLMM
count_mod <- galamm(
  formula = y ~ lbas * treat + lage + v4 + (1 | subj),
  data = epilep, family = poisson
)

# Extract coefficients
coef(count_mod)
```

`cognition` *Simulated Data with Measurements of Cognitive Abilities*

Description

Simulated dataset mimicking the measurement of abilities in three cognitive domains. The latent traits (cognitive ability in a given domain) are based on the functions in `mgcv::gamSim` (Wood 2017), and depend on the explanatory variable x.

Usage

```
cognition
```

Format

`cognition` **A data frame with 14400 rows and 7 columns::**

id Subject ID.

domain Factor variable denoting the cognitive domain.

x Explanatory variable.

timepoint Factor variable denoting the timepoint.

item Factor variable denoting the item within the tests of each cognitive domain.

trials Number of trials, if applicable.

y Matrix with two columns. The first column is the response variable: for domain 1 a real number, for domain 2 a binomially distributed variable based on a single trial, for domain 3 a real number. The second column equals 1 if the response is conditionally Gaussian and 2 if the response is conditionally binomial.

References

Wood SN (2017). *Generalized Additive Models: An Introduction with R*, 2 edition. Chapman and Hall/CRC.

See Also

Other datasets: `diet`, `epilep`, `hsced`, `latent_covariates`, `latent_covariates_long`, `lifespan`, `mresp`, `mresp_hsced`

confint.galamm *Confidence intervals for model parameters*

Description

Confidence intervals for model parameters

Usage

```
## S3 method for class 'galamm'
confint(object, parm, level = 0.95, method = "Wald", ...)
```

Arguments

object	An object of class galamm returned from galamm.
parm	Parameters for which to compute intervals. Use "theta" to get all variance parameters, "beta" to get all fixed regression coefficients, "lambda" to get all factor loadings, and "weights" to get all weights. The parameter can also be given as a numeric vector with indices specifying the parameters. When given as characters, the arguments are case sensitive.
level	Decimal number specifying the confidence level. Defaults to 0.95.
method	Character of length one specifying the type of confidence interval. Currently only "Wald" is available. The argument is case sensitive.
...	Other arguments passed on to other methods. Currently not used.

Value

A matrix with the requested confidence intervals.

See Also

fixef.galamm() for fixed effects, coef.galamm() for coefficients more generally, and vcov.galamm() for the variance-covariance matrix. confint() is the generic function.

Other details of model fit: VarCorr, appraise.galamm(), coef.galamm(), derivatives.galamm(), deviance.galamm(), factor_loadings.galamm(), family.galamm(), fitted.galamm(), fixef, formula.galamm(), llikAIC(), logLik.galamm(), model.frame.galamm(), nobs.galamm(), predict.galamm(), print.VarCorr.galamm(), ranef.galamm(), residuals.galamm(), response(), sigma.galamm(), vcov.galamm()

Examples

```
# Poisson GLMM
count_mod <- galamm(
  formula = y ~ lbas * treat + lage + v4 + (1 | subj),
  data = epilep, family = poisson
)
```

```
confint(count_mod, parm = "beta", level = .99)
```

`derivatives.galamm` *Derivatives of estimated smooth via finite differences*

Description

This function uses `gratia::derivatives` to compute derivatives of estimated smooth via finite differences. See `gratia::derivatives()` for details. When `object` is not of class `galamm`, it is forwarded to `gratia::derivatives()`.

Usage

```
## S3 method for class 'galamm'
derivatives(object, ...)
```

Arguments

`object`	An object of class `galamm` returned from `galamm`.
`...`	Other arguments passed on to `gratia::derivatives()`.

Value

A tibble.

See Also

Other details of model fit: `VarCorr`, `appraise.galamm()`, `coef.galamm()`, `confint.galamm()`, `deviance.galamm()`, `factor_loadings.galamm()`, `family.galamm()`, `fitted.galamm()`, `fixef`, `formula.galamm()`, `llikAIC()`, `logLik.galamm()`, `model.frame.galamm()`, `nobs.galamm()`, `predict.galamm()`, `print.VarCorr.galamm()`, `ranef.galamm()`, `residuals.galamm()`, `response()`, `sigma.galamm()`, `vcov.galamm()`

Examples

```
dat <- subset(cognition, domain == 1 & item == "11")
dat$y <- dat$y[, 1]
mod <- galamm(y ~ s(x) + (1 | id), data = dat)

dd <- derivatives(mod)
draw(dd)
```

`deviance.galamm` *Extract deviance of galamm object*

Description

Extract deviance of galamm object

Usage

```
## S3 method for class 'galamm'
deviance(object, ...)
```

Arguments

`object`	Object of class `galamm`, returned from `galamm`.
`...`	Other arguments passed on to other methods. Currently not used.

Value

A numeric value giving the deviance of the model fit.

See Also

`logLik.galamm()` for a function returning the log likelihood and `deviance()` for the generic function.

Other details of model fit: `VarCorr`, `appraise.galamm()`, `coef.galamm()`, `confint.galamm()`, `derivatives.galamm()`, `factor_loadings.galamm()`, `family.galamm()`, `fitted.galamm()`, `fixef`, `formula.galamm()`, `llikAIC()`, `logLik.galamm()`, `model.frame.galamm()`, `nobs.galamm()`, `predict.galamm()`, `print.VarCorr.galamm()`, `ranef.galamm()`, `residuals.galamm()`, `response()`, `sigma.galamm()`, `vcov.galamm()`

Examples

```
# Linear mixed model with heteroscedastic residuals
mod <- galamm(
  formula = y ~ x + (1 | id),
  dispformula = ~ (1 | item),
  data = hsced
)

# Extract deviance
deviance(mod)
```

`diet` *Diet Data*

Description

Longitudinal epilepsy data from Morris et al. (1977). This documentation is based on Chapter 14.2 of Skrondal and Rabe-Hesketh (2004), where the dataset is used. See also Rabe-Hesketh et al. (2003).

Usage

```
diet
```

Format

`diet` **A data frame with 236 rows and 7 columns::**

id Subject ID.

age Age (standardized).

bus Dummy variable indicating whether the subject is a bus driver or banking staff.

item Integer indicating whether the outcome is fiber intake at time 1 (item = 1), fiber intake at time 2 (item = 2), or coronary heart disease (item = 3).

y Matrix with two column. The first column is the outcome, and the second column an index which equals 1 if the response is conditionally Gaussian and 2 if the response is conditionally binomial.

chd Dummy variable indicating whether y is an indicator for coronary heart disease, coded as 0/1.

fiber Dummy variable indicating whether y is a fiber measurement at either timepoint 1 or 2.

fiber2 Dummy variable indicating whether y is a fiber measurement at timepoint 2.

Source

GLLAMM, Chapter 14.2.

References

Morris JN, Marr JW, Clayton DG (1977). "Diet and Heart: A Postscript." *Br Med J*, **2**(6098), 1307–1314. ISSN 0007-1447, 1468-5833. doi:10.1136/bmj.2.6098.1307.

Rabe-Hesketh S, Pickles A, Skrondal A (2003). "Correcting for Covariate Measurement Error in Logistic Regression Using Nonparametric Maximum Likelihood Estimation." *Statistical Modelling*, **3**(3), 215–232. ISSN 1471-082X. doi:10.1191/1471082X03st056oa.

Skrondal A, Rabe-Hesketh S (2004). *Generalized Latent Variable Modeling*, Interdisciplinary Statistics Series. Chapman and Hall/CRC, Boca Raton, Florida.

See Also

Other datasets: cognition, epilep, hsced, latent_covariates, latent_covariates_long, lifespan, mresp, mresp_hsced

---

`draw.galamm` *Draw method for galamm objects*

Description

This function uses `gratia::draw` to visualize smooth terms. See `gratia::draw()` for details. When `object` is not of class `galamm`, it is forwarded to `gratia::draw()`.

Usage

```
## S3 method for class 'galamm'
draw(object, ...)
```

Arguments

`object`	An object of class `galamm` returned from `galamm`.
`...`	Other arguments passed on to `gratia::draw()`.

Value

A ggplot object.

See Also

Other summary functions: `anova.galamm()`, `plot_smooth.galamm()`, `print.galamm()`, `print.summary.galamm()`, `summary.galamm()`

Examples

```
dat <- subset(cognition, domain == 1 & item == "11")
dat$y <- dat$y[, 1]
mod <- galamm(y ~ s(x) + (1 | id), data = dat)

draw(mod)
```

`epilep` *Epilepsy Data*

Description

Longitudinal epilepsy data from Leppik et al. (1987). This documentation is based on Chapter 11.3 of Skrondal and Rabe-Hesketh (2004), where the dataset is used.

Usage

```
epilep
```

Format

`epilep` **A data frame with 236 rows and 7 columns::**

subj Subject ID.

y Number of seizures.

treat Dummy variable for treatment group.

visit Time at visit.

v4 Dummy for visit 4.

lage Logarithm of age.

lbas Logarithm of a quarter of the number of seizures in the eight weeks preceding entry into the trial.

Source

GLLAMM, Chapter 11.3.

References

Leppik IE, Dreifuss FE, Porter R, Bowman T, Santilli N, Jacobs M, Crosby C, Cloyd J, Stackman J, Graves N, Sutula T, Welty T, Vickery J, Brundage R, Gates J, Gumnit RJ, Gutierrez A (1987). "A Controlled Study of Progabide in Partial Seizures: Methodology and Results." *Neurology*, **37**(6), 963–963. ISSN 0028-3878, 1526-632X. doi:10.1212/WNL.37.6.963.

Skrondal A, Rabe-Hesketh S (2004). *Generalized Latent Variable Modeling*, Interdisciplinary Statistics Series. Chapman and Hall/CRC, Boca Raton, Florida.

See Also

Other datasets: `cognition`, `diet`, `hsced`, `latent_covariates`, `latent_covariates_long`, `lifespan`, `mresp`, `mresp_hsced`

`extract_optim_parameters.galamm` *Extract parameters from fitted model for use as initial values*

Description

This function extracts parameter values from a fitted model object in a form that can be directly provided as initial values for a new model fit.

Usage

```
## S3 method for class 'galamm'
extract_optim_parameters(object)
```

Arguments

`object`	Object of class `galamm` returned from `galamm`.

Value

A `list` object containing the following elements:

- `theta` Numerical vector of variance components, i.e., entries of the lower Cholesky form of the covariance matrix of random effects.
- `beta` Fixed regression coefficients.
- `lambda` Factor loadings.
- `weights` Weights for heteroscedastic residuals.

See Also

Other optimization functions: `galamm_control()`

Examples

```
# Fit linear mixed model with heteroscedastic residuals
mod <- galamm(
  formula = y ~ x + (1 | id),
  dispformula = ~ (1 | item),
  data = hsced
)

# Extract parameters
start <- extract_optim_parameters(mod)

# Fit again using the Nelder-Mead algorithm, using start as initial values:
mod_nm <- galamm(
  formula = y ~ x + (1 | id),
  dispformula = ~ (1 | item),
```

```
  data = hsced,
  start = start,
  control = galamm_control(method = "Nelder-Mead")
)
```

`factor_loadings.galamm` *Extract factor loadings from galamm object*

Description

Extract factor loadings from galamm object

Usage

```
## S3 method for class 'galamm'
factor_loadings(object)
```

Arguments

`object`	Object of class `galamm` returned from `galamm`.

Details

This function has been named `factor_loadings` rather than just `loadings` to avoid conflict with `stats::loadings`.

Value

A matrix containing the estimated factor loadings with corresponding standard deviations.

Author(s)

The example for this function comes from `PLmixed`, with authors Nicholas Rockwood and Minjeong Jeon (Rockwood and Jeon 2019).

See Also

`fixef.galamm()` for fixed regression coefficients, `confint.galamm()` for confidence intervals, and `coef.galamm()` for coefficients more generally.

Other details of model fit: `VarCorr`, `appraise.galamm()`, `coef.galamm()`, `confint.galamm()`, `derivatives.galamm()`, `deviance.galamm()`, `family.galamm()`, `fitted.galamm()`, `fixef`, `formula.galamm()`, `llikAIC()`, `logLik.galamm()`, `model.frame.galamm()`, `nobs.galamm()`, `predict.galamm()`, `print.VarCorr.galamm()`, `ranef.galamm()`, `residuals.galamm()`, `response()`, `sigma.galamm()`, `vcov.galamm()`

Examples

```
# Logistic mixed model with factor loadings, example from PLmixed
data("IRTsim", package = "PLmixed")

# Reduce data size for the example to run faster
IRTsub <- IRTsim[IRTsim$item < 4, ]
IRTsub <- IRTsub[sample(nrow(IRTsub), 300), ]
IRTsub$item <- factor(IRTsub$item)

# Fix loading for first item to 1, and estimate the two others freely
loading_matrix <- matrix(c(1, NA, NA), ncol = 1)

# Estimate model
mod <- galamm(y ~ item + (0 + ability | sid) + (0 + ability | school),
  data = IRTsub, family = binomial, load_var = "item",
  factor = "ability", lambda = loading_matrix
)

# Show estimated factor loadings, with standard errors
factor_loadings(mod)
```

`family.galamm` *Extract family or families from fitted galamm*

Description

This function returns a list of families for an object of class `galamm`, returned from `galamm`.

Usage

```
## S3 method for class 'galamm'
family(object, ...)
```

Arguments

`object`	An object of class `galamm` returned from `galamm`.
`...`	Optional arguments passed on to other methods. Currently not used.

Value

A list of family objects.

See Also

galamm()

Other details of model fit: VarCorr, appraise.galamm(), coef.galamm(), confint.galamm(), derivatives.galamm(), deviance.galamm(), factor_loadings.galamm(), fitted.galamm(), fixef, formula.galamm(), llikAIC(), logLik.galamm(), model.frame.galamm(), nobs.galamm(), predict.galamm(), print.VarCorr.galamm(), ranef.galamm(), residuals.galamm(), response(), sigma.galamm(), vcov.galamm()

Examples

```
# Mixed response model
loading_matrix <- matrix(c(1, NA), ncol = 1)
families <- gfam(list(gaussian, binomial))

mixed_resp <- galamm(
  formula = y ~ x + (0 + level | id),
  data = mresp,
  family = families,
  load_var = "itemgroup",
  lambda = loading_matrix,
  factor = "level"
)

# This model has two family objects
family(mixed_resp)
```

---

fitted.galamm *Extract model fitted values*

Description

Extracts fitted values from a model including random effects.

Usage

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

Arguments

object	An object of class galamm returned from galamm.
...	Optional arguments passed on to other methods. Currently not used.

Value

A numerical vector with fit values for each row in the input data.

See Also

Other details of model fit: VarCorr, appraise.galamm(), coef.galamm(), confint.galamm(), derivatives.galamm(), deviance.galamm(), factor_loadings.galamm(), family.galamm(), fixef, formula.galamm(), llikAIC(), logLik.galamm(), model.frame.galamm(), nobs.galamm(), predict.galamm(), print.VarCorr.galamm(), ranef.galamm(), residuals.galamm(), response(), sigma.galamm(), vcov.galamm()

Examples

```
# Linear mixed model with heteroscedastic residuals
mod <- galamm(
  formula = y ~ x + (1 | id),
  dispformula = ~ (1 | item),
  data = hsced
)

# Extract fitted values and plot against x
plot(hsced$x, fitted(mod))
```

---

fixef *Extract fixed effects from galamm objects*

Description

Extract the fixed regression coefficients.

Usage

```
## S3 method for class 'galamm'
fixef(object, ...)
```

Arguments

object	An object of class galamm, returned from galamm.
...	Optional arguments passed on to other methods. Currently not used.

Value

A named numeric vector containing the requested fixed effects.

See Also

ranef.galamm() for random effects, coef.galamm() for coefficients more generally, and confint.galamm() for confidence intervals.

Other details of model fit: VarCorr, appraise.galamm(), coef.galamm(), confint.galamm(), derivatives.galamm(), deviance.galamm(), factor_loadings.galamm(), family.galamm(),

fitted.galamm(), formula.galamm(), llikAIC(), logLik.galamm(), model.frame.galamm(), nobs.galamm(), predict.galamm(), print.VarCorr.galamm(), ranef.galamm(), residuals.galamm(), response(), sigma.galamm(), vcov.galamm()

Examples

```
# Poisson GLMM
count_mod <- galamm(
  formula = y ~ lbas * treat + lage + v4 + (1 | subj),
  data = epilep, family = poisson
)

# Extract fixed effects
fixef(count_mod)
```

---

formula.galamm *Extract formula from fitted galamm object*

Description

Extract formula from fitted galamm object

Usage

```
## S3 method for class 'galamm'
formula(x, ...)
```

Arguments

x	Object of class galamm returned from galamm.
...	Optional arguments passed on to other methods. Currently not used.

Value

The formula used to fit the model.

See Also

Other details of model fit: VarCorr, appraise.galamm(), coef.galamm(), confint.galamm(), derivatives.galamm(), deviance.galamm(), factor_loadings.galamm(), family.galamm(), fitted.galamm(), fixef, llikAIC(), logLik.galamm(), model.frame.galamm(), nobs.galamm(), predict.galamm(), print.VarCorr.galamm(), ranef.galamm(), residuals.galamm(), response(), sigma.galamm(), vcov.galamm()

Examples

```
# Mixed response model ----------------------------------------------------

# The mresp dataset contains a mix of binomial and Gaussian responses.

# We need to estimate a factor loading which scales the two response types.
loading_matrix <- matrix(c(1, NA), ncol = 1)

# Define mapping to families.
families <- gfam(list(gaussian, binomial))


# Fit the model
mod <- galamm(
  formula = y ~ x + (0 + level | id),
  data = mresp,
  family = families,
  factor = "level",
  load_var = "itemgroup",
  lambda = loading_matrix
)

# Formula
formula(mod)
```

`galamm` *Fit a generalized additive latent and mixed model*

Description

This function fits a generalized additive latent and mixed model (GALAMMs), as described in Sørensen et al. (2023). The building blocks of these models are generalized additive mixed models (GAMMs) (Wood 2017), of which generalized linear mixed models (Breslow and Clayton 1993; Harville 1977; Henderson 1975; Laird and Ware 1982) are special cases. GALAMMs extend upon GAMMs by allowing factor structures, as commonly used to model hypothesized latent traits underlying observed measurements. In this sense, GALAMMs are an extension of generalized linear latent and mixed models (GLLAMMs) (Skrondal and Rabe-Hesketh 2004; Rabe-Hesketh et al. 2004) which allows semiparametric estimation. The implemented algorithm used to compute model estimates is described in Sørensen et al. (2023), and is an extension of the algorithm used for fitting generalized linear mixed models by the `lme4` package (Bates et al. 2015). The syntax used to define factor structures is based on that used by the `PLmixed` package, which is detailed in Rockwood and Jeon (2019).

Usage

```
galamm(
  formula,
```

```
  dispformula = NULL,
  weights = NULL,
  data,
  family = gaussian,
  family_mapping = NULL,
  load_var = NULL,
  load.var = NULL,
  lambda = NULL,
  factor = NULL,
  factor_interactions = NULL,
  na.action = getOption("na.action"),
  start = NULL,
  control = galamm_control()
)
```

Arguments

formula	A formula specifying the model. Smooth terms are defined in the style of the mgcv and gamm4 packages, see (Wood 2017) for an introduction. Random effects are specified using lme4 syntax, which is described in detail in (Bates et al. 2015). Factor loadings will also be part of the model formula, and is based on the syntax of the PLmixed package (Rockwood and Jeon 2019).
dispformula	An optional formula object specifying an expression for the residual variance. Defaults to NULL, corresponding to homoscedastic errors. The formula is defined in lme4 style; see vignettes and examples for details.
weights	Deprecated. Use dispformula instead.
data	A data.frame containing all the variables specified by the model formula, with the exception of factor loadings.
family	A family object specifying the response distribution of the model. Currently gaussian, binomial, and poisson with canonical link functions are supported. Models with mixed response types, in which responses are distributed according to one of the three supported distributions, can be set up using the gfam() function.
family_mapping	Deprecated. See the documentation to gfam() for how to set up mixed response type models.
load_var	Optional character specifying the name of the variable in data identifying what the factors load onto. Default to NULL, which means that there are no loading variables. Argument is case sensitive.
load.var	Deprecated. Use load_var instead.
lambda	Optional factor loading matrix. Numerical values indicate that the given value is fixed, while NA means that the entry is a parameter to be estimated. Numerical values can only take the values 0 or 1. The number of columns of lambda must be identical to the number of elements in factor. Defaults to NULL, which means that there is no factor loading matrix. If lambda is provided as a vector, it will be converted to a matrix with a single column.

`factor` Optional character vector whose jth entry corresponds to the jth column of the corresponding matrix in `lambda`. The number of elements in `factor` must be equal to the number of columns in `lambda`. Defaults to `NULL`, which means that there are no factor loadings. Argument is case sensitive.

`factor_interactions` Optional list of length equal to the number of columns in `lambda`. Each list element should be a `formula` object containing the write-hand side of a regression model, of the form `~ x + z`. Defaults to `NULL`, which means that no factor interactions are used.

`na.action` Character of length one specifying a function which indicates what should happen when the data contains `NA`s. The defaults is set to the `na.action` setting of `options`, which can be seen with `options("na.action")`. The other alternatives are `"na.fail"` or `"na.exclude"`, which means that the function fails if there as `NA`s in `data`.

`start` Optional named list of starting values for parameters. Possible names of list elements are `"theta"`, `"beta"`, `"lambda"`, and `"weights"`, all of should be numerical vectors with starting values. Default to `NULL`, which means that some relatively sensible defaults are used. Names of parameters must be given in all lower case.

`control` Optional control object for the optimization procedure of class `galamm_control` resulting from calling `galamm_control`. Defaults to `NULL`, which means that the defaults of `galamm_control` are used.

Value

An object of type `galamm` as described in galammObject.

References

Bates DM, Mächler M, Bolker B, Walker S (2015). "Fitting Linear Mixed-Effects Models Using Lme4." *Journal of Statistical Software*, **67**(1), 1–48. ISSN 1548-7660. doi:10.18637/jss.v067.i01.

Breslow NE, Clayton DG (1993). "Approximate Inference in Generalized Linear Mixed Models." *Journal of the American Statistical Association*, **88**(421), 9–25. ISSN 0162-1459. doi:10.2307/2290687.

Harville DA (1977). "Maximum Likelihood Approaches to Variance Component Estimation and to Related Problems." *Journal of the American Statistical Association*, **72**(358), 320–338. ISSN 0162-1459. doi:10.2307/2286796.

Henderson CR (1975). "Best Linear Unbiased Estimation and Prediction under a Selection Model." *Biometrics*, **31**(2), 423–447. ISSN 0006-341X. doi:10.2307/2529430.

Laird NM, Ware JH (1982). "Random-Effects Models for Longitudinal Data." *Biometrics*, **38**(4), 963–974. ISSN 0006-341X. doi:10.2307/2529876.

Rabe-Hesketh S, Skrondal A, Pickles A (2004). "Generalized Multilevel Structural Equation Modeling." *Psychometrika*, **69**(2), 167–190. ISSN 1860-0980. doi:10.1007/BF02295939.

Rockwood NJ, Jeon M (2019). "Estimating Complex Measurement and Growth Models Using the R Package PLmixed." *Multivariate Behavioral Research*, **54**(2), 288–306. ISSN 0027-3171. doi:10.1080/00273171.2018.1516541.

Skrondal A, Rabe-Hesketh S (2004). *Generalized Latent Variable Modeling*, Interdisciplinary Statistics Series. Chapman and Hall/CRC, Boca Raton, Florida.

Sørensen Ø, Fjell AM, Walhovd KB (2023). "Longitudinal Modeling of Age-Dependent Latent Traits with Generalized Additive Latent and Mixed Models." *Psychometrika*, **88**(2), 456–486. ISSN 1860-0980. doi:10.1007/s1133602309910z.

Wood SN (2017). *Generalized Additive Models: An Introduction with R*, 2 edition. Chapman and Hall/CRC.

See Also

Other modeling functions: `galammObject`, `gfam()`, `s`, `t2`

Examples

```
# Mixed response model ------------------------------------------------------

# The mresp dataset contains a mix of binomial and Gaussian responses.

# We need to estimate a factor loading which scales the two response types.
loading_matrix <- matrix(c(1, NA), ncol = 1)

# Define mapping to families.
families <- gfam(list(gaussian, binomial))

# Fit the model
mod <- galamm(
  formula = y ~ x + (0 + level | id),
  data = mresp,
  family = families,
  factor = "level",
  load_var = "itemgroup",
  lambda = loading_matrix
)

# Summary information
summary(mod)


# Heteroscedastic model -----------------------------------------------------
# Residuals allowed to differ according to the item variable
# We also set the initial value of the random intercept standard deviation
# to 1
mod <- galamm(
  formula = y ~ x + (1 | id), dispformula = ~ (1 | item),
```

```
  data = hsced, start = list(theta = 1)
)
summary(mod)

# Generalized additive mixed model with factor structures -------------------

# The cognition dataset contains simulated measurements of three latent
# time-dependent processes, corresponding to individuals' abilities in
# cognitive domains. We focus here on the first domain, and take a single
# random timepoint per person:
dat <- subset(cognition, domain == 1)
dat <- split(dat, f = dat$id)
dat <- lapply(dat, function(x) x[x$timepoint %in% sample(x$timepoint, 1), ])
dat <- do.call(rbind, dat)
dat$item <- factor(dat$item)

# At each timepoint there are three items measuring ability in the cognitive
# domain. We fix the factor loading for the first measurement to one, and
# estimate the remaining two. This is specified in the loading matrix.
loading_matrix <- matrix(c(1, NA, NA), ncol = 1)

# We can now estimate the model.
mod <- galamm(
  formula = y ~ 0 + item + sl(x, factor = "loading") +
    (0 + loading | id),
  data = dat,
  load_var = "item",
  lambda = loading_matrix,
  factor = "loading"
)

# We can plot the estimated smooth term
plot_smooth(mod, shade = TRUE)


# Interaction between observed and latent covariates ------------------------
# Define the loading matrix
lambda <- matrix(c(1, NA, NA), ncol = 1)

# Define the regression functions, one for each row in the loading matrix
factor_interactions <- list(~1, ~1, ~x)

# Fit the model
mod <- galamm(
  formula = y ~ type + x:response + (0 + loading | id),
  data = latent_covariates,
  load_var = "type",
  lambda = lambda,
  factor = "loading",
  factor_interactions = factor_interactions
)

# The summary output now include an interaction between the latent variable
```

```
# and x, for predicting the third element in "type"
summary(mod)
```

`galammObject` *Class "galamm"*

Description

An S3 class for objects returned by `galamm()`. Contains information about the fitted model, random effects, parameters, and smooth terms.

Components

- `call`: the matched call used when fitting the model.
- `random_effects`: a list with:
  - `b`: random effects in original parametrization.
  - `u`: random effects standardized to have identity covariance matrix.
- `model`: a list containing:
  - `deviance`: deviance of final model.
  - `deviance_residuals`: deviance residuals.
  - `df`: degrees of freedom.
  - `family`: one or more family objects.
  - `factor_interactions`: list of formulas for latent-observed interactions.
  - `fit`: fitted values.
  - `fit_population`: fitted values excluding random effects.
  - `hessian`: Hessian matrix of the final model.
  - `lmod`: linear model object from `lme4::lFormula()`.
  - `loglik`: log-likelihood of the final model.
  - `n`: number of observations.
  - `pearson_residual`: Pearson residuals.
  - `reduced_hessian`: logical; whether full Hessian or reduced version.
  - `response`: numeric vector of response values.
  - `weights_object`: object with weights (or `NULL`).
- `parameters`: a list with model parameters and metadata:
  - `beta_inds`: indices of fixed regression coefficients.
  - `dispersion_parameter`: dispersion parameters.
  - `lambda_dummy`: dummy factor loading matrix.
  - `lambda_inds`: indices of factor loadings.
  - `lambda_interaction_inds`: indices of latent-observed interactions.
  - `parameter_estimates`: numeric vector of parameter estimates.
  - `parameter_names`: names of all parameters.
  - `theta_inds`: indices of variance components (Cholesky entries).
  - `weights_inds`: indices of estimated weights.
- `gam`: list of smooth term info (empty list if none).

See Also

Other modeling functions: `galamm()`, `gfam()`, `s`, `t2`

`galamm_control` *Control values for galamm fit*

Description

This function can be called for controling the optimization procedure used when fitting GALAMMs using `galamm`.

Usage

```
galamm_control(
  optim_control = list(),
  method = c("L-BFGS-B", "Nelder-Mead"),
  maxit_conditional_modes = 10,
  pirls_tol_abs = 0.01,
  reduced_hessian = FALSE
)
```

Arguments

`optim_control`	List containing optimization parameters. If `method = "L-BFGS-B"` it is passed on to `stats::optim`'s `control` argument and if `method = "Nelder-Mead"`, it is passed on to `lme4::Nelder_Mead`'s control argument. If not otherwise specified, and `method = "L-BFGS-B"`, the following arguments are set to non-default values: `fnscale = -1` and `lmm = 20`.
`method`	Character string defining the algorithm to be used for maximizing the marginal log-likelihood. The default is `"L-BFGS-B"`, which uses the limited memory Broyden-Fletcher-Goldfarb-Shanno algorithm with box constrained as implemented in `stats::optim`. The other options is `"Nelder-Mead"`, which calls the Nelder-Mead algorithm with box constraints implemented in `lme4::Nelder_Mead`. The argument is case sensitive.
`maxit_conditional_modes`	Maximum number of iterations in penalized iteratively reweighted least squares algorithm. Ignored if `family = "gaussian"` for all observations, since then a single step gives the exact answer.
`pirls_tol_abs`	Absolute convergence criterion for penalized iteratively reweighted least squares algorithm. Defaults to 0.01, which means that when the reduction in marginal likelihood between two iterations is below 0.01, the iterations stop.
`reduced_hessian`	Logical value. Defaults to `TRUE`, which means that the full Hessian matrix at the maximum marginal likelihood solution is computed. If `FALSE`, a reduced Hessian matrix with second order partial derivatives with respect to fixed regression coefficients and factor loadings. The latter can help is the full Hessian is not positive definite.

Value

Object of class `galamm_control`, which typically will be provided as an argument to `galamm`.

References

Bates DM, Mächler M, Bolker B, Walker S (2015). "Fitting Linear Mixed-Effects Models Using Lme4." *Journal of Statistical Software*, **67**(1), 1–48. ISSN 1548-7660. doi:10.18637/jss.v067.i01.

BROYDEN CG (1970). "The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations." *IMA Journal of Applied Mathematics*, **6**(1), 76–90. ISSN 0272-4960. doi:10.1093/imamat/6.1.76.

Byrd RH, Lu P, Nocedal J, Zhu C (1995). "A Limited Memory Algorithm for Bound Constrained Optimization." *SIAM Journal on Scientific Computing*, **16**(5), 1190–1208. ISSN 1064-8275. doi:10.1137/0916069.

Fletcher R (1970). "A New Approach to Variable Metric Algorithms." *The Computer Journal*, **13**(3), 317–322. ISSN 0010-4620. doi:10.1093/comjnl/13.3.317.

Goldfarb D (1970). "A Family of Variable-Metric Methods Derived by Variational Means." *Mathematics of Computation*, **24**(109), 23–26. ISSN 0025-5718, 1088-6842. doi:10.1090/S0025571819700258249 6.

Nelder JA, Mead R (1965). "A Simplex Method for Function Minimization." *The Computer Journal*, **7**(4), 308–313. ISSN 0010-4620. doi:10.1093/comjnl/7.4.308.

Shanno DF (1970). "Conditioning of Quasi-Newton Methods for Function Minimization." *Mathematics of Computation*, **24**(111), 647–656. ISSN 0025-5718, 1088-6842. doi:10.1090/S0025571819700274029X.

See Also

`galamm()`

Other optimization functions: `extract_optim_parameters.galamm()`

Examples

```
# Define control object with quite a high degree of verbosity (trace = 6)
# and using the last 20 BFGS updates to estimate the Hessian in L-BFGS-B.
control <- galamm_control(optim_control = list(trace = 6, lmm = 20))
```

`gfam` *Grouped families*

Description

This function is inspired by a function of the same name in the `mgcv` package (Wood 2017), and supports setting up mixed response type models. When using this function, the response variable must be supported as a two-column matrix, in which the first column contains the response and the second column contains an index mapping the response to the list elements provided in the argument `fl` to this function.

Usage

```
gfam(fl)
```

Arguments

fl	A list of families. Currently gaussian, binomial, and poisson with canonical link functions are supported.

Value

An object of class "galamm_extended_family".

References

Wood SN (2017). *Generalized Additive Models: An Introduction with R*, 2 edition. Chapman and Hall/CRC.

See Also

Other modeling functions: galamm(), galammObject, s, t2

---

hsced *Example Data with Heteroscedastic Residuals*

Description

Simulated dataset with residual standard deviation that varies between items.

Usage

```
hsced
```

Format

hsced **A data frame with 1200 rows and 5 columns::**

id Subject ID.

age Timepoint.

item Item indicator.

x Explanatory variable

y Outcome.

References

There are no references for Rd macro \insertAllCites on this help page.

See Also

Other datasets: cognition, diet, epilep, latent_covariates, latent_covariates_long, lifespan, mresp, mresp_hsced

latent_covariates *Simulated Data with Latent and Observed Covariates Interaction*

Description

Simulated dataset for use in examples and testing with a latent covariate interacting with an observed covariate.

Usage

```
latent_covariates
```

Format

`latent_covariates` **A data frame with 600 rows and 5 columns::**

id Subject ID.

type Type of observation in the `y` variable. If it equals `"measurement1"` or `"measurement2"` then the observation is a measurement of the latent variable. If it equals `"response"`, then the observation is the actual response.

x Explanatory variable.

y Observed response. Note, this includes both the actual response, and the measurements of the latent variable, since mathematically they are all treated as responses.

response Dummy variable indicating whether the given row is a response or not.

See Also

Other datasets: `cognition`, `diet`, `epilep`, `hsced`, `latent_covariates_long`, `lifespan`, `mresp`, `mresp_hsced`

latent_covariates_long *Simulated Longitudinal Data with Latent and Observed Covariates Interaction*

Description

Simulated dataset for use in examples and testing with a latent covariate interacting with an observed covariate. In this data, each response has been measured six times for each subject.

Usage

```
latent_covariates_long
```

Format

`latent_covariates_long` **A data frame with 800 rows and 5:**

columns:

id Subject ID.

type Type of observation in the `y` variable. If it equals `"measurement1"` or `"measurement2"` then the observation is a measurement of the latent variable. If it equals `"response"`, then the observation is the actual response.

x Explanatory variable.

y Observed response. Note, this includes both the actual response, and the measurements of the latent variable, since mathematically they are all treated as responses.

response Dummy variable indicating whether the given row is a response or not.

See Also

Other datasets: `cognition`, `diet`, `epilep`, `hsced`, `latent_covariates`, `lifespan`, `mresp`, `mresp_hsced`

`lifespan` *Simulated Dataset with Lifespan Trajectories of Three Cognitive Domains*

Description

This dataset is simulated based on the data used in Section 4.1 of Sørensen et al. (2023).

Usage

```
lifespan
```

Format

`lifespan` **A data frame with 54,457 rows and 7 columns::**

id Subject ID.

domain Cognitive domain being measured. One of `"epmem"` for episodic memory, `"wmem"` for working memory and `"execfun"` for executive function/speed.

timepoint Integer indicating the timepoint number.

age Age of participant at the timepoint.

test The particular test at this observation.

y Response. For `"epmem"` and `"wmem"` this is the number of successes in 16 trials. For `"execfun"` it is the time in seconds to complete the task.

retest Integer indicating whether the participant has taken the test at a previous timepoint.

domainepmem Dummy variable for `domain=="epmem"`.

domainwmem Dummy variable for `domain=="wmem"`.

domainexecfun Dummy variable for `domain=="execfun"`.

References

Sørensen Ø, Fjell AM, Walhovd KB (2023). "Longitudinal Modeling of Age-Dependent Latent Traits with Generalized Additive Latent and Mixed Models." *Psychometrika*, **88**(2), 456–486. ISSN 1860-0980. doi:10.1007/s11336023099102.

See Also

Other datasets: `cognition`, `diet`, `epilep`, `hsced`, `latent_covariates`, `latent_covariates_long`, `mresp`, `mresp_hsced`

`logLik.galamm` *Extract Log-Likelihood of galamm Object*

Description

Extract Log-Likelihood of galamm Object

Usage

```
## S3 method for class 'galamm'
logLik(object, ...)
```

Arguments

`object`	Object
`...`	Other arguments

Value

Object of class `logLik`

See Also

`deviance.galamm()` for a function returning deviance and `logLik()` for the generic function.

Other details of model fit: `VarCorr`, `appraise.galamm()`, `coef.galamm()`, `confint.galamm()`, `derivatives.galamm()`, `deviance.galamm()`, `factor_loadings.galamm()`, `family.galamm()`, `fitted.galamm()`, `fixef`, `formula.galamm()`, `llikAIC()`, `model.frame.galamm()`, `nobs.galamm()`, `predict.galamm()`, `print.VarCorr.galamm()`, `ranef.galamm()`, `residuals.galamm()`, `response()`, `sigma.galamm()`, `vcov.galamm()`

Examples

```
# Linear mixed model with heteroscedastic residuals
mod <- galamm(
  formula = y ~ x + (1 | id),
  dispformula = ~ (1 | item),
  data = hsced
)

# Extract log likelihood
logLik(mod)
```

`model.frame.galamm` *Extract the model frame from a galamm object*

Description

Extract the model frame from a galamm object

Usage

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

Arguments

`formula`	An object of type `galamm` as described in galammObject.
`...`	Other arguments. Currently not used.

Value

A data frame.

See Also

Other details of model fit: `VarCorr`, `appraise.galamm()`, `coef.galamm()`, `confint.galamm()`, `derivatives.galamm()`, `deviance.galamm()`, `factor_loadings.galamm()`, `family.galamm()`, `fitted.galamm()`, `fixef`, `formula.galamm()`, `llikAIC()`, `logLik.galamm()`, `nobs.galamm()`, `predict.galamm()`, `print.VarCorr.galamm()`, `ranef.galamm()`, `residuals.galamm()`, `response()`, `sigma.galamm()`, `vcov.galamm()`

mresp *Simulated Mixed Response Data*

Description

Very basic mixed response dataset with one set of normally distributed responses and one set of binomially distributed responses.

Usage

```
mresp
```

Format

`mresp` **A data frame with 4000 rows and 5 columns::**

id Subject ID.

x Predictor variable.

y Matrix whose first column is the response and whose second column equals 1 if the observation is conditionally normally distributed and 2 if the observation is conditionally binomially distributed.

itemgroup Factor variable which equals "a" for the normally distributed responses and "b" for the binomially distributed response (with 1 trial).

See Also

Other datasets: `cognition`, `diet`, `epilep`, `hsced`, `latent_covariates`, `latent_covariates_long`, `lifespan`, `mresp_hsced`

mresp_hsced *Simulated Mixed Response Data with Heteroscedastic Residuals*

Description

Mixed response dataset with one set of normally distributed responses and one set of binomially distributed responses. The normally distributed response follow two different residual standard deviations.

Usage

```
mresp_hsced
```

Format

`mresp_hsced` **A data frame with 4000 rows and 5 columns::**

id Subject ID.

x Predictor variable.

y Matrix whose first column is the response and whose second column equals 1 if the observation is conditionally normally distributed and 2 if the observation is conditionally binomially distributed.

itemgroup Factor variable which equals "a" for the normally distributed responses and "b" for the binomially distributed response (with 1 trial).

grp Grouping variable denoting which of the two residual standard deviations apply. Only relevant for the normally distributed responses.

isgauss Dummy variable indicating whether the observation on the given line is normally (Gaussian) distributed or not.

See Also

Other datasets: `cognition`, `diet`, `epilep`, `hsced`, `latent_covariates`, `latent_covariates_long`, `lifespan`, `mresp`

`nobs.galamm` *Extract the Number of Observations from a galamm Fit*

Description

Extract the Number of Observations from a galamm Fit

Usage

```
## S3 method for class 'galamm'
nobs(object, ...)
```

Arguments

`object`	An object of class `galamm` returned from `galamm`.
`...`	Optional arguments passed on to other methods. Currently not used.

Value

A number

See Also

Other details of model fit: `VarCorr`, `appraise.galamm()`, `coef.galamm()`, `confint.galamm()`, `derivatives.galamm()`, `deviance.galamm()`, `factor_loadings.galamm()`, `family.galamm()`, `fitted.galamm()`, `fixef`, `formula.galamm()`, `llikAIC()`, `logLik.galamm()`, `model.frame.galamm()`, `predict.galamm()`, `print.VarCorr.galamm()`, `ranef.galamm()`, `residuals.galamm()`, `response()`, `sigma.galamm()`, `vcov.galamm()`

Examples

```
# Example model from lme4
data(sleepstudy, package = "lme4")
fm1 <- galamm(Reaction ~ Days + (Days | Subject), data = sleepstudy)

# There are 180 observations, which matches the number of rows in sleepstudy
nobs(fm1)
nrow(sleepstudy)
```

`plot.galamm` *Diagnostic plots for galamm objects*

Description

This function provides diagnostic plots for models fitted with `galamm()`. See the `residuals.galamm()` function for definition of the residuals being used.

Usage

```
## S3 method for class 'galamm'
plot(x, form = resid(., type = "pearson") ~ fitted(.), abline = NULL, ...)
```

Arguments

`x`	An object of class `galamm` returned from `galamm`.
`form`	An option formula specifying the desired type of plot. Conditioning variables are specified with a vertical bar.
`abline`	An optional numeric vector specifying the intercept and slope of a line to be added to the plot.
`...`	Optional arguments passed on to the plot functions in the `lattice` package.

Details

The interface of this function is designed to be similar to the `plot.merMod` function from `lme4` (Bates et al. 2015).

Value

A plot is displayed.

Author(s)

Douglas Bates, Martin Maechler, Ben Bolker, and Steven Walker, with modifications by Øystein Sørensen.

References

Bates DM, Mächler M, Bolker B, Walker S (2015). "Fitting Linear Mixed-Effects Models Using Lme4." *Journal of Statistical Software*, **67**(1), 1–48. ISSN 1548-7660. doi:10.18637/jss.v067.i01.

See Also

`residuals.galamm()` for extracting residuals and `plot()` for the generic function.

Other diagnostics: `qqmath.galamm()`

Examples

```
## Linear mixed model example from lme4
data("sleepstudy", package = "lme4")
mod <- galamm(Reaction ~ Days + (Days | Subject), data = sleepstudy)

# Diagnostic plot of Pearson residuals versus fitted values
plot(mod)

# Include a straight line at zero
plot(mod, abline = c(0, 0))

# Diagnostic plot of Pearson residuals versus fitted values per subject
plot(mod, form = resid(., type = "pearson") ~ fitted(.) | Subject)

# Residuals plotted against time with a straight line at zero
plot(mod, form = resid(., type = "pearson") ~ Days, abline = c(0, 0))

# Residuals plotted against time per subject with a straight line at zero
plot(mod, form = resid(., type = "pearson") ~ Days | Subject,
     abline = c(0, 0))

# Box plot of residuals by Subject
plot(mod, Subject ~ resid(., scaled=TRUE))

## Logistic mixed model example from lme4
data("cbpp", package = "lme4")
mod <- galamm(cbind(incidence, size - incidence) ~ period + (1 | herd),
              data = cbpp, family = binomial)

# Diagnostic plot using Pearson residuals
plot(mod)

# Diagnostic plot using deviance residuals
plot(mod, resid(., type = "deviance") ~ fitted(.))

# Diagnostic plot per herd
plot(mod, resid(., type = "deviance") ~ fitted(.) | herd)

## Linear mixed model with factor structures
# See vignette on linear mixed models with factor structures for details
data(KYPSsim, package = "PLmixed")
```

```
KYPSsim <- KYPSsim[KYPSsim$sid < 100, ]
KYPSsim$time <- factor(KYPSsim$time)

loading_matrix <- rbind(c(1, 0), c(NA, 0), c(NA, 1), c(NA, NA))
factors <- c("ms", "hs")
load_var <- "time"
form <- esteem ~ time + (0 + ms | mid) + (0 + hs | hid) + (1 | sid)

mod <- galamm(formula = form, data = KYPSsim, factor = factors,
              load_var = load_var, lambda = loading_matrix)

# Pearson residuals plotted against fitted value
plot(mod)

# Actual value plotted against fitted value with a line crossing the diagonal
plot(mod, form = esteem ~ fitted(.), abline = c(0, 1))
```

`plot_smooth.galamm` *Plot smooth terms for galamm fits*

Description

Plots smooth terms of a fitted `galamm` object. This function is a thin wrapper around `mgcv::plot.gam` (Wood 2017).

Usage

```
## S3 method for class 'galamm'
plot_smooth(object, ...)
```

Arguments

`object`	Object of class `galamm` returned from `galamm`.
`...`	Other optional arguments, passed on to `mgcv::plot.gam`.

Value

A plot is displayed on the screen.

References

Wood SN (2017). *Generalized Additive Models: An Introduction with R*, 2 edition. Chapman and Hall/CRC.

See Also

Other summary functions: `anova.galamm()`, `draw.galamm()`, `print.galamm()`, `print.summary.galamm()`, `summary.galamm()`

Examples

```
# Generalized additive mixed model with factor structures -------------------

# The cognition dataset contains simulated measurements of three latent
# time-dependent processes, corresponding to individuals' abilities in
# cognitive domains. We focus here on the first domain, and take a single
# random timepoint per person:
dat <- subset(cognition, domain == 1)
dat <- split(dat, f = dat$id)
dat <- lapply(dat, function(x) x[x$timepoint %in% sample(x$timepoint, 1), ])
dat <- do.call(rbind, dat)
dat$item <- factor(dat$item)

# At each timepoint there are three items measuring ability in the cognitive
# domain. We fix the factor loading for the first measurement to one, and
# estimate the remaining two. This is specified in the loading matrix.
loading_matrix <- matrix(c(1, NA, NA), ncol = 1)

# We can now estimate the model.
mod <- galamm(
  formula = y ~ 0 + item + sl(x, factor = "loading") +
    (0 + loading | id),
  data = dat,
  load_var = "item",
  lambda = loading_matrix,
  factor = "loading"
)

# We can plot the estimated smooth term
plot_smooth(mod, shade = TRUE)

# We can turn off the rug at the bottom
plot_smooth(mod, shade = TRUE, rug = FALSE)
```

`predict.galamm` *Predictions from a model at new data values*

Description

Predictions are given at the population level, i.e., with random effects set to zero. For fitted models including random effects, see `fitted.galamm`. For mixed response models, only predictions on the scale of the linear predictors is supported.

Usage

```
## S3 method for class 'galamm'
predict(object, newdata = NULL, type = c("link", "response", "lpmatrix"), ...)
```

Arguments

object	An object of class galamm returned from galamm.
newdata	Data from for which to evaluate predictions, in a data.frame. Defaults to "NULL", which means that the predictions are evaluate at the data used to fit the model.
type	Character argument specifying the type of prediction object to be returned. Case sensitive.
...	Optional arguments passed on to other methods. Currently used for models with smooth terms, for which these arguments are forwarded to mgcv::predict.gam.

Value

A numeric vector of predicted values.

See Also

fitted.galamm() for model fits, residuals.galamm() for residuals, and predict() for the generic function.

Other details of model fit: VarCorr, appraise.galamm(), coef.galamm(), confint.galamm(), derivatives.galamm(), deviance.galamm(), factor_loadings.galamm(), family.galamm(), fitted.galamm(), fixef, formula.galamm(), llikAIC(), logLik.galamm(), model.frame.galamm(), nobs.galamm(), print.VarCorr.galamm(), ranef.galamm(), residuals.galamm(), response(), sigma.galamm(), vcov.galamm()

Examples

```
# Poisson GLMM
count_mod <- galamm(
  formula = y ~ lbas * treat + lage + v4 + (1 | subj),
  data = epilep, family = poisson
)

# Plot response versus link:
plot(
  predict(count_mod, type = "link"),
  predict(count_mod, type = "response")
)

# Predict on a new dataset
nd <- data.frame(lbas = c(.3, .2), treat = c(0, 1), lage = 0.2, v4 = -.2)
predict(count_mod, newdata = nd)
predict(count_mod, newdata = nd, type = "response")

# Semiparametric model
dat <- subset(cognition, domain == 1 & item == "11")
dat$y <- dat$y[, 1]
mod <- galamm(y ~ s(x) + (1 | id), data = dat)

# Get the linear predictor matrix
```

```
Xp <- predict(mod, type = "lpmatrix")
# Compute the estimated smooth function
# See the vignette on posterior sampling for details
betas <- coef(mod$gam)
fit <- Xp %*% betas
plot(dat$x, fit)
```

`print.galamm` *Print method for GALAMM fits*

Description

Print method for GALAMM fits

Usage

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

Arguments

`x`	An object of class `galamm` returned from `galamm`.
`...`	Further arguments passed on to other methods. Currently not used.

Value

Summary printed to screen. Invisibly returns the argument `x`.

See Also

`summary.galamm()` for the summary function and `print()` for the generic.

Other summary functions: `anova.galamm()`, `draw.galamm()`, `plot_smooth.galamm()`, `print.summary.galamm()`, `summary.galamm()`

Examples

```
# Linear mixed model with heteroscedastic residuals
mod <- galamm(
  formula = y ~ x + (1 | id),
  dispformula = ~ (1 | item),
  data = hsced
)

print(mod)
```

`print.summary.galamm` *Print method for summary GALAMM fits*

Description

Print method for summary GALAMM fits

Usage

```
## S3 method for class 'summary.galamm'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

`x`	An object of class `summary.galamm` returned from `summary.galamm`.
`digits`	Number of digits to present in outputs.
`...`	Further arguments passed on to other methods. Currently used by `stats::printCoefmat` for printing approximate significance of smooth terms.

Value

Summary printed to screen. Invisibly returns the argument `x`.

Author(s)

Some of the code for producing summary information has been derived from the summary methods of `mgcv` (author: Simon Wood) and `lme4` (Bates et al. 2015) (authors: Douglas M. Bates, Martin Maechler, Ben Bolker, and Steve Walker).

References

There are no references for Rd macro `\insertAllCites` on this help page.

See Also

`summary.galamm()` for the summary function and `print()` for the generic function.

Other summary functions: `anova.galamm()`, `draw.galamm()`, `plot_smooth.galamm()`, `print.galamm()`, `summary.galamm()`

Examples

```
# Linear mixed model with heteroscedastic residuals
mod <- galamm(
  formula = y ~ x + (1 | id),
  dispformula = ~ (1 | item),
  data = hsced
)
```

```
summary(mod)
```

`print.VarCorr.galamm` *Print method for variance-covariance objects*

Description

Print method for variance-covariance objects

Usage

```
## S3 method for class 'VarCorr.galamm'
print(
  x,
  digits = max(3, getOption("digits") - 2),
  comp = c("Std.Dev.", "Variance"),
  corr = any(comp == "Std.Dev."),
  ...
)
```

Arguments

`x`	An object of class `c("VarCorr.galamm", "VarCorr.merMod")`, returned from `VarCorr.galamm`.
`digits`	Optional arguments specifying number of digits to use when printing.
`comp`	Character vector of length 1 or 2 specifying which variance components to print. Case sensitive. Can take one of the values "Std.Dev." and "Variance".
`corr`	Logical value indicating whether covariances or correlations should be printed.
`...`	Optional arguments passed on to other methods. Currently not used.

Value

The variance-covariance information is printed to the console and the argument `x` is silently returned.

Author(s)

This function is derived from `lme4:::print.VarCorr.merMod` written by Douglas M. Bates, Martin Maechler, Ben Bolker, and Steve Walker.

References

Bates DM, Mächler M, Bolker B, Walker S (2015). "Fitting Linear Mixed-Effects Models Using Lme4." *Journal of Statistical Software*, **67**(1), 1–48. ISSN 1548-7660. doi:10.18637/jss.v067.i01.

See Also

VarCorr.galamm() for the function creating the variance-covariance objects.

Other details of model fit: VarCorr, appraise.galamm(), coef.galamm(), confint.galamm(), derivatives.galamm(), deviance.galamm(), factor_loadings.galamm(), family.galamm(), fitted.galamm(), fixef, formula.galamm(), llikAIC(), logLik.galamm(), model.frame.galamm(), nobs.galamm(), predict.galamm(), ranef.galamm(), residuals.galamm(), response(), sigma.galamm(), vcov.galamm()

Examples

```
# Linear mixed model with heteroscedastic residuals
mod <- galamm(
  formula = y ~ x + (1 | id),
  dispformula = ~ (1 | item),
  data = hsced
)

# Extract information on variance and covariance
VarCorr(mod)
```

---

qqmath.galamm *Quantile-quantile plots for galamm objects*

Description

Quantile-quantile plots for galamm objects

Usage

```
## S3 method for class 'galamm'
qqmath(x, data = NULL, ...)
```

Arguments

x	An object of class galamm, returned from galamm.
data	Ignored. Required for S3 method compatibility.
...	Optional parameters passed on to other methods. Currently not used.

Value

A quantile-quantile plot.

Author(s)

This function is derived from lme4:::qqmath.merMod, written by Douglas Bates, Martin Maechler, Ben Bolker, Steve Walker.

See Also

Other diagnostics: `plot.galamm()`

Examples

```
## Linear mixed model example from lme4
data("sleepstudy", package = "lme4")
mod <- galamm(Reaction ~ Days + (Days | Subject), data = sleepstudy)
qqmath(mod)
```

`ranef.galamm` *Extract random effects from galamm object.*

Description

Extract random effects from galamm object.

Usage

```
## S3 method for class 'galamm'
ranef(object, ...)
```

Arguments

`object`	An object of class `galamm`, returned from `galamm`.
`...`	Optional parameters passed on to other methods. Currently not used.

Value

An object of class `ranef.galamm`, containing the requested random effects.

Author(s)

This function is derived from `lme4::ranef.merMod`, written by Douglas Bates, Martin Maechler, Ben Bolker, Steve Walker.

References

Bates DM, Mächler M, Bolker B, Walker S (2015). "Fitting Linear Mixed-Effects Models Using Lme4." *Journal of Statistical Software*, **67**(1), 1–48. ISSN 1548-7660. doi:10.18637/jss.v067.i01.

See Also

fixef.galamm() for fixed effects and coef.galamm() for coefficients more generally.

Other details of model fit: VarCorr, appraise.galamm(), coef.galamm(), confint.galamm(), derivatives.galamm(), deviance.galamm(), factor_loadings.galamm(), family.galamm(), fitted.galamm(), fixef, formula.galamm(), llikAIC(), logLik.galamm(), model.frame.galamm(), nobs.galamm(), predict.galamm(), print.VarCorr.galamm(), residuals.galamm(), response(), sigma.galamm(), vcov.galamm()

Examples

```
# Poisson GLMM
count_mod <- galamm(
  formula = y ~ lbas * treat + lage + v4 + (1 | subj),
  data = epilep, family = poisson
)

# Extract random effects
ranef(count_mod)
```

---

residuals.galamm *Residuals of galamm objects*

Description

Computes residuals for models fit with galamm() using the definitions in Chapter 8 of Dunn and Smyth (2018). Define y as the response and $\hat{\mu}$ as the model fit. Importantly, $\hat{\mu}$ includes all random effects. Also define $V(\cdot)$ as the variance function of the model family, and w as the weight. The Pearson residual is then

$$r_P = (y - \hat{\mu})/\sqrt{V(\hat{\mu})/w}.$$

Furthermore, let $sgn(\cdot)$ be the function which returns the sign of its argument and let $d(y, \hat{\mu})$ be the model deviance. The deviance residual is then

$$r_D = sgn(y - \hat{\mu})\sqrt{wd(y, \hat{\mu})}.$$

Usage

```
## S3 method for class 'galamm'
residuals(object, type = c("pearson", "deviance"), scaled = FALSE, ...)
```

Arguments

object	An object of class galamm returned from galamm.
type	Character of length one describing the type of residuals to be returned. One of "pearson" and "deviance". Argument is case sensitive.
scaled	Logical value specifying whether to scale the residuals by their standard deviation. Defaults to FALSE.
...	Optional arguments passed on to other methods. Currently not used.

Value

Numeric vector of residual values.

References

Dunn PK, Smyth GK (2018). *Generalized Linear Models With Examples in R*, Springer Texts in Statistics. Springer, New York, NY. ISBN 978-1-4419-0117-0 978-1-4419-0118-7. doi:10.1007/9781441901187.

See Also

`fitted.galamm()` for model fitted values, `predict.galamm()` for model predictions, and `plot.galamm()` for diagnostic plots. The generic function is `residuals()`.

Other details of model fit: `VarCorr`, `appraise.galamm()`, `coef.galamm()`, `confint.galamm()`, `derivatives.galamm()`, `deviance.galamm()`, `factor_loadings.galamm()`, `family.galamm()`, `fitted.galamm()`, `fixef`, `formula.galamm()`, `llikAIC()`, `logLik.galamm()`, `model.frame.galamm()`, `nobs.galamm()`, `predict.galamm()`, `print.VarCorr.galamm()`, `ranef.galamm()`, `response()`, `sigma.galamm()`, `vcov.galamm()`

Examples

```
# Poisson GLMM
count_mod <- galamm(
  formula = y ~ lbas * treat + lage + v4 + (1 | subj),
  data = epilep, family = poisson
)

# Extract residuals
residuals(count_mod)
```

---

response *Extract response values*

Description

Extracts response values from a model.

Usage

```
response(object, ...)
```

Arguments

`object`	An object of class `galamm` returned from `galamm`.
`...`	Optional arguments passed on to other methods. Currently not used.

Value

A numerical vector with fit response values for each row in the input data.

See Also

Other details of model fit: VarCorr, appraise.galamm(), coef.galamm(), confint.galamm(), derivatives.galamm(), deviance.galamm(), factor_loadings.galamm(), family.galamm(), fitted.galamm(), fixef, formula.galamm(), llikAIC(), logLik.galamm(), model.frame.galamm(), nobs.galamm(), predict.galamm(), print.VarCorr.galamm(), ranef.galamm(), residuals.galamm(), sigma.galamm(), vcov.galamm()

Examples

```
# Linear mixed model with heteroscedastic residuals
mod <- galamm(
  formula = y ~ x + (1 | id),
  dispformula = ~ (1 | item),
  data = hsced
)

# Plot response versus fitted values
plot(fitted(mod), response(mod))
```

---

s *Set up smooth term with factor loading*

Description

This is a very thin wrapper around mgcv::s. It enables the specification of loading variables for smooth terms. The last letter "l", which stands for "loading", has been added to avoid namespace conflicts with mgcv and gamm4.

Usage

```
sl(..., factor = NULL)
```

Arguments

...	Arguments passed on to mgcv::s.
factor	Optional character argument specifying the loading variable. Case sensitive.

Details

The documentation of the function mgcv::s should be consulted for details on how to properly set up smooth terms. In particular, note that these terms distinguish between ordered and unordered factor terms in the by variable, which can be provided in ... and is forwarded to mgcv::s.

Value

An object of class `xx.smooth.spec`, where `xx` is a basis identifying code given by the `bs` argument of `s`. It differs from the smooth returned by `mgcv::s` in that it has an additional attribute named `"factor"` which specifies any factor loading which this smooth term should be multiplied with in order to produce the observed outcome.

References

Wood SN (2003). "Thin Plate Regression Splines." *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, **65**(1), 95–114. ISSN 1369-7412.

Wood SN (2017). *Generalized Additive Models: An Introduction with R*, 2 edition. Chapman and Hall/CRC.

See Also

Other modeling functions: `galamm()`, `galammObject`, `gfam()`, `t2`

Examples

```
# Linear mixed model with factor structures
dat <- subset(cognition, domain == 1 & timepoint == 1)
loading_matrix <- matrix(c(1, NA, NA), ncol = 1)


# Model with four thin-plate regression splines as basis functions
mod <- galamm(
  formula = y ~ 0 + item + sl(x, k = 4, factor = "loading"),
  data = dat,
  load_var = "item",
  lambda = loading_matrix,
  factor = "loading"
)

# Model with four cubic regression splines as basis functions
mod <- galamm(
  formula = y ~ 0 + item +
    sl(x, bs = "cr", k = 4, factor = "loading"),
  data = dat,
  load_var = "item",
  lambda = loading_matrix,
  factor = "loading"
)
```

`sigma.galamm` *Extract square root of dispersion parameter from galamm object*

Description

Extracts the square root of the dispersion parameter(s) from an object of class `galamm`, returned from `galamm`. In the case of conditionally Gaussian responses, this is the residual standard deviation. When there are multiple dispersion parameters, e.g., with mixed response type models, the square root of all of them are returned in a numeric vector.

Usage

```
## S3 method for class 'galamm'
sigma(object, ...)
```

Arguments

`object`	An object of class `galamm`, returned from `galamm`.
`...`	Optional parameters passed on to other methods. Currently not used.

Value

The square root of one or more dispersion parameters.

See Also

`galamm()`

Other details of model fit: `VarCorr`, `appraise.galamm()`, `coef.galamm()`, `confint.galamm()`, `derivatives.galamm()`, `deviance.galamm()`, `factor_loadings.galamm()`, `family.galamm()`, `fitted.galamm()`, `fixef`, `formula.galamm()`, `llikAIC()`, `logLik.galamm()`, `model.frame.galamm()`, `nobs.galamm()`, `predict.galamm()`, `print.VarCorr.galamm()`, `ranef.galamm()`, `residuals.galamm()`, `response()`, `vcov.galamm()`

Examples

```
# Linear mixed model with heteroscedastic residuals
mod <- galamm(
  formula = y ~ x + (1 | id),
  dispformula = ~ (1 | item),
  data = hsced
)

# Extract residual standard deviation.
sigma(mod)

# The residual standard deviation applies to the base case. The variance
# function shown in the model output shows the estimated multiplier for
# various grouping levels:
summary(mod)
```

summary.galamm *Summarizing GALAMM fits*

Description

Summary method for class "galamm".

Usage

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

Arguments

object	An object of class galamm returned from galamm.
...	Further arguments passed on to other methods. Currently not used.

Value

A list of summary statistics of the fitted model of class summary.galamm, containing the following elements:

- AICtab a table of model fit measures, returned by llikAIC.
- call the matched call used when fitting the model.
- fixef a matrix with fixed effect estimated, returned by fixef.
- gam List containing information about smooth terms in the model. If no smooth terms are contained in the model, then it is a list of length zero.
- model a list with various elements related to the model setup and fit. See ?galamm for details.
- parameters A list object with model parameters and related information. See ?galamm for details.
- Lambda An object containing the estimated factor loadings. Returned from factor_loadings.galamm. If there are no estimated factor loadings, then this object is NULL.
- random_effects a list containing the random effects. See ?galamm for details.
- VarCorr An object of class VarCorr.galamm, returned from VarCorr.galamm.
- weights An object containing information about estimated variance functions, when there are heteroscedastic residuals. Otherwise the object is NULL.

Author(s)

Some of the code for producing summary information has been derived from the summary methods of mgcv (author: Simon Wood) and lme4 (Bates et al. 2015) (authors: Douglas M. Bates, Martin Maechler, Ben Bolker, and Steve Walker).

See Also

print.summary.galamm() for the print method and summary() for the generic.

Other summary functions: anova.galamm(), draw.galamm(), plot_smooth.galamm(), print.galamm(), print.summary.galamm()

Examples

```
# Linear mixed model with heteroscedastic residuals
mod <- galamm(
  formula = y ~ x + (1 | id),
  dispformula = ~ (1 | item),
  data = hsced
)

summary(mod)
```

---

t2 *Set up smooth term with factor loading*

Description

This is a very thin wrapper around mgcv::t2. It enables the specification of loading variables for smooth terms. The last letter "l", which stands for "loading", has been added to avoid namespace conflicts with mgcv and gamm4.

Usage

```
t2l(..., factor = NULL)
```

Arguments

...	Arguments passed on to mgcv::t2.
factor	Optional character of length one specifying the loading variable. Case sensitive.

Details

The documentation of the function mgcv::t2 should be consulted for details on how to properly set up smooth terms. In particular, note that these terms distinguish between ordered and unordered factor terms in the by variable, which can be provided in ... and is forwarded to mgcv::t2.

Value

An object of class xx.smooth.spec, where xx is a basis identifying code given by the bs argument of t2. It differs from the smooth returned by mgcv::s in that it has an additional attribute named "factor" which specifies any factor loading which this smooth term should be multiplied with in order to produce the observed outcome.

References

Wood SN (2003). "Thin Plate Regression Splines." *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, **65**(1), 95–114. ISSN 1369-7412.

Wood SN (2017). *Generalized Additive Models: An Introduction with R*, 2 edition. Chapman and Hall/CRC.

See Also

Other modeling functions: `galamm()`, `galammObject`, `gfam()`, `s`

Examples

```
# Linear mixed model with factor structures
dat <- subset(cognition, domain == 1 & timepoint == 1)
loading_matrix <- matrix(c(1, NA, NA), ncol = 1)

# Model with four cubic regression splines as basis functions
mod <- galamm(
  formula = y ~ 0 + item + t2l(x, k = 4, factor = "loading"),
  data = dat,
  load_var = "item",
  lambda = loading_matrix,
  factor = "loading"
)

# Model with four thin-plate regression splines as basis functions
mod <- galamm(
  formula = y ~ 0 + item +
    t2l(x, bs = "tp", k = 4, factor = "loading"),
  data = dat,
  load_var = "item",
  lambda = loading_matrix,
  factor = "loading"
)
```

`VarCorr` *Extract variance and correlation components from model*

Description

Extract variance and correlation components from model

Usage

```
## S3 method for class 'galamm'
VarCorr(x, sigma = 1, ...)
```

Arguments

x	An object of class galamm returned from galamm.
sigma	Numeric value used to multiply the standard deviations. Defaults to 1.
...	Other arguments passed onto other methods. Currently not used.

Value

An object of class c("VarCorr.galamm", "VarCorr.merMod").

See Also

print.VarCorr.galamm() for the print function.

Other details of model fit: appraise.galamm(), coef.galamm(), confint.galamm(), derivatives.galamm(), deviance.galamm(), factor_loadings.galamm(), family.galamm(), fitted.galamm(), fixef, formula.galamm(), llikAIC(), logLik.galamm(), model.frame.galamm(), nobs.galamm(), predict.galamm(), print.VarCorr.galamm(), ranef.galamm(), residuals.galamm(), response(), sigma.galamm(), vcov.galamm()

Examples

```
# Linear mixed model with heteroscedastic residuals
mod <- galamm(
  formula = y ~ x + (1 | id),
  dispformula = ~ (1 | item),
  data = hsced
)

# Extract information on variance and covariance
VarCorr(mod)

# Convert to data frame
# (this invokes lme4's function as.data.frame.VarCorr.merMod)
as.data.frame(VarCorr(mod))
```

---

vcov.galamm *Calculate variance-covariance matrix for GALAMM fit*

Description

Calculate variance-covariance matrix for GALAMM fit

Usage

```
## S3 method for class 'galamm'
vcov(object, parm = "beta", ...)
```

Arguments

object	Object of class galamm returned from galamm.
parm	The parameters for which the variance-covariance matrix should be calculated. Character vector with one or more of the elements "theta", "beta", "lambda", and "weights". Can also be an integer vector. When given as a character, it must be in only lowercase letters.
...	Further arguments passed on to other methods. Currently not used.

Value

A variance-covariance matrix.

See Also

confint.galamm() for the method computing confidence intervals. See vcov() for the generic function.

Other details of model fit: VarCorr, appraise.galamm(), coef.galamm(), confint.galamm(), derivatives.galamm(), deviance.galamm(), factor_loadings.galamm(), family.galamm(), fitted.galamm(), fixef, formula.galamm(), llikAIC(), logLik.galamm(), model.frame.galamm(), nobs.galamm(), predict.galamm(), print.VarCorr.galamm(), ranef.galamm(), residuals.galamm(), response(), sigma.galamm()

Examples

```
# Linear mixed model with heteroscedastic residuals
mod <- galamm(
  formula = y ~ x + (1 | id),
  dispformula = ~ (1 | item),
  data = hsced
)

# Extract covariance matrix for fixed regression coefficients
vcov(mod, parm = "beta")

# and then for weights, which gives us the variance.
vcov(mod, parm = "weights")
```

Index

∗ **datasets**
cognition, 6
diet, 10
epilep, 12
hsced, 27
latent_covariates, 28
latent_covariates_long, 28
lifespan, 29
mresp, 32
mresp_hsced, 32
∗ **details of model fit**
appraise.galamm, 4
coef.galamm, 5
confint.galamm, 7
derivatives.galamm, 8
deviance.galamm, 9
factor_loadings.galamm, 14
family.galamm, 15
fitted.galamm, 16
fixef, 17
formula.galamm, 18
logLik.galamm, 30
model.frame.galamm, 31
nobs.galamm, 33
predict.galamm, 37
print.VarCorr.galamm, 41
ranef.galamm, 43
residuals.galamm, 44
response, 45
sigma.galamm, 47
VarCorr, 51
vcov.galamm, 52
∗ **diagnostics**
plot.galamm, 34
qqmath.galamm, 42
∗ **modeling functions**
galamm, 19
galammObject, 24
gfam, 26
s, 46
t2, 50
∗ **optimization functions**
extract_optim_parameters.galamm, 13
galamm_control, 25
∗ **summary functions**
anova.galamm, 3
draw.galamm, 11
plot_smooth.galamm, 36
print.galamm, 39
print.summary.galamm, 40
summary.galamm, 49

anova(), *4*
anova.galamm, 3
anova.galamm(), *11*, *36*, *39, 40*, *50*
appraise *(*appraise.galamm*)*, 4
appraise.galamm, 4
appraise.galamm(), *5*, *7–9*, *14*, *16–18*, *30, 31*, *33*, *38*, *42*, *44–46*, *48*, *52, 53*

coef(), *5*
coef.galamm, 5
coef.galamm(), *5*, *7–9*, *14*, *16–18*, *30, 31*, *33*, *38*, *42*, *44–46*, *48*, *52, 53*
cognition, 6, *11, 12*, *27–30*, *32, 33*
confint(), *7*
confint.galamm, 7
confint.galamm(), *5*, *8, 9*, *14*, *16–18*, *30, 31*, *33*, *38*, *42*, *44–46*, *48*, *52, 53*

derivatives *(*derivatives.galamm*)*, 8
derivatives.galamm, 8
derivatives.galamm(), *5*, *7*, *9*, *14*, *16–18*, *30, 31*, *33*, *38*, *42*, *44–46*, *48*, *52, 53*
deviance(), *9*
deviance.galamm, 9
deviance.galamm(), *5*, *7, 8*, *14*, *16–18*, *30, 31*, *33*, *38*, *42*, *44–46*, *48*, *52, 53*

diet, *6*, 10, *12*, *27–30*, *32, 33*
draw *(*draw.galamm*)*, 11
draw.galamm, 11
draw.galamm(), *4*, *36*, *39, 40*, *50*

epilep, *6*, *11*, 12, *27–30*, *32, 33*
extract_optim_parameters *(*extract_optim_parameters.galamm*)*, 13
extract_optim_parameters.galamm, 13
extract_optim_parameters.galamm(), *26*

factor_loadings *(*factor_loadings.galamm*)*, 14
factor_loadings.galamm, 14, *49*
factor_loadings.galamm(), *5*, *7–9*, *16–18*, *30, 31*, *33*, *38*, *42*, *44–46*, *48*, *52, 53*
family.galamm, 15
family.galamm(), *5*, *7–9*, *14*, *17, 18*, *30, 31*, *33*, *38*, *42*, *44–46*, *48*, *52, 53*
fitted.galamm, 16, *37*
fitted.galamm(), *5*, *7–9*, *14*, *16*, *18*, *30, 31*, *33*, *38*, *42*, *44–46*, *48*, *52, 53*
fixef, *5*, *7–9*, *14*, *16, 17*, 17, *18*, *30, 31*, *33*, *38*, *42*, *44–46*, *48, 49*, *52, 53*
fixef.galamm(), *5*, *7*, *14*, *44*
formula.galamm, 18
formula.galamm(), *5*, *7–9*, *14*, *16–18*, *30, 31*, *33*, *38*, *42*, *44–46*, *48*, *52, 53*

galamm, *3–5*, *7–9*, *11*, *13–18*, 19, *25, 26*, *33*, *34*, *36*, *38, 39*, *42–45*, *48, 49*, *52, 53*
galamm(), *16*, *24–27*, *34*, *44*, *47, 48*, *51*
galamm_control, *21*, 25
galamm_control(), *13*
galammObject, *21, 22*, 24, *27*, *31*, *47*, *51*
gfam, 26
gfam(), *20*, *22*, *25*, *47*, *51*
gratia::appraise(), *4*
gratia::derivatives(), *8*
gratia::draw(), *11*

hsced, *6*, *11, 12*, 27, *28–30*, *32, 33*

latent_covariates, *6*, *11, 12*, *27*, 28, *29, 30*, *32, 33*
latent_covariates_long, *6*, *11, 12*, *27, 28*, 28, *30*, *32, 33*
lifespan, *6*, *11, 12*, *27–29*, 29, *32, 33*
llikAIC, *49*
llikAIC(), *5*, *7–9*, *14*, *16–18*, *30, 31*, *33*, *38*, *42*, *44–46*, *48*, *52, 53*
lme4::lFormula(), *24*
logLik(), *30*
logLik.galamm, 30
logLik.galamm(), *5*, *7–9*, *14*, *16–18*, *31*, *33*, *38*, *42*, *44–46*, *48*, *52, 53*

model.frame.galamm, 31
model.frame.galamm(), *5*, *7–9*, *14*, *16–18*, *30*, *33*, *38*, *42*, *44–46*, *48*, *52, 53*
mresp, *6*, *11, 12*, *27–30*, 32, *33*
mresp_hsced, *6*, *11, 12*, *27–30*, *32*, 32

nobs.galamm, 33
nobs.galamm(), *5*, *7–9*, *14*, *16–18*, *30, 31*, *38*, *42*, *44–46*, *48*, *52, 53*

plot(), *35*
plot.galamm, 34
plot.galamm(), *43*, *45*
plot_smooth *(*plot_smooth.galamm*)*, 36
plot_smooth.galamm, 36
plot_smooth.galamm(), *4*, *11*, *39, 40*, *50*
predict(), *38*
predict.galamm, 37
predict.galamm(), *5*, *7–9*, *14*, *16–18*, *30, 31*, *33*, *42*, *44–46*, *48*, *52, 53*
print(), *39, 40*
print.galamm, 39
print.galamm(), *4*, *11*, *36*, *40*, *50*
print.summary.galamm, 40
print.summary.galamm(), *4*, *11*, *36*, *39*, *50*
print.VarCorr.galamm, 41
print.VarCorr.galamm(), *5*, *7–9*, *14*, *16–18*, *30, 31*, *33*, *38*, *44–46*, *48*, *52, 53*

qqmath *(*qqmath.galamm*)*, 42
qqmath.galamm, 42
qqmath.galamm(), *35*

ranef *(*ranef.galamm*)*, 43
ranef.galamm, 43
ranef.galamm(), *5*, *7–9*, *14*, *16–18*, *30, 31*, *33*, *38*, *42*, *45, 46*, *48*, *52, 53*
residuals(), *45*
residuals.galamm, 44
residuals.galamm(), *5*, *7–9*, *14*, *16–18*, *30*, *31*, *33–35*, *38*, *42*, *44*, *46*, *48*, *52, 53*

response, 45
response(), *5*, *7–9*, *14*, *16–18*, *30, 31*, *33*, *38*, *42*, *44, 45*, *48*, *52, 53*

s, *22*, *25*, *27*, 46, *51*
sigma.galamm, 47
sigma.galamm(), *5*, *7–9*, *14*, *16–18*, *30, 31*, *33*, *38*, *42*, *44–46*, *52, 53*
sl *(*s*)*, 46
summary(), *50*
summary.galamm, *40*, 49
summary.galamm(), *4*, *11*, *36*, *39, 40*

t2, *22*, *25*, *27*, *47*, 50
t2l *(*t2*)*, 50

VarCorr, *5*, *7–9*, *14*, *16–18*, *30, 31*, *33*, *38*, *42*, *44–46*, *48*, 51, *53*
VarCorr.galamm, *41*, *49*
VarCorr.galamm(), *42*
vcov(), *53*
vcov.galamm, 52
vcov.galamm(), *5*, *7–9*, *14*, *16–18*, *30, 31*, *33*, *38*, *42*, *44–46*, *48*, *52*