Package 'SBMTrees'

May 7, 2026

Type Package

Title Longitudinal Sequential Imputation and Prediction with Bayesian Trees Mixed-Effects Models for Longitudinal Data

Version 1.5

Date 2026-02-06

Author Jungang Zou [aut, cre],
Liangyuan Hu [aut],
Robert McCulloch [ctb],
Rodney Sparapani [ctb],
Charles Spanbauer [ctb],
Robert Gramacy [ctb],
Jean-Sebastien Roy [ctb]

Maintainer Jungang Zou <jungang.zou@gmail.com>

Description Implements a sequential imputation framework using Bayesian Mixed-Effects Trees ('SBMTrees') for handling missing data in longitudinal studies. The package supports a variety of models, including non-linear relationships and non-normal random effects and residuals, leveraging Dirichlet Process priors for increased flexibility. Key features include handling Missing at Random (MAR) longitudinal data, imputation of both covariates and outcomes, and generating posterior predictive samples for further analysis. The methodology is designed for applications in epidemiology, biostatistics, and other fields requiring robust handling of missing data in longitudinal settings.

License GPL-2

Encoding UTF-8

Depends R (>= 4.1.0)

Imports Rcpp, lme4, Matrix, arm, dplyr, mvtnorm, sn, mice, nnet, MASS

LinkingTo Rcpp, RcppArmadillo, RcppDist, RcppProgress, pg

RoxygenNote 7.3.3

SystemRequirements GNU make

Suggests knitr, rmarkdown, mitml

VignetteBuilder knitr

NeedsCompilation yes

Repository CRAN
Date/Publication 2026-02-12 08:10:27 UTC

Contents

`SBMTrees-package` *Sequential Imputation with Bayesian Trees Mixed-Effects Models*

Description

The SBMTrees package implements a Bayesian non-parametric framework for imputing missing covariates and outcomes in longitudinal data under the Missing at Random (MAR) assumption. Its core model, the Bayesian Trees Mixed-Effects Model (BMTrees), extends Mixed-Effects BART by employing centralized Dirichlet Process (CDP) Normal Mixture priors. This allows handling non-normal random effects and errors, addressing model misspecification, and capturing complex relationships.

Details

SBMTrees offers tools for predicting and imputing missing values in longitudinal data using Bayesian Trees Mixed-Effects Models. The package supports various semiparametric variants, including BMTrees_R and BMTrees_RE, and integrates mixedBART as a baseline model. Key functionalities include:

- `BMTrees_prediction`: Predicts longitudinal outcomes based on mixed-effects models.

- `sequential_imputation`: Imputes missing covariates and outcomes sequentially in longitudinal datasets.

The package supports flexibility in specifying priors for random effects and errors, making it suitable for diverse longitudinal data settings. Core computations leverage efficient Gibbs samplers implemented in C++.

This package modifies and extends C++ code originally derived from the BART3 package, developed by Rodney Sparapani, which is licensed under the GNU General Public License version 2 (GPL-2).

The modified code is redistributed in accordance with the GPL-2 license. For more details on the modifications, see the package's documentation.

Note

This package and all associated documentation are licensed under the GNU General Public License version 2 (GPL-2). See the LICENSE file for the full text of the license.

Author(s)

Jungang Zou <jungang.zou@gmail.com>

References

BART3 package: https://github.com/rsparapa/bnptools/tree/master, originally developed by Rodney Sparapani.

See Also

BMTrees_prediction, sequential_imputation

---

apply_locf_nocb *Initialize Missing Values using LOCF and NOCB*

Description

Imputes missing values in longitudinal data using a hierarchical three-step strategy to ensure complete data for model initialization. The process prioritizes within-subject information using Last Observation Carried Forward (LOCF) and Next Observation Carried Backward (NOCB), falling back to cross-sectional summary statistics (mean or mode) only when a subject has absolutely no observed data for a specific variable.

Usage

```
apply_locf_nocb(X, subject_id, is_binary)
```

Arguments

X	A data.frame or matrix containing the variables to be imputed. Columns correspond to variables.
subject_id	A vector of subject identifiers with length equal to nrow(X).
is_binary	A vector of length ncol(X) indicating the type of each variable. Values can be TRUE/1 (for binary variables) or FALSE/0 (for continuous variables).

Details

Pre-requisite: The rows of `X` must be ordered by time within each subject prior to calling this function.

The imputation proceeds in three specific stages:

1. **Subject-wise LOCF**: For each subject, missing values are filled using the immediately preceding observed value (forward fill). This handles gaps in the middle or end of a subject's timeline.
2. **Subject-wise NOCB**: For each subject, any remaining missing values (typically at the start of the timeline, before the first observation) are filled using the next available observed value (backward fill).
3. **Global Fallback**: If a subject has *no* observed data for a specific variable (i.e., the entire column is `NA` for that `subject_id`), the function imputes these values using the global statistics calculated from the rest of the population:
   - **Continuous variables**: Imputed with the global mean.
   - **Binary variables**: Imputed with the global mode (ties default to 0).

Value

A data.frame with the same dimensions as `X` but with all missing values imputed.

Examples

```
# Create a toy dataset with missing values
X <- data.frame(
  cont = c(NA, 5, NA,   NA, NA, NA),  # Subj 1: Gap/Lead/Trail, Subj 2: All NA
  bin  = c(0, NA, 1,    1, 1, 0)      # Subj 1: Gap,            Subj 2: Complete
)
subject_id <- c(1, 1, 1, 2, 2, 2)
is_binary <- c(FALSE, TRUE)

# Run imputation
X_imputed <- apply_locf_nocb(X, subject_id, is_binary)
```

`BMLMM_prediction` *Bayesian Mixed Linear Models for Predicting Longitudinal Outcomes with DP Priors*

Description

Provides predictions for outcomes in longitudinal data using Bayesian Mixed Linear Models (BMLMM). Unlike the tree-based variant, this function assumes a linear relationship for fixed effects while maintaining the flexible centralized Dirichlet Process (DP) framework for random effects and residuals. It predicts values for test data while accounting for complex error structures.

Usage

```
BMLMM_prediction(
  X_train,
  Y_train,
  Z_train,
  subject_id_train,
  X_test,
  Z_test,
  subject_id_test,
  model = c("BMTrees", "BMTrees_R", "BMTrees_RE", "mixedBART"),
  binary = FALSE,
  nburn = 3000L,
  npost = 4000L,
  skip = 1L,
  verbose = TRUE,
  seed = NULL,
  tol = 1e-20,
  add_intercept = TRUE
)
```

Arguments

`X_train`	A matrix of covariates in the training set.
`Y_train`	A numeric or logical vector of outcomes in the training set.
`Z_train`	A matrix of random predictors in the training set.
`subject_id_train`	A character vector of subject IDs in the training set.
`X_test`	A matrix of covariates in the testing set.
`Z_test`	A matrix of random predictors in the testing set.
`subject_id_test`	A character vector of subject IDs in the testing set.
`model`	A character string specifying the distribution assumptions for residuals and random effects. Options are: • `"BMTrees"` (default): DP priors for both residuals and random effects. • `"BMTrees_R"`: DP prior for residuals, Normal prior for random effects. • `"BMTrees_RE"`: Normal prior for residuals, DP prior for random effects. • `"mixedBART"`: Normal priors for both residuals and random effects.
`binary`	Logical. Indicates whether the outcome is binary (`TRUE`) or continuous (`FALSE`). Default: `FALSE`.
`nburn`	An integer specifying the number of burn-in iterations for the Gibbs sampler. Default: `3000L`.
`npost`	An integer specifying the number of posterior samples to collect. Default: `4000L`.
`skip`	An integer indicating the thinning interval for MCMC samples. Default: `1L`.

verbose	Logical. If TRUE, displays MCMC progress. If FALSE, shows a progress bar. Default: TRUE.
seed	An optional integer for setting the random seed to ensure reproducibility. Default: NULL.
tol	A numeric tolerance value to prevent numerical overflow and underflow in the model. Default: 1e-20.
add_intercept	Logical. If TRUE, adds a column of ones (intercept) to the covariate matrices X_train and X_test. Default: TRUE.

Value

A list containing posterior samples and predictions:

post_beta Posterior samples of the regression coefficients (fixed effects).

post_lmm_train Posterior samples of the fixed-effects predictions ($X\beta$) on training data.

post_Sigma Posterior samples of covariance matrices in random effects.

post_lambda_G Posterior samples of lambda parameter in DP normal mixture on random errors.

post_lambda_F Posterior samples of lambda parameter in DP normal mixture on random-effects.

post_B Posterior samples of the coefficients in random effects.

post_random_effect_train Posterior samples of random effects for training data.

post_sigma Posterior samples of error deviation.

post_expectation_y_train Posterior expectations of training data outcomes, equal to fixed-effects + random effects.

post_expectation_y_test Posterior expectations of testing data outcomes, equal to fixed-effects + random effects.

post_predictive_y_train Posterior predictive distributions for training outcomes, equal to fixed-effects + random effects + predictive residual.

post_predictive_y_test Posterior predictive distributions for testing outcomes, equal to fixed-effects + random effects + predictive residual.

post_eta Posterior samples of location parameters in DP normal mixture on random errors.

post_mu Posterior samples of location parameters in DP normal mixture on random effects.

Note

This function utilizes modified C++ code originally derived from the BART3 package (Bayesian Additive Regression Trees). The original package was developed by Rodney Sparapani and is licensed under GPL-2. Modifications were made by Jungang Zou, 2024.

References

For more information about the original BART3 package, see: https://github.com/rsparapa/bnptools/tree/master/BART3

Examples

```
data <- simulation_prediction_conti(
   train_prop = 0.7,
   n_subject = 20,
   seed = 1,
   nonlinear = FALSE,
   residual = "normal",
   randeff = "MVN"
)
model <- BMLMM_prediction(
   X_train = data$X_train,
   Y_train = data$Y_train,
   Z_train = data$Z_train,
   subject_id_train = data$subject_id_train,
   X_test = data$X_test,
   Z_test = data$Z_test,
   subject_id_test = data$subject_id_test,
   model = "BMTrees",
   binary = FALSE,
   nburn = 0L, npost = 1L, skip = 1L, verbose = FALSE, seed = 1
)
```

`BMTrees_prediction` *Bayesian Trees Mixed-Effects Models for Predicting Longitudinal Outcomes*

Description

Provides predictions for outcomes in longitudinal data using Bayesian Trees Mixed-Effects Models (BMTrees) and its semiparametric variants. The function predicts values for test data while accounting for random effects, complex relationships, and potential model misspecification.

Usage

```
BMTrees_prediction(
  X_train,
  Y_train,
  Z_train,
  subject_id_train,
  X_test,
  Z_test,
  subject_id_test,
  model = c("BMTrees", "BMTrees_R", "BMTrees_RE", "mixedBART"),
  binary = FALSE,
  nburn = 3000L,
  npost = 4000L,
  skip = 1L,
  verbose = TRUE,
```

```
  seed = NULL,
  tol = 1e-20,
  ntrees = 200,
  pi_DP = 0.99,
  k = 2
)
```

Arguments

X_train	A matrix of covariates in the training set.
Y_train	A numeric or logical vector of outcomes in the training set.
Z_train	A matrix of random predictors in the training set.
subject_id_train	A character vector of subject IDs in the training set.
X_test	A matrix of covariates in the testing set.
Z_test	A matrix of random predictors in the testing set.
subject_id_test	A character vector of subject IDs in the testing set.
model	A character string specifying the predictive model. Options are "BMTrees", "BMTrees_R", "BMTrees_RE", and "mixedBART". Default: "BMTrees".
binary	Logical. Indicates whether the outcome is binary (TRUE) or continuous (FALSE). Default: FALSE.
nburn	An integer specifying the number of burn-in iterations for Gibbs sampler. Default: 3000L.
npost	An integer specifying the number of posterior samples to collect. Default: 4000L.
skip	An integer indicating the thinning interval for MCMC samples. Default: 1L.
verbose	Logical. If TRUE, displays MCMC progress. If FALSE, shows a progress bar. Default: TRUE.
seed	An optional integer for setting the random seed to ensure reproducibility. Default: NULL.
tol	A numeric tolerance value to prevent numerical overflow and underflow in the model. Default: 1e-20.
ntrees	An integer specifying the number of trees in BART. Default: 200.
pi_DP	A value between 0 and 1 for calculating the empirical prior in the DP prior. Default: 0.99.
k	A numeric value for the BART prior parameter controlling the standard deviation of the terminal node values. Default: 2.0.

Value

A list containing posterior samples and predictions:

post_tree_train Posterior samples of the fixed-effects from BART on training data.

post_Sigma Posterior samples of covariance matrices in random effects.

post_lambda_G Posterior samples of lambda parameter in DP normal mixture on random errors.

post_lambda_F Posterior samples of lambda parameter in DP normal mixture on random-effects.

post_B Posterior samples of the coefficients in random effects.

post_random_effect_train Posterior samples of random effects for training data.

post_sigma Posterior samples of error deviation.

post_expectation_y_train Posterior expectations of training data outcomes, equal to fixed-effects + random effects.

post_expectation_y_test Posterior expectations of testing data outcomes, equal to fixed-effects + random effects.

post_predictive_y_train Posterior predictive distributions for training outcomes, equal to fixed-effects + random effects + predictive residual.

post_predictive_y_test Posterior predictive distributions for testing outcomes, equal to fixed-effects + random effects + predictive residual.

post_eta Posterior samples of location parameters in DP normal mixture on random errors.

post_mu Posterior samples of location parameters in DP normal mixture on random effects.

Note

This function utilizes modified C++ code originally derived from the BART3 package (Bayesian Additive Regression Trees). The original package was developed by Rodney Sparapani and is licensed under GPL-2. Modifications were made by Jungang Zou, 2024.

References

For more information about the original BART3 package, see: https://github.com/rsparapa/bnptools/tree/master/BART3

Examples

```
data <- simulation_prediction_conti(
  train_prop = 0.7,
  n_subject = 20,
  seed = 1234,
  nonlinear = TRUE,
  residual = "normal",
  randeff = "MVN"
)
model <- BMTrees_prediction(
  X_train = data$X_train,
  Y_train = data$Y_train,
  Z_train = data$Z_train,
  subject_id_train = data$subject_id_train,
  X_test = data$X_test,
  Z_test = data$Z_test,
  subject_id_test = data$subject_id_test,
  model = "BMTrees",
  binary = FALSE,
  nburn = 0L, npost = 1L, skip = 1L, verbose = FALSE, seed = 1234
)
```

`sequential_imputation` *Longitudinal Sequential Imputation for Longitudinal Missing Data*

Description

Implements sequential imputation for missing covariates and outcomes in longitudinal data. The function uses a Bayesian non-parametric framework with mixed-effects models to handle both normal and non-normal random effects and errors. It sequentially imputes missing values by constructing univariate models in a fixed order, initializing with LOCF/NOCB, and ensuring consistency with a valid joint distribution.

Usage

```
sequential_imputation(
  X,
  Y,
  Z = NULL,
  subject_id,
  type,
  binary_outcome = FALSE,
  model = c("BMTrees", "BMTrees_R", "BMTrees_RE", "mixedBART"),
  outcome_model = c("BMTrees", "BMLM"),
  nburn = 0L,
  npost = 3L,
  skip = 1L,
  verbose = TRUE,
  seed = NULL,
  tol = 1e-20,
  k = 2,
  ntrees = 200,
  reordering = TRUE,
  pi_DP = 0.99
)
```

Arguments

`X`	A matrix of missing covariates.
`Y`	A vector of missing outcomes (numeric or logical).
`Z`	A matrix of complete random predictors. Default: `NULL`.
`subject_id`	A vector of subject IDs corresponding to the rows of `X` and `Y`. Can be integer, factor, or character.
`type`	A vector indicating whether each covariate in `X` is binary (1) or continuous (0).
`binary_outcome`	A logical value indicating whether the outcome `Y` is binary. Default: `FALSE`.
`model`	A character vector specifying the imputation model for the covariates. Options are `"BMTrees"` (default), `"BMTrees_R"` (residual DP), `"BMTrees_RE"` (random effect DP), and `"mixedBART"`.

outcome_model	A character vector specifying the model used for the outcome. Options are "BMTrees" (default) or "BMLM" (Bayesian Mixed Linear Model). If "BMLM" is selected, posterior estimates for beta and sigma are returned.
nburn	An integer specifying the number of burn-in iterations. Default: 0.
npost	An integer specifying the number of sampling iterations. Default: 3.
skip	An integer specifying the interval for keeping samples in the sampling phase. Default: 1.
verbose	A logical value indicating whether to display progress and MCMC information. Default: TRUE.
seed	A random seed for reproducibility. Default: NULL.
tol	A small numerical tolerance to prevent numerical overflow or underflow in the model. Default: 1e-20.
k	A numeric value for the BART prior parameter controlling the standard deviation of the terminal node values. Default: 2.0.
ntrees	An integer specifying the number of trees in BART. Default: 200.
reordering	A logical value indicating whether to apply a reordering strategy for sorting covariates based on missingness. Default: TRUE.
pi_DP	A value between 0 and 1 for calculating the empirical prior in the DP prior. Default: 0.99.

Details

The function builds on the Bayesian Trees Mixed-Effects Model (BMTrees), which extends Mixed-Effects BART by using centralized Dirichlet Process Normal Mixture priors. This framework handles non-normal random effects and errors, addresses model misspecification, and captures complex relationships.

The algorithm initializes missing values using Last Observation Carried Forward (LOCF) and Next Observation Carried Backward (NOCB) before starting the MCMC sequential imputation process.

Value

A list containing:

imputed_data	A three-dimensional array of imputed data with dimensions (npost / skip, N, p + 1), where N is the number of observations and p is the number of covariates. The last column represents the outcome Y.
posterior_sigma	(Only if outcome_model = "BMLM") A vector of posterior samples for the error standard deviation.
posterior_beta	(Only if outcome_model = "BMLM") A matrix of posterior samples for the regression coefficients.

Note

This function utilizes modified C++ code originally derived from the BART3 package (Bayesian Additive Regression Trees). The original package was developed by Rodney Sparapani and is licensed under GPL-2. Modifications were made by Jungang Zou, 2024.

References

For more information about the original BART3 package, see: https://github.com/rsparapa/bnptools/tree/master/BART3

Examples

```
data <- simulation_imputation(NNY = TRUE, NNX = TRUE, n_subject = 10, seed = 123)
BMTrees <- sequential_imputation(X = data$data_M[,3:5], Y = data$data_M$Y, Z = data$Z,
  subject_id = data$data_M$subject_id, type = c(0, 0, 0),
  outcome_model = "BMLM", binary_outcome = FALSE, model = "BMTrees", nburn = 0,
  npost = 1, skip = 1, verbose = FALSE, seed = 123)

# Access imputed data
dim(BMTrees$imputed_data)
```

---

simulation_imputation *Simulate Longitudinal Data with Missing Values for Imputation*

Description

Generates synthetic longitudinal data specifically designed to evaluate missing data imputation methods. The function creates a complex dataset with:

- **Time-varying covariates** with autoregressive structures and random effects.
- **Non-linear relationships** and interactions between covariates.
- **Mixed data types** (continuous and binary/logical).
- **Non-normal Distributions** (optional) for both random effects and residuals (Skew-t, t-distribution).
- **Missing Data Mechanisms**:
  - *Intermittent Missingness*: Generated via logistic models conditioned on outcomes and other covariates.
  - *Loss to Follow-up (LTFU)*: Simulates subject dropout starting from time point 4 based on values at time point 3.

Usage

```
simulation_imputation(NNY = TRUE, NNX = TRUE, n_subject = 1000, seed = NULL)
```

Arguments

NNY	A logical value. If TRUE, the outcome Y is generated using non-normal distributions (Skew-t random effects, t-distribution residuals). If FALSE, it uses standard Normal distributions. Default: TRUE.
NNX	A logical value. If TRUE, the covariates X_7 through X_12 are generated using non-normal distributions (Mixture models, Skew-t random effects). If FALSE, they use standard Normal distributions. Default: TRUE.
n_subject	An integer specifying the number of subjects. Default: 1000.
seed	An optional integer for setting the random seed to ensure reproducibility. Default: NULL.

Details

The simulation process creates 12 covariates (`X_1` to `X_12`):

- `X_1` to `X_6`: Base covariates generated via multivariate normal distributions with autoregressive sigma. `X_4, X_5, X_6` are converted to binary.
- `X_7` to `X_12`: Derived covariates dependent on the base set, involving non-linear transformations (squares, logs, interactions).

Missingness is introduced in two stages:

1. **Intermittent Missingness**: For variables `X_7` to `X_12`, missingness indicators are drawn from Bernoulli distributions where the probability depends on the outcome `Y` and other covariates.
2. **Dropout**: A "Loss to Follow-up" indicator is generated based on data at time point 3. If a subject drops out, all values for time points 4 and 5 become `NA`.

Value

A list containing the following components:

data_E A data frame of the **complete** data (ground truth) without any missing values.

data_M A data frame of the **incomplete** data, containing `NA`s introduced by intermittent missingness and dropout.

data_O A duplicate of `data_E` used internally for generating missingness probabilities.

Z A matrix of random predictors (intercept and time slopes) used in generation.

pair A matrix summarizing the missing data pattern (generated via `mice::md.pattern`).

Examples

```
# Simulate data with non-normal errors and random effects
sim_data <- simulation_imputation(NNY = TRUE, NNX = TRUE, n_subject = 10, seed = 123)

# View missing data pattern
sim_data$pair
```

---

simulation_imputation_LTFU

Simulate Longitudinal Data with Loss to Follow-up (LTFU) for Imputation

---

Description

Generates synthetic longitudinal data specifically designed to stress-test imputation methods against **Loss to Follow-up (Dropout)**. While it includes intermittent missingness, the parameters are tuned to simulate scenarios where subjects permanently leave the study based on their characteristics at specific time points.

Usage

```
simulation_imputation_LTFU(
  NNY = TRUE,
  NNX = TRUE,
  n_subject = 1000,
  seed = NULL
)
```

Arguments

NNY	A logical value. If TRUE, the outcome Y is generated using non-normal distributions (Skew-t random effects, t-distribution residuals). If FALSE, it uses standard Normal distributions. Default: TRUE.
NNX	A logical value. If TRUE, the covariates X_7 through X_12 are generated using non-normal distributions (Mixture models, Skew-t random effects). If FALSE, they use standard Normal distributions. Default: TRUE.
n_subject	An integer specifying the number of subjects. Default: 1000.
seed	An optional integer for setting the random seed to ensure reproducibility. Default: NULL.

Details

The data generation process mirrors simulation_imputation regarding covariate structure (time-varying, non-linear, mixed types), but utilizes specific coefficients to drive the missingness mechanisms:

1. Loss to Follow-up (LTFU): Dropout is simulated based on the subject's state at **time point 3**. A logistic model determines the probability of dropout using:

- The outcome Y at time 3.
- Covariates X_1, X_2, and X_3 at time 3.

If a subject is selected for LTFU, all their observations for **time points 4 and 5** are set to NA.

2. Intermittent Missingness: Variable-specific missingness is applied to X_7 through X_12 using logistic models that depend on the concurrent outcome Y, other covariates, and the previous value of the variable itself (autoregressive missingness).

Value

A list containing the following components:

data_E A data frame of the **complete** data (ground truth) without any missing values.

data_M A data frame of the **incomplete** data, containing NAs introduced by intermittent missingness and significant LTFU.

data_O A duplicate of data_E used internally for generating missingness probabilities.

Z A matrix of random predictors (intercept and time slopes) used in generation.

pair A matrix summarizing the missing data pattern (generated via mice::md.pattern).

Examples

```
lt_data <- simulation_imputation_LTFU(NNY = TRUE, NNX = TRUE, n_subject = 10, seed = 42)
```

`simulation_prediction_binary`
Simulate Binary Longitudinal Data for Prediction

Description

Generates synthetic longitudinal data with binary outcomes, designed for evaluating classification and prediction models. The function creates a latent continuous variable based on covariates and random effects, then converts it into binary outcomes using various link functions (corresponding to the `residual` argument).

Usage

```
simulation_prediction_binary(
  train_prop = 0.7,
  n_subject = 1000,
  n_obs_per_sub = 5,
  seed = NULL,
  nonlinear = FALSE,
  residual = c("normal", "logistic", "t3", "t2"),
  randeff = c("MVN", "MVN_mixture", "skewed_MVN", "MVT3", "MVT2")
)
```

Arguments

`train_prop`	A numeric value between 0 and 1 indicating the proportion of the population to be used for the training set. Default: `0.7`.
`n_subject`	An integer specifying the total number of subjects in the population. Default: `1000`.
`n_obs_per_sub`	An integer specifying the number of observations per subject. Default: `5`.
`seed`	An optional integer for setting the random seed to ensure reproducibility. Default: `NULL`.
`nonlinear`	A logical value. If `TRUE`, the latent variable is generated using a complex nonlinear function of the covariates. If `FALSE`, it is a linear combination. Default: `FALSE`.
`residual`	A character string specifying the link function (CDF) used to generate probabilities from the latent variable. This effectively acts as the error distribution assumption in a Generalized Linear Mixed Model (GLMM) context: • `"normal"`: Uses the standard normal CDF (Probit link). • `"logistic"`: Uses the logistic CDF (Logit link). • `"t3"`: Uses the Student's t (df=3) CDF.

- "t2": Uses the Student's t (df=2) CDF.

randeff — A character string specifying the distribution of the random effects added to the latent variable. Options are:

- "MVN": Multivariate Normal distribution.
- "MVN_mixture": Mixture of Multivariate Normal distributions.
- "skewed_MVN": Multivariate Skew-normal distribution.
- "MVT3": Multivariate t-distribution with 3 degrees of freedom.
- "MVT2": Multivariate t-distribution with 2 degrees of freedom.

Details

The function simulates a latent continuous variable Y^* based on fixed effects (linear or nonlinear X) and random effects (Z * Bi). This latent variable is scaled and then transformed into a probability p using the CDF specified by residual.

For the training set, the observed outcome Y_train is sampled from a Bernoulli distribution with probability p. For the testing set, the function returns the probability p itself (Y_test), allowing for precise evaluation of the model's ability to estimate propensity scores or risk.

Value

A list containing the following components:

subject_id_train A vector of subject IDs for the training set.

Z_train A matrix of random predictors (time/intercept) for the training set.

X_train A matrix of covariates for the training set.

Y_train A vector of **observed binary outcomes** (0 or 1) for the training set.

subject_id_test A vector of subject IDs for the testing set.

Z_test A matrix of random predictors for the testing set.

X_test A matrix of covariates for the testing set.

Y_test A vector of **true probabilities** for the testing set. These represent the ground truth propensity scores (0 to 1) used for evaluation.

X_pop A matrix of covariates for the entire population.

y_pop A vector of true probabilities for the entire population.

I A logical vector indicating which observations belong to the training set.

X_src Duplicate of X_train, provided for convenience.

Y_src Vector of true probabilities for the training set (unlike Y_train which is binary).

Examples

```
# Simulate data with logistic link (Logit) and mixture of normal random effects
sim_bin <- simulation_prediction_binary(
  train_prop = 0.7,
  n_subject = 500,
  residual = "logistic",
  randeff = "MVN_mixture",
  seed = 123
)
```

`simulation_prediction_conti` *Simulate Continuous Longitudinal Data for Prediction*

Description

Generates synthetic longitudinal data with continuous outcomes, specifically designed for evaluating prediction models. The function creates a population of subjects with correlated covariates and outcomes, then splits them into training and testing sets. It offers flexible options for simulating non-normal random effects (e.g., skewed, mixtures, t-distributions) and residuals, as well as nonlinear relationships.

Usage

```
simulation_prediction_conti(
  train_prop = 0.7,
  n_subject = 1000,
  n_obs_per_sub = 5,
  seed = NULL,
  nonlinear = FALSE,
  residual = c("normal", "normal_mixture", "skewed_normal", "t3", "t2"),
  randeff = c("MVN", "MVN_mixture", "skewed_MVN", "MVT3", "MVT2")
)
```

Arguments

`train_prop`	A numeric value between 0 and 1 indicating the proportion of the population to be used for the training set. Default: `0.7`.
`n_subject`	An integer specifying the total number of subjects in the population. Default: `1000`.
`n_obs_per_sub`	An integer specifying the number of observations per subject. Default: `5`.
`seed`	An optional integer for setting the random seed to ensure reproducibility. Default: `NULL`.
`nonlinear`	A logical value. If `TRUE`, the outcome `Y` is generated using a complex nonlinear function of the covariates. If `FALSE`, `Y` is a linear combination of covariates. Default: `FALSE`.
`residual`	A character string specifying the distribution of the residual errors added to the training outcome. Options are: • `"normal"`: Standard normal distribution. • `"normal_mixture"`: Mixture of two normal distributions. • `"skewed_normal"`: Skew-normal distribution. • `"t3"`: Student's t-distribution with 3 degrees of freedom. • `"t2"`: Student's t-distribution with 2 degrees of freedom.
`randeff`	A character string specifying the distribution of the random effects. Options are:

- "MVN": Multivariate Normal distribution.
- "MVN_mixture": Mixture of Multivariate Normal distributions.
- "skewed_MVN": Multivariate Skew-normal distribution.
- "MVT3": Multivariate t-distribution with 3 degrees of freedom.
- "MVT2": Multivariate t-distribution with 2 degrees of freedom.

Details

The function first simulates correlated covariates X using a multivariate normal distribution, adding subject-specific random variations. The outcome Y is then constructed based on X (either linearly or nonlinearly) and combined with random effects Z * Bi drawn from the specified randeff distribution.

The data is split into training and testing sets based on train_prop. Crucially, residual noise (specified by residual) is added **only** to Y_train. The Y_test values represent the conditional mean (Fixed + Random Effects) and serve as the ground truth for prediction tasks aiming to recover the de-noised signal.

Value

A list containing the following components:

subject_id_train A vector of subject IDs for the training set.

Z_train A matrix of random predictors (time/intercept) for the training set.

X_train A matrix of covariates for the training set.

Y_train A vector of observed outcomes for the training set (Signal + Random Effects + Residual Error).

subject_id_test A vector of subject IDs for the testing set.

Z_test A matrix of random predictors for the testing set.

X_test A matrix of covariates for the testing set.

Y_test A vector of "true" outcomes for the testing set (Signal + Random Effects), without residual error.

X_pop A matrix of covariates for the entire population.

y_pop A vector of "true" outcomes for the entire population (Signal + Random Effects).

I A logical vector indicating which observations belong to the training set.

X_src Duplicate of X_train, provided for convenience.

Y_src Duplicate of Y_train, provided for convenience.

Examples

```
sim_data <- simulation_prediction_conti(
  train_prop = 0.7,
  n_subject = 200,
  n_obs_per_sub = 5,
  nonlinear = TRUE,
  residual = "normal",
```

```
  randeff = "skewed_MVN",
  seed = 123
)
```

Index