Package: mitey (via r-universe)

November 9, 2026

Title Serial Interval and Case Reproduction Number Estimation

Version 0.4.0

Author Kylie Ainslie [aut, cre, cph]
(<https://orcid.org/0000-0001-5419-7206>)

Maintainer Kylie Ainslie <ainslie.kylie@gmail.com>

Description Provides methods to estimate serial intervals and time-varying case reproduction numbers from infectious disease outbreak data. Serial intervals measure the time between symptom onset in linked transmission pairs, while case reproduction numbers quantify how many secondary cases each infected individual generates over time. These parameters are essential for understanding transmission dynamics, evaluating control measures, and informing public health responses. The package implements the maximum likelihood framework from Vink et al. (2014) <doi:10.1093/aje/kwu209> for serial interval estimation and the retrospective method from Wallinga & Lipsitch (2007) <doi:10.1098/rspb.2006.3754> for reproduction number estimation. Originally developed for scabies transmission analysis but applicable to other infectious diseases including influenza, COVID-19, and emerging pathogens. Designed for epidemiologists, public health researchers, and infectious disease modelers working with outbreak surveillance data.

Date 2026-06-04

License EUPL-1.2

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 8.0.0

Imports fdrtool, ggplot2, stats

URL https://github.com/kylieainslie/mitey,
https://kylieainslie.github.io/mitey/

BugReports https://github.com/kylieainslie/mitey/issues

Suggests BH, brms, patchwork, broom, cowplot, dplyr, EpiLPS, extraDistr, flextable, forcats, ftExtra, ggridges, gt, here, ISOweek, knitr, lubridate, officer, openxlsx, outbreaks, purrr, RcppEigen, rmarkdown, stringr, testthat (>= 3.0.0), tidybayes, tidyr, viridis, zoo

Config/testthat/edition 3

VignetteBuilder knitr

Depends R (>= 4.0.0)

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

Date/Publication 2026-06-04 22:36:54 UTC

RemoteUrl https://github.com/kylieainslie/mitey

RemoteRef HEAD

RemoteSha 62b60ee9a1ca140daa014d242a64d139eb7a8495

Contents

compare_n_routes 2
generate_synthetic_epidemic 4
plot_si_fit 6
plot_si_fit_result 8
si_estim 9
smooth_estimates 11
wallinga_lipsitch 12

Index **16**

`compare_n_routes` *Compare Serial Interval Models Across Different Numbers of Transmission Routes*

Description

Fits the serial interval mixture model for each number of transmission routes from 2 up to `n_routes_max`, computes model selection criteria (AIC and BIC) for each fit, and prints a summary table to help identify the optimal number of routes.

Usage

```
compare_n_routes(dat, n_routes_max = 8L, dist = "normal", n = 50, ...)
```

Arguments

`dat`	numeric vector; index case-to-case (ICC) intervals in days.
`n_routes_max`	integer; maximum number of transmission routes to evaluate. Models are fitted for `n_routes` from 2 to `n_routes_max`. Defaults to `8L`.
`dist`	character; assumed parametric family for the serial interval distribution. Must be either `"normal"` (default) or `"gamma"`.
`n`	integer; number of EM algorithm iterations passed to `si_estim`. Defaults to 50.
`...`	additional arguments passed to `si_estim`.

Details

For each value of `n_routes` in `2:n_routes_max`, the function:

1. Calls `si_estim` to fit the mixture model and retrieve the log-likelihood and estimated weights.
2. Counts the number of free parameters p:
   - Normal distribution: $p = 2 + (2 \times n_routes - 2)$
   - Gamma distribution: $p = 2 + (n_routes - 1)$
3. Computes AIC and BIC:

$$AIC = -2\ell + 2p$$

$$BIC = -2\ell + p\log(n)$$

4. Prints the fitted mixture plot via `plot_si_fit_result`.

The model with the lowest BIC (or AIC) is reported as the best-fitting number of routes.

Value

A data frame (returned invisibly) with one row per value of `n_routes` and the following columns:

n_routes Number of transmission routes fitted.

mu Estimated mean of the serial interval (days).

sigma Estimated standard deviation of the serial interval (days).

loglik Log-likelihood of the fitted model.

aic Akaike Information Criterion.

bic Bayesian Information Criterion.

n_params Number of free parameters in the model.

converged Logical; whether the EM algorithm converged.

iterations Number of EM iterations performed.

A summary table and the best `n_routes` according to BIC and AIC are printed as side effects.

See Also

`si_estim`, `plot_si_fit_result`

Examples

```
set.seed(42)
icc_data <- c(
  abs(rnorm(30, mean = 0,  sd = 2)),
  rnorm(80, mean = 12, sd = 3),
  rnorm(30, mean = 24, sd = 4)
)
icc_data <- round(pmax(icc_data, 0))

# Compare 2 to 6 routes with normal distribution
results <- compare_n_routes(icc_data, n_routes_max = 6, dist = "normal", n = 50)

# Compare using gamma distribution
results_gam <- compare_n_routes(icc_data, n_routes_max = 5, dist = "gamma", n = 50)
```

---

generate_synthetic_epidemic

Generate Synthetic Epidemic Data Using the Renewal Equation

Description

Simulates epidemic incidence data with known reproduction numbers using the renewal equation framework. This function is useful for testing and validating reproduction number estimation methods, as it generates synthetic outbreaks with ground truth R values that can be compared against estimated values.

Usage

```
generate_synthetic_epidemic(
  true_r,
  si_mean,
  si_sd,
  si_dist = "gamma",
  initial_cases = 10
)
```

Arguments

true_r	numeric vector; the true time-varying reproduction numbers. The length of this vector determines the number of days in the simulated epidemic
si_mean	numeric; the mean of the serial interval distribution in days
si_sd	numeric; the standard deviation of the serial interval distribution in days
si_dist	character; the distribution family for the serial interval. Must be either "gamma" (default) or "normal". Gamma is recommended as it naturally restricts to positive values
initial_cases	integer; the number of cases on the first day of the epidemic. Defaults to 10

Details

The function implements the discrete renewal equation:

$$\lambda_t = \sum_{s=1}^{t-1} I_s \cdot R_s \cdot w(t-s)$$

where λ_t is the expected number of new infections at time t, I_s is the incidence at time s, R_s is the reproduction number at time s, and $w(t-s)$ is the probability mass function of the serial interval distribution for interval $t-s$.

New cases at each time point are drawn from a Poisson distribution with mean λ_t, introducing realistic stochastic variation while maintaining the specified reproduction number trajectory.

The serial interval distribution is truncated at the 99th percentile to avoid computationally expensive calculations for very long tails. For the normal distribution, the probability mass function is normalized to ensure proper probability weights.

This function is particularly useful for:

- Validating reproduction number estimation methods
- Testing the performance of epidemiological models
- Generating realistic epidemic scenarios for research
- Creating training data for machine learning approaches

Value

A data frame with three columns:

- `date`: Date sequence starting from "2023-01-01"
- `true_r`: The input reproduction number values
- `incidence`: Simulated daily case counts

References

Fraser C (2007). Estimating individual and household reproduction numbers in an emerging epidemic. PLoS One, 2(8), e758.

Cori A, Ferguson NM, Fraser C, Cauchemez S (2013). A new framework and software to estimate time-varying reproduction numbers during epidemics. American Journal of Epidemiology, 178(9), 1505-1512.

See Also

`wallinga_lipsitch` for reproduction number estimation methods that can be applied to the generated data

Examples

```
# Simple epidemic with constant R = 1.5
constant_r <- rep(1.5, 30)
epidemic1 <- generate_synthetic_epidemic(
  true_r = constant_r,
  si_mean = 7,
  si_sd = 3,
  si_dist = "gamma"
)
head(epidemic1)

# Epidemic with declining R (e.g., intervention effect)
declining_r <- seq(2.0, 0.5, length.out = 50)
epidemic2 <- generate_synthetic_epidemic(
  true_r = declining_r,
  si_mean = 5,
  si_sd = 2,
  si_dist = "gamma",
  initial_cases = 5
)

# Epidemic with seasonal pattern
days <- 100
seasonal_r <- 1.2 + 0.5 * sin(2 * pi * (1:days) / 365 * 7) # Weekly seasonality
epidemic3 <- generate_synthetic_epidemic(
  true_r = seasonal_r,
  si_mean = 6,
  si_sd = 2.5,
  si_dist = "normal"
)

# Plot the results
if (require(ggplot2)) {
  library(ggplot2)
  ggplot(epidemic1, aes(x = date)) +
    geom_col(aes(y = incidence), alpha = 0.7) +
    geom_line(aes(y = true_r * 10), color = "red") +
    labs(title = "Synthetic Epidemic",
         y = "Daily Incidence",
         subtitle = "Red line: True R × 10")
}
```

`plot_si_fit` *Visualize Serial Interval Distribution Fit to Outbreak Data*

Description

Creates a diagnostic plot showing the fitted serial interval mixture distribution overlaid on a histogram of observed index case-to-case (ICC) intervals from outbreak data.

Usage

```
plot_si_fit(
  dat,
  mean,
  sd,
  weights,
  dist = "normal",
  scaling_factor = 1,
  n_routes = 4L
)
```

Arguments

dat	numeric vector; the index case-to-case (ICC) intervals in days.
mean	numeric; the estimated mean of the serial interval distribution in days.
sd	numeric; the estimated standard deviation of the serial interval distribution in days.
weights	numeric vector of length n_routes - 1; the estimated weights for each transmission route component, starting from Co-primary. The last route weight is derived internally as 1 - sum(weights).
dist	character; the distribution family. Must be either "normal" (default) or "gamma".
scaling_factor	numeric; multiplicative factor to adjust the height of the fitted density curve. Defaults to 1.
n_routes	integer; number of transmission routes modelled. Must be >= 2. Defaults to 4. Must match the value used in `si_estim()`.

Value

A `ggplot2` object.

References

Vink MA, Bootsma MCJ, Wallinga J (2014). Serial intervals of respiratory infectious diseases: A systematic review and analysis. American Journal of Epidemiology, 180(9), 865-875.

See Also

`si_estim` for serial interval estimation, `f_norm` and `f_gam` for the underlying mixture distribution functions

Examples

```
# Example 1: 4 routes, normal distribution
set.seed(123)
icc_data <- c(
  rnorm(20, mean = 0, sd = 2),
  rnorm(50, mean = 12, sd = 3),
  rnorm(20, mean = 24, sd = 4)
```

```
)
icc_data <- round(pmax(icc_data, 0))

plot_si_fit(
  dat = icc_data,
  mean = 12.5,
  sd = 3.2,
  weights = c(0.2, 0.6, 0.15, 0.05),
  dist = "normal",
  n_routes = 4
)

# Example 2: 5 routes, gamma distribution
plot_si_fit(
  dat = icc_data,
  mean = 12.0,
  sd = 3.5,
  weights = c(0.25, 0.65, 0.05, 0.03, 0.02),
  dist = "gamma",
  n_routes = 5,
  scaling_factor = 0.8
)
```

`plot_si_fit_result` *Plot Serial Interval Fit from si_estim Result*

Description

A convenience wrapper for `plot_si_fit` that accepts the output from `si_estim` directly, automatically handling the weight aggregation for different distribution types and number of routes.

Usage

```
plot_si_fit_result(
  si_result,
  dat,
  dist = c("normal", "gamma"),
  scaling_factor = 1
)
```

Arguments

`si_result`	list; the output from `si_estim` containing mean, sd, wts, and n_routes components.
`dat`	numeric vector; the index case-to-case (ICC) intervals in days.
`dist`	character; the distribution family. Must be either "normal" (default) or "gamma". Should match the distribution used in `si_estim()`.
`scaling_factor`	numeric; multiplicative factor to adjust the height of the fitted density curve. Defaults to 1.

Details

This function reads `n_routes` directly from the `si_result` object and aggregates component weights automatically:

- For normal distribution: the Co-primary weight is taken directly, and weights for routes 2 to n_routes are aggregated by summing the two symmetric components (positive and negative).
- For gamma distribution: weights are taken directly as the first n_routes - 1 values from `si_result$wts`.

Value

A `ggplot2` object.

See Also

`si_estim`, `plot_si_fit`

Examples

```
set.seed(123)
icc_data <- c(
  abs(rnorm(15, mean = 0, sd = 2)),
  rnorm(40, mean = 12, sd = 3),
  rnorm(15, mean = 24, sd = 4)
)
icc_data <- round(pmax(icc_data, 0))


# 4 routes (default)
result <- si_estim(icc_data, n = 50)
plot_si_fit_result(result, icc_data, dist = "normal")

# 5 routes
result5 <- si_estim(icc_data, n = 50, n_routes = 5)
plot_si_fit_result(result5, icc_data, dist = "normal")
```

---

si_estim *Estimate Serial Interval Distribution Using the Vink Method*

Description

Estimates the mean and standard deviation of the serial interval distribution from outbreak data using the Expectation-Maximization (EM) algorithm developed by Vink et al. (2014). The serial interval is defined as the time between symptom onset in a primary case and symptom onset in a secondary case infected by that primary case.

Usage

```
si_estim(
  dat,
  n = 50,
  dist = "normal",
  init = NULL,
  tol = 1e-06,
  n_starts = 1,
  n_routes = 4L,
  wind = 1
)
```

Arguments

dat	numeric vector; index case-to-case (ICC) intervals in days.
n	integer; number of EM algorithm iterations to perform. Defaults to 50.
dist	character; the assumed parametric family for the serial interval distribution. Must be either "normal" (default) or "gamma".
init	numeric vector of length 2; initial values for the mean and standard deviation. If NULL (default), uses the sample mean and sample standard deviation.
tol	numeric; convergence tolerance for the EM algorithm. Defaults to 1e-6.
n_starts	integer; number of random restarts for the EM algorithm. Defaults to 1.
n_routes	integer; number of transmission routes to model. Must be >= 2. Defaults to 4 (Co-Primary, Primary-Secondary, Primary-Tertiary, Primary-Quaternary). Increasing this allows modelling longer transmission chains.
wind	The window censure interval

Value

A named list containing:

- mean: Estimated mean of the serial interval distribution (days)
- sd: Estimated standard deviation of the serial interval distribution (days)
- wts: Numeric vector of estimated component weights. Length is 2*n_routes - 1 for normal distribution, n_routes for gamma distribution.
- converged: Logical indicating whether the algorithm converged.
- iterations: Integer indicating the number of iterations performed.
- loglik: Log-likelihood of the fitted model.
- n_restarts: Number of restarts performed.
- n_routes: Number of transmission routes used.

References

Vink MA, Bootsma MCJ, Wallinga J (2014). Serial intervals of respiratory infectious diseases: A systematic review and analysis. American Journal of Epidemiology, 180(9), 865-875. doi:10.1093/aje/kwu209

Examples

```
# Example 1: Basic usage with simulated data, default 4 routes
set.seed(123)
simulated_icc <- c(
  rep(1, 20), rep(2, 25), rep(3, 15), rep(4, 8)
)
result <- si_estim(simulated_icc)


# Example 2: Using 5 routes
result_5routes <- si_estim(simulated_icc, n_routes = 5)

# Example 3: Using gamma distribution with 3 routes
result_gamma <- si_estim(simulated_icc, dist = "gamma", n_routes = 3)
```

---

smooth_estimates *Apply Moving Average Smoothing to R Estimates*

Description

Applies temporal smoothing to reproduction number estimates using a centered moving average window. Handles missing and infinite values appropriately.

Usage

```
smooth_estimates(r_estimate, window)
```

Arguments

r_estimate	numeric vector; reproduction number estimates to smooth. Can contain NA or infinite values
window	integer; size of the smoothing window in time units. Window is centered around each point

Value

numeric vector; smoothed reproduction number estimates of the same length as input. Returns NA for points with insufficient valid neighboring values

`wallinga_lipsitch` *Estimate Time-Varying Case Reproduction Number Using Wallinga-Lipsitch Method*

Description

Estimates the time-varying case reproduction number (R_c) from daily incidence data using the method developed by Wallinga and Lipsitch (2007). The case reproduction number represents the average number of secondary infections generated by cases with symptom onset at time t, making it useful for retrospective outbreak analysis.

Usage

```
wallinga_lipsitch(
  incidence,
  dates,
  si_mean,
  si_sd,
  si_dist = "gamma",
  smoothing = 0,
  bootstrap = FALSE,
  n_bootstrap = 1000,
  conf_level = 0.95,
  shift = FALSE
)
```

Arguments

`incidence`	numeric vector; daily case counts. Must be non-negative integers or counts. Length must match `dates`
`dates`	vector; dates corresponding to each incidence count. Must be the same length as `incidence`. Can be Date objects or anything coercible to dates
`si_mean`	numeric; mean of the serial interval distribution in days. Must be positive. Typically estimated from contact tracing data or literature
`si_sd`	numeric; standard deviation of the serial interval distribution in days. Must be positive
`si_dist`	character; distribution family for the serial interval. Options: • `"gamma"` (default): Recommended for most applications as it naturally restricts to positive values • `"normal"`: Allows negative serial intervals, useful when co-primary infections are suspected
`smoothing`	integer; window size for temporal smoothing of R estimates. Use 0 for no smoothing (default), or positive integers for moving average smoothing over the specified number of days

`bootstrap`	logical; whether to calculate bootstrap confidence intervals. Defaults to `FALSE`. Setting to `TRUE` increases computation time but provides uncertainty quantification
`n_bootstrap`	integer; number of bootstrap samples when `bootstrap = TRUE`. More samples provide more stable intervals but increase computation time. Defaults to 1000
`conf_level`	numeric; confidence level for bootstrap intervals, between 0 and 1. Defaults to 0.95 (95% confidence intervals)
`shift`	logical; whether to shift R estimates forward by one mean serial interval. When `TRUE`, adds a `shifted_date` column for comparison with instantaneous reproduction number estimates. Defaults to `FALSE`

Details

The method calculates the relative likelihood that each earlier case infected each later case based on their time differences and the serial interval distribution, then aggregates these likelihoods to estimate reproduction numbers. The approach makes minimal assumptions beyond specifying the serial interval distribution.

Key features:

- **Pairwise likelihood approach**: Considers all epidemiologically plausible transmission pairs (earlier to later cases)
- **Right-truncation correction**: Adjusts for unobserved future cases (see `calculate_truncation_correction`)
- **Bootstrap confidence intervals**: Quantifies estimation uncertainty
- **Temporal shifting**: Optional alignment with instantaneous R estimates
- **Flexible smoothing**: User-controlled temporal smoothing of estimates

The Wallinga-Lipsitch method estimates the case reproduction number by:

1. Computing transmission likelihoods from each earlier case to each later case based on the serial interval distribution
2. Normalizing these likelihoods so they sum to 1 for each potential infectee
3. Aggregating normalized likelihoods to estimate expected secondary cases per primary case
4. Applying corrections for right-truncation bias

Right-truncation correction accounts for secondary cases that may occur after the observation period ends (see `calculate_truncation_correction`). This correction is particularly important for recent cases in the time series.

Bootstrap confidence intervals are calculated by resampling individual cases with replacement, providing non-parametric uncertainty estimates that account for both Poisson sampling variation and method uncertainty.

Value

A data frame with the following columns:

- `date`: Original input dates
- `incidence`: Original input case counts

- R: Estimated case reproduction number
- R_corrected: Case reproduction number with right-truncation correction
- R_lower, R_upper: Bootstrap confidence intervals for R (if bootstrap = TRUE)
- R_corrected_lower, R_corrected_upper: Bootstrap confidence intervals for R_corrected (if bootstrap = TRUE)
- shifted_date: Dates shifted forward by mean serial interval (if shift = TRUE)

Note

The case reproduction number differs from the instantaneous reproduction number in timing: R_c reflects the reproductive potential of cases by their symptom onset date, while instantaneous R reflects transmission potential at the time of infection. Use shift = TRUE for comparisons with instantaneous R estimates.

References

Wallinga J, Lipsitch M (2007). How generation intervals shape the relationship between growth rates and reproductive numbers. Proceedings of the Royal Society B: Biological Sciences, 274(1609), 599-604. doi:10.1098/rspb.2006.3754

See Also

si_estim for serial interval estimation, generate_synthetic_epidemic for testing data, calculate_truncation_correct for right-truncation correction

Examples

```
# Example 1: Basic usage with synthetic data
set.seed(123)
dates <- seq(as.Date("2023-01-01"), by = "day", length.out = 30)
incidence <- c(1, 2, 4, 7, 12, 15, 18, 20, 22, 19,
               16, 14, 11, 9, 7, 5, 4, 3, 2, 1,
               rep(0, 10))

# Estimate reproduction number
result <- wallinga_lipsitch(
  incidence = incidence,
  dates = dates,
  si_mean = 7,
  si_sd = 3,
  si_dist = "gamma"
)

# View results
head(result)

# Example 2: With bootstrap confidence intervals
result_ci <- wallinga_lipsitch(
  incidence = incidence,
  dates = dates,
```

```
  si_mean = 7,
  si_sd = 3,
  si_dist = "gamma",
  bootstrap = TRUE,
  n_bootstrap = 500  # Reduced for faster example
)

# Plot results with confidence intervals
if (require(ggplot2)) {
  library(ggplot2)
  ggplot(result_ci, aes(x = date)) +
    geom_ribbon(aes(ymin = R_corrected_lower, ymax = R_corrected_upper),
                alpha = 0.3, fill = "blue") +
    geom_line(aes(y = R_corrected), color = "blue", size = 1) +
    geom_hline(yintercept = 1, linetype = "dashed", color = "red") +
    labs(x = "Date", y = "Reproduction Number",
         title = "Time-varying Reproduction Number") +
    theme_minimal()
}

# Example 3: With smoothing and shifting
result_smooth <- wallinga_lipsitch(
  incidence = incidence,
  dates = dates,
  si_mean = 7,
  si_sd = 3,
  si_dist = "gamma",
  smoothing = 7,  # 7-day smoothing window
  shift = TRUE    # Shift for comparison with instantaneous R
)

# Example 4: Using normal distribution for serial interval
result_normal <- wallinga_lipsitch(
  incidence = incidence,
  dates = dates,
  si_mean = 6,
  si_sd = 2,
  si_dist = "normal",
  smoothing = 5
)
```

Index

calculate_truncation_correction, *13, 14*
compare_n_routes, 2

f_gam, *7*
f_norm, *7*

generate_synthetic_epidemic, 4, *14*

plot_si_fit, 6, *8, 9*
plot_si_fit_result, *3*, 8

si_estim, *3*, *7, 8*, 9, *9*, *14*
smooth_estimates, 11

wallinga_lipsitch, *5*, 12