Package: CGMissingDataR (via r-universe)

May 29, 2026

Title Impute Missing Glucose Values in CGM Data

Version 0.0.2

Description Imputes missing glucose values in repeated-measures continuous glucose monitoring (CGM) data. Workflows create time-series features from raw timestamps, support model selection, and return the user's original columns plus an imputed glucose column. Methods include multiple imputation by chained equations (MICE; Azur et al. (2011) <doi:10.1002/mpr.329>), Random Forest regression (Breiman (2001) <doi:10.1023/A:1010933404324>), k-nearest-neighbor regression (Zhang (2016) <doi:10.21037/atm.2016.03.37>), XGBoost (Chen and Guestrin (2016) <doi:10.1145/2939672.2939785>), LightGBM (Ke et al. (2017) <https://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision>), and ARIMA forecasting with the forecast framework (Hyndman and Khandakar (2008) <doi:10.18637/jss.v027.i03>). A Python-compatible backend uses 'reticulate' to call 'pandas', 'scikit-learn', 'statsmodels', Python 'xgboost', and optional Python 'lightgbm'.

License GPL (>= 2)

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 4.3)

RoxygenNote 7.3.3

Imports mice, FNN, ranger, data.table, xgboost, lightgbm, forecast, CGManalyzer, lifecycle, reticulate, shiny

Suggests testthat (>= 3.0.0), spelling, knitr, rmarkdown

Config/testthat/edition 3

NeedsCompilation no

Language en-US

URL https://zhanglabuky.github.io/CGMmissingDataR/,
https://github.com/ZhangLabUKY/CGMmissingDataR

BugReports https://github.com/ZhangLabUKY/CGMmissingDataR/issues

LazyData true

VignetteBuilder knitr

Config/pak/sysreqs cmake make libicu-dev libpng-dev libuv1-dev python3 libx11-dev zlib1g-dev

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

Date/Publication 2026-05-29 21:45:08 UTC

RemoteUrl https://github.com/zhanglabuky/cgmmissingdatar

RemoteRef HEAD

RemoteSha b755964671a05e43cd1ee15630fe4a46a77b30fe

Contents

`CGMExmplDat10Pct` *Example dataset for CGMissingData*

Description

A small multi-subject CGM dataset intended for real missing-value imputation examples. It contains 50 deterministic missing glucose values.

Usage

```
CGMExmplDat10Pct
```

Format

A data frame with 500 rows and 5 variables:

USUBJID Numeric subject identifier.

SEX Synthetic sex of the subject.

LBORRES Laboratory Observed Result for Glucose (numeric), with deterministic missing values.

Time Raw timestamp in `yyyy:mm:dd:hh:nn` format.

AGE Synthetic age in years.

hba1c Synthetic HbA1c value.

Examples

```
data("CGMExmplDat10Pct")
```

`CGMExmplDat5Pct` *Example dataset for CGMissingData*

Description

A small multi-subject CGM dataset intended for real missing-value imputation examples. It contains 50 deterministic missing glucose values.

Usage

```
CGMExmplDat5Pct
```

Format

A data frame with 500 rows and 5 variables:

USUBJID Numeric subject identifier.

SEX Synthetic sex of the subject.

LBORRES Laboratory Observed Result for Glucose (numeric), with deterministic missing values.

Time Raw timestamp in `yyyy:mm:dd:hh:nn` format.

AGE Synthetic age in years.

hba1c Synthetic HbA1c value.

Examples

```
data("CGMExmplDat5Pct")
```

`run_app` *Launch the CGMmissingDataR Shiny App*

Description

Launches a Shiny app for uploading a CGM data file, selecting the target, subject, timestamp, and feature columns, running `run_missing_glucose_imputation()`, previewing the imputed data, and downloading the completed data as a CSV file.

Usage

```
run_app()
```

Value

Invisibly returns the result of `shiny::runApp()`.

Examples

```
## Not run:
# Run the CGMmissingDataR Shiny app
run_app()

## End(Not run)
```

`run_missing_glucose_imputation` *Impute missing glucose values using selectable MICE-based methods*

Description

Imputes missing glucose values in continuous glucose monitoring (CGM) data. The function handles both explicit missing glucose values already coded as `NA` and implicit missing readings caused by timestamp gaps. Before imputation, each subject is regularized to an equal `interval_minutes` timestamp grid; missing timestamp gaps are converted into explicit rows with `target_col = NA`, then imputed using the selected backend and final imputation method.

Usage

```
run_missing_glucose_imputation(
  data,
  target_col,
  feature_cols = NULL,
  id_col = "USUBJID",
  time_col = "Time",
  time_format = "yyyy:mm:dd:hh:nn",
  time_unit = "minute",
  models = "auto",
  rf_n_estimators = 200,
  knn_k = 7,
  xgb_nrounds = 300,
  lgb_nrounds = 400,
  n_threads = 1L,
  arima_order = c(4L, 1L, 0L),
  seed = 42,
  lag_k = c(1L, 2L, 3L),
  add_rollmean = TRUE,
  roll_window = 3L,
  interval_minutes = 5L,
```

```
  missing_warning_threshold = 0.2,
  study_start = NULL,
  study_end = NULL,
  use_arima_if_missing_leq = 0.05,
  arima_min_history = 20L,
  imputer_backend = c("mice", "sklearn"),
  export = FALSE
)
```

Arguments

data	A data.frame, an object coercible to data.frame, or a path to a CSV file.
target_col	Single character string: target glucose column with missing values to impute. Python default name is "glucose_value".
feature_cols	Optional character vector of base feature columns. If NULL, the Python pipeline feature set is used when available: TimeSeries, TimeDifferenceMinutes, id_col, AGE, SEX, HBA1C, lag1, lag2, lag3, and rollmean. If supplied, the listed columns are used together with the generated time, subject, lag, and rolling-mean columns that exist in the data.
id_col	Character string: subject identifier column. Python default name is "subjectid".
time_col	Character string: raw timestamp column. Python default name is "timestamp".
time_format	Retained for compatibility with the old R function. The Python-engine path uses pandas timestamp parsing.
time_unit	Retained for compatibility with the old R function and not used by the strict Python-engine path.
models	Final real-imputation method selector. Use NULL or "auto" to keep the default missing-rate rule: MICE+ARIMA when the target missing rate is less than or equal to use_arima_if_missing_leq, otherwise MICE+XGBoost. Use exactly one of "arima", "xgboost", "rf", "knn", or "lightgbm" to force a specific method regardless of missing rate.
rf_n_estimators	Integer number of Random Forest trees. Used when models = "rf".
knn_k	Integer number of nearest neighbors. Used when models = "knn".
xgb_nrounds	Integer number of XGBoost boosting rounds. Used when models = "xgboost" and may be used by models = "auto" when the missing-rate rule selects XGBoost.
lgb_nrounds	Integer number of LightGBM boosting rounds. Used when models = "lightgbm".
n_threads	Integer number of model-fitting threads for engines that support thread controls. The default 1L is conservative for CRAN and shared systems. Increase for faster local XGBoost, Random Forest, and LightGBM runs. ARIMA and kNN do not use this setting.
arima_order	Integer vector of length 3. Python default is c(4L, 1L, 0L).
seed	Integer seed for reproducible MICE, tree-based models, and the Python-compatible backend. Default is 42.

`lag_k` Integer vector of target lags to compute. Python default is `c(1L, 2L, 3L)`.

`add_rollmean` Logical: add rolling mean of prior target values. Python always adds this; setting `FALSE` is allowed only for compatibility.

`roll_window` Integer rolling mean window. Python default is 3.

`interval_minutes`
Expected spacing, in minutes, between consecutive CGM readings. The default is `5`. The function uses this value to regularize each subject's timestamps to an equal-interval grid before imputation.

`missing_warning_threshold`
Numeric value between 0 and 1. If the missingness rate in `target_col` after timestamp-gap regularization exceeds this threshold, a warning is issued. Default is `0.20`.

`study_start` Optional study start timestamp. If supplied, the function reports subjects whose first observed CGM timestamp occurs after this time. Leading study time is not imputed.

`study_end` Optional study end timestamp. If supplied, the function reports subjects whose last observed CGM timestamp occurs before this time. Trailing study time is not imputed.

`use_arima_if_missing_leq`
Numeric missing-rate threshold used only when `models` is `NULL` or `"auto"`. If the target missing rate is less than or equal to this value, segmentwise ARIMA is used; otherwise XGBoost is used. Default is 0.05.

`arima_min_history`
Minimum number of prior observations required before fitting ARIMA for a missing segment. Python default is 20.

`imputer_backend`
One of `"mice"` or `"sklearn"`. `"mice"` uses the R package `mice` as the CRAN-safe R-native backend. `"sklearn"` uses Python modules through `reticulate` for a Python-compatible workflow.

`export` Logical; if `TRUE`, writes the returned imputed data frame to a timestamped CSV file in the current working directory. Default is `FALSE`.

Details

The imputation workflow first parses and sorts timestamps within each subject. Each subject is regularized to an equal `interval_minutes` grid. If a reading is missing because the timestamp is absent from the input data, a new row is inserted and the target glucose value is set to `NA`. These inserted missing values are then imputed using the same workflow as explicit `NA` values. The deterministic interval grid is controlled by this package; `CGManalyzer`'s equal-interval helper is called internally for workflow consistency.

Internally, the function creates time features, lag features, and rolling-mean features to support imputation. MICE first completes the target and feature matrix. The selected final method then fills the missing glucose positions in `imputed_glucose_value`: either by segmentwise ARIMA or by a supervised model trained on observed glucose values and the MICE-completed feature matrix. These engineered columns are used only during model fitting and are removed from the returned data frame.

imputed_glucose_value is returned as a continuous numeric model estimate. Users who require whole-number glucose values for reporting can round this column after imputation.

Missingness warnings are based on the data after timestamp-gap regularization, so both explicit NA glucose values and rows created from timestamp gaps contribute to the reported missingness rate. The function also warns when long contiguous missing blocks of at least 12 or 24 hours are detected. If study_start or study_end is supplied, leading or trailing study-period coverage gaps are reported but are not imputed.

Value

A data.frame containing the original user-supplied columns plus imputed_glucose_value, the completed glucose column. The original target column is left unchanged, so values that were originally missing or created from timestamp gaps remain NA in target_col, while their completed values are stored in imputed_glucose_value.

Examples

```
data("CGMExmplDat5Pct")
out <- run_missing_glucose_imputation(
  CGMExmplDat5Pct,
  target_col = "LBORRES",
  feature_cols = c("AGE", "hba1c"),
  id_col = "USUBJID",
  time_col = "Time",
  imputer_backend = "mice"
)
head(subset(out, is.na(LBORRES)))
```

---

run_missingness_benchmark

Run missingness benchmark (target-masking with LAG features)

Description

[Deprecated]

This function is deprecated. Use run_missing_glucose_imputation() for real missing glucose values.

This function implements missingness benchmarking by masking the target column at various rates and evaluating imputation and predictive performance of MICE, Random Forest, and KNN methods. Additionally, it includes LAG features of the target variable to assess their impact on imputation and prediction. The function returns a data.frame summarizing the Mask Rate, Method, MRD (Mean Relative Difference), and Masked Count for each method and mask rate.

Usage

```
run_missingness_benchmark(
  data,
  target_col,
  feature_cols = NULL,
  id_col = "USUBJID",
  time_col = "TimeSeries",
  mask_rates = c(0.05, 0.1, 0.2, 0.3, 0.4),
  mask_type = c("random", "block"),
  rf_n_estimators = 400,
  knn_k = 7,
  seed = 42,
  lag_k = c(1, 2, 3),
  add_rollmean = TRUE,
  roll_window = 3
)
```

Arguments

`data`	A data.frame (or object coercible to data.frame), OR a path to a CSV file.
`target_col`	Single character string: name of the outcome column to mask/impute (e.g., "LBORRES", "Glucose").
`feature_cols`	Character vector of base feature columns (excluding the target). If NULL, uses all columns except `target_col`.
`id_col`	Character string: subject identifier column used for LAG features (default "USUBJID").
`time_col`	Character string: time-ordering column used for LAG features (default "TimeSeries").
`mask_rates`	Numeric vector in (0, 1): fraction of rows to mask (default 0.05, 0.10, 0.20, 0.30, 0.40).
`mask_type`	One of `"random"` or `"block"`.
`rf_n_estimators`	Integer: number of trees for random forest (default 400).
`knn_k`	Integer: number of neighbors for kNN (default 7).
`seed`	Integer: random seed used for MICE and models (default 42).
`lag_k`	Integer vector of lags to compute on the target (default c(1,2,3)).
`add_rollmean`	Logical: add rolling mean feature of prior target values (default TRUE).
`roll_window`	Integer: rolling window length for rollmean (default 3).

Details

LAG features are computed using `data.table::shift()` (fast lag/lead). The rolling mean is computed with `data.table::frollmean()` using `align="right"` and `fill=NA`.

Value

A data.frame with columns: MaskRate, Method, MRD, MaskedCount.

Index