

Package: serodynamics (via r-universe)

October 9, 2026

Title Modeling Longitudinal Antibody Responses to Infection

Version 0.1.0.9024

Description Implements Bayesian hierarchical models for estimating antibody kinetic parameters from longitudinal serological data. Fits two-phase within-host models capturing antibody rise, peak, and decay following pathogen infection, using 'JAGS' for posterior inference. Designed as the upstream companion to the 'serocalculator' package for end-to-end seroepidemiological analysis. Methods are described in Teunis and colleagues (2016) <[doi:10.1016/j.epidem.2016.04.001](https://doi.org/10.1016/j.epidem.2016.04.001)> and Teunis and van Eijkeren (2020) <[doi:10.1002/sim.8578](https://doi.org/10.1002/sim.8578)>.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

URL <https://github.com/UCD-SERG/serodynamics>,
<https://ucd-serg.github.io/serodynamics/>

BugReports <https://github.com/UCD-SERG/serodynamics/issues>

SystemRequirements JAGS (>= 4.3.1)

Imports cli, coda, dplyr (>= 1.2.0), ggmcmm (>= 1.5.1.2), ggplot2, lifecycle, purrr, rlang, runjags, scales, serocalculator (>= 1.4.1), stats, tibble, tidyr, tidyselect, utils

Suggests Hmisc, knitr, readr, rlist, sessioninfo, spelling, stringr, testthat (>= 3.0.0), tidyverse, withr, vdiff, fs, here, rjags, rmarkdown, magrittr, rex, quarto

Language en-US

Config/testthat/edition 3

Config/needs/test withr, vdiff, here, readr, magrittr, rlist

Config/needs/check commonmark, xml2

Depends R (>= 4.1.0)

LazyData true

VignetteBuilder knitr, quarto
Config/roxygen2/version 8.1.0
Config/pak/sysreqs
cmake make jags libicu-dev libuv1-dev libssl-dev libudunits2-dev libx11-dev zlib1g-dev
Repository https://ucd-serg.r-universe.dev
Date/Publication 2026-09-24 18:01:39 UTC
RemoteUrl https://github.com/ucd-serg/serodynamics
RemoteRef HEAD
RemoteSha c051209356caba77c29fe26d130d7d4504c6649a

Contents

as_case_data	2
autoplot.case_data	3
initsfunction	5
load_data	6
nepal_sees	7
nepal_sees_jags_output	7
plot_density	8
plot_ess	9
plot_predicted_curve	11
plot_residuals	13
plot_rhat	14
plot_serocurve	16
plot_trace	18
postprocess_jags_output	19
prep_data	21
prep_priors	22
run_serodynamics	24
serodynamics_example	28
sim_case_data	29
summarize_posterior	30
Index	32

as_case_data	<i>Convert data into case_data</i>
--------------	------------------------------------

Description

Convert data into case_data

Usage

```
as_case_data(
  data,
  id_var = "index_id",
  biomarker_var = "antigen_iso",
  value_var = "value",
  time_in_days = "timeindays"
)
```

Arguments

data	a data.frame
id_var	a character string naming the column in data denoting participant ID
biomarker_var	a character string naming the column in data denoting which biomarker is being reported in value_var (e.g. "antigen_iso")
value_var	a character string naming the column in data with biomarker measurements
time_in_days	a character string naming the column in data with elapsed time since seroconversion

Value

a case_data object

Examples

```
set.seed(1)
serocalculator::typhoid_curves_nostrat_100 |>
  sim_case_data(n = 5) |>
  as_case_data(
    id_var = "id",
    biomarker_var = "antigen_iso",
    time_in_days = "timeindays",
    value_var = "value"
  )
```

autoplot.case_data	<i>Plot case data</i>
--------------------	-----------------------

Description

Plot case data

Usage

```
## S3 method for class 'case_data'
autoplot(object, log_y = TRUE, log_x = FALSE, ...)
```

Arguments

- object** a case_data object
- log_y** whether to log-transform the y-axis
- log_x** whether to log-transform the x-axis
- ...** Arguments passed on to [ggplot2::geom_point](#), [ggplot2::geom_line](#)
- mapping** Set of aesthetic mappings created by [aes\(\)](#). If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
- data** The data to be displayed in this layer. There are three options:
 If NULL, the default, the data is inherited from the plot data as specified in the call to [ggplot\(\)](#).
 A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See [fortify\(\)](#) for which variables will be created.
 A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. `~ head(.x, 10)`).
- stat** The statistical transformation to use on the data for this layer. When using a `geom_*()` function to construct a layer, the `stat` argument can be used to override the default coupling between geoms and stats. The `stat` argument accepts the following:
- A Stat ggproto subclass, for example `StatCount`.
 - A string naming the stat. To give the stat as a string, strip the function name of the `stat_` prefix. For example, to use `stat_count()`, give the stat as "count".
 - For more information and other ways to specify the stat, see the [layer stat](#) documentation.
- position** A position adjustment to use on the data for this layer. This can be used in various ways, including to prevent overplotting and improving the display. The `position` argument accepts the following:
- The result of calling a position function, such as `position_jitter()`. This method allows for passing extra arguments to the position.
 - A string naming the position adjustment. To give the position as a string, strip the function name of the `position_` prefix. For example, to use `position_jitter()`, give the position as "jitter".
 - For more information and other ways to specify the position, see the [layer position](#) documentation.
- na.rm** If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
- show.legend** logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display. To include legend keys for all levels, even when no data exists, use TRUE. If NA, all levels are shown in legend, but unobserved levels are omitted.

`inherit.aes` If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `annotation_borders()`.

`arrow` Arrow specification, as created by `grid::arrow()`.

`arrow.fill` fill colour to use for the arrow head (if closed). NULL means use colour aesthetic.

`lineend` Line end style (round, butt, square).

`linejoin` Line join style (round, mitre, bevel).

`linemitre` Line mitre limit (number greater than 1).

`orientation` The orientation of the layer. The default (NA) automatically determines the orientation from the aesthetic mapping. In the rare event that this fails it can be given explicitly by setting orientation to either "x" or "y". See the *Orientation* section for more detail.

`direction` direction of stairs: 'vh' for vertical then horizontal, 'hv' for horizontal then vertical, or 'mid' for step half-way between adjacent x-values.

Value

a `ggplot2::ggplot`

Examples

```
set.seed(1)
sim_case_data <-
  serocalculator::typhoid_curves_nostrat_100 |>
  sim_case_data(n = 5, max_n_obs = 20, followup_interval = 14)

sim_case_data |>
  autoplot(alpha = .5)
```

initsfunction	<i>JAGS chain initialization function</i>
---------------	---

Description

JAGS chain initialization function

Usage

```
initsfunction(chain)
```

Arguments

`chain` an `integer` specifying which chain to initialize

Value

a [list](#) of RNG seeds and names

Examples

```
initsfunction(1)
```

load_data	<i>load and format data</i>
-----------	-----------------------------

Description

to add

Usage

```
load_data(  
  datapath = "inst/extdata/",  
  datafile = "TyphoidCaseData_github_09.30.21.csv"  
)
```

Arguments

datapath	path to data folder
datafile	data file name

Value

a [list](#) with the following elements:

- `smpl.t` = time since symptom/fever onset for each participant, max 7 visits
- `logy` = log antibody response at each time-point (max 7) for each of 7 biomarkers for each participant
- `ntest` is max number of biomarkers
- `nsmpl` = max number of samples per participant
- `nsubj` = number of study participants
- `ndim` = number of parameters to model(`y0`, `y1`, `t1`, `alpha`, `shape`)
- `my.hyp`, `prec.hyp`, `omega` and `wishdf` are all parameters to describe the shape of priors for (`y0`, `y1`, `t1`, `alpha`, `shape`)

nepal_sees	<i>SEES Typhoid data</i>
------------	--------------------------

Description

A subset of data from the SEES project highlighting Typhoid longitudinal data from Nepal.

Usage

```
nepal_sees
```

Format

```
nepal_sees:
```

A `base::data.frame()` with 904 rows and 8 columns:

Country Country name

person_id ID identifying a study participant

sample_id ID identifying sample taken

bldculres Pathogen participant tested positive for; Typhoid or paratyphoid

antigen_iso The antigen/antibody combination included in the assay

studyvisit Categorical estimated time frame for when sample was taken; 28 days, 3_months, 6_months, 12_months, baseline, or 18_months

dayssincefeveronset Continuous measurement showing how exact days since symptom onset

result Continuous variable describing ELISA result

Source

reference study: [doi:10.1016/S26665247\(22\)001148](https://doi.org/10.1016/S26665247(22)001148)

nepal_sees_jags_output	<i>SEES Typhoid run_serodynamics jags output</i>
------------------------	--

Description

A `run_serodynamics()` output using the `nepal_sees` example data set as input and stratifying by column "bldculres", which is the diagnosis type (typhoid or paratyphoid). Keeping only IDs "newperson", "sees_npl_1", "sees_npl_2".

Usage

```
nepal_sees_jags_output
```

Format

An S3 object of class `sr_model`: A [tibble::tbl_df](#) that contains the posterior predictive distribution of the person-specific parameters for a "new person" with no observed data (Subject = "newperson") and posterior distributions of the person-specific parameters for two arbitrarily-chosen subjects ("sees_npl_1" and "sees_npl_2"). Contains 40,000 rows, 7 columns, and model attributes.

Iteration Number of sampling iterations: 500 iterations

Chain Number of MCMC chains run: 2 chains run

Parameter Parameter being estimated

Iso_type Antibody/antigen type combination being evaluated: HlyE_IgA and HlyE_IgG

Stratification The variable used to stratify jags model: typhi and paratyphi

Subject ID of subject being evaluated: newperson, sees_npl_1, sees_npl_2

value Estimated value of the parameter

attributes A [list](#) of attributes that summarize the jags inputs, priors, optional `jags_post` mcmc object, and population-level parameters. Includes a `population_params` attribute with posterior samples of the population-level parameters (`mu.par`, `prec.par`, `prec.log`), indexed by Iteration, Chain, Parameter, Iso_type, Stratification, and Population_Parameter.

Source

reference study: [doi:10.1016/S26665247\(22\)001148](https://doi.org/10.1016/S26665247(22)001148)

plot_density

Density Plot Diagnostics

Description

`plot_density()` takes a [list](#) output from `run_serodynamics()` to create density plots for each chain run in the mcmc estimation. Defaults will produce every combination of antigen/antibody, parameters, and stratifications, unless otherwise specified. Antigen/antibody combinations and stratifications will vary by analysis. The antibody dynamic curve includes the following parameters:

- `y0` = baseline antibody concentration
- `y1` = peak antibody concentration
- `t1` = time to peak
- `shape` = shape parameter
- `alpha` = decay rate

Usage

```
plot_density(
  data,
  iso = unique(data$Iso_type),
  param = unique(data$Parameter),
  strat = unique(data$Stratification)
)
```

Arguments

data	A list outputted from run_serodynamics() .
iso	Specify character string to produce plots of only a specific antigen/antibody combination, entered with quotes. Default outputs all antigen/antibody combinations.
param	Specify character string to produce plots of only a specific parameter, entered with quotes. Options include: • alpha = posterior estimate of decay rate • shape = posterior estimate of shape parameter • t1 = posterior estimate of time to peak • y0 = posterior estimate of baseline antibody concentration • y1 = posterior estimate of peak antibody concentration
strat	Specify character string to produce plots of specific stratification entered in quotes.

Value

A [base::list\(\)](#) of [ggplot2::ggplot\(\)](#) objects producing density plots for all the specified input.

Author(s)

Sam Schildhauer

Examples

```
data <- serodynamics::nepal_sees_jags_output

# Specifying isotype and stratification for traceplot.
plot_density(
  data = data,
  iso = "HlyE_IgA",
  strat = "typhi")
```

plot_ess

Plot Effective Sample Size Diagnostics

Description

plot_ess() takes a [list](#) output from [run_serodynamics\(\)](#) to create summary diagnostics for each chain run in the mcmc estimation. Defaults will produce every combination of antigen/antibody, parameters, and stratifications, unless otherwise specified. At least 2 chains are required to run function. Antigen/antibody combinations and stratifications will vary by analysis. The antibody dynamic curve includes the following parameters:

- y0 = baseline antibody concentration
- y1 = peak antibody concentration

- t1 = time to peak
- shape = shape parameter
- alpha = decay rate

Usage

```
plot_ess(
  data,
  iso = unique(data$Iso_type),
  param = unique(data$Parameter),
  strat = unique(data$Stratification)
)
```

Arguments

data	A list outputted from run_serodynamics() .
iso	Specify character string to produce plots of only a specific antigen/antibody combination, entered with quotes. Default outputs all antigen/antibody combinations.
param	Specify character string to produce plots of only a specific parameter, entered with quotes. Options include: • y0 = posterior estimate of baseline antibody concentration • y1 = posterior estimate of peak antibody concentration • t1 = posterior estimate of time to peak • shape = posterior estimate of shape parameter • alpha = posterior estimate of decay rate
strat	Specify character string to produce plots of specific stratification entered in quotes.

Value

A [list](#) of [ggplot2::ggplot](#) objects showing the proportion of effective samples taken/total samples taken for all parameter iso combinations. The estimate with the highest proportion of effective samples taken will be listed first.

Author(s)

Sam Schildhauer

Examples

```
data <- serodynamics::nepal_sees_jags_output

plot_ess(data = data,
  iso = "HlyE_IgA",
  strat = "typhi")
```

plot_predicted_curve *Generate Predicted Antibody Response Curves (Median + 95% CI)*

Description

Plots a median antibody response curve with a 95% credible interval ribbon, using MCMC samples from the posterior distribution. Optionally overlays observed data, applies logarithmic spacing on the y- and x-axes, and shows all individual sampled curves.

Usage

```
plot_predicted_curve(
  model,
  ids,
  antigen_iso,
  dataset = NULL,
  legend_obs = "Observed data",
  legend_median = "Median prediction",
  show_quantiles = TRUE,
  log_y = FALSE,
  log_x = FALSE,
  show_all_curves = FALSE,
  alpha_samples = 0.3,
  xlim = NULL,
  ylab = NULL,
  facet_by_id = length(ids) > 1,
  ncol = NULL
)
```

Arguments

model	An <code>sr_model</code> object (returned by run_serodynamics) containing samples from the posterior distribution of the model parameters.
ids	The participant IDs to plot; for example, "sees_np1_128".
antigen_iso	The antigen isotype to plot; for example, "HlyE_IgA" or "HlyE_IgG".
dataset	(Optional) A tibble::tbl_df with observed antibody response data. Must contain: • timeindays • value • id • antigen_iso
legend_obs	Label for observed data in the legend.
legend_median	Label for the median prediction line.
show_quantiles	logical ; if TRUE (default), plots the 2.5%, 50%, and 97.5% quantiles.
log_y	logical ; if TRUE , applies a log10 transformation to the y-axis.

log_x **logical**; if **TRUE**, applies a **log10** transformation to the x-axis.
show_all_curves **logical**; if **TRUE**, overlays all individual sampled curves.
alpha_samples Numeric; transparency level for individual curves (default = 0.3).
xlim (Optional) A numeric vector of length 2 providing custom x-axis limits.
ylab (Optional) A string for the y-axis label. If **NULL** (default), the label is automatically set to "ELISA units" or "ELISA units (log scale)" based on the **log_y** argument.
facet_by_id **logical**; if **TRUE**, facets the plot by 'id'. Defaults to **TRUE** when multiple IDs are provided.
ncol **integer**; number of columns for faceting.

Value

A **ggplot2::ggplot** object displaying predicted antibody response curves with a median curve and a 95% credible interval band as default.

Examples

```

sees_model <- serodynamics::nepal_sees_jags_output |>
  dplyr::filter(Iteration <= 50) # use 50 of 500 iterations for faster examples
sees_data <- serodynamics::nepal_sees

# Plot (linear axes) with all individual curves + median ribbon
p1 <- plot_predicted_curve(
  model          = sees_model,
  dataset        = sees_data,
  ids            = "sees_npl_128",
  antigen_iso    = "HlyE_IgA",
  show_quantiles = TRUE,
  log_y          = FALSE,
  log_x          = FALSE,
  show_all_curves = TRUE
)
print(p1)

# Plot (log10 y-axis) with all individual curves + median ribbon
p2 <- plot_predicted_curve(
  model          = sees_model,
  dataset        = sees_data,
  ids            = "sees_npl_128",
  antigen_iso    = "HlyE_IgA",
  show_quantiles = TRUE,
  log_y          = TRUE,
  log_x          = FALSE,
  show_all_curves = TRUE
)
print(p2)

# Plot with custom x-axis limits (0-600 days)

```

```

p3 <- plot_predicted_curve(
  model          = sees_model,
  dataset        = sees_data,
  ids            = "sees_npl_128",
  antigen_iso    = "HlyE_IgA",
  show_quantiles = TRUE,
  log_y          = FALSE,
  log_x          = FALSE,
  show_all_curves = TRUE,
  xlim           = c(0, 600)
)
print(p3)

# Multi-ID, faceted plot (single antigen):
p4 <- plot_predicted_curve(
  model          = sees_model,
  dataset        = sees_data,
  ids            = c("sees_npl_128", "sees_npl_131"),
  antigen_iso    = "HlyE_IgA",
  show_all_curves = TRUE,
  facet_by_id    = TRUE
)
print(p4)

```

plot_residuals

Plot Residuals Over Time

Description

Plots residuals over time and facets by antigen-isotype (Iso_type). The mean absolute residual for each facet is annotated in the upper-right corner. Fitted and residual values are calculated on demand via `calc_fit_mod()`, using the `original_data`, `strat`, and `decay_type` attributes stored on model by `run_serodynamics()`, and include both natural-scale and log10-scale medians and 2.5%/97.5% posterior quantiles.

Usage

```

plot_residuals(
  model,
  ids = NULL,
  antigen_isos = NULL,
  log_y = TRUE,
  show_interval = TRUE,
  connect_lines = FALSE
)

```

Arguments

model	An <code>sr_model</code> object (returned by <code>run_serodynamics()</code>), with <code>original_data</code> , <code>strat</code> , and <code>decay_type</code> attributes (see <code>calc_fit_mod()</code>).
ids	(Optional) Participant IDs to include. When supplied, points (and, if <code>connect_lines = TRUE</code> , lines) are colored by subject; otherwise no color is used.
antigen_isos	(Optional) Antigen-isotypes (<code>antigen_iso</code>) to include.
log_y	logical ; if <code>TRUE</code> (default), plots the residual computed on the log10 scale ($\log_{10}(\text{observed}) - \log_{10}(\text{fitted})$); if <code>FALSE</code> , plots the natural-scale residual.
show_interval	logical ; if <code>TRUE</code> (default), draws an error bar around each residual spanning its 2.5%/97.5% posterior interval, to visualize the precision of the posterior.
connect_lines	logical ; if <code>TRUE</code> , connects each subject's residuals over time with a line. Default <code>FALSE</code> .

Value

A `ggplot2::ggplot` object.

Examples

```
plot_residuals(
  model = serodynamics::nepal_sees_jags_output,
  ids = c("sees_npl_128", "sees_npl_131"),
  antigen_isos = c("HlyE-IgA", "HlyE-IgG")
)
```

plot_rhat

Rhat Plot Diagnostics

Description

`plot_rhat()` takes a **list** output from `run_serodynamics()` to produce dotplots of potential scale reduction factors (Rhat) for each chain run in the mcmc estimation. Rhat values analyze the spread of chains compared to pooled values with a goal of observing $\text{rhat} < 1.10$ for convergence. Defaults will produce every combination of antigen/antibody, parameters, and stratifications, unless otherwise specified. Antigen/antibody combinations and stratifications will vary by analysis. The antibody dynamic curve includes the following parameters:

- `y0` = baseline antibody concentration
- `y1` = peak antibody concentration
- `t1` = time to peak
- `shape` = shape parameter
- `alpha` = decay rate

Usage

```
plot_rhat(
  data,
  iso = unique(data$Iso_type),
  param = unique(data$Parameter),
  strat = unique(data$Stratification)
)
```

Arguments

data	A list outputted from <code>run_serodynamics()</code> .
iso	Specify character string to produce plots of only a specific antigen/antibody combination, entered with quotes. Default outputs all antigen/antibody combinations.
param	Specify character string to produce plots of only a specific parameter, entered with quotes. Options include: • y_0 = posterior estimate of baseline antibody concentration • y_1 = posterior estimate of peak antibody concentration • t_1 = posterior estimate of time to peak • shape = posterior estimate of shape parameter • alpha = posterior estimate of decay rate
strat	Specify character string to produce plots of specific stratification entered in quotes.

Value

A [list](#) of `ggplot2::ggplot` objects producing dotplots with rhat values for all the specified input.

Author(s)

Sam Schildhauer

Examples

```
data <- serodynamics::nepal_sees_jags_output

plot_rhat(data = data,
  iso = "HlyE_IgA",
  strat = "typhi")
```

plot_serocurve

Plot Estimated Serodynamic Curves at Predictive or Population Level

Description

Plots the estimated antibody response curve derived from posterior samples of population-level (`mu.par`) or the predictive distribution from a fitted `run_serodynamics()` model. A median curve with an optional 95% credible interval ribbon is produced for each requested antigen-isotype and stratification combination.

Usage

```
plot_serocurve(
  model,
  antigen_iso = unique(model$Iso_type),
  strat = unique(model$Stratification),
  param_source = "predictive",
  show_ci = TRUE,
  log_y = FALSE,
  log_x = FALSE,
  xlim = NULL,
  facet_by_strat = FALSE,
  ...
)
```

Arguments

<code>model</code>	An <code>sr_model</code> object returned by <code>run_serodynamics()</code> .
<code>antigen_iso</code>	A character vector of antigen-isotype combinations to plot. Defaults to all antigen-isotypes present in the subject-level draws of <code>model</code> (<code>model\$Iso_type</code>); in normal usage these match the levels available in <code>attr(model, "population_params")</code> .
<code>strat</code>	A character vector of stratification levels to include. Defaults to all stratification levels present in the subject-level draws of <code>model</code> (<code>model\$Stratification</code>); in normal usage these match the levels available in <code>attr(model, "population_params")</code> .
<code>param_source</code>	character ; which posterior samples to use for the curve. Options: "predictive" (default): uses the predictive distribution for a new individual drawn from the population-level prior. "population": uses population-level <code>mu.par</code> samples stored in <code>attr(model, "population_params")</code>. Requires the model to have been fitted with <code>run_serodynamics(..., with_pop_params = TRUE)</code>.
<code>show_ci</code>	logical ; if TRUE (default), draws a 95% credible interval ribbon around the median curve.
<code>log_y</code>	logical ; if TRUE , applies a log10 transformation to the y-axis.
<code>log_x</code>	logical ; if TRUE , applies a log10 transformation to the x-axis.

`xlim` (Optional) A numeric vector of length 2 providing custom x-axis limits.

`facet_by_strat` **logical**; if **TRUE**, facets the plot by stratification level. When **FALSE** (default), different stratification levels are shown as different colours on the same panel.

... Arguments passed on to `add_serocurve_facets`

`facet_by_antigen_iso` **logical**; if **TRUE**, facets the plot by antigen-isotype. Defaults to **TRUE** when multiple antigen-isotypes are requested.

`ncol` **integer**; number of columns when faceting. If **NULL** (default), a sensible value is chosen automatically.

Value

A `ggplot2::ggplot` object.

Examples

```
# nepal_sees_jags_output already includes population_params
model <- serodynamics::nepal_sees_jags_output

# Predictive curve for a single antigen-isotype and stratum
p1 <- plot_serocurve(
  model      = model,
  antigen_iso = "HlyE-IgA",
  strat      = "typhi"
)
print(p1)

# Predictive curves for both stratifications, colored by stratum
p2 <- plot_serocurve(
  model      = model,
  antigen_iso = "HlyE-IgA"
)
print(p2)

# Predictive curve faceting by stratification instead of coloring
p3 <- plot_serocurve(
  model      = model,
  antigen_iso = "HlyE-IgA",
  facet_by_strat = TRUE
)
print(p3)

# Population level curve for multiple antigen-isotypes, faceted, without CI
p4 <- plot_serocurve(
  model      = model,
  antigen_iso = c("HlyE-IgA", "HlyE-IgG"),
  param_source = "population",
  facet_by_antigen_iso = TRUE,
  show_ci      = FALSE
)
print(p4)
```

```
# Population level distribution
p5 <- plot_serocurve(
  model      = model,
  antigen_iso = "HlyE_IgA",
  param_source = "population"
)
print(p5)
```

plot_trace

Trace Plot Diagnostics

Description

plot_trace() takes a [list](#) output from [run_serodynamics\(\)](#) to create trace plots for each chain run in the mcmc estimation. Defaults will produce every combination of antigen/antibody, parameters, and stratifications, unless otherwise specified. Antigen/antibody combinations and stratifications will vary by analysis. The antibody dynamic curve includes the following parameters:

- y0 = baseline antibody concentration
- y1 = peak antibody concentration
- t1 = time to peak
- shape = shape parameter
- alpha = decay rate

Usage

```
plot_trace(
  data,
  iso = unique(data$Iso_type),
  param = unique(data$Parameter),
  strat = unique(data$Stratification)
)
```

Arguments

data	A list outputted from run_serodynamics() .
iso	Specify character string to produce plots of only a specific antigen/antibody combination, entered with quotes. Default outputs all antigen/antibody combinations.
param	Specify character string to produce plots of only a specific parameter, entered with quotes. Options include: • alpha = posterior estimate of decay rate • shape = posterior estimate of shape parameter • t1 = posterior estimate of time to peak • y0 = posterior estimate of baseline antibody concentration • y1 = posterior estimate of peak antibody concentration
strat	Specify character string to produce plots of specific stratification entered in quotes.

Value

A [list](#) of [ggplot2::ggplot](#) objects producing trace plots for all the specified input.

Author(s)

Sam Schildhauer

Examples

```
data <- serodynamics::nepal_sees_jags_output

# Specifying isotype, parameter, and stratification for traceplot.
plot_trace(
  data = data,
  iso = "HlyE_IgA",
  strat = "typhi")
```

postprocess_jags_output
Postprocess JAGS output

Description

Postprocess JAGS output

Usage

```
postprocess_jags_output(jags_output, ids, antigen_isos)
```

Arguments

jags_output	output from runjags::run.jags()
ids	IDs of individuals being sampled (JAGS discards this information, so it has to be re-attached)
antigen_isos	names of biomarkers being modeled (JAGS discards this information, so it has to be re-attached)

Value

a [tibble::tbl_df](#)

Examples

```

set.seed(1)
raw_data <-
  serocalculator::typhoid_curves_nostrat_100 |>
  dplyr::filter(
    antigen_iso |> stringr::str_starts(pattern = "HlyE")
  ) |>
  sim_case_data(
    n = 5,
    antigen_isos = c("HlyE-IgA", "HlyE-IgG")
  )
prepped_data <- prep_data(raw_data)
priors <- prep_priors(max_antigens = prepped_data$n_antigen_isos,
  mu_hyp_param = c(1.0, 7.0, 1.0, -4.0, -1.0),
  prec_hyp_param = c(1.0, 0.00001, 1.0, 0.001, 1.0),
  omega_param = c(1.0, 50.0, 1.0, 10.0, 1.0),
  wishdf_param = 20,
  prec_logy_hyp_param = c(4.0, 1.0))

nchains <- 2
# nr of MC chains to run simultaneously
nadapt <- 100
# nr of iterations for adaptation
nburnin <- 100
# nr of iterations to use for burn-in
nmc <- 100
# nr of samples in posterior chains
niter <- 100
# nr of iterations for posterior sample
nthin <- round(niter / nmc)
# thinning needed to produce nmc from niter

tomonitor <- c("y0", "y1", "t1", "alpha", "shape")

file_mod <- serodynamics_example("model.jags")

set.seed(11325)
jags_post <- runjags::run.jags(
  model = file_mod,
  data = c(prepped_data, priors),
  inits = initsfunction,
  method = "parallel",
  adapt = nadapt,
  burnin = nburnin,
  thin = nthin,
  sample = nmc,
  n.chains = nchains,
  monitor = tomonitor,
  summarise = FALSE
)

curve_params <- jags_post |> postprocess_jags_output(
  ids = attr(prepped_data, "ids"),

```

```

    antigen_isos = attr(prepped_data, "antigens")
  )

  print(curve_params)

```

prep_data

prepare data for JAGs

Description

prepare data for JAGs

Usage

```

prep_data(
  dataframe,
  biomarker_column = get_biomarker_names_var(dataframe),
  verbose = FALSE,
  add_newperson = TRUE
)

```

Arguments

dataframe a [data.frame](#) containing ...

biomarker_column [character](#) string indicating which column contains antigen-isotype names

verbose whether to produce verbose messaging

add_newperson whether to add an extra record with missing data

Value

a prep_data object (a [list](#) with extra attributes ...)

Examples

```

set.seed(1)
raw_data <-
  serocalculator::typhoid_curves_nostrat_100 |>
  sim_case_data(n = 5)
prep_data <- prep_data(raw_data)

```

```
prep_priors
```

Prepare priors

Description

Takes multiple [vector](#) inputs to allow for modifiable priors. Priors must be specified as an option in `run_serodynamics()`.

Usage

```
prep_priors(
  max_antigens,
  mu_hyp_param = NULL,
  prec_hyp_param = NULL,
  omega_param = NULL,
  wishdf_param = NULL,
  prec_logy_hyp_param = NULL,
  decay_type = "power"
)
```

Arguments

- | | |
|-----------------------------|---|
| <code>max_antigens</code> | An integer specifying how many antigen-isotypes (biomarkers) will be modeled. |
| <code>mu_hyp_param</code> | <p>A numeric vector of 5 values representing the prior mean for the population level parameters parameters (y0, y1, t1, alpha, r) for each biomarker. Will be 5 values long specified by the user, representing the following parameters:</p> <ul style="list-style-type: none"> • y0 = baseline antibody concentration • y1 = peak antibody concentration • t1 = time to peak • alpha = decay rate • r = shape parameter (If running <code>decay_type == "exponential"</code> no shape parameter needs to be specified). When <code>decay_type = "exponential"</code> only 4 parameters (y0, y1, t1, alpha) are used. |
| <code>prec_hyp_param</code> | A numeric vector of 5 values corresponding to hyperprior diagonal entries for the precision matrix (i.e. inverse variance) representing prior covariance of uncertainty around <code>mu_hyp_param</code> . Will be 5 values long corresponding to the 5 estimated parameters (4 values when <code>decay_type = "exponential"</code>). |
| <code>omega_param</code> | A numeric vector of 5 values corresponding to the diagonal entries representing the Wishart hyperprior distributions of <code>prec_hyp_param</code> , describing how much we expect parameters to vary between individuals. Will be 5 values long corresponding to the 5 estimated parameters (4 values when <code>decay_type = "exponential"</code>). |
| <code>wishdf_param</code> | An integer vector of 1 value specifying the degrees of freedom for the Wishart hyperprior distribution of <code>prec_hyp_param</code> . Must be 1 value long. |

- The value of `wishdf_param` controls how informative the Wishart prior is. Higher values lead to tighter priors on individual variation. Lower values (e.g., 5–10) make the prior more weakly informative, which can help improve convergence if the model is over-regularized.

`prec_logy_hyp_param`

A [numeric vector](#) of 2 values corresponding to hyperprior diagonal entries on the log-scale for the precision matrix (i.e. inverse variance) representing prior beliefs of individual variation. Must be 2 values long.

`decay_type`

A [character](#) string specifying the decay function used in the model. Options are "power" and "exponential". Default is "power". The "power" option uses $y(t) = (y1^{(1-shape)} - (1-shape)*alpha*(t-t1))^{1/(1-shape)}$. The "exponential" option uses $y(t) = y1 * \exp(-alpha*(t-t1))$. The exponential model does not estimate shape; its processed output includes `shape = 1` as a fixed value to preserve the common output structure. Note: `prep_priors()` validates 5-element prior vector for power decay and 4-element prior vectors for exponential decay (the fifth (`shape`) prior is excluded in exponential decay).

Value

A "curve_params_priors" object (a subclass of [list](#) with the inputs to `prep_priors()` attached as [attributes](#) entry named "used_priors"), containing the following elements:

- "n_params": Corresponds to the 5 parameters being estimated. When `decay_type = "exponential"`, 4 parameters are being estimated.
- "mu.hyp": A [matrix](#) of hyperpriors with dimensions `max_antigens` x 5 (# of parameters), representing the mean of the hyperprior distribution for the five seroresponse parameters: `y0`, `y1`, `t1`, `alpha`, and `r` (`max_antigens` x 4 when `decay_type = "exponential"`, including `y0`, `y1`, `t1`, and `alpha`).
- "prec.hyp": A three-dimensional [numeric array](#) with dimensions `max_antigens` x 5 (# of parameters), containing the precision matrices of the hyperprior distributions of `mu.hyp`, for each biomarker (`max_antigens` x 4 when `decay_type = "exponential"`).
- "omega": A three-dimensional [numeric array](#) with 5 [matrix](#), each with dimensions `max_antigens` x 5 (# of parameters), representing the precision matrix of Wishart hyper-priors for `prec.hyp` (`max_antigens` x 4 when `decay_type = "exponential"`).
- "wishdf": A [vector](#) of 2 values specifying the degrees of freedom for the Wishart distribution used in the subject-level precision prior.
- "prec.logy.hyp": A [matrix](#) of hyper-parameters for the precision (inverse variance) of individual variation measuring `max_antigens` x 2, on the log-scale.
- `used_priors` = inputs to `prep_priors()` attached as attributes.

Examples

```
prep_priors(max_antigens = 2,
            mu_hyp_param = c(1.0, 7.0, 1.0, -4.0, -1.0),
            prec_hyp_param = c(1.0, 0.00001, 1.0, 0.001, 1.0),
            omega_param = c(1.0, 50.0, 1.0, 10.0, 1.0),
            wishdf_param = 20,
            prec_logy_hyp_param = c(4.0, 1.0))
```

run_serodynamics	<i>Run Jags Model</i>
------------------	-----------------------

Description

run_serodynamics() takes a data frame and adjustable MCMC inputs to `runjags::run.jags()` as an MCMC Bayesian model to estimate antibody dynamic curve parameters. The `rjags::jags.model()` models seroresponse dynamics to an infection. Priors are required for modeling and must be specified prior to running run_serodynamics. The antibody dynamic curve includes the following parameters:

- y0 = baseline antibody concentration
- y1 = peak antibody concentration
- t1 = time to peak
- shape = shape parameter, estimated for power decay and fixed at 1 for exponential decay
- alpha = decay rate

Usage

```
run_serodynamics(
  data,
  file_mod = NULL,
  decay_type = "power",
  nchain = 4,
  nadapt = 0,
  nburn = 0,
  nmc = 100,
  niter = 100,
  strat = NA,
  with_post = FALSE,
  with_pop_params = FALSE,
  preclogy_per_iso = FALSE,
  ...
)
```

Arguments

data	A <code>base::data.frame()</code> with the following columns.
file_mod	The model file path. If NULL, a built-in model file is selected based on decay_type. If supplied, its parameterization must be compatible with decay_type.
decay_type	A <code>character</code> string specifying the decay function used in the model. Options are "power" and "exponential". Default is "power". The "power" option uses $y(t) = (y1^{(1-shape)} - (1-shape)*alpha*(t-t1))^{(1/(1-shape))}$. The "exponential" option uses $y(t) = y1 * \exp(-alpha*(t-t1))$. The exponential model does not estimate shape; its processed output includes shape = 1 as a fixed value to preserve the common output structure. Note: <code>prep_priors()</code> validates 5-element

	prior vector for power decay and 4-element prior vectors for exponential decay (the fifth (shape) prior is excluded in exponential decay).
nchain	An integer between 1 and 4 that specifies the number of MCMC chains to be run per jags model.
nadapt	An integer specifying the number of adaptations per chain.
nburn	An integer specifying the number of burn ins before sampling.
nmc	An integer specifying the number of samples in posterior chains.
niter	An integer specifying the number of iterations.
strat	A character string specifying the stratification variable, entered in quotes.
with_post	A logical value specifying whether a raw <code>jags.post</code> object should be included as an optional "jags.post" attribute on the returned <code>sr_model</code> tibble (see Value section below for details). Note: These objects can be large.
with_pop_params	A logical value specifying whether population level parameters should be included as an attribute entitled <code>population_params</code> . Excluded by default. Note: These objects can be large.
preclogy_per_iso	A logical value. When TRUE and <code>with_pop_params</code> is also TRUE, the Parameter column for <code>prec.logy</code> rows in <code>population_params</code> will contain the antigen/isotype label (e.g., "HlyE-IgA") rather than the constant "prec.logy". This allows grouping by Parameter to obtain per-isotype precision estimates directly. Default is FALSE (all <code>prec.logy</code> rows share <code>Parameter = "prec.logy"</code> ; the <code>Iso_type</code> column distinguishes isotypes).
...	Arguments passed on to prep_priors
max_antigens	An integer specifying how many antigen-isotypes (biomarkers) will be modeled.
mu_hyp_param	A numeric vector of 5 values representing the prior mean for the population level parameters parameters (y0, y1, t1, alpha, r) for each biomarker. Will be 5 values long specified by the user, representing the following parameters: • y0 = baseline antibody concentration • y1 = peak antibody concentration • t1 = time to peak • alpha = decay rate • r = shape parameter (If running <code>decay_type == "exponential"</code> no shape parameter needs to be specified). When <code>decay_type = "exponential"</code> only 4 parameters (y0, y1, t1, alpha) are used.
prec_hyp_param	A numeric vector of 5 values corresponding to hyperprior diagonal entries for the precision matrix (i.e. inverse variance) representing prior covariance of uncertainty around <code>mu_hyp_param</code> . Will be 5 values long corresponding to the 5 estimated parameters (4 values when <code>decay_type = "exponential"</code>).
omega_param	A numeric vector of 5 values corresponding to the diagonal entries representing the Wishart hyperprior distributions of <code>prec_hyp_param</code> ,

describing how much we expect parameters to vary between individuals. Will be 5 values long corresponding to the 5 estimated parameters (4 values when `decay_type = "exponential"`).

`wishdf_param` An [integer vector](#) of 1 value specifying the degrees of freedom for the Wishart hyperprior distribution of `prec_hyp_param`. Must be 1 value long.

- The value of `wishdf_param` controls how informative the Wishart prior is. Higher values lead to tighter priors on individual variation. Lower values (e.g., 5–10) make the prior more weakly informative, which can help improve convergence if the model is over-regularized.

`prec_logy_hyp_param` A [numeric vector](#) of 2 values corresponding to hyperprior diagonal entries on the log-scale for the precision matrix (i.e. inverse variance) representing prior beliefs of individual variation. Must be 2 values long.

Value

An `sr_model` class object: a subclass of [tibble::tbl_df](#) that contains MCMC samples from the joint posterior distribution of the model parameters, conditional on the provided input data, including the following:

- `Iteration` = Number of sampling iterations
- `Chain` = Number of MCMC chains run; between 1 and 4
- `Parameter` = Parameter being estimated. Includes the following:
 - `y0` = Posterior estimate of baseline antibody concentration
 - `y1` = Posterior estimate of peak antibody concentration
 - `t1` = Posterior estimate of time to peak
 - `alpha` = Posterior estimate of decay rate
 - `shape` = shape parameter, estimated for power decay and fixed at 1 for exponential decay
- `Iso_type` = Antibody/antigen type combination being evaluated
- `Stratification` = The variable used to stratify jags model
- `Subject` = ID of subject being evaluated
- `value` = Estimated value of the parameter
- The following [attributes](#) are included in the output:
 - `class`: Class of the output object.
 - `nChains`: Number of chains run.
 - `nParameters`: The amount of parameters estimated in the model.
 - `nIterations`: Number of iteration specified.
 - `nBurnin`: Number of burn ins.
 - `nThin`: Thinning number (`niter/nmc`).
 - `population_params`: Optionally included modeled population parameters, returned as a [data.frame](#) and excluded by default. Columns include `Iteration`, `Chain`, `Parameter`, `Iso_type`, `Stratification`, `Population_Parameter`, and `value`.
 - * `Population_Parameter` identifies which modeled population parameter is represented:

- `mu.par` = The population means of the host-specific model parameters (on logarithmic scales). Note: `y1` and `shape` are transformed; `shape` is not included for exponential decay.
- `prec.par` = The population precision matrix of the hyperparameters (with diagonal elements equal to inverse variances). The two parameters listed (separated by commas) represent the pairwise precision relationship between specified parameters.
- `prec.logy` = A vector of population precisions (inverse variances), one per antigen/isotype combination.
- `priors`: A [list](#) that summarizes the input priors, including:
 - * `mu_hyp_param`
 - * `prec_hyp_param`
 - * `omega_param`
 - * `wishdf`
 - * `prec_logy_hyp_param`
- `original_data`: The original input data, stored so that fitted and residual values can be computed on demand (e.g. by `plot_residuals()` via `calc_fit_mod()`). The attachment of `fitted_residuals` as an attribute was removed in version 0.1.0.9017.
- `strat`: The stratification variable name (or `NA` when no stratification was used), stored alongside `original_data` for the same purpose.
- `decay_type`: The decay function used ("power" or "exponential").
- An optional "`jags.post`" attribute, included when argument `with_post` = TRUE.

Author(s)

Sam Schildhauer

Examples

```
if (!is.element(runjags::findjags(), c("", NULL))) {
  library(runjags)
  set.seed(1)
  library(dplyr)
  strat1 <- serocalculator::typhoid_curves_nostrat_100 |>
    sim_case_data(n = 100) |>
    mutate(strat = "stratum 2")
  strat2 <- serocalculator::typhoid_curves_nostrat_100 |>
    sim_case_data(n = 100) |>
    mutate(strat = "stratum 1")

  dataset <- bind_rows(strat1, strat2)

  fitted_model <- run_serodynamics(
    data = dataset, # The data set input
    file_mod = serodynamics_example("model.jags"),
    nchain = 4, # Number of mcmc chains to run
    nadapt = 100, # Number of adaptations to run
    nburn = 100, # Number of unrecorded samples before sampling begins
    nmc = 1000,
```

```

    niter = 2000, # Number of iterations
    strat = "strat", # Variable to be stratified
    mu_hyp_param = c(1.0, 7.0, 1.0, -4.0, -1.0),
    prec_hyp_param = c(1.0, 0.00001, 1.0, 0.001, 1.0),
    omega_param = c(1.0, 50.0, 1.0, 10.0, 1.0),
    wishdf_param = 20,
    prec_logy_hyp_param = c(4.0, 1.0)
  )
}

```

serodynamics_example *Get path to an example file*

Description

The [serodynamics](#) package comes bundled with a number of sample files in its `inst/extdata` directory. This `serodynamics_example()` function make those sample files easy to access.

Usage

```
serodynamics_example(file = NULL)
```

Arguments

`file` Name of file. If `NULL`, the example files will be listed.

Details

Adapted from `readr::readr_example()` following the guidance in <https://r-pkgs.org/data.html#sec-data-example-path-helper>.

Value

a [character](#) string providing the path to the file specified by `file`, or a vector of available files if `file = NULL`.

Examples

```

serodynamics_example()
serodynamics_example(
  "SEES_Case_Nepal_ForSeroKinetics_02-13-2025.csv")

```

sim_case_data	<i>Simulate longitudinal case follow-up data from a homogeneous population</i>
---------------	--

Description

Simulate longitudinal case follow-up data from a homogeneous population

Usage

```
sim_case_data(
  n,
  curve_params,
  antigen_isos = get_biomarker_levels(curve_params),
  max_n_obs = 10,
  dist_n_obs = tibble::tibble(n_obs = 1:max_n_obs, prob = 1/max_n_obs),
  followup_interval = 7,
  followup_variance = 1
)
```

Arguments

n	integer number of cases to simulate
curve_params	a curve_params object from serocalculator::as_curve_params , assumed to be unstratified
antigen_isos	character vector : which antigen isotypes to simulate
max_n_obs	maximum number of observations
dist_n_obs	distribution of number of observations (tibble::tbl_df)
followup_interval	integer
followup_variance	integer

Value

a case_data object

Examples

```
set.seed(1)
serocalculator::typhoid_curves_nostrat_100 |>
  sim_case_data(n = 100)
```

summarize_posterior	<i>Summary Table of Jags Posterior Estimates</i>
---------------------	--

Description

summarize_posterior() takes an sr_model tibble returned by [run_serodynamics](#) to summary table for parameter, antigen/antibody, and stratification combination. Defaults will produce every combination of antigen/antibody, parameters, and stratifications, unless otherwise specified. Antigen/antibody combinations and stratifications will vary by analysis. The antibody dynamic curve includes the following parameters:

- y0 = baseline antibody concentration
- y1 = peak antibody concentration
- t1 = time to peak
- shape = shape parameter
- alpha = decay rate

Usage

```
summarize_posterior(
  data,
  iso = unique(data$Iso_type),
  param = unique(data$Parameter),
  strat = unique(data$Stratification)
)
```

Arguments

data	An sr_model tibble returned by run_serodynamics .
iso	Specify character string to produce tables of only a specific antigen/antibody combination, entered with quotes. Default outputs all antigen/antibody combinations.
param	Specify character string to produce tables of only a specific parameter, entered with quotes. Options include: • alpha = posterior estimate of decay rate • shape = posterior estimate of shape parameter • t1 = posterior estimate of time to peak • y0 = posterior estimate of baseline antibody concentration • y1 = posterior estimate of peak antibody concentration
strat	Specify character string to produce tables of specific stratification entered in quotes.

Value

A [data.frame](#) summarizing estimate mean, standard deviation (SD), median, and quantiles (2.5%, 25.0%, 50.0%, 75.0%, 97.5%).

Author(s)

Sam Schildhauer

Examples

```
summarize_posterior(data = serodynamics::nepal_sees_jags_output)
```

Index

* **datasets**
 nepal_sees, 7
 nepal_sees_jags_output, 7

add_serocurve_facets, 17
aes(), 4
annotation_borders(), 5
array, 23
as_case_data, 2
attributes, 23, 26
autoplot.case_data, 3

base::data.frame(), 7, 24
base::list(), 9

calc_fit_mod(), 13, 14, 27
character, 3, 9, 10, 15, 16, 18, 21, 23–25, 28–30

data.frame, 3, 21, 26, 30

FALSE, 17
fortify(), 4

ggplot(), 4
ggplot2::geom_line, 4
ggplot2::geom_point, 4
ggplot2::ggplot, 5, 10, 12, 14, 15, 17, 19
ggplot2::ggplot(), 9
grid::arrow(), 5

initsfunction, 5
integer, 5, 12, 17, 22, 25, 26, 29

layer position, 4
layer stat, 4
list, 6, 8–10, 14, 15, 18, 19, 21, 23, 27
load_data, 6
log10, 11, 12, 16
logical, 11, 12, 14, 16, 17, 25

matrix, 23

NA, 27
nepal_sees, 7, 7
nepal_sees_jags_output, 7
NULL, 17
numeric, 22, 23, 25, 26

plot_density, 8
plot_ess, 9
plot_predicted_curve, 11
plot_residuals, 13
plot_residuals(), 27
plot_rhat, 14
plot_serocurve, 16
plot_trace, 18
postprocess_jags_output, 19
prep_data, 21
prep_priors, 22, 25

readr::readr_example(), 28
rjags::jags.model(), 24
run_serodynamics, 11, 24, 30
run_serodynamics(), 7–10, 13–16, 18
runjags::run.jags(), 19, 24

serocalculator::as_curve_params, 29
serodynamics, 28
serodynamics_example, 28
sim_case_data, 29
summarize_posterior, 30

tibble::tbl_df, 8, 11, 19, 26, 29
TRUE, 11, 12, 16, 17

vector, 22, 23, 25, 26, 29