Package 'idmodelr'

October 13, 2022

Type Package

Version 0.4.0

Title Infectious Disease Model Library and Utilities

Description Explore a range of infectious disease models in a consistent framework. The primary aim of 'idmodelr' is to provide a library of infectious disease models for researchers, students, and other interested individuals. These models can be used to understand the underlying dynamics and as a reference point when developing models for research. 'idmodelr' also provides a range of utilities. These include: plotting functionality; a simulation wrapper; scenario analysis tooling; an interactive dashboard; tools for handling mult-dimensional models; and both model and parameter look up tables. Unlike other modelling packages such as 'pomp' (<https://kingaa.github.io/pomp/>), 'libbi' (<http://libbi.org>) and 'EpiModel' (<http://www.epimodel.org>), 'idmodelr' serves primarily as an educational resource. It is most comparable to epirecipes (<http://epirecip.es/epicookbook/chapters/simple>) but provides a more consistent framework, an R based workflow, and additional utility tooling. After users have explored model dynamics with 'idmodelr' they may then implement their model using one of these packages in order to utilise the model fitting tools they provide. For newer modellers, this package reduces the barrier to entry by containing multiple infectious disease models, providing a consistent framework for simulation and visualisation, and signposting towards other, more research focussed, resources.

License GPL-3

Depends R (>= 3.3.0)

Imports dplyr (>= 0.8.3), rlang (>= 0.4.0), ggplot2 (>= 3.2.0), viridis (>= 0.5.1), magrittr (>= 1.5), purrr (>= 0.3.2), future (>= 1.14.0), furrr (>= 0.1.0), stringr (>= 1.4.0), tibble (>= 2.1.3), tidyr (>= 0.8.3), deSolve (>= 1.23)

Suggests testthat (>= 2.1.1), rmarkdown (>= 1.13), knitr (>= 1.23), pkgnet (>= 0.4.0), DT (>= 0.7), vdiffr (>= 0.3.1), spelling

VignetteBuilder knitr

URL https://samabbott.co.uk/idmodelr/, https://github.com/seabbs/idmodelr

BugReports https://github.com/seabbs/idmodelr/issues

Encoding UTF-8

RoxygenNote 7.2.1

LazyData true

Language en-GB

NeedsCompilation no

Author Sam Abbott [aut, cre] (<https://orcid.org/0000-0001-8057-8037>),
Akira Endo [aut] (<https://orcid.org/0000-0001-6377-7296>)

Maintainer Sam Abbott <contact@samabbott.co.uk>

Repository CRAN

Date/Publication 2022-09-02 13:10:02 UTC

R topics documented:

`add_pointer_struct` *Adds Pointer Structure to R objects*

Description

Adds Pointer Structure to R objects

Usage

```
add_pointer_struct(char, length)
```

Arguments

`char`	A character vector.
`length`	The length of the returned object.

Value

A character vector of the concatenated character string with sequential numbering

Examples

```
## For a single variable
add_pointer_struct("S", 3)

## For multiple variables
add_pointer_struct(c("S", "I"), length = 3)
```

aggregate_model *A Function to Apply Multiple Aggregations to Model Output*

Description

A Function to Apply Multiple Aggregations to Model Output

Usage

```
aggregate_model(
  df,
  aggregate_to = NULL,
  compartments = NULL,
  strat = NULL,
  hold_out_var = NULL,
  id_col = NULL,
  groups = NULL,
  new_var = "incidence",
  total_pop = TRUE,
  summary_var = FALSE,
  test = FALSE
)
```

Arguments

df	A dataframe of Model Output.
aggregate_to	A character vector or list specifying the aggregation operations to perform on the model output. Operations are carried out in the order specified. Implemented options are; disease, demographic, and incidence.
compartments	A character vector or list specifying the unique compartments to aggregate. May either be specified once for all aggregation functions or for each function separately.
strat	The number of stratified groups in the model.
hold_out_var	A character vector or list specifying the unique compartments that will not be aggregated. May either be specified once for all aggregation functions or for each function separately. If compartments is set then this argument does not need to be used.
id_col	A character string containing the name of the new id column.
groups	A character vector with length equal to the level of stratification. Used to name the stratified levels.
new_var	A character vector specifying the new variable to add when aggregating incidence.
total_pop	A logical (defaults to TRUE) indicating if the total population should be calculated when summarising the model demographics.

summary_var	A logical (defaults to FALSE), specifying whether to add an additional summary variable across all stratified levels.
test,	Logical defaults to FALSE. For testing, returns the processed inputs rather than performing the aggregation.

Value

An aggregated dataframe.

See Also

aggregate_model aggregate_model_internal combine_to_age_model combine_strat_model_output summarise_var_by_strat

Examples

```
df <- data.frame(A1 = 1, B1 = 1, A2 = 1, B2 = 1, A3 = 1, B3 = 1)
aggregate_model(df, aggregate_to = "incidence",
                compartments = c("A", "B"), strat = 3,
                summary_var = TRUE)
```

---

aggregate_model_internal

An Internal Function to Aggregate Model Output Using other Idmodelr functions.

Description

An Internal Function to Aggregate Model Output Using other Idmodelr functions.

Usage

```
aggregate_model_internal(
  df,
  aggregate_to = NULL,
  compartments = NULL,
  strat = NULL,
  hold_out_var = NULL,
  new_var = "incidence",
  id_col = NULL,
  groups = NULL,
  total_pop = TRUE,
  summary_var = FALSE
)
```

Arguments

df	A dataframe of Model Output.
aggregate_to	A character vector specifying the aggregation function to apply possible values are; disease, demographic, or incidence.
compartments	A character vector specifying the unique compartments to aggregate.
strat	The number of stratified groups in the model.
hold_out_var	A character vector specifying the unique compartments not to aggregate.
new_var	A character vector specifying the new variable to add when aggregating incidence.
id_col	A character string containing the name of the new id column.
groups	A character vector with length equal to the level of stratification. Used to name the stratified levels.
total_pop	A logical (defaults to TRUE) indicating if the total population should be calculated when summarising the model demographics.
summary_var	A logical (defaults to FALSE), specifying whether to add an additional summary variable across all stratified levels.

Value

An aggregated dataframe.

See Also

aggregate_model aggregate_model_internal combine_to_age_model combine_strat_model_output summarise_var_by_strat

Examples

```
df <- data.frame(time = 1, A1 = 1, B1 = 1, A2 = 1, B2 = 1, A3 = 1, B3 = 1)

## Incidence
aggregate_model_internal(df, aggregate_to = "incidence",
                         compartments = c("A", "B"), strat = 3,
                         summary_var = TRUE)

## Demographic
aggregate_model_internal(df, aggregate_to = "demographic",
                         compartments = c("A", "B"), strat = 3,
                         summary_var = TRUE)
## Disease
aggregate_model_internal(df, aggregate_to = "disease",
                         compartments = c("A", "B"), strat = 3,
                         summary_var = TRUE)
## Tidy (long)
aggregate_model_internal(df, aggregate_to = "tidy",
                         compartments = c("A", "B"), hold_out_var = "time", strat = 3,
                         summary_var = TRUE, id_col = "Age",
                         groups = c("Children", "Young adults", "Adults"))
```

combine_strat_model_output *Reduces the Dimensionality of a Stratified Model*

Description

Reduces the dimensions of stratified model output. Default behaviour is to remove stratification for all variables. However, variables to reduce the dimensions of can be selected, as can variables to preserve with there structure intact.

Usage

```
combine_strat_model_output(
  df,
  strat = NULL,
  compartments = NULL,
  hold_out_var = NULL
)
```

Arguments

df	A data frame with variables stratified using numeric labels.
strat	An integer specifying the number of stratifications to reduce.
compartments	A character vector specifying the unique population compartments.
hold_out_var	A character vector specifying the variables to keep unchanged. Defaults to NULL

Value

A dataframe of model output that has its dimensionality reduced

See Also

combine_to_age_model

Examples

```
df <- data.frame(S1 = NA, S2 = NA, S3 = NA, time = NA)
combine_strat_model_output(df, 3, compartments = "S", hold_out_var = "time")
```

combine_to_age_model *Combine an Infectious Disease Model To a Demographic Model*

Description

Similarly to combine_strat_model_output this functions reduces the dimension of model output into just the demographic components.

Usage

```
combine_to_age_model(
  df,
  age_com = NULL,
  compartments = NULL,
  hold_out_var = NULL,
  total_pop = TRUE
)
```

Arguments

df	A dataframe of model output.
age_com	Integer indicating the number of age compartments.
compartments	A character vector of the disease model compartments to combine.
hold_out_var	A character vector specifying the variables to keep unchanged. Defaults to NULL
total_pop	A logical indicating whether to calculate the total population. Defaults to true.

Value

A dataframe which summarises the demographic process of a model.

See Also

combine_strat_model_output

Examples

```
df <- data.frame(S1 = c(1,2), S2 = c(1, 3), E1 = c(4, 1), E2 = c(3, 4), time = c(1, 2))

combine_to_age_model(df, age_com = 2, hold_out_var = "time")
```

`estimate_norm_dist_from_ci` *A Function to Estimate a Normal Distribution from Credible or Confidence Intervals*

Description

A Function to Estimate a Normal Distribution from Credible or Confidence Intervals

Usage

```
estimate_norm_dist_from_ci(
  lower_interval = NULL,
  upper_interval = NULL,
  interval = "95%"
)
```

Arguments

`lower_interval`	Numeric, the lower CI.
`upper_interval`	Numeric, the upper CI
`interval`	A character string indicating the percentage interval the CI represents.

Value

A dataframe containing the mean and standard deviation of the normal distribution summarised by the provided CI's.

Examples

```
## Run function to estimate normal distribution for a 95% CI of 1, to 2

df <- estimate_norm_dist_from_ci(1,2)

## Check

x <- rnorm(10000, df$mean, df$sd)

quantile(x, c(0.025, 0.975))
```

gather_strat_multi_variable *A Function to Gather Multiple Stratified Variables into a Tidy Format*

Description

A Function to Gather Multiple Stratified Variables into a Tidy Format

Usage

```
gather_strat_multi_variable(
  df,
  id_col,
  compartments = NULL,
  hold_out_var = NULL,
  strat = NULL,
  groups = NULL
)
```

Arguments

df	A data frame with variables stratified using numeric labels.
id_col	A character string containing the name of the new id column.
compartments	A character vector specifying the unique population compartments.
hold_out_var	A character vector specifying the variables to keep unchanged. Defaults to NULL
strat	An integer specifying the number of stratifications to reduce.
groups	A character vector with length equal to the level of stratification. Used to name the stratified levels.

Value

A dataframe of stratified model output with multiple Tidy variables.

Examples

```
df <- data.frame(time = 0, A1 = 1, A2 = 2, A3 = 3, B1 = 2, B2 = 3, B3 = 0)
gather_strat_multi_variable(df, id_col = "Age", compartment = c("A", "B"), hold_out_var = "time",
                            strat = 3, groups = c("Children", "Young adults", "Adults"))
```

gather_strat_variable *A Function to Gather a Stratified Variable into a Tidy Format*

Description

A Function to Gather a Stratified Variable into a Tidy Format

Usage

```
gather_strat_variable(
  df,
  id_col,
  compartment,
  hold_out_var = NULL,
  strat,
  groups = NULL
)
```

Arguments

df	A data frame with variables stratified using numeric labels.
id_col	A character string containing the name of the new id column.
compartment	The compartment to reduce the dimension of.
hold_out_var	A character vector specifying the variables to keep unchanged. Defaults to NULL
strat	An integer specifying the number of stratifications to reduce.
groups	A character vector with length equal to the level of stratification. Used to name the stratified levels.

Value

A dataframe of stratified model output with a single Tidy variable.

Examples

```
df <- data.frame(time = 0, A1 = 1, A2 = 2, A3 = 3)
gather_strat_variable(df, id_col = "Age", compartment = "A",
                      strat = 3, groups = c("Children", "Young adults", "Adults"))
```

`generate_parameter_permutations` *A Function to Generate Parameter Permutations*

Description

A function to generate parameter permutations from a generic sampling function (or if not given from the input parameters). This function can be used to rapidly generate new parameter combinations given parameters to be varied, and scenarios to be investigated.

Usage

```
generate_parameter_permutations(
  variable_params = NULL,
  fixed_params = NULL,
  sample_params = NULL,
  excluded_params = NULL,
  scenarios = NULL,
  sampling_function = NULL,
  parameter_samples = 1,
  repeat_sample = TRUE,
  rerun = FALSE,
  ...
)
```

Arguments

`variable_params`
: A dataframe containing any parameter variations to be investigated. If these parameters would normally be sampled then they should be added to the excluded_params argument.

`fixed_params`
: A named vector of parameters that are not sampled by the sampling function. If these parameters would usually be sampled then they should be added to the excluded_params argument.

`sample_params`
: A named vector of parameters to be sampled. If a sampling function is not supplied these parameters will be used in the final permutation dataframe.

`excluded_params`
: A character vector indicating which parameters should have there sampled values kept.

`scenarios`
: A dataframe of possible scenarios to investigate. It must contain a scenario variable to identify each separate scenarios. If parameters are included here that would normally be sampled then they should be added to the excluded_params argument

`sampling_function`
: A sampling function, this should be designed such that it's input is a matrix with each parameter having a named row. It should return it's output in the same

format. If not supplied defaults to passing through parameters, this may not be the required behaviour.

`parameter_samples`
: The number of parameter samples to take, defaults to one.

`repeat_sample`
: A logical (defaults to `TRUE`) which indicates if each scenario should independently sample from the sampling function. If set to `FALSE` then each scenario will share the same sampled parameter set.

`rerun`
: A logical indicating if the function should be rerun or saved results should be loaded. Defaults to `FALSE`.

`...`
: Additional arguments to be passed to the sampling_function.

Value

A dataframe containing sampled parameter permutations

Examples

```
scenarios <- data.frame(scenario = c("test_1", "test_2"), scenario_param = c(0, 1))
variable_params <-  data.frame(variable = c(0, 0.5, 1))
fixed_params <- c(fixed_1 = 2, fixed_2 = c(1, 3, 4))
sample_params <- c(sample_1 = 2, sample_2 = c(2, 1))

generate_parameter_permutations(variable_params, fixed_params, sample_params,
                                excluded_params = c("variable"), scenarios,
                                parameter_samples = 1)
```

`model_details` *Model Details*

Description

Details on models implemented in `idmodelr`.

Usage

```
model_details
```

Format

A data frame with 22 rows and 14 variables.

model Name of the model function.

model_family Name of the model family (i.e. SIR)

time Discrete or continuous time

type Deterministic or stochastic

recovered Does this model included a recovered population (yes/no).

exposed Does this model included an exposed population (yes/no).

treated Does this model included a treated population (yes/no).

susceptible Does this model included a post infection susceptible population (yes/no).

risk_stratified Is this model risk stratified (yes/no).

non_exponential Does this model contain non-exponential rates (yes/no).

simple_demographics Does this model contain simple (i.e. births = deaths) demographics (yes/no).

vaccination Does this model include vaccination (yes/no).

disease_example Which diseases (if any) can this model be used for.

language What language is this model written in (i.e. R, C etc.).

parameters A list of parameters required by the model.

`model_df_to_vector` *Extracts a Single Column, Summarises if from Simulation*

Description

Extracts a Single Column, Summarises if from Simulation

Usage

```
model_df_to_vector(df, com_var, id_var = NULL, sum_fn = NULL)
```

Arguments

`df`	A data frame of dynamic system output.
`com_var`	The vector to be compared; use unquoted name (NSE).
`id_var`	A character string indicating the id variable to summarise over if required.
`sum_fn`	The summary function to be used, defaults to median.

Value

Returns a numeric vector, summarised if required.

Examples

```
library(dplyr)
## Extract a vector with no repeats
model_df_to_vector(iris, Petal.Length)

## Extract a vector and summarise
df <- bind_rows(iris %>% mutate(sim = 1, id = 1:length(sim)),
 iris %>% mutate(sim = 2 , id = 1:length(sim)))

model_df_to_vector(df, Petal.Length, "id", sum_fn = mean)
```

parameter_details *Parameter Details*

Description

Details on the parameters used in models implemented in `idmodelr`.

Usage

```
parameter_details
```

Format

A data frame with 14 rows and 6 variables.

parameter Name of the parameter.

parameter_family Descriptive parameter family (i.e. transmission).

description What is this parameter.

type What type of parameter is this (i.e rate, proportion, probability etc.).

risk_stratified Is this parameter used in risk stratified models (yes/no).

non_exponential Is this parameter a non-exponential rate (yes/no).

plot_model *Plot Compartment Populations over Time for a Model Simulation*

Description

Make separate plots for each model compartment. Assumes model output is structured as that produced from `solve_ode`.

Usage

```
plot_model(sim, prev_sim = NULL, model_labels = NULL, facet = TRUE)
```

Arguments

`sim`	A tibble of model output as formatted by `solve_ode`. Optionally a list of simulations can be passed when comparing multiple model runs.
`prev_sim`	A second tibble of model output formatted as for `sim`. Used to compare to model runs. Can only be supplied if `sim` is not a list.
`model_labels`	A character vector of model names. Defaults to `c("Current", "Previous")` when two model simulations are used and the list names when `sim` is a list. If `sim` is unnamed the index of the list is used.
`facet`	Logical, defaults to `TRUE`. If `FALSE` then the plot will not be faceted otherwise it will be.

Value

A Plot of each model compartments population over time.

Examples

```
## Intialise
N = 100000
I_0 = 1
S_0 = N - I_0
R_0 = 1.1
beta = R_0

##Time for model to run over
tbegin = 0
tend = 50
times <- seq(tbegin, tend, 1)

##Vectorise input
parameters <- as.matrix(c(beta = beta))
inits <- as.matrix(c(S = S_0, I = I_0))

sim <- solve_ode(model = SI_ode, inits, parameters, times, as.data.frame = TRUE)

plot_model(sim, facet = FALSE)

plot_model(sim, facet = TRUE)

## Compare with an updated model run

#'## Intialise
R_0 = 1.3
beta = R_0
parameters <- as.matrix(c(beta = beta))

new_sim <- solve_ode(model = SI_ode, inits, parameters, times, as.data.frame = TRUE)


plot_model(new_sim,sim, facet = FALSE)

plot_model(new_sim, sim, facet = TRUE)

## Passing in the simulations as a list
plot_model(list("Current" = new_sim, "Previous" = sim), facet = TRUE)
```

`required_parameters` *Lookup the Details of Parameters Required by a Model*

Description

This function simplifies the process of checking which parameters a given `idmodelr` model depends on. It is effectively an interface to `parameter_details` via `model_details`. As fuzzy matching has been used it can also given information of the parameter requirements of a subset of the available models.

Usage

```
required_parameters(model = NULL)
```

Arguments

`model`	A character string containing the name of the model of interest. Defaults to `NULL`.

Value

A dataframe extracted from `parameter_details` containing the details of the parameters required by the model of interest.

Examples

```
## Check the parameters required by the "SIR_ode" model
required_parameters("SIR_ode")


## Use fizzy matching to look at parameters for all SIR models
required_parameters("SIR")
```

`scenario_analysis` *A Function to Perform Scenario Analysis for a Generic Model Object.*

Description

This function uses parameter permutations produced by `generate_parameter_permutations` to simulate from a supplied model function. It can be used to examine multiple scenarios, with any number of parameter variations, for multiple samples.

Usage

```
scenario_analysis(
  parameter_df,
  variable_params = NULL,
  model = NULL,
  sim_fn = NULL,
  summary_fn = NULL,
  cores = 1,
```

```
  rerun = FALSE,
  verbose = FALSE,
  by_row = FALSE,
  test = FALSE,
  ...
)
```

Arguments

parameter_df	A dataframe of parameter permutations as produced by generate_parameter_permutations. Using the default options it will save results when run for the first time, and afterwards load them in.
variable_params	A character vector containing the names of the parameters that are varied in parameter_df.
model	A model compatible with your sim_fn.
sim_fn	A generic simulation function, with the first argument as the model object, a params argument, and a as.data.frame argument.
summary_fn	A function which accepts a single dataframe argument customised to fit with the standard output of scenario_analysis and your simulate_model function. Defaults to NULL for which no summarisation takes place. Warning: If a previous analysis has been saved, changing this option will not summarise the result. The analysis must be rerun.
cores	The number of cores to use for the scenario analysis, defaults to 1.
rerun	A logical indicating if the function should be rerun or saved results should be loaded. Defaults to FALSE.
verbose	Logical (defaults to FALSE) indicating if progress information should be printed to the console.
by_row	Logical (defaults to FALSE) indicating if inputted parameters should be inputted as a block to sim_fn or individually. If TRUE then function will always return a tibble. Does not currently work with sim_fn that produces multiple simulations for a single parameter set - for this scenario a block based approach or post processing is required.
test	A logical (defaults to FALSE) if TRUE function uses multicore functionality regardless of the number of cores specified.
...	Pass additional arguments to sim_fn. Only implemented when a single core is used.

Value

A tidy dataframe containing simulated model trajectories for each scenario varied parameter combination. Use ‘tidyr::unnest“ to examine all simulation results.

Examples

```
scenarios <- data.frame(scenario = c("test_1", "test_2"), scenario_param = c(0, 1))
variable_params <-  data.frame(variable = c(0, 0.5, 1))
fixed_params <- c(fixed_1 = 2, fixed_2 = c(1, 3, 4))
sample_params <- c(sample_1 = 2, sample_2 = c(2, 1))

parameter_df <- generate_parameter_permutations(variable_params, fixed_params, sample_params,
                                                excluded_params = c("variable"), scenarios,
                                                   parameter_samples = 10)

## set up dummy simulation function (returning an empty dataframe)
dummy_sim_fn <- function(object, inits, params, times, as.data.frame) {
 x <- tibble::tibble()
 return(x)
}

## Set up dummy summary function
dummy_sum_fn <- function(df){
df <- dplyr::mutate(df, summarised_simulations = simulations)

return(df)
}
dummy_model <- function(){}

## Run scenario analysis
scenario_analysis(parameter_df, variable_params = "variable", model = dummy_model,
                  sim_fn = dummy_sim_fn, cores = 1, summary_fn = dummy_sum_fn)
```

`SEIRS_demographics_ode` *Susceptible-Exposed-Infected-Recovered-Susceptible Model with Simple Demographics*

Description

Susceptible-Exposed-Infected-Recovered-Susceptible Model with Simple Demographics

Usage

```
SEIRS_demographics_ode(t, x, params)
```

Arguments

`t`	The timestep over which to calculate derivatives
`x`	A numeric vector of compartment populations.
`params`	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
E_0 <- 10
I_0 <- 1
R_0 <- 0
beta <- 3
gamma <-  1/2
tau <- 2
chi <- 0.5
mu <- 1/81

parameters <- c(beta = beta, gamma = gamma, chi = chi,
                tau = tau, mu = mu)
inits <- c(S = S_0, E = E_0, I = I_0, R_0 = R_0)

SEIRS_demographics_ode(1, inits, parameters)
```

SEIRS_ode *Susceptible-Exposed-Infected-Recovered-Susceptible Model*

Description

Susceptible-Exposed-Infected-Recovered-Susceptible Model

Usage

```
SEIRS_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
E_0 <- 10
I_0 <- 1
R_0 <- 0
beta <- 3
gamma <-  1/2
```

```
tau <- 2
chi <- 0.5

parameters <- c(beta = beta, gamma = gamma,
                chi = chi, tau = tau)
inits <- c(S = S_0, E = E_0, I = I_0, R_0 = R_0)

SEIRS_ode(1, inits, parameters)
```

`SEIR_demographics_ode` *Susceptible-Exposed-Infected-Recovered Model with Simple Demographics*

Description

Susceptible-Exposed-Infected-Recovered Model with Simple Demographics

Usage

```
SEIR_demographics_ode(t, x, params)
```

Arguments

`t`	The timestep over which to calculate derivatives
`x`	A numeric vector of compartment populations.
`params`	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
E_0 <- 10
I_0 <- 1
R_0 <- 0
beta <- 3
gamma <-  1/2
tau <- 2
mu <- 1/81

parameters <- c(beta = beta, gamma = gamma, tau = tau, mu = mu)
inits <- c(S = S_0, E = E_0, I = I_0, R_0 = R_0)

SEIR_demographics_ode(1, inits, parameters)
```

SEIR_ode *Susceptible-Exposed-Infected-Recovered Model*

Description

Susceptible-Exposed-Infected-Recovered Model

Usage

```
SEIR_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
E_0 <- 10
I_0 <- 1
R_0 <- 0
beta <- 3
gamma <-  1/2
tau <- 2

parameters <- c(beta = beta, gamma = gamma, tau = tau)
inits <- c(S = S_0, E = E_0, I = I_0, R_0 = R_0)

SEIR_ode(1, inits, parameters)
```

SEIS_demographics_ode *Susceptible-Exposed-Infected-Susceptible Model with Simple Demographics*

Description

Susceptible-Exposed-Infected-Susceptible Model with Simple Demographics

Usage

```
SEIS_demographics_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
E_0 <- 10
I_0 <- 1
beta <- 3
chi <- 0.5
gamma <- 1/2
mu <- 1/81
parameters <- c(beta = beta, gamma = gamma,
 chi = chi, mu = mu)
inits <- c(S = S_0, E = E_0, I = I_0)

SEIS_demographics_ode(1, inits, parameters)
```

SEIS_ode *Susceptible-Exposed-Infected-Susceptible Model*

Description

Susceptible-Exposed-Infected-Susceptible Model

Usage

```
SEIS_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
E_0 <- 10
I_0 <- 1
beta <- 3
chi <- 0.5
gamma = 1/2
parameters <- c(beta = beta, gamma = gamma, chi = chi)
inits <- c(S = S_0, E = E_0, I = I_0)

SEIS_ode(1, inits, parameters)
```

SEI_demographics_ode *Susceptible-Exposed-Infected Model with Simple Demographics*

Description

Susceptible-Exposed-Infected Model with Simple Demographics

Usage

```
SEI_demographics_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
E_0 <- 10
I_0 <- 1
beta <- 3
gamma <- 1/2
mu <- 1/81
```

```
parameters <- c(beta = beta, gamma = gamma, mu = mu)
inits <- c(S = S_0, E = E_0, I = I_0)

SEI_demographics_ode(1, inits, parameters)
```

SEI_ode *Susceptible-Exposed-Infected Model*

Description

Susceptible-Exposed-Infected Model

Usage

```
SEI_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
E_0 <- 10
I_0 <- 1
beta <- 3
gamma = 1/2

parameters <- c(beta = beta, gamma = gamma)
inits <- c(S = S_0, E = E_0, I = I_0)

SEI_ode(1, inits, parameters)
```

SHLIR_demographics_ode *Susceptible-High-risk-latent-Low-risk-latent-Infected-Recovered Model with Simple Demographics*

Description

Susceptible-High-risk-latent-Low-risk-latent-Infected-Recovered Model with Simple Demographics

Usage

```
SHLIR_demographics_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
H_0 <- 10
L_0 <- 0
I_0 <- 1
R_0 <- 0
beta = 3 # Rate of transmission
gamma_H = 1/5 # Rate of progression to active symptoms from high risk latent
nu = 1/2 #Rate of progression from high to low risk latent
gamma_L = 1/100 # Rate of progression to active symptoms for low risk latent
tau = 1/2 # Rate of recovery
mu = 1/81 # Rate of natural mortality
parameters <- c(beta = beta, gamma_H = gamma_H, gamma_L = gamma_L, nu = nu, tau = tau, mu = mu)
inits <- c(S = S_0, H = H_0, L = L_0, I = I_0, R_0 = R_0)

SHLIR_demographics_ode(1, inits, parameters)
```

SHLIR_ode *Susceptible-High-risk-latent-Low-risk-latent-Infected-Recovered Model*

Description

Susceptible-High-risk-latent-Low-risk-latent-Infected-Recovered Model

Usage

```
SHLIR_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
H_0 <- 10
L_0 <- 0
I_0 <- 1
R_0 <- 0
beta = 3 # Rate of transmission
gamma_H = 1/5 # Rate of progression to active symptoms from high risk latent
nu = 1/2 #Rate of progression from high to low risk latent
gamma_L = 1/100 # Rate of progression to active symptoms for low risk latent
tau = 1/2 # Rate of recovery

parameters <- c(beta = beta, gamma_H = gamma_H, gamma_L = gamma_L, nu = nu, tau = tau)
inits <- c(S = S_0, H = H_0, L = L_0, I = I_0, R_0 = R_0)

SHLIR_ode(1, inits, parameters)
```

SHLITR_demographics_ode *Susceptible-High-risk-latent-Low-risk-latent-Infected-Treated-Recovered Model with Simple Demographics*

Description

A more complex SHLIR model flow diagram, treatment,reinfection, and simple demographics for those who have recovered from active disease.

Usage

```
SHLITR_demographics_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
## initialise
inits <- c(
# General population
S = 800,
H = 0,
L = 0,
I = 0,
Tr = 0,
R = 0
)

parameters <- c(
beta = 3, # Rate of transmission
gamma_H = 1/5, # Rate of progression to active symptoms from high risk latent
nu = 1/2, #Rate of progression from high to low risk latent
gamma_L = 1/100, # Rate of progression to active symptoms for low risk latent
epsilon = 1/3, # Rate of treatment
tau = 1/2, # Rate of recovery
mu = 1/81 # Rate of natural mortality
)
```

```
SHLITR_demographics_ode(1, inits, parameters)
```

SHLITR_ode *Susceptible-High-risk-latent-Low-risk-latent-Infected-Treated-Recovered Model*

Description

A more complex SHLIR model flow diagram, treatment, and reinfection for those who have recovered from active disease.

Usage

```
SHLITR_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
## initialise
inits <- c(
# General population
S = 800,
H = 0,
L = 0,
I = 0,
Tr = 0,
R = 0
)

parameters <- c(
beta = 3, # Rate of transmission
gamma_H = 1/5, # Rate of progression to active symptoms from high risk latent
nu = 1/2, #Rate of progression from high to low risk latent
gamma_L = 1/100, # Rate of progression to active symptoms for low risk latent
epsilon = 1/3, # Rate of treatment
tau = 1/2 # Rate of recovery
)
```

```
SHLITR_ode(1, inits, parameters)
```

`SHLITR_risk_demographics_ode` *Susceptible-High-risk-latent-Low-risk-latent-Infected-Treated-Recovered Model with Demographics, Stratified by Risk*

Description

A more complex SHLIR model flow diagram, with risk groups, treatment, and reinfection for those who have recovered from active disease

Usage

```
SHLITR_risk_demographics_ode(t, x, params)
```

Arguments

`t`	The timestep over which to calculate derivatives
`x`	A numeric vector of compartment populations.
`params`	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
## initialise
inits <- c(
# General population
S = 800,
H = 0,
L = 0,
I = 0,
Tr = 0,
R = 0,
## High risk population
S_H = 199,
H_H = 0,
L_H = 0,
I_H = 1,
Tr_H = 0,
R_H = 0
)

parameters <- c(
```

```
beta = 3, # Rate of transmission
beta_H = 6, # High risk rate of transmission
gamma_H = 1/5, # Rate of progression to active symptoms from high risk latent
nu = 1/2, #Rate of progression from high to low risk latent
gamma_L = 1/100, # Rate of progression to active symptoms for low risk latent
epsilon = 1/3, # Rate of treatment
tau = 1/2, # Rate of recovery
mu = 1/81, # Rate of natural mortality
p = 0.2, # proportion of new births that are high risk
M = 0.2 # Between group mixing
)

SHLITR_risk_demographics_ode(1, inits, parameters)
```

`SHLITR_risk_ode` *Susceptible-High-risk-latent-Low-risk-latent-Infected-Treated-Recovered Model, Stratified by Risk*

Description

A more complex SHLIR model flow diagram, with risk groups, treatment, and reinfection for those who have recovered from active disease

Usage

```
SHLITR_risk_ode(t, x, params)
```

Arguments

`t`	The timestep over which to calculate derivatives
`x`	A numeric vector of compartment populations.
`params`	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
## initialise
inits <- c(
# General population
S = 800,
H = 0,
L = 0,
I = 0,
Tr = 0,
```

```
R = 0,
## High risk population
S_H = 199,
H_H = 0,
L_H = 0,
I_H = 1,
Tr_H = 0,
R_H = 0
)

parameters <- c(
beta = 3, # Rate of transmission
beta_H = 6, # High risk rate of transmission
gamma_H = 1/5, # Rate of progression to active symptoms from high risk latent
nu = 1/2, #Rate of progression from high to low risk latent
gamma_L = 1/100, # Rate of progression to active symptoms for low risk latent
epsilon = 1/3, # Rate of treatment
tau = 1/2, # Rate of recovery
M = 0.2 # Between group mixing
)

SHLITR_risk_ode(1, inits, parameters)
```

`simulate_model` *A Function to Simulate a Model from a Generic Simulation Function, with Pre and Post Processing*

Description

A Function to Simulate a Model from a Generic Simulation Function, with Pre and Post Processing

Usage

```
simulate_model(
  model,
  sim_fn,
  inits = NULL,
  params = NULL,
  times = NULL,
  as_tibble = TRUE,
  by_row = FALSE,
  aggregate_to = NULL,
  compartments = NULL,
  strat = NULL,
  hold_out_var = NULL,
  new_var = "incidence",
  total_pop = TRUE,
  summary_var = FALSE,
```

```
  verbose = FALSE,
  ...
)
```

Arguments

model	A model compatible with your sim_fn.
sim_fn	A generic simulation function, with the first argument as the model object, a params argument, and a as.data.frame argument.
inits	A dataframe of initial conditions, optionally a named vector can be used.
params	A dataframe of parameters, with each parameter as a variable. Optionally a named vector can be used.
times	A vector of the times to sample the model for, from a starting time to a final time.
as_tibble	Logical (defaults to TRUE) indicating if the output should be returned as a tibble, otherwise returned as the default sim_fn output.
by_row	Logical (defaults to FALSE) indicating if inputted parameters should be inputted as a block to sim_fn or individually. If TRUE then function will always return a tibble. Does not currently work with sim_fn that produces multiple simulations for a single parameter set - for this scenario a block based approach or post processing is required.
aggregate_to	A character vector or list specifying the aggregation operations to perform on the model output. Operations are carried out in the order specified. Implemented options are; disease, demographic, and incidence.
compartments	A character vector or list specifying the unique compartments to aggregate. May either be specified once for all aggregation functions or for each function separately.
strat	The number of stratified groups in the model.
hold_out_var	A character vector or list specifying the unique compartments that will not be aggregated. May either be specified once for all aggregation functions or for each function separately. If compartments is set then this argument does not need to be used.
new_var	A character vector specifying the new variable to add when aggregating incidence.
total_pop	A logical (defaults to TRUE) indicating if the total population should be calculated when summarising the model demographics.
summary_var	A logical (defaults to FALSE), specifying whether to add an additional summary variable across all stratified levels.
verbose	Logical (defaults to FALSE) indicating if progress information should be printed to the console.
...	Additional arguments to pass to sim_fn

Value

Trajectories as a tibble, optionally returns the default sim_fn output.

See Also

aggregate_model

Examples

```
## Intialise
N = 100000
I_0 = 1
S_0 = N - I_0
R_0 = 1.1
beta = R_0

##Time for model to run over
tbegin = 0
tend = 50
times <- seq(tbegin, tend, 1)

##Vectorise input
parameters <- data.frame(beta = beta)
inits <- data.frame(S = S_0, I = I_0)

SI_sim <- simulate_model(model = SI_ode, sim_fn = solve_ode, inits, parameters, times)
```

SIRS_demographics_ode *Susceptible-Infected-Recovered-Susceptible Model with Simple Demographics*

Description

Susceptible-Infected-Recovered-Susceptible Model with Simple Demographics

Usage

```
SIRS_demographics_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
I_0 <- 1
R_0 <- 0
beta <- 3
chi <- 0.5
tau <- 2
mu <- 1/81
dt <- 1

parameters <- c(beta = beta, tau = tau, mu = mu)
inits <- c(S = S_0, I = I_0, R_0 = R_0)

SIRS_demographics_ode(1, inits, parameters)
```

SIRS_ode *Susceptible-Infected-Recovered-Susceptible Model*

Description

Susceptible-Infected-Recovered-Susceptible Model

Usage

```
SIRS_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
I_0 <- 1
R_0 <- 0
beta <- 3
chi <- 0.5
tau <- 2
dt <- 1
```

```
parameters <- c(beta = beta, tau = tau, chi = chi)
inits <- c(S = S_0, I = I_0, R_0 = R_0)

SIRS_ode(1, inits, parameters)
```

SIRS_vaccination_demographics_ode *Susceptible-Infected-Recovered-Susceptible Model with Simple Demographics and Vaccination*

Description

Susceptible-Infected-Recovered-Susceptible Model with Simple Demographics and Vaccination

Usage

```
SIRS_vaccination_demographics_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_u_0 <- 989
I_u_0 <- 1
R_u_0 <- 0
S_v_0 <- 0
I_v_0 <- 0
R_v_0 <- 0
beta <- 3
tau <- 2
chi <- 0.5
mu <- 1/81
alpha <- 0.8
lambda <- 0.7
dt <- 1

parameters <- c(beta = beta, tau = tau, mu = mu,
                alpha = 0.8, lambda = 0.7)
inits <- c(S_u = S_u_0, I_u = I_u_0, R_u_0 = R_u_0,
           S_v = S_v_0, I_v = I_v_0, R_v_0 = R_v_0)
```

```
SIRS_vaccination_demographics_ode(1, inits, parameters)
```

SIRS_vaccination_ode *Susceptible-Infected-Recovered-Susceptible Model with Vaccination*

Description

Susceptible-Infected-Recovered-Susceptible Model with Vaccination

Usage

```
SIRS_vaccination_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_u_0 <- 989
I_u_0 <- 1
R_u_0 <- 0
S_v_0 <- 0
I_v_0 <- 0
R_v_0 <- 0
beta <- 3
chi <- 0.5
tau <- 2
lambda <- 0.7
dt <- 1

parameters <- c(beta = beta, tau = tau,
                chi = chi, lambda = 0.7)
inits <- c(S_u = S_u_0, I_u = I_u_0, R_u_0 = R_u_0,
           S_v = S_v_0, I_v = I_v_0, R_v_0 = R_v_0)

SIRS_vaccination_ode(1, inits, parameters)
```

SIR_demographics_ode *Susceptible-Infected-Recovered Model with Simple Demographics*

Description

Susceptible-Infected-Recovered Model with Simple Demographics

Usage

```
SIR_demographics_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
I_0 <- 1
R_0 <- 0
beta <- 3
tau <- 2
mu <- 1/81


parameters <- c(beta = beta, tau = tau, mu = mu)
inits <- c(S = S_0, I = I_0, R_0 = R_0)

SIR_demographics_ode(1, inits, parameters)
```

SIR_ode *Susceptible-Infected-Recovered Model*

Description

Susceptible-Infected-Recovered Model

Usage

```
SIR_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 989
I_0 <- 1
R_0 <- 0
beta <- 3
tau <- 2

parameters <- c(beta = beta, tau = tau)
inits <- c(S = S_0, I = I_0, R_0 = R_0)

SIR_ode(1, inits, parameters)
```

SIR_vaccination_demographics_ode

Susceptible-Infected-Recovered Model with Simple Demographics and Vaccination

---

Description

Susceptible-Infected-Recovered Model with Simple Demographics and Vaccination

Usage

```
SIR_vaccination_demographics_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_u_0 <- 989
I_u_0 <- 1
R_u_0 <- 0
S_v_0 <- 0
I_v_0 <- 0
R_v_0 <- 0
beta <- 3
tau <- 2
mu <- 1/81
alpha <- 0.8
lambda <- 0.7

parameters <- c(beta = beta, tau = tau, mu = mu,
                alpha = 0.8, lambda = 0.7)
inits <- c(S_u = S_u_0, I_u = I_u_0, R_u_0 = R_u_0,
           S_v = S_v_0, I_v = I_v_0, R_v_0 = R_v_0)

SIR_vaccination_demographics_ode(1, inits, parameters)
```

SIR_vaccination_ode *Susceptible-Infected-Recovered Model with Vaccination*

Description

Susceptible-Infected-Recovered Model with Vaccination

Usage

```
SIR_vaccination_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_u_0 <- 989
I_u_0 <- 1
R_u_0 <- 0
```

```
S_v_0 <- 0
I_v_0 <- 0
R_v_0 <- 0
beta <- 3
tau <- 2
lambda <- 0.7

parameters <- c(beta = beta, tau = tau,
                lambda = 0.7)
inits <- c(S_u = S_u_0, I_u = I_u_0, R_u_0 = R_u_0,
           S_v = S_v_0, I_v = I_v_0, R_v_0 = R_v_0)

SIR_vaccination_ode(1, inits, parameters)
```

`SIS_demographics_ode` *Susceptible-Infected-Susceptible Model with Simple Demographics*

Description

Susceptible-Infected-Susceptible Model with Simple Demographics

Usage

```
SIS_demographics_ode(t, x, params)
```

Arguments

`t`	The timestep over which to calculate derivatives
`x`	A numeric vector of compartment populations.
`params`	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 999
I_0 <- 1
beta <- 3
chi <- 2
mu <- 1/81
dt <- 1
parameters <- c(beta = beta, mu = mu)
inits <- c(S = S_0, I = I_0)

SIS_demographics_ode(1, inits, parameters)
```

SIS_ode *Susceptible-Infected-Susceptible Model*

Description

Susceptible-Infected-Susceptible Model

Usage

```
SIS_ode(t, x, params)
```

Arguments

t	The timestep overwhich to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 999
I_0 <- 1
beta <- 3
chi <- 2
parameters <- c(beta = beta, chi = chi)
inits <- c(S = S_0, I = I_0)

SIS_ode(1, inits, parameters)
```

SI_demographics_ode *Susceptible-Infected Model with Simple Demographics*

Description

Susceptible-Infected Model with Simple Demographics

Usage

```
SI_demographics_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 999
I_0 <- 1
beta <- 3
mu <- 1/81

parameters <- c(beta = beta, mu = mu)
inits <- c(S = S_0, I = I_0)

SI_demographics_ode(1, inits, parameters)
```

SI_ode *Susceptible-Infected Model*

Description

Susceptible-Infected Model

Usage

```
SI_ode(t, x, params)
```

Arguments

t	The timestep over which to calculate derivatives
x	A numeric vector of compartment populations.
params	A named vector of parameter values.

Value

A vector of derivatives

Examples

```
##Model Input
S_0 <- 999
I_0 <- 1
beta <- 3
parameters <- c(beta = beta)
inits <- c(S = S_0, I = I_0)

SI_ode(1, inits, parameters)
```

`solve_ode` *A Simple Wrapper for lsoda*

Description

This function acts as a simple wrapper for lsoda, allowing for multiple parameter sets and initial conditions. It also allows `lsoda` to be used within the idmodelr framework.

Usage

```
solve_ode(
  model = NULL,
  inits = NULL,
  params = NULL,
  times = NULL,
  as.data.frame = TRUE,
  ...
)
```

Arguments

`model`	A model formatted as required by `lsoda`, see `SI_ode` for an example.
`inits`	The initial state (states) of the model. Can either be supplied as a named vector or as a matrix with each row representing a parameter.
`params`	A named vector or matrix of parameters. The matrix must have a row for each parameter and if `inits` is specified as a matrix then `params` must have the same number of columns
`times`	A numeric vector of the times for which explicit model estimates are required, this does not effect the timestep used by the solver
`as.data.frame`	A logical (defaults to `TRUE`) indicating if the results should be returned as a data frame.
`...`	Additional arguments to pass to `lsoda`.

Value

A dataframe or lsoda object containing a single or multiple model trajectories

See Also

lsoda SI_ode

Examples

```
## Intialise
N = 100000
I_0 = 1
S_0 = N - I_0
R_0 = 1.1
beta = R_0

##Time for model to run over
tbegin = 0
tend = 50
times <- seq(tbegin, tend, 1)

##Vectorise input
parameters <- as.matrix(c(beta = beta))
inits <- as.matrix(c(S = S_0, I = I_0))

solve_ode(model = SI_ode, inits, parameters, times, as.data.frame = TRUE)
```

---

summarise_model *Summarise a Model Simulation*

Description

Provides simple summary statistics for a model produced using solve_ode. These include the final population sizes, the time and size of the maximum epidemic peak, and the duration of the epidemic.

Usage

```
summarise_model(sim)
```

Arguments

sim	A tibble of model output as produced by solve_ode.

Value

A tibble of summary information for a model simulation.

Examples

```
## Intialise
N = 100000
I_0 = 1
S_0 = N - I_0
R_0 = 1.1
beta = R_0

##Time for model to run over
tbegin = 0
tend = 50
times <- seq(tbegin, tend, 1)

##Vectorise input
parameters <- as.matrix(c(beta = beta))
inits <- as.matrix(c(S = S_0, I = I_0))

sim <- solve_ode(model = SI_ode, inits, parameters, times, as.data.frame = TRUE)

summarise_model(sim)
```

`summarise_strat_var` *Sum a Stratified Variable*

Description

Sum a Stratified Variable

Usage

```
summarise_strat_var(df, vars, strat = NULL, new_var = "sum")
```

Arguments

`df`	A dataframe of model output.
`vars`	A character vector containing the unstratified variables to summarise
`strat`	The number of stratifications present in the data set
`new_var`	The name of the summarised variable

Value

Returns the original dataframe with an additional summarised variable

See Also

summarise_var_by_strat

Examples

```
df <- dplyr::mutate(iris, Petal.Length1 = Petal.Length, Petal.Length2 = Petal.Length)
df <- tibble::as_tibble(df)

summarise_strat_var(df, vars = c("Petal.Length"), strat = 2, new_var = "sum")
```

`summarise_var_by_strat` *Sum a Stratified Variable by Stratification Level*

Description

Sum a Stratified Variable by Stratification Level

Usage

```
summarise_var_by_strat(df, vars, strat = NULL, new_var, summary_var = FALSE)
```

Arguments

`df`	A dataframe of model output.
`vars`	A character vector containing the unstratified variables to summarise
`strat`	The number of stratifications present in the data set
`new_var`	The name of the summarised variable
`summary_var`	A logical (defaults to `FALSE`), specifying whether to add an additional summary variable across all stratified levels when aggregating incidence.

Details

Takes compartmental infectious disease output and adds summary statistics for each stratified population, optionally adding a final summary statistic for the whole population.

Value

An updated data frame containing the summarised variable for each stratified level and for the whole population.

See Also

summarise_var_by_strat

Examples

```
df <- data.frame(A = 1, B = 2)
summarise_var_by_strat(df, vars = c("A", "B"), new_var = "C")

df <- data.frame(A1 = 1, B1 = 1, A2 = 1, B2 = 1, A3 = 1, B3 = 1)
summarise_var_by_strat(df, vars = c("A", "B"), strat = 3, new_var = "C")
summarise_var_by_strat(df, vars = c("A", "B"), strat = 3, new_var = "C", summary_var = TRUE)
```

Index