

epiworld

0.0-1

Generated by Doxygen 1.9.1

1 Example: 00-hello-world	1
2 Benchmarking	3
3 Contributor Code of Conduct	5
4 epiworld c++ template library	7
4.1 Main features	7
4.2 Algorithm	7
4.3 Hello world (C++)	8
4.4 Surveillance simulation	8
4.4.1 Preliminary results	9
4.4.2 Cases detected	10
5 MIT License	11
6 model1	13
7 EPI Simulator	15
7.1 Disease dynamics	15
7.2 Network dynamics	15
7.3 Contagion dynamics	15
7.4 Time dynamics	15
7.5 Updating agent's status	16
7.5.1 Other parameters	16
8 Contributor Covenant Code of Conduct	17
8.1 Our Pledge	17
8.2 Our Standards	17
8.3 Enforcement Responsibilities	18
8.4 Scope	18
8.5 Enforcement	18
8.6 Enforcement Guidelines	18
8.6.1 1. Correction	18
8.6.2 2. Warning	18
8.6.3 3. Temporary Ban	19
8.6.4 4. Permanent Ban	19
8.7 Attribution	19
9 MIT License	21
10 epiworld 1.0	23
11 Namespace Index	25
11.1 Namespace List	25

12 Class Index	27
12.1 Class List	27
13 File Index	29
13.1 File List	29
14 Namespace Documentation	31
14.1 epiworld::sampler Namespace Reference	31
14.1.1 Detailed Description	31
14.1.2 Function Documentation	31
14.1.2.1 make_sample_virus_neighbors()	31
14.1.2.2 make_update_susceptible()	32
14.1.2.3 sample_virus_single()	32
14.2 sampler Namespace Reference	34
14.2.1 Detailed Description	34
14.2.2 Function Documentation	34
14.2.2.1 make_sample_virus_neighbors()	34
14.2.2.2 make_update_susceptible()	35
14.2.2.3 sample_virus_single()	35
15 Class Documentation	39
15.1 Action< TSeq > Struct Template Reference	39
15.1.1 Detailed Description	39
15.1.2 Constructor & Destructor Documentation	40
15.1.2.1 Action()	40
15.2 epiworld::Action< TSeq > Struct Template Reference	41
15.2.1 Detailed Description	41
15.2.2 Constructor & Destructor Documentation	41
15.2.2.1 Action()	41
15.3 AdjList Class Reference	42
15.3.1 Constructor & Destructor Documentation	43
15.3.1.1 AdjList()	43
15.3.2 Member Function Documentation	43
15.3.2.1 read_edgelist()	43
15.4 epiworld::AdjList Class Reference	44
15.4.1 Constructor & Destructor Documentation	44
15.4.1.1 AdjList() [1/2]	44
15.4.1.2 AdjList() [2/2]	45
15.4.2 Member Function Documentation	45
15.4.2.1 read_edgelist()	45
15.5 Agent< TSeq > Class Template Reference	46
15.5.1 Detailed Description	48
15.5.2 Member Function Documentation	48

15.5.2.1 operator()	48
15.6 epiworld::Agent< TSeq > Class Template Reference	49
15.6.1 Detailed Description	51
15.6.2 Member Function Documentation	51
15.6.2.1 operator()	51
15.7 AgentsSample< TSeq > Class Template Reference	52
15.7.1 Detailed Description	52
15.8 epiworld::AgentsSample< TSeq > Class Template Reference	53
15.8.1 Detailed Description	53
15.9 DataBase< TSeq > Class Template Reference	53
15.9.1 Detailed Description	55
15.9.2 Member Function Documentation	55
15.9.2.1 record_variant()	55
15.9.2.2 reproductive_number()	55
15.9.2.3 transition_probability()	56
15.10 epiworld::DataBase< TSeq > Class Template Reference	56
15.10.1 Detailed Description	58
15.10.2 Member Function Documentation	58
15.10.2.1 get_today_total()	59
15.10.2.2 record_variant() [1/2]	59
15.10.2.3 record_variant() [2/2]	59
15.10.2.4 reproductive_number()	60
15.10.2.5 transition_probability()	60
15.11 Entities< TSeq > Class Template Reference	60
15.11.1 Detailed Description	61
15.12 epiworld::Entities< TSeq > Class Template Reference	61
15.12.1 Detailed Description	61
15.13 Entities_const< TSeq > Class Template Reference	62
15.13.1 Detailed Description	62
15.14 epiworld::Entities_const< TSeq > Class Template Reference	63
15.14.1 Detailed Description	63
15.15 Entity< TSeq > Class Template Reference	63
15.16 epiworld::Entity< TSeq > Class Template Reference	64
15.17 epiworld::LFMCMC< TData > Class Template Reference	65
15.17.1 Detailed Description	66
15.18 LFMCMC< TData > Class Template Reference	66
15.18.1 Detailed Description	67
15.19 epiworld::Model< TSeq > Class Template Reference	67
15.19.1 Detailed Description	75
15.19.2 Member Function Documentation	76
15.19.2.1 add_global_action()	76
15.19.2.2 add_param()	76

15.19.2.3 add_status_susceptible()	77
15.19.2.4 init()	77
15.19.2.5 pop_from_adjlist()	78
15.19.2.6 reset() [1/2]	78
15.19.2.7 reset() [2/2]	79
15.19.2.8 reset_status_codes()	79
15.19.2.9 run_multiple()	79
15.19.2.10 set_agents_data()	80
15.19.2.11 set_backup()	80
15.19.2.12 set_rand_engine()	80
15.19.2.13 set_rewire_fun()	81
15.19.2.14 set_user_data()	81
15.19.2.15 write_data() [1/2]	81
15.19.2.16 write_data() [2/2]	82
15.19.2.17 write_edgelist()	82
15.20 Model< TSeq > Class Template Reference	83
15.20.1 Detailed Description	89
15.20.2 Member Function Documentation	89
15.20.2.1 add_global_action()	89
15.20.2.2 reset()	89
15.20.2.3 run_multiple()	90
15.20.2.4 set_agents_data()	90
15.20.2.5 write_data()	90
15.21 Network< Nettype, Nodetype, Edgetype > Class Template Reference	91
15.22 epiworld::Person< TSeq > Class Template Reference	92
15.23 epiworld::PersonTools< TSeq > Class Template Reference	92
15.23.1 Detailed Description	93
15.24 PersonTools< TSeq > Class Template Reference	93
15.25 epiworld::PersonViruses< TSeq > Class Template Reference	94
15.25.1 Detailed Description	94
15.26 epiworld::Progress Class Reference	94
15.26.1 Detailed Description	95
15.27 Progress Class Reference	95
15.27.1 Detailed Description	95
15.28 epiworld::Queue< TSeq > Class Template Reference	95
15.28.1 Detailed Description	96
15.29 Queue< TSeq > Class Template Reference	96
15.29.1 Detailed Description	96
15.30 epiworld::QueueValues Class Reference	97
15.31 QueueValues Class Reference	97
15.32 RandGraph Class Reference	97
15.33 epiworld::SAMPLETYPE Class Reference	98

15.34 SAMPLETYPE Class Reference	98
15.35 epiworld::Tool< TSeq > Class Template Reference	98
15.35.1 Detailed Description	100
15.35.2 Member Function Documentation	100
15.35.2.1 get_susceptibility_reduction()	100
15.36 Tool< TSeq > Class Template Reference	101
15.36.1 Detailed Description	102
15.37 epiworld::Tools< TSeq > Class Template Reference	102
15.37.1 Detailed Description	102
15.38 Tools< TSeq > Class Template Reference	103
15.38.1 Detailed Description	103
15.39 epiworld::Tools_const< TSeq > Class Template Reference	104
15.39.1 Detailed Description	104
15.40 Tools_const< TSeq > Class Template Reference	104
15.40.1 Detailed Description	105
15.41 epiworld::UserData< TSeq > Class Template Reference	105
15.41.1 Detailed Description	106
15.41.2 Constructor & Destructor Documentation	107
15.41.2.1 UserData()	107
15.42 UserData< TSeq > Class Template Reference	107
15.42.1 Detailed Description	108
15.42.2 Constructor & Destructor Documentation	108
15.42.2.1 UserData()	108
15.43 epiworld::vecHasher< T > Struct Template Reference	109
15.43.1 Detailed Description	109
15.44 vecHasher< T > Struct Template Reference	109
15.44.1 Detailed Description	109
15.45 epiworld::Virus< TSeq > Class Template Reference	110
15.45.1 Detailed Description	112
15.45.2 Member Function Documentation	112
15.45.2.1 get_prob_infecting()	112
15.46 Virus< TSeq > Class Template Reference	113
15.46.1 Detailed Description	114
15.47 epiworld::Viruses< TSeq > Class Template Reference	115
15.47.1 Detailed Description	115
15.48 Viruses< TSeq > Class Template Reference	116
15.48.1 Detailed Description	116
15.49 epiworld::Viruses_const< TSeq > Class Template Reference	116
15.49.1 Detailed Description	117
15.50 Viruses_const< TSeq > Class Template Reference	117
15.50.1 Detailed Description	117

16 File Documentation	119
16.1 include/epiworld/agent-meat-status.hpp File Reference	119
16.1.1 Detailed Description	120
Index	121

Chapter 1

Example: 00-hello-world

Output from the program:

Running the model...

```
||||| done.
[epiworld-debug] DEBUGGING ON (compiled with EPI_DEBUG defined)

SIMULATION STUDY
Population size      : 10000
Number of entitites  : 0
Days (duration)     : 100 (of 100)
Number of variants   : 1
Last run elapsed t   : 40.00ms
Rewiring             : off
Virus(es):
- covid 19 (baseline prevalence: 50 seeds)
Tool(s):
- vaccine (baseline prevalence: 50.00%)
- Immunity (covid 19) (originated in the model...)
Model parameters:
(none)
Distribution of the population at time 100:
- (0) Susceptible : 9950 -> 70
- (1) Exposed     : 50 -> 70
- (2) Recovered   : 0 -> 9271
- (3) Removed     : 0 -> 589
Transition Probabilities:
- Susceptible 0.95 0.05 0.00 0.00
- Exposed      0.00 0.85 0.14 0.01
- Recovered    0.00 0.00 1.00 0.00
- Removed      0.00 0.00 0.00 1.00
```


Chapter 2

Benchmarking

Here we keep a list of scenarios where we compare epiworld with other ABM simulation engines. Although the comparison is made at the speed level, we also list features of capabilities and main differences between the engines.

Chapter 3

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 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 $\sim \text{Gamma}(3, 1)$
- Duration of the disease $\sim \text{Gamma}(12, 1)$
- 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
## - Surveillance prob. : 1.0e-04
## - Vax efficacy       : 0.9000
## - Vax redux transmission : 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
## - Surveillance prob. : 0.0020
## - Vax efficacy       : 0.9000
## - Vax redux transmission : 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_asymp),
    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_asymp),
    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

Copyright (c) 2021 George G. Vega Yon

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

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 efficacy 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

```  $P(i \text{ gets the disease from } j \mid \text{their states}) = C(k) * (1 - E(i, k)) * (1 - E(j, k))$  ```

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:  

$$\Pr(h \rightarrow s | i, t, k, u) \sim \sum(g \in \{u, v\}) (N(i, t-1, s, k | g) + \sum(j \neq i) F(i, j) * N(j, t-1, s, k | g)) * C(k) / (N(i) + \sum(j \neq i) N(j))$$
  - b. Vaccinated individuals can become sick of variant  $k$  with probability:  $\Pr(v \rightarrow s | i, t, k, v) \sim \Pr(h \rightarrow s | i, t, k) * (1 - E(v, k))$ .
  - b. Recovered individuals can become sick of variant  $k$  with probability:  $\Pr(v \rightarrow s | i, t, k, r) \sim \Pr(h \rightarrow s | i, t, k) * (1 - E(r, k))$ .
  - 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  $P(u \rightarrow v) \sim V(A(i, t), B(i))$ .
  - 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:

$$\begin{aligned} P(\text{Variant } k | \text{at most 1}) &= P(\text{at most 1} | \text{Variant } k) * P(\text{Variant } k) / P(\text{at most 1}) \\ &= P(\text{only variant } k) / P(\text{variant } k) * P(\text{Variant } k) / P(\text{at most 1}) \\ &= P(\text{only variant } k) / P(\text{at most 1}) \end{aligned}$$

Where

$$\begin{aligned} P(\text{only variant } k) &= P(k) * \text{Prod}(m \neq v) (1 - P(m)) \\ P(\text{at most 1}) &= P(\text{None}) + \text{Sum}(v \text{ in variants}) P(v) * \text{Prod}(m \neq v) (1 - P(m)) \\ P(\text{None}) &= \text{Prod}(v \text{ in variants}) (1 - P(v)) \end{aligned}$$

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](#) changes 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](mailto: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](https://www.contributor-covenant.org/version/2/0/code_of_conduct.html), version 2.0, available at [https://www.contributor-covenant.org/version/2/0/code\\_of\\_conduct.html](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

Copyright (c) 2021 George G. Vega Yon

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.



## Chapter 10

# epiworld 1.0

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





# Chapter 11

## Namespace Index

### 11.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

<a href="#">epiworld::sampler</a>	Functions for sampling viruses . . . . .	<a href="#">31</a>
<a href="#">sampler</a>	Functions for sampling viruses . . . . .	<a href="#">34</a>



## Chapter 12

# Class Index

### 12.1 Class List

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

<a href="#">Action&lt; TSeq &gt;</a>	
Action data for update an agent	39
<a href="#">epiworld::Action&lt; TSeq &gt;</a>	
Action data for update an agent	41
<a href="#">AdjList</a>	42
<a href="#">epiworld::AdjList</a>	44
<a href="#">Agent&lt; TSeq &gt;</a>	
Agent (agents)	46
<a href="#">epiworld::Agent&lt; TSeq &gt;</a>	
Agent (agents)	49
<a href="#">AgentsSample&lt; TSeq &gt;</a>	
Sample of agents	52
<a href="#">epiworld::AgentsSample&lt; TSeq &gt;</a>	
Sample of agents	53
<a href="#">DataBase&lt; TSeq &gt;</a>	
Statistical data about the process	53
<a href="#">epiworld::DataBase&lt; TSeq &gt;</a>	
Statistical data about the process	56
<a href="#">Entities&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (useful for building iterators)	60
<a href="#">epiworld::Entities&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (useful for building iterators)	61
<a href="#">Entities_const&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (const) (useful for iterators)	62
<a href="#">epiworld::Entities_const&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (const) (useful for iterators)	63
<a href="#">Entity&lt; TSeq &gt;</a>	63
<a href="#">epiworld::Entity&lt; TSeq &gt;</a>	64
<a href="#">epiworld::LFMCMC&lt; TData &gt;</a>	
Likelihood-Free Markov Chain Monte Carlo	65
<a href="#">LFMCMC&lt; TData &gt;</a>	
Likelihood-Free Markov Chain Monte Carlo	66
<a href="#">epiworld::Model&lt; TSeq &gt;</a>	
Core class of epiworld	67
<a href="#">Model&lt; TSeq &gt;</a>	
Core class of epiworld	83

<a href="#">Network&lt; Nettype, Nodetype, Edgetype &gt;</a>	91
<a href="#">epiworld::Person&lt; TSeq &gt;</a>	92
<a href="#">epiworld::PersonTools&lt; TSeq &gt;</a>	
List of tools available for the individual to	92
<a href="#">PersonTools&lt; TSeq &gt;</a>	93
<a href="#">epiworld::PersonViruses&lt; TSeq &gt;</a>	
Set of viruses in host	94
<a href="#">epiworld::Progress</a>	
A simple progress bar	94
<a href="#">Progress</a>	
A simple progress bar	95
<a href="#">epiworld::Queue&lt; TSeq &gt;</a>	
Controls which agents are verified at each step	95
<a href="#">Queue&lt; TSeq &gt;</a>	
Controls which agents are verified at each step	96
<a href="#">epiworld::QueueValues</a>	97
<a href="#">QueueValues</a>	97
<a href="#">RandGraph</a>	97
<a href="#">epiworld::SAMPLETYPE</a>	98
<a href="#">SAMPLETYPE</a>	98
<a href="#">epiworld::Tool&lt; TSeq &gt;</a>	
Tools for defending the agent against the virus	98
<a href="#">Tool&lt; TSeq &gt;</a>	
Tools for defending the agent against the virus	101
<a href="#">epiworld::Tools&lt; TSeq &gt;</a>	
Set of tools (useful for building iterators)	102
<a href="#">Tools&lt; TSeq &gt;</a>	
Set of tools (useful for building iterators)	103
<a href="#">epiworld::Tools_const&lt; TSeq &gt;</a>	
Set of <a href="#">Tools</a> (const) (useful for iterators)	104
<a href="#">Tools_const&lt; TSeq &gt;</a>	
Set of <a href="#">Tools</a> (const) (useful for iterators)	104
<a href="#">epiworld::UserData&lt; TSeq &gt;</a>	
Personalized data by the user	105
<a href="#">UserData&lt; TSeq &gt;</a>	
Personalized data by the user	107
<a href="#">epiworld::vecHasher&lt; T &gt;</a>	
Vector hasher	109
<a href="#">vecHasher&lt; T &gt;</a>	
Vector hasher	109
<a href="#">epiworld::Virus&lt; TSeq &gt;</a>	
Virus	110
<a href="#">Virus&lt; TSeq &gt;</a>	
Virus	113
<a href="#">epiworld::Viruses&lt; TSeq &gt;</a>	
Set of viruses (useful for building iterators)	115
<a href="#">Viruses&lt; TSeq &gt;</a>	
Set of viruses (useful for building iterators)	116
<a href="#">epiworld::Viruses_const&lt; TSeq &gt;</a>	
Set of <a href="#">Viruses</a> (const) (useful for iterators)	116
<a href="#">Viruses_const&lt; TSeq &gt;</a>	
Set of <a href="#">Viruses</a> (const) (useful for iterators)	117

## Chapter 13

# File Index

### 13.1 File List

Here is a list of all documented files with brief descriptions:

<b>epiworld.hpp</b>	??
include/epiworld/ <b>adjlist-bones.hpp</b>	??
include/epiworld/ <b>adjlist-meat.hpp</b>	??
include/epiworld/ <b>agent-actions-meat.hpp</b>	??
include/epiworld/ <b>agent-bones.hpp</b>	??
include/epiworld/ <b>agent-meat-status.hpp</b>	
Sampling functions are getting big, so we keep them in a separate file	119
include/epiworld/ <b>agent-meat-virus-sampling.hpp</b>	??
include/epiworld/ <b>agent-meat.hpp</b>	??
include/epiworld/ <b>agentssample-bones.hpp</b>	??
include/epiworld/ <b>config.hpp</b>	??
include/epiworld/ <b>database-bones.hpp</b>	??
include/epiworld/ <b>database-meat.hpp</b>	??
include/epiworld/ <b>entities-bones.hpp</b>	??
include/epiworld/ <b>entity-bones.hpp</b>	??
include/epiworld/ <b>entity-meat.hpp</b>	??
include/epiworld/ <b>epiworld-macros.hpp</b>	??
include/epiworld/ <b>epiworld.hpp</b>	??
include/epiworld/ <b>misc.hpp</b>	??
include/epiworld/ <b>model-bones.hpp</b>	??
include/epiworld/ <b>model-meat-print.hpp</b>	??
include/epiworld/ <b>model-meat.hpp</b>	??
include/epiworld/ <b>network-bones.hpp</b>	??
include/epiworld/ <b>progress.hpp</b>	??
include/epiworld/ <b>queue-bones.hpp</b>	??
include/epiworld/ <b>randgraph.hpp</b>	??
include/epiworld/ <b>random_graph.hpp</b>	??
include/epiworld/ <b>seq_processing.hpp</b>	??
include/epiworld/ <b>tool-bones.hpp</b>	??
include/epiworld/ <b>tool-meat.hpp</b>	??
include/epiworld/ <b>tools-bones.hpp</b>	??
include/epiworld/ <b>userdata-bones.hpp</b>	??
include/epiworld/ <b>userdata-meat.hpp</b>	??
include/epiworld/ <b>virus-bones.hpp</b>	??
include/epiworld/ <b>virus-meat.hpp</b>	??

include/epiworld/ <b>viruses-bones.hpp</b>	??
include/epiworld/math/ <b>lfmcmc.hpp</b>	??
include/epiworld/math/lfmcmc/ <b>lfmcmc-bones.hpp</b>	??
include/epiworld/math/lfmcmc/ <b>lfmcmc-meat-print.hpp</b>	??
include/epiworld/math/lfmcmc/ <b>lfmcmc-meat.hpp</b>	??
include/epiworld/models/ <b>models.hpp</b>	??
include/epiworld/models/ <b>seir.hpp</b>	??
include/epiworld/models/ <b>seirconnected.hpp</b>	??
include/epiworld/models/ <b>sir.hpp</b>	??
include/epiworld/models/ <b>sirconnected.hpp</b>	??
include/epiworld/models/ <b>sis.hpp</b>	??
include/epiworld/models/ <b>surveillance.hpp</b>	??
rpackage/inst/include/ <b>epiworld.hpp</b>	??
rpackage/inst/include/models/ <b>immune_system.hpp</b>	??
rpackage/inst/include/models/ <b>seirconnected.hpp</b>	??
rpackage/inst/include/models/ <b>sir.hpp</b>	??
rpackage/inst/include/models/ <b>sirconnected.hpp</b>	??
rpackage/inst/include/models/ <b>surveillance.hpp</b>	??
rpackage/src/ <b>epiworld-common.hpp</b>	??
tests/ <b>tests.hpp</b>	??

## Chapter 14

# Namespace Documentation

### 14.1 epiworld::sampler Namespace Reference

Functions for sampling viruses.

#### Functions

- `template<typename TSeq >`  
`std::function< void(Agent< TSeq > *, Model< TSeq > *)> make\_update\_susceptible (std::vector< epiworld_fast_uint > exclude={})`  
*Make a function to sample from neighbors.*
- `template<typename TSeq = int>`  
`std::function< Virus< TSeq > *(Agent< TSeq > *, Model< TSeq > *)> make\_sample\_virus\_neighbors (std::vector< epiworld_fast_uint > exclude={})`  
*Make a function to sample from neighbors.*
- `template<typename TSeq = int>`  
`Virus< TSeq > * sample\_virus\_single (Agent< TSeq > *p, Model< TSeq > *m)`  
*Sample from neighbors pool of viruses (at most one)*

#### 14.1.1 Detailed Description

Functions for sampling viruses.

#### 14.1.2 Function Documentation

##### 14.1.2.1 `make_sample_virus_neighbors()`

```
template<typename TSeq = int>
std::function<Virus<TSeq>*(Agent<TSeq>*,Model<TSeq>*)> epiworld::sampler::make_sample_virus_neighbors (
 std::vector< epiworld_fast_uint > exclude = {}) [inline]
```

Make a function to sample from neighbors.

This is akin to the function `default_update_susceptible`, with the difference that it will create a function that supports excluding states from the sampling frame. For example, individuals who have acquired a virus can be excluded if in incubation state.

## Template Parameters

<i>TSeq</i>	
-------------	--

## Parameters

<i>exclude</i>	unsigned vector of states that need to be excluded from the sampling
----------------	----------------------------------------------------------------------

## Returns

Virus<TSeq>\* of the selected virus. If none selected (or none available,) returns a nullptr;

**14.1.2.2 make\_update\_susceptible()**

```
template<typename TSeq >
std::function<void (Agent<TSeq>*, Model<TSeq>*)> epiworld::sampler::make_update_susceptible (
 std::vector< epiworld_fast_uint > exclude = {}) [inline]
```

Make a function to sample from neighbors.

This is akin to the function `default_update_susceptible`, with the difference that it will create a function that supports excluding states from the sampling frame. For example, individuals who have acquired a virus can be excluded if in incubation state.

## Template Parameters

<i>TSeq</i>	
-------------	--

## Parameters

<i>exclude</i>	unsigned vector of states that need to be excluded from the sampling
----------------	----------------------------------------------------------------------

## Returns

Virus<TSeq>\* of the selected virus. If none selected (or none available,) returns a nullptr;

**14.1.2.3 sample\_virus\_single()**

```
template<typename TSeq = int>
Virus<TSeq>* epiworld::sampler::sample_virus_single (
 Agent< TSeq > * p,
 Model< TSeq > * m) [inline]
```

Sample from neighbors pool of viruses (at most one)



This function samples at most one virus from the pool of viruses from its neighbors. If no virus is selected, the function returns a `nullptr`, otherwise it returns a pointer to the selected virus.

This can be used to build a new update function (EPI\_NEW\_UPDATEFUN.)

## Template Parameters

<i>TSeq</i>	
-------------	--

## Parameters

<i>p</i>	Pointer to person
<i>m</i>	Pointer to the model

## Returns

Virus<TSeq>\* of the selected virus. If none selected (or none available,) returns a nullptr;

## 14.2 sampler Namespace Reference

Functions for sampling viruses.

### Functions

- `template<typename TSeq > std::function< void(Agent< TSeq > *, Model< TSeq > *)> make\_update\_susceptible (std::vector< epiworld_fast_uint > exclude={})`  
*Make a function to sample from neighbors.*
- `template<typename TSeq = int> std::function< Virus< TSeq > *(Agent< TSeq > *, Model< TSeq > *)> make\_sample\_virus\_neighbors (std::vector< epiworld_fast_uint > exclude={})`  
*Make a function to sample from neighbors.*
- `template<typename TSeq = int> Virus< TSeq > * sample\_virus\_single (Agent< TSeq > *p, Model< TSeq > *m)`  
*Sample from neighbors pool of viruses (at most one)*

### 14.2.1 Detailed Description

Functions for sampling viruses.

### 14.2.2 Function Documentation

#### 14.2.2.1 `make_sample_virus_neighbors()`

```
template<typename TSeq = int>
std::function<Virus<TSeq>*(Agent<TSeq>*,Model<TSeq>*)> sampler::make_sample_virus_neighbors
(
 std::vector< epiworld_fast_uint > exclude = {}) [inline]
```

Make a function to sample from neighbors.

This is akin to the function `default_update_susceptible`, with the difference that it will create a function that supports excluding states from the sampling frame. For example, individuals who have acquired a virus can be excluded if in incubation state.

## Template Parameters

<i>TSeq</i>	
-------------	--

## Parameters

<i>exclude</i>	unsigned vector of states that need to be excluded from the sampling
----------------	----------------------------------------------------------------------

## Returns

Virus<TSeq>\* of the selected virus. If none selected (or none available,) returns a nullptr;

## 14.2.2.2 make\_update\_susceptible()

```
template<typename TSeq >
std::function<void (Agent<TSeq>*, Model<TSeq>*)> sampler::make_update_susceptible (
 std::vector< epiworld_fast_uint > exclude = {}) [inline]
```

Make a function to sample from neighbors.

This is akin to the function `default_update_susceptible`, with the difference that it will create a function that supports excluding states from the sampling frame. For example, individuals who have acquired a virus can be excluded if in incubation state.

## Template Parameters

<i>TSeq</i>	
-------------	--

## Parameters

<i>exclude</i>	unsigned vector of states that need to be excluded from the sampling
----------------	----------------------------------------------------------------------

## Returns

Virus<TSeq>\* of the selected virus. If none selected (or none available,) returns a nullptr;

## 14.2.2.3 sample\_virus\_single()

```
template<typename TSeq = int>
Virus<TSeq>* sampler::sample_virus_single (
 Agent< TSeq > * p,
 Model< TSeq > * m) [inline]
```

Sample from neighbors pool of viruses (at most one)

This function samples at most one virus from the pool of viruses from its neighbors. If no virus is selected, the function returns a `nullptr`, otherwise it returns a pointer to the selected virus.

This can be used to build a new update function (`EPI_NEW_UPDATEFUN.`)

## Template Parameters

<i>TSeq</i>	
-------------	--

## Parameters

<i>p</i>	Pointer to person
<i>m</i>	Pointer to the model

## Returns

Virus<TSeq>\* of the selected virus. If none selected (or none available,) returns a nullptr;



# Chapter 15

## Class Documentation

### 15.1 Action< TSeq > Struct Template Reference

Action data for update an agent.

```
#include <config.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

#### 15.1.1 Detailed Description

```
template<typename TSeq>
struct Action< TSeq >
```

Action data for update an agent.

## Template Parameters

<i>TSeq</i>	
-------------	--

## 15.1.2 Constructor &amp; Destructor Documentation

## 15.1.2.1 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

<i>agent_</i>	<a href="#">Agent</a> over who the action will happen
<i>virus_</i>	<a href="#">Virus</a> to add
<i>tool_</i>	<a href="#">Tool</a> to add
<i>virus_idx</i>	Index of virus to be removed (if needed)
<i>tool_idx</i>	Index of tool to be removed (if needed)
<i>new_status_↔</i>	Next status
<i>queue_</i>	Efect on the queue
<i>call_</i>	The action call (if needed)
<i>idx_agent_↔</i>	Location of agent in object.
<i>idx_object_↔</i>	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



## 15.2 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**

### 15.2.1 Detailed Description

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

[Action](#) data for update an agent.

Template Parameters

<a href="#">TSeq</a>	
----------------------	--

### 15.2.2 Constructor & Destructor Documentation

#### 15.2.2.1 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

<i>agent_</i>	<a href="#">Agent</a> over who the action will happen
<i>virus_</i>	<a href="#">Virus</a> to add
<i>tool_</i>	<a href="#">Tool</a> to add
<i>virus_idx</i>	Index of virus to be removed (if needed)
<i>tool_idx</i>	Index of tool to be removed (if needed)
<i>new_↔ status_</i>	Next status
<i>queue_</i>	Efect on the queue
<i>call_</i>	The action call (if needed)
<i>idx_agent↔ _</i>	Location of agent in object.
<i>idx_object↔ _</i>	Location of object in agent.

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

- epiworld.hpp

## 15.3 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.*

## 15.3.1 Constructor & Destructor Documentation

### 15.3.1.1 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

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

## 15.3.2 Member Function Documentation

### 15.3.2.1 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

<i>fn</i>	Path to the file
<i>skip</i>	Number of lines to skip (e.g., 1 if there's a header)
<i>directed</i>	true if the network is directed
<i>size</i>	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

## 15.4 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

### 15.4.1 Constructor & Destructor Documentation

#### 15.4.1.1 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

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

## 15.4.1.2 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

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

## 15.4.2 Member Function Documentation

## 15.4.2.1 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

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

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

- `epiworld.hpp`

## 15.5 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\_engine** ()
- [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
- [Entities](#)< TSeq > **get\_entities** ()
- const [Entities\\_const](#)< TSeq > **get\_entities** () const

#### Add/Remove Virus/Tool

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

*Parameters*

tool	<i>Tool to add</i>
virus	<i>Virus to add</i>
status_new	<i>Status after the change</i>
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	<i>A pointer to a virus.</i>
---	------------------------------

*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 >
- class **Entities**< TSeq >
- class **AgentsSample**< 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)

### 15.5.1 Detailed Description

```
template<typename TSeq>
class Agent< TSeq >
```

[Agent](#) (agents)

Template Parameters

<i>TSeq</i>	Sequence type (should match TSeq across the model)
-------------	----------------------------------------------------

### 15.5.2 Member Function Documentation

#### 15.5.2.1 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

<i>j</i>	
----------	--

## Returns

double&amp;

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

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

## 15.6 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\\_engine](#) ()
- [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
- [Entities](#)< TSeq > [get\\_entities](#) ()
- const [Entities\\_const](#)< TSeq > [get\\_entities](#) () const

### Add/Remove Virus/Tool

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

*Parameters*

tool	<i>Tool to add</i>
virus	<i>Virus to add</i>
status_new	<i>Status after the change</i>
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	<i>A pointer to a virus.</i>
---	------------------------------

*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 >
- class **Entities**< TSeq >
- class **AgentsSample**< 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)

### 15.6.1 Detailed Description

```
template<typename TSeq>
class epiworld::Agent< TSeq >
```

[Agent](#) (agents)

Template Parameters

<i>TSeq</i>	Sequence type (should match TSeq across the model)
-------------	----------------------------------------------------

### 15.6.2 Member Function Documentation

#### 15.6.2.1 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

<i>j</i>	
----------	--

## Returns

double&

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

- epiworld.hpp

## 15.7 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, bool truncate=false)
- **AgentsSample** ([Entity](#)< TSeq > &entity\_, size\_t n, bool truncate=false)
- **AgentsSample** ([Agent](#)< TSeq > &agent\_, size\_t n, bool truncate=false)
- 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

### 15.7.1 Detailed Description

```
template<typename TSeq>
class AgentsSample< TSeq >
```

Sample of agents.

This class allows sampling agents from Entity<TSeq> and [Model](#)<TSeq>.

#### Template Parameters

<i>TSeq</i>	
-------------	--

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

- include/epiworld/agentssample-bones.hpp

## 15.8 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, bool truncate=false)
- [AgentsSample](#) ([Entity](#)< TSeq > &entity\_, size\_t n, bool truncate=false)
- [AgentsSample](#) ([Agent](#)< TSeq > &agent\_, size\_t n, bool truncate=false)
- 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

### 15.8.1 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

<i>TSeq</i>	
-------------	--

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

- epiworld.hpp

## 15.9 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	<i>std::string, The status, e.g., 0, 1, 2, ...</i>
------	----------------------------------------------------

### 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)

### 15.9.1 Detailed Description

```
template<typename TSeq>
class DataBase< TSeq >
```

Statistical data about the process.

#### Template Parameters

<i>TSeq</i>	
-------------	--

### 15.9.2 Member Function Documentation

#### 15.9.2.1 record\_variant()

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

Registering a new variant.

#### Parameters

<i>v</i>	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.
----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### 15.9.2.2 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  $R_0$  (basic reproductive number) or  $R_t/R$  (the effective reproductive number) will depend on whether the virus is allowed to circulate naïvely or not, respectively.

#### Parameters

<i>fn</i>	File where to write out the reproductive number.
-----------	--------------------------------------------------

### 15.9.2.3 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

## 15.10 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	<i>std::string, The status, e.g., 0, 1, 2, ...</i>
------	----------------------------------------------------

**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)

**15.10.1 Detailed Description**

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

Statistical data about the process.

**Template Parameters**

<i>TSeq</i>	
-------------	--

**15.10.2 Member Function Documentation**

## 15.10.2.1 get\_today\_total()

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

Get recorded information from the model.

## Parameters

<i>what</i>	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 @[

## 15.10.2.2 record\_variant() [1/2]

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

Registering a new variant.

## Parameters

<i>v</i>	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.
----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## 15.10.2.3 record\_variant() [2/2]

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

Registering a new variant.

## Parameters

<i>v</i>	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.
----------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### 15.10.2.4 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  $R_0$  (basic reproductive number) or  $R_t/R$  (the effective reproductive number) will depend on whether the virus is allowed to circulate naïvely or not, respectively.

##### Parameters

<i>fn</i>	File where to write out the reproductive number.
-----------	--------------------------------------------------

#### 15.10.2.5 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

## 15.11 Entities< TSeq > Class Template Reference

Set of [Entities](#) (useful for building iterators)

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

### Public Member Functions

- **Entities** ([Agent](#)< TSeq > &p)
- `std::vector< Entity< TSeq > * >::iterator` **begin** ()
- `std::vector< Entity< TSeq > * >::iterator` **end** ()
- [Entity](#)< TSeq > \*& **operator**() (size\_t i)
- [Entity](#)< TSeq > \*& **operator**[] (size\_t i)
- size\_t **size** () const noexcept

## Friends

- class **Entity**< TSeq >
- class **Agent**< TSeq >

### 15.11.1 Detailed Description

```
template<typename TSeq>
class Entities< TSeq >
```

Set of [Entities](#) (useful for building iterators)

Template Parameters

<i>TSeq</i>	
-------------	--

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

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

## 15.12 epiworld::Entities< TSeq > Class Template Reference

Set of [Entities](#) (useful for building iterators)

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

## Public Member Functions

- **Entities** ([Agent](#)< TSeq > &p)
- std::vector< [Entity](#)< TSeq > \* >::iterator **begin** ()
- std::vector< [Entity](#)< TSeq > \* >::iterator **end** ()
- [Entity](#)< TSeq > \*& **operator**() (size\_t i)
- [Entity](#)< TSeq > \*& **operator[]** (size\_t i)
- size\_t **size** () const noexcept

## Friends

- class **Entity**< TSeq >
- class **Agent**< TSeq >

### 15.12.1 Detailed Description

```
template<typename TSeq>
class epiworld::Entities< TSeq >
```

Set of [Entities](#) (useful for building iterators)

## Template Parameters

<i>TSeq</i>	
-------------	--

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

- epiworld.hpp

## 15.13 Entities\_const< TSeq > Class Template Reference

Set of [Entities](#) (const) (useful for iterators)

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

### Public Member Functions

- **Entities\_const** (const [Agent](#)< TSeq > &p)
- std::vector< [Entity](#)< TSeq > \* >::const\_iterator **begin** ()
- std::vector< [Entity](#)< TSeq > \* >::const\_iterator **end** ()
- const [Entity](#)< TSeq > \*& **operator()** (size\_t i)
- const [Entity](#)< TSeq > \*& **operator[]** (size\_t i)
- size\_t **size** () const noexcept

### Friends

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

### 15.13.1 Detailed Description

```
template<typename TSeq>
class Entities_const< TSeq >
```

Set of [Entities](#) (const) (useful for iterators)

## Template Parameters

<i>TSeq</i>	
-------------	--

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

- include/epiworld/entities-bones.hpp

## 15.14 epiworld::Entities\_const< TSeq > Class Template Reference

Set of [Entities](#) (const) (useful for iterators)

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

### Public Member Functions

- **Entities\_const** (const [Agent](#)< TSeq > &p)
- std::vector< [Entity](#)< TSeq > \* >::const\_iterator **begin** ()
- std::vector< [Entity](#)< TSeq > \* >::const\_iterator **end** ()
- const [Entity](#)< TSeq > \*& **operator**() (size\_t i)
- const [Entity](#)< TSeq > \*& **operator**[] (size\_t i)
- size\_t **size** () const noexcept

### Friends

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

#### 15.14.1 Detailed Description

```
template<typename TSeq>
class epiworld::Entities_const< TSeq >
```

Set of [Entities](#) (const) (useful for iterators)

Template Parameters

<i>TSeq</i>	
-------------	--

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

- epiworld.hpp

## 15.15 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`
- `Agent< TSeq > * operator[] (size_t i)`
- `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/agent-bones.hpp`
- `include/epiworld/entity-bones.hpp`
- `include/epiworld/entity-meat.hpp`

## 15.16 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`
- `Agent< TSeq > * operator[] (size_t i)`
- `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

## 15.17 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\_if\_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\_engine** ()
- 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)

### 15.17.1 Detailed Description

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

Likelihood-Free Markov Chain Monte Carlo.

Template Parameters

<i>TData</i>	Type of data that is generated
--------------	--------------------------------

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

- epiworld.hpp

## 15.18 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_if_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

<code>eng</code>	
------------------	--

- `void set_rand_engine (std::mt19937 &eng)`
- `std::mt19937 * get_rand_engine ()`
- `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)`

### 15.18.1 Detailed Description

```
template<typename TData>
class LFMCMC< TData >
```

Likelihood-Free Markov Chain Monte Carlo.

#### Template Parameters

<code>TData</code>	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`

## 15.19 `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\_engine** ()
  - 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	<i>Random number generator</i>
s	<i>Seed</i>

- void **set\_rand\_engine** (std::mt19937 &eng)
- std::mt19937 \* **get\_rand\_engine** ()
- void **seed** (unsigned int s)
- void **set\_rand\_norm** (epiworld\_double mean, epiworld\_double sd)
- void **set\_rand\_unif** (epiworld\_double a, epiworld\_double b)
- void **set\_rand\_exp** (epiworld\_double lambda)
- void **set\_rand\_gamma** (epiworld\_double alpha, epiworld\_double beta)
- void **set\_rand\_lognormal** (epiworld\_double mean, epiworld\_double shape)
- epiworld\_double **runif** ()
- epiworld\_double **runif** (epiworld\_double a, epiworld\_double b)
- epiworld\_double **rnorm** ()
- epiworld\_double **rnorm** (epiworld\_double mean, epiworld\_double sd)
- epiworld\_double **rgamma** ()
- epiworld\_double **rgamma** (epiworld\_double alpha, epiworld\_double beta)
- epiworld\_double **rexp** ()
- epiworld\_double **rexp** (epiworld\_double lambda)
- epiworld\_double **rlognormal** ()
- epiworld\_double **rlognormal** (epiworld\_double mean, epiworld\_double shape)

**Add Virus/Tool to the model**

*This is done before the model has been initialized.*

*Parameters*

v	<i><a href="#">Virus</a> to be added</i>
t	<i><a href="#">Tool</a> to be added</i>
preval	<i>Initial prevalence (initial state.) It can be specified as a proportion (between zero and one,) or an integer indicating number of individuals.</i>

- void **add\_virus** ([Virus](#)< TSeq > v, epiworld\_double preval)
- void **add\_virus\_n** ([Virus](#)< TSeq > v, unsigned int preval)
- void **add\_virus\_fun** ([Virus](#)< TSeq > v, VirusToAgentFun< TSeq > fun)
- void **add\_tool** ([Tool](#)< TSeq > t, epiworld\_double preval)
- void **add\_tool\_n** ([Tool](#)< TSeq > t, unsigned int preval)
- void **add\_tool\_fun** ([Tool](#)< TSeq > t, ToolToAgentFun< TSeq > fun)
- 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	<i>std::string Filename of the edgelist file.</i>
skip	<i>int Number of lines to skip in fn.</i>
directed	<i>bool Whether the graph is directed or not.</i>
size	<i>Size of the network.</i>
al	<i><a href="#">AdjList</a> to read into the model.</i>

- 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 <i>run_multiple</i> , 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=make\_save\_run< TSeq >(), bool **reset**=true, bool verbose=true)

### Rewire the network preserving the degree sequence.

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

#### 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	<code>std::string</code> 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 **queueing\_on** ()  
Activates the queueing system (default.)
- void **queueing\_off** ()  
Deactivates the queueing system.
- bool **is\_queueing\_on** () const  
Query if the queueing 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 ()`.

In the case of the function `read_params`, users can pass a file listing parameters to be included in the model. Each line in the file should have the following structure:

```
[name of parameter 1]: [value in double]
[name of parameter 2]: [value in double]
...
```

The only condition for parameter names is that these do not include a colon.

### Parameters

<i>initial_val</i>	
<i>pname</i>	Name of the parameter to add or to fetch
<i>fn</i>	Path to the file containing parameters

### 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)`
- `void read_params (std::string fn)`
- `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)`

### 15.19.1 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

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

## 15.19.2 Member Function Documentation

### 15.19.2.1 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

<i>fun</i>	A function to be called on the prescribed dates
<i>date</i>	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.

### 15.19.2.2 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

<i>initial_val</i>	
<i>pname</i>	Name of the parameter to add or to fetch

**Returns**

The current value of the parameter in the model. @

**15.19.2.3 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**

<i>s</i>	unsigned int Code of the status
<i>lab</i>	std::string Name of the status.

**Returns**

`add_status*` returns nothing.

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

**15.19.2.4 init()**

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

@

Functions to run the model

**Parameters**

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

### 15.19.2.5 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

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

### 15.19.2.6 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

### 15.19.2.7 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

### 15.19.2.8 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

<i>codes</i>	In the following order: Susceptible, Infected, Removed
<i>names</i>	Names matching the codes
<i>verbose</i>	When <code>true</code> , it will print the new mappings.

### 15.19.2.9 run\_multiple()

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

## Parameters

<i>nexperiments</i>	Multiple runs of the simulation
---------------------	---------------------------------

**15.19.2.10 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

<i>data</i> ↔ —	Pointer to the first element of an array of size <code>size() * ncols_</code> .
<i>ncols</i> ↔ —	Number of features included in the data.

**15.19.2.11 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. @[

**15.19.2.12 set\_rand\_engine()**

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

Random number generation.

## Parameters

<i>eng</i>	@[
------------	----



**15.19.2.13 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  $\{(i,j), (k,l)\} \rightarrow \{(i,l), (k,j)\}$ , the reciprocal is also true, i.e.,  $\{(j,i), (l,k)\} \rightarrow \{(j,k), (l,i)\}$ .

**Parameters**

<i>proportion</i>	Proportion of ties to be rewired.
-------------------	-----------------------------------

**Returns**

A rewired version of the network. @

**15.19.2.14 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**

<i>names</i>	@
--------------	---

**15.19.2.15 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

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

#### 15.19.2.16 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

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

#### 15.19.2.17 write\_edgelist()

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

Export the network data in edgelist form.

## Parameters

<i>fn</i>	std::string. File name.
<i>source</i>	Integer vector
<i>target</i>	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

## 15.20 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	<i>Random number generator</i>
s	<i>Seed</i>

- void **set\_rand\_engine** (std::mt19937 &eng)
- std::mt19937 \* **get\_rand\_engine** ()
- void **seed** (unsigned int s)
- void **set\_rand\_norm** (epiworld\_double mean, epiworld\_double sd)
- void **set\_rand\_unif** (epiworld\_double a, epiworld\_double b)
- void **set\_rand\_exp** (epiworld\_double lambda)
- void **set\_rand\_gamma** (epiworld\_double alpha, epiworld\_double beta)
- void **set\_rand\_lognormal** (epiworld\_double mean, epiworld\_double shape)
- epiworld\_double **runif** ()
- epiworld\_double **runif** (epiworld\_double a, epiworld\_double b)
- epiworld\_double **rnorm** ()
- epiworld\_double **rnorm** (epiworld\_double mean, epiworld\_double sd)
- epiworld\_double **rgamma** ()
- epiworld\_double **rgamma** (epiworld\_double alpha, epiworld\_double beta)
- epiworld\_double **rexp** ()
- epiworld\_double **rexp** (epiworld\_double lambda)
- epiworld\_double **rlognormal** ()
- epiworld\_double **rlognormal** (epiworld\_double mean, epiworld\_double shape)

### Add Virus/Tool to the model

*This is done before the model has been initialized.*

#### Parameters

v	<i><a href="#">Virus</a> to be added</i>
t	<i><a href="#">Tool</a> to be added</i>
preval	<i>Initial prevalence (initial state.) It can be specified as a proportion (between zero and one,) or an integer indicating number of individuals.</i>

- void **add\_virus** ([Virus](#)< TSeq > v, epiworld\_double preval)
- void **add\_virus\_n** ([Virus](#)< TSeq > v, unsigned int preval)
- void **add\_virus\_fun** ([Virus](#)< TSeq > v, VirusToAgentFun< TSeq > fun)
- void **add\_tool** ([Tool](#)< TSeq > t, epiworld\_double preval)
- void **add\_tool\_n** ([Tool](#)< TSeq > t, unsigned int preval)
- void **add\_tool\_fun** ([Tool](#)< TSeq > t, ToolToAgentFun< TSeq > fun)
- 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	<i>std::string Filename of the edgelist file.</i>
skip	<i>int Number of lines to skip in <i>fn</i>.</i>
directed	<i>bool Whether the graph is directed or not.</i>
size	<i>Size of the network.</i>
al	<i><a href="#">AdjList</a> to read into the model.</i>

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

- 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 nexperments, std::function< void(size\_t, [Model](#)< TSeq > \*)> fun=make\_save\_run< TSeq >(), bool **reset**=true, bool verbose=true)

### Rewire the network preserving the degree sequence.

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

#### Parameters

proportion	<i>Proportion of ties to be rewired.</i>
------------	------------------------------------------

#### 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	<i>std::string</i> . File name.
source	<i>Integer vector</i>
target	<i>Integer vector</i>

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	<i>std::string</i> 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	<i>string vector with the names of the variables.</i>
-------	-------------------------------------------------------

- 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 **queueing\_on** ()  
Activates the queueing system (default.)
- void **queueing\_off** ()  
Deactivates the queueing system.
- bool **is\_queueing\_on** () const  
Query if the queueing 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()`.

In the case of the function `read_params`, users can pass a file listing parameters to be included in the model. Each line in the file should have the following structure:

```
[name of parameter 1]: [value in double]
[name of parameter 2]: [value in double]
...
```

The only condition for parameter names is that these do not include a colon.

## Parameters

<i>initial_val</i>	
<i>pname</i>	Name of the parameter to add or to fetch
<i>fn</i>	Path to the file containing parameters

## 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)`
- `void read_params (std::string fn)`
- `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)`



## 15.20.1 Detailed Description

```
template<typename TSeq>
class Model< TSeq >
```

Core class of epiworld.

The model class provides the wrapper that puts together [Agent](#), [Virus](#), and [Tools](#).

### Template Parameters

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

## 15.20.2 Member Function Documentation

### 15.20.2.1 add\_global\_action()

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

Set a global action.

#### Parameters

<i>fun</i>	A function to be called on the prescribed dates
<i>date</i>	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.

### 15.20.2.2 reset()

```
template<typename TSeq >
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

### 15.20.2.3 run\_multiple()

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

#### Parameters

<i>nexperiments</i>	Multiple runs of the simulation
---------------------	---------------------------------

### 15.20.2.4 set\_agents\_data()

```
template<typename TSeq >
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

<i>data_</i> —	Pointer to the first element of an array of size <code>size() * ncols_</code> .
<i>ncols_</i> —	Number of features included in the data.

### 15.20.2.5 write\_data()

```
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

```

Wrapper of DataBase::write\_data

#### Parameters

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

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

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

## 15.21 Network< Nettype, Nodetype, Edgetype > Class Template Reference

### Public Member Functions

- **NType** ()
- Edgetype **operator()** (int i, int j)
- bool **is\_directed** () const
- size\_t **vcount** () const
- size\_t **ecount** () const
- void **add\_edge** (int i, int j)
- void **rm\_edge** (int i, int j)

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

- include/epiworld/network-bones.hpp

## 15.22 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\_engine** ()
- [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

## 15.23 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 >

### 15.23.1 Detailed Description

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

List of tools available for the individual to.

#### Template Parameters

<i>TSeq</i>	
-------------	--

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

- epiworld.hpp

## 15.24 PersonTools< TSeq > Class Template Reference

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

- include/epiworld/config.hpp

## 15.25 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 >

### 15.25.1 Detailed Description

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

Set of viruses in host.

#### Template Parameters

<i>TSeq</i>	
-------------	--

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

- rpackage/inst/include/epiworld.hpp

## 15.26 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** ()

### 15.26.1 Detailed Description

A simple progress bar.

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

- epiworld.hpp

## 15.27 Progress Class Reference

A simple progress bar.

```
#include <progress.hpp>
```

## Public Member Functions

- **Progress** (int n\_, int width\_)
- void **start** ()
- void **next** ()
- void **end** ()

### 15.27.1 Detailed Description

A simple progress bar.

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

- include/epiworld/progress.hpp

## 15.28 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)
- 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** ()

### 15.28.1 Detailed Description

```
template<typename TSeq>
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

<i>TSeq</i>	
-------------	--

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

- epiworld.hpp

## 15.29 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)
- void **set\_model** ([Model](#)< TSeq > \*m)

### 15.29.1 Detailed Description

```
template<typename TSeq>
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

<i>TSeq</i>	
-------------	--

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

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

## 15.30 epiworld::QueueValues Class Reference

### Static Public Attributes

- static const int **NoOne** = 0
- static const int **OnlySelf** = 1
- static const int **Everyone** = 2

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

- epiworld.hpp

## 15.31 QueueValues Class Reference

### Static Public Attributes

- static const int **NoOne** = 0
- static const int **OnlySelf** = 1
- static const int **Everyone** = 2

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

- include/epiworld/epiworld-macros.hpp

## 15.32 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

## 15.33 epiworld::SAMPLETYPE Class Reference

### Static Public Attributes

- static const int **MODEL** = 0
- static const int **ENTITY** = 1
- static const int **AGENT** = 2

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

- epiworld.hpp

## 15.34 SAMPLETYPE Class Reference

### Static Public Attributes

- static const int **MODEL** = 0
- static const int **ENTITY** = 1
- static const int **AGENT** = 2

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

- include/epiworld/agentssample-bones.hpp

## 15.35 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) **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	<i>The virus over which to operate</i>
fun	<i>the function to be used</i>

#### 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)

### 15.35.1 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

<i>TSeq</i>	Type of sequence
-------------	------------------

### 15.35.2 Member Function Documentation

#### 15.35.2.1 get\_susceptibility\_reduction()

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

Get and set the tool functions.

#### Parameters

<i>v</i>	The virus over which to operate
<i>fun</i>	the function to be used

#### Returns

epiworld\_double @[]

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

- epiworld.hpp

## 15.36 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	<i>The virus over which to operate</i>
fun	<i>the function to be used</i>

#### 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)

### 15.36.1 Detailed Description

```
template<typename TSeq>
class Tool< TSeq >
```

[Tools](#) for defending the agent against the virus.

#### Template Parameters

<i>TSeq</i>	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

## 15.37 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 >

### 15.37.1 Detailed Description

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

Set of tools (useful for building iterators)

## Template Parameters

<i>TSeq</i>	
-------------	--

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

- epiworld.hpp

## 15.38 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 >

### 15.38.1 Detailed Description

```
template<typename TSeq>
class Tools< TSeq >
```

Set of tools (useful for building iterators)

## Template Parameters

<i>TSeq</i>	
-------------	--

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

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

## 15.39 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 >

### 15.39.1 Detailed Description

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

Set of [Tools](#) (const) (useful for iterators)

Template Parameters

<i>TSeq</i>	
-------------	--

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

- epiworld.hpp

## 15.40 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 >

### 15.40.1 Detailed Description

```
template<typename TSeq>
class Tools_const< TSeq >
```

Set of [Tools](#) (const) (useful for iterators)

#### Template Parameters

<i>TSeq</i>	
-------------	--

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

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

## 15.41 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 <code>ncol ()</code> (if vector), otherwise a <code>epiworld_double</code> .
j	Index of the data point, from 0 to <code>ncol () - 1</code> .

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

### Access data

#### Parameters

i	Row (0 through <code>ndays - 1</code> .)
j	Column (0 through <code>ncols ()</code> ).

#### 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 >`

### 15.41.1 Detailed Description

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

Personalized data by the user.

#### Template Parameters

<code>TSeq</code>	
-------------------	--

## 15.41.2 Constructor & Destructor Documentation

### 15.41.2.1 UserData()

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

Construct a new User Data object.

#### Parameters

<i>names</i>	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

## 15.42 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

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

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

### Access data

#### Parameters

i	Row (0 through <i>ndays</i> - 1.)
j	Column (0 through <i>ncols</i> ()).

#### 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 >

## 15.42.1 Detailed Description

```
template<typename TSeq>
class UserData< TSeq >
```

Personalized data by the user.

#### Template Parameters

<i>TSeq</i>	
-------------	--

## 15.42.2 Constructor & Destructor Documentation

### 15.42.2.1 UserData()

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

Construct a new User Data object.

#### Parameters

<i>names</i>	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

## 15.43 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

#### 15.43.1 Detailed Description

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

Vector hasher.

Template Parameters

<i>T</i>	
----------	--

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

- epiworld.hpp

## 15.44 vecHasher< T > Struct Template Reference

Vector hasher.

```
#include <misc.hpp>
```

### Public Member Functions

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

#### 15.44.1 Detailed Description

```
template<typename T>
struct vecHasher< T >
```

Vector hasher.

## Template Parameters

<i>T</i>	
----------	--

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

- include/epiworld/misc.hpp

## 15.45 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	<i>The virus over which to operate</i>
fun	<i>the function to be used</i>

#### 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	<i>After the virus/tool is added to the agent.</i>
end	<i>After the virus/tool is removed.</i>
removed	<i>After the agent (<a href="#">Agent</a>) is removed.</i>

- 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)

### 15.45.1 Detailed Description

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

[Virus](#).

Template Parameters

<i>TSeq</i>	
-------------	--

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

Template Parameters

<i>TSeq</i>	
-------------	--

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

### 15.45.2 Member Function Documentation

#### 15.45.2.1 get\_prob\_infecting()

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

Get and set the tool functions.



**Parameters**

<i>v</i>	The virus over which to operate
<i>fun</i>	the function to be used

**Returns**

epiworld\_double @[]

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

- epiworld.hpp

## 15.46 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**

<i>v</i>	<i>The virus over which to operate</i>
<i>fun</i>	<i>the function to be used</i>

**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 ( <a href="#">Agent</a> ) 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)

## 15.46.1 Detailed Description

```
template<typename TSeq>
class Virus< TSeq >
```

[Virus](#).

#### Template Parameters

<i>TSeq</i>	
-------------	--

Raw transmissibility of a virus should be a function of its genetic sequence. Nonetheless, transmissibility can be

reduced as a result of having one or more tools to fight the virus. Because of this, transmissibility 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

## 15.47 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 >

#### 15.47.1 Detailed Description

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

Set of viruses (useful for building iterators)

##### Template Parameters

<i>TSeq</i>	
-------------	--

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

- epiworld.hpp

## 15.48 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 >

### 15.48.1 Detailed Description

```
template<typename TSeq>
class Viruses< TSeq >
```

Set of viruses (useful for building iterators)

Template Parameters

<i>TSeq</i>	
-------------	--

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

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

## 15.49 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 >

### 15.49.1 Detailed Description

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

Set of [Viruses](#) (const) (useful for iterators)

#### Template Parameters

<i>TSeq</i>	
-------------	--

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

- epiworld.hpp

## 15.50 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 >

### 15.50.1 Detailed Description

```
template<typename TSeq>
class Viruses_const< TSeq >
```

Set of [Viruses](#) (const) (useful for iterators)

**Template Parameters**

<i>TSeq</i>	
-------------	--

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

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

## Chapter 16

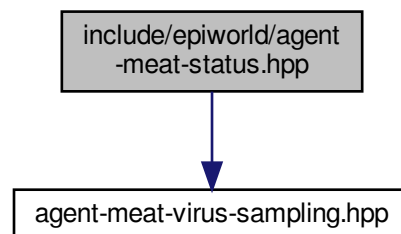
# File Documentation

### 16.1 include/epiworld/agent-meat-status.hpp File Reference

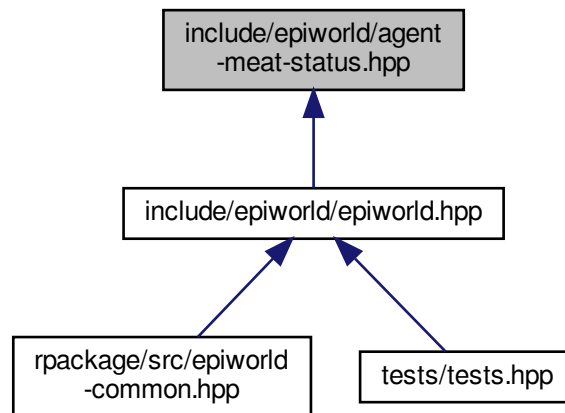
Sampling functions are getting big, so we keep them in a separate file.

```
#include "agent-meat-virus-sampling.hpp"
```

Include dependency graph for agent-meat-status.hpp:



This graph shows which files directly or indirectly include this file:



## Functions

- `template<typename TSeq = EPI_DEFAULT_TSEQ>`  
`void default_update_susceptible (Agent< TSeq > *p, Model< TSeq > *m)`
- `template<typename TSeq = EPI_DEFAULT_TSEQ>`  
`void default_update_exposed (Agent< TSeq > *p, Model< TSeq > *m)`

### 16.1.1 Detailed Description

Sampling functions are getting big, so we keep them in a separate file.

#### Author

George G. Vega Yon (g.vegayon en gmail)

#### Version

0.1

#### Date

2022-06-15

#### Copyright

Copyright (c) 2022



# Index

Action  
  Action< TSeq >, 40  
  epiworld::Action< TSeq >, 41  
Action< TSeq >, 39  
  Action, 40  
add\_global\_action  
  epiworld::Model< TSeq >, 76  
  Model< TSeq >, 89  
add\_param  
  epiworld::Model< TSeq >, 76  
add\_status\_susceptible  
  epiworld::Model< TSeq >, 77  
AdjList, 42  
  AdjList, 43  
  epiworld::AdjList, 44, 45  
  read\_edgelist, 43  
Agent< TSeq >, 46  
  operator(), 48  
AgentsSample< TSeq >, 52  
  
DataBase< TSeq >, 53  
  record\_variant, 55  
  reproductive\_number, 55  
  transition\_probability, 56  
  
Entities< TSeq >, 60  
Entities\_const< TSeq >, 62  
Entity< TSeq >, 63  
epiworld::Action< TSeq >, 41  
  Action, 41  
epiworld::AdjList, 44  
  AdjList, 44, 45  
  read\_edgelist, 45  
epiworld::Agent< TSeq >, 49  
  operator(), 51  
epiworld::AgentsSample< TSeq >, 53  
epiworld::DataBase< TSeq >, 56  
  get\_today\_total, 58  
  record\_variant, 59  
  reproductive\_number, 60  
  transition\_probability, 60  
epiworld::Entities< TSeq >, 61  
epiworld::Entities\_const< TSeq >, 63  
epiworld::Entity< TSeq >, 64  
epiworld::LFMCMC< TData >, 65  
epiworld::Model< TSeq >, 67  
  add\_global\_action, 76  
  add\_param, 76  
  add\_status\_susceptible, 77  
  init, 77  
  pop\_from\_adjlist, 77  
  reset, 78  
  reset\_status\_codes, 79  
  run\_multiple, 79  
  set\_agents\_data, 80  
  set\_backup, 80  
  set\_rand\_engine, 80  
  set\_rewire\_fun, 81  
  set\_user\_data, 81  
  write\_data, 81, 82  
  write\_edgelist, 82  
epiworld::Person< TSeq >, 92  
epiworld::PersonTools< TSeq >, 92  
epiworld::PersonViruses< TSeq >, 94  
epiworld::Progress, 94  
epiworld::Queue< TSeq >, 95  
epiworld::QueueValues, 97  
epiworld::sampler, 31  
  make\_sample\_virus\_neighbors, 31  
  make\_update\_susceptible, 32  
  sample\_virus\_single, 32  
epiworld::SAMPLETYPE, 98  
epiworld::Tool< TSeq >, 98  
  get\_susceptibility\_reduction, 100  
epiworld::Tools< TSeq >, 102  
epiworld::Tools\_const< TSeq >, 104  
epiworld::UserData< TSeq >, 105  
  UserData, 107  
epiworld::vecHasher< T >, 109  
epiworld::Virus< TSeq >, 110  
  get\_prob\_infecting, 112  
epiworld::Viruses< TSeq >, 115  
epiworld::Viruses\_const< TSeq >, 116  
  
get\_prob\_infecting  
  epiworld::Virus< TSeq >, 112  
get\_susceptibility\_reduction  
  epiworld::Tool< TSeq >, 100  
get\_today\_total  
  epiworld::DataBase< TSeq >, 58  
  
include/epiworld/agent-meat-status.hpp, 119  
init  
  epiworld::Model< TSeq >, 77  
  
LFMCMC< TData >, 66  
  
make\_sample\_virus\_neighbors  
  epiworld::sampler, 31  
  sampler, 34

make\_update\_susceptible  
     epiworld::sampler, 32  
     sampler, 35  
 Model< TSeq >, 83  
     add\_global\_action, 89  
     reset, 89  
     run\_multiple, 90  
     set\_agents\_data, 90  
     write\_data, 90  
  
 Network< Nettype, Nodetype, Edgetype >, 91  
  
 operator()  
     Agent< TSeq >, 48  
     epiworld::Agent< TSeq >, 51  
  
 PersonTools< TSeq >, 93  
 pop\_from\_adjlist  
     epiworld::Model< TSeq >, 77  
 Progress, 95  
  
 Queue< TSeq >, 96  
 QueueValues, 97  
  
 RandGraph, 97  
 read\_edgelist  
     AdjList, 43  
     epiworld::AdjList, 45  
 record\_variant  
     DataBase< TSeq >, 55  
     epiworld::DataBase< TSeq >, 59  
 reproductive\_number  
     DataBase< TSeq >, 55  
     epiworld::DataBase< TSeq >, 60  
 reset  
     epiworld::Model< TSeq >, 78  
     Model< TSeq >, 89  
 reset\_status\_codes  
     epiworld::Model< TSeq >, 79  
 run\_multiple  
     epiworld::Model< TSeq >, 79  
     Model< TSeq >, 90  
  
 sample\_virus\_single  
     epiworld::sampler, 32  
     sampler, 35  
 sampler, 34  
     make\_sample\_virus\_neighbors, 34  
     make\_update\_susceptible, 35  
     sample\_virus\_single, 35  
 SAMPLETYPE, 98  
 set\_agents\_data  
     epiworld::Model< TSeq >, 80  
     Model< TSeq >, 90  
 set\_backup  
     epiworld::Model< TSeq >, 80  
 set\_rand\_engine  
     epiworld::Model< TSeq >, 80  
 set\_rewire\_fun  
     epiworld::Model< TSeq >, 81  
  
 set\_user\_data  
     epiworld::Model< TSeq >, 81  
  
 Tool< TSeq >, 101  
 Tools< TSeq >, 103  
 Tools\_const< TSeq >, 104  
 transition\_probability  
     DataBase< TSeq >, 56  
     epiworld::DataBase< TSeq >, 60  
  
 UserData  
     epiworld::UserData< TSeq >, 107  
     UserData< TSeq >, 108  
 UserData< TSeq >, 107  
     UserData, 108  
  
 vecHasher< T >, 109  
 Virus< TSeq >, 113  
 Viruses< TSeq >, 116  
 Viruses\_const< TSeq >, 117  
  
 write\_data  
     epiworld::Model< TSeq >, 81, 82  
     Model< TSeq >, 90  
 write\_edgelist  
     epiworld::Model< TSeq >, 82