epiworld

0.0-1

Generated by Doxygen 1.9.1

Chapter 1

Example: 00-hello-world

Output from the program:

```
Running the model...
_____________________________________________________________________________
|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||| done.
_____________________________________________________________________________
SIMULATION STUDY
Population size    : 100000
Days (duration)    : 100 (of 100)
Number of variants : 1
Last run elapsed t : 256.00ms
Rewiring           : off
Virus(es):
 - covid 19 (baseline prevalence: 5 seeds)
Tool(s):
 - vaccine (baseline prevalence: 50.00%)
Model parameters:
Distribution of the population at time 100:
 - Total healthy (S)   :   99995 -> 97390
 - Total recovered (S) :       0 -> 2554
 - Total infected (I)  :       5 -> 56
 - Total removed (R)   :       0 -> 0
(S): Susceptible, (I): Infected, (R): Recovered
_____________________________________________________________________________
```

Chapter 2

Contributor Code of Conduct

As contributors and maintainers of this project, we pledge to respect all people who contribute through reporting issues, posting feature requests, updating documentation, submitting pull requests or patches, and other activities.

We are committed to making participation in this project a harassment-free experience for everyone, regardless of level of experience, gender, gender identity and expression, sexual orientation, disability, personal appearance, body size, race, ethnicity, age, or religion.

Examples of unacceptable behavior by participants include the use of sexual language or imagery, derogatory comments or personal attacks, trolling, public or private harassment, insults, or other unprofessional conduct.

Project maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct. Project maintainers who do not follow the Code of Conduct may be removed from the project team.

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by opening an issue or contacting one or more of the project maintainers.

This Code of Conduct is adapted from the Contributor Covenant (http://contributor-covenant.org), version 1.0.0, available at http://contributor-covenant.org/version/1/0/0/

Chapter 3

SIR Multiple Runs (b)

```
./02b-sir_multiple_runs.o
```

```
##
## ____________________________________________________________________________
## SIMULATION STUDY
##
## Population size    : 10000
## Days (duration)    : 0 (of 60)
## Number of variants : 1
## Last run elapsed t : -
## Rewiring           : off
##
## Virus(es):
##  - a virus (baseline prevalence: 10.00%)
## Tool(s):
##  - Immune system (baseline prevalence: 100.00%)
##
## Model parameters:
##  - Immune contagion_reduction : 0.5000
##  - Immune recovery            : 0.5000
##  - Infectiousness             : 0.8000
##  - Post immunity              : 0.8000
##
## Distribution of the population at time 0:
##  - Total healthy (S)   : 9000
##  - Total recovered (S) : 0
##  - Total infected (I)  : 1000
##  - Total removed (R)   : 0
##
## (S): Susceptible, (I): Infected, (R): Recovered
## ____________________________________________________________________________
##
## Starting multiple runs (100)
## ____________________________________________________________________________
## ____________________________________________________________________________
## |||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||| done.
##  done.
## last run elapsed time : 0.00s
## total elapsed time    : 3.00s
## total runs            : 100
## mean run elapsed time : 0.03s
```

```
dat <- read.csv("02b-sir-multiple-runs.csv")
dat <- with(
  dat,
  rbind(
    data.frame(run_id = run_id, status = "healthy", n = healthy),
    data.frame(run_id = run_id, status = "infected", n = infected),
    data.frame(run_id = run_id, status = "recovered", n = recovered),
    data.frame(run_id = run_id, status = "removed", n = removed)
  ))
library(ggplot2)
ggplot(dat, aes(y = n, colour = status)) +
  geom_boxplot()
```

Chapter 4

epiworld c++ template library

4.1 Main features

This C++ template-header-only library provides a general framework for epidemiologic simulation. The main features of the library are:

1. Four key classes: `Model`, `Person`, `Tool`, and `Virus`.
2. The model features a social networks of `Persons`.
3. `Persons` can have multiple `Tools` as a defense system.
4. `Tools` can reduce contagion rate, transmissibility, death rates, and improve recovery rates.
5. `Viruses` can mutate (generating new variants).
6. `Models` can feature multiple states, e.g., `HEALTHY`, `SUSCEPTIBLE`, etc.
7. `Models` can have an arbitrary number of parameters.
8. **REALLY FAST** About 6.5 Million person/day simulations per second.

4.2 Algorithm

Setup

- Create viruses.
- Create tools (arbitrary).
- Set model parameters (arbitrary).
- Create global events (e.g., surveillance).
- Set up the population: small world network (default).
- Set up rewiring (optional).
- Set statuses (arbitrary number of them).

Run

1. Distribute the tool(s) and virus(es)
2. For each t in 1 -> Duration:
    - Update status for susceptible/infected/removed(?)
    - Mutate virus(es) (each individual)
    - Run global actions (e.g., surveillance)
    - Run rewiring algorithm

Along update:

- Contagion events are applied recorded.
- New variants are recorded.
- Optional user data is recorded.

4.3 Hello world (C++)

```
#include "include/epiworld/epiworld.hpp"
int main()
{
  // Creating a virus
  epiworld::Virus<> covid19("covid 19");
  covid19.set_infectiousness(.8);

  // Creating a tool
  epiworld::Tool<> vax("vaccine");
  vax.set_contagion_reduction(.95);
  // Creating a model
  epiworld::Model<> model;
  // Adding the tool and virus
  model.add_virus(covid19, .01);
  model.add_tool(vax, .5);
  // Generating a random pop
  model.population_from_adjlist(
    epiworld::rgraph_smallworld(1000, 5, .2)
  );
  // Initializing setting days and seed
  model.init(60, 123123);
  // Running the model
  model.run();
  model.print();
  return;
}
```

4.4 Surveillance simulation

- Incubation time of the disease $\sim$ `Gamma(3, 1)`
- Duration of the disease $\sim$`Gamma(12, 1)`
- Probability of becoming symptomatic: 0.9
- Prob. of transmission: 1.0.
- Vaccinated population: 25%
- Vaccine efficacy: .9.
- Vaccine reduction on transmission: 0.5.
- Surveillance program of x% of the population at random.
- Individuals who test positive become isolated.

4.4.1 Preliminary results

```
# With low surveillance
pop_size <- 20e3
pop_seed <- pop_size * .01
s_levels <- c(0.0001, 0.002)
system(sprintf("./07-surveillance.o %i %i 100 %.04f 2>&1", pop_seed, pop_size, s_levels[1]), intern = TRUE)
       |>
  cat(sep = "\n")
```

```
## Running the model...
## ________________________________________________________________________________
## ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||| done.
##
## ________________________________________________________________________________________________
## SIMULATION STUDY
##
## Population size    : 20000
## Days (duration)    : 200 (of 200)
## Number of variants : 1
## Last run elapsed t : 505.00ms
## Rewiring           : off
##
## Virus(es):
##  - Covid19 (baseline prevalence: 100 seeds)
## Tool(s):
##  - Vaccine (baseline prevalence: 25.00%)
##
## Model parameters:
##  - Infect period         : 12.0000
##  - Latent period         : 3.0000
##  - Prob of symptoms      : 0.7000
##  - Prob of transmission  : 1.0000
##  - Prob. death           : 0.0010
##  - Prob. reinfect        : 0.1000
##  - Surveilance prob.     : 1.0e-04
##  - Vax efficacy          : 0.9000
##  - Vax redux transmision : 0.5000
##
## Distribution of the population at time 200:
##  - Total susceptible (S)           :   19900 -> 2106
##  - Total recovered (S)             :       0 -> 17369
##  - Total latent (I)                :     100 -> 109
##  - Total symptomatic (I)           :       0 -> 155
##  - Total symptomatic isolated (I)  :       0 -> 2
##  - Total asymptomatic (I)          :       0 -> 72
##  - Total asymptomatic isolated (I) :       0 -> 0
##  - Total removed (R)               :       0 -> 187
##
## (S): Susceptible, (I): Infected, (R): Recovered
## ________________________________________________________________________________________________
```

```
hist1 <- read.csv("07-surveillance_hist.txt", sep = " ")
surv1 <- read.csv("07-surveillance_user_data.txt", sep = " ")
# With high surveillance
system(sprintf("./07-surveillance.o %i %i 100 %.04f 2>&1", pop_seed, pop_size, s_levels[2]), intern = TRUE)
       |>
  cat(sep = "\n")
```

```
## Running the model...
## ________________________________________________________________________________
## ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||| done.
##
## ________________________________________________________________________________________________
## SIMULATION STUDY
##
## Population size    : 20000
## Days (duration)    : 200 (of 200)
## Number of variants : 1
## Last run elapsed t : 530.00ms
## Rewiring           : off
##
## Virus(es):
```

```
##  - Covid19 (baseline prevalence: 100 seeds)
## Tool(s):
##  - Vaccine (baseline prevalence: 25.00%)
##
## Model parameters:
##  - Infect period         : 12.0000
##  - Latent period         : 3.0000
##  - Prob of symptoms      : 0.7000
##  - Prob of transmission  : 1.0000
##  - Prob. death           : 0.0010
##  - Prob. reinfect        : 0.1000
##  - Surveilance prob.     : 0.0020
##  - Vax efficacy          : 0.9000
##  - Vax redux transmision : 0.5000
##
## Distribution of the population at time 200:
##  - Total susceptible (S)           :   19900 -> 2125
##  - Total recovered (S)             :       0 -> 17325
##  - Total latent (I)                :     100 -> 109
##  - Total symptomatic (I)           :       0 -> 155
##  - Total symptomatic isolated (I)  :       0 -> 8
##  - Total asymptomatic (I)          :       0 -> 76
##  - Total asymptomatic isolated (I) :       0 -> 1
##  - Total removed (R)               :       0 -> 201
##
## (S): Susceptible, (I): Infected, (R): Recovered
## ________________________________________________________________________________
```

```
hist2 <- read.csv("07-surveillance_hist.txt", sep = " ")
surv2 <- read.csv("07-surveillance_user_data.txt", sep = " ")
hist_comb <- rbind(
  cbind(sim = as.character(s_levels[1]), hist1),
  cbind(sim = as.character(s_levels[2]), hist2)
  )
ggplot(hist_comb, aes(x = date, y = counts + 1, colour = status, linetype=sim)) +
  geom_line() +
  # scale_y_log10() +
  labs(y = "Counts (log)")
```

4.4.2 Cases detected

```
survdat <- rbind(
  with(surv1, rbind(
    data.frame(Id = as.character(s_levels[1]), Date = date, Type = "N Sampled",  n = nsampled),
    data.frame(Id = as.character(s_levels[1]), Date = date, Type = "N detected",  n = ndetected),
    data.frame(Id = as.character(s_levels[1]), Date = date, Type = "N detected Asymp",  n =
       ndetected_asympt),
    data.frame(Id = as.character(s_levels[1]), Date = date, Type = "N Asymp",  n = nasymptomatic)
  )),
  with(surv2, rbind(
    data.frame(Id = as.character(s_levels[2]), Date = date, Type = "N Sampled",  n = nsampled),
    data.frame(Id = as.character(s_levels[2]), Date = date, Type = "N detected",  n = ndetected),
    data.frame(Id = as.character(s_levels[2]), Date = date, Type = "N detected Asymp",  n =
       ndetected_asympt),
    data.frame(Id = as.character(s_levels[2]), Date = date, Type = "N Asymp",  n = nasymptomatic)
  ))
)
ggplot(survdat, aes(x = Date, y = n + 1, colour = Type)) +
  geom_line() +
  facet_wrap(~Id) +
  scale_y_log10() +
  labs(y = "Counts (log)")
```

Chapter 5

MIT License

Chapter 6

model1

The dynamics of the simulation process are:

1. Discrete Markov process.
2. The simulation has the following parameters:

   a. New variant emergence at rate `X.` b. For each variant `k`:

   - Unvaccinated individuals become sick rate `C(k),`
   - Mortality rate `D(k),`
   - Recovery rate `H(k),`
   - Vaccines have an efficaccy rate `E(v,k)` and pseudo vaccines (recovered) have efficacy rate `E(r,k) < E(v,k).` In general, the probability of `i` acquiring the disease `k` from `j` will be equal to

   ``` P(i gets the disease from j | their states) = C(k) * (1 - E(i,k)) * (1 - E(j, k)) ```

   where `(i,j) in (u,v,r).` Efficacy rate for unvaccinated is zero.

   - Vaccinated individuals have a reduced mortality rate `D(k,v) > D(k),` and recovered individuals `D(k,r) in (D(k,v), D(k)]`
   - Vaccinated individuals have an increased recovery rate `H(k,v) > H(k)`, whereas recovered's rate `H(k,r) in [H(k), H(k,v)).`

   The sum of mortality and recovery rates is less than one since the difference represents no change.

   c. Each country vaccinates citizens at rate `V` function of `A` (availability) and `B` (citizens' acceptance rate.) d. In each country `i,` the entire population `N(i)` distributes between the following states:

   - Healthy unvaccinated (`N(i,t,u)`),
   - Healthy vaccinated (`N(i,t,v)`),
   - Deceased (`N(i,t,d)`),
   - Recovered (`N(i,t,r)`),
   - Unvaccinated and sick with variant (`N(i,t,s,k|u)`) `k.`, and
   - Vaccinated and sick with variant (`N(i,t,s,k|v)`) `k.`

   Total sick are `N(i,t,k,s) = sum(g in {u,v}) N(i,t,k,s|g)`

   Globally, we keep track of the prevalence of new variants. Variants can disappear if no more individuals port the variant, i.e., the prevalence rate `P(k,t)=sum(i) N(i,s,k)` equals zero.

   d. Vaccines are manufactured at each country at rates `M(i)` and uniformly shared with other countries at rate `S(i).` c. Population flows between each country pair `(i,j)` at a rate `F(i,j).` Flows between countries do not change Population and are symmetric.
   ```

3. The simulation process is as follows:

   (a) Countries are initialized with a total population `N(i)`.

   (b) Variant zero initializes at a random location `i`, with an initial prevalence `P(k,t) = N(i,t,k)`.

   (c) For time `t` in `(0,T)` do:

   a. Unvaccinated individuals can become sick of variant `k` with probability:

   ``` Pr(h->s|i,t,k,u) ∼ sum(g in {u,v}) (N(i,t - 1,s,k|g) + sum(j != i) F(i,j) ∗ N(j,t-1,s,k|g)) ∗ C(k) / (N(i) + sum(j != i) N(j)) ```

   b. Vaccinated individuals can become sick of variant `k` with probability: `Pr(v->s|i,t,k,v) ∼ Pr(h->s|i,t,k) * (1 - E(v,k))`.

   b. Recovered individuals can become sick of variant `k` with probability: `Pr(v->s|i,t,k,r) ∼ Pr(h->s|i,t,k) * (1 - E(r,k))`.

   c. Sick individuals with variant `k` die with probability `D(k)` or recover with probability `H(k)`, otherwise they stay infected; with the rates depending on their vaccination status `v` or `n`.

   d. Unvaccinated individuals vaccinate in country `i` with probability `P(u->v) ∼ V(A(i,t), B(i))`.

   e. The country vaccine supply changes.
   ```

Chapter 7

EPI Simulator

7.1 Disease dynamics

Diseases continuously evolve in time. Changes in their genetic sequence make them more or less resistant to the particular version of the vaccine. Mutations also affect the transmissibility level and mortality rate of the disease. Using this approach allows making vaccination efficacy a function of compatibility between the variant and the vaccine.

When an individual becomes infected, the disease accumulates mutations in the new host. Ultimately, there is no single version of the disease present in the model, but rather an infinite number of them, each slightly different from the other.

7.2 Network dynamics

We can assume that the Population is organized in fully connected blocks for the first version of the model. Block sizes and the number of connections between blocks are Poisson random variables. Individuals interact with all the members of their blocks, and bridging individuals allow the disease to move across blocks.

7.3 Contagion dynamics

The transmission of the disease will be governed by the number of vaccinated, infected, and recovered within each block. Transmission between blocks will be treated in the same way, although individuals bridging the block will only interact with others within the block and their direct connections across the blocks.

7.4 Time dynamics

Time dynamics has two components, how biology evolves and how agents react.

The model develops as a continuous-time Markov process. Each block of individuals takes action at rates `L(i|N(i))` function of the local number of infections. This way, if

7.5 Updating agent's status

Like most other components, updating agents' states can be personalized. A naive approach allows agents to get infected with a single virus or stay as-is. The probability of this event is conditional on acquiring at most one virus. Since these are independent events, the conditional probability is computed as follows:

```
P(Variant k|at most 1) = P(at most 1| Variant k) * P(Variant k)/P(at most 1)
                       = P(only variant k)/P(variant k) * P(Variant k)/P(at most 1)
                       = P(only variant k)/P(at most 1)
```

Where

```
P(only variant k) = P(k) * Prod(m!=v) (1 - P(m))
P(at most 1) = P(None) + Sum(v in variants) P(v) * Prod(m != v) (1 - P(m))
P(None) = Prod(v in variants) (1 - P(v))
```

Furthermore, the (Variant, Person) pairs are treated independently.

7.5.1 Other parameters

- Who did you get the infection from.
- Omicron is 1.5 more infectious than delta.
- Surveillance:
  - Pull people to be tested at random.
  - Or at symptoms.
  - A mix of the two.
- Define a class for passing extra functions and datasets, for example, testing surveillance.
- Exposed people become infectious after k days.
- Network changesthe can be a function of an ERGM. Apply K steps throughout time.
- Add progress bar.

Chapter 8

Contributor Covenant Code of Conduct

8.1 Our Pledge

We as members, contributors, and leaders pledge to make participation in our community a harassment-free experience for everyone, regardless of age, body size, visible or invisible disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, religion, or sexual identity and orientation.

We pledge to act and interact in ways that contribute to an open, welcoming, diverse, inclusive, and healthy community.

8.2 Our Standards

Examples of behavior that contributes to a positive environment for our community include:

- Demonstrating empathy and kindness toward other people
- Being respectful of differing opinions, viewpoints, and experiences
- Giving and gracefully accepting constructive feedback
- Accepting responsibility and apologizing to those affected by our mistakes, and learning from the experience
- Focusing on what is best not just for us as individuals, but for the overall community

Examples of unacceptable behavior include:

- The use of sexualized language or imagery, and sexual attention or advances of any kind
- Trolling, insulting or derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or email address, without their explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

8.3 Enforcement Responsibilities

Community leaders are responsible for clarifying and enforcing our standards of acceptable behavior and will take appropriate and fair corrective action in response to any behavior that they deem inappropriate, threatening, offensive, or harmful.

Community leaders have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, and will communicate reasons for moderation decisions when appropriate.

8.4 Scope

This Code of Conduct applies within all community spaces, and also applies when an individual is officially representing the community in public spaces. Examples of representing our community include using an official e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event.

8.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported to the community leaders responsible for enforcement at g.vegayon@gmail.com. All complaints will be reviewed and investigated promptly and fairly.

All community leaders are obligated to respect the privacy and security of the reporter of any incident.

8.6 Enforcement Guidelines

Community leaders will follow these Community Impact Guidelines in determining the consequences for any action they deem in violation of this Code of Conduct:

8.6.1 1. Correction

Community Impact: Use of inappropriate language or other behavior deemed unprofessional or unwelcome in the community.

Consequence: A private, written warning from community leaders, providing clarity around the nature of the violation and an explanation of why the behavior was inappropriate. A public apology may be requested.

8.6.2 2. Warning

Community Impact: A violation through a single incident or series of actions.

Consequence: A warning with consequences for continued behavior. No interaction with the people involved, including unsolicited interaction with those enforcing the Code of Conduct, for a specified period of time. This includes avoiding interactions in community spaces as well as external channels like social media. Violating these terms may lead to a temporary or permanent ban.

8.6.3 3. Temporary Ban

Community Impact: A serious violation of community standards, including sustained inappropriate behavior.

Consequence: A temporary ban from any sort of interaction or public communication with the community for a specified period of time. No public or private interaction with the people involved, including unsolicited interaction with those enforcing the Code of Conduct, is allowed during this period. Violating these terms may lead to a permanent ban.

8.6.4 4. Permanent Ban

Community Impact: Demonstrating a pattern of violation of community standards, including sustained inappropriate behavior, harassment of an individual, or aggression toward or disparagement of classes of individuals.

Consequence: A permanent ban from any sort of public interaction within the community.

8.7 Attribution

This Code of Conduct is adapted from the Contributor Covenant, version 2.0, available at https://www.contributor-covenant.org/version/2/0/code_of_conduct.html.

Community Impact Guidelines were inspired by Mozilla's code of conduct enforcement ladder.

For answers to common questions about this code of conduct, see the FAQ at https://www.contributor-covenant.org/faq. Translations are available at https://www.contributor-covenant.org/translations.

Chapter 9

MIT License

Chapter 10

epiworld 1.0

- Added a `NEWS.md` file to track changes to the package.

Chapter 11

Class Index

11.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 12

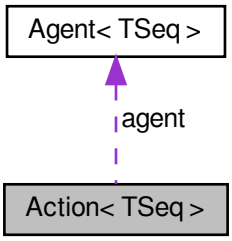
Class Documentation

12.1 Action< TSeq > Struct Template Reference

Action data for update an agent.

`#include <config.hpp>`

Collaboration diagram for Action< TSeq >:



Public Member Functions

- Action (Agent< TSeq > *agent_, VirusPtr< TSeq > virus_, ToolPtr< TSeq > tool_, Entity< TSeq > *entity_, epiworld_fast_int new_status_, epiworld_fast_int queue_, ActionFun< TSeq > call_, int idx_agent_, int idx_object_)

  Construct a new Action object.

Public Attributes

- Agent< TSeq > * **agent**
- VirusPtr< TSeq > **virus**
- ToolPtr< TSeq > **tool**
- Entity< TSeq > * **entity**
- epiworld_fast_int **new_status**
- epiworld_fast_int **queue**
- ActionFun< TSeq > **call**
- int **idx_agent**
- int **idx_object**

12.1.1 Detailed Description

template<typename TSeq>
struct Action< TSeq >

Action data for update an agent.

Template Parameters

TSeq	

12.1.2 Constructor & Destructor Documentation

12.1.2.1 Action()

```
template<typename TSeq >
Action< TSeq >::Action (
            Agent< TSeq > * agent_,
            VirusPtr< TSeq > virus_,
            ToolPtr< TSeq > tool_,
            Entity< TSeq > * entity_,
            epiworld_fast_int new_status_,
            epiworld_fast_int queue_,
            ActionFun< TSeq > call_,
            int idx_agent_,
            int idx_object_ )  [inline]
```

Construct a new Action object.

All the parameters are rather optional.

Parameters

*agent_*	Agent over who the action will happen
*virus_*	Virus to add
*tool_*	Tool to add
virus_idx	Index of virus to be removed (if needed)
tool_idx	Index of tool to be removed (if needed)
*new_status_*	Next status
*queue_*	Efect on the queue
*call_*	The action call (if needed)
*idx_agent_*	Location of agent in object.
*idx_object_*	Location of object in agent.

The documentation for this struct was generated from the following files:

- include/epiworld/agent-bones.hpp
- include/epiworld/config.hpp

12.2 epiworld::Action< TSeq > Struct Template Reference

Action data for update an agent.

`#include <epiworld.hpp>`

Public Member Functions

- Action (Agent< TSeq > *agent_, VirusPtr< TSeq > virus_, ToolPtr< TSeq > tool_, Entity< TSeq > *entity_, epiworld_fast_int new_status_, epiworld_fast_int queue_, ActionFun< TSeq > call_, int idx_agent_, int idx_object_)

  Construct a new Action object.

Public Attributes

- Agent< TSeq > * **agent**
- VirusPtr< TSeq > **virus**
- ToolPtr< TSeq > **tool**
- Entity< TSeq > * **entity**
- epiworld_fast_int **new_status**
- epiworld_fast_int **queue**
- ActionFun< TSeq > **call**
- int **idx_agent**
- int **idx_object**

12.2.1 Detailed Description

template<typename TSeq>
struct epiworld::Action< TSeq >

Action data for update an agent.

Template Parameters

TSeq	

12.2.2 Constructor & Destructor Documentation

12.2.2.1 Action()

```
template<typename TSeq >
epiworld::Action< TSeq >::Action (
            Agent< TSeq > * agent_,
            VirusPtr< TSeq > virus_,
            ToolPtr< TSeq > tool_,
            Entity< TSeq > * entity_,
            epiworld_fast_int new_status_,
            epiworld_fast_int queue_,
            ActionFun< TSeq > call_,
            int idx_agent_,
            int idx_object_ )  [inline]
```

Construct a new Action object.

All the parameters are rather optional.

Parameters

*agent_*	Agent over who the action will happen
*virus_*	Virus to add
*tool_*	Tool to add
virus_idx	Index of virus to be removed (if needed)
tool_idx	Index of tool to be removed (if needed)
*new_status_*	Next status
*queue_*	Efect on the queue
*call_*	The action call (if needed)
*idx_agent_*	Location of agent in object.
*idx_object_*	Location of object in agent.

The documentation for this struct was generated from the following file:

- epiworld.hpp

12.3 AdjList Class Reference

Public Member Functions

- AdjList (const std::vector< unsigned int > &source, const std::vector< unsigned int > &target, int size, bool directed)

  Construct a new Adj List object.
- void read_edgelist (std::string fn, int size, int skip=0, bool directed=true)

  Read an edgelist.
- std::map< unsigned int, unsigned int > **operator()** (unsigned int i) const
- void **print** (unsigned int limit=20u) const
- size_t vcount () const

Number of vertices/nodes in the network.

- size_t ecount () const

  Number of edges/arcs/ties in the network.

- std::vector< std::map< unsigned int, unsigned int > > & **get_dat** ()
- bool is_directed () const

  `true` if the network is directed.

12.3.1 Constructor & Destructor Documentation

12.3.1.1 AdjList()

```
AdjList::AdjList (
            const std::vector< unsigned int > & source,
            const std::vector< unsigned int > & target,
            int size,
            bool directed )  [inline]
```

Construct a new Adj List object.

Ids in the network are assume to range from `0` to `size - 1`.

Parameters

source	Unsigned int vector with the source
target	Unsigned int vector with the target
size	Number of vertices in the network.
directed	Bool true if the network is directed

12.3.2 Member Function Documentation

12.3.2.1 read_edgelist()

```
void AdjList::read_edgelist (
            std::string fn,
            int size,
            int skip = 0,
            bool directed = true )  [inline]
```

Read an edgelist.

Ids in the network are assume to range from `0` to `size - 1`.

Parameters

fn	Path to the file
skip	Number of lines to skip (e.g., 1 if there's a header)
directed	`true` if the network is directed
size	Number of vertices in the network.

The documentation for this class was generated from the following files:

- include/epiworld/adjlist-bones.hpp
- include/epiworld/adjlist-meat.hpp

12.4 epiworld::AdjList Class Reference

Public Member Functions

- AdjList (const std::vector< unsigned int > &source, const std::vector< unsigned int > &target, int size, bool directed)

  Construct a new Adj List object.
- void read_edgelist (std::string fn, int size, int skip=0, bool directed=true)

  Read an edgelist.
- std::map< unsigned int, unsigned int > **operator()** (unsigned int i) const
- void **print** (unsigned int limit=20u) const
- size_t vcount () const

  Number of vertices/nodes in the network.
- size_t ecount () const

  Number of edges/arcs/ties in the network.
- std::vector< std::map< unsigned int, unsigned int > > & **get_dat** ()
- bool is_directed () const

  `true` *if the network is directed.*
- AdjList (const std::vector< unsigned int > &source, const std::vector< unsigned int > &target, bool directed, int min_id=-1, int max_id=-1)

  Construct a new Adj List object.
- void **read_edgelist** (std::string fn, int skip=0, bool directed=true, int min_id=-1, int max_id=-1)
- std::map< unsigned int, unsigned int > **operator()** (unsigned int i) const
- void **print** (unsigned int limit=20u) const
- unsigned int **get_id_max** () const
- unsigned int **get_id_min** () const
- size_t **vcount** () const
- size_t **ecount** () const
- std::map< unsigned int, std::map< unsigned int, unsigned int > > & **get_dat** ()
- bool **is_directed** () const

12.4.1 Constructor & Destructor Documentation

12.4.1.1 AdjList() [1/2]

```
AdjList::AdjList (
            const std::vector< unsigned int > & source,
            const std::vector< unsigned int > & target,
            int size,
            bool directed )  [inline]
```

Construct a new Adj List object.

Ids in the network are assume to range from `0` to `size - 1`.

Parameters

source	Unsigned int vector with the source
target	Unsigned int vector with the target
size	Number of vertices in the network.
directed	Bool true if the network is directed

12.4.1.2 AdjList() [2/2]

```
AdjList::AdjList (
            const std::vector< unsigned int > & source,
            const std::vector< unsigned int > & target,
            bool directed,
            int min_id = -1,
            int max_id = -1 )  [inline]
```

Construct a new Adj List object.

It will create an adjacency list object with `maxid - minid + 1` nodes. If min_id and max_id are not specified (both < 0), then the program will try to figure them out automatically by looking at the range of the observed ids.

Parameters

source	Unsigned int vector with the source
target	Unsigned int vector with the target
directed	Bool true if the network is directed
min_id	int min id.
max_id	int max id.

12.4.2 Member Function Documentation

12.4.2.1 read_edgelist()

```
void AdjList::read_edgelist (
            std::string fn,
```

```
int size,
int skip = 0,
bool directed = true )  [inline]
```

Read an edgelist.

Ids in the network are assume to range from `0` to `size - 1`.

Parameters

fn	Path to the file
skip	Number of lines to skip (e.g., 1 if there's a header)
directed	`true` if the network is directed
size	Number of vertices in the network.

The documentation for this class was generated from the following file:

- epiworld.hpp

12.5 Agent< TSeq > Class Template Reference

Agent (agents)

`#include <agent-bones.hpp>`

Public Member Functions

- **Agent** (const Agent< TSeq > &p)
- int get_id () const

  Id of the individual.
- std::mt19937 * **get_rand_endgine** ()
- Model< TSeq > * **get_model** ()
- VirusPtr< TSeq > & **get_virus** (int i)
- Viruses< TSeq > **get_viruses** ()
- const Viruses_const< TSeq > **get_viruses** () const
- size_t **get_n_viruses** () const noexcept
- ToolPtr< TSeq > & **get_tool** (int i)
- Tools< TSeq > **get_tools** ()
- const Tools_const< TSeq > **get_tools** () const
- size_t **get_n_tools** () const noexcept
- void **mutate_variant** ()
- void **add_neighbor** (Agent< TSeq > *p, bool check_source=true, bool check_target=true)
- std::vector< Agent< TSeq > * > & **get_neighbors** ()
- void **change_status** (epiworld_fast_uint new_status, epiworld_fast_int queue=0)
- const epiworld_fast_uint & **get_status** () const
- void **reset** ()
- bool **has_tool** (unsigned int t) const
- bool **has_tool** (std::string name) const
- bool **has_virus** (unsigned int t) const
- bool **has_virus** (std::string name) const
- void **print** (bool compressed=false) const
- const std::vector< Entity< TSeq > * > **get_entities** ()

Add/Remove Virus/Tool

Any of these is ultimately reflected at the end of the iteration.

Parameters

tool	*Tool to add*
virus	*Virus to add*
status_new	*Status after the change*
queue	

- void **add_tool** (ToolPtr< TSeq > tool, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **add_tool** (Tool< TSeq > tool, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **add_virus** (VirusPtr< TSeq > virus, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **add_virus** (Virus< TSeq > virus, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **add_entity** (Entity< TSeq > &entity, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_tool** (epiworld_fast_uint tool_idx, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_tool** (ToolPtr< TSeq > &tool, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_virus** (epiworld_fast_uint virus_idx, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_virus** (VirusPtr< TSeq > &virus, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_entity** (epiworld_fast_uint entity_idx, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_entity** (Entity< TSeq > &entity, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void rm_agent_by_virus (epiworld_fast_uint virus_idx, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)

  Agent removed by virus.
- void rm_agent_by_virus (VirusPtr< TSeq > &virus, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)

  Agent removed by virus.

Get the rates (multipliers) for the agent

Parameters

v	*A pointer to a virus.*

Returns

epiworld_double

- epiworld_double **get_susceptibility_reduction** (VirusPtr< TSeq > v)
- epiworld_double **get_transmission_reduction** (VirusPtr< TSeq > v)
- epiworld_double **get_recovery_enhancer** (VirusPtr< TSeq > v)
- epiworld_double **get_death_reduction** (VirusPtr< TSeq > v)

- double & operator() (size_t j)

  Access the j-th column of the agent.
- double & **operator[ ]** (size_t j)

Friends

- class **Model**< **TSeq** >
- class **Virus**< **TSeq** >
- class **Viruses**< **TSeq** >
- class **Viruses_const**< **TSeq** >
- class **Tool**< **TSeq** >
- class **Tools**< **TSeq** >
- class **Queue**< **TSeq** >
- void **default_add_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_add_tool** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_add_entity** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_tool** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_entity** (Action< TSeq > &a, Model< TSeq > *m)

12.5.1 Detailed Description

template<typename TSeq = int>
class Agent< TSeq >

Agent (agents)

Template Parameters

TSeq	Sequence type (should match `TSeq` across the model)

12.5.2 Member Function Documentation

12.5.2.1 operator()()

```
template<typename TSeq = int>
double& Agent< TSeq >::operator() (
            size_t j )
```

Access the j-th column of the agent.

If an external array has been specified, then these two functions can be used to access additional agent's features not included in the model.

The `operator[]` method is with no boundary check, whereas the `operator()` method checks boundaries. The former can result in a segfault.

Parameters

j	

Returns

double&

The documentation for this class was generated from the following file:

- include/epiworld/agent-bones.hpp

12.6 epiworld::Agent< TSeq > Class Template Reference

Agent (agents)

```
#include <epiworld.hpp>
```

Public Member Functions

- **Agent** (const Agent< TSeq > &p)
- int get_id () const

  Id of the individual.
- std::mt19937 * **get_rand_endgine** ()
- Model< TSeq > * **get_model** ()
- VirusPtr< TSeq > & **get_virus** (int i)
- Viruses< TSeq > **get_viruses** ()
- const Viruses_const< TSeq > **get_viruses** () const
- size_t **get_n_viruses** () const noexcept
- ToolPtr< TSeq > & **get_tool** (int i)
- Tools< TSeq > **get_tools** ()
- const Tools_const< TSeq > **get_tools** () const
- size_t **get_n_tools** () const noexcept
- void **mutate_variant** ()
- void **add_neighbor** (Agent< TSeq > *p, bool check_source=true, bool check_target=true)
- std::vector< Agent< TSeq > * > & **get_neighbors** ()
- void **change_status** (epiworld_fast_uint new_status, epiworld_fast_int queue=0)
- const epiworld_fast_uint & **get_status** () const
- void **reset** ()
- bool **has_tool** (unsigned int t) const
- bool **has_tool** (std::string name) const
- bool **has_virus** (unsigned int t) const
- bool **has_virus** (std::string name) const
- void **print** (bool compressed=false) const
- const std::vector< Entity< TSeq > * > **get_entities** ()

Add/Remove Virus/Tool

Any of these is ultimately reflected at the end of the iteration.

Parameters

tool	*Tool to add*
virus	*Virus to add*
status_new	*Status after the change*
queue	

- void **add_tool** (ToolPtr< TSeq > tool, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **add_tool** (Tool< TSeq > tool, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **add_virus** (VirusPtr< TSeq > virus, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **add_virus** (Virus< TSeq > virus, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **add_entity** (Entity< TSeq > &entity, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_tool** (epiworld_fast_uint tool_idx, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_tool** (ToolPtr< TSeq > &tool, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_virus** (epiworld_fast_uint virus_idx, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_virus** (VirusPtr< TSeq > &virus, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_entity** (epiworld_fast_uint entity_idx, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_entity** (Entity< TSeq > &entity, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void rm_agent_by_virus (epiworld_fast_uint virus_idx, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)

  Agent removed by virus.
- void rm_agent_by_virus (VirusPtr< TSeq > &virus, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)

  Agent removed by virus.

Get the rates (multipliers) for the agent

Parameters

v	*A pointer to a virus.*

Returns

epiworld_double

- epiworld_double **get_susceptibility_reduction** (VirusPtr< TSeq > v)
- epiworld_double **get_transmission_reduction** (VirusPtr< TSeq > v)
- epiworld_double **get_recovery_enhancer** (VirusPtr< TSeq > v)
- epiworld_double **get_death_reduction** (VirusPtr< TSeq > v)

- double & operator() (size_t j)

  Access the j-th column of the agent.
- double & **operator[ ]** (size_t j)

Friends

- class **Model**< **TSeq** >
- class **Virus**< **TSeq** >
- class **Viruses**< **TSeq** >
- class **Viruses_const**< **TSeq** >
- class **Tool**< **TSeq** >
- class **Tools**< **TSeq** >
- class **Queue**< **TSeq** >
- void **default_add_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_add_tool** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_add_entity** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_tool** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_entity** (Action< TSeq > &a, Model< TSeq > *m)

12.6.1 Detailed Description

template<typename TSeq = int>
class epiworld::Agent< TSeq >

Agent (agents)

Template Parameters

TSeq	Sequence type (should match `TSeq` across the model)

12.6.2 Member Function Documentation

12.6.2.1 operator()()

```
template<typename TSeq >
double & Agent< TSeq >::operator() (
             size_t j )  [inline]
```

Access the j-th column of the agent.

If an external array has been specified, then these two functions can be used to access additional agent's features not included in the model.

The `operator[]` method is with no boundary check, whereas the `operator()` method checks boundaries. The former can result in a segfault.

Parameters

j	

Returns

double&

The documentation for this class was generated from the following file:

- epiworld.hpp

12.7 AgentsSample< TSeq > Class Template Reference

Sample of agents.

```
#include <agentssample-bones.hpp>
```

Public Member Functions

- AgentsSample ()=delete
  Default constructor.
- AgentsSample (const AgentsSample< TSeq > &a)=delete
  Copy constructor.
- AgentsSample (AgentsSample< TSeq > &&a)=delete
  Move constructor.
- **AgentsSample** (Model< TSeq > &model_, size_t n)
- **AgentsSample** (Entity< TSeq > &entity_, size_t n)
- std::vector< Agent< TSeq > * >::iterator **begin** ()
- std::vector< Agent< TSeq > * >::iterator **end** ()
- Agent< TSeq > * **operator[ ]** (size_t n)
- Agent< TSeq > * **operator()** (size_t n)
- const size_t **size** () const noexcept

12.7.1 Detailed Description

template<typename TSeq>
class AgentsSample< TSeq >

Sample of agents.

This class allows sampling agents from Entity<TSeq> and Model<TSeq>.

Template Parameters

TSeq	

The documentation for this class was generated from the following file:

- include/epiworld/agentssample-bones.hpp

12.8 epiworld::AgentsSample< TSeq > Class Template Reference

Sample of agents.

`#include <epiworld.hpp>`

Public Member Functions

- AgentsSample ()=delete
  Default constructor.
- AgentsSample (const AgentsSample< TSeq > &a)=delete
  Copy constructor.
- AgentsSample (AgentsSample< TSeq > &&a)=delete
  Move constructor.

- **AgentsSample** (Model< TSeq > &model_, size_t n)
- **AgentsSample** (Entity< TSeq > &entity_, size_t n)
- std::vector< Agent< TSeq > * >::iterator **begin** ()
- std::vector< Agent< TSeq > * >::iterator **end** ()
- Agent< TSeq > * **operator[ ]** (size_t n)
- Agent< TSeq > * **operator()** (size_t n)
- const size_t **size** () const noexcept

12.8.1 Detailed Description

template<typename TSeq>
class epiworld::AgentsSample< TSeq >

Sample of agents.

This class allows sampling agents from Entity<TSeq> and Model<TSeq>.

Template Parameters

TSeq	

The documentation for this class was generated from the following file:

- epiworld.hpp

12.9 DataBase< TSeq > Class Template Reference

Statistical data about the process.

`#include <database-bones.hpp>`

Public Member Functions

- **DataBase** (Model< TSeq > &m)
- void record_variant (Virus< TSeq > &v)

  Registering a new variant.
- void **record_tool** (Tool< TSeq > &t)
- void **set_seq_hasher** (std::function< std::vector< int >(TSeq)> fun)
- void **set_model** (Model< TSeq > &m)
- Model< TSeq > * **get_model** ()
- void **record** ()
- const std::vector< TSeq > & **get_sequence** () const
- const std::vector< int > & **get_nexposed** () const
- size_t **size** () const
- void **write_data** (std::string fn_variant_info, std::string fn_variant_hist, std::string fn_tool_info, std::string fn_tool_hist, std::string fn_total_hist, std::string fn_transmission, std::string fn_transition, std::string fn_↩ reproductive_number) const
- void **record_transmission** (int i, int j, int variant, int i_expo_date)

- size_t **get_n_variants** () const
- size_t **get_n_tools** () const
- void **reset** ()
- void **set_user_data** (std::vector< std::string > names)
- void **add_user_data** (std::vector< epiworld_double > x)
- void **add_user_data** (unsigned int j, epiworld_double x)
- UserData< TSeq > & **get_user_data** ()
- std::vector< epiworld_double > transition_probability (bool print=true) const

  Calculates the transition probabilities.

Get recorded information from the model

Parameters

what	*std::string, The status, e.g., 0, 1, 2, ...*

Returns

In `get_today_total`, the current counts of `what`.
In `get_today_variant`, the current counts of `what` for each variant.
In `get_hist_total`, the time series of `what`
In `get_hist_variant`, the time series of what for each variant.
In `get_hist_total_date` and `get_hist_variant_date` the corresponding dates

- int **get_today_total** (std::string what) const
- int **get_today_total** (epiworld_fast_uint what) const
- void **get_today_total** (std::vector< std::string > *status=nullptr, std::vector< int > *counts=nullptr) const
- void **get_today_variant** (std::vector< std::string > &status, std::vector< int > &id, std::vector< int > &counts) const
- void **get_hist_total** (std::vector< int > *date, std::vector< std::string > *status, std::vector< int > *counts) const
- void **get_hist_variant** (std::vector< int > &date, std::vector< int > &id, std::vector< std::string > &status, std::vector< int > &counts) const

- MapVec_type< int, int > reproductive_number () const

  Computes the reproductive number of each case.
- void **reproductive_number** (std::string fn) const

Friends

- class **Model**< **TSeq** >
- void **default_add_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_add_tool** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_tool** (Action< TSeq > &a, Model< TSeq > *m)

12.9.1 Detailed Description

template<typename TSeq>
class DataBase< TSeq >

Statistical data about the process.

Template Parameters

TSeq	

12.9.2 Member Function Documentation

12.9.2.1 record_variant()

```
template<typename TSeq >
void DataBase< TSeq >::record_variant (
            Virus< TSeq > & v )  [inline]
```

Registering a new variant.

Parameters

v	Pointer to the new variant. Since variants are originated in the agent, the numbers simply move around. From the parent variant to the new variant. And the total number of infected does not change.

12.9.2.2 reproductive_number()

```
template<typename TSeq >
MapVec_type< int, int > DataBase< TSeq >::reproductive_number  [inline]
```

Computes the reproductive number of each case.

By definition, whereas it computes R0 (basic reproductive number) or Rt/R (the effective reproductive number) will depend on whether the virus is allowed to circulate naïvely or not, respectively.

Parameters

fn	File where to write out the reproductive number.

12.9.2.3 transition_probability()

```
template<typename TSeq >
std::vector< epiworld_double > DataBase< TSeq >::transition_probability (
            bool print = true ) const  [inline]
```

Calculates the transition probabilities.

Returns

std::vector< epiworld_double >

The documentation for this class was generated from the following files:

- include/epiworld/database-bones.hpp
- include/epiworld/database-meat.hpp

12.10 epiworld::DataBase< TSeq > Class Template Reference

Statistical data about the process.

`#include <epiworld.hpp>`

Public Member Functions

- **DataBase** (Model< TSeq > &m)
- void record_variant (Virus< TSeq > &v)

  Registering a new variant.
- void **record_tool** (Tool< TSeq > &t)
- void **set_seq_hasher** (std::function< std::vector< int >(TSeq)> fun)
- void **set_model** (Model< TSeq > &m)
- Model< TSeq > * **get_model** ()
- void **record** ()
- const std::vector< TSeq > & **get_sequence** () const
- const std::vector< int > & **get_nexposed** () const
- size_t **size** () const
- void **write_data** (std::string fn_variant_info, std::string fn_variant_hist, std::string fn_tool_info, std::string fn_tool_hist, std::string fn_total_hist, std::string fn_transmission, std::string fn_transition, std::string fn_↩ reproductive_number) const
- void **record_transmission** (int i, int j, int variant, int i_expo_date)
- size_t **get_n_variants** () const
- size_t **get_n_tools** () const
- void **reset** ()
- void **set_user_data** (std::vector< std::string > names)
- void **add_user_data** (std::vector< epiworld_double > x)
- void **add_user_data** (unsigned int j, epiworld_double x)
- UserData< TSeq > & **get_user_data** ()
- std::vector< epiworld_double > transition_probability (bool print=true) const

  Calculates the transition probabilities.
- **DataBase** (int freq=1)
- void record_variant (Virus< TSeq > *v)

  Registering a new variant.
- void **set_seq_hasher** (std::function< std::vector< int >(TSeq)> fun)
- void **set_model** (Model< TSeq > &m)
- Model< TSeq > * **get_model** ()
- void **record** ()
- const std::vector< TSeq > & **get_sequence** () const
- const std::vector< int > & **get_nexposed** () const
- size_t **size** () const

- void **up_exposed** (Virus< TSeq > *v, epiworld_fast_uint new_status)
- void **down_exposed** (Virus< TSeq > *v, epiworld_fast_uint prev_status)
- void **state_change** (epiworld_fast_uint prev_status, epiworld_fast_uint new_status)
- void **record_transition** (epiworld_fast_uint from, epiworld_fast_uint to)
- int get_today_total (std::string what) const

  Get recorded information from the model.
- int **get_today_total** (epiworld_fast_uint what) const
- void **get_today_total** (std::vector< std::string > *status=nullptr, std::vector< int > *counts=nullptr) const
- void **get_today_variant** (std::vector< std::string > &status, std::vector< int > &id, std::vector< int > &counts) const
- void **get_hist_total** (std::vector< int > *date, std::vector< std::string > *status, std::vector< int > *counts) const
- void **get_hist_variant** (std::vector< int > &date, std::vector< int > &id, std::vector< std::string > &status, std::vector< int > &counts) const
- void write_data (std::string fn_variant_info, std::string fn_variant_hist, std::string fn_total_hist, std::string fn_transmission, std::string fn_transition) const

  @]
- void **record_transmission** (int i, int j, int variant)
- size_t **get_nvariants** () const
- void **reset** ()
- void **set_user_data** (std::vector< std::string > names)
- void **add_user_data** (std::vector< epiworld_double > x)
- void **add_user_data** (unsigned int j, epiworld_double x)
- UserData< TSeq > & **get_user_data** ()

Get recorded information from the model

Parameters

what	*std::string, The status, e.g., 0, 1, 2, ...*

Returns

In `get_today_total`, the current counts of `what`.

In `get_today_variant`, the current counts of `what` for each variant.

In `get_hist_total`, the time series of `what`

In `get_hist_variant`, the time series of what for each variant.

In `get_hist_total_date` and `get_hist_variant_date` the corresponding dates

- int **get_today_total** (std::string what) const
- int **get_today_total** (epiworld_fast_uint what) const
- void **get_today_total** (std::vector< std::string > *status=nullptr, std::vector< int > *counts=nullptr) const
- void **get_today_variant** (std::vector< std::string > &status, std::vector< int > &id, std::vector< int > &counts) const
- void **get_hist_total** (std::vector< int > *date, std::vector< std::string > *status, std::vector< int > *counts) const
- void **get_hist_variant** (std::vector< int > &date, std::vector< int > &id, std::vector< std::string > &status, std::vector< int > &counts) const

- MapVec_type< int, int > reproductive_number () const

  Computes the reproductive number of each case.
- void **reproductive_number** (std::string fn) const

Friends

- class **Model**< **TSeq** >
- void **default_add_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_add_tool** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_tool** (Action< TSeq > &a, Model< TSeq > *m)

12.10.1 Detailed Description

template<**typename TSeq**>
class epiworld::DataBase< **TSeq** >

Statistical data about the process.

Template Parameters

TSeq	

12.10.2 Member Function Documentation

12.10.2.1 get_today_total()

```
template<typename TSeq >
int epiworld::DataBase< TSeq >::get_today_total (
            std::string what ) const
```

Get recorded information from the model.

Parameters

what	std::string, The status, e.g., 0, 1, 2, ...

Returns

In `get_today_total`, the current counts of `what`.

In `get_today_variant`, the current counts of `what` for each variant.

In `get_hist_total`, the time series of `what`

In `get_hist_variant`, the time series of what for each variant.

In `get_hist_total_date` and `get_hist_variant_date` the corresponding dates @[

12.10.2.2 record_variant() [1/2]

```
template<typename TSeq >
void DataBase< TSeq >::record_variant (
            Virus< TSeq > & v )  [inline]
```

Registering a new variant.

Parameters

v	Pointer to the new variant. Since variants are originated in the agent, the numbers simply move around. From the parent variant to the new variant. And the total number of infected does not change.

12.10.2.3 record_variant() [2/2]

```
template<typename TSeq >
void DataBase< TSeq >::record_variant (
            Virus< TSeq > * v )  [inline]
```

Registering a new variant.

Parameters

v	Pointer to the new variant. Since variants are originated in the host, the numbers simply move around. From the parent variant to the new variant. And the total number of infected does not change.

12.10.2.4 reproductive_number()

```
template<typename TSeq >
MapVec_type< int, int > DataBase< TSeq >::reproductive_number  [inline]
```

Computes the reproductive number of each case.

By definition, whereas it computes R0 (basic reproductive number) or Rt/R (the effective reproductive number) will depend on whether the virus is allowed to circulate naïvely or not, respectively.

Parameters

fn	File where to write out the reproductive number.

12.10.2.5 transition_probability()

```
template<typename TSeq >
```

```
std::vector< epiworld_double > DataBase< TSeq >::transition_probability (
            bool print = true ) const  [inline]
```

Calculates the transition probabilities.

Returns

std::vector< epiworld_double >

The documentation for this class was generated from the following file:

- epiworld.hpp

12.11 Entity< TSeq > Class Template Reference

Public Member Functions

- **Entity** (std::string name)
- void **add_agent** (Agent< TSeq > &p)
- void **add_agent** (Agent< TSeq > *p)
- void **rm_agent** (size_t idx)
- size_t **size** () const noexcept
- void **set_location** (std::vector< epiworld_double > loc)
- std::vector< epiworld_double > & **get_location** ()
- std::vector< Agent< TSeq > * >::iterator **begin** ()
- std::vector< Agent< TSeq > * >::iterator **end** ()
- std::vector< Agent< TSeq > * >::const_iterator **begin** () const
- std::vector< Agent< TSeq > * >::const_iterator **end** () const
- int **get_id** () const noexcept
- const std::string & **get_name** () const noexcept
- void **set_status** (epiworld_fast_int init, epiworld_fast_int post)
- void **set_queue** (epiworld_fast_int init, epiworld_fast_int post)
- void **get_status** (epiworld_fast_int *init, epiworld_fast_int *post)
- void **get_queue** (epiworld_fast_int *init, epiworld_fast_int *post)

Friends

- class **Agent**< **TSeq** >
- class **AgentsSample**< **TSeq** >
- class **Model**< **TSeq** >
- void **default_add_entity** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_entity** (Action< TSeq > &a, Model< TSeq > *m)

The documentation for this class was generated from the following files:

- include/epiworld/agentssample-bones.hpp
- include/epiworld/entity-bones.hpp

12.12 epiworld::Entity< TSeq > Class Template Reference

Public Member Functions

- **Entity** (std::string name)
- void **add_agent** (Agent< TSeq > &p)
- void **add_agent** (Agent< TSeq > *p)
- void **rm_agent** (size_t idx)
- size_t **size** () const noexcept
- void **set_location** (std::vector< epiworld_double > loc)
- std::vector< epiworld_double > & **get_location** ()
- std::vector< Agent< TSeq > * >::iterator **begin** ()
- std::vector< Agent< TSeq > * >::iterator **end** ()
- std::vector< Agent< TSeq > * >::const_iterator **begin** () const
- std::vector< Agent< TSeq > * >::const_iterator **end** () const
- int **get_id** () const noexcept
- const std::string & **get_name** () const noexcept
- void **set_status** (epiworld_fast_int init, epiworld_fast_int post)
- void **set_queue** (epiworld_fast_int init, epiworld_fast_int post)
- void **get_status** (epiworld_fast_int *init, epiworld_fast_int *post)
- void **get_queue** (epiworld_fast_int *init, epiworld_fast_int *post)

Friends

- class **Agent< TSeq >**
- class **AgentsSample< TSeq >**
- class **Model< TSeq >**
- void **default_add_entity** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_entity** (Action< TSeq > &a, Model< TSeq > *m)

The documentation for this class was generated from the following file:

- epiworld.hpp

12.13 epiworld::LFMCMC< TData > Class Template Reference

Likelihood-Free Markov Chain Monte Carlo.

```
#include <epiworld.hpp>
```

Public Member Functions

- void **run** (std::vector< epiworld_double > param_init, size_t n_samples_, epiworld_double epsilon_)
- **LFMCMC** (TData &observed_data_)
- void **set_observed_data** (TData &observed_data_)
- void **set_proposal_fun** (LFMCMCProposalFun< TData > fun)
- void **set_simulation_fun** (LFMCMCSimFun< TData > fun)
- void **set_summary_fun** (LFMCMCSummaryFun< TData > fun)
- void **set_kernel_fun** (LFMCMCKernelFun< TData > fun)
- const size_t **get_n_samples** ()
- const size_t **get_n_statistics** ()
- const size_t **get_n_parameters** ()
- const epiworld_double **get_epsilon** ()
- const std::vector< epiworld_double > & **get_params_now** ()
- const std::vector< epiworld_double > & **get_params_prev** ()
- const std::vector< epiworld_double > & **get_params_init** ()
- const std::vector< epiworld_double > & **get_statistics_obs** ()
- const std::vector< epiworld_double > & **get_statistics_hist** ()
- const std::vector< bool > & **get_statistics_accepted** ()
- const std::vector< epiworld_double > & **get_posterior_lf_prob** ()
- const std::vector< epiworld_double > & **get_drawn_prob** ()
- std::vector< TData > * **get_sampled_data** ()
- void **set_par_names** (std::vector< std::string > names)
- void **set_stats_names** (std::vector< std::string > names)
- void **print** ()

Random number generation

Parameters

eng	

- void **set_rand_engine** (std::mt19937 &eng)
- std::mt19937 * **get_rand_endgine** ()
- void **seed** (unsigned int s)
- void **set_rand_gamma** (epiworld_double alpha, epiworld_double beta)
- epiworld_double **runif** ()
- epiworld_double **rnorm** ()
- epiworld_double **rgamma** ()
- epiworld_double **runif** (epiworld_double lb, epiworld_double ub)
- epiworld_double **rnorm** (epiworld_double mean, epiworld_double sd)
- epiworld_double **rgamma** (epiworld_double alpha, epiworld_double beta)

12.13.1 Detailed Description

template<typename TData>
class epiworld::LFMCMC< TData >

Likelihood-Free Markov Chain Monte Carlo.

Template Parameters

TData	Type of data that is generated

The documentation for this class was generated from the following file:

- epiworld.hpp

12.14 LFMCMC< TData > Class Template Reference

Likelihood-Free Markov Chain Monte Carlo.

`#include <lfmcmc-bones.hpp>`

Public Member Functions

- void **run** (std::vector< epiworld_double > param_init, size_t n_samples_, epiworld_double epsilon_)
- **LFMCMC** (TData &observed_data_)
- void **set_observed_data** (TData &observed_data_)
- void **set_proposal_fun** (LFMCMCProposalFun< TData > fun)
- void **set_simulation_fun** (LFMCMCSimFun< TData > fun)
- void **set_summary_fun** (LFMCMCSummaryFun< TData > fun)
- void **set_kernel_fun** (LFMCMCKernelFun< TData > fun)
- const size_t **get_n_samples** ()
- const size_t **get_n_statistics** ()
- const size_t **get_n_parameters** ()
- const epiworld_double **get_epsilon** ()
- const std::vector< epiworld_double > & **get_params_now** ()
- const std::vector< epiworld_double > & **get_params_prev** ()
- const std::vector< epiworld_double > & **get_params_init** ()
- const std::vector< epiworld_double > & **get_statistics_obs** ()
- const std::vector< epiworld_double > & **get_statistics_hist** ()
- const std::vector< bool > & **get_statistics_accepted** ()
- const std::vector< epiworld_double > & **get_posterior_lf_prob** ()
- const std::vector< epiworld_double > & **get_drawn_prob** ()
- std::vector< TData > * **get_sampled_data** ()
- void **set_par_names** (std::vector< std::string > names)
- void **set_stats_names** (std::vector< std::string > names)
- void **print** ()

Random number generation

Parameters

eng	

- void **set_rand_engine** (std::mt19937 &eng)
- std::mt19937 * **get_rand_endgine** ()
- void **seed** (unsigned int s)
- void **set_rand_gamma** (epiworld_double alpha, epiworld_double beta)
- epiworld_double **runif** ()
- epiworld_double **rnorm** ()
- epiworld_double **rgamma** ()
- epiworld_double **runif** (epiworld_double lb, epiworld_double ub)

- epiworld_double **rnorm** (epiworld_double mean, epiworld_double sd)
- epiworld_double **rgamma** (epiworld_double alpha, epiworld_double beta)

12.14.1 Detailed Description

template<typename TData>
class LFMCMC< TData >

Likelihood-Free Markov Chain Monte Carlo.

Template Parameters

TData	Type of data that is generated

The documentation for this class was generated from the following files:

- include/epiworld/math/lfmcmc/lfmcmc-bones.hpp
- include/epiworld/math/lfmcmc/lfmcmc-meat-print.hpp
- include/epiworld/math/lfmcmc/lfmcmc-meat.hpp

12.15 epiworld::Model< TSeq > Class Template Reference

Core class of epiworld.

`#include <epiworld.hpp>`

Public Member Functions

- DataBase< TSeq > & **get_db** ()
- epiworld_double & **operator()** (std::string pname)
- size_t **size** () const
- size_t **get_n_variants** () const
- size_t **get_n_tools** () const
- unsigned int **get_ndays** () const
- unsigned int **get_n_replicates** () const
- void **set_ndays** (unsigned int ndays)
- bool **get_verbose** () const
- void **verbose_off** ()
- void **verbose_on** ()
- int today () const

  The current time of the model.
- void write_data (std::string fn_variant_info, std::string fn_variant_hist, std::string fn_tool_info, std::string fn_tool_hist, std::string fn_total_hist, std::string fn_transmission, std::string fn_transition, std::string fn_reproductive_number) const

  Wrapper of `DataBase::write_data`
- std::map< std::string, epiworld_double > & **params** ()
- void reset ()

  Reset the model.

- void **print** () const
- Model< TSeq > && **clone** () const
- void **get_elapsed** (std::string unit="auto", epiworld_double *last_elapsed=nullptr, epiworld_double *total_elapsed=nullptr, std::string *unit_abbr=nullptr, bool print=true) const
- void add_global_action (std::function< void(Model< TSeq > *)> fun, int date=-99)

  Set a global action.
- void **run_global_actions** ()
- void **clear_status_set** ()
- const std::vector< VirusPtr< TSeq > > & **get_viruses** () const
- const std::vector< ToolPtr< TSeq > > & **get_tools** () const
- void set_agents_data (double *data_, size_t ncols_)

  Set the agents data object.
- **Model** (const Model< TSeq > &m)
- **Model** (Model< TSeq > &&m)
- Model< TSeq > & **operator=** (const Model< TSeq > &m)
- void **clone_population** (std::vector< Person< TSeq > > &p, std::map< int, int > &p_ids, bool &d, Model< TSeq > *m=nullptr) const
- void **clone_population** (const Model< TSeq > &m)
- void set_backup ()

  Set the backup object.
- void **restore_backup** ()
- DataBase< TSeq > & get_db ()

  @]
- epiworld_double & **operator()** (std::string pname)
- size_t **size** () const
- void set_rand_engine (std::mt19937 &eng)

  Random number generation.
- std::mt19937 * **get_rand_endgine** ()
- void **seed** (unsigned int s)
- void **set_rand_gamma** (epiworld_double alpha, epiworld_double beta)
- epiworld_double **runif** ()
- epiworld_double **rnorm** ()
- epiworld_double **rnorm** (epiworld_double mean, epiworld_double sd)
- epiworld_double **rgamma** ()
- epiworld_double **rgamma** (epiworld_double alpha, epiworld_double beta)
- void add_virus (Virus< TSeq > v, epiworld_double preval)

  @]
- void **add_virus_n** (Virus< TSeq > v, unsigned int preval)
- void **add_tool** (Tool< TSeq > t, epiworld_double preval)
- void **add_tool_n** (Tool< TSeq > t, unsigned int preval)
- void pop_from_adjlist (std::string fn, int skip=0, bool directed=false, int min_id=-1, int max_id=-1)

  Accessing population of the model.
- void **pop_from_adjlist** (AdjList al)
- bool **is_directed** () const
- std::vector< Person< TSeq > > * **get_population** ()
- void **pop_from_random** (unsigned int n=1000, unsigned int k=5, bool d=false, epiworld_double p=.01)
- void init (unsigned int ndays, unsigned int seed)

  @]
- void **update_status** ()
- void **mutate_variant** ()
- void **next** ()
- void **run** ()

- void **run_multiple** (unsigned int nexperiments, std::function< void(Model< TSeq > *)> fun, bool reset, bool verbose)
- void record_variant (Virus< TSeq > *v)

  @]
- int **get_nvariants** () const
- unsigned int **get_ndays** () const
- void **set_ndays** (unsigned int ndays)
- bool **get_verbose** () const
- void **verbose_off** ()
- void **verbose_on** ()
- int **today** () const
- void set_rewire_fun (std::function< void(std::vector< Person< TSeq >> *, Model< TSeq > *, epiworld_double)> fun)

  Rewire the network preserving the degree sequence.
- void **set_rewire_prop** (epiworld_double prop)
- epiworld_double **get_rewire_prop** () const
- void **rewire** ()
- void set_update_susceptible (UpdateFun< TSeq > fun)

  @]
- void **set_update_exposed** (UpdateFun< TSeq > fun)
- void **set_update_removed** (UpdateFun< TSeq > fun)
- void write_data (std::string fn_variant_info, std::string fn_variant_hist, std::string fn_total_hist, std::string fn_transmission, std::string fn_transition) const

  Wrapper of `DataBase::write_data`
- void write_edgelist (std::string fn) const

  Export the network data in edgelist form.
- void **write_edgelist** (std::vector< unsigned int > &source, std::vector< unsigned int > &target) const
- std::map< std::string, epiworld_double > & params ()

  @]
- void reset ()

  Reset the model.
- void **print** () const
- Model< TSeq > && **clone** () const
- void add_status_susceptible (epiworld_fast_uint s, std::string lab)

  Adds extra statuses to the model.
- void **add_status_exposed** (epiworld_fast_uint s, std::string lab)
- void **add_status_removed** (epiworld_fast_uint s, std::string lab)
- void **add_status_susceptible** (std::string lab)
- void **add_status_exposed** (std::string lab)
- void **add_status_removed** (std::string lab)
- const std::vector< epiworld_fast_uint > & **get_status_susceptible** () const
- const std::vector< epiworld_fast_uint > & **get_status_exposed** () const
- const std::vector< epiworld_fast_uint > & **get_status_removed** () const
- const std::vector< std::string > & **get_status_susceptible_labels** () const
- const std::vector< std::string > & **get_status_exposed_labels** () const
- const std::vector< std::string > & **get_status_removed_labels** () const
- void **print_status_codes** () const
- epiworld_fast_uint **get_default_susceptible** () const
- epiworld_fast_uint **get_default_exposed** () const
- epiworld_fast_uint **get_default_removed** () const
- void reset_status_codes (std::vector< epiworld_fast_uint > codes, std::vector< std::string > names, bool verbose=true)

  @]

- epiworld_double add_param (epiworld_double initial_val, std::string pname)

  Setting and accessing parameters from the model.
- epiworld_double **set_param** (std::string pname)
- epiworld_double **get_param** (unsigned int k)
- epiworld_double **get_param** (std::string pname)
- epiworld_double **par** (unsigned int k)
- epiworld_double **par** (std::string pname)
- void get_elapsed (std::string unit="auto", epiworld_double *last_elapsed=nullptr, epiworld_double *total_↩ elapsed=nullptr, unsigned int *n_replicates=nullptr, std::string *unit_abbr=nullptr, bool print=true) const

  @]
- void set_user_data (std::vector< std::string > names)

  Set the user data object.
- void **add_user_data** (unsigned int j, epiworld_double x)
- void **add_user_data** (std::vector< epiworld_double > x)
- UserData< TSeq > & **get_user_data** ()
- void add_global_action (std::function< void(Model< TSeq > *)> fun, int date)

  @]
- void **run_global_actions** ()
- void **clear_status_set** ()
- void **toggle_visited** ()
- void **queuing_on** ()
- void **queuing_off** ()
- bool **is_queuing_on** () const
- Queue< TSeq > & **get_queue** ()

Set the backup object

`backup` *can be used to restore the entire object after a run. This can be useful if the user wishes to have individuals start with the same network from the beginning.*

- void **set_backup** ()
- void **restore_backup** ()

Random number generation

Parameters

eng	*Random number generator*
s	*Seed*

- void **set_rand_engine** (std::mt19937 &eng)
- std::mt19937 * **get_rand_endgine** ()
- void **seed** (unsigned int s)
- void **set_rand_gamma** (epiworld_double alpha, epiworld_double beta)
- epiworld_double **runif** ()
- epiworld_double **rnorm** ()
- epiworld_double **rnorm** (epiworld_double mean, epiworld_double sd)
- epiworld_double **rgamma** ()
- epiworld_double **rgamma** (epiworld_double alpha, epiworld_double beta)

Add Virus/Tool to the model

This is done before the model has been initialized.

Parameters

v	*Virus to be added*
t	*Tool to be added*
preval	*Initial prevalence (initial state.) It can be specified as a proportion (between zero and one,) or an integer indicating number of individuals.*

- void **add_virus** (Virus< TSeq > v, epiworld_double preval)
- void **add_virus_n** (Virus< TSeq > v, unsigned int preval)
- void **add_tool** (Tool< TSeq > t, epiworld_double preval)
- void **add_tool_n** (Tool< TSeq > t, unsigned int preval)
- void **add_entity** (Entity< TSeq > e, epiworld_double preval)
- void **add_entity_n** (Entity< TSeq > e, unsigned int preval)
- void **add_entity_fun** (Entity< TSeq > e, EntityToAgentFun< TSeq > fun)

Accessing population of the model

Parameters

fn	*std::string Filename of the edgelist file.*
skip	*int Number of lines to skip in `fn`.*
directed	*bool Whether the graph is directed or not.*
size	*Size of the network.*
al	*AdjList to read into the model.*

- void **agents_from_adjlist** (std::string fn, int size, int skip=0, bool directed=false)
- void **agents_from_adjlist** (AdjList al)
- bool **is_directed** () const
- std::vector< Agent< TSeq > > * **get_agents** ()
- void **agents_smallworld** (unsigned int n=1000, unsigned int k=5, bool d=false, epiworld_double p=.01)

Functions to run the model

Parameters

seed	*Seed to be used for Pseudo-RNG.*
ndays	*Number of days (steps) of the simulation.*
fun	*In the case of `run_multiple`, a function that is called after each experiment.*

- void **init** (unsigned int ndays, unsigned int seed)
- void **update_status** ()
- void **mutate_variant** ()
- void **next** ()
- void run ()

  Runs the simulation (after initialization)
- void run_multiple (unsigned int nexperiments, std::function< void(size_t, Model< TSeq > *)> fun=save_run< TSeq >(), bool reset=true, bool verbose=true)

Rewire the network preserving the degree sequence.

This implementation assumes an undirected network, thus if {(i,j), (k,l)} -> {(i,l), (k,j)}, the reciprocal is also true, i.e., {(j,i), (l,k)} -> {(j,k), (l,i)}.

Parameters

proportion	*Proportion of ties to be rewired.*

Returns

A rewired version of the network.

- void **set_rewire_fun** (std::function< void(std::vector< Agent< TSeq >> *, Model< TSeq > *, epiworld_double)> fun)
- void **set_rewire_prop** (epiworld_double prop)
- epiworld_double **get_rewire_prop** () const
- void **rewire** ()

Export the network data in edgelist form

Parameters

fn	*std::string. File name.*
source	*Integer vector*
target	*Integer vector*

When passing the source and target, the function will write the edgelist on those.

- void **write_edgelist** (std::string fn) const
- void **write_edgelist** (std::vector< unsigned int > &source, std::vector< unsigned int > &target) const

Manage status (states) in the model

The functions `get_status` return the current values for the statuses included in the model.

Parameters

lab	`std::string` *Name of the status.*

Returns

`add_status*` *returns nothing.*

`get_status_*` *returns a vector of pairs with the statuses and their labels.*

- void **add_status** (std::string lab, UpdateFun< TSeq > fun=nullptr)
- const std::vector< std::string > & **get_status** () const
- const std::vector< UpdateFun< TSeq > > & **get_status_fun** () const
- void **print_status_codes** () const

Set the user data object

Parameters

names	*string vector with the names of the variables.*

- void set_user_data (std::vector< std::string > names)
  [@
- void **add_user_data** (unsigned int j, epiworld_double x)
- void **add_user_data** (std::vector< epiworld_double > x)

- UserData< TSeq > & **get_user_data** ()

Queuing system

When queueing is on, the model will keep track of which agents are either in risk of exposure or exposed. This then is used at each step to act only on the aforementioned agents.

- void queuing_on ()
  Activates the queuing system (default.)
- void queuing_off ()
  Deactivates the queuing system.
- bool is_queuing_on () const
  Query if the queuing system is on.
- Queue< TSeq > & get_queue ()
  Retrieve the `Queue` object.

Get the susceptibility reduction object

Parameters

v	

Returns

epiworld_double

- void **set_susceptibility_reduction_mixer** (MixerFun< TSeq > fun)
- void **set_transmission_reduction_mixer** (MixerFun< TSeq > fun)
- void **set_recovery_enhancer_mixer** (MixerFun< TSeq > fun)
- void **set_death_reduction_mixer** (MixerFun< TSeq > fun)

Friends

- class **Agent**< **TSeq** >
- class **AgentsSample**< **TSeq** >
- class **DataBase**< **TSeq** >
- class **Queue**< **TSeq** >
- class **Person**< **TSeq** >

Tool Mixers

These functions combine the effects tools have to deliver a single effect. For example, wearing a mask, been vaccinated, and the immune system combine together to jointly reduce the susceptibility for a given virus.

- std::vector< epiworld_double > **array_double_tmp**
- std::vector< Virus< TSeq > * > **array_virus_tmp**
- **Model** ()
- **Model** (const Model< TSeq > &m)
- **Model** (Model< TSeq > &&m)
- Model< TSeq > & **operator=** (const Model< TSeq > &m)
- void **clone_population** (std::vector< Agent< TSeq > > &p, bool &d, Model< TSeq > *m=nullptr) const
- void **clone_population** (const Model< TSeq > &m)

Setting and accessing parameters from the model

Tools can incorporate parameters included in the model. Internally, parameters in the tool are stored as pointers to an std::map<> of parameters in the model. Using the `unsigned int` method directly fetches the parameters in the order these were added to the tool. Accessing parameters via the `std::string` method involves searching the parameter directly in the std::map<> member of the model (so it is not recommended.)

The `par()` function members are aliases for `get_param()`.

Parameters

initial_val	
pname	Name of the parameter to add or to fetch

Returns

The current value of the parameter in the model.

- epiworld_double * **p0**
- epiworld_double * **p1**
- epiworld_double * **p2**
- epiworld_double * **p3**
- epiworld_double * **p4**
- epiworld_double * **p5**
- epiworld_double * **p6**
- epiworld_double * **p7**
- epiworld_double * **p8**
- epiworld_double * **p9**
- epiworld_double * **p10**
- epiworld_double * **p11**
- epiworld_double * **p12**
- epiworld_double * **p13**
- epiworld_double * **p14**
- epiworld_double * **p15**
- epiworld_double * **p16**
- epiworld_double * **p17**
- epiworld_double * **p18**
- epiworld_double * **p19**
- epiworld_double * **p20**
- epiworld_double * **p21**
- epiworld_double * **p22**
- epiworld_double * **p23**
- epiworld_double * **p24**
- epiworld_double * **p25**
- epiworld_double * **p26**
- epiworld_double * **p27**
- epiworld_double * **p28**
- epiworld_double * **p29**
- epiworld_double * **p30**
- epiworld_double * **p31**
- epiworld_double * **p32**
- epiworld_double * **p33**
- epiworld_double * **p34**

- epiworld_double * **p35**
- epiworld_double * **p36**
- epiworld_double * **p37**
- epiworld_double * **p38**
- epiworld_double * **p39**
- unsigned int **npar_used** = 0u
- epiworld_double **add_param** (epiworld_double initial_val, std::string pname)
- epiworld_double **get_param** (unsigned int k)
- epiworld_double **get_param** (std::string pname)
- epiworld_double **par** (unsigned int k)
- epiworld_double **par** (std::string pname)

12.15.1 Detailed Description

template<typename TSeq = bool>
class epiworld::Model< TSeq >

Core class of epiworld.

The model class provides the wrapper that puts together `Agent`, `Virus`, and `Tools`.

Template Parameters

TSeq	Type of sequence. In principle, users can build models in which virus and human sequence is represented as numeric vectors (if needed.)

12.15.2 Member Function Documentation

12.15.2.1 add_global_action()

```
template<typename TSeq >
void Model< TSeq >::add_global_action (
             std::function< void(Model< TSeq > *)> fun,
             int date = -99 )  [inline]
```

Set a global action.

Parameters

fun	A function to be called on the prescribed dates
date	Integer indicating when the function is called (see details)

When date is less than zero, then the function is called at the end of every day. Otherwise, the function will be called only at the end of the indicated date.

12.15.2.2 add_param()

```
template<typename TSeq = bool>
epiworld_double epiworld::Model< TSeq >::add_param (
             epiworld_double initial_val,
             std::string pname )
```

Setting and accessing parameters from the model.

Tools can incorporate parameters included in the model. Internally, parameters in the tool are stored as pointers to an std::map<> of parameters in the model. Using the `unsigned int` method directly fetches the parameters in the order these were added to the tool. Accessing parameters via the `std::string` method involves searching the parameter directly in the std::map<> member of the model (so it is not recommended.)

The function `set_param()` can be used when the parameter already exists in the model.

The `par()` function members are aliases for `get_param()`.

Parameters

initial_val	
pname	Name of the parameter to add or to fetch

Returns

The current value of the parameter in the model. @[

12.15.2.3 add_status_susceptible()

```
template<typename TSeq >
void Model< TSeq >::add_status_susceptible (
             epiworld_fast_uint s,
             std::string lab )  [inline]
```

Adds extra statuses to the model.

Adding values of `s` that are already present in the model will result in an error.

The functions `get_status_*` return the current values for the statuses included in the model.

Parameters

s	`unsigned int` Code of the status
lab	`std::string` Name of the status.

Returns

`add_status*` returns nothing.

`get_status_*` returns a vector of pairs with the statuses and their labels. @[

12.15.2.4 init()

```
template<typename TSeq = bool>
void epiworld::Model< TSeq >::init (
             unsigned int ndays,
             unsigned int seed )
```

@]

Functions to run the model

Parameters

seed	Seed to be used for Pseudo-RNG.
ndays	Number of days (steps) of the simulation.
fun	In the case of `run_multiple`, a function that is called after each experiment. @[

12.15.2.5 pop_from_adjlist()

```
template<typename TSeq >
void Model< TSeq >::pop_from_adjlist (
             std::string fn,
             int skip = 0,
             bool directed = false,
             int min_id = -1,
             int max_id = -1 )  [inline]
```

Accessing population of the model.

Parameters

fn	std::string Filename of the edgelist file.
skip	int Number of lines to skip in `fn`.
directed	bool Whether the graph is directed or not.
min_id	int Minimum id number (if negative, the program will try to guess from the data.)
max_id	int Maximum id number (if negative, the program will try to guess from the data.)
al	AdjList to read into the model. @[

12.15.2.6 reset() [1/2]

```
template<typename TSeq >
void Model< TSeq >::reset  [inline]
```

Reset the model.

Resetting the model will:

- clear the database
- restore the population (if `set_backup()` was called before)
- re-distribute tools
- re-distribute viruses
- set the date to 0

12.15.2.7 reset() [2/2]

```
template<typename TSeq = bool>
void epiworld::Model< TSeq >::reset ( )
```

Reset the model.

Resetting the model will:

- clear the database
- restore the population (if `set_backup()` was called before)
- re-distribute tools
- re-distribute viruses
- set the date to 0

12.15.2.8 reset_status_codes()

```
template<typename TSeq >
void Model< TSeq >::reset_status_codes (
             std::vector< epiworld_fast_uint > codes,
             std::vector< std::string > names,
             bool verbose = true )  [inline]
```

@]

Reset all the status codes of the model

The default values are those specified in the enum STATUS.

Parameters

codes	In the following order: Susceptible, Infected, Removed
names	Names matching the codes
verbose	When `true`, it will print the new mappings.

12.15.2.9 run_multiple()

```
template<typename TSeq >
void Model< TSeq >::run_multiple (
            unsigned int nexperiments,
            std::function< void(size_t, Model< TSeq > *)> fun = save_run<TSeq>(),
            bool reset = true,
            bool verbose = true )  [inline]
```

Parameters

nexperiments	Multiple runs of the simulation

12.15.2.10 set_agents_data()

```
template<typename TSeq >
void Model< TSeq >::set_agents_data (
            double * data_,
            size_t ncols_ )  [inline]
```

Set the agents data object.

The data should be an array with the data stored in a column major order, i.e., by column.

Parameters

*data_*	Pointer to the first element of an array of size `size() * ncols_`.
*ncols_*	Number of features included in the data.

12.15.2.11 set_backup()

```
template<typename TSeq = bool>
void epiworld::Model< TSeq >::set_backup ( )
```

Set the backup object.

`backup` can be used to restore the entire object after a run. This can be useful if the user wishes to have individuals start with the same network from the beginning. @[

12.15.2.12 set_rand_engine()

```
template<typename TSeq = bool>
void epiworld::Model< TSeq >::set_rand_engine (
             std::mt19937 & eng )
```

Random number generation.

Parameters

eng	@[

12.15.2.13 set_rewire_fun()

```
template<typename TSeq >
void Model< TSeq >::set_rewire_fun (
             std::function< void(std::vector< Person< TSeq >> *, Model< TSeq > *, epiworld↩
_double)> fun )  [inline]
```

Rewire the network preserving the degree sequence.

This implementation assumes an undirected network, thus if {(i,j), (k,l)} -> {(i,l), (k,j)}, the reciprocal is also true, i.e., {(j,i), (l,k)} -> {(j,k), (l,i)}.

Parameters

proportion	Proportion of ties to be rewired.

Returns

A rewired version of the network. @[

12.15.2.14 set_user_data()

```
template<typename TSeq = bool>
void epiworld::Model< TSeq >::set_user_data (
             std::vector< std::string > names )
```

Set the user data object.

Parameters

names	[@

12.15.2.15 write_data() [1/2]

```
template<typename TSeq >
void Model< TSeq >::write_data (
            std::string fn_variant_info,
            std::string fn_variant_hist,
            std::string fn_tool_info,
            std::string fn_tool_hist,
            std::string fn_total_hist,
            std::string fn_transmission,
            std::string fn_transition,
            std::string fn_reproductive_number ) const  [inline]
```

Wrapper of `DataBase::write_data`

Parameters

fn_variant_info	Filename. Information about the variant.
fn_variant_hist	Filename. History of the variant.
fn_tool_info	Filename. Information about the tool.
fn_tool_hist	Filename. History of the tool.
fn_total_hist	Filename. Aggregated history (status)
fn_transmission	Filename. Transmission history.
fn_transition	Filename. Markov transition history.
fn_reproductive_number	Filename. Case by case reproductive number

12.15.2.16 write_data() [2/2]

```
template<typename TSeq >
void Model< TSeq >::write_data (
            std::string fn_variant_info,
            std::string fn_variant_hist,
            std::string fn_total_hist,
            std::string fn_transmission,
            std::string fn_transition ) const  [inline]
```

Wrapper of `DataBase::write_data`

Parameters

fn_variant_info	Filename. Information about the variant.
fn_variant_hist	Filename. History of the variant.
fn_total_hist	Filename. Aggregated history (status)
fn_transmission	Filename. Transmission history.
fn_transition	Filename. Markov transition history.

12.15.2.17 write_edgelist()

```
template<typename TSeq = bool>
void epiworld::Model< TSeq >::write_edgelist (
             std::string fn ) const
```

Export the network data in edgelist form.

Parameters

fn	std::string. File name.
source	Integer vector
target	Integer vector

When passing the source and target, the function will write the edgelist on those. [@

The documentation for this class was generated from the following file:

- epiworld.hpp

12.16 Model< TSeq > Class Template Reference

Core class of epiworld.

```
#include <model-bones.hpp>
```

Public Member Functions

- DataBase< TSeq > & **get_db** ()
- epiworld_double & **operator()** (std::string pname)
- size_t **size** () const
- size_t **get_n_variants** () const
- size_t **get_n_tools** () const
- unsigned int **get_ndays** () const
- unsigned int **get_n_replicates** () const
- void **set_ndays** (unsigned int ndays)
- bool **get_verbose** () const
- void **verbose_off** ()
- void **verbose_on** ()
- int today () const

  The current time of the model.
- void write_data (std::string fn_variant_info, std::string fn_variant_hist, std::string fn_tool_info, std::string fn_tool_hist, std::string fn_total_hist, std::string fn_transmission, std::string fn_transition, std::string fn_reproductive_number) const

  Wrapper of `DataBase::write_data`
- std::map< std::string, epiworld_double > & **params** ()
- void reset ()

  Reset the model.
- void **print** () const
- Model< TSeq > && **clone** () const

- void **get_elapsed** (std::string unit="auto", epiworld_double *last_elapsed=nullptr, epiworld_double *total_↩ elapsed=nullptr, std::string *unit_abbr=nullptr, bool print=true) const
- void add_global_action (std::function< void(Model< TSeq > *)> fun, int date=-99)

  Set a global action.
- void **run_global_actions** ()
- void **clear_status_set** ()
- const std::vector< VirusPtr< TSeq > > & **get_viruses** () const
- const std::vector< ToolPtr< TSeq > > & **get_tools** () const
- void set_agents_data (double *data_, size_t ncols_)

  Set the agents data object.

Set the backup object

`backup` *can be used to restore the entire object after a run. This can be useful if the user wishes to have individuals start with the same network from the beginning.*

- void **set_backup** ()
- void **restore_backup** ()

Random number generation

Parameters

eng	*Random number generator*
s	*Seed*

- void **set_rand_engine** (std::mt19937 &eng)
- std::mt19937 * **get_rand_endgine** ()
- void **seed** (unsigned int s)
- void **set_rand_gamma** (epiworld_double alpha, epiworld_double beta)
- epiworld_double **runif** ()
- epiworld_double **rnorm** ()
- epiworld_double **rnorm** (epiworld_double mean, epiworld_double sd)
- epiworld_double **rgamma** ()
- epiworld_double **rgamma** (epiworld_double alpha, epiworld_double beta)

Add Virus/Tool to the model

This is done before the model has been initialized.

Parameters

v	*Virus to be added*
t	*Tool to be added*
preval	*Initial prevalence (initial state.) It can be specified as a proportion (between zero and one,) or an integer indicating number of individuals.*

- void **add_virus** (Virus< TSeq > v, epiworld_double preval)
- void **add_virus_n** (Virus< TSeq > v, unsigned int preval)
- void **add_tool** (Tool< TSeq > t, epiworld_double preval)
- void **add_tool_n** (Tool< TSeq > t, unsigned int preval)
- void **add_entity** (Entity< TSeq > e, epiworld_double preval)
- void **add_entity_n** (Entity< TSeq > e, unsigned int preval)
- void **add_entity_fun** (Entity< TSeq > e, EntityToAgentFun< TSeq > fun)

Accessing population of the model

Parameters

fn	*std::string Filename of the edgelist file.*
skip	*int Number of lines to skip in `fn`.*
directed	*bool Whether the graph is directed or not.*
size	*Size of the network.*
al	*AdjList to read into the model.*

- void **agents_from_adjlist** (std::string fn, int size, int skip=0, bool directed=false)
- void **agents_from_adjlist** (AdjList al)
- bool **is_directed** () const
- std::vector< Agent< TSeq > > * **get_agents** ()
- void **agents_smallworld** (unsigned int n=1000, unsigned int k=5, bool d=false, epiworld_double p=.01)

Functions to run the model

Parameters

seed	*Seed to be used for Pseudo-RNG.*
ndays	*Number of days (steps) of the simulation.*
fun	*In the case of `run_multiple`, a function that is called after each experiment.*

- void **init** (unsigned int ndays, unsigned int seed)
- void **update_status** ()
- void **mutate_variant** ()
- void **next** ()
- void run ()

  Runs the simulation (after initialization)
- void run_multiple (unsigned int nexperiments, std::function< void(size_t, Model< TSeq > *)> fun=save_run< TSeq >(), bool reset=true, bool verbose=true)

Rewire the network preserving the degree sequence.

This implementation assumes an undirected network, thus if {(i,j), (k,l)} -> {(i,l), (k,j)}, the reciprocal is also true, i.e., {(j,i), (l,k)} -> {(j,k), (l,i)}.

Parameters

proportion	*Proportion of ties to be rewired.*

Returns

A rewired version of the network.

- void **set_rewire_fun** (std::function< void(std::vector< Agent< TSeq >> *, Model< TSeq > *, epiworld_double)> fun)
- void **set_rewire_prop** (epiworld_double prop)
- epiworld_double **get_rewire_prop** () const
- void **rewire** ()

Export the network data in edgelist form

Parameters

fn	*std::string. File name.*

Parameters

source	*Integer vector*
target	*Integer vector*

When passing the source and target, the function will write the edgelist on those.

- void **write_edgelist** (std::string fn) const
- void **write_edgelist** (std::vector< unsigned int > &source, std::vector< unsigned int > &target) const

Manage status (states) in the model

The functions `get_status` return the current values for the statuses included in the model.

Parameters

lab	`std::string` *Name of the status.*

Returns

`add_status*` *returns nothing.*

`get_status_*` *returns a vector of pairs with the statuses and their labels.*

- void **add_status** (std::string lab, UpdateFun< TSeq > fun=nullptr)
- const std::vector< std::string > & **get_status** () const
- const std::vector< UpdateFun< TSeq > > & **get_status_fun** () const
- void **print_status_codes** () const

Set the user data object

Parameters

names	*string vector with the names of the variables.*

- void set_user_data (std::vector< std::string > names)
  [@
- void **add_user_data** (unsigned int j, epiworld_double x)
- void **add_user_data** (std::vector< epiworld_double > x)
- UserData< TSeq > & **get_user_data** ()

Queuing system

When queueing is on, the model will keep track of which agents are either in risk of exposure or exposed. This then is used at each step to act only on the aforementioned agents.

- void queuing_on ()
  Activates the queuing system (default.)
- void queuing_off ()
  Deactivates the queuing system.
- bool is_queuing_on () const
  Query if the queuing system is on.
- Queue< TSeq > & get_queue ()
  Retrieve the `Queue` object.

Get the susceptibility reduction object

Parameters

v	

Returns

epiworld_double

- void **set_susceptibility_reduction_mixer** (MixerFun< TSeq > fun)
- void **set_transmission_reduction_mixer** (MixerFun< TSeq > fun)
- void **set_recovery_enhancer_mixer** (MixerFun< TSeq > fun)
- void **set_death_reduction_mixer** (MixerFun< TSeq > fun)

Friends

- class **Agent**< **TSeq** >
- class **AgentsSample**< **TSeq** >
- class **DataBase**< **TSeq** >
- class **Queue**< **TSeq** >

Tool Mixers

These functions combine the effects tools have to deliver a single effect. For example, wearing a mask, been vaccinated, and the immune system combine together to jointly reduce the susceptibility for a given virus.

- std::vector< epiworld_double > **array_double_tmp**
- std::vector< Virus< TSeq > * > **array_virus_tmp**
- **Model** ()
- **Model** (const Model< TSeq > &m)
- **Model** (Model< TSeq > &&m)
- Model< TSeq > & **operator=** (const Model< TSeq > &m)
- void **clone_population** (std::vector< Agent< TSeq > > &p, bool &d, Model< TSeq > *m=nullptr) const
- void **clone_population** (const Model< TSeq > &m)

Setting and accessing parameters from the model

Tools can incorporate parameters included in the model. Internally, parameters in the tool are stored as pointers to an std::map<> of parameters in the model. Using the `unsigned int` method directly fetches the parameters in the order these were added to the tool. Accessing parameters via the `std::string` method involves searching the parameter directly in the std::map<> member of the model (so it is not recommended.)

The `par()` function members are aliases for `get_param()`.

Parameters

initial_val	
pname	Name of the parameter to add or to fetch

Returns

The current value of the parameter in the model.

- epiworld_double * **p0**
- epiworld_double * **p1**
- epiworld_double * **p2**
- epiworld_double * **p3**
- epiworld_double * **p4**
- epiworld_double * **p5**
- epiworld_double * **p6**
- epiworld_double * **p7**
- epiworld_double * **p8**
- epiworld_double * **p9**
- epiworld_double * **p10**
- epiworld_double * **p11**
- epiworld_double * **p12**
- epiworld_double * **p13**
- epiworld_double * **p14**
- epiworld_double * **p15**
- epiworld_double * **p16**
- epiworld_double * **p17**
- epiworld_double * **p18**
- epiworld_double * **p19**
- epiworld_double * **p20**
- epiworld_double * **p21**
- epiworld_double * **p22**
- epiworld_double * **p23**
- epiworld_double * **p24**
- epiworld_double * **p25**
- epiworld_double * **p26**
- epiworld_double * **p27**
- epiworld_double * **p28**
- epiworld_double * **p29**
- epiworld_double * **p30**
- epiworld_double * **p31**
- epiworld_double * **p32**
- epiworld_double * **p33**
- epiworld_double * **p34**
- epiworld_double * **p35**
- epiworld_double * **p36**
- epiworld_double * **p37**
- epiworld_double * **p38**
- epiworld_double * **p39**
- unsigned int **npar_used** = 0u
- epiworld_double **add_param** (epiworld_double initial_val, std::string pname)
- epiworld_double **get_param** (unsigned int k)
- epiworld_double **get_param** (std::string pname)
- epiworld_double **par** (unsigned int k)
- epiworld_double **par** (std::string pname)

12.16.1 Detailed Description

template<typename TSeq = int>
class Model< TSeq >

Core class of epiworld.

The model class provides the wrapper that puts together Agent, Virus, and Tools.

Template Parameters

TSeq	Type of sequence. In principle, users can build models in which virus and human sequence is represented as numeric vectors (if needed.)

12.16.2 Member Function Documentation

12.16.2.1 add_global_action()

```
template<typename TSeq = int>
void Model< TSeq >::add_global_action (
            std::function< void(Model< TSeq > *)> fun,
            int date = -99 )
```

Set a global action.

Parameters

fun	A function to be called on the prescribed dates
date	Integer indicating when the function is called (see details)

When date is less than zero, then the function is called at the end of every day. Otherwise, the function will be called only at the end of the indicated date.

12.16.2.2 reset()

```
template<typename TSeq = int>
void Model< TSeq >::reset ( )
```

Reset the model.

Resetting the model will:

- clear the database
- restore the population (if `set_backup()` was called before)
- re-distribute tools
- re-distribute viruses
- set the date to 0

12.16.2.3 run_multiple()

```
template<typename TSeq = int>
void Model< TSeq >::run_multiple (
            unsigned int nexperiments,
            std::function< void(size_t, Model< TSeq > *)> fun = save_run< TSeq >(),
            bool reset = true,
            bool verbose = true )
```

Parameters

nexperiments	Multiple runs of the simulation

12.16.2.4 set_agents_data()

```
template<typename TSeq = int>
void Model< TSeq >::set_agents_data (
            double * data_,
            size_t ncols_ )
```

Set the agents data object.

The data should be an array with the data stored in a column major order, i.e., by column.

Parameters

*data_*	Pointer to the first element of an array of size `size() * ncols_`.
*ncols_*	Number of features included in the data.

12.16.2.5 write_data()

```
template<typename TSeq = int>
void Model< TSeq >::write_data (
            std::string fn_variant_info,
            std::string fn_variant_hist,
            std::string fn_tool_info,
            std::string fn_tool_hist,
            std::string fn_total_hist,
            std::string fn_transmission,
            std::string fn_transition,
            std::string fn_reproductive_number ) const
```

Wrapper of `DataBase::write_data`

Parameters

fn_variant_info	Filename. Information about the variant.
fn_variant_hist	Filename. History of the variant.
fn_tool_info	Filename. Information about the tool.
fn_tool_hist	Filename. History of the tool.
fn_total_hist	Filename. Aggregated history (status)
fn_transmission	Filename. Transmission history.
fn_transition	Filename. Markov transition history.
fn_reproductive_number	Filename. Case by case reproductive number

The documentation for this class was generated from the following files:

- include/epiworld/agent-meat-status.hpp
- include/epiworld/model-bones.hpp

12.17 epiworld::Person< TSeq > Class Template Reference

Public Member Functions

- void **init** (epiworld_fast_uint baseline_status)
- void **add_tool** (int d, Tool< TSeq > tool)
- void **add_virus** (Virus< TSeq > *virus)
- void **rm_virus** (Virus< TSeq > *virus)
- epiworld_double **get_susceptibility_reduction** (Virus< TSeq > *v)
- epiworld_double **get_transmission_reduction** (Virus< TSeq > *v)
- epiworld_double **get_recovery_enhancer** (Virus< TSeq > *v)
- epiworld_double **get_death_reduction** (Virus< TSeq > *v)
- int **get_id** () const
- unsigned int **get_index** () const
- std::mt19937 * **get_rand_endgine** ()
- Model< TSeq > * **get_model** ()
- Virus< TSeq > & **get_virus** (int i)
- PersonViruses< TSeq > & **get_viruses** ()
- Tool< TSeq > & **get_tool** (int i)
- PersonTools< TSeq > & **get_tools** ()
- void **mutate_variant** ()
- void **add_neighbor** (Person< TSeq > *p, bool check_source=true, bool check_target=true)
- std::vector< Person< TSeq > * > & **get_neighbors** ()
- void **update_status** ()
- const epiworld_fast_uint & **get_status** () const
- void **reset** ()
- void **set_update_susceptible** (UpdateFun< TSeq > fun)
- void **set_update_exposed** (UpdateFun< TSeq > fun)
- void **set_update_removed** (UpdateFun< TSeq > fun)
- bool **has_tool** (unsigned int t) const
- bool **has_tool** (std::string name) const
- bool **has_virus** (unsigned int t) const
- bool **has_virus** (std::string name) const
- bool **visited** () const
- void **toggle_visited** ()

Friends

- class **Model**< **TSeq** >
- class **Tool**< **TSeq** >
- class **Queue**< **TSeq** >

The documentation for this class was generated from the following file:

- rpackage/inst/include/epiworld.hpp

12.18 epiworld::PersonTools< TSeq > Class Template Reference

List of tools available for the individual to.

`#include <epiworld.hpp>`

Public Member Functions

- void add_tool (int date, Tool< TSeq > tool)
  @]
- epiworld_double **get_susceptibility_reduction** (Virus< TSeq > *v)
- epiworld_double **get_transmission_reduction** (Virus< TSeq > *v)
- epiworld_double **get_recovery_enhancer** (Virus< TSeq > *v)
- epiworld_double **get_death_reduction** (Virus< TSeq > *v)
- void **set_susceptibility_reduction_mixer** (MixerFun< TSeq > fun)
- void **set_transmission_reduction_mixer** (MixerFun< TSeq > fun)
- void **set_recovery_enhancer_mixer** (MixerFun< TSeq > fun)
- void **set_death_reduction_mixer** (MixerFun< TSeq > fun)
- size_t **size** () const
- Tool< TSeq > & **operator()** (int i)
- Person< TSeq > * **get_person** ()
- Model< TSeq > * **get_model** ()
- void **reset** ()
- bool **has_tool** (unsigned int t) const
- bool **has_tool** (std::string name) const

Friends

- class **Person**< **TSeq** >
- class **Model**< **TSeq** >

12.18.1 Detailed Description

template<**typename TSeq = bool**>
class epiworld::PersonTools< **TSeq** >

List of tools available for the individual to.

Template Parameters

TSeq	

The documentation for this class was generated from the following file:

- epiworld.hpp

12.19 PersonTools< TSeq > Class Template Reference

The documentation for this class was generated from the following file:

- include/epiworld/config.hpp

12.20 epiworld::PersonViruses< TSeq > Class Template Reference

Set of viruses in host.

`#include <epiworld.hpp>`

Public Member Functions

- void **add_virus** (epiworld_fast_uint new_status, Virus< TSeq > v)
- size_t **size** () const
- int **size_active** () const
- Virus< TSeq > & **operator()** (int i)
- void **mutate** ()
- void **reset** ()
- void **deactivate** (Virus< TSeq > &v)
- Person< TSeq > * **get_host** ()
- bool **has_virus** (unsigned int v) const
- bool **has_virus** (std::string vname) const

Friends

- class **Person**< **TSeq** >
- class **Model**< **TSeq** >

12.20.1 Detailed Description

template<**typename TSeq = bool**>
class epiworld::PersonViruses< **TSeq** >

Set of viruses in host.

Template Parameters

TSeq	

The documentation for this class was generated from the following file:

- rpackage/inst/include/epiworld.hpp

12.21 epiworld::Progress Class Reference

A simple progress bar.

`#include <epiworld.hpp>`

Public Member Functions

- **Progress** (int n_, int width_)
- void **start** ()
- void **next** ()
- void **end** ()
- **Progress** (int n_, int width_)
- void **start** ()
- void **next** ()
- void **end** ()

12.21.1 Detailed Description

A simple progress bar.

The documentation for this class was generated from the following file:

- epiworld.hpp

12.22 Progress Class Reference

A simple progress bar.

`#include <progress.hpp>`

Public Member Functions

- **Progress** (int n_, int width_)
- void **start** ()
- void **next** ()
- void **end** ()

12.22.1 Detailed Description

A simple progress bar.

The documentation for this class was generated from the following file:

- include/epiworld/progress.hpp

12.23 epiworld::Queue< TSeq > Class Template Reference

Controls which agents are verified at each step.

`#include <epiworld.hpp>`

Public Member Functions

- void **operator+=** (Agent< TSeq > *p)
- void **operator-=** (Agent< TSeq > *p)
- epiworld_fast_int **operator[ ]** (unsigned int i) const
- void **set_model** (Model< TSeq > *m)
- void **operator+=** (Person< TSeq > *p)
- void **operator-=** (Person< TSeq > *p)
- epiworld_fast_int **operator[ ]** (unsigned int i) const
- void **set_model** (Model< TSeq > *m)
- void **update** ()

12.23.1 Detailed Description

template<typename TSeq = int>
class epiworld::Queue< TSeq >

Controls which agents are verified at each step.

The idea is that only agents who are either in an infected state or have an infected neighbor should be checked. Otherwise it makes no sense (no chance to recover or capture the disease).

Template Parameters

TSeq	

The documentation for this class was generated from the following file:

- epiworld.hpp

12.24 Queue< TSeq > Class Template Reference

Controls which agents are verified at each step.

`#include <queue-bones.hpp>`

Public Member Functions

- void **operator+=** (Agent< TSeq > *p)
- void **operator-=** (Agent< TSeq > *p)
- epiworld_fast_int **operator[ ]** (unsigned int i) const
- void **set_model** (Model< TSeq > *m)

12.24.1 Detailed Description

template<typename TSeq = int>
class Queue< TSeq >

Controls which agents are verified at each step.

The idea is that only agents who are either in an infected state or have an infected neighbor should be checked. Otherwise it makes no sense (no chance to recover or capture the disease).

Template Parameters

TSeq	

The documentation for this class was generated from the following files:

- include/epiworld/agent-bones.hpp
- include/epiworld/queue-bones.hpp

12.25 RandGraph Class Reference

Public Member Functions

- **RandGraph** (int N_)
- void **init** (int s)
- void **set_rand_engine** (std::mt19937 &e)
- epiworld_double **runif** ()

The documentation for this class was generated from the following file:

- include/epiworld/random_graph.hpp

12.26 epiworld::Tool< TSeq > Class Template Reference

Tools for defending the agent against the virus.

```
#include <epiworld.hpp>
```

Public Member Functions

- **Tool** (std::string name="unknown tool")
- void **set_sequence** (TSeq d)
- void **set_sequence_unique** (TSeq d)
- void **set_sequence** (std::shared_ptr< TSeq > d)
- std::shared_ptr< TSeq > **get_sequence** ()
- TSeq & **get_sequence_unique** ()
- void **set_name** (std::string name)
- std::string **get_name** () const
- Agent< TSeq > * **get_agent** ()
- int **get_id** () const
- void **set_id** (int id)
- void **set_date** (int d)
- int **get_date** () const
- void **set_status** (epiworld_fast_int init, epiworld_fast_int post)
- void **set_queue** (epiworld_fast_int init, epiworld_fast_int post)
- void **get_status** (epiworld_fast_int *init, epiworld_fast_int *post)
- void **get_queue** (epiworld_fast_int *init, epiworld_fast_int *post)
- **Tool** (std::string name="unknown tool")
- void **set_sequence** (TSeq d)
- void **set_sequence_unique** (TSeq d)
- void **set_sequence** (std::shared_ptr< TSeq > d)
- std::shared_ptr< TSeq > **get_sequence** ()
- TSeq & **get_sequence_unique** ()
- epiworld_double get_susceptibility_reduction (Virus< TSeq > *v)

  Get and set the tool functions.
- epiworld_double **get_transmission_reduction** (Virus< TSeq > *v)
- epiworld_double **get_recovery_enhancer** (Virus< TSeq > *v)
- epiworld_double **get_death_reduction** (Virus< TSeq > *v)
- void **set_susceptibility_reduction_fun** (ToolFun< TSeq > fun)
- void **set_transmission_reduction_fun** (ToolFun< TSeq > fun)
- void **set_recovery_enhancer_fun** (ToolFun< TSeq > fun)
- void **set_death_reduction_fun** (ToolFun< TSeq > fun)
- void **set_susceptibility_reduction** (epiworld_double *prob)
- void **set_transmission_reduction** (epiworld_double *prob)
- void **set_recovery_enhancer** (epiworld_double *prob)
- void **set_death_reduction** (epiworld_double *prob)
- void **set_susceptibility_reduction** (epiworld_double prob)
- void **set_transmission_reduction** (epiworld_double prob)
- void **set_recovery_enhancer** (epiworld_double prob)
- void **set_death_reduction** (epiworld_double prob)
- void set_name (std::string name)

  @]
- std::string **get_name** () const
- Person< TSeq > * **get_person** ()
- unsigned int **get_id** () const

Get and set the tool functions

Parameters

v	*The virus over which to operate*
fun	*the function to be used*

Returns

epiworld_double

- epiworld_double **get_susceptibility_reduction** (VirusPtr< TSeq > v)
- epiworld_double **get_transmission_reduction** (VirusPtr< TSeq > v)
- epiworld_double **get_recovery_enhancer** (VirusPtr< TSeq > v)
- epiworld_double **get_death_reduction** (VirusPtr< TSeq > v)
- void **set_susceptibility_reduction_fun** (ToolFun< TSeq > fun)
- void **set_transmission_reduction_fun** (ToolFun< TSeq > fun)
- void **set_recovery_enhancer_fun** (ToolFun< TSeq > fun)
- void **set_death_reduction_fun** (ToolFun< TSeq > fun)
- void **set_susceptibility_reduction** (epiworld_double *prob)
- void **set_transmission_reduction** (epiworld_double *prob)
- void **set_recovery_enhancer** (epiworld_double *prob)
- void **set_death_reduction** (epiworld_double *prob)
- void **set_susceptibility_reduction** (epiworld_double prob)
- void **set_transmission_reduction** (epiworld_double prob)
- void **set_recovery_enhancer** (epiworld_double prob)
- void **set_death_reduction** (epiworld_double prob)

Friends

- class **Agent**< **TSeq** >
- class **Model**< **TSeq** >
- class **PersonTools**< **TSeq** >
- class **Person**< **TSeq** >
- void **default_add_tool** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_tool** (Action< TSeq > &a, Model< TSeq > *m)

12.26.1 Detailed Description

template<typename TSeq = bool>
class epiworld::Tool< TSeq >

Tools for defending the agent against the virus.

Tools for defending the host against the virus.

Template Parameters

TSeq	Type of sequence

12.26.2 Member Function Documentation

12.26.2.1 get_susceptibility_reduction()

```
template<typename TSeq >
epiworld_double Tool< TSeq >::get_susceptibility_reduction (
            Virus< TSeq > * v )  [inline]
```

Get and set the tool functions.

Parameters

v	The virus over which to operate
fun	the function to be used

Returns

epiworld_double @[

The documentation for this class was generated from the following file:

- epiworld.hpp

12.27 Tool< TSeq > Class Template Reference

Tools for defending the agent against the virus.

`#include <tool-bones.hpp>`

Public Member Functions

- **Tool** (std::string name="unknown tool")
- void **set_sequence** (TSeq d)
- void **set_sequence_unique** (TSeq d)
- void **set_sequence** (std::shared_ptr< TSeq > d)
- std::shared_ptr< TSeq > **get_sequence** ()
- TSeq & **get_sequence_unique** ()
- void **set_name** (std::string name)
- std::string **get_name** () const
- Agent< TSeq > * **get_agent** ()
- int **get_id** () const
- void **set_id** (int id)
- void **set_date** (int d)
- int **get_date** () const
- void **set_status** (epiworld_fast_int init, epiworld_fast_int post)
- void **set_queue** (epiworld_fast_int init, epiworld_fast_int post)
- void **get_status** (epiworld_fast_int *init, epiworld_fast_int *post)
- void **get_queue** (epiworld_fast_int *init, epiworld_fast_int *post)

Get and set the tool functions

Parameters

v	*The virus over which to operate*
fun	*the function to be used*

Returns

epiworld_double

- epiworld_double **get_susceptibility_reduction** (VirusPtr< TSeq > v)
- epiworld_double **get_transmission_reduction** (VirusPtr< TSeq > v)
- epiworld_double **get_recovery_enhancer** (VirusPtr< TSeq > v)
- epiworld_double **get_death_reduction** (VirusPtr< TSeq > v)
- void **set_susceptibility_reduction_fun** (ToolFun< TSeq > fun)
- void **set_transmission_reduction_fun** (ToolFun< TSeq > fun)
- void **set_recovery_enhancer_fun** (ToolFun< TSeq > fun)
- void **set_death_reduction_fun** (ToolFun< TSeq > fun)
- void **set_susceptibility_reduction** (epiworld_double *prob)
- void **set_transmission_reduction** (epiworld_double *prob)
- void **set_recovery_enhancer** (epiworld_double *prob)
- void **set_death_reduction** (epiworld_double *prob)
- void **set_susceptibility_reduction** (epiworld_double prob)
- void **set_transmission_reduction** (epiworld_double prob)
- void **set_recovery_enhancer** (epiworld_double prob)
- void **set_death_reduction** (epiworld_double prob)

Friends

- class **Agent**< **TSeq** >
- class **Model**< **TSeq** >
- void **default_add_tool** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_tool** (Action< TSeq > &a, Model< TSeq > *m)

12.27.1 Detailed Description

template<**typename TSeq = int**>
class Tool< **TSeq** >

Tools for defending the agent against the virus.

Template Parameters

TSeq	Type of sequence

The documentation for this class was generated from the following files:

- include/epiworld/agent-bones.hpp
- include/epiworld/tool-bones.hpp
- include/epiworld/tool-meat.hpp

12.28 epiworld::Tools< TSeq > Class Template Reference

Set of tools (useful for building iterators)

```
#include <epiworld.hpp>
```

Public Member Functions

- **Tools** (Agent< TSeq > &p)
- std::vector< ToolPtr< TSeq > >::iterator **begin** ()
- std::vector< ToolPtr< TSeq > >::iterator **end** ()
- ToolPtr< TSeq > & **operator()** (size_t i)
- ToolPtr< TSeq > & **operator[ ]** (size_t i)
- size_t **size** () const noexcept

Friends

- class **Tool**< **TSeq** >
- class **Agent**< **TSeq** >

12.28.1 Detailed Description

template<**typename TSeq**>
class epiworld::Tools< **TSeq** >

Set of tools (useful for building iterators)

Template Parameters

TSeq	

The documentation for this class was generated from the following file:

- epiworld.hpp

12.29 Tools< TSeq > Class Template Reference

Set of tools (useful for building iterators)

`#include <tools-bones.hpp>`

Public Member Functions

- **Tools** (Agent< TSeq > &p)
- std::vector< ToolPtr< TSeq > >::iterator **begin** ()
- std::vector< ToolPtr< TSeq > >::iterator **end** ()
- ToolPtr< TSeq > & **operator()** (size_t i)
- ToolPtr< TSeq > & **operator[ ]** (size_t i)
- size_t **size** () const noexcept

Friends

- class **Tool**< **TSeq** >
- class **Agent**< **TSeq** >

12.29.1 Detailed Description

template<typename TSeq>
class Tools< TSeq >

Set of tools (useful for building iterators)

Template Parameters

TSeq	

The documentation for this class was generated from the following files:

- include/epiworld/agent-bones.hpp
- include/epiworld/tools-bones.hpp

12.30 epiworld::Tools_const< TSeq > Class Template Reference

Set of Tools (const) (useful for iterators)

`#include <epiworld.hpp>`

Public Member Functions

- **Tools_const** (const Agent< TSeq > &p)
- std::vector< ToolPtr< TSeq > >::const_iterator **begin** ()
- std::vector< ToolPtr< TSeq > >::const_iterator **end** ()
- const ToolPtr< TSeq > & **operator()** (size_t i)
- const ToolPtr< TSeq > & **operator[]** (size_t i)
- size_t **size** () const noexcept

Friends

- class **Tool< TSeq >**
- class **Agent< TSeq >**

12.30.1 Detailed Description

template<typename TSeq>
class epiworld::Tools_const< TSeq >

Set of Tools (const) (useful for iterators)

Template Parameters

TSeq	

The documentation for this class was generated from the following file:

- epiworld.hpp

12.31 Tools_const< TSeq > Class Template Reference

Set of Tools (const) (useful for iterators)

`#include <tools-bones.hpp>`

Public Member Functions

- **Tools_const** (const Agent< TSeq > &p)
- std::vector< ToolPtr< TSeq > >::const_iterator **begin** ()
- std::vector< ToolPtr< TSeq > >::const_iterator **end** ()
- const ToolPtr< TSeq > & **operator()** (size_t i)
- const ToolPtr< TSeq > & **operator[]** (size_t i)
- size_t **size** () const noexcept

Friends

- class **Tool< TSeq >**
- class **Agent< TSeq >**

12.31.1 Detailed Description

template<typename TSeq>
class Tools_const< TSeq >

Set of Tools (const) (useful for iterators)

Template Parameters

TSeq	

The documentation for this class was generated from the following files:

- include/epiworld/agent-bones.hpp
- include/epiworld/tools-bones.hpp

12.32 epiworld::UserData< TSeq > Class Template Reference

Personalized data by the user.

`#include <epiworld.hpp>`

Public Member Functions

- **UserData** (Model< TSeq > &m)
- UserData (std::vector< std::string > names)

  Construct a new User Data object.
- std::vector< std::string > & **get_names** ()
- std::vector< int > & **get_dates** ()
- std::vector< epiworld_double > & **get_data** ()
- void **get_all** (std::vector< std::string > *names=nullptr, std::vector< int > *date=nullptr, std::vector< epiworld_double > *data=nullptr)
- unsigned int **nrow** () const
- unsigned int **ncol** () const
- void **write** (std::string fn)
- void **print** () const
- **UserData** (std::vector< std::string > names)
- void **add** (std::vector< epiworld_double > x)
- void **add** (unsigned int j, epiworld_double x)
- epiworld_double & **operator()** (unsigned int i, unsigned int j)
- epiworld_double & **operator()** (unsigned int i, std::string name)
- std::vector< std::string > & **get_names** ()
- std::vector< int > & **get_dates** ()
- std::vector< epiworld_double > & **get_data** ()
- void **get_all** (std::vector< std::string > *names=nullptr, std::vector< int > *date=nullptr, std::vector< epiworld_double > *data=nullptr)
- unsigned int **nrow** () const
- unsigned int **ncol** () const
- void **write** (std::string fn)
- void **print** () const

Append data

Parameters

x	*A vector of length* `ncol()` *(if vector), otherwise a* `epiworld_double`.
j	*Index of the data point, from 0 to* `ncol() - 1`.

- void **add** (std::vector< epiworld_double > x)
- void **add** (unsigned int j, epiworld_double x)

Access data

Parameters

i	*Row (0 through ndays - 1.)*
j	*Column (0 through* `ncols()`*).*

Returns

epiworld_double&

- epiworld_double & **operator()** (unsigned int i, unsigned int j)
- epiworld_double & **operator()** (unsigned int i, std::string name)

Friends

- class **Model**< **TSeq** >
- class **DataBase**< **TSeq** >

12.32.1 Detailed Description

template<**typename TSeq**>
class epiworld::UserData< **TSeq** >

Personalized data by the user.

Template Parameters

TSeq	

12.32.2 Constructor & Destructor Documentation

12.32.2.1 UserData()

```
template<typename TSeq >
UserData< TSeq >::UserData (
            std::vector< std::string > names )  [inline]
```

Construct a new User Data object.

Parameters

names	A vector of names. The length of the vector sets the number of columns to record.

The documentation for this class was generated from the following file:

- epiworld.hpp

12.33 UserData< TSeq > Class Template Reference

Personalized data by the user.

```
#include <userdata-bones.hpp>
```

Public Member Functions

- **UserData** (Model< TSeq > &m)
- UserData (std::vector< std::string > names)

  Construct a new User Data object.
- std::vector< std::string > & **get_names** ()
- std::vector< int > & **get_dates** ()
- std::vector< epiworld_double > & **get_data** ()
- void **get_all** (std::vector< std::string > *names=nullptr, std::vector< int > *date=nullptr, std::vector< epiworld_double > *data=nullptr)
- unsigned int **nrow** () const
- unsigned int **ncol** () const
- void **write** (std::string fn)
- void **print** () const

Append data

Parameters

x	*A vector of length* `ncol()` *(if vector), otherwise a* `epiworld_double`.
j	*Index of the data point, from 0 to* `ncol() - 1`.

- void **add** (std::vector< epiworld_double > x)
- void **add** (unsigned int j, epiworld_double x)

Access data

Parameters

i	*Row (0 through ndays - 1.)*
j	*Column (0 through* `ncols()`*).*

Returns

epiworld_double&

- epiworld_double & **operator()** (unsigned int i, unsigned int j)
- epiworld_double & **operator()** (unsigned int i, std::string name)

Friends

- class **Model**< **TSeq** >
- class **DataBase**< **TSeq** >

12.33.1 Detailed Description

template<**typename TSeq**>
class UserData< **TSeq** >

Personalized data by the user.

Template Parameters

TSeq	

12.33.2 Constructor & Destructor Documentation

12.33.2.1 UserData()

```
template<typename TSeq >
UserData< TSeq >::UserData (
            std::vector< std::string > names )  [inline]
```

Construct a new User Data object.

Parameters

names	A vector of names. The length of the vector sets the number of columns to record.

The documentation for this class was generated from the following files:

- include/epiworld/database-bones.hpp
- include/epiworld/userdata-bones.hpp
- include/epiworld/userdata-meat.hpp

12.34 epiworld::vecHasher< T > Struct Template Reference

Vector hasher.

`#include <epiworld.hpp>`

Public Member Functions

- std::size_t **operator()** (std::vector< T > const &dat) const noexcept
- std::size_t **operator()** (std::vector< T > const &dat) const noexcept

12.34.1 Detailed Description

template<typename T>
struct epiworld::vecHasher< T >

Vector hasher.

Template Parameters

T	

The documentation for this struct was generated from the following file:

- epiworld.hpp

12.35 vecHasher< T > Struct Template Reference

Vector hasher.

`#include <misc.hpp>`

Public Member Functions

- std::size_t **operator()** (std::vector< T > const &dat) const noexcept

12.35.1 Detailed Description

template<typename T>
struct vecHasher< T >

Vector hasher.

Template Parameters

T	

The documentation for this struct was generated from the following file:

- include/epiworld/misc.hpp

12.36 epiworld::Virus< TSeq > Class Template Reference

Virus.

`#include <epiworld.hpp>`

Public Member Functions

- **Virus** (std::string name="unknown virus")
- void **mutate** ()
- void **set_mutation** (MutFun< TSeq > fun)
- const TSeq * **get_sequence** ()
- void **set_sequence** (TSeq sequence)
- Agent< TSeq > * **get_agent** ()
- void **set_agent** (Agent< TSeq > *p, epiworld_fast_uint idx)
- Model< TSeq > * **get_model** ()
- void **set_date** (int d)
- int **get_date** () const
- void **set_id** (int idx)
- int **get_id** () const
- void **set_name** (std::string name)
- std::string **get_name** () const
- std::vector< epiworld_double > & **get_data** ()
- **Virus** (std::string name="unknown virus")
- void **mutate** ()
- void **set_mutation** (MutFun< TSeq > fun)
- const TSeq * **get_sequence** ()
- void **set_sequence** (TSeq sequence)
- Person< TSeq > * **get_host** ()
- Model< TSeq > * **get_model** ()
- void **set_date** (int d)
- int **get_date** () const
- void **set_id** (int idx)
- int **get_id** () const
- bool **is_active** () const
- void **deactivate** ()
- epiworld_double get_prob_infecting ()

  Get and set the tool functions.
- epiworld_double **get_prob_recovery** ()
- epiworld_double **get_prob_death** ()
- void **post_recovery** ()
- void **set_post_recovery** (PostRecoveryFun< TSeq > fun)
- void **set_post_immunity** (epiworld_double prob)
- void **set_post_immunity** (epiworld_double *prob)
- void **set_prob_infecting_fun** (VirusFun< TSeq > fun)
- void **set_prob_recovery_fun** (VirusFun< TSeq > fun)
- void **set_prob_death_fun** (VirusFun< TSeq > fun)
- void **set_prob_infecting** (epiworld_double *prob)
- void **set_prob_recovery** (epiworld_double *prob)
- void **set_prob_death** (epiworld_double *prob)
- void **set_prob_infecting** (epiworld_double prob)
- void **set_prob_recovery** (epiworld_double prob)
- void **set_prob_death** (epiworld_double prob)
- void set_name (std::string name)

  @]
- std::string **get_name** () const
- std::vector< epiworld_double > & **get_data** ()

Get and set the tool functions

Parameters

v	*The virus over which to operate*
fun	*the function to be used*

Returns

epiworld_double

- epiworld_double **get_prob_infecting** ()
- epiworld_double **get_prob_recovery** ()
- epiworld_double **get_prob_death** ()
- void **post_recovery** ()
- void **set_post_recovery** (PostRecoveryFun< TSeq > fun)
- void **set_post_immunity** (epiworld_double prob)
- void **set_post_immunity** (epiworld_double *prob)
- void **set_prob_infecting_fun** (VirusFun< TSeq > fun)
- void **set_prob_recovery_fun** (VirusFun< TSeq > fun)
- void **set_prob_death_fun** (VirusFun< TSeq > fun)
- void **set_prob_infecting** (epiworld_double *prob)
- void **set_prob_recovery** (epiworld_double *prob)
- void **set_prob_death** (epiworld_double *prob)
- void **set_prob_infecting** (epiworld_double prob)
- void **set_prob_recovery** (epiworld_double prob)
- void **set_prob_death** (epiworld_double prob)

Get and set the status and queue

After applied, viruses can change the status and affect the queue of agents. These function sets the default values, which are retrieved when adding or removing a virus does not specify a change in status or in queue.

Parameters

init	*After the virus/tool is added to the agent.*
end	*After the virus/tool is removed.*
removed	*After the agent (Agent) is removed.*

- void **set_status** (epiworld_fast_int init, epiworld_fast_int end, epiworld_fast_int removed=-99)
- void **set_queue** (epiworld_fast_int init, epiworld_fast_int end, epiworld_fast_int removed=-99)
- void **get_status** (epiworld_fast_int *init, epiworld_fast_int *end, epiworld_fast_int *removed=-99)
- void **get_queue** (epiworld_fast_int *init, epiworld_fast_int *end, epiworld_fast_int *removed=-99)

Friends

- class **Agent**< **TSeq** >
- class **Model**< **TSeq** >
- class **DataBase**< **TSeq** >
- class **Person**< **TSeq** >
- class **PersonViruses**< **TSeq** >
- void **default_add_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_virus** (Action< TSeq > &a, Model< TSeq > *m)

12.36.1 Detailed Description

template<typename TSeq = bool>
class epiworld::Virus< TSeq >

Virus.

Template Parameters

TSeq	

Raw transmisibility of a virus should be a function of its genetic sequence. Nonetheless, transmisibility can be reduced as a result of having one or more tools to fight the virus. Because of this, transmisibility should be a function of the agent.

Template Parameters

TSeq	

Raw transmisibility of a virus should be a function of its genetic sequence. Nonetheless, transmisibility can be reduced as a result of having one or more tools to fight the virus. Because of this, transmisibility should be a function of the host.

12.36.2 Member Function Documentation

12.36.2.1 get_prob_infecting()

```
template<typename TSeq = bool>
epiworld_double epiworld::Virus< TSeq >::get_prob_infecting ( )
```

Get and set the tool functions.

Parameters

v	The virus over which to operate
fun	the function to be used

Returns

epiworld_double @[

The documentation for this class was generated from the following file:

- epiworld.hpp

12.37 Virus< TSeq > Class Template Reference

Virus.

```
#include <virus-bones.hpp>
```

Public Member Functions

- **Virus** (std::string name="unknown virus")
- void **mutate** ()
- void **set_mutation** (MutFun< TSeq > fun)
- const TSeq * **get_sequence** ()
- void **set_sequence** (TSeq sequence)
- Agent< TSeq > * **get_agent** ()
- void **set_agent** (Agent< TSeq > *p, epiworld_fast_uint idx)
- Model< TSeq > * **get_model** ()
- void **set_date** (int d)
- int **get_date** () const
- void **set_id** (int idx)
- int **get_id** () const
- void **set_name** (std::string name)
- std::string **get_name** () const
- std::vector< epiworld_double > & **get_data** ()

Get and set the tool functions

Parameters

v	*The virus over which to operate*
fun	*the function to be used*

Returns

epiworld_double

- epiworld_double **get_prob_infecting** ()
- epiworld_double **get_prob_recovery** ()
- epiworld_double **get_prob_death** ()
- void **post_recovery** ()
- void **set_post_recovery** (PostRecoveryFun< TSeq > fun)
- void **set_post_immunity** (epiworld_double prob)
- void **set_post_immunity** (epiworld_double *prob)
- void **set_prob_infecting_fun** (VirusFun< TSeq > fun)
- void **set_prob_recovery_fun** (VirusFun< TSeq > fun)
- void **set_prob_death_fun** (VirusFun< TSeq > fun)
- void **set_prob_infecting** (epiworld_double *prob)
- void **set_prob_recovery** (epiworld_double *prob)
- void **set_prob_death** (epiworld_double *prob)
- void **set_prob_infecting** (epiworld_double prob)
- void **set_prob_recovery** (epiworld_double prob)
- void **set_prob_death** (epiworld_double prob)

Get and set the status and queue

After applied, viruses can change the status and affect the queue of agents. These function sets the default values, which are retrieved when adding or removing a virus does not specify a change in status or in queue.

Parameters

init	*After the virus/tool is added to the agent.*
end	*After the virus/tool is removed.*
removed	*After the agent (Agent) is removed.*

- void **set_status** (epiworld_fast_int init, epiworld_fast_int end, epiworld_fast_int removed=-99)
- void **set_queue** (epiworld_fast_int init, epiworld_fast_int end, epiworld_fast_int removed=-99)
- void **get_status** (epiworld_fast_int *init, epiworld_fast_int *end, epiworld_fast_int *removed=-99)
- void **get_queue** (epiworld_fast_int *init, epiworld_fast_int *end, epiworld_fast_int *removed=-99)

Friends

- class **Agent**< **TSeq** >
- class **Model**< **TSeq** >
- class **DataBase**< **TSeq** >
- void **default_add_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_virus** (Action< TSeq > &a, Model< TSeq > *m)

12.37.1 Detailed Description

template<typename TSeq = int>
class Virus< TSeq >

Virus.

Template Parameters

TSeq	

Raw transmisibility of a virus should be a function of its genetic sequence. Nonetheless, transmisibility can be reduced as a result of having one or more tools to fight the virus. Because of this, transmisibility should be a function of the agent.

The documentation for this class was generated from the following files:

- include/epiworld/agent-bones.hpp
- include/epiworld/virus-bones.hpp
- include/epiworld/virus-meat.hpp

12.38 epiworld::Viruses< TSeq > Class Template Reference

Set of viruses (useful for building iterators)

`#include <epiworld.hpp>`

Public Member Functions

- **Viruses** (Agent< TSeq > &p)
- std::vector< VIRUSPTR >::iterator **begin** ()
- std::vector< VIRUSPTR >::iterator **end** ()
- VIRUSPTR & **operator()** (size_t i)
- VIRUSPTR & **operator[ ]** (size_t i)
- size_t **size** () const noexcept

Friends

- class **Virus**< **TSeq** >
- class **Agent**< **TSeq** >

12.38.1 Detailed Description

template<**typename TSeq**>
class epiworld::Viruses< **TSeq** >

Set of viruses (useful for building iterators)

Template Parameters

TSeq	

The documentation for this class was generated from the following file:

- epiworld.hpp

12.39 Viruses< TSeq > Class Template Reference

Set of viruses (useful for building iterators)

`#include <viruses-bones.hpp>`

Public Member Functions

- **Viruses** (Agent< TSeq > &p)
- std::vector< VIRUSPTR >::iterator **begin** ()
- std::vector< VIRUSPTR >::iterator **end** ()
- VIRUSPTR & **operator()** (size_t i)
- VIRUSPTR & **operator[ ]** (size_t i)
- size_t **size** () const noexcept

Friends

- class **Virus**< **TSeq** >
- class **Agent**< **TSeq** >

12.39.1 Detailed Description

template<**typename TSeq**>
class Viruses< **TSeq** >

Set of viruses (useful for building iterators)

Template Parameters

TSeq	

The documentation for this class was generated from the following files:

- include/epiworld/agent-bones.hpp
- include/epiworld/viruses-bones.hpp

12.40 epiworld::Viruses_const< TSeq > Class Template Reference

Set of Viruses (const) (useful for iterators)

`#include <epiworld.hpp>`

Public Member Functions

- **Viruses_const** (const Agent< TSeq > &p)
- std::vector< VIRUSPTR >::const_iterator **begin** ()
- std::vector< VIRUSPTR >::const_iterator **end** ()
- const VIRUSPTR & **operator()** (size_t i)
- const VIRUSPTR & **operator[ ]** (size_t i)
- size_t **size** () const noexcept

Friends

- class **Virus**< **TSeq** >
- class **Agent**< **TSeq** >

12.40.1 Detailed Description

template<typename TSeq>
class epiworld::Viruses_const< TSeq >

Set of Viruses (const) (useful for iterators)

Template Parameters

TSeq	

The documentation for this class was generated from the following file:

- epiworld.hpp

12.41 Viruses_const< TSeq > Class Template Reference

Set of Viruses (const) (useful for iterators)

`#include <viruses-bones.hpp>`

Public Member Functions

- **Viruses_const** (const Agent< TSeq > &p)
- std::vector< VIRUSPTR >::const_iterator **begin** ()
- std::vector< VIRUSPTR >::const_iterator **end** ()
- const VIRUSPTR & **operator()** (size_t i)
- const VIRUSPTR & **operator[ ]** (size_t i)
- size_t **size** () const noexcept

Friends

- class **Virus**< **TSeq** >
- class **Agent**< **TSeq** >

12.41.1 Detailed Description

template<typename TSeq>
class Viruses_const< TSeq >

Set of Viruses (const) (useful for iterators)

Template Parameters

TSeq	

The documentation for this class was generated from the following files:

- include/epiworld/agent-bones.hpp
- include/epiworld/viruses-bones.hpp

Index