

Package: multimput (via r-universe)

October 12, 2026

Type Package

Title Using Multiple Imputation to Address Missing Data

Version 0.2.15

Description Accompanying package for the paper: Working with population totals in the presence of missing data comparing imputation methods in terms of bias and precision. Published in 2017 in the Journal of Ornithology volume 158 page 603–615 (<[doi:10.1007/s10336-016-1404-9](https://doi.org/10.1007/s10336-016-1404-9)>).

License GPL-3

URL <https://doi.org/10.5281/zenodo.598331>,
<https://github.com/inbo/multimput>,
<https://inbo.github.io/multimput/>

BugReports <https://github.com/inbo/multimput/issues>

Depends R (>= 3.0.0)

Imports assertthat, digest, dplyr, INLA (>= 22.01.19), lme4, methods, mvtnorm, purrr, rlang, tibble, tidyr, tidyselect

Suggests ggplot2, knitr, MASS, mgcv, rmarkdown, sn, testthat

VignetteBuilder knitr

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

Config/checklist/communities inbo

Config/checklist/keywords missing data, multiple imputation, Rubin

Encoding UTF-8

Language en-GB

LazyData TRUE

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Collate 'raw_imputed_class.R' 'aggregated_imputed_class.R'
'aggregate_impute.R' 'check_old_names.R' 'datasets.R'
'generate_data.R' 'hurdle_impute.R' 'import_s3_classes.R'
'impute_generic.R' 'impute_glmermod.R' 'impute_inla.R'
'impute_lm.R' 'missing_at_random.R' 'missing_current_count.R'
'missing_observed.R' 'missing_volunteer.R' 'model_impute.R'

Repository <https://thierryo.r-universe.dev>
Date/Publication 2025-02-27 16:24:50 UTC
RemoteUrl <https://github.com/inbo/multimput>
RemoteRef HEAD
RemoteSha 654e208239cc80e7322d8904ada7fae11a07d344

Contents

aggregate_impute	2
aggregatedImputed-class	3
generate_data	4
hurdle_impute	5
impute	6
maybeInla-class	7
missing_at_random	7
missing_current_count	8
missing_observed	8
missing_volunteer	9
model_impute	10
rawImputed-class	12
waterfowl	12
Index	13

aggregate_impute	<i>Aggregate an imputed dataset</i>
------------------	-------------------------------------

Description

Aggregate an imputed dataset

Usage

```
aggregate_impute(object, grouping, fun, filter = list(), join)

## S4 method for signature 'ANY'
aggregate_impute(object, grouping, fun, filter = list(), join)

## S4 method for signature 'rawImputed'
aggregate_impute(object, grouping, fun, filter = list(), join)
```

```
## S4 method for signature 'aggregatedImputed'
aggregate_impute(object, grouping, fun, filter = list(), join)
```

Arguments

object	A rawImputed object.
grouping	A vector of variables names to group the aggregation on.
fun	The function to aggregate.
filter	An optional argument to filter the raw dataset before aggregation. Will be passed to <code>dplyr::filter()</code> .
join	An optional argument to filter the raw dataset based on a data.frame. A <code>dplyr::semi_join()</code> will be applied with join or each element of join in case join is a list.

Examples

```
dataset <- generate_data(n_year = 10, n_site = 50, n_run = 1)
dataset$Count[sample(nrow(dataset), 50)] <- NA
model <- lm(Count ~ Year + factor(Period) + factor(Site), data = dataset)
imputed <- impute(data = dataset, model = model)
aggregate_impute(imputed, grouping = c("Year", "Period"), fun = sum)
```

aggregatedImputed-class

The aggregatedImputed class Holds an aggregated imputation data set

Description

The aggregatedImputed class Holds an aggregated imputation data set

Slots

Covariate A data.frame with the covariates.

Imputation A matrix with aggregated imputed values.

generate_data	<i>Generate simulated data</i>
---------------	--------------------------------

Description

Generate data for a regular monitoring design. The counts follow a negative binomial distribution with given size parameters and the true mean μ depending on a year, period and site effect. All effects are independent from each other and have, on the log-scale, a normal distribution with zero mean and given standard deviation.

Usage

```
generate_data(
  intercept = 2,
  n_year = 24,
  n_period = 6,
  n_site = 20,
  year_factor = FALSE,
  period_factor = FALSE,
  site_factor = FALSE,
  trend = 0.01,
  sd_rw_year = 0.1,
  amplitude_period = 1,
  mean_phase_period = 0,
  sd_phase_period = 0.2,
  sd_site = 1,
  sd_rw_site = 0.02,
  sd_noise = 0.01,
  size = 2,
  n_run = 10,
  as_list = FALSE,
  details = FALSE
)
```

Arguments

intercept	The global mean on the log-scale.
n_year	The number of years.
n_period	The number of periods.
n_site	The number of sites.
year_factor	Convert year to a factor. Defaults to FALSE.
period_factor	Convert period to a factor. Defaults to FALSE.
site_factor	Convert site to a factor. Defaults to FALSE.
trend	The long-term linear trend on the log-scale.
sd_rw_year	The standard deviation of the year effects on the log-scale.

amplitude_period	The amplitude of the periodic effect on the log-scale.
mean_phase_period	The mean of the phase of the periodic effect among years. Defaults to 0.
sd_phase_period	The standard deviation of the phase of the periodic effect among years.
sd_site	The standard deviation of the site effects on the log-scale.
sd_rw_site	The standard deviation of the random walk along year per site on the log-scale.
sd_noise	The standard deviation of the noise effects on the log-scale.
size	The size parameter of the negative binomial distribution.
n_run	The number of runs with the same mu.
as_list	Return the dataset as a list rather than a data.frame. Defaults to FALSE.
details	Add variables containing the year, period and site effects. Defaults tot FALSE.

Value

A data.frame with five variables. Year, Month and Site are factors identifying the location and time of monitoring. Mu is the true mean of the negative binomial distribution in the original scale. Count are the simulated counts.

hurdle_impute	<i>Combine two models into a hurdle model</i>
---------------	---

Description

Multiplies the imputed values for the presence model with those of the count model. Please make sure that the order of the observations in both models is identical. The resulting object will contain the union of the covariates of both models. Variables with the same name and different values get a presence_ or count_ prefix.

Usage

```
hurdle_impute(presence, count)
```

Arguments

presence	the rawImputed object for the presence.
count	the rawImputed object for counts.

impute

*Impute a dataset***Description**

Impute a dataset

Usage

```

impute(model, ..., extra, n_imp = 19)

## S4 method for signature 'ANY'
impute(model, ..., extra, n_imp = 19)

## S4 method for signature 'glmerMod'
impute(model, data, ..., extra, n_imp)

## S4 method for signature 'maybeInla'
impute(
  model,
  ...,
  seed = 0L,
  num_threads = NULL,
  parallel_configs = TRUE,
  extra,
  n_imp = 19
)

## S4 method for signature 'lm'
impute(model, data, ..., extra, n_imp)

```

Arguments

model	model to impute the dataset
...	other arguments. See details
extra	a <code>data.frame</code> with extra observations not used in the model. They will be added in subsequent analyses.
n_imp	the number of imputations. Defaults to 19.
data	The dataset holding both the observed and the missing values
seed	See the same argument in <code>INLA::inla.qsample()</code> 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 the example in <code>INLA::inla.posterior.sample()</code> . 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.

`num_threads` The number of threads to use in the format "A:B" defining the number threads in the outer (A) and inner (B) layer for nested parallelism. A "0" will be replaced intelligently. `seed != 0` requires serial computations.

`parallel_configs` Logical. If TRUE and not on Windows, then try to run each configuration in parallel (not Windows) using A threads (see `num_threads`), where each of them is using B:0 threads.

Examples

```
dataset <- generate_data(n_year = 10, n_site = 50, n_run = 1)
dataset$Count[sample(nrow(dataset), 50)] <- NA
model <- lm(Count ~ Year + factor(Period) + factor(Site), data = dataset)
impute(model, dataset)
```

maybeInla-class	<i>The maybeInla class</i>
-----------------	----------------------------

Description

A superclass holding either NULL or an object of the inla class.

missing_at_random	<i>Generate missing data at random</i>
-------------------	--

Description

The observed values will be either equal to the counts or missing. The probability of missing is the inverse of the counts + 1.

Usage

```
missing_at_random(
  dataset,
  proportion = 0.25,
  count_variable = "Count",
  observed_variable = "Observed"
)
```

Arguments

`dataset` A dataset to a the observation with missing data.

`proportion` The proportion of observations that will be missing.

`count_variable` The name of the variable holding the counts.

`observed_variable` The name of the variable holding the observed values = either count or missing.

missing_current_count *Generate missing data depending on the counts*

Description

The observed values will be either equal to the counts or missing. The probability of missing is the inverse of the counts + 1.

Usage

```
missing_current_count(
  dataset,
  proportion = 0.25,
  count_variable = "Count",
  observed_variable = "Observed"
)
```

Arguments

dataset	A dataset to a the observation with missing data.
proportion	The proportion of observations that will be missing.
count_variable	The name of the variable holding the counts.
observed_variable	The name of the variable holding the observed values = either count or missing.

missing_observed *Generate missing data based on the observed patterns in the real dataset.*

Description

The observed values will be either equal to the counts or missing. The probability of missing is the inverse of the counts + 1.

Usage

```
missing_observed(
  dataset,
  count_variable = "Count",
  observed_variable = "Observed",
  site_variable = "Site",
  year_variable = "Year",
  period_variable = "Period"
)
```

Arguments

dataset	A dataset to a the observation with missing data.
count_variable	The name of the variable holding the counts.
observed_variable	The name of the variable holding the observed values = either count or missing.
site_variable	The name of the variable holding the sites.
year_variable	The name of the variable holding the years.
period_variable	The name of the variable holding the period.

missing_volunteer	<i>Generate missing data mimicking choices made by volunteers.</i>
-------------------	--

Description

The observed values will be either equal to the counts or missing. The probability of missing is the inverse of the counts + 1.

Usage

```
missing_volunteer(
  dataset,
  proportion = 0.25,
  count_variable = "Count",
  observed_variable = "Observed",
  year_variable = "Year",
  site_variable = "Site",
  max_count = 100
)
```

Arguments

dataset	A dataset to a the observation with missing data.
proportion	The proportion of observations that will be missing.
count_variable	The name of the variable holding the counts.
observed_variable	The name of the variable holding the observed values = either count or missing.
year_variable	The name of the variable holding the years.
site_variable	The name of the variable holding the sites.
max_count	The maximum count.

model_impute	<i>Model an imputed dataset</i>
--------------	---------------------------------

Description

Model an imputed dataset

Usage

```
model_impute(  
  object,  
  model_fun,  
  rhs,  
  model_args = list(),  
  extractor,  
  extractor_args = list(),  
  filter = list(),  
  mutate = list(),  
  ...,  
  timeout = 600  
)  
  
## S4 method for signature 'ANY'  
model_impute(  
  object,  
  model_fun,  
  rhs,  
  model_args = list(),  
  extractor,  
  extractor_args = list(),  
  filter = list(),  
  mutate = list(),  
  ...,  
  timeout = 600  
)  
  
## S4 method for signature 'aggregatedImputed'  
model_impute(  
  object,  
  model_fun,  
  rhs,  
  model_args = list(),  
  extractor,  
  extractor_args = list(),  
  filter = list(),  
  mutate = list(),  
  ...,
```

```

    timeout = 600
  )

```

Arguments

object	The imputed dataset.
model_fun	The function to apply on each imputation set. Or a string with the name of the function. Include the package name when the function is not in one of the base R packages. For example: "glm" or "INLA::inla".
rhs	The right hand side of the model.
model_args	An optional list of arguments to pass to the model function.
extractor	A function which return a matrix or data.frame. The first column should contain the estimate, the second the standard error of the estimate.
extractor_args	An optional list of arguments to pass to the extractor function.
filter	An optional argument to filter the aggregated dataset. Either a function which takes the Covariate slot as an argument. Or a list which will be passed to the .dots argument of <code>dplyr::filter()</code> . You can filter on the covariates in the aggregated dataset. Besides those you can also filter on Imputation_min and Imputation_max. These variables represent the lowest and highest value of the imputations per row in the data.
mutate	An optional argument to alter the aggregated dataset. Will be passed to the .dots argument of <code>dplyr::mutate()</code> . This is mainly useful for simple conversions, e.g. factors to numbers and vice versa.
...	currently ignored.
timeout	Maximum duration allowed for fitting a single imputation model in seconds. Defaults to 600 seconds (10 minutes).

Examples

```

dataset <- generate_data(n_year = 10, n_site = 50, n_run = 1)
dataset$Count[sample(nrow(dataset), 50)] <- NA
model <- lm(Count ~ Year + factor(Period) + factor(Site), data = dataset)
imputed <- impute(data = dataset, model = model)
aggr <- aggregate_impute(imputed, grouping = c("Year", "Period"), fun = sum)
extractor <- function(model) {
  summary(model)$coefficients[, c("Estimate", "Std. Error")]
}
model_impute(
  object = aggr,
  model_fun = lm,
  rhs = "0 + factor(Year)",
  extractor = extractor
)

```

rawImputed-class	<i>The rawImputed class Holds a dataset and imputed values</i>
------------------	--

Description

The rawImputed class Holds a dataset and imputed values

Slots

Data A data.frame with the data.

Response A character holding the name of the response variable.

Minimum An optional character holding the name of the variable with the minimum.

Imputation A matrix with imputed values.

Extra A data.frame with extra data to add to the imputations. This data is not used in the imputation model. It must contain the same variables as the original data.

waterfowl	<i>The observation pattern in the Flemish waterfowl dataset</i>
-----------	---

Description

Data for fig 1 and 2 in Onkelinx et al

Usage

```
data(waterfowl)
```

Format

A data frame with 77157 rows and 5 variables

Details

- Site Site ID.
- Winter Winter ID.
- Period ID of the month.
- Species Number of observed species.
- Birds Total number of birds.

Index

* datasets

- waterfowl, [12](#)
- aggregate_impute, [2](#)
- aggregate_impute, aggregatedImputed-method
(aggregate_impute), [2](#)
- aggregate_impute, ANY-method
(aggregate_impute), [2](#)
- aggregate_impute, rawImputed-method
(aggregate_impute), [2](#)
- aggregatedImputed-class, [3](#)
- dplyr::filter(), [3](#), [11](#)
- dplyr::mutate(), [11](#)
- dplyr::semi_join(), [3](#)
- generate_data, [4](#)
- hurdle_impute, [5](#)
- impute, [6](#)
- impute, ANY-method (impute), [6](#)
- impute, glmerMod-method (impute), [6](#)
- impute, lm-method (impute), [6](#)
- impute, maybeInla-method (impute), [6](#)
- INLA::inla.posterior.sample(), [6](#)
- INLA::inla.qsample(), [6](#)
- maybeInla-class, [7](#)
- missing_at_random, [7](#)
- missing_current_count, [8](#)
- missing_observed, [8](#)
- missing_volunteer, [9](#)
- model_impute, [10](#)
- model_impute, aggregatedImputed-method
(model_impute), [10](#)
- model_impute, ANY-method (model_impute),
[10](#)
- rawImputed-class, [12](#)
- waterfowl, [12](#)