Package: inlatools (via r-universe)

November 3, 2024

Title Diagnostic Tools for INLA Models

Version 0.0.3

Description Several functions which can be useful to choose sensible priors and diagnose the fitted model.

License GPL-3

URL https://inbo.github.io/inlatools/, https://github.com/inbo/inlatools

BugReports https://github.com/inbo/inlatools/issues

Depends R (>= 3.5.0)

Imports assertthat, dplyr, ggplot2, INLA (>= 19.09.03), methods, purrr, rlang, scales, tibble, tidyr

Suggests knitr, rmarkdown, sn, testthat

VignetteBuilder knitr

Additional_repositories https://inla.r-inla-download.org/R/stable

Config/checklist/communities inbo

Config/checklist/keywords INLA; dispersion; distribution; prior; anomaly; residuals

Encoding UTF-8

Language en-GB

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Repository https://inbo.r-universe.dev

RemoteUrl https://github.com/inbo/inlatools

RemoteRef HEAD

RemoteSha e711776e8325734557593937c3ef439794e04b7c

Contents

dgpoisson 2
dispersion 3
dispersion_check 4
distribution_check 5
fast_aggregation_check 6
fast_distribution_check 7
fitted,inla-method 8
generate_data 8
get_anomaly 10
get_observed 11
plot.dispersion_check 11
plot.distribution_check 12
plot.sim_iid 13
plot.sim_rw 14
prec2sd 16
residuals,inla-method 16
rzanbinom 17
rzapois 18
rzinbinom 18
rzipois 19
select_change 19
select_divergence 20
select_poly 20
select_quantile 21
simulate_iid 21
simulate_rw 22
var_nbinom 23

Index **24**

`dgpoisson` *The generalised Poisson distribution*

Description

The generalised Poisson distribution

Usage

```
dgpoisson(y, mu, phi)

rgpoisson(n, mu, phi)
```

Arguments

`y`	a vector of positive integers for which to calculate the density
`mu`	a vector of averages for which to calculate the density
`phi`	a single overdispersion parameter
`n`	the number of simulated values

Value

a matrix with the density for each combination of `y` (rows) and `mu` (cols)

See Also

Other statistics: `dispersion()`, `fitted,inla-method`, `get_observed()`, `residuals,inla-method`

Other statistics: `dispersion()`, `fitted,inla-method`, `get_observed()`, `residuals,inla-method`

`dispersion` *Calculate a measure for dispersion*

Description

The measure is calculated as the average of the squared Pearson residuals

Usage

```
dispersion(observed, fitted, variance)
```

Arguments

`observed`	the observed values
`fitted`	the fitted values
`variance`	the variance of the fitted values

See Also

Other statistics: `dgpoisson()`, `fitted,inla-method`, `get_observed()`, `residuals,inla-method`

Examples

```
library(INLA)
set.seed(20181202)
model <- inla(
  poisson ~ 1,
  family = "poisson",
  data = data.frame(
    poisson = rpois(20, lambda = 10),
    base = 1
  ),
  control.predictor = list(compute = TRUE)
)
dispersion(
  observed = get_observed(model),
  fitted = fitted(model),
  variance = fitted(model)
)
```

dispersion_check *Use simulations to check for overdispersion or underdispersion*

Description

Use simulations to check for overdispersion or underdispersion

Usage

```
dispersion_check(object, nsim = 1000)

## S4 method for signature 'inla'
dispersion_check(object, nsim = 1000)
```

Arguments

`object`	the INLA model
`nsim`	the number of simulation

See Also

Other checks: `distribution_check()`, `fast_aggregation_check()`, `fast_distribution_check()`, `get_anomaly()`

Examples

```
library(INLA)
set.seed(20181202)
model <- inla(
  poisson ~ 1,
  family = "poisson",
  data = data.frame(
    poisson = rpois(20, lambda = 10),
    base = 1
  ),
  control.predictor = list(compute = TRUE)
)
dc <- dispersion_check(model)
str(dc)
```

distribution_check *Use simulations to compare the observed distribution with the modelled distribution*

Description

Use simulations to compare the observed distribution with the modelled distribution

Usage

```
distribution_check(object, nsim = 1000, seed = 0L)

## S4 method for signature 'inla'
distribution_check(object, nsim = 1000, seed = 0L)
```

Arguments

object	the INLA model
nsim	the number of simulation
seed	See the same argument in ?inla.qsample for further information. In order to produce reproducible results, you ALSO need to make sure the RNG in R is in the same state, see example below. When seed is non-zero, num.threads is forced to "1:1" and parallel.configs is set to FALSE, since parallel sampling would not produce a reproducible sequence of pseudo-random numbers.

See Also

Other checks: dispersion_check(), fast_aggregation_check(), fast_distribution_check(), get_anomaly()

Examples

```
library(INLA)
model <- inla(
  poisson ~ 1,
  family = "poisson",
  data = data.frame(
    poisson = rpois(20, lambda = 10),
    base = 1
  ),
  control.predictor = list(compute = TRUE),
  control.compute = list(config = TRUE)
)
distribution_check(model, seed = 20181202)
```

fast_aggregation_check *Fast aggregation check Compare the observed and modelled aggregated values.*

Description

Fast aggregation check Compare the observed and modelled aggregated values.

Usage

```
fast_aggregation_check(
  object,
  grouping_vars,
  fun = sum,
  remove_na = TRUE,
  nsim = 1000
)

## S4 method for signature 'inla'
fast_aggregation_check(
  object,
  grouping_vars,
  fun = sum,
  remove_na = TRUE,
  nsim = 1000
)
```

Arguments

object	the INLA model
grouping_vars	character vector of variable names to group by.
fun	function to apply to the aggregated values.
remove_na	logical. Indicated whether to remove observations where the response variable is a missing values.
nsim	the number of simulation

See Also

Other checks: dispersion_check(), distribution_check(), fast_distribution_check(), get_anomaly()

fast_distribution_check

Use simulations to compare the observed distribution with the modelled distribution

Description

This check uses the fitted values and thus ignores the uncertainty on the predictions

Usage

```
fast_distribution_check(object, nsim = 1000)

## S4 method for signature 'inla'
fast_distribution_check(object, nsim = 1000)

## S4 method for signature 'list'
fast_distribution_check(object, nsim = 1000)
```

Arguments

object	the INLA model
nsim	the number of simulation

See Also

Other checks: dispersion_check(), distribution_check(), fast_aggregation_check(), get_anomaly()

Examples

```
library(INLA)
set.seed(20181202)
model <- inla(
  poisson ~ 1,
  family = "poisson",
  data = data.frame(
    poisson = rpois(20, lambda = 10),
    base = 1
  ),
  control.predictor = list(compute = TRUE)
)
fast_distribution_check(model)
```

fitted,inla-method *Extract the fitted values from an INLA model*

Description

Extract the fitted values from an INLA model

Usage

```
## S4 method for signature 'inla'
fitted(object, ...)
```

Arguments

object	an object for which the extraction of model fitted values is meaningful.
...	other arguments.

See Also

Other statistics: dgpoisson(), dispersion(), get_observed(), residuals,inla-method

Examples

```
library(INLA)
set.seed(20181202)
model <- inla(#'   poisson ~ 1,
  poisson ~ 1,
  family = "poisson",
  data = data.frame(
    poisson = rpois(20, lambda = 10),
    base = 1
  ),
  control.predictor = list(compute = TRUE)
)
fitted(model)
```

generate_data *Generate dummy data with several distributions*

Description

All distributions share the same latent variable $\eta_{ij} = a + b_i$ with $b_i = N(0, \sigma_r)$.

Usage

```
generate_data(
  a = 0,
  sigma_random = 0.5,
  n_random = 20,
  n_replicate = 10,
  nb_size = 1,
  b_size = 5,
  zero_inflation = 0.5
)
```

Arguments

`a`	the intercept of the latent variable
`sigma_random`	The standard error for the random effect σ_r.
`n_random`	the number of random effect levels (groups).
`n_replicate`	the number of observation per random effect level.
`nb_size`	the size parameter of the negative binomial distribution. Passed to the `size` parameter of `stats::rnbinom()`.
`b_size`	the size parameter of the binomial distribution. Passed to the `size` parameter of `stats::rbinom()`.
`zero_inflation`	the probability the the observed value stems for the a point mass in zero.

Details

- The Poisson distribution uses $\lambda = e^{\eta_{ij}}$.
- The negation binomial distribution uses $\mu = e^{\eta_{ij}}$.
- The binomial distribution uses $\pi_{ij} = e^{\eta_{ij}}/(e^{\eta_{ij}} + 1)$.

Value

A `data.frame`

- `id` the id of the random effect.
- `eta` the latent variable.
- `zero_inflation` use the point mass in zero.
- `poisson` the Poisson distributed variable.
- `zipoisson` the zero-inflated Poisson distributed variable.
- `negbin` the negative binomial distributed variable.
- `zinegbin` the zero-inflated negative binomial distributed variable.
- `binom` the binomial distributed variable.

See Also

Other utils: `plot.dispersion_check()`, `plot.distribution_check()`

Examples

```
set.seed(20181202)
head(generate_data())
```

---

get_anomaly *Get a set of anomalies of the model*

Description

Returns a named list with an element called observations and one element for every random effect. The random effect components use the name of the random effect.

Usage

```
get_anomaly(object, n = 10)

## S4 method for signature 'inla'
get_anomaly(object, n = 20)
```

Arguments

object	the INLA model
n	the number of anomalies per criterion. Defaults to 10.

Details

observations is a subset of the original data.frame. It contains the rows with the n largest and n smallest values of the Pearson residuals. The random effect components contain a subset of the random effects. Here we select the rows with the n largest and n lowest values of the mean.

See Also

Other checks: dispersion_check(), distribution_check(), fast_aggregation_check(), fast_distribution_check

Examples

```
library(INLA)
set.seed(20181202)
model <- inla(
  poisson ~ 1,
  family = "poisson",
  data = data.frame(
    poisson = rpois(20, lambda = 10),
    base = 1
  ),
  control.predictor = list(compute = TRUE)
)
dc <- get_anomaly(model, n = 2)
str(dc)
```

get_observed *get the observed values from the model object*

Description

get the observed values from the model object

Usage

```
get_observed(object)
```

Arguments

object	the INLA model

See Also

Other statistics: dgpoisson(), dispersion(), fitted,inla-method, residuals,inla-method

Examples

```
library(INLA)
set.seed(20181202)
model <- inla(
  poisson ~ 1,
  family = "poisson",
  data = data.frame(
    poisson = rpois(20, lambda = 10),
    base = 1
  ),
  control.predictor = list(compute = TRUE)
)
get_observed(model)
```

plot.dispersion_check *Plot the results from a dispersion check*

Description

Plot the results from a dispersion check

Usage

```
## S3 method for class 'dispersion_check'
plot(x, y, ...)
```

Arguments

x	a dispersion_check object. Which is the output of \link{dispersion_check}
y	currently ignored
...	currently ignored

Value

A \link[ggplot2]{ggplot} object.

See Also

Other utils: generate_data(), plot.distribution_check()

Examples

```
library(INLA)
set.seed(20181202)
model <- inla(
  poisson ~ 1,
  family = "poisson",
  data = data.frame(
    poisson = rpois(20, lambda = 10),
    base = 1
  ),
  control.predictor = list(compute = TRUE)
)
dc <- dispersion_check(model)
plot(dc)
```

plot.distribution_check

Plot the results from a distribution check

Description

Plot the results from a distribution check

Usage

```
## S3 method for class 'distribution_check'
plot(x, y, ..., n = FALSE, scales = "fixed")
```

Arguments

x	a distribution_check object. Which is the output of \link{fast_distribution_check}
y	currently ignored
...	currently ignored
n	display the number of observations
scales	Should scales be fixed ("fixed", the default), free ("free"), or free in one dimension ("free_x", "free_y")?

Value

A \link[ggplot2]{ggplot} object.

See Also

Other utils: generate_data(), plot.dispersion_check()

Examples

```
library(INLA)
set.seed(20181202)
model <- inla(
  poisson ~ 1,
  family = "poisson",
  data = data.frame(
    poisson = rpois(20, lambda = 10),
    base = 1
  ),
  control.predictor = list(compute = TRUE)
)
fdc <- fast_distribution_check(model)
plot(fdc)
```

plot.sim_iid *Plot Simulated Random Intercepts*

Description

Plot Simulated Random Intercepts

Usage

```
## S3 method for class 'sim_iid'
plot(
  x,
  y,
  ...,
  link = c("identity", "log", "logit"),
  baseline,
  center = c("mean", "bottom", "top"),
  quantiles = c(0.025, 0.1, 0.5, 0.9, 0.975)
)
```

Arguments

x	A sim_iid object. Which is the output of \link{simulate_iid}.
y	Currently ignored.
...	Currently ignored.
link	Which link to use for back transformation.
baseline	Optional baseline for the time series.
center	Defines how to centre the time series to the baseline. Options are: start all time series start at the baseline; mean the average of the time series is the baseline; bottom the lowest value of the time series equals the baseline; top the highest value of the time series equals the baseline.
quantiles	Which quantiles are shown on the plot.

Value

A \link[ggplot2]{ggplot} object.

See Also

Other priors: plot.sim_rw(), select_change(), select_divergence(), select_poly(), select_quantile(), simulate_iid(), simulate_rw()

Examples

```
set.seed(20181202)
x <- simulate_iid(sigma = 0.25)
plot(x)
plot(x, link = "log")
plot(x, link = "logit")
```

---

plot.sim_rw *Plot Simulated Random Walks*

Description

Plot Simulated Random Walks

Usage

```
## S3 method for class 'sim_rw'
plot(
  x,
  y,
  ...,
  link = c("identity", "log", "logit"),
  baseline,
  center = c("start", "mean", "bottom", "top")
)
```

Arguments

`x`	An `sim_rw` object. Which is the output of `\link{simulate_rw}`
`y`	Currently ignored.
`...`	Currently ignored.
`link`	Which link to use for back transformation.
`baseline`	Optional baseline for the time series.
`center`	Defines how to centre the time series to the baseline. Options are: `start` all time series start at the baseline; `mean` the average of the time series is the baseline; `bottom` the lowest value of the time series equals the baseline; `top` the highest value of the time series equals the baseline.

Value

A `\link[ggplot2]{ggplot}` object.

See Also

Other priors: `plot.sim_iid()`, `select_change()`, `select_divergence()`, `select_poly()`, `select_quantile()`, `simulate_iid()`, `simulate_rw()`

Examples

```
set.seed(20181202)
x <- simulate_rw(sigma = 0.05, start = -10, length = 40)
plot(x)
plot(select_quantile(x))
plot(select_quantile(x), link = "log")
plot(select_quantile(x), link = "logit")
x <- simulate_rw(sigma = 0.001, start = -10, length = 40, order = 2)
plot(x)
plot(select_quantile(x))
plot(select_quantile(x), link = "log")
plot(select_quantile(x), link = "logit")
```

prec2sd *Convert the posterior marginal of a precision to a standard deviation*

Description

Convert the posterior marginal of a precision to a standard deviation

Usage

```
prec2sd(marg)
```

Arguments

marg	A matrix with columns "y" and "x" where "y" is the marginal of the precision.

Value

A data.frame with the mean, standard deviation and 2.5%, 25%, 50%, 75% and 97.5% quantiles of the posterior of the standard deviation.

Examples

```
stopifnot(require(INLA))
model <- inla(Sepal.Length ~ Species, data = iris, family = "gaussian")
marg <- model$marginals.hyperpar[["Precision for the Gaussian observations"]]
prec2sd(marg)
```

residuals,inla-method *Calculate the Residuals From an INLA Model*

Description

Calculate the Residuals From an INLA Model

Usage

```
## S4 method for signature 'inla'
residuals(
  object,
  type = c("pearson", "deviance", "working", "response", "partial"),
  ...
)
```

Arguments

object An inla object.

type Currently only Pearson residuals are available. Other types are only listed for compatibility with the default residuals function.

... Currently ignored.

See Also

Other statistics: dgpoisson(), dispersion(), fitted,inla-method, get_observed()

Examples

```
library(INLA)
set.seed(20181202)
model <- inla(
  poisson ~ 1,
  family = "poisson",
  data = data.frame(
    poisson = rpois(20, lambda = 10),
    base = 1
  ),
  control.predictor = list(compute = TRUE)
)
residuals(model)
```

rzanbinom *Zero altered negative binomial*

Description

Zero altered negative binomial

Usage

```
rzanbinom(n, mu, size, prob, tol = 2e-10)
```

Arguments

n number of random values to return.

mu alternative parametrization via mean: see 'Details'.

size target for number of successful trials, or dispersion parameter (the shape parameter of the gamma mixing distribution). Must be strictly positive, need not be integer.

prob the point mass of zero

tol the tolerance for low numbers

`rzapois` *Zero-altered Poisson*

Description

Generate random numbers from a zero-altered Poisson distribution

Usage

```
rzapois(n, lambda, prob, tol = 2e-10)
```

Arguments

`n`	number of random values to return.
`lambda`	vector of (non-negative) means.
`prob`	the point mass of zero
`tol`	the tolerance for low numbers

`rzinbinom` *Zero inflated negative binomial*

Description

Zero inflated negative binomial

Usage

```
rzinbinom(n, mu, size, prob)
```

Arguments

`n`	number of random values to return.
`mu`	alternative parametrization via mean: see ‘Details’.
`size`	target for number of successful trials, or dispersion parameter (the shape parameter of the gamma mixing distribution). Must be strictly positive, need not be integer.
`prob`	the mass of extra zero’s

rzipois *Zero-inflated Poisson*

Description

Generate random numbers from a zero-inflated Poisson distribution

Usage

```
rzipois(n, lambda, prob)
```

Arguments

n	number of random values to return.
lambda	vector of (non-negative) means.
prob	the mass of extra zero's

select_change *Select Fast Changing Simulations from an* sim_rw *Object*

Description

This functions count the number of changes in direction in each simulation. It returns the subset with the highest number of direction changes

Usage

```
select_change(x, n = 10)
```

Arguments

x	An sim_rw object. Which is the output of \link{simulate_rw}
n	The number of simulations to plot when only a subset is shown.

See Also

Other priors: plot.sim_iid(), plot.sim_rw(), select_divergence(), select_poly(), select_quantile(), simulate_iid(), simulate_rw()

select_divergence *Select Diverging Simulations from an sim_rw Object*

Description

The selection will contain the most extreme simulations base on either the minimum effect or the maximum effect within the simulation.

Usage

```
select_divergence(x, n = 10)
```

Arguments

x	An sim_rw object. Which is the output of \link{simulate_rw}
n	The number of simulations to plot when only a subset is shown.

See Also

Other priors: plot.sim_iid(), plot.sim_rw(), select_change(), select_poly(), select_quantile(), simulate_iid(), simulate_rw()

select_poly *Select Random Walks Best Matching Some Polygon Coefficients*

Description

The target coefficients will be rescaled to have norm 1. The coefficients of the simulations will be rescaled by the largest norm over all simulations.

Usage

```
select_poly(x, coefs = c(0, -1), n = 10)
```

Arguments

x	An sim_rw object. Which is the output of \link{simulate_rw}
coefs	The polynomial coefficients.
n	The number of simulations to plot when only a subset is shown.

See Also

Other priors: plot.sim_iid(), plot.sim_rw(), select_change(), select_divergence(), select_quantile(), simulate_iid(), simulate_rw()

`select_quantile` *select the quantiles from an* `sim_rw` *object*

Description

select the quantiles from an `sim_rw` object

Usage

```
select_quantile(x, quantiles = c(0.025, 0.1, 0.25, 0.5, 0.75, 0.975))
```

Arguments

`x`	An `sim_rw` object. Which is the output of `\link{simulate_rw}`
`quantiles`	a vector of quantiles

See Also

Other priors: `plot.sim_iid()`, `plot.sim_rw()`, `select_change()`, `select_divergence()`, `select_poly()`, `simulate_iid()`, `simulate_rw()`

`simulate_iid` *simulate data from a second order random walk*

Description

simulate data from a second order random walk

Usage

```
simulate_iid(sigma = NULL, tau = NULL, n_sim = 1000)
```

Arguments

`sigma`	the standard deviation of the random intercept
`tau`	the precision of the random intercept
`n_sim`	the number of simulations

Value

a data.frame with simulated time series from the random walk

See Also

Other priors: `plot.sim_iid()`, `plot.sim_rw()`, `select_change()`, `select_divergence()`, `select_poly()`, `select_quantile()`, `simulate_rw()`

Examples

```
set.seed(20181202)
x <- simulate_iid(sigma = 0.25)
head(x)
```

---

simulate_rw *simulate data from a second order random walk*

Description

simulate data from a second order random walk

Usage

```
simulate_rw(
  sigma = NULL,
  tau = NULL,
  length = 10,
  start = 1,
  order = 1,
  n_sim = 1000
)
```

Arguments

sigma	the standard deviation of the random walk process
tau	the precision of the random walk process
length	the length of the time series
start	the starting values of the time series
order	1 for first order random walk or 2 for second order random walk
n_sim	the number of simulations

Value

a data.frame with simulated time series from the random walk

See Also

Other priors: plot.sim_iid(), plot.sim_rw(), select_change(), select_divergence(), select_poly(), select_quantile(), simulate_iid()

Examples

```
set.seed(20181202)
x <- simulate_rw(sigma = 0.1, start = -10, length = 40)
head(x)
y <- simulate_rw(sigma = 0.001, start = -10, length = 40, order = 2)
head(y)
```

`var_nbinom` *Variance of the negative binomial distribution*

Description

The type 1 zero-inflated negative binomial distribution is a standard negative binomial distribution with an additional point mass at zero.

The type 1 zero-inflated poisson is a standard Poisson distribution with an additional point mass at zero.

Usage

```
var_nbinom(mu, size)

var_zinbinom1(mu, size, zero)

var_zipois1(mu, zero)
```

Arguments

`mu`	mean of the distribution. Must be on the original scale.
`size`	Size of the negative binomial distribution. Must be strict positive.
`zero`	Probability of the point mass at zero.

Index

∗ **checks**
    dispersion_check, 4
    distribution_check, 5
    fast_aggregation_check, 6
    fast_distribution_check, 7
    get_anomaly, 10
∗ **priors**
    plot.sim_iid, 13
    plot.sim_rw, 14
    select_change, 19
    select_divergence, 20
    select_poly, 20
    select_quantile, 21
    simulate_iid, 21
    simulate_rw, 22
∗ **statistics**
    dgpoisson, 2
    dispersion, 3
    fitted,inla-method, 8
    get_observed, 11
    residuals,inla-method, 16
∗ **utils**
    generate_data, 8
    plot.dispersion_check, 11
    plot.distribution_check, 12
∗ **variance**
    var_nbinom, 23

dgpoisson, 2, *3*, *8*, *11*, *17*
dispersion, *3*, 3, *8*, *11*, *17*
dispersion_check, 4, *5–7*, *10*
dispersion_check,inla-method
        *(*dispersion_check*)*, 4
distribution_check, *4*, 5, *6*, *7*, *10*
distribution_check,inla-method
        *(*distribution_check*)*, 5

fast_aggregation_check, *4*, *5*, 6, *7*, *10*
fast_aggregation_check,inla-method
        *(*fast_aggregation_check*)*, 6
fast_distribution_check, *4–6*, 7, *10*
fast_distribution_check,inla-method
        *(*fast_distribution_check*)*, 7
fast_distribution_check,list-method
        *(*fast_distribution_check*)*, 7
fitted,inla-method, 8

generate_data, 8, *12*, *13*
get_anomaly, *4–7*, 10
get_anomaly,inla-method *(*get_anomaly*)*,
        10
get_observed, *3*, *8*, 11, *17*

plot.dispersion_check, *9*, 11, *13*
plot.distribution_check, *9*, *12*, 12
plot.sim_iid, 13, *15*, *19–22*
plot.sim_rw, *14*, 14, *19–22*
prec2sd, 16

residuals,inla-method, 16
rgpoisson *(*dgpoisson*)*, 2
rzanbinom, 17
rzapois, 18
rzinbinom, 18
rzipois, 19

select_change, *14*, *15*, 19, *20–22*
select_divergence, *14*, *15*, *19*, *20*, 20, *21*, *22*
select_poly, *14*, *15*, *19*, *20*, 20, *21*, *22*
select_quantile, *14*, *15*, *19–21*, 21, *22*
simulate_iid, *14*, *15*, *19–21*, 21, *22*
simulate_rw, *14*, *15*, *19–21*, 22

var_nbinom, 23
var_zinbinom1 *(*var_nbinom*)*, 23
var_zipois1 *(*var_nbinom*)*, 23