Package: HHBayes (via r-universe)

October 29, 2026

Title Bayesian Household Transmission Modeling with 'Stan'

Version 0.1.1

Description Provides a streamlined pipeline to simulate household infection dynamics, estimate transmission parameters, and visualize epidemic timelines. Uses a Bayesian approach with 'Stan' that models transmission probability as a function of viral load, seasonality and infectivity, multiple infection episodes (reinfections), and waning immunity modeling. Li et al. (2026) <doi:10.64898/2026.04.01.26349903>).

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Biarch true

Depends R (>= 3.4.0)

Imports methods, Rcpp (>= 0.12.0), rstan (>= 2.18.1), rstantools (>= 2.4.0), dplyr, ggplot2, tidyr, ggpubr, scales, deSolve, rlang

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

SystemRequirements GNU make

Suggests knitr, rmarkdown, RcppParallel (>= 5.0.1), cowplot

VignetteBuilder knitr

URL https://github.com/keli5734/HHBayes

BugReports https://github.com/keli5734/HHBayes/issues

NeedsCompilation yes

Author Ke Li [aut, cre] (ORCID: <https://orcid.org/0000-0002-1252-670X>), Yiren Hou [aut] (ORCID: <https://orcid.org/0009-0005-0422-4268>)

Maintainer Ke Li <ke.li.kl662@yale.edu>

Config/pak/sysreqs make

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

Date/Publication 2026-04-10 15:10:28 UTC

RemoteUrl https://github.com/cran/HHBayes

RemoteRef HEAD

RemoteSha 57eb35cb5b446769c0de7d3d1a4adfd95b4bdd7b

Contents

calculate_sar_quick . . . 2
calculate_sar_robust . . . 3
fill_missing_viral_data . . . 4
fit_household_model . . . 5
plot_covariate_effects . . . 7
plot_epidemic_curve . . . 7
plot_household_timeline . . . 8
plot_posterior_distributions . . . 10
prepare_stan_data . . . 11
reconstruct_transmission_chains . . . 13
simulate_multiple_households_comm . . . 14
summarize_attack_rates . . . 17

Index **18**

`calculate_sar_quick` *Quick Secondary Attack Rate (SAR) Calculator*

Description

A drop-in replacement for a legacy `calculate_sar_quick()` function. Calls `calculate_sar_robust` with recommended defaults: `generation = "strict_secondary"` and `pooling = "mean_of_hh"`.

Usage

```
calculate_sar_quick(sim_result)
```

Arguments

`sim_result`	A named list containing a `hh_df` data frame. See `calculate_sar_robust` for details.

Value

See `calculate_sar_robust`.

See Also

`calculate_sar_robust`

Examples

```
# Quick example
sim <- simulate_multiple_households_comm(
  n_households = 2,
  start_date = "2024-07-01",
  end_date = "2024-07-30",
  seed = 42
)
sim$hh_df$infector_id <- NA
sim$hh_df$susceptible_at_start <- TRUE
sar <- calculate_sar_quick(sim)
sar$overall
```

`calculate_sar_robust` *Calculate Secondary Attack Rate (SAR)*

Description

Computes household secondary attack rates (SAR) from simulation output, with fixes for generation conflation, co-index ambiguity, inconsistent pooling, community importation, infinite first-infection time, and susceptibility at study entry.

Usage

```
calculate_sar_robust(
  sim_result,
  generation = c("strict_secondary", "all_subsequent"),
  pooling = c("mean_of_hh", "pooled"),
  alpha_flag = TRUE,
  verbose = TRUE
)
```

Arguments

`sim_result`	A named list containing at minimum a `hh_df` data frame with household simulation output. Required columns: `hh_id`, `person_id`, `role`, `infection_time`, `infector_id`, `susceptible_at_start`.
`generation`	Character string controlling which infections count toward the SAR numerator. One of: `"strict_secondary"` (default) Only generation-2 infections (i.e., directly infected by the index case). `"all_subsequent"` All non-index infections, regardless of generation (original behavior).
`pooling`	Character string controlling how household-level SARs are aggregated. One of: `"mean_of_hh"` (default) Each household receives equal weight (recommended). `"pooled"` Individual-level pooling; larger households receive more weight.

alpha_flag	Logical. If TRUE (default), emits a warning when post-index infections with no recorded within-household infector are detected, suggesting possible community importation.
verbose	Logical. If TRUE (default), prints diagnostic messages via rlang::inform().

Value

A named list with two elements:

overall Numeric scalar: the overall household SAR.

by_age Data frame of role-specific SARs, including n_households, sar, sar_sd, sar_se, sar_lo, sar_hi, n_secondary, n_suscept, and overall_sar.

Examples

```
# Quick example with 2 households
sim <- simulate_multiple_households_comm(
  n_households = 2,
  start_date = "2024-07-01",
  end_date = "2024-07-30",
  seed = 42
)
sim$hh_df$infector_id <- NA
sim$hh_df$susceptible_at_start <- TRUE
sar <- calculate_sar_robust(sim, generation = "all_subsequent")
sar$overall
```

---

fill_missing_viral_data

Fill Missing Viral Data (Cycle Threshold or Log10) Based on Episode Start

---

Description

This function imputes missing viral data during confirmed episodes using theoretical trajectories defined by the parameters.

Usage

```
fill_missing_viral_data(
  df,
  viral_col_name,
  type = c("ct_value", "log10"),
  params_list,
  detection_limit
)
```

Arguments

df	Data frame containing person_id, episode_id, date, role_name, and the viral column.
viral_col_name	String. Name of the column containing viral data (e.g. "ct_value").
type	String. Either "ct_value" or "log10".
params_list	List of parameters for the curves (role-specific).
detection_limit	Numeric. Value to assign for non-infected days (e.g. 45 for Ct, 0 for log10).

Value

The original data frame with NAs in the viral column filled.

Examples

```
# Create a sample data frame with some missing Ct values
df <- data.frame(
  person_id = rep(1, 10),
  episode_id = c(rep(0, 3), rep(1, 5), rep(0, 2)),
  date = seq(as.Date("2024-07-01"), by = "day", length.out = 10),
  role_name = "adult",
  ct_value = c(45, 45, 45, NA, 30, 28, NA, 35, 45, 45)
)

# Define Ct parameters for imputation
ct_params <- list(
  adult = list(Cpeak = 33, r = 1.5, d = 1.2, t_peak = 5)
)

# Fill missing values
df_filled <- fill_missing_viral_data(
  df = df,
  viral_col_name = "ct_value",
  type = "ct_value",
  params_list = ct_params,
  detection_limit = 45
)
df_filled$ct_value
```

---

fit_household_model *Fit Household Transmission Model*

Description

Fits the compiled Stan model to the prepared household data.

Usage

```
fit_household_model(
  stan_data,
  iter = 2000,
  chains = 4,
  warmup = 1000,
  init_fun = NULL,
  ...
)
```

Arguments

stan_data	A list of data formatted by prepare_stan_data.
iter	Integer. Number of iterations per chain (including warmup). Defaults to 2000.
chains	Integer. Number of Markov chains. Defaults to 4.
warmup	Integer. Number of warmup iterations. Defaults to 1000.
init_fun	Function or List. Initial values for the sampler. If NULL, uses robust defaults tailored for this model.
...	Additional arguments passed to rstan::sampling (e.g., cores, seed).

Value

A stanfit object containing the posterior samples.

Examples

```
# Show default initial values (does not require Stan compilation)
sim <- simulate_multiple_households_comm(
  n_households = 2,
  start_date = "2024-07-01",
  end_date = "2024-07-30",
  seed = 42
)
stan_input <- prepare_stan_data(
  df_clean = sim$diagnostic_df,
  study_start_date = as.Date("2024-07-01"),
  study_end_date = as.Date("2024-07-30"),
  use_vl_data = TRUE,
  seed = 123
)
# Inspect data dimensions passed to the model
stan_input$N   # number of person-episodes
stan_input$H   # number of households
stan_input$R   # number of roles
```

plot_covariate_effects *Plot Covariate Coefficients (Forest Plot)*

Description

Generates a forest plot showing the estimated log-linear covariate effects on susceptibility and infectivity, with median and 95 percent credible intervals.

Usage

```
plot_covariate_effects(fit, stan_data)
```

Arguments

fit	A stanfit object returned by fit_household_model.
stan_data	A list of data formatted by prepare_stan_data.

Value

A ggplot object displaying covariate effect estimates.

Examples

```
# This function requires a fitted 'Stan' model with covariates.
# Prepare input data with a covariate (runs in seconds):
sim <- simulate_multiple_households_comm(
  n_households = 2,
  start_date = "2024-07-01",
  end_date = "2024-07-30",
  seed = 42
)
names(sim$hh_df)
```

plot_epidemic_curve *Plot Dual-Axis Epidemic Curve (Binned)*

Description

Overlays a stacked bar chart of simulated infections with a line chart of surveillance data. Both datasets are aggregated by bin_width days.

Usage

```
plot_epidemic_curve(
  sim_result,
  surveillance_df,
  start_date_str = "2024-07-01",
  bin_width = 7
)
```

Arguments

sim_result	Output object from simulate_multiple_households_comm.
surveillance_df	Data frame with date and cases columns.
start_date_str	String. Start date of the simulation.
bin_width	Integer. Aggregation window in days (default 7).

Value

A ggplot object.

Examples

```
# Quick example with minimal data
surv_df <- data.frame(
  date = seq(as.Date("2024-07-01"), as.Date("2024-07-30"), by = "day"),
  cases = rpois(30, lambda = 3)
)
sim <- simulate_multiple_households_comm(
  n_households = 2,
  surveillance_df = surv_df,
  start_date = "2024-07-01",
  end_date = "2024-07-30",
  seed = 42
)
plot_epidemic_curve(sim, surv_df, start_date_str = "2024-07-01")
```

---

plot_household_timeline

Plot Household Timeline with Person-Centric Reinfections

Description

Displays a timeline for a single household showing each member's infection events and the estimated transmission links (with probabilities) between them.

Usage

```
plot_household_timeline(
  trans_df,
  stan_data,
  target_hh_id,
  start_date_str = "2024-10-08",
  prob_cutoff = 0,
  plot_width = 11,
  plot_height = 7
)
```

Arguments

trans_df	A data frame of transmission events (e.g., from extract_transmission_pairs).
stan_data	A list of data formatted by prepare_stan_data.
target_hh_id	Integer. The household ID to plot.
start_date_str	Character. Study start date in "YYYY-MM-DD" format. Defaults to "2024-10-08".
prob_cutoff	Numeric. Minimum transmission probability to display. Defaults to 0.
plot_width	Numeric. Width of the plot in inches. Defaults to 11.
plot_height	Numeric. Height of the plot in inches. Defaults to 7.

Value

A ggplot object displaying the household infection timeline.

Examples

```
# Reconstruct chains and plot a household timeline (no 'Stan' fit needed)
sim <- simulate_multiple_households_comm(
  n_households = 2,
  start_date = "2024-07-01",
  end_date = "2024-07-30",
  seed = 42
)
stan_input <- prepare_stan_data(
  df_clean = sim$diagnostic_df,
  study_start_date = as.Date("2024-07-01"),
  study_end_date = as.Date("2024-07-30"),
  use_vl_data = TRUE
)
sim_params <- list(
  beta1 = 8e-3, beta2 = 8e-3, alpha_comm = 5e-4,
  gen_shape = 3, gen_rate = 1,
  phi = c(1, 4, 5, 1), kappa = c(1, 1, 1.2, 1)
)
chains <- reconstruct_transmission_chains(
  stan_data = stan_input,
```

```
  sim_params = sim_params
)
if (nrow(chains) > 0) {
  plot_household_timeline(
    trans_df = chains,
    stan_data = stan_input,
    target_hh_id = 1,
    start_date_str = "2024-07-01"
  )
}
```

`plot_posterior_distributions` *Plot Posterior Distributions (Phi and Kappa)*

Description

Creates violin plots of the posterior distributions for role-specific susceptibility (phi) and infectivity (kappa) parameters on the log10 scale.

Usage

```
plot_posterior_distributions(fit)
```

Arguments

`fit`	A `stanfit` object returned by `fit_household_model`.

Value

A `ggplot` object displaying posterior violin plots.

Examples

```
# This function requires a fitted 'Stan' model object.
# Prepare input data (runs in seconds):
sim <- simulate_multiple_households_comm(
  n_households = 2,
  start_date = "2024-07-01",
  end_date = "2024-07-30",
  seed = 42
)
stan_input <- prepare_stan_data(
  df_clean = sim$diagnostic_df,
  study_start_date = as.Date("2024-07-01"),
  study_end_date = as.Date("2024-07-30"),
  use_vl_data = TRUE
)
names(stan_input)
```

prepare_stan_data *Prepare Data for Stan Model*

Description

Transforms a diagnostic data frame into the structured list that the Stan household transmission model expects. Handles column renaming, infection window imputation, viral load gap-filling, covariate matrix construction, contact matrix generation, and prior specification.

Usage

```
prepare_stan_data(
  df_clean,
  surveillance_df = NULL,
  role_levels = c("adult", "infant", "toddler", "elderly"),
  study_start_date = as.Date("2024-07-01"),
  study_end_date = as.Date("2025-07-01"),
  seasonal_forcing_list = NULL,
  use_vl_data = TRUE,
  use_curve_logic = FALSE,
  covariates_susceptibility = NULL,
  covariates_infectivity = NULL,
  model_type = "empirical",
  ODE_params_list = NULL,
  delta = 0,
  role_mixing_matrix = NULL,
  recovery_params = NULL,
  imputation_params = NULL,
  priors = list(),
  seed = 123
)
```

Arguments

df_clean	Data frame with observation data (must contain episode_id from simulation or data).
surveillance_df	Data frame with columns date and cases.
role_levels	Character vector. Role categories to use. Defaults to c("adult", "infant", "toddler", "elderly").
study_start_date	Date. Start date of the study period. Defaults to 2024-07-01.
study_end_date	Date. End date of the study period. Defaults to 2025-07-01.
seasonal_forcing_list	List of numeric vectors (one per role) for seasonal forcing. NULL for no forcing.
use_vl_data	Logical. Whether to include viral load data. Defaults to TRUE.

use_curve_logic
: Logical. Whether to use the Gamma generation interval curve as a fallback for infectiousness when use_vl_data = FALSE. When use_vl_data = TRUE this argument is ignored. Defaults to FALSE.

covariates_susceptibility
: Vector of column names to use as covariates for susceptibility.

covariates_infectivity
: Vector of column names to use as covariates for infectivity.

model_type
: String, either "empirical" or "ODE" (ordinary differential equation).

ODE_params_list
: List of ordinary differential equation (ODE) parameters (beta, delta, etc.) by role.

delta
: Numeric. Household size scaling parameter. When delta > 0, transmission rates are scaled by (1/max(household_size, 1))^delta. Should match the value used in simulation. Defaults to 0 (no scaling).

role_mixing_matrix
: 4x4 Matrix defining contact weights between roles.

recovery_params
: List of Gamma parameters (shape, scale) for the immunity tail by role.

imputation_params
: List of parameters for mechanistic viral curves (Cpeak, r, d, t_peak) by role.

priors
: List of flexible priors (dist, params).

seed
: Integer. Random seed for reproducibility. Defaults to 123.

Value

A named list formatted for input to the Stan model.

Examples

```
# Quick example with 2 households over a short period
sim <- simulate_multiple_households_comm(
  n_households = 2,
  start_date = "2024-07-01",
  end_date = "2024-07-30",
  seed = 42
)
stan_input <- prepare_stan_data(
  df_clean = sim$diagnostic_df,
  study_start_date = as.Date("2024-07-01"),
  study_end_date = as.Date("2024-07-30"),
  use_vl_data = TRUE,
  seed = 123
)
names(stan_input)
stan_input$N
stan_input$H
```

`reconstruct_transmission_chains` *Reconstruct Transmission Chains*

Description

Saves ALL potential infectors for each infection episode. Works in two modes: estimation mode (supply a `fit` object) or simulation mode (supply a `sim_params` list of known ground-truth values).

Usage

```
reconstruct_transmission_chains(
  fit = NULL,
  stan_data,
  sim_params = NULL,
  min_prob_threshold = 0.01
)
```

Arguments

`fit`	A `stanfit` object from `rstan`. Required unless `sim_params` is provided.
`stan_data`	The list of data passed to Stan (or used in simulation).
`sim_params`	A named list of ground-truth parameters for simulation mode. Required fields: `beta1`, `beta2`, `alpha_comm`, `gen_shape`, `gen_rate`, `phi`, `kappa`. Optional: `vl_midpoint`, `vl_slope`, `beta_susc`, `beta_inf`.
`min_prob_threshold`	Minimum probability to retain a transmission link. Default `0.01`.

Value

A data frame with columns: `target`, `hh_id`, `day`, `source`, `prob`.

Examples

```
# Reconstruct chains using known simulation parameters (no 'Stan' fit needed)
sim <- simulate_multiple_households_comm(
  n_households = 2,
  start_date = "2024-07-01",
  end_date = "2024-07-30",
  seed = 42
)
stan_input <- prepare_stan_data(
  df_clean = sim$diagnostic_df,
  study_start_date = as.Date("2024-07-01"),
  study_end_date = as.Date("2024-07-30"),
  use_vl_data = TRUE
)
sim_params <- list(
```

```
  beta1 = 8e-3, beta2 = 8e-3, alpha_comm = 5e-4,
  gen_shape = 3, gen_rate = 1,
  phi = c(1, 4, 5, 1), kappa = c(1, 1, 1.2, 1)
)
chains <- reconstruct_transmission_chains(
  stan_data = stan_input,
  sim_params = sim_params,
  min_prob_threshold = 0.01
)
head(chains)
```

`simulate_multiple_households_comm` *Simulate Household Transmission*

Description

Simulates infection dynamics across multiple households with community forcing.

Usage

```
simulate_multiple_households_comm(
  n_households = 50,
  surveillance_df = NULL,
  start_date = "2024-07-01",
  end_date = "2025-06-30",
  alpha_comm_by_role = 5e-04,
  beta1 = 0.008,
  beta2 = 0.008,
  delta = 0,
  phi_by_role = c(adult = 1, infant = 4, toddler = 5, elderly = 1),
  kappa_by_role = c(adult = 1, infant = 1, toddler = 1.2, elderly = 1),
  infectious_shape = 3,
  infectious_scale = 1,
  waning_shape = 16,
  waning_scale = 10,
  peak_day = 1,
  width = 4,
  verbose = FALSE,
  seasonal_forcing_list = NULL,
  detect_threshold_log10 = 1e-06,
  detect_threshold_Ct = 40,
  surveillance_interval = 1,
  test_daily = FALSE,
  viral_testing = "viral load",
  V_ref = 3,
  V_rho = 2.5,
  Ct_50 = 40,
```

```
  Ct_delta = 2,
  VL_params_list = NULL,
  Ct_params_list = NULL,
  household_profile_list = NULL,
  perfect_detection = TRUE,
  contact_mat = NULL,
  role_mixing_matrix = NULL,
  model_type = "empirical",
  ODE_params_list = NULL,
  covariates_config = NULL,
  seed = NULL,
  max_infections = Inf
)
```

Arguments

n_households	Integer. Number of households to simulate. Defaults to 50.
surveillance_df	Data frame with columns date and cases for community forcing. NULL for none.
start_date	Character. Simulation start date ("YYYY-MM-DD"). Defaults to "2024-07-01".
end_date	Character. Simulation end date ("YYYY-MM-DD"). Defaults to "2025-06-30".
alpha_comm_by_role	Numeric or named vector. Community acquisition rate by role. Defaults to 5e-4.
beta1	Numeric. Within-household transmission rate (pathway 1). Defaults to 8e-3.
beta2	Numeric. Within-household transmission rate (pathway 2). Defaults to 8e-3.
delta	Numeric. Co-infection parameter. Defaults to 0.
phi_by_role	Named numeric vector. Susceptibility multipliers by role.
kappa_by_role	Named numeric vector. Infectivity multipliers by role.
infectious_shape	Numeric. Shape parameter for the Gamma infectious period. Defaults to 3.
infectious_scale	Numeric. Scale parameter for the Gamma infectious period. Defaults to 1.
waning_shape	Numeric. Shape parameter for the Gamma immunity waning period. Defaults to 16.
waning_scale	Numeric. Scale parameter for the Gamma immunity waning period. Defaults to 10.
peak_day	Numeric. Day of peak infectiousness. Defaults to 1.
width	Numeric. Width of the infectiousness peak. Defaults to 4.
verbose	Logical. Print progress messages. Defaults to FALSE.
seasonal_forcing_list	List of numeric vectors for seasonal forcing by role. NULL for none.
detect_threshold_log10	Numeric. Detection threshold on log10 viral load scale. Defaults to 1e-6.

`detect_threshold_Ct`
: Numeric. Detection threshold on cycle threshold (Ct) scale. Defaults to 99.

`surveillance_interval`
: Integer. Days between surveillance tests. Defaults to 1.

`test_daily`
: Logical. Whether to test daily. Defaults to FALSE.

`viral_testing`
: Character. Type of viral testing: "viral load" or "Ct" (cycle threshold). Defaults to "viral load".

`V_ref`
: Numeric. Reference viral load for transmission scaling. Defaults to 3.0.

`V_rho`
: Numeric. Exponent for viral load scaling. Defaults to 2.5.

`Ct_50`
: Numeric. Cycle threshold (Ct) value at 50 percent detection probability. Defaults to 40.

`Ct_delta`
: Numeric. Steepness of the Ct detection curve. Defaults to 2.

`VL_params_list`
: List of viral load trajectory parameters by role. NULL for defaults.

`Ct_params_list`
: List of cycle threshold (Ct) trajectory parameters by role. NULL for defaults.

`household_profile_list`
: List defining household composition probabilities. NULL for defaults.

`perfect_detection`
: Logical. Whether detection is perfect. Defaults to TRUE.

`contact_mat`
: Matrix. Custom contact matrix between individuals. NULL for default.

`role_mixing_matrix`
: Matrix. Contact weights between roles. 4x4 Matrix where element (i,j) represents contact weight FROM role j TO role i. For asymmetric patterns, role_mixing_matrix(adult, infant) != role_mixing_matrix(infant, adult). Use dimnames(role_mixing_matrix) <- list(role_levels, role_levels). NULL for default.

`model_type`
: Character. Either "empirical" or "ODE" (ordinary differential equation). Defaults to "empirical".

`ODE_params_list`
: List of ordinary differential equation (ODE) parameters by role. NULL for defaults.

`covariates_config`
: List defining covariate configurations. NULL for none.

`seed`
: Integer. Random seed. NULL for no seed.

`max_infections`
: Numeric. Maximum infections per person. Defaults to Inf.

Value

A list with two elements: hh_df (household-level results) and diagnostic_df (testing results).

Examples

```
# Quick simulation with 2 households over a short period
sim <- simulate_multiple_households_comm(
  n_households = 2,
  start_date = "2024-07-01",
  end_date = "2024-07-30",
```

```
  seed = 42
)
head(sim$hh_df)
head(sim$diagnostic_df)
```

---

summarize_attack_rates *Summarize Attack Rates and Reinfections*

Description

Calculates the Primary Attack Rate (proportion of people infected at least once) and a separate summary of Reinfections (secondary episodes).

Usage

```
summarize_attack_rates(sim_result)
```

Arguments

sim_result	A list object returned by simulate_multiple_households_comm.

Value

A list containing four data frames:

primary_overall Overall primary attack rate statistics.

primary_by_role Primary attack rate stratified by age group.

reinf_overall Overall summary of reinfection counts and rates.

reinf_by_role Reinfection counts stratified by age group.

Examples

```
# Quick example with 2 households
sim <- simulate_multiple_households_comm(
  n_households = 2,
  start_date = "2024-07-01",
  end_date = "2024-07-30",
  seed = 42
)
ar <- summarize_attack_rates(sim)
ar$primary_overall
```

Index

calculate_sar_quick, 2
calculate_sar_robust, 2, 3

fill_missing_viral_data, 4
fit_household_model, 5

plot_covariate_effects, 7
plot_epidemic_curve, 7
plot_household_timeline, 8
plot_posterior_distributions, 10
prepare_stan_data, 11

reconstruct_transmission_chains, 13

simulate_multiple_households_comm, 14
summarize_attack_rates, 17