

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 Namespace Index	17
8.1 Namespace List	17
9 Hierarchical Index	19
9.1 Class Hierarchy	19
10 Class Index	21
10.1 Class List	21
11 File Index	25
11.1 File List	25
12 Namespace Documentation	27
12.1 epiworld::sampler Namespace Reference	27
12.1.1 Detailed Description	27
12.1.2 Function Documentation	27
12.1.2.1 make_sample_virus_neighbors()	27
12.1.2.2 make_update_susceptible()	28
12.1.2.3 sample_virus_single()	28
12.2 sampler Namespace Reference	30

12.2.1 Detailed Description	30
12.2.2 Function Documentation	30
12.2.2.1 make_sample_virus_neighbors()	30
12.2.2.2 make_update_susceptible()	31
12.2.2.3 sample_virus_single()	31
13 Class Documentation	35
13.1 Action< TSeq > Struct Template Reference	35
13.1.1 Detailed Description	35
13.1.2 Constructor & Destructor Documentation	36
13.1.2.1 Action()	36
13.2 epiworld::Action< TSeq > Struct Template Reference	37
13.2.1 Detailed Description	37
13.2.2 Constructor & Destructor Documentation	37
13.2.2.1 Action()	37
13.3 AdjList Class Reference	38
13.3.1 Constructor & Destructor Documentation	39
13.3.1.1 AdjList()	39
13.3.2 Member Function Documentation	39
13.3.2.1 read_edgelist()	39
13.4 epiworld::AdjList Class Reference	40
13.4.1 Constructor & Destructor Documentation	40
13.4.1.1 AdjList()	40
13.4.2 Member Function Documentation	41
13.4.2.1 read_edgelist()	41
13.5 Agent< TSeq > Class Template Reference	41
13.5.1 Detailed Description	43
13.5.2 Member Function Documentation	44
13.5.2.1 get_entities()	44
13.5.2.2 swap_neighbors()	44
13.5.3 Friends And Related Function Documentation	44
13.5.3.1 default_rm_entity	45
13.6 epiworld::Agent< TSeq > Class Template Reference	45
13.6.1 Detailed Description	47
13.6.2 Member Function Documentation	47
13.6.2.1 get_entities()	47
13.6.2.2 swap_neighbors()	48
13.6.3 Friends And Related Function Documentation	48
13.6.3.1 default_rm_entity	48
13.7 AgentsSample< TSeq > Class Template Reference	49
13.7.1 Detailed Description	49
13.7.2 Constructor & Destructor Documentation	49

13.7.2.1 AgentsSample()	49
13.8 epiworld::AgentsSample< TSeq > Class Template Reference	50
13.8.1 Detailed Description	51
13.8.2 Constructor & Destructor Documentation	51
13.8.2.1 AgentsSample()	51
13.9 DataBase< TSeq > Class Template Reference	52
13.9.1 Detailed Description	53
13.9.2 Member Function Documentation	53
13.9.2.1 generation_time()	54
13.9.2.2 operator==() [1/3]	54
13.9.2.3 operator==() [2/3]	54
13.9.2.4 operator==() [3/3]	55
13.9.2.5 record_variant()	55
13.9.2.6 reproductive_number()	55
13.9.2.7 transition_probability()	55
13.10 epiworld::DataBase< TSeq > Class Template Reference	56
13.10.1 Detailed Description	57
13.10.2 Member Function Documentation	58
13.10.2.1 generation_time()	58
13.10.2.2 operator==()	58
13.10.2.3 record_variant()	58
13.10.2.4 reproductive_number()	59
13.10.2.5 transition_probability()	59
13.11 Entities< TSeq > Class Template Reference	59
13.11.1 Detailed Description	60
13.12 epiworld::Entities< TSeq > Class Template Reference	60
13.12.1 Detailed Description	61
13.13 Entities_const< TSeq > Class Template Reference	61
13.13.1 Detailed Description	61
13.14 epiworld::Entities_const< TSeq > Class Template Reference	62
13.14.1 Detailed Description	62
13.15 Entity< TSeq > Class Template Reference	63
13.15.1 Friends And Related Function Documentation	63
13.15.1.1 default_rm_entity	63
13.16 epiworld::Entity< TSeq > Class Template Reference	64
13.16.1 Friends And Related Function Documentation	64
13.16.1.1 default_rm_entity	64
13.17 epiworld::LFMCMC< TData > Class Template Reference	65
13.17.1 Detailed Description	65
13.18 LFMCMC< TData > Class Template Reference	66
13.18.1 Detailed Description	67
13.19 epiworld::Model< TSeq > Class Template Reference	67

13.19.1 Detailed Description	74
13.19.2 Member Function Documentation	74
13.19.2.1 actions_add()	75
13.19.2.2 actions_run()	75
13.19.2.3 add_global_action()	75
13.19.2.4 clone_ptr()	76
13.19.2.5 load_agents_entities_ties()	76
13.19.2.6 reset()	77
13.19.2.7 run_multiple()	77
13.19.2.8 set_agents_data()	77
13.19.2.9 set_name()	78
13.19.2.10 write_data()	78
13.19.3 Member Data Documentation	79
13.19.3.1 time_elapsed	79
13.20 Model< TSeq > Class Template Reference	79
13.20.1 Detailed Description	86
13.20.2 Member Function Documentation	86
13.20.2.1 actions_add()	86
13.20.2.2 actions_run()	87
13.20.2.3 add_global_action()	87
13.20.2.4 clone_ptr()	88
13.20.2.5 load_agents_entities_ties()	88
13.20.2.6 reset()	88
13.20.2.7 run_multiple()	89
13.20.2.8 set_agents_data()	89
13.20.2.9 set_name()	90
13.20.2.10 write_data()	90
13.20.3 Member Data Documentation	90
13.20.3.1 time_elapsed	90
13.21 epiworld::epimodels::ModelSEIR< TSeq > Class Template Reference	91
13.21.1 Detailed Description	92
13.21.2 Member Data Documentation	92
13.21.2.1 update_exposed_seir	92
13.21.2.2 update_infected_seir	93
13.22 ModelSEIR< TSeq > Class Template Reference	93
13.22.1 Detailed Description	94
13.22.2 Member Data Documentation	94
13.22.2.1 update_exposed_seir	94
13.22.2.2 update_infected_seir	95
13.23 epiworld::epimodels::ModelSEIRCONN< TSeq > Class Template Reference	95
13.23.1 Constructor & Destructor Documentation	96
13.23.1.1 ModelSEIRCONN()	96

13.23.2 Member Function Documentation	97
13.23.2.1 clone_ptr()	97
13.24 ModelSEIRCONN< TSeq > Class Template Reference	97
13.24.1 Constructor & Destructor Documentation	98
13.24.1.1 ModelSEIRCONN()	99
13.24.2 Member Function Documentation	99
13.24.2.1 clone_ptr()	99
13.25 ModelSEIRCONNLogit< TSeq > Class Template Reference	100
13.25.1 Constructor & Destructor Documentation	101
13.25.1.1 ModelSEIRCONNLogit()	101
13.26 epiworld::epimodels::ModelSEIRD< TSeq > Class Template Reference	101
13.26.1 Detailed Description	103
13.27 ModelSEIRD< TSeq > Class Template Reference	103
13.27.1 Detailed Description	104
13.28 epiworld::epimodels::ModelSIR< TSeq > Class Template Reference	105
13.28.1 Detailed Description	106
13.29 ModelSIR< TSeq > Class Template Reference	107
13.29.1 Detailed Description	108
13.30 epiworld::epimodels::ModelSIRCONN< TSeq > Class Template Reference	108
13.30.1 Constructor & Destructor Documentation	109
13.30.1.1 ModelSIRCONN()	110
13.30.2 Member Function Documentation	110
13.30.2.1 clone_ptr()	110
13.31 ModelSIRCONN< TSeq > Class Template Reference	111
13.31.1 Constructor & Destructor Documentation	112
13.31.1.1 ModelSIRCONN()	112
13.31.2 Member Function Documentation	112
13.31.2.1 clone_ptr()	112
13.32 epiworld::epimodels::ModelSIS< TSeq > Class Template Reference	113
13.32.1 Detailed Description	114
13.33 ModelSIS< TSeq > Class Template Reference	114
13.33.1 Detailed Description	115
13.34 epiworld::epimodels::ModelSURV< TSeq > Class Template Reference	116
13.35 ModelSURV< TSeq > Class Template Reference	118
13.36 Network< Nettype, Nodetype, Edgetype > Class Template Reference	120
13.37 epiworld::PersonTools< TSeq > Class Template Reference	120
13.38 PersonTools< TSeq > Class Template Reference	120
13.39 epiworld::Progress Class Reference	120
13.39.1 Detailed Description	121
13.40 Progress Class Reference	121
13.40.1 Detailed Description	121
13.41 epiworld::Queue< TSeq > Class Template Reference	121

13.41.1 Detailed Description	121
13.42 Queue< TSeq > Class Template Reference	122
13.42.1 Detailed Description	122
13.43 epiworld::QueueValues Class Reference	123
13.44 QueueValues Class Reference	123
13.45 RandGraph Class Reference	123
13.46 epiworld::SAMPLETYPE Class Reference	123
13.47 SAMPLETYPE Class Reference	124
13.48 epiworld::Tool< TSeq > Class Template Reference	124
13.48.1 Detailed Description	125
13.49 Tool< TSeq > Class Template Reference	125
13.49.1 Detailed Description	127
13.50 epiworld::Tools< TSeq > Class Template Reference	127
13.50.1 Detailed Description	127
13.51 Tools< TSeq > Class Template Reference	128
13.51.1 Detailed Description	128
13.52 epiworld::Tools_const< TSeq > Class Template Reference	129
13.52.1 Detailed Description	129
13.53 Tools_const< TSeq > Class Template Reference	129
13.53.1 Detailed Description	130
13.54 epiworld::UserData< TSeq > Class Template Reference	130
13.54.1 Detailed Description	131
13.54.2 Constructor & Destructor Documentation	131
13.54.2.1 UserData()	131
13.55 UserData< TSeq > Class Template Reference	132
13.55.1 Detailed Description	133
13.55.2 Constructor & Destructor Documentation	133
13.55.2.1 UserData()	133
13.56 epiworld::vecHasher< T > Struct Template Reference	134
13.56.1 Detailed Description	134
13.57 vecHasher< T > Struct Template Reference	134
13.57.1 Detailed Description	134
13.58 epiworld::Virus< TSeq > Class Template Reference	135
13.58.1 Detailed Description	136
13.59 Virus< TSeq > Class Template Reference	137
13.59.1 Detailed Description	138
13.60 epiworld::Viruses< TSeq > Class Template Reference	139
13.60.1 Detailed Description	139
13.61 Viruses< TSeq > Class Template Reference	139
13.61.1 Detailed Description	140
13.62 epiworld::Viruses_const< TSeq > Class Template Reference	140
13.62.1 Detailed Description	140

13.63 Viruses_const< TSeq > Class Template Reference	141
13.63.1 Detailed Description	141
14 File Documentation	143
14.1 include/epiworld/agent-meat-state.hpp File Reference	143
14.1.1 Detailed Description	144
Index	145

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 entities  : 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 states (arbitrary number of them).

Run

1. Distribute the tool(s) and virus(es)
2. For each t in 1 -> Duration:
 - Update state 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 = state, 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 the can be a function of an ERGM. Apply K steps throughout time.
- Add progress bar.

## Chapter 8

# Namespace Index

### 8.1 Namespace List

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

<a href="#">epiworld::sampler</a>	Functions for sampling viruses . . . . .	<a href="#">27</a>
<a href="#">sampler</a>	Functions for sampling viruses . . . . .	<a href="#">30</a>



## Chapter 9

# Hierarchical Index

### 9.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Action< TSeq > . . . . .	35
epiworld::Action< TSeq > . . . . .	37
AdjList . . . . .	38
epiworld::AdjList . . . . .	40
Agent< TSeq > . . . . .	41
epiworld::Agent< TSeq > . . . . .	45
AgentsSample< TSeq > . . . . .	49
epiworld::AgentsSample< TSeq > . . . . .	50
DataBase< TSeq > . . . . .	52
epiworld::DataBase< TSeq > . . . . .	56
Entities< TSeq > . . . . .	59
epiworld::Entities< TSeq > . . . . .	60
Entities_const< TSeq > . . . . .	61
epiworld::Entities_const< TSeq > . . . . .	62
Entity< TSeq > . . . . .	63
epiworld::Entity< TSeq > . . . . .	64
epiworld::LFMCMC< TData > . . . . .	65
LFMCMC< TData > . . . . .	66
epiworld::Model< TSeq > . . . . .	67
Model< TSeq > . . . . .	79
epiworld::Model< EPI_DEFAULT_TSEQ > . . . . .	67
ModelSEIRCONN< TSeq > . . . . .	97
ModelSEIRCONNLogit< TSeq > . . . . .	100
ModelSIRCONN< TSeq > . . . . .	111
ModelSURV< TSeq > . . . . .	118
epiworld::epimodels::ModelSEIRCONN< TSeq > . . . . .	95
epiworld::epimodels::ModelSIRCONN< TSeq > . . . . .	108
epiworld::epimodels::ModelSURV< TSeq > . . . . .	116
epiworld::Model< int > . . . . .	67
ModelSEIR< TSeq > . . . . .	93
ModelSEIRD< TSeq > . . . . .	103
ModelSIR< TSeq > . . . . .	107
ModelSIS< TSeq > . . . . .	114
epiworld::epimodels::ModelSEIR< TSeq > . . . . .	91
epiworld::epimodels::ModelSEIRD< TSeq > . . . . .	101

epiworld::epimodels::ModelSIR< TSeq > . . . . .	105
epiworld::epimodels::ModelSIS< TSeq > . . . . .	113
Network< Nettype, Nodetype, Edgetype > . . . . .	120
epiworld::PersonTools< TSeq > . . . . .	120
PersonTools< TSeq > . . . . .	120
epiworld::Progress . . . . .	120
Progress . . . . .	121
epiworld::Queue< TSeq > . . . . .	121
Queue< TSeq > . . . . .	122
epiworld::QueueValues . . . . .	123
QueueValues . . . . .	123
RandGraph . . . . .	123
epiworld::SAMPLETYPE . . . . .	123
SAMPLETYPE . . . . .	124
epiworld::Tool< TSeq > . . . . .	124
Tool< TSeq > . . . . .	125
epiworld::Tools< TSeq > . . . . .	127
Tools< TSeq > . . . . .	128
epiworld::Tools_const< TSeq > . . . . .	129
Tools_const< TSeq > . . . . .	129
epiworld::UserData< TSeq > . . . . .	130
UserData< TSeq > . . . . .	132
epiworld::vecHasher< T > . . . . .	134
vecHasher< T > . . . . .	134
epiworld::Virus< TSeq > . . . . .	135
Virus< TSeq > . . . . .	137
epiworld::Viruses< TSeq > . . . . .	139
Viruses< TSeq > . . . . .	139
epiworld::Viruses_const< TSeq > . . . . .	140
Viruses_const< TSeq > . . . . .	141

## Chapter 10

# Class Index

### 10.1 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	35
<a href="#">epiworld::Action&lt; TSeq &gt;</a>	
Action data for update an agent	37
<a href="#">AdjList</a>	38
<a href="#">epiworld::AdjList</a>	40
<a href="#">Agent&lt; TSeq &gt;</a>	
Agent (agents)	41
<a href="#">epiworld::Agent&lt; TSeq &gt;</a>	
Agent (agents)	45
<a href="#">AgentsSample&lt; TSeq &gt;</a>	
Sample of agents	49
<a href="#">epiworld::AgentsSample&lt; TSeq &gt;</a>	
Sample of agents	50
<a href="#">DataBase&lt; TSeq &gt;</a>	
Statistical data about the process	52
<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)	59
<a href="#">epiworld::Entities&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (useful for building iterators)	60
<a href="#">Entities_const&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (const) (useful for iterators)	61
<a href="#">epiworld::Entities_const&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (const) (useful for iterators)	62
<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	79

<a href="#">epiworld::epimodels::ModelSEIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed (SEIR) model	91
<a href="#">ModelSEIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed (SEIR) model	93
<a href="#">epiworld::epimodels::ModelSEIRCONN&lt; TSeq &gt;</a>	95
<a href="#">ModelSEIRCONN&lt; TSeq &gt;</a>	97
<a href="#">ModelSEIRCONNLogit&lt; TSeq &gt;</a>	100
<a href="#">epiworld::epimodels::ModelSEIRD&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model	101
<a href="#">ModelSEIRD&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model	103
<a href="#">epiworld::epimodels::ModelSIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed (SIR) model	105
<a href="#">ModelSIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed (SIR) model	107
<a href="#">epiworld::epimodels::ModelSIRCONN&lt; TSeq &gt;</a>	108
<a href="#">ModelSIRCONN&lt; TSeq &gt;</a>	111
<a href="#">epiworld::epimodels::ModelSIS&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Susceptible (SIS) model	113
<a href="#">ModelSIS&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Susceptible (SIS) model	114
<a href="#">epiworld::epimodels::ModelSURV&lt; TSeq &gt;</a>	116
<a href="#">ModelSURV&lt; TSeq &gt;</a>	118
<a href="#">Network&lt; Nettype, Nodetype, Edgetype &gt;</a>	120
<a href="#">epiworld::PersonTools&lt; TSeq &gt;</a>	120
<a href="#">PersonTools&lt; TSeq &gt;</a>	120
<a href="#">epiworld::Progress</a>	
A simple progress bar	120
<a href="#">Progress</a>	
A simple progress bar	121
<a href="#">epiworld::Queue&lt; TSeq &gt;</a>	
Controls which agents are verified at each step	121
<a href="#">Queue&lt; TSeq &gt;</a>	
Controls which agents are verified at each step	122
<a href="#">epiworld::QueueValues</a>	123
<a href="#">QueueValues</a>	123
<a href="#">RandGraph</a>	123
<a href="#">epiworld::SAMPLETYPE</a>	123
<a href="#">SAMPLETYPE</a>	124
<a href="#">epiworld::Tool&lt; TSeq &gt;</a>	
Tools for defending the agent against the virus	124
<a href="#">Tool&lt; TSeq &gt;</a>	
Tools for defending the agent against the virus	125
<a href="#">epiworld::Tools&lt; TSeq &gt;</a>	
Set of tools (useful for building iterators)	127
<a href="#">Tools&lt; TSeq &gt;</a>	
Set of tools (useful for building iterators)	128
<a href="#">epiworld::Tools_const&lt; TSeq &gt;</a>	
Set of <a href="#">Tools</a> (const) (useful for iterators)	129
<a href="#">Tools_const&lt; TSeq &gt;</a>	
Set of <a href="#">Tools</a> (const) (useful for iterators)	129
<a href="#">epiworld::UserData&lt; TSeq &gt;</a>	
Personalized data by the user	130
<a href="#">UserData&lt; TSeq &gt;</a>	
Personalized data by the user	132
<a href="#">epiworld::vecHasher&lt; T &gt;</a>	
Vector hasher	134



<a href="#">vecHasher&lt; T &gt;</a>	
Vector hasher . . . . .	134
<a href="#">epiworld::Virus&lt; TSeq &gt;</a>	
Virus . . . . .	135
<a href="#">Virus&lt; TSeq &gt;</a>	
Virus . . . . .	137
<a href="#">epiworld::Viruses&lt; TSeq &gt;</a>	
Set of viruses (useful for building iterators) . . . . .	139
<a href="#">Viruses&lt; TSeq &gt;</a>	
Set of viruses (useful for building iterators) . . . . .	139
<a href="#">epiworld::Viruses_const&lt; TSeq &gt;</a>	
Set of <a href="#">Viruses</a> (const) (useful for iterators) . . . . .	140
<a href="#">Viruses_const&lt; TSeq &gt;</a>	
Set of <a href="#">Viruses</a> (const) (useful for iterators) . . . . .	141



# Chapter 11

## File Index

### 11.1 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-state.hpp</b>	
Sampling functions are getting big, so we keep them in a separate file	143
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>seirconnected_logit.hpp</b> . . . . .	??
include/epiworld/models/ <b>seird.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> . . . . .	??
tests/ <b>tests.hpp</b> . . . . .	??

## Chapter 12

# Namespace Documentation

### 12.1 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)*

#### 12.1.1 Detailed Description

Functions for sampling viruses.

#### 12.1.2 Function Documentation

##### 12.1.2.1 `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;

**12.1.2.2 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;

**12.1.2.3 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;

## 12.2 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)*

### 12.2.1 Detailed Description

Functions for sampling viruses.

### 12.2.2 Function Documentation

#### 12.2.2.1 [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;

## 12.2.2.2 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;

## 12.2.2.3 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 13

# Class Documentation

### 13.1 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\_state\_, 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\_state
- epiworld\_fast\_int queue
- ActionFun< TSeq > call
- int idx\_agent
- int idx\_object

#### 13.1.1 Detailed Description

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

Action data for update an agent.

## Template Parameters

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

## 13.1.2 Constructor &amp; Destructor Documentation

## 13.1.2.1 Action()

```
template<typename TSeq >
Action< TSeq >::Action (
 Agent< TSeq > * agent_,
 VirusPtr< TSeq > virus_,
 ToolPtr< TSeq > tool_,
 Entity< TSeq > * entity_,
 epiworld_fast_int new_state_,
 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_↔ state_</i>	Next state
<i>queue_</i>	Effect 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

## 13.2 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\_state\_, 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\_state**
- epiworld\_fast\_int **queue**
- [ActionFun](#)< TSeq > **call**
- int **idx\_agent**
- int **idx\_object**

### 13.2.1 Detailed Description

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

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

Template Parameters

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

### 13.2.2 Constructor & Destructor Documentation

#### 13.2.2.1 Action()

```
template<typename TSeq >
epiworld::Action< TSeq >::Action (
 Agent< TSeq > * agent_,
```

```

VirusPtr< TSeq > virus_,
ToolPtr< TSeq > tool_,
Entity< TSeq > * entity_,
epiworld_fast_int new_state_,
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_↔ state_</i>	Next state
<i>queue_</i>	Effect 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

## 13.3 AdjList Class Reference

### Public Member Functions

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

*Construct a new Adj List object.*

- **AdjList** ([AdjList](#) &&a)
- **AdjList** (const [AdjList](#) &a)
- [AdjList](#) & **operator=** (const [AdjList](#) &a)
- void [read\\_edgelist](#) (std::string fn, int size, int skip=0, bool directed=true)

*Read an edgelist.*

- std::map< int, int > **operator()** (epiworld\_fast\_uint i) const
- void **print** (epiworld\_fast\_uint 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< int, int > > & **get\_dat** ()
- bool [is\\_directed](#) () const

*true if the network is directed.*



### 13.3.1 Constructor & Destructor Documentation

#### 13.3.1.1 AdjList()

```
AdjList::AdjList (
 const std::vector< int > & source,
 const std::vector< 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

### 13.3.2 Member Function Documentation

#### 13.3.2.1 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

## 13.4 epiworld::AdjList Class Reference

### Public Member Functions

- [AdjList](#) (const std::vector< int > &source, const std::vector< int > &target, int size, bool directed)  
*Construct a new Adj List object.*
- [AdjList](#) ([AdjList](#) &&a)
- [AdjList](#) (const [AdjList](#) &a)
- [AdjList](#) & [operator=](#) (const [AdjList](#) &a)
- void [read\\_edgelist](#) (std::string fn, int size, int skip=0, bool directed=true)  
*Read an edgelist.*
- std::map< int, int > [operator\(\)](#) (epiworld\_fast\_uint i) const
- void [print](#) (epiworld\_fast\_uint 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< int, int > > & [get\\_dat](#) ()
- bool [is\\_directed](#) () const  
*true if the network is directed.*

### 13.4.1 Constructor & Destructor Documentation

#### 13.4.1.1 AdjList()

```
AdjList::AdjList (
 const std::vector< int > & source,
 const std::vector< 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

## 13.4.2 Member Function Documentation

### 13.4.2.1 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`

## 13.5 Agent< TSeq > Class Template Reference

[Agent](#) (agents)

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

### Public Member Functions

- **Agent** ([Agent](#)< TSeq > &&p)
- **Agent** (const [Agent](#)< TSeq > &p)
- [Agent](#)< TSeq > & **operator=** (const [Agent](#)< TSeq > &other\_agent)
- int [get\\_id](#) () const  
*Id of the individual.*
- 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)
- void **swap\_neighbors** ([Agent](#)< TSeq > &other, size\_t n\_this, size\_t n\_other)  
*Swaps neighbors between the current agent and agent *other**
- std::vector< [Agent](#)< TSeq > \* > **get\_neighbors** ()
- size\_t **get\_n\_neighbors** () const
- void **change\_state** ([Model](#)< TSeq > \*model, epiworld\_fast\_uint new\_state, epiworld\_fast\_int queue=0)
- const epiworld\_fast\_uint & **get\_state** () const
- void **reset** ()
- bool **has\_tool** (epiworld\_fast\_uint t) const
- bool **has\_tool** (std::string name) const
- bool **has\_virus** (epiworld\_fast\_uint t) const
- bool **has\_virus** (std::string name) const
- void **print** ([Model](#)< TSeq > \*model, bool compressed=false) const
- [Entities](#)< TSeq > **get\_entities** ()  
*Access the *j*-th column of the agent.*
- const [Entities\\_const](#)< TSeq > **get\_entities** () const
- const [Entity](#)< TSeq > & **get\_entity** (size\_t i) const
- [Entity](#)< TSeq > & **get\_entity** (size\_t i)
- size\_t **get\_n\_entities** () const
- bool **operator==** (const [Agent](#)< TSeq > &other) const
- bool **operator!=** (const [Agent](#)< TSeq > &other) const

### Add/Remove Virus/Tool

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

#### Parameters

tool	<a href="#">Tool</a> to add
virus	<a href="#">Virus</a> to add
status_new	state after the change
queue	

- void **add\_tool** ([ToolPtr](#)< TSeq > tool, [Model](#)< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **add\_tool** ([Tool](#)< TSeq > tool, [Model](#)< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **add\_virus** ([VirusPtr](#)< TSeq > virus, [Model](#)< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **add\_virus** ([Virus](#)< TSeq > virus, [Model](#)< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **add\_entity** ([Entity](#)< TSeq > &entity, [Model](#)< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_tool** (epiworld\_fast\_uint tool\_idx, [Model](#)< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_tool** ([ToolPtr](#)< TSeq > &tool, [Model](#)< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_virus** (epiworld\_fast\_uint virus\_idx, [Model](#)< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_virus** ([VirusPtr](#)< TSeq > &virus, [Model](#)< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_entity** (epiworld\_fast\_uint entity\_idx, [Model](#)< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_entity** ([Entity](#)< TSeq > &entity, [Model](#)< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)

- void [rm\\_agent\\_by\\_virus](#) (epiworld\_fast\_uint virus\_idx, [Model](#)< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)  
*Agent removed by virus.*
- void [rm\\_agent\\_by\\_virus](#) (VirusPtr< TSeq > &virus, [Model](#)< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)  
*Agent removed by virus.*

### Get the rates (multipliers) for the agent

#### Parameters

v	A pointer to a virus.
---	-----------------------

#### Returns

*epiworld\_double*

- epiworld\_double **get\_susceptibility\_reduction** (VirusPtr< TSeq > v, [Model](#)< TSeq > \*model)
- epiworld\_double **get\_transmission\_reduction** (VirusPtr< TSeq > v, [Model](#)< TSeq > \*model)
- epiworld\_double **get\_recovery\_enhancer** (VirusPtr< TSeq > v, [Model](#)< TSeq > \*model)
- epiworld\_double **get\_death\_reduction** (VirusPtr< TSeq > v, [Model](#)< TSeq > \*model)

### Friends

- class [Model](#)< TSeq >
- class [Virus](#)< TSeq >
- class [Viruses](#)< TSeq >
- class [Viruses\\_const](#)< TSeq >
- class [Tool](#)< TSeq >
- class [Tools](#)< TSeq >
- class [Tools\\_const](#)< 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)

## 13.5.1 Detailed Description

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

[Agent](#) (agents)

#### Template Parameters

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

## 13.5.2 Member Function Documentation

### 13.5.2.1 `get_entities()`

```
template<typename TSeq >
Entities< TSeq > Agent< TSeq >::get_entities [inline]
```

Access the  $j$ -th column of the agent.

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

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

#### Parameters

$j$	
-----	--

#### Returns

`double&`

### 13.5.2.2 `swap_neighbors()`

```
template<typename TSeq >
void Agent< TSeq >::swap_neighbors (
 Agent< TSeq > & other,
 size_t n_this,
 size_t n_other) [inline]
```

Swaps neighbors between the current agent and agent `other`

#### Parameters

<i>other</i>	
<i>n_this</i>	
<i>n_other</i>	

## 13.5.3 Friends And Related Function Documentation

### 13.5.3.1 default\_rm\_entity

```
template<typename TSeq >
void default_rm_entity (
 Action< TSeq > & a,
 Model< TSeq > * m) [friend]
```

< Last entity of the agent

< Last agent of the entity

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

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

## 13.6 epiworld::Agent< TSeq > Class Template Reference

[Agent](#) (agents)

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

### Public Member Functions

- **Agent** ([Agent](#)< TSeq > &&p)
- **Agent** (const [Agent](#)< TSeq > &p)
- [Agent](#)< TSeq > & **operator=** (const [Agent](#)< TSeq > &other\_agent)
- int [get\\_id](#) () const  
*Id of the individual.*
- 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)
- void **swap\_neighbors** ([Agent](#)< TSeq > &other, size\_t n\_this, size\_t n\_other)  
*Swaps neighbors between the current agent and agent *other**
- std::vector< [Agent](#)< TSeq > \* > **get\_neighbors** ()
- size\_t **get\_n\_neighbors** () const
- void **change\_state** ([Model](#)< TSeq > \*model, epiworld\_fast\_uint new\_state, epiworld\_fast\_int queue=0)
- const epiworld\_fast\_uint & **get\_state** () const
- void **reset** ()
- bool **has\_tool** (epiworld\_fast\_uint t) const
- bool **has\_tool** (std::string name) const
- bool **has\_virus** (epiworld\_fast\_uint t) const
- bool **has\_virus** (std::string name) const
- void **print** ([Model](#)< TSeq > \*model, bool compressed=false) const

- **Entities**< TSeq > **get\_entities** ()  
Access the *j*-th column of the agent.
- const **Entities\_const**< TSeq > **get\_entities** () const
- const **Entity**< TSeq > & **get\_entity** (size\_t i) const
- **Entity**< TSeq > & **get\_entity** (size\_t i)
- size\_t **get\_n\_entities** () const
- bool **operator==** (const **Agent**< TSeq > &other) const
- bool **operator!=** (const **Agent**< TSeq > &other) 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>state after the change</i>
queue	

- void **add\_tool** (ToolPtr< TSeq > tool, **Model**< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **add\_tool** (**Tool**< TSeq > tool, **Model**< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **add\_virus** (VirusPtr< TSeq > virus, **Model**< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **add\_virus** (**Virus**< TSeq > virus, **Model**< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **add\_entity** (**Entity**< TSeq > &entity, **Model**< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_tool** (epiworld\_fast\_uint tool\_idx, **Model**< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_tool** (ToolPtr< TSeq > &tool, **Model**< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_virus** (epiworld\_fast\_uint virus\_idx, **Model**< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_virus** (VirusPtr< TSeq > &virus, **Model**< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_entity** (epiworld\_fast\_uint entity\_idx, **Model**< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_entity** (**Entity**< TSeq > &entity, **Model**< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_agent\_by\_virus** (epiworld\_fast\_uint virus\_idx, **Model**< TSeq > \*model, epiworld\_fast\_int status\_new=-99, epiworld\_fast\_int queue=-99)  
*Agent removed by virus.*
- void **rm\_agent\_by\_virus** (VirusPtr< TSeq > &virus, **Model**< TSeq > \*model, 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, [Model](#)< TSeq > \*model)
- epiworld\_double **get\_transmission\_reduction** (VirusPtr< TSeq > v, [Model](#)< TSeq > \*model)
- epiworld\_double **get\_recovery\_enhancer** (VirusPtr< TSeq > v, [Model](#)< TSeq > \*model)
- epiworld\_double **get\_death\_reduction** (VirusPtr< TSeq > v, [Model](#)< TSeq > \*model)

## Friends

- class **Model**< TSeq >
- class **Virus**< TSeq >
- class **Viruses**< TSeq >
- class **Viruses\_const**< TSeq >
- class **Tool**< TSeq >
- class **Tools**< TSeq >
- class **Tools\_const**< 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)

## 13.6.1 Detailed Description

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

[Agent](#) (agents)

Template Parameters

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

## 13.6.2 Member Function Documentation

### 13.6.2.1 get\_entities()

```
template<typename TSeq >
Entities< TSeq > Agent< TSeq >::get_entities [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&

### 13.6.2.2 swap\_neighbors()

```
template<typename TSeq >
void Agent< TSeq >::swap_neighbors (
 Agent< TSeq > & other,
 size_t n_this,
 size_t n_other) [inline]
```

Swaps neighbors between the current agent and agent `other`

#### Parameters

<i>other</i>	
<i>n_this</i>	
<i>n_other</i>	

## 13.6.3 Friends And Related Function Documentation

### 13.6.3.1 default\_rm\_entity

```
template<typename TSeq >
void default_rm_entity (
 Action< TSeq > & a,
 Model< TSeq > * m) [friend]
```

< Last entity of the agent

< Last agent of the entity

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

- epiworld.hpp

## 13.7 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](#) ([Model](#)< TSeq > \*model, [Entity](#)< TSeq > &entity\_, size\_t n, bool truncate=false)
- [AgentsSample](#) ([Model](#)< TSeq > \*model, [Agent](#)< TSeq > &agent\_, size\_t n, bool truncate=false)  
*Sample from the agent's entities.*
- std::vector< [Agent](#)< TSeq > \* >::iterator **begin** ()
- std::vector< [Agent](#)< TSeq > \* >::iterator **end** ()
- [Agent](#)< TSeq > \* **operator[]** (size\_t n)
- [Agent](#)< TSeq > \* **operator()** (size\_t n)
- size\_t **size** () const noexcept

### 13.7.1 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>	
-------------	--

### 13.7.2 Constructor & Destructor Documentation

#### 13.7.2.1 AgentsSample()

```
template<typename TSeq >
AgentsSample< TSeq >::AgentsSample (
 Model< TSeq > * model,
```

```

 Agent< TSeq > & agent_,
 size_t n,
 bool truncate = false) [inline]

```

Sample from the agent's entities.

For example, how many individuals the agent contacts in a given point in time.

#### Template Parameters

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

#### Parameters

<i>agent</i> ↔	
—	
<i>n</i>	Sample size
<i>truncate</i>	If the agent has fewer than <i>n</i> connections, then <i>truncate</i> = true will automatically reduce the number of possible samples. Otherwise, if false, then it returns an error.

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

- include/epiworld/agentssample-bones.hpp

## 13.8 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** ([Model](#)< TSeq > \*model, [Entity](#)< TSeq > &entity\_, size\_t n, bool truncate=false)
- [AgentsSample](#) ([Model](#)< TSeq > \*model, [Agent](#)< TSeq > &agent\_, size\_t n, bool truncate=false)  
*Sample from the agent's entities.*
- std::vector< [Agent](#)< TSeq > \* >::iterator **begin** ()
- std::vector< [Agent](#)< TSeq > \* >::iterator **end** ()
- [Agent](#)< TSeq > \* **operator[]** (size\_t n)
- [Agent](#)< TSeq > \* **operator()** (size\_t n)
- size\_t **size** () const noexcept

### 13.8.1 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>	
-------------	--

### 13.8.2 Constructor & Destructor Documentation

#### 13.8.2.1 AgentsSample()

```
template<typename TSeq >
AgentsSample< TSeq >::AgentsSample (
 Model< TSeq > * model,
 Agent< TSeq > & agent_,
 size_t n,
 bool truncate = false) [inline]
```

Sample from the agent's entities.

For example, how many individuals the agent contacts in a given point in time.

Template Parameters

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

Parameters

<i>agent_</i>	
<i>n</i>	Sample size
<i>truncate</i>	If the agent has fewer than <i>n</i> connections, then <i>truncate</i> = true will automatically reduce the number of possible samples. Otherwise, if false, then it returns an error.

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

- epiworld.hpp

## 13.9 DataBase< TSeq > Class Template Reference

Statistical data about the process.

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

### Public Member Functions

- **DataBase** ([Model](#)< TSeq > &m)
- **DataBase** (const [DataBase](#)< TSeq > &db)
- 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 [reset](#) ()
- [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, std::string fn\_generation\_time) 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 [set\\_user\\_data](#) (std::vector< std::string > names)
- void [add\\_user\\_data](#) (std::vector< epiworld\_double > x)
- void [add\\_user\\_data](#) (epiworld\_fast\_uint j, epiworld\_double x)
- [UserData](#)< TSeq > & [get\\_user\\_data](#) ()
- std::vector< epiworld\_double > [transition\\_probability](#) (bool print=true) const
  - Calculates the transition probabilities.*
- bool [operator==](#) (const [DataBase](#)< TSeq > &other) const
- bool [operator!=](#) (const [DataBase](#)< TSeq > &other) const
- bool [operator==](#) (const [DataBase](#)< std::vector< int >> &other) const
- bool [operator==](#) (const [DataBase](#)< std::vector< int >> &other) const

### Get recorded information from the model

#### Parameters

what	<i>std::string, The state, 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 > \*state=nullptr, std::vector< int > \*counts=nullptr) const
  - void **get\_today\_variant** (std::vector< std::string > &state, std::vector< int > &id, std::vector< int > &counts) const
  - void **get\_hist\_total** (std::vector< int > \*date, std::vector< std::string > \*state, std::vector< int > \*counts) const
  - void **get\_hist\_variant** (std::vector< int > &date, std::vector< int > &id, std::vector< std::string > &state, std::vector< int > &counts) const
  - void **get\_hist\_tool** (std::vector< int > &date, std::vector< int > &id, std::vector< std::string > &state, std::vector< int > &counts) const
  - void **get\_hist\_transition\_matrix** (std::vector< std::string > &state\_from, std::vector< std::string > &state\_to, std::vector< int > &date, std::vector< int > &counts, bool skip\_zeros) const
- 
- MapVec\_type< int, int > **reproductive\_number** () const  
*Computes the reproductive number of each case.*
  - void **reproductive\_number** (std::string fn) const
- 
- void **generation\_time** (std::vector< int > &agent\_id, std::vector< int > &virus\_id, std::vector< int > &time, std::vector< int > &gentime) const
  - void **generation\_time** (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)

### 13.9.1 Detailed Description

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

Statistical data about the process.

Template Parameters

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

### 13.9.2 Member Function Documentation

### 13.9.2.1 generation\_time()

```
template<typename TSeq >
void DataBase< TSeq >::generation_time (
 std::vector< int > & agent_id,
 std::vector< int > & virus_id,
 std::vector< int > & time,
 std::vector< int > & gentime) const [inline]
```

Calculates the generating time

#### Parameters

<i>agent_id, virus_id, time, gentime</i>	vectors where to save the values agent_id
------------------------------------------	-------------------------------------------

### 13.9.2.2 operator==( ) [1/3]

```
bool DataBase< std::vector< int > >::operator== (
 const DataBase< std::vector< int >> & other) const [inline]
```

< Date of the transmission eve,

< Id of the sour,

< Id of the targ,

< Id of the varia,

< Date when the source acquired the varia,

### 13.9.2.3 operator==( ) [2/3]

```
bool DataBase< std::vector< int > >::operator== (
 const DataBase< std::vector< int >> & other) const [inline]
```

< Date of the transmission eve,

< Id of the sour,

< Id of the targ,

< Id of the varia,

< Date when the source acquired the varia,



**13.9.2.4 operator==( ) [3/3]**

```
template<typename TSeq >
bool DataBase< TSeq >::operator==(
 const DataBase< TSeq > & other) const [inline]
```

< Date of the transmission eve

< Id of the sour

< Id of the targ

< Id of the varia

< Date when the source acquired the varia

**13.9.2.5 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.
----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**13.9.2.6 reproductive\_number()**

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

Computes the reproductive number of each case.

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

**Parameters**

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

**13.9.2.7 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

## 13.10 epiworld::DataBase< TSeq > Class Template Reference

Statistical data about the process.

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

### Public Member Functions

- **DataBase** ([Model](#)< TSeq > &m)
- **DataBase** (const [DataBase](#)< TSeq > &db)
- 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 [reset](#) ()
- [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, std::string fn\_generation\_time) 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 [set\\_user\\_data](#) (std::vector< std::string > names)
- void [add\\_user\\_data](#) (std::vector< epiworld\_double > x)
- void [add\\_user\\_data](#) (epiworld\_fast\_uint j, epiworld\_double x)
- [UserData](#)< TSeq > & [get\\_user\\_data](#) ()
- std::vector< epiworld\_double > [transition\\_probability](#) (bool print=true) const
 

*Calculates the transition probabilities.*
- bool [operator==](#) (const [DataBase](#)< TSeq > &other) const
- bool [operator!=](#) (const [DataBase](#)< TSeq > &other) const

**Get recorded information from the model**

### Parameters

what	<i>std::string, The state, 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 > *state=nullptr, std::vector< int > *counts=nullptr) const`
- `void get_today_variant (std::vector< std::string > &state, std::vector< int > &id, std::vector< int > &counts) const`
- `void get_hist_total (std::vector< int > *date, std::vector< std::string > *state, std::vector< int > *counts) const`
- `void get_hist_variant (std::vector< int > &date, std::vector< int > &id, std::vector< std::string > &state, std::vector< int > &counts) const`
- `void get_hist_tool (std::vector< int > &date, std::vector< int > &id, std::vector< std::string > &state, std::vector< int > &counts) const`
- `void get_hist_transition_matrix (std::vector< std::string > &state_from, std::vector< std::string > &state_to, std::vector< int > &date, std::vector< int > &counts, bool skip_zeros) const`
- `MapVec_type< int, int > reproductive_number () const`  
*Computes the reproductive number of each case.*
- `void reproductive_number (std::string fn) const`
- `void generation_time (std::vector< int > &agent_id, std::vector< int > &virus_id, std::vector< int > &time, std::vector< int > &gentime) const`
- `void generation_time (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)`

## 13.10.1 Detailed Description

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

Statistical data about the process.

## Template Parameters

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

## 13.10.2 Member Function Documentation

### 13.10.2.1 generation\_time()

```
template<typename TSeq >
void DataBase< TSeq >::generation_time (
 std::vector< int > & agent_id,
 std::vector< int > & virus_id,
 std::vector< int > & time,
 std::vector< int > & gentime) const [inline]
```

Calculates the generating time

## Parameters

<i>agent_id, virus_id, time, gentime</i>	vectors where to save the values agent_id
------------------------------------------	-------------------------------------------

### 13.10.2.2 operator==()

```
template<typename TSeq >
bool DataBase< TSeq >::operator== (
 const DataBase< TSeq > & other) const [inline]
```

< Date of the transmission eve

< Id of the sour

< Id of the targ

< Id of the varia

< Date when the source acquired the varia

### 13.10.2.3 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.
----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**13.10.2.4 reproductive\_number()**

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

Computes the reproductive number of each case.

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

## Parameters

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

**13.10.2.5 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

**13.11 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
- bool **operator==** (const [Entities](#)< TSeq > &other) const

## Friends

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

### 13.11.1 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

## 13.12 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
- bool **operator==** (const [Entities](#)< TSeq > &other) const

## Friends

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

### 13.12.1 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

## 13.13 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
- bool **operator==** (const [Entities\\_const](#)< TSeq > &other) const

## Friends

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

### 13.13.1 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

## 13.14 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
- bool **operator==** (const [Entities\\_const](#)< TSeq > &other) const

### Friends

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

#### 13.14.1 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



## 13.15 Entity< TSeq > Class Template Reference

### Public Member Functions

- **Entity** (std::string name)
- void **add\_agent** (Agent< TSeq > &p, Model< TSeq > \*model)
- void **add\_agent** (Agent< TSeq > \*p, Model< TSeq > \*model)
- 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\_state** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **set\_queue** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **get\_state** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- void **get\_queue** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- void **reset** ()
- bool **operator==** (const Entity< TSeq > &other) const
- bool **operator!=** (const Entity< TSeq > &other) const

### 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)

### 13.15.1 Friends And Related Function Documentation

#### 13.15.1.1 default\_rm\_entity

```
template<typename TSeq >
void default_rm_entity (
 Action< TSeq > & a,
 Model< TSeq > * m) [friend]
```

< Last entity of the agent

< Last agent of the entity

< Last entity of the agent

< Last agent of the entity

< Last entity of the agent

< Last agent of the entity

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

## 13.16 epiworld::Entity< TSeq > Class Template Reference

### Public Member Functions

- **Entity** (std::string name)
- void **add\_agent** ([Agent](#)< TSeq > &p, [Model](#)< TSeq > \*model)
- void **add\_agent** ([Agent](#)< TSeq > \*p, [Model](#)< TSeq > \*model)
- 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\_state** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **set\_queue** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **get\_state** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- void **get\_queue** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- void **reset** ()
- bool **operator==** (const [Entity](#)< TSeq > &other) const
- bool **operator!=** (const [Entity](#)< TSeq > &other) const

### 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)

### 13.16.1 Friends And Related Function Documentation

#### 13.16.1.1 default\_rm\_entity

```
template<typename TSeq >
void default_rm_entity (
 Action< TSeq > & a,
 Model< TSeq > * m) [friend]
```

< Last entity of the agent

< Last agent of the entity

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

- epiworld.hpp

## 13.17 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)
- size\_t **get\_n\_samples** () const
- size\_t **get\_n\_statistics** () const
- size\_t **get\_n\_parameters** () const
- epiworld\_double **get\_epsilon** () const
- 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)
- std::vector< epiworld\_double > **get\_params\_mean** ()
- std::vector< epiworld\_double > **get\_stats\_mean** ()
- void **print** ()

### Random number generation

#### Parameters

eng	
-----	--

- void **set\_rand\_engine** (std::mt19937 &eng)
- std::mt19937 & **get\_rand\_engine** ()
- void **seed** (epiworld\_fast\_uint 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)

### 13.17.1 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

## 13.18 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)
- size\_t **get\_n\_samples** () const
- size\_t **get\_n\_statistics** () const
- size\_t **get\_n\_parameters** () const
- epiworld\_double **get\_epsilon** () const
- 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)
- std::vector< epiworld\_double > **get\_params\_mean** ()
- std::vector< epiworld\_double > **get\_stats\_mean** ()
- void **print** ()

### Random number generation

*Parameters*

eng	
-----	--

- void **set\_rand\_engine** (std::mt19937 &eng)
- std::mt19937 & **get\_rand\_engine** ()
- void **seed** (epiworld\_fast\_uint 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)

**13.18.1 Detailed Description**

```
template<typename TData>
class 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 files:

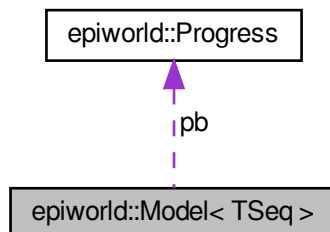
- include/epiworld/math/lfmcmc/lfmcmc-bones.hpp
- include/epiworld/math/lfmcmc/lfmcmc-meat-print.hpp
- include/epiworld/math/lfmcmc/lfmcmc-meat.hpp

**13.19 epiworld::Model< TSeq > Class Template Reference**

Core class of epiworld.

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

Collaboration diagram for epiworld::Model< TSeq >:



## Public Member Functions

- [DataBase](#)< TSeq > & **get\_db** ()
- `epiworld_double` & **operator()** (std::string pname)
- `size_t` **size** () const
- void [load\\_agents\\_entities\\_ties](#) (std::string fn, int skip)  
*Associate agents-entities from a file.*
- `size_t` **get\_n\_variants** () const
- `size_t` **get\_n\_tools** () const
- `epiworld_fast_uint` **get\_ndays** () const
- `epiworld_fast_uint` **get\_n\_replicates** () const
- void **set\_ndays** (`epiworld_fast_uint` 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, std::string fn\_generation\_time) const  
*Wrapper of DataBase::write\_data*
- std::map< std::string, `epiworld_double` > & **params** ()
- void [reset](#) ()  
*Reset the model.*
- void **print** (bool lite=false) 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\_state\_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.*
- void [set\\_name](#) (std::string name)  
*Set the name object.*
- std::string **get\_name** () const
- bool **operator==** (const [Model](#)< TSeq > &other) const
- bool **operator!=** (const [Model](#)< TSeq > &other) const

### 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** ()

### Random number generation

#### Parameters

eng	Random number generator
s	Seed

- void **set\_rand\_engine** (std::mt19937 &eng)
- std::mt19937 & **get\_rand\_engine** ()
- void **seed** (size\_t 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, epiworld\_fast\_uint 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, epiworld\_fast\_uint preval)
- void **add\_tool\_fun** ([Tool](#)< TSeq > t, ToolToAgentFun< TSeq > fun)
- void **add\_entity** ([Entity](#)< TSeq > e)
- void **rm\_virus** (size\_t virus\_pos)
- void **rm\_tool** (size\_t tool\_pos)
- void **rm\_entity** (size\_t entity\_pos)

### 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 <code>fn</code>.</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\_edgelist** (const std::vector< int > &source, const std::vector< int > &target, int size, bool directed)
- void **agents\_from\_adjlist** ([AdjList](#) al)
- bool **is\_directed** () const
- std::vector< [Agent](#)< TSeq > > & **get\_agents** ()

- `std::vector< Entity < TSeq > > & get_entities ()`
- `void agents_smallworld (epiworld_fast_uint n=1000, epiworld_fast_uint k=5, bool d=false, epiworld_double p=.01)`
- `void agents_empty_graph (epiworld_fast_uint n=1000)`

### 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 <code>run_multiple</code>, a function that is called after each experiment.</i>

- `void update_state ()`
- `void mutate_variant ()`
- `void next ()`
- virtual `void run (epiworld_fast_uint ndays, int seed=-1)`  
*Runs the simulation (after initialization)*
- `void run_multiple (epiworld_fast_uint ndays, epiworld_fast_uint nexperiments, int seed_=-1, std::function< void(size_t, Model< TSeq > *)> fun=make_save_run< TSeq >(), bool reset=true, bool verbose=true, int nthreads=1)`

### 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. File name.</i>
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< epiworld_fast_uint > &source, std::vector< epiworld_fast_uint > &target) const`



**Manage state (states) in the model**

The functions `get_state` return the current values for the states included in the model.

**Parameters**

lab	<code>std::string</code> Name of the state.
-----	---------------------------------------------

**Returns**

`add_state*` returns nothing.

`get_state_*` returns a vector of pairs with the states and their labels.

- void **add\_state** (std::string lab, UpdateFun< TSeq > fun=nullptr)
- const std::vector< std::string > & **get\_state** () const
- const std::vector< UpdateFun< TSeq > > & **get\_state\_fun** () const
- void **print\_state\_codes** () const

**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 `epiworld_fast_uint` 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**

initial_val	
pname	Name of the parameter to add or to fetch
fn	Path to the file containing parameters

**Returns**

The current value of the parameter in the model.

- epiworld\_double **add\_param** (epiworld\_double initial\_val, std::string pname)
- void **read\_params** (std::string fn)
- epiworld\_double **get\_param** (epiworld\_fast\_uint k)
- epiworld\_double **get\_param** (std::string pname)
- epiworld\_double **par** (epiworld\_fast\_uint k)
- epiworld\_double **par** (std::string pname)

**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** (epiworld\_fast\_uint j, epiworld\_double x)
- void **add\_user\_data** (std::vector< epiworld\_double > x)
- [UserData](#)< TSeq > & **get\_user\_data** ()

### Queuing system

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

- void [queuing\\_on](#) ()  
*Activates the queuing system (default.)*
- void [queuing\\_off](#) ()  
*Deactivates the queuing system.*
- bool [is\\_queuing\\_on](#) () const  
*Query if the queuing system is on.*
- [Queue](#)< TSeq > & [get\\_queue](#) ()  
*Retrieve the [Queue](#) object.*

### Get the susceptibility reduction object

#### Parameters

v	
---	--

#### Returns

*epiworld\_double*

- void **set\_susceptibility\_reduction\_mixer** (MixerFun< TSeq > fun)
- void **set\_transmission\_reduction\_mixer** (MixerFun< TSeq > fun)
- void **set\_recovery\_enhancer\_mixer** (MixerFun< TSeq > fun)
- void **set\_death\_reduction\_mixer** (MixerFun< TSeq > fun)

## Protected Member Functions

- void **dist\_tools** ()
- void **dist\_virus** ()
- void **chrono\_start** ()
- void **chrono\_end** ()
- void [actions\\_add](#) ([Agent](#)< TSeq > \*agent\_, [VirusPtr](#)< TSeq > virus\_, [ToolPtr](#)< TSeq > tool\_, [Entity](#)< TSeq > \*entity\_, epiworld\_fast\_uint new\_state\_, epiworld\_fast\_int queue\_, [ActionFun](#)< TSeq > call\_, int idx\_agent\_, int idx\_object\_)  
*Construct a new [Action](#) object.*
- void [actions\\_run](#) ()  
*Executes the stored action.*

## Protected Attributes

- std::string [name](#) = ""  
*Name of the model.*
- [DataBase](#)< TSeq > **db** = [DataBase](#)<TSeq>(\*this)
- std::vector< [Agent](#)< TSeq > > **population** = {}
- bool **using\_backup** = true
- std::vector< [Agent](#)< TSeq > > **population\_backup** = {}

- bool **directed** = false
- std::vector< VirusPtr< TSeq > > **viruses** = {}
- std::vector< epiworld\_double > **prevalence\_virus** = {}  
*Initial prevalence\_virus of each virus.*
- std::vector< bool > **prevalence\_virus\_as\_proportion** = {}
- std::vector< VirusToAgentFun< TSeq > > **viruses\_dist\_funs** = {}
- std::vector< ToolPtr< TSeq > > **tools** = {}
- std::vector< epiworld\_double > **prevalence\_tool** = {}
- std::vector< bool > **prevalence\_tool\_as\_proportion** = {}
- std::vector< ToolToAgentFun< TSeq > > **tools\_dist\_funs** = {}
- std::vector< Entity< TSeq > > **entities** = {}
- std::vector< Entity< TSeq > > **entities\_backup** = {}
- std::mt19937 **engine**
- std::uniform\_real\_distribution **runifd** = std::uniform\_real\_distribution<> (0.0, 1.0)
- std::normal\_distribution **rnormd** = std::normal\_distribution<> (0.0)
- std::gamma\_distribution **rgammad** = std::gamma\_distribution<> ()
- std::lognormal\_distribution **rlognormald** = std::lognormal\_distribution<> ()
- std::exponential\_distribution **rexp** = std::exponential\_distribution<> ()
- std::function< void(std::vector< Agent< TSeq > > \*, Model< TSeq > \*, epiworld\_double)> **rewire\_fun**
- epiworld\_double **rewire\_prop** = 0.0
- std::map< std::string, epiworld\_double > **parameters**
- epiworld\_fast\_uint **ndays** = 0
- **Progress pb**
- std::vector< UpdateFun< TSeq > > **status\_fun** = {}
- std::vector< std::string > **status\_labels** = {}
- epiworld\_fast\_uint **nstatus** = 0u
- bool **verbose** = true
- int **current\_date** = 0
- std::chrono::time\_point< std::chrono::steady\_clock > **time\_start**
- std::chrono::time\_point< std::chrono::steady\_clock > **time\_end**
- std::chrono::duration< epiworld\_double, std::micro > **time\_elapsed**
- epiworld\_fast\_uint **n\_replicates** = 0u
- std::vector< std::function< void(Model< TSeq > \*)> > **global\_action\_functions**
- std::vector< int > **global\_action\_dates**
- **Queue< TSeq > queue**
- bool **use\_queueing** = true
- std::vector< Action< TSeq > > **actions** = {}  
*Variables used to keep track of the actions to be made regarding viruses.*
- epiworld\_fast\_uint **nactions** = 0u

#### Auxiliary variables for AgentsSample<TSeq> iterators

These variables+objects are used by the AgentsSample< TSeq > class for building efficient iterators over agents. The idea is to reduce the memory allocation, so only during the first call of AgentsSample< TSeq >::AgentsSample(Model< TSeq >) these vectors are allocated.

- std::vector< Agent< TSeq > \* > **sampled\_population**
- size\_t **sampled\_population\_n** = 0u
- std::vector< size\_t > **population\_left**
- size\_t **population\_left\_n** = 0u

#### Agents features

Optionally, a model can include an external data source pointing to agents information. The data can then be access through the Agent::operator() method.

- double \* **population\_data** = nullptr
- size\_t **population\_data\_n\_features** = 0u

## 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.

- MixerFun< TSeq > **susceptibility\_reduction\_mixer** = susceptibility\_reduction\_mixer\_default<TSeq>
  - MixerFun< TSeq > **transmission\_reduction\_mixer** = transmission\_reduction\_mixer\_default<TSeq>
  - MixerFun< TSeq > **recovery\_enhancer\_mixer** = recovery\_enhancer\_mixer\_default<TSeq>
  - MixerFun< TSeq > **death\_reduction\_mixer** = death\_reduction\_mixer\_default<TSeq>
  - std::vector< epiworld\_double > **array\_double\_tmp**
  - std::vector< [Virus](#)< TSeq > \* > **array\_virus\_tmp**
  - virtual [Model](#)< TSeq > \* **clone\_ptr** ()
- Advanced usage: Makes a copy of data and returns it as undeleted pointer.*
- **Model** ()
  - **Model** (const [Model](#)< TSeq > &m)
  - **Model** ([Model](#)< TSeq > &m)=delete
  - **Model** ([Model](#)< TSeq > &&m)
  - [Model](#)< TSeq > & **operator=** (const [Model](#)< TSeq > &m)
  - virtual ~**Model** ()
  - void **clone\_population** (std::vector< [Agent](#)< TSeq > > &other\_population, std::vector< [Entity](#)< TSeq > > &other\_entities, [Model](#)< TSeq > \*other\_model, bool &other\_directed) const
  - void **clone\_population** (const [Model](#)< TSeq > &other\_model)

### 13.19.1 Detailed Description

```
template<typename TSeq>
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.)
-------------	-----------------------------------------------------------------------------------------------------------------------------------------

### 13.19.2 Member Function Documentation

### 13.19.2.1 actions\_add()

```
template<typename TSeq >
void Model< TSeq >::actions_add (
 Agent< TSeq > * agent_,
 VirusPtr< TSeq > virus_,
 ToolPtr< TSeq > tool_,
 Entity< TSeq > * entity_,
 epiworld_fast_uint new_state_,
 epiworld_fast_int queue_,
 ActionFun< TSeq > call_,
 int idx_agent_,
 int idx_object_) [inline], [protected]
```

Construct a new [Action](#) object.

#### Parameters

<i>agent_</i>	<a href="#">Agent</a> over which the action will be called
<i>virus_</i>	<a href="#">Virus</a> pointer included in the action
<i>tool_</i>	<a href="#">Tool</a> pointer included in the action
<i>entity_</i>	<a href="#">Entity</a> pointer included in the action
<i>new_↔ state_</i>	New state of the agent
<i>call_</i>	Function the action will call
<i>queue_</i>	Change in the queue
<i>idx_↔ agent_</i>	Location of agent in object.
<i>idx_↔ object_</i>	Location of object in agent.

### 13.19.2.2 actions\_run()

```
template<typename TSeq >
void Model< TSeq >::actions_run [inline], [protected]
```

Executes the stored action.

#### Parameters

<i>model_↔ _</i>	<a href="#">Model</a> over which it will be executed.
----------------------	-------------------------------------------------------

### 13.19.2.3 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.

### 13.19.2.4 clone\_ptr()

```
template<typename TSeq >
Model< TSeq > * Model< TSeq >::clone_ptr [inline], [protected], [virtual]
```

Advanced usage: Makes a copy of data and returns it as undeleted pointer.

#### Parameters

<i>copy</i>	
-------------	--

Reimplemented in [ModelSIRCONN< TSeq >](#), [ModelSEIRCONN< TSeq >](#), [epiworld::epimodels::ModelSEIRCONN< TSeq >](#), and [epiworld::epimodels::ModelSIRCONN< TSeq >](#).

### 13.19.2.5 load\_agents\_entities\_ties()

```
template<typename TSeq >
void Model< TSeq >::load_agents_entities_ties (
 std::string fn,
 int skip) [inline]
```

Associate agents-entities from a file.

The structure of the file should be two columns separated by space. The first column indexing between 0 and nagents-1, and the second column between 0 and nentities - 1.

#### Parameters

<i>fn</i>	Path to the file.
<i>skip</i>	How many rows to skip.

### 13.19.2.6 reset()

```
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

### 13.19.2.7 run\_multiple()

```
template<typename TSeq >
void Model< TSeq >::run_multiple (
 epiworld_fast_uint ndays,
 epiworld_fast_uint nexperiments,
 int seed_ = -1,
 std::function< void(size_t, Model< TSeq > *)> fun = make_save_run<TSeq>(),
 bool reset = true,
 bool verbose = true,
 int nthreads = 1) [inline]
```

#### Parameters

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

### 13.19.2.8 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.

**13.19.2.9 set\_name()**

```
template<typename TSeq >
void Model< TSeq >::set_name (
 std::string name) [inline]
```

Set the name object.

## Parameters

<i>name</i>	
-------------	--

**13.19.2.10 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,
 std::string fn_generation_time) 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 (state)
<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



### 13.19.3 Member Data Documentation

#### 13.19.3.1 time\_elapsed

```
template<typename TSeq >
std::chrono::duration<epiworld_double, std::micro> epiworld::Model< TSeq >::time_elapsed [protected]
```

##### Initial value:

```
=
 std::chrono::duration<epiworld_double, std::micro>::zero()
```

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

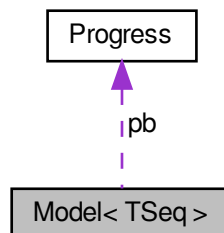
- epiworld.hpp

## 13.20 Model< TSeq > Class Template Reference

Core class of epiworld.

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

Collaboration diagram for Model< TSeq >:



### Public Member Functions

- [DataBase](#)< TSeq > & **get\_db** ()
- epiworld\_double & **operator()** (std::string pname)
- size\_t **size** () const
- void [load\\_agents\\_entities\\_ties](#) (std::string fn, int skip)  
*Associate agents-entities from a file.*
- size\_t **get\_n\_variants** () const
- size\_t **get\_n\_tools** () const
- epiworld\_fast\_uint **get\_ndays** () const
- epiworld\_fast\_uint **get\_n\_replicates** () const

- void **set\_ndays** (epiworld\_fast\_uint ndays)
- bool **get\_verbos** () 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, std::string fn\_generation\_time) const

*Wrapper of DataBase::write\_data*

- std::map< std::string, epiworld\_double > & **params** ()
- void **reset** ()

*Reset the model.*

- void **print** (bool lite=false) 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\_state\_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.*

- void **set\_name** (std::string name)

*Set the name object.*

- std::string **get\_name** () const
- bool **operator==** (const **Model**< TSeq > &other) const
- bool **operator!=** (const **Model**< TSeq > &other) const

### 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** ()

### Random number generation

#### Parameters

eng	Random number generator
s	Seed

- void **set\_rand\_engine** (std::mt19937 &eng)
- std::mt19937 & **get\_rand\_engine** ()
- void **seed** (size\_t 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

<code>v</code>	<i>Virus to be added</i>
<code>t</code>	<i>Tool to be added</i>
<code>preval</code>	<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`, `epiworld_fast_uint 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`, `epiworld_fast_uint preval`)
- `void` **add\_tool\_fun** (`Tool< TSeq > t`, `ToolToAgentFun< TSeq > fun`)
- `void` **add\_entity** (`Entity< TSeq > e`)
- `void` **rm\_virus** (`size_t virus_pos`)
- `void` **rm\_tool** (`size_t tool_pos`)
- `void` **rm\_entity** (`size_t entity_pos`)

### Accessing population of the model

#### Parameters

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

- `void` **agents\_from\_adjlist** (`std::string fn`, `int size`, `int skip=0`, `bool directed=false`)
- `void` **agents\_from\_edgelist** (`const std::vector< int > &source`, `const std::vector< int > &target`, `int size`, `bool directed`)
- `void` **agents\_from\_adjlist** (`AdjList al`)
- `bool` **is\_directed** () `const`
- `std::vector< Agent< TSeq > > &` **get\_agents** ()
- `std::vector< Entity< TSeq > > &` **get\_entities** ()
- `void` **agents\_smallworld** (`epiworld_fast_uint n=1000`, `epiworld_fast_uint k=5`, `bool d=false`, `epiworld_double p=.01`)
- `void` **agents\_empty\_graph** (`epiworld_fast_uint n=1000`)

### 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 <code>run_multiple</code>, a function that is called after each experiment.</i>

- void **update\_state** ()
- void **mutate\_variant** ()
- void **next** ()
- virtual void **run** (epiworld\_fast\_uint ndays, int seed=-1)  
*Runs the simulation (after initialization)*
- void **run\_multiple** (epiworld\_fast\_uint ndays, epiworld\_fast\_uint nexperiments, int seed\_=-1, std::function< void(size\_t, [Model](#)< TSeq > \*)> fun=make\_save\_run< TSeq >(), bool **reset**=true, bool verbose=true, int nthreads=1)

**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. File name.</i>
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< epiworld\_fast\_uint > &source, std::vector< epiworld\_fast\_uint > &target) const

**Manage state (states) in the model**

*The functions `get_state` return the current values for the states included in the model.*

**Parameters**

lab	<i>std::string Name of the state.</i>
-----	---------------------------------------

**Returns**

`add_state*` returns nothing.

`get_state_*` returns a vector of pairs with the states and their labels.

- void **add\_state** (std::string lab, UpdateFun< TSeq > fun=nullptr)
- const std::vector< std::string > & **get\_state** () const
- const std::vector< UpdateFun< TSeq > > & **get\_state\_fun** () const
- void **print\_state\_codes** () const

**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 `epiworld_fast_uint` 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**

initial_val	
pname	Name of the parameter to add or to fetch
fn	Path to the file containing parameters

**Returns**

The current value of the parameter in the model.

- `epiworld_double` **add\_param** (`epiworld_double` initial\_val, `std::string` pname)
- void **read\_params** (`std::string` fn)
- `epiworld_double` **get\_param** (`epiworld_fast_uint` k)
- `epiworld_double` **get\_param** (`std::string` pname)
- `epiworld_double` **par** (`epiworld_fast_uint` k)
- `epiworld_double` **par** (`std::string` pname)

**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** (`epiworld_fast_uint` j, `epiworld_double` x)
- void **add\_user\_data** (`std::vector< epiworld_double >` x)
- `UserData< TSeq >` & **get\_user\_data** ()

**Queuing system**

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

- void [queuing\\_on](#) ()  
*Activates the queuing system (default.)*
- void [queuing\\_off](#) ()  
*Deactivates the queuing system.*
- bool [is\\_queuing\\_on](#) () const  
*Query if the queuing system is on.*
- [Queue](#)< TSeq > & [get\\_queue](#) ()  
*Retrieve the [Queue](#) object.*

### Get the susceptibility reduction object

#### Parameters

v	
---	--

#### Returns

*epiworld\_double*

- void **set\_susceptibility\_reduction\_mixer** (MixerFun< TSeq > fun)
- void **set\_transmission\_reduction\_mixer** (MixerFun< TSeq > fun)
- void **set\_recovery\_enhancer\_mixer** (MixerFun< TSeq > fun)
- void **set\_death\_reduction\_mixer** (MixerFun< TSeq > fun)

### Protected Member Functions

- void **dist\_tools** ()
- void **dist\_virus** ()
- void **chrono\_start** ()
- void **chrono\_end** ()
- void [actions\\_add](#) ([Agent](#)< TSeq > \*agent\_, [VirusPtr](#)< TSeq > virus\_, [ToolPtr](#)< TSeq > tool\_, [Entity](#)< TSeq > \*entity\_, epiworld\_fast\_uint new\_state\_, epiworld\_fast\_int queue\_, [ActionFun](#)< TSeq > call\_, int idx\_agent\_, int idx\_object\_)  
*Construct a new [Action](#) object.*
- void [actions\\_run](#) ()  
*Executes the stored action.*

### Protected Attributes

- std::string **name** = ""  
*Name of the model.*
- [DataBase](#)< TSeq > **db** = [DataBase](#)<TSeq>(\*this)
- std::vector< [Agent](#)< TSeq > > **population** = {}
- bool **using\_backup** = true
- std::vector< [Agent](#)< TSeq > > **population\_backup** = {}
- bool **directed** = false
- std::vector< [VirusPtr](#)< TSeq > > **viruses** = {}
- std::vector< epiworld\_double > [prevalence\\_virus](#) = {}  
*Initial prevalence\_virus of each virus.*
- std::vector< bool > **prevalence\_virus\_as\_proportion** = {}
- std::vector< [VirusToAgentFun](#)< TSeq > > **viruses\_dist\_funs** = {}
- std::vector< [ToolPtr](#)< TSeq > > **tools** = {}
- std::vector< epiworld\_double > **prevalence\_tool** = {}

- `std::vector< bool > prevalence_tool_as_proportion = {}`
  - `std::vector< ToolToAgentFun< TSeq > > tools_dist_funs = {}`
  - `std::vector< Entity< TSeq > > entities = {}`
  - `std::vector< Entity< TSeq > > entities_backup = {}`
  - `std::mt19937 engine`
  - `std::uniform_real_distribution runifd = std::uniform_real_distribution<> (0.0, 1.0)`
  - `std::normal_distribution rnormd = std::normal_distribution<>(0.0)`
  - `std::gamma_distribution rgammad = std::gamma_distribution<>()`
  - `std::lognormal_distribution rlognormald = std::lognormal_distribution<>()`
  - `std::exponential_distribution rexp = std::exponential_distribution<>()`
  - `std::function< void(std::vector< Agent< TSeq > > *, Model< TSeq > *, epiworld_double)> rewire_fun`
  - `epiworld_double rewire_prop = 0.0`
  - `std::map< std::string, epiworld_double > parameters`
  - `epiworld_fast_uint ndays = 0`
  - `Progress pb`
  - `std::vector< UpdateFun< TSeq > > status_fun = {}`
  - `std::vector< std::string > status_labels = {}`
  - `epiworld_fast_uint nstatus = 0u`
  - `bool verbose = true`
  - `int current_date = 0`
  - `std::chrono::time_point< std::chrono::steady_clock > time_start`
  - `std::chrono::time_point< std::chrono::steady_clock > time_end`
  - `std::chrono::duration< epiworld_double, std::micro > time_elapsed`
  - `epiworld_fast_uint n_replicates = 0u`
  - `std::vector< std::function< void(Model< TSeq > *)> > global_action_functions`
  - `std::vector< int > global_action_dates`
  - `Queue< TSeq > queue`
  - `bool use_queueing = true`
  - `std::vector< Action< TSeq > > actions = {}`
- Variables used to keep track of the actions to be made regarding viruses.*
- `epiworld_fast_uint nactions = 0u`

### Auxiliary variables for AgentsSample<TSeq> iterators

*These variables+objects are used by the AgentsSample< TSeq> class for building efficient iterators over agents. The idea is to reduce the memory allocation, so only during the first call of AgentsSample< TSeq>::AgentsSample(Model< TSeq>) these vectors are allocated.*

- `std::vector< Agent< TSeq > * > sampled_population`
- `size_t sampled_population_n = 0u`
- `std::vector< size_t > population_left`
- `size_t population_left_n = 0u`

### Agents features

*Optionally, a model can include an external data source pointing to agents information. The data can then be access through the Agent::operator() method.*

- `double * population_data = nullptr`
- `size_t population_data_n_features = 0u`

## 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.

- MixerFun< TSeq > **susceptibility\_reduction\_mixer** = susceptibility\_reduction\_mixer\_default<TSeq>
  - MixerFun< TSeq > **transmission\_reduction\_mixer** = transmission\_reduction\_mixer\_default<TSeq>
  - MixerFun< TSeq > **recovery\_enhancer\_mixer** = recovery\_enhancer\_mixer\_default<TSeq>
  - MixerFun< TSeq > **death\_reduction\_mixer** = death\_reduction\_mixer\_default<TSeq>
  - std::vector< epiworld\_double > **array\_double\_tmp**
  - std::vector< [Virus](#)< TSeq > \* > **array\_virus\_tmp**
  - virtual [Model](#)< TSeq > \* **clone\_ptr** ()
- Advanced usage: Makes a copy of data and returns it as undeleted pointer.*
- **Model** ()
  - **Model** (const [Model](#)< TSeq > &m)
  - **Model** ([Model](#)< TSeq > &m)=delete
  - **Model** ([Model](#)< TSeq > &&m)
  - [Model](#)< TSeq > & **operator=** (const [Model](#)< TSeq > &m)
  - virtual ~**Model** ()
  - void **clone\_population** (std::vector< [Agent](#)< TSeq > > &other\_population, std::vector< [Entity](#)< TSeq > > &other\_entities, [Model](#)< TSeq > \*other\_model, bool &other\_directed) const
  - void **clone\_population** (const [Model](#)< TSeq > &other\_model)

### 13.20.1 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.)
-------------	-----------------------------------------------------------------------------------------------------------------------------------------

### 13.20.2 Member Function Documentation

#### 13.20.2.1 actions\_add()

```
template<typename TSeq >
void Model< TSeq >::actions_add (
 Agent< TSeq > * agent_,
 VirusPtr< TSeq > virus_,
```



```

ToolPtr< TSeq > tool_,
Entity< TSeq > * entity_,
epiworld_fast_uint new_state_,
epiworld_fast_int queue_,
ActionFun< TSeq > call_,
int idx_agent_,
int idx_object_) [inline], [protected]

```

Construct a new [Action](#) object.

#### Parameters

<i>agent_</i>	<a href="#">Agent</a> over which the action will be called
<i>virus_</i>	<a href="#">Virus</a> pointer included in the action
<i>tool_</i>	<a href="#">Tool</a> pointer included in the action
<i>entity_</i>	<a href="#">Entity</a> pointer included in the action
<i>new_↔ state_</i>	New state of the agent
<i>call_</i>	Function the action will call
<i>queue_</i>	Change in the queue
<i>idx_↔ agent_</i>	Location of agent in object.
<i>idx_↔ object_</i>	Location of object in agent.

### 13.20.2.2 actions\_run()

```

template<typename TSeq >
void Model< TSeq >::actions_run [inline], [protected]

```

Executes the stored action.

#### Parameters

<i>model_↔ _</i>	<a href="#">Model</a> over which it will be executed.
----------------------	-------------------------------------------------------

### 13.20.2.3 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.

**13.20.2.4 clone\_ptr()**

```
template<typename TSeq >
Model< TSeq > * Model< TSeq >::clone_ptr [inline], [protected], [virtual]
```

Advanced usage: Makes a copy of data and returns it as undeleted pointer.

## Parameters

<i>copy</i>	
-------------	--

**13.20.2.5 load\_agents\_entities\_ties()**

```
template<typename TSeq >
void Model< TSeq >::load_agents_entities_ties (
 std::string fn,
 int skip) [inline]
```

Associate agents-entities from a file.

The structure of the file should be two columns separated by space. The first column indexing between 0 and nagents-1, and the second column between 0 and nentities - 1.

## Parameters

<i>fn</i>	Path to the file.
<i>skip</i>	How many rows to skip.

**13.20.2.6 reset()**

```
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

### 13.20.2.7 run\_multiple()

```
template<typename TSeq >
void Model< TSeq >::run_multiple (
 epiworld_fast_uint ndays,
 epiworld_fast_uint nexperiments,
 int seed_ = -1,
 std::function< void(size_t, Model< TSeq > *)> fun = make_save_run<TSeq>(),
 bool reset = true,
 bool verbose = true,
 int nthreads = 1) [inline]
```

#### Parameters

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

### 13.20.2.8 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.

### 13.20.2.9 set\_name()

```
template<typename TSeq >
void Model< TSeq >::set_name (
 std::string name) [inline]
```

Set the name object.

#### Parameters

<i>name</i>	
-------------	--

### 13.20.2.10 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,
 std::string fn_generation_time) 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 (state)
<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

## 13.20.3 Member Data Documentation

### 13.20.3.1 time\_elapsed

```
template<typename TSeq >
std::chrono::duration<epiworld_double, std::micro> Model< TSeq >::time_elapsed [protected]
```

**Initial value:**

```
=
 std::chrono::duration<epiworld_double, std::micro>::zero()
```

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

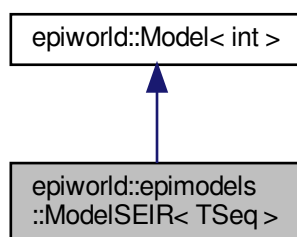
- include/epiworld/agent-bones.hpp
- include/epiworld/model-bones.hpp
- include/epiworld/model-meat-print.hpp
- include/epiworld/model-meat.hpp

## 13.21 epiworld::epimodels::ModelSEIR< TSeq > Class Template Reference

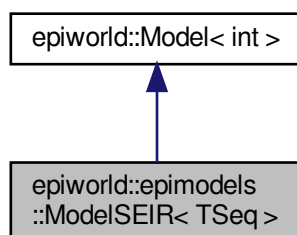
Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.

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

Inheritance diagram for epiworld::epimodels::ModelSEIR< TSeq >:



Collaboration diagram for epiworld::epimodels::ModelSEIR< TSeq >:



## Public Member Functions

- **ModelSEIR** ([ModelSEIR](#)< TSeq > &model, std::string vname, epiworld\_double prevalence, epiworld\_double infectiousness, epiworld\_double incubation\_days, epiworld\_double recovery)
- **ModelSEIR** (std::string vname, epiworld\_double prevalence, epiworld\_double infectiousness, epiworld\_double incubation\_days, epiworld\_double recovery)

## Public Attributes

- epiworld::UpdateFun< TSeq > **update\_exposed\_seir**
- epiworld::UpdateFun< TSeq > **update\_infected\_seir**

## Additional Inherited Members

### 13.21.1 Detailed Description

```
template<typename TSeq = int>
class epiworld::epimodels::ModelSEIR< TSeq >
```

Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.

#### Parameters

<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>initial_prevalence</i>	epiworld_double Initial prevalence
<i>initial_efficacy</i>	epiworld_double Initial susceptibility_reduction of the immune system
<i>initial_recovery</i>	epiworld_double Initial recovery rate of the immune system

### 13.21.2 Member Data Documentation

#### 13.21.2.1 update\_exposed\_seir

```
template<typename TSeq = int>
epiworld::UpdateFun<TSeq> epiworld::epimodels::ModelSEIR< TSeq >::update_exposed_seir
```

#### Initial value:

```
= [] (
 epiworld::Agent<TSeq> * p,
 epiworld::Model<TSeq> * m
) -> void {
 if (m->runif() < 1.0/(m->par("Incubation days")))
 p->change_state(m, ModelSEIR<TSeq>::INFECTED);
 return;
}
```

## 13.21.2.2 update\_infected\_seir

```
template<typename TSeq = int>
epiworld::UpdateFun<TSeq> epiworld::epimodels::ModelSEIR< TSeq >::update_infected_seir
```

## Initial value:

```
= [] (
 epiworld::Agent<TSeq> * p,
 epiworld::Model<TSeq> * m
) -> void {
 if (m->runif() < (m->par("Immune recovery")))
 p->rm_virus(0, m);
 return;
}
```

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

- epiworld.hpp

## 13.22 ModelSEIR&lt; TSeq &gt; Class Template Reference

Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.

```
#include <seir.hpp>
```

Inheritance diagram for ModelSEIR< TSeq >:



Collaboration diagram for ModelSEIR< TSeq >:



## Public Member Functions

- **ModelSEIR** ([ModelSEIR](#)< TSeq > &model, std::string vname, epiworld\_double prevalence, epiworld\_double infectiousness, epiworld\_double incubation\_days, epiworld\_double recovery)
- **ModelSEIR** (std::string vname, epiworld\_double prevalence, epiworld\_double infectiousness, epiworld\_double incubation\_days, epiworld\_double recovery)

## Public Attributes

- epiworld::UpdateFun< TSeq > **update\_exposed\_seir**
- epiworld::UpdateFun< TSeq > **update\_infected\_seir**

## Additional Inherited Members

### 13.22.1 Detailed Description

```
template<typename TSeq = int>
class ModelSEIR< TSeq >
```

Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.

#### Parameters

<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>initial_prevalence</i>	epiworld_double Initial prevalence
<i>initial_efficacy</i>	epiworld_double Initial susceptibility_reduction of the immune system
<i>initial_recovery</i>	epiworld_double Initial recovery rate of the immune system

### 13.22.2 Member Data Documentation

#### 13.22.2.1 update\_exposed\_seir

```
template<typename TSeq = int>
epiworld::UpdateFun<TSeq> ModelSEIR< TSeq >::update_exposed_seir
```

#### Initial value:

```
= [] (
 epiworld::Agent<TSeq> * p,
 epiworld::Model<TSeq> * m
) -> void {
 if (m->runif() < 1.0/(m->par("Incubation days")))
 p->change_state(m, ModelSEIR<TSeq>::INFECTED);
 return;
}
```



## 13.22.2.2 update\_infected\_seir

```
template<typename TSeq = int>
epiworld::UpdateFun<TSeq> ModelSEIR< TSeq >::update_infected_seir
```

## Initial value:

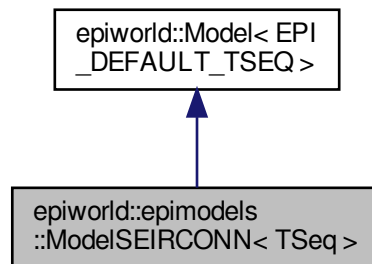
```
= [] (
 epiworld::Agent<TSeq> * p,
 epiworld::Model<TSeq> * m
) -> void {
 if (m->runif() < (m->par("Immune recovery")))
 p->rm_virus(0, m);
 return;
}
```

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

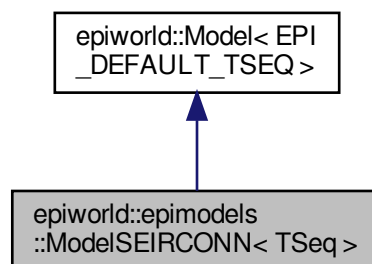
- include/epiworld/models/seir.hpp

## 13.23 epiworld::epimodels::ModelSEIRCONN< TSeq > Class Template Reference

Inheritance diagram for epiworld::epimodels::ModelSEIRCONN< TSeq >:



Collaboration diagram for epiworld::epimodels::ModelSEIRCONN< TSeq >:



## Public Member Functions

- [ModelSEIRCONN](#) ([ModelSEIRCONN](#)< TSeq > &model, std::string vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double prob\_transmission, epiworld\_double incubation\_days, epiworld\_double prob\_recovery)

*Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.*

- **ModelSEIRCONN** (std::string vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double prob\_transmission, epiworld\_double incubation\_days, epiworld\_double prob\_recovery)
- void [run](#) (epiworld\_fast\_uint ndays, int seed=-1)

*Runs the simulation (after initialization)*

- [Model](#)< TSeq > \* [clone\\_ptr](#) ()

*Advanced usage: Makes a copy of data and returns it as undeleted pointer.*

## Public Attributes

- std::vector< [epiworld::Agent](#)<> \* > **tracked\_agents\_infected** = {}
- std::vector< [epiworld::Agent](#)<> \* > **tracked\_agents\_infected\_next** = {}
- bool **tracked\_started** = false
- int **tracked\_ninfected** = 0
- int **tracked\_ninfected\_next** = 0

## Static Public Attributes

- static const int **SUSCEPTIBLE** = 0
- static const int **EXPOSED** = 1
- static const int **INFECTED** = 2
- static const int **RECOVERED** = 3

## Additional Inherited Members

### 13.23.1 Constructor & Destructor Documentation

#### 13.23.1.1 ModelSEIRCONN()

```
template<typename TSeq >
ModelSEIRCONN< TSeq >::ModelSEIRCONN (
 ModelSEIRCONN< TSeq > & model,
 std::string vname,
 epiworld_fast_uint n,
 epiworld_double prevalence,
 epiworld_double contact_rate,
 epiworld_double prob_transmission,
 epiworld_double incubation_days,
 epiworld_double prob_recovery) [inline]
```

Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.

## Parameters

<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>prevalence</i>	Initial prevalence (proportion)
<i>contact_rate</i>	Average number of contacts (interactions) per step.
<i>prob_transmission</i>	Probability of transmission
<i>prob_recovery</i>	Probability of recovery

## 13.23.2 Member Function Documentation

## 13.23.2.1 clone\_ptr()

```
template<typename TSeq >
Model< TSeq > * ModelSEIRCONN< TSeq >::clone_ptr [inline], [virtual]
```

Advanced usage: Makes a copy of data and returns it as undeleted pointer.

## Parameters

<i>copy</i>	
-------------	--

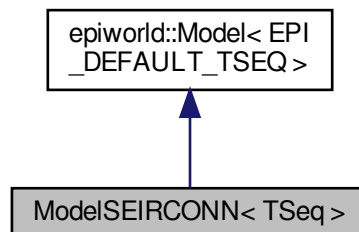
Reimplemented from [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

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

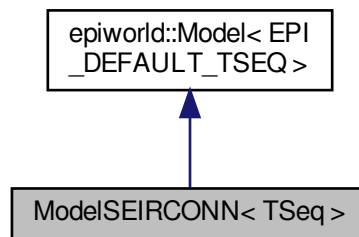
- [epiworld.hpp](#)

## 13.24 ModelSEIRCONN&lt; TSeq &gt; Class Template Reference

Inheritance diagram for ModelSEIRCONN< TSeq >:



Collaboration diagram for ModelSEIRCONN< TSeq >:



## Public Member Functions

- [ModelSEIRCONN](#) ([ModelSEIRCONN](#)< TSeq > &model, std::string vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double prob\_transmission, epiworld\_double incubation\_days, epiworld\_double prob\_recovery)

*Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.*

- **ModelSEIRCONN** (std::string vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double prob\_transmission, epiworld\_double incubation\_days, epiworld\_double prob\_recovery)
- void [run](#) (epiworld\_fast\_uint ndays, int seed=-1)

*Runs the simulation (after initialization)*

- [Model](#)< TSeq > \* [clone\\_ptr](#) ()

*Advanced usage: Makes a copy of data and returns it as undeleted pointer.*

## Public Attributes

- std::vector< [epiworld::Agent](#)<> \* > **tracked\_agents\_infected** = {}
- std::vector< [epiworld::Agent](#)<> \* > **tracked\_agents\_infected\_next** = {}
- bool **tracked\_started** = false
- int **tracked\_ninfected** = 0
- int **tracked\_ninfected\_next** = 0

## Static Public Attributes

- static const int **SUSCEPTIBLE** = 0
- static const int **EXPOSED** = 1
- static const int **INFECTED** = 2
- static const int **RECOVERED** = 3

## Additional Inherited Members

### 13.24.1 Constructor & Destructor Documentation

### 13.24.1.1 ModelSEIRCONN()

```
template<typename TSeq >
ModelSEIRCONN< TSeq >::ModelSEIRCONN (
 ModelSEIRCONN< TSeq > & model,
 std::string vname,
 epiworld_fast_uint n,
 epiworld_double prevalence,
 epiworld_double contact_rate,
 epiworld_double prob_transmission,
 epiworld_double incubation_days,
 epiworld_double prob_recovery) [inline]
```

Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.

#### Parameters

<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>prevalence</i>	Initial prevalence (proportion)
<i>contact_rate</i>	Average number of contacts (interactions) per step.
<i>prob_transmission</i>	Probability of transmission
<i>prob_recovery</i>	Probability of recovery

## 13.24.2 Member Function Documentation

### 13.24.2.1 clone\_ptr()

```
template<typename TSeq >
Model< TSeq > * ModelSEIRCONN< TSeq >::clone_ptr [inline], [virtual]
```

Advanced usage: Makes a copy of data and returns it as undeleted pointer.

#### Parameters

<i>copy</i>	
-------------	--

Reimplemented from [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

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

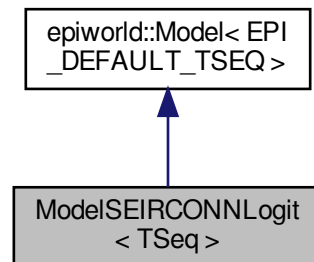
- include/epiworld/models/seirconnected.hpp

## 13.25 ModelSEIRCONNLogit< TSeq > Class Template Reference

Inheritance diagram for ModelSEIRCONNLogit< TSeq >:



Collaboration diagram for ModelSEIRCONNLogit< TSeq >:



### Public Member Functions

- [ModelSEIRCONNLogit](#) ([ModelSEIRCONNLogit](#)< TSeq > &model, std::string vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double reproductive\_number, epiworld\_double prob\_transmission, epiworld\_double incubation\_days, epiworld\_double prob\_recovery, double \*covars, std::vector< double > logit\_params)

*Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.*

- **ModelSEIRCONNLogit** (std::string vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double reproductive\_number, epiworld\_double prob\_transmission, epiworld\_double incubation\_days, epiworld\_double prob\_recovery double \*covars, std::vector< double > logit\_params)

### Public Attributes

- std::vector< [epiworld::Agent](#)<> \* > **tracked\_agents\_infected** = {}
- std::vector< [epiworld::Agent](#)<> \* > **tracked\_agents\_infected\_next** = {}
- bool **tracked\_started** = false
- int **tracked\_ninfected** = 0
- int **tracked\_ninfected\_next** = 0

## Additional Inherited Members

### 13.25.1 Constructor & Destructor Documentation

#### 13.25.1.1 ModelSEIRCONNLogit()

```
template<typename TSeq >
ModelSEIRCONNLogit< TSeq >::ModelSEIRCONNLogit (
 ModelSEIRCONNLogit< TSeq > & model,
 std::string vname,
 epiworld_fast_uint n,
 epiworld_double prevalence,
 epiworld_double reproductive_number,
 epiworld_double prob_transmission,
 epiworld_double incubation_days,
 epiworld_double prob_recovery,
 double * covars,
 std::vector< double > logit_params) [inline]
```

Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.

#### Parameters

<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>prevalence</i>	Initial prevalence (proportion)
<i>reproductive_number</i>	Reproductive number (beta)
<i>prob_transmission</i>	Probability of transmission
<i>prob_recovery</i>	Probability of recovery

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

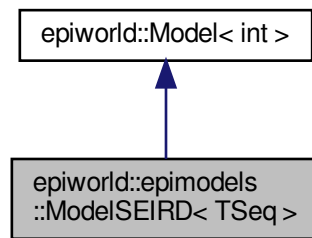
- include/epiworld/models/seirconnected\_logit.hpp

## 13.26 epiworld::epimodels::ModelSEIRD< TSeq > Class Template Reference

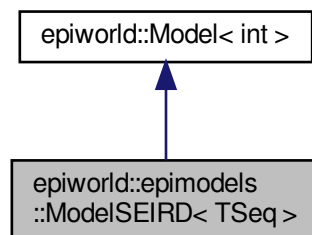
Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model.

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

Inheritance diagram for `epiworld::epimodels::ModelSEIRD< TSeq >`:



Collaboration diagram for `epiworld::epimodels::ModelSEIRD< TSeq >`:



## Public Member Functions

- **ModelSEIRD** ([ModelSEIRD](#)< TSeq > &model, std::string vname)
- **ModelSEIRD** (std::string vname, epiworld\_double prevalence, epiworld\_double incu\_shape, epiworld\_double incu\_rate, epiworld\_double infe\_shape, epiworld\_double infe\_rate, epiworld\_double p\_hosp, epiworld\_double p\_hosp\_rec, epiworld\_double p\_hosp\_die, epiworld\_double p\_transmission, epiworld\_double p\_transmission\_entity, size\_t n\_entities, size\_t n\_interactions)
- **ModelSEIRD** (std::string fn, std::string vname)

## Protected Types

- enum **S** {  
**Susceptible** , **Exposed** , **Infected** , **Hospitalized** ,  
**Recovered** , **Deceased** }



## Static Protected Member Functions

- static void **update\_exposed** ([epiworld::Agent](#)< TSeq > \*p, [epiworld::Model](#)< TSeq > \*m)
- static void **update\_infected** ([epiworld::Agent](#)< TSeq > \*p, [epiworld::Model](#)< TSeq > \*m)
- static void **update\_hospitalized** ([epiworld::Agent](#)< TSeq > \*p, [epiworld::Model](#)< TSeq > \*m)
- static void **contact** ([Model](#)< TSeq > \*m)

*Transmission by contact outside home.*

## Additional Inherited Members

### 13.26.1 Detailed Description

```
template<typename TSeq = int>
class epiworld::epimodels::ModelSEIRD< TSeq >
```

Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model.

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

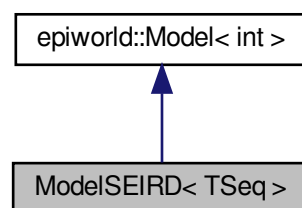
- [epiworld.hpp](#)

## 13.27 ModelSEIRD< TSeq > Class Template Reference

Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model.

```
#include <seird.hpp>
```

Inheritance diagram for ModelSEIRD< TSeq >:



Collaboration diagram for ModelSEIRD< TSeq >:



## Public Member Functions

- **ModelSEIRD** ([ModelSEIRD](#)< TSeq > &model, std::string vname)
- **ModelSEIRD** (std::string vname, epiworld\_double prevalence, epiworld\_double incu\_shape, epiworld\_double incu\_rate, epiworld\_double infe\_shape, epiworld\_double infe\_rate, epiworld\_double p\_hosp, epiworld\_double p\_hosp\_rec, epiworld\_double p\_hosp\_die, epiworld\_double p\_transmission, epiworld\_double p\_transmission\_entity, size\_t n\_entities, size\_t n\_interactions)
- **ModelSEIRD** (std::string fn, std::string vname)

## Protected Types

- enum **S** {  
**Susceptible** , **Exposed** , **Infected** , **Hospitalized** ,  
**Recovered** , **Deceased** }

## Static Protected Member Functions

- static void **update\_exposed** ([epiworld::Agent](#)< TSeq > \*p, [epiworld::Model](#)< TSeq > \*m)
- static void **update\_infected** ([epiworld::Agent](#)< TSeq > \*p, [epiworld::Model](#)< TSeq > \*m)
- static void **update\_hospitalized** ([epiworld::Agent](#)< TSeq > \*p, [epiworld::Model](#)< TSeq > \*m)
- static void **contact** ([Model](#)< TSeq > \*m)

*Transmission by contact outside home.*

## Additional Inherited Members

### 13.27.1 Detailed Description

```
template<typename TSeq = int>
class ModelSEIRD< TSeq >
```

Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model.

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

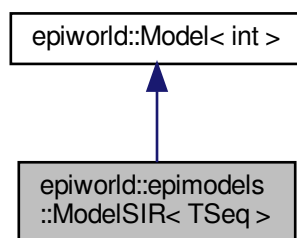
- include/epiworld/models/seird.hpp

## 13.28 epiworld::epimodels::ModelSIR< TSeq > Class Template Reference

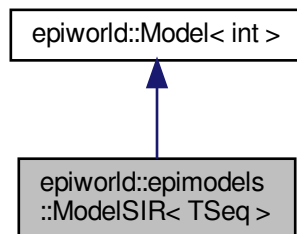
Template for a Susceptible-Infected-Removed (SIR) model.

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

Inheritance diagram for epiworld::epimodels::ModelSIR< TSeq >:



Collaboration diagram for epiworld::epimodels::ModelSIR< TSeq >:



### Public Member Functions

- **ModelSIR** ([ModelSIR](#)< TSeq > &model, std::string vname, epiworld\_double prevalence, epiworld\_double infectiousness, epiworld\_double recovery)
- **ModelSIR** (std::string vname, epiworld\_double prevalence, epiworld\_double infectiousness, epiworld\_double recovery)

## Additional Inherited Members

### 13.28.1 Detailed Description

```
template<typename TSeq = int>
class epiworld::epimodels::ModelSIR< TSeq >
```

Template for a Susceptible-Infected-Removed (SIR) model.

## Parameters

<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>initial_prevalence</i>	epiworld_double Initial prevalence
<i>initial_efficiency</i>	epiworld_double Initial susceptibility_reduction of the immune system
<i>initial_recovery</i>	epiworld_double Initial recovery rate of the immune system

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

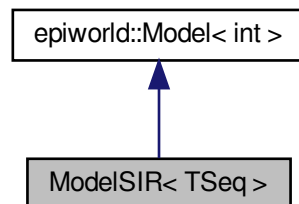
- epiworld.hpp

## 13.29 ModelSIR< TSeq > Class Template Reference

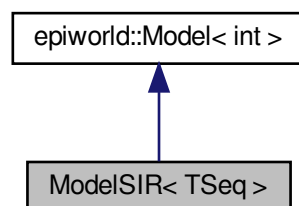
Template for a Susceptible-Infected-Removed (SIR) model.

```
#include <sir.hpp>
```

Inheritance diagram for ModelSIR< TSeq >:



Collaboration diagram for ModelSIR< TSeq >:



## Public Member Functions

- **ModelSIR** ([ModelSIR](#)< TSeq > &model, std::string vname, epiworld\_double prevalence, epiworld\_double infectiousness, epiworld\_double recovery)
- **ModelSIR** (std::string vname, epiworld\_double prevalence, epiworld\_double infectiousness, epiworld\_double recovery)

## Additional Inherited Members

### 13.29.1 Detailed Description

```
template<typename TSeq = int>
class ModelSIR< TSeq >
```

Template for a Susceptible-Infected-Removed (SIR) model.

#### Parameters

<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>initial_prevalence</i>	epiworld_double Initial prevalence
<i>initial_efficacy</i>	epiworld_double Initial susceptibility_reduction of the immune system
<i>initial_recovery</i>	epiworld_double Initial recovery rate of the immune system

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

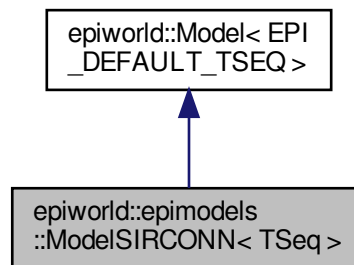
- include/epiworld/models/sir.hpp

## 13.30 epiworld::epimodels::ModelSIRCONN< TSeq > Class Template Reference

Inheritance diagram for epiworld::epimodels::ModelSIRCONN< TSeq >:



Collaboration diagram for epiworld::epimodels::ModelSIRCONN< TSeq >:



## Public Member Functions

- [ModelSIRCONN](#) ([ModelSIRCONN](#)< TSeq > &model, std::string vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double prob\_transmission, epiworld\_double prob\_recovery)  
*Template for a Susceptible-Infected-Removed (SIR) model.*
- **ModelSIRCONN** (std::string vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double prob\_transmission, epiworld\_double prob\_recovery)
- void [run](#) (epiworld\_fast\_uint ndays, int seed=-1)  
*Runs the simulation (after initialization)*
- [Model](#)< TSeq > \* [clone\\_ptr](#) ()  
*Advanced usage: Makes a copy of data and returns it as undeleted pointer.*

## Public Attributes

- std::vector< [epiworld::Agent](#)< TSeq > \* > **tracked\_agents\_infected** = {}
- std::vector< [epiworld::Agent](#)< TSeq > \* > **tracked\_agents\_infected\_next** = {}
- bool **tracked\_started** = false
- int **tracked\_ninfected** = 0
- int **tracked\_ninfected\_next** = 0
- epiworld\_double **tracked\_current\_infect\_prob** = 0.0

## Additional Inherited Members

### 13.30.1 Constructor & Destructor Documentation

### 13.30.1.1 ModelSIRCONN()

```
template<typename TSeq >
ModelSIRCONN< TSeq >::ModelSIRCONN (
 ModelSIRCONN< TSeq > & model,
 std::string vname,
 epiworld_fast_uint n,
 epiworld_double prevalence,
 epiworld_double contact_rate,
 epiworld_double prob_transmission,
 epiworld_double prob_recovery) [inline]
```

Template for a Susceptible-Infected-Removed (SIR) model.

#### Parameters

<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>prevalence</i>	Initial prevalence (proportion)
<i>contact_rate</i>	Average number of contacts (interactions) per step.
<i>prob_transmission</i>	Probability of transmission
<i>prob_recovery</i>	Probability of recovery

## 13.30.2 Member Function Documentation

### 13.30.2.1 clone\_ptr()

```
template<typename TSeq >
Model< TSeq > * ModelSIRCONN< TSeq >::clone_ptr [inline], [virtual]
```

Advanced usage: Makes a copy of data and returns it as undeleted pointer.

#### Parameters

<i>copy</i>	
-------------	--

Reimplemented from [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

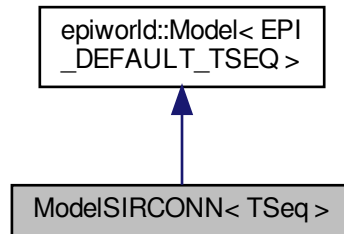
The documentation for this class was generated from the following file:

- [epiworld.hpp](#)

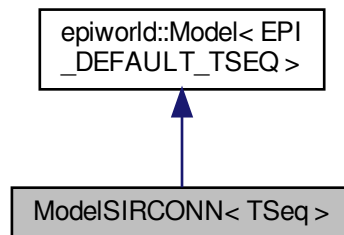


## 13.31 ModelSIRCONN< TSeq > Class Template Reference

Inheritance diagram for ModelSIRCONN< TSeq >:



Collaboration diagram for ModelSIRCONN< TSeq >:



### Public Member Functions

- [ModelSIRCONN](#) ([ModelSIRCONN](#)< TSeq > &model, std::string vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double prob\_transmission, epiworld\_double prob\_recovery)  
*Template for a Susceptible-Infected-Removed (SIR) model.*
- **ModelSIRCONN** (std::string vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double prob\_transmission, epiworld\_double prob\_recovery)
- void [run](#) (epiworld\_fast\_uint ndays, int seed=-1)  
*Runs the simulation (after initialization)*
- [Model](#)< TSeq > \* [clone\\_ptr](#) ()  
*Advanced usage: Makes a copy of data and returns it as undeleted pointer.*

## Public Attributes

- `std::vector< epiworld::Agent < TSeq > * > tracked_agents_infected = {}`
- `std::vector< epiworld::Agent < TSeq > * > tracked_agents_infected_next = {}`
- `bool tracked_started = false`
- `int tracked_ninfected = 0`
- `int tracked_ninfected_next = 0`
- `epiworld_double tracked_current_infect_prob = 0.0`

## Additional Inherited Members

### 13.31.1 Constructor & Destructor Documentation

#### 13.31.1.1 ModelSIRCONN()

```
template<typename TSeq >
ModelSIRCONN< TSeq >::ModelSIRCONN (
 ModelSIRCONN< TSeq > & model,
 std::string vname,
 epiworld_fast_uint n,
 epiworld_double prevalence,
 epiworld_double contact_rate,
 epiworld_double prob_transmission,
 epiworld_double prob_recovery) [inline]
```

Template for a Susceptible-Infected-Removed (SIR) model.

#### Parameters

<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>prevalence</i>	Initial prevalence (proportion)
<i>contact_rate</i>	Average number of contacts (interactions) per step.
<i>prob_transmission</i>	Probability of transmission
<i>prob_recovery</i>	Probability of recovery

### 13.31.2 Member Function Documentation

#### 13.31.2.1 clone\_ptr()

```
template<typename TSeq >
Model< TSeq > * ModelSIRCONN< TSeq >::clone_ptr [inline], [virtual]
```

Advanced usage: Makes a copy of data and returns it as undeleted pointer.

## Parameters

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

Reimplemented from [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

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

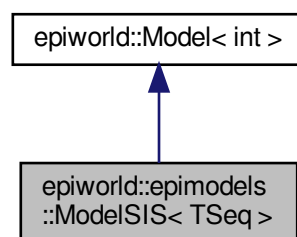
- include/epiworld/models/sirconnected.hpp

## 13.32 epiworld::epimodels::ModelSIS< TSeq > Class Template Reference

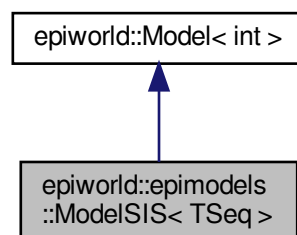
Template for a Susceptible-Infected-Susceptible (SIS) model.

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

Inheritance diagram for epiworld::epimodels::ModelSIS< TSeq >:



Collaboration diagram for epiworld::epimodels::ModelSIS< TSeq >:



## Public Member Functions

- **ModelSIS** ([ModelSIS](#)< TSeq > &model, std::string vname, epiworld\_double prevalence, epiworld\_double infectiousness, epiworld\_double recovery)
- **ModelSIS** (std::string vname, epiworld\_double prevalence, epiworld\_double infectiousness, epiworld\_double recovery)

## Additional Inherited Members

### 13.32.1 Detailed Description

```
template<typename TSeq = int>
class epiworld::epimodels::ModelSIS< TSeq >
```

Template for a Susceptible-Infected-Susceptible (SIS) model.

#### Parameters

<i>vname</i>	std::string Name of the virus
<i>initial_prevalence</i>	epiworld_double Initial prevalence
<i>initial_efficacy</i>	epiworld_double Initial susceptibility_reduction of the immune system
<i>initial_recovery</i>	epiworld_double Initial recovery rate of the immune system

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

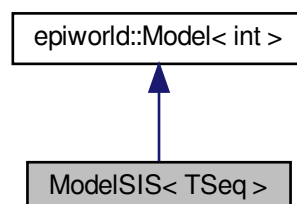
- epiworld.hpp

## 13.33 ModelSIS< TSeq > Class Template Reference

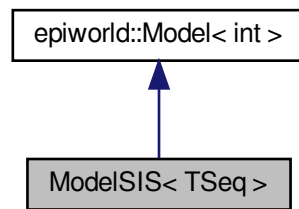
Template for a Susceptible-Infected-Susceptible (SIS) model.

```
#include <sis.hpp>
```

Inheritance diagram for ModelSIS< TSeq >:



Collaboration diagram for ModelSIS< TSeq >:



## Public Member Functions

- **ModelSIS** ([ModelSIS](#)< TSeq > &model, std::string vname, epiworld\_double prevalence, epiworld\_double infectiousness, epiworld\_double recovery)
- **ModelSIS** (std::string vname, epiworld\_double prevalence, epiworld\_double infectiousness, epiworld\_double recovery)

## Additional Inherited Members

### 13.33.1 Detailed Description

```
template<typename TSeq = int>
class ModelSIS< TSeq >
```

Template for a Susceptible-Infected-Susceptible (SIS) model.

#### Parameters

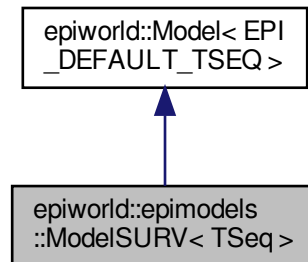
<i>vname</i>	std::string Name of the virus
<i>initial_prevalence</i>	epiworld_double Initial prevalence
<i>initial_efficiency</i>	epiworld_double Initial susceptibility_reduction of the immune system
<i>initial_recovery</i>	epiworld_double Initial recovery rate of the immune system

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

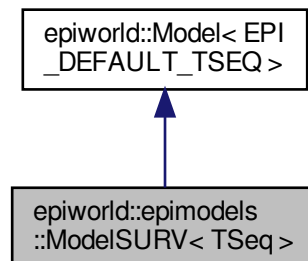
- include/epiworld/models/sis.hpp

### 13.34 epiworld::epimodels::ModelSURV< TSeq > Class Template Reference

Inheritance diagram for epiworld::epimodels::ModelSURV< TSeq >:



Collaboration diagram for epiworld::epimodels::ModelSURV< TSeq >:



#### Public Member Functions

##### Construct a new ModelSURV object

The [ModelSURV](#) class simulates a surveillance model where agents can be isolated, even if asymptomatic.

##### Parameters

vname	<i>String. Name of the virus</i>
prevalence	<i>Integer. Number of initial cases of the virus.</i>
efficacy_vax	<i>Double. Efficacy of the vaccine (1 - P(acquire the disease)).</i>
latent_period	<i>Double. Shape parameter of a Gamma (latent_period, 1) distribution. This coincides with the expected number of latent days.</i>

*Parameters*

<code>infect_period</code>	<i>Double. Shape parameter of a Gamma (<code>infected_period</code>, 1) distribution. This coincides with the expected number of infectious days.</i>
<code>prob_symptoms</code>	<i>Double. Probability of generating symptoms.</i>
<code>prop_vaccinated</code>	<i>Double. Probability of vaccination. Coincides with the initial prevalence of vaccinated individuals.</i>
<code>prop_vax_redux_transm</code>	<i>Double. Factor by which the vaccine reduces transmissibility.</i>
<code>prop_vax_redux_infect</code>	<i>Double. Factor by which the vaccine reduces the chances of becoming infected.</i>
<code>surveillance_prob</code>	<i>Double. Probability of testing an agent.</i>
<code>prob_transmission</code>	<i>Double. Raw transmission probability.</i>
<code>prob_death</code>	<i>Double. Raw probability of death for symptomatic individuals.</i>
<code>prob_noreinfect</code>	<i>Double. Probability of no re-infection.</i>

This model features the following states:

- *Susceptible*
- *Latent*
- *Symptomatic*
- *Symptomatic isolated*
- *Asymptomatic*
- *Asymptomatic isolated*
- *Recovered*
- *Removed*

*Returns*

An object of class `epiworld_surv`

- **ModelSURV** ()
- **ModelSURV** (`ModelSURV< TSeq >` &model, `std::string` vname, `epiworld_fast_uint` prevalence=50, `epiworld_double` efficacy\_vax=0.9, `epiworld_double` latent\_period=3u, `epiworld_double` infect\_period=6u, `epiworld_double` prob\_symptoms=0.6, `epiworld_double` prop\_vaccinated=0.25, `epiworld_double` prop\_vax\_redux\_transm=0.5, `epiworld_double` prop\_vax\_redux\_infect=0.5, `epiworld_double` surveillance\_prob=0.001, `epiworld_double` prob\_transmission=1.0, `epiworld_double` prob\_death=0.001, `epiworld_double` prob\_noreinfect=0.9)
- **ModelSURV** (`std::string` vname, `epiworld_fast_uint` prevalence=50, `epiworld_double` efficacy\_vax=0.9, `epiworld_double` latent\_period=3u, `epiworld_double` infect\_period=6u, `epiworld_double` prob\_symptoms=0.6, `epiworld_double` prop\_vaccinated=0.25, `epiworld_double` prop\_vax\_redux\_transm=0.5, `epiworld_double` prop\_vax\_redux\_infect=0.5, `epiworld_double` surveillance\_prob=0.001, `epiworld_double` prob\_transmission=1.0, `epiworld_double` prob\_death=0.001, `epiworld_double` prob\_noreinfect=0.9)

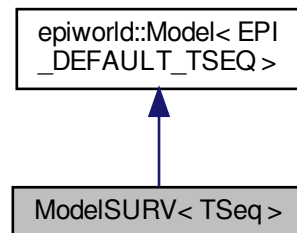
**Additional Inherited Members**

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

- `epiworld.hpp`

### 13.35 ModelSURV< TSeq > Class Template Reference

Inheritance diagram for ModelSURV< TSeq >:



Collaboration diagram for ModelSURV< TSeq >:



#### Public Member Functions

##### Construct a new ModelSURV object

The [ModelSURV](#) class simulates a surveillance model where agents can be isolated, even if asyptomatic.

##### Parameters

vname	<i>String. Name of the virus</i>
prevalence	<i>Integer. Number of initial cases of the virus.</i>
efficacy_vax	<i>Double. Efficacy of the vaccine (1 - P(acquire the disease)).</i>
latent_period	<i>Double. Shape parameter of a Gamma (latent_period, 1) distribution. This coincides with the expected number of latent days.</i>
infect_period	<i>Double. Shape parameter of a Gamma (infected_period, 1) distribution. This coincides with the expected number of infectious days.</i>
prob_symptoms	<i>Double. Probability of generating symptoms.</i>



*Parameters*

prop_vaccinated	<i>Double. Probability of vaccination. Coincides with the initial prevalence of vaccinated individuals.</i>
prop_vax_redux_transm	<i>Double. Factor by which the vaccine reduces transmissibility.</i>
prop_vax_redux_infect	<i>Double. Factor by which the vaccine reduces the chances of becoming infected.</i>
surveillance_prob	<i>Double. Probability of testing an agent.</i>
prob_transmission	<i>Double. Raw transmission probability.</i>
prob_death	<i>Double. Raw probability of death for symptomatic individuals.</i>
prob_noreinfect	<i>Double. Probability of no re-infection.</i>

*This model features the following states:*

- *Susceptible*
- *Latent*
- *Symptomatic*
- *Symptomatic isolated*
- *Asymptomatic*
- *Asymptomatic isolated*
- *Recovered*
- *Removed*

*Returns*

*An object of class `epiworld_surv`*

- **ModelSURV** ()
- **ModelSURV** ([ModelSURV](#)< TSeq > &model, std::string vname, epiworld\_fast\_uint prevalence=50, epiworld\_double efficacy\_vax=0.9, epiworld\_double latent\_period=3u, epiworld\_double infect\_period=6u, epiworld\_double prob\_symptoms=0.6, epiworld\_double prop\_vaccinated=0.25, epiworld\_double prop\_vax\_redux\_transm=0.5, epiworld\_double prop\_vax\_redux\_infect=0.5, epiworld\_double surveillance\_prob=0.001, epiworld\_double prob\_transmission=1.0, epiworld\_double prob\_death=0.001, epiworld\_double prob\_noreinfect=0.9)
- **ModelSURV** (std::string vname, epiworld\_fast\_uint prevalence=50, epiworld\_double efficacy\_vax=0.9, epiworld\_double latent\_period=3u, epiworld\_double infect\_period=6u, epiworld\_double prob\_symptoms=0.6, epiworld\_double prop\_vaccinated=0.25, epiworld\_double prop\_vax\_redux\_transm=0.5, epiworld\_double prop\_vax\_redux\_infect=0.5, epiworld\_double surveillance\_prob=0.001, epiworld\_double prob\_transmission=1.0, epiworld\_double prob\_death=0.001, epiworld\_double prob\_noreinfect=0.9)

**Additional Inherited Members**

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

- include/epiworld/models/surveillance.hpp

### 13.36 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

### 13.37 epiworld::PersonTools< TSeq > Class Template Reference

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

- epiworld.hpp

### 13.38 PersonTools< TSeq > Class Template Reference

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

- include/epiworld/config.hpp

### 13.39 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** ()

### 13.39.1 Detailed Description

A simple progress bar.

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

- epiworld.hpp

## 13.40 Progress Class Reference

A simple progress bar.

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

### Public Member Functions

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

### 13.40.1 Detailed Description

A simple progress bar.

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

- include/epiworld/progress.hpp

## 13.41 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[]** (epiworld\_fast\_uint i)
- void **reset** ()
- bool **operator==** (const [Queue](#)< TSeq > &other) const
- bool **operator!=** (const [Queue](#)< TSeq > &other) const

### Friends

- class **Model**< TSeq >

### 13.41.1 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`

## 13.42 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[]` (`epiworld_fast_uint i`)
- void **reset** ()
- bool **operator==** (const [Queue](#)< TSeq > &other) const
- bool **operator!=** (const [Queue](#)< TSeq > &other) const

### Friends

- class **Model**< TSeq >

### 13.42.1 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`

## 13.43 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

## 13.44 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

## 13.45 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

## 13.46 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

## 13.47 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

## 13.48 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** (std::shared\_ptr< TSeq > d)
- std::shared\_ptr< TSeq > **get\_sequence** ()
- 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\_state** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **set\_queue** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **get\_state** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- void **get\_queue** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- bool **operator==** (const [Tool](#)< TSeq > &other) const
- bool **operator!=** (const [Tool](#)< TSeq > &other) 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, [Model](#)< TSeq > \*model)
- *epiworld\_double* **get\_transmission\_reduction** (VirusPtr< TSeq > v, [Model](#)< TSeq > \*model)
- *epiworld\_double* **get\_recovery\_enhancer** (VirusPtr< TSeq > v, [Model](#)< TSeq > \*model)
- *epiworld\_double* **get\_death\_reduction** (VirusPtr< TSeq > v, [Model](#)< TSeq > \*model)
- 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)

**13.48.1 Detailed Description**

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

- *epiworld.hpp*

**13.49 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** (std::shared\_ptr< TSeq > d)
- std::shared\_ptr< TSeq > **get\_sequence** ()
- 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\_state** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **set\_queue** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **get\_state** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- void **get\_queue** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- bool **operator==** (const [Tool](#)< TSeq > &other) const
- bool **operator!=** (const [Tool](#)< TSeq > &other) const
- bool **operator==** (const [Tool](#)< std::vector< int >> &other) const
- bool **operator==** (const [Tool](#)< std::vector< int >> &other) 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, [Model](#)< TSeq > \*model)
- epiworld\_double **get\_transmission\_reduction** (VirusPtr< TSeq > v, [Model](#)< TSeq > \*model)
- epiworld\_double **get\_recovery\_enhancer** (VirusPtr< TSeq > v, [Model](#)< TSeq > \*model)
- epiworld\_double **get\_death\_reduction** (VirusPtr< TSeq > v, [Model](#)< TSeq > \*model)
- 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)



### 13.49.1 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

## 13.50 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 >

### 13.50.1 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

## 13.51 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 >

### 13.51.1 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

## 13.52 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** () const
- std::vector< ToolPtr< TSeq > >::const\_iterator **end** () const
- 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 >

#### 13.52.1 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

## 13.53 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** () const
- std::vector< ToolPtr< TSeq > >::const\_iterator **end** () const
- 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 >

### 13.53.1 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

## 13.54 epiworld::UserData< TSeq > Class Template Reference

Personalized data by the user.

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

### Public Member Functions

- **UserData** ([Model](#)< TSeq > &m)
- **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)
- epiworld\_fast\_uint **nrow** () const
- epiworld\_fast\_uint **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** (epiworld\_fast\_uint 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()** (epiworld\_fast\_uint i, epiworld\_fast\_uint j)
- epiworld\_double & **operator()** (epiworld\_fast\_uint i, std::string name)

**Friends**

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

**13.54.1 Detailed Description**

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

Personalized data by the user.

**Template Parameters**

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

**13.54.2 Constructor & Destructor Documentation****13.54.2.1 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

## 13.55 UserData< TSeq > Class Template Reference

Personalized data by the user.

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

**Public Member Functions**

- **UserData** ([Model](#)< TSeq > &m)
- **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)
- epiworld\_fast\_uint **nrow** () const
- epiworld\_fast\_uint **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** (epiworld\_fast\_uint j, epiworld\_double x)

**Access data****Parameters**

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

*Returns**epiworld\_double&*

- *epiworld\_double* & **operator()** (*epiworld\_fast\_uint* i, *epiworld\_fast\_uint* j)
- *epiworld\_double* & **operator()** (*epiworld\_fast\_uint* i, *std::string* name)

**Friends**

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

**13.55.1 Detailed Description**

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

Personalized data by the user.

**Template Parameters**

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

**13.55.2 Constructor & Destructor Documentation****13.55.2.1 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

## 13.56 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`

#### 13.56.1 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`

## 13.57 vecHasher< T > Struct Template Reference

Vector hasher.

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

### Public Member Functions

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

#### 13.57.1 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

## 13.58 epiworld::Virus< TSeq > Class Template Reference

[Virus.](#)

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

### Public Member Functions

- **Virus** (std::string name="unknown virus")
- void **mutate** ([Model](#)< TSeq > \*model)
- 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)
- 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** ()
- bool **operator==** (const [Virus](#)< TSeq > &other) const
- bool **operator!=** (const [Virus](#)< TSeq > &other) const
- void **print** () 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\_prob\_infecting** ([Model](#)< TSeq > \*model)
- epiworld\_double **get\_prob\_recovery** ([Model](#)< TSeq > \*model)
- epiworld\_double **get\_prob\_death** ([Model](#)< TSeq > \*model)
- void **post\_recovery** ([Model](#)< TSeq > \*model)
- 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 state and queue

After applied, viruses can change the state 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 state 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\_state** (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\_state** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*end, epiworld\_fast\_int \*removed=nullptr)
- void **get\_queue** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*end, epiworld\_fast\_int \*removed=nullptr)

### 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)

## 13.58.1 Detailed Description

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

[Virus](#).

#### Template Parameters

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

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 file:

- `epiworld.hpp`

## 13.59 Virus< TSeq > Class Template Reference

Virus.

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

### Public Member Functions

- **Virus** (std::string name="unknown virus")
- void **mutate** (Model< TSeq > \*model)
- 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)
- 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** ()
- bool **operator==** (const Virus< TSeq > &other) const
- bool **operator!=** (const Virus< TSeq > &other) const
- void **print** () const
- bool **operator==** (const Virus< std::vector< int >> &other) const
- bool **operator==** (const Virus< std::vector< int >> &other) 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\_prob\_infecting** (Model< TSeq > \*model)
- epiworld\_double **get\_prob\_recovery** (Model< TSeq > \*model)
- epiworld\_double **get\_prob\_death** (Model< TSeq > \*model)
- void **post\_recovery** (Model< TSeq > \*model)
- 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 state and queue

After applied, viruses can change the state 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 state 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\_state** (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\_state** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*end, epiworld\_fast\_int \*removed=nullptr)
- void **get\_queue** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*end, epiworld\_fast\_int \*removed=nullptr)

### 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)

## 13.59.1 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

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

### Friends

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

#### 13.60.1 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

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

## Friends

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

### 13.61.1 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

## 13.62 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< TSeq > >::const\_iterator **begin** () const
- std::vector< VirusPtr< TSeq > >::const\_iterator **end** () const
- const VirusPtr< TSeq > & **operator()** (size\_t i)
- const VirusPtr< TSeq > & **operator[]** (size\_t i)
- size\_t **size** () const noexcept

## Friends

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

### 13.62.1 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

## 13.63 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< TSeq > >::const\_iterator **begin** () const
- std::vector< VirusPtr< TSeq > >::const\_iterator **end** () const
- const VirusPtr< TSeq > & **operator()** (size\_t i)
- const VirusPtr< TSeq > & **operator[]** (size\_t i)
- size\_t **size** () const noexcept

### Friends

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

### 13.63.1 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 14

# File Documentation

### 14.1 include/epiworld/agent-meat-state.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-state.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)`

### 14.1.1 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 >, 36  
  epiworld::Action< TSeq >, 37  
Action< TSeq >, 35  
  Action, 36  
actions\_add  
  epiworld::Model< TSeq >, 74  
  Model< TSeq >, 86  
actions\_run  
  epiworld::Model< TSeq >, 75  
  Model< TSeq >, 87  
add\_global\_action  
  epiworld::Model< TSeq >, 75  
  Model< TSeq >, 87  
AdjList, 38  
  AdjList, 39  
  epiworld::AdjList, 40  
  read\_edgelist, 39  
Agent< TSeq >, 41  
  default\_rm\_entity, 44  
  get\_entities, 44  
  swap\_neighbors, 44  
AgentsSample  
  AgentsSample< TSeq >, 49  
  epiworld::AgentsSample< TSeq >, 51  
AgentsSample< TSeq >, 49  
  AgentsSample, 49  
clone\_ptr  
  epiworld::epimodels::ModelSEIRCONN< TSeq >, 97  
  epiworld::epimodels::ModelSIRCONN< TSeq >, 110  
  epiworld::Model< TSeq >, 76  
  Model< TSeq >, 88  
  ModelSEIRCONN< TSeq >, 99  
  ModelSIRCONN< TSeq >, 112  
DataBase< TSeq >, 52  
  generation\_time, 53  
  operator==, 54  
  record\_variant, 55  
  reproductive\_number, 55  
  transition\_probability, 55  
default\_rm\_entity  
  Agent< TSeq >, 44  
  Entity< TSeq >, 63  
  epiworld::Agent< TSeq >, 48  
  epiworld::Entity< TSeq >, 64  
Entities< TSeq >, 59  
Entities\_const< TSeq >, 61  
Entity< TSeq >, 63  
  default\_rm\_entity, 63  
epiworld::Action< TSeq >, 37  
  Action, 37  
epiworld::AdjList, 40  
  AdjList, 40  
  read\_edgelist, 41  
epiworld::Agent< TSeq >, 45  
  default\_rm\_entity, 48  
  get\_entities, 47  
  swap\_neighbors, 48  
epiworld::AgentsSample< TSeq >, 50  
  AgentsSample, 51  
epiworld::DataBase< TSeq >, 56  
  generation\_time, 58  
  operator==, 58  
  record\_variant, 58  
  reproductive\_number, 59  
  transition\_probability, 59  
epiworld::Entities< TSeq >, 60  
epiworld::Entities\_const< TSeq >, 62  
epiworld::Entity< TSeq >, 64  
  default\_rm\_entity, 64  
epiworld::epimodels::ModelSEIR< TSeq >, 91  
  update\_exposed\_seir, 92  
  update\_infected\_seir, 92  
epiworld::epimodels::ModelSEIRCONN< TSeq >, 95  
  clone\_ptr, 97  
  ModelSEIRCONN, 96  
epiworld::epimodels::ModelSEIRD< TSeq >, 101  
epiworld::epimodels::ModelSIR< TSeq >, 105  
epiworld::epimodels::ModelSIRCONN< TSeq >, 108  
  clone\_ptr, 110  
  ModelSIRCONN, 109  
epiworld::epimodels::ModelSIS< TSeq >, 113  
epiworld::epimodels::ModelSURV< TSeq >, 116  
epiworld::LFMCMC< TData >, 65  
epiworld::Model< TSeq >, 67  
  actions\_add, 74  
  actions\_run, 75  
  add\_global\_action, 75  
  clone\_ptr, 76  
  load\_agents\_entities\_ties, 76  
  reset, 76  
  run\_multiple, 77  
  set\_agents\_data, 77  
  set\_name, 78

- time\_elapsed, 79
- write\_data, 78
- epiworld::PersonTools< TSeq >, 120
- epiworld::Progress, 120
- epiworld::Queue< TSeq >, 121
- epiworld::QueueValues, 123
- epiworld::sampler, 27
  - make\_sample\_virus\_neighbors, 27
  - make\_update\_susceptible, 28
  - sample\_virus\_single, 28
- epiworld::SAMPLETYPE, 123
- epiworld::Tool< TSeq >, 124
- epiworld::Tools< TSeq >, 127
- epiworld::Tools\_const< TSeq >, 129
- epiworld::UserData< TSeq >, 130
  - UserData, 131
- epiworld::vecHasher< T >, 134
- epiworld::Virus< TSeq >, 135
- epiworld::Viruses< TSeq >, 139
- epiworld::Viruses\_const< TSeq >, 140
- generation\_time
  - DataBase< TSeq >, 53
  - epiworld::DataBase< TSeq >, 58
- get\_entities
  - Agent< TSeq >, 44
  - epiworld::Agent< TSeq >, 47
- include/epiworld/agent-meet-state.hpp, 143
- LFMCMC< TData >, 66
- load\_agents\_entities\_ties
  - epiworld::Model< TSeq >, 76
  - Model< TSeq >, 88
- make\_sample\_virus\_neighbors
  - epiworld::sampler, 27
  - sampler, 30
- make\_update\_susceptible
  - epiworld::sampler, 28
  - sampler, 31
- Model< TSeq >, 79
  - actions\_add, 86
  - actions\_run, 87
  - add\_global\_action, 87
  - clone\_ptr, 88
  - load\_agents\_entities\_ties, 88
  - reset, 88
  - run\_multiple, 89
  - set\_agents\_data, 89
  - set\_name, 89
  - time\_elapsed, 90
  - write\_data, 90
- ModelSEIR< TSeq >, 93
  - update\_exposed\_seir, 94
  - update\_infected\_seir, 94
- ModelSEIRCONN
  - epiworld::epimodels::ModelSEIRCONN< TSeq >, 96
- ModelSEIRCONN< TSeq >, 98
  - clone\_ptr, 99
  - ModelSEIRCONN, 98
- ModelSEIRCONNLogit
  - ModelSEIRCONNLogit< TSeq >, 101
- ModelSEIRCONNLogit< TSeq >, 100
  - ModelSEIRCONNLogit, 101
- ModelSEIRD< TSeq >, 103
- ModelSIR< TSeq >, 107
- ModelSIRCONN
  - epiworld::epimodels::ModelSIRCONN< TSeq >, 109
  - ModelSIRCONN< TSeq >, 112
- ModelSIRCONN< TSeq >, 111
  - clone\_ptr, 112
  - ModelSIRCONN, 112
- ModelSIS< TSeq >, 114
- ModelSURV< TSeq >, 118
- Network< Nettype, Nodetype, Edgetype >, 120
- operator==
  - DataBase< TSeq >, 54
  - epiworld::DataBase< TSeq >, 58
- PersonTools< TSeq >, 120
- Progress, 121
- Queue< TSeq >, 122
- QueueValues, 123
- RandGraph, 123
- read\_edgelist
  - AdjList, 39
  - epiworld::AdjList, 41
- record\_variant
  - DataBase< TSeq >, 55
  - epiworld::DataBase< TSeq >, 58
- reproductive\_number
  - DataBase< TSeq >, 55
  - epiworld::DataBase< TSeq >, 59
- reset
  - epiworld::Model< TSeq >, 76
  - Model< TSeq >, 88
- run\_multiple
  - epiworld::Model< TSeq >, 77
  - Model< TSeq >, 89
- sample\_virus\_single
  - epiworld::sampler, 28
  - sampler, 31
- sampler, 30
  - make\_sample\_virus\_neighbors, 30
  - make\_update\_susceptible, 31
  - sample\_virus\_single, 31
- SAMPLETYPE, 124
- set\_agents\_data
  - epiworld::Model< TSeq >, 77
  - Model< TSeq >, 89

set\_name  
  epiworld::Model< TSeq >, 78  
  Model< TSeq >, 89

swap\_neighbors  
  Agent< TSeq >, 44  
  epiworld::Agent< TSeq >, 48

time\_elapsed  
  epiworld::Model< TSeq >, 79  
  Model< TSeq >, 90

Tool< TSeq >, 125

Tools< TSeq >, 128

Tools\_const< TSeq >, 129

transition\_probability  
  DataBase< TSeq >, 55  
  epiworld::DataBase< TSeq >, 59

update\_exposed\_seir  
  epiworld::epimodels::ModelSEIR< TSeq >, 92  
  ModelSEIR< TSeq >, 94

update\_infected\_seir  
  epiworld::epimodels::ModelSEIR< TSeq >, 92  
  ModelSEIR< TSeq >, 94

UserData  
  epiworld::UserData< TSeq >, 131  
  UserData< TSeq >, 133

UserData< TSeq >, 132  
  UserData, 133

vecHasher< T >, 134

Virus< TSeq >, 137

Viruses< TSeq >, 139

Viruses\_const< TSeq >, 141

write\_data  
  epiworld::Model< TSeq >, 78  
  Model< TSeq >, 90