

epiworld

0.8.2

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 General parameters	11
5.1 Compartmental Models	11
5.1.1 SIR Model	11
5.1.2 SEIR Model	12
5.2 Agent-Based Model Approach	12
5.2.1 Mathematical preliminaries	13
5.2.2 Simulation study	13
5.3 Comparing ABM with Compartmental Models	13
5.3.1 SIR	13
5.3.2 SEIR	14
5.3.3 Rates	14
6 MIT License	15
7 model1	17
8 Mixing probabilities in connected model	19
8.1 Case 1: No grouping	19
8.2 Case 2: Grouping	20
9 EPI Simulator	21
9.1 Disease dynamics	21
9.2 Network dynamics	21
9.3 Contagion dynamics	21
9.4 Time dynamics	21
9.5 Updating agent's status	22
9.5.1 Other parameters	22
10 Namespace Index	23
10.1 Namespace List	23
11 Hierarchical Index	25

11.1 Class Hierarchy	25
12 Class Index	27
12.1 Class List	27
13 File Index	29
13.1 File List	29
14 Namespace Documentation	31
14.1 sampler Namespace Reference	31
14.1.1 Detailed Description	31
14.1.2 Function Documentation	31
14.1.2.1 make_sample_virus_neighbors()	31
14.1.2.2 make_update_susceptible()	32
14.1.2.3 sample_virus_single()	32
15 Class Documentation	35
15.1 AdjList Class Reference	35
15.1.1 Constructor & Destructor Documentation	35
15.1.1.1 AdjList()	35
15.1.2 Member Function Documentation	36
15.1.2.1 read_edgelist()	36
15.2 Agent< TSeq > Class Template Reference	36
15.2.1 Detailed Description	39
15.2.2 Member Function Documentation	39
15.2.2.1 operator>()	39
15.2.2.2 swap_neighbors()	39
15.2.3 Friends And Related Function Documentation	40
15.2.3.1 default_rm_entity	40
15.3 AgentsSample< TSeq > Class Template Reference	40
15.3.1 Detailed Description	41
15.3.2 Constructor & Destructor Documentation	41
15.3.2.1 AgentsSample()	41
15.4 DataBase< TSeq > Class Template Reference	42
15.4.1 Detailed Description	43
15.4.2 Member Function Documentation	44
15.4.2.1 get_generation_time()	44
15.4.2.2 get_reproductive_number()	44
15.4.2.3 get_transition_probability()	45
15.4.2.4 get_transmissions()	45
15.4.2.5 operator==() [1/2]	45
15.4.2.6 operator==() [2/2]	46
15.4.2.7 record_virus()	46
15.5 Entities< TSeq > Class Template Reference	46

15.5.1 Detailed Description	47
15.6 Entities_const< TSeq > Class Template Reference	47
15.6.1 Detailed Description	48
15.7 Entity< TSeq > Class Template Reference	48
15.7.1 Constructor & Destructor Documentation	49
15.7.1.1 Entity()	49
15.7.2 Friends And Related Function Documentation	49
15.7.2.1 default_rm_entity	49
15.8 Event< TSeq > Struct Template Reference	50
15.8.1 Detailed Description	50
15.8.2 Constructor & Destructor Documentation	51
15.8.2.1 Event()	51
15.9 GlobalEvent< TSeq > Class Template Reference	52
15.9.1 Detailed Description	52
15.9.2 Constructor & Destructor Documentation	52
15.9.2.1 GlobalEvent()	52
15.10 LFMCMC< TData > Class Template Reference	53
15.10.1 Detailed Description	54
15.11 Model< TSeq > Class Template Reference	54
15.11.1 Detailed Description	62
15.11.2 Member Function Documentation	62
15.11.2.1 add_globlevent()	62
15.11.2.2 clone_ptr()	63
15.11.2.3 draw()	63
15.11.2.4 events_add()	63
15.11.2.5 events_run()	64
15.11.2.6 load_agents_entities_ties()	64
15.11.2.7 reset()	65
15.11.2.8 run_multiple()	65
15.11.2.9 set_agents_data()	65
15.11.2.10 set_name()	66
15.11.2.11 write_data()	66
15.11.3 Member Data Documentation	67
15.11.3.1 initial_states_fun	67
15.11.3.2 recovery_enhancer_mixer	67
15.11.3.3 rnbinoMd	67
15.11.3.4 runifd	68
15.11.3.5 susceptibility_reduction_mixer	68
15.11.3.6 time_elapsed	68
15.11.3.7 transmission_reduction_mixer	68
15.12 ModelDiagram Class Reference	69
15.13 ModelDiffNet< TSeq > Class Template Reference	69

15.13.1 Detailed Description	70
15.14 ModelMeaslesQuarantine< TSeq > Class Template Reference	70
15.14.1 Detailed Description	72
15.14.2 Constructor & Destructor Documentation	73
15.14.2.1 ModelMeaslesQuarantine()	73
15.14.3 Member Function Documentation	74
15.14.3.1 clone_ptr()	74
15.14.3.2 quarantine_agents()	74
15.14.3.3 reset()	75
15.15 ModelSEIR< TSeq > Class Template Reference	75
15.15.1 Detailed Description	76
15.15.2 Member Function Documentation	77
15.15.2.1 initial_states()	77
15.15.3 Member Data Documentation	77
15.15.3.1 update_exposed_seir	77
15.15.3.2 update_infected_seir	77
15.16 ModelSEIRCONN< TSeq > Class Template Reference	78
15.16.1 Constructor & Destructor Documentation	79
15.16.1.1 ModelSEIRCONN()	79
15.16.2 Member Function Documentation	79
15.16.2.1 initial_states()	79
15.17 ModelSEIRD< TSeq > Class Template Reference	80
15.17.1 Detailed Description	81
15.17.2 Constructor & Destructor Documentation	81
15.17.2.1 ModelSEIRD() [1/2]	81
15.17.2.2 ModelSEIRD() [2/2]	82
15.17.3 Member Data Documentation	82
15.17.3.1 update_exposed_seir	82
15.18 ModelSEIRDCONN< TSeq > Class Template Reference	83
15.18.1 Constructor & Destructor Documentation	84
15.18.1.1 ModelSEIRDCONN()	84
15.18.2 Member Function Documentation	85
15.18.2.1 initial_states()	85
15.19 ModelSEIRMixing< TSeq > Class Template Reference	85
15.19.1 Constructor & Destructor Documentation	86
15.19.1.1 ModelSEIRMixing() [1/2]	87
15.19.1.2 ModelSEIRMixing() [2/2]	87
15.19.2 Member Function Documentation	88
15.19.2.1 initial_states()	88
15.20 ModelSIR< TSeq > Class Template Reference	88
15.20.1 Detailed Description	89
15.20.2 Member Function Documentation	90

15.20.2.1 initial_states()	90
15.21 ModelSIRCONN< TSeq > Class Template Reference	90
15.21.1 Constructor & Destructor Documentation	91
15.21.1.1 ModelSIRCONN()	92
15.21.2 Member Function Documentation	92
15.21.2.1 get_n_infected()	92
15.21.2.2 initial_states()	92
15.22 ModelSIRD< TSeq > Class Template Reference	93
15.22.1 Detailed Description	94
15.22.2 Constructor & Destructor Documentation	94
15.22.2.1 ModelSIRD()	94
15.22.3 Member Function Documentation	95
15.22.3.1 initial_states()	95
15.23 ModelSIRDCONN< TSeq > Class Template Reference	95
15.23.1 Constructor & Destructor Documentation	96
15.23.1.1 ModelSIRDCONN()	96
15.24 ModelSIRLogit< TSeq > Class Template Reference	97
15.24.1 Detailed Description	98
15.24.2 Constructor & Destructor Documentation	99
15.24.2.1 ModelSIRLogit()	99
15.25 ModelSIRMixing< TSeq > Class Template Reference	100
15.25.1 Constructor & Destructor Documentation	101
15.25.1.1 ModelSIRMixing() [1/2]	101
15.25.1.2 ModelSIRMixing() [2/2]	101
15.25.2 Member Function Documentation	102
15.25.2.1 initial_states()	102
15.26 ModelSIS< TSeq > Class Template Reference	102
15.26.1 Detailed Description	103
15.27 ModelSISD< TSeq > Class Template Reference	104
15.27.1 Detailed Description	105
15.28 ModelSURV< TSeq > Class Template Reference	105
15.29 Network< Nettype, Nodetype, Edgetype > Class Template Reference	107
15.30 PersonTools< TSeq > Class Template Reference	108
15.31 Progress Class Reference	108
15.31.1 Detailed Description	108
15.32 Queue< TSeq > Class Template Reference	108
15.32.1 Detailed Description	109
15.33 RandGraph Class Reference	109
15.34 SAMPLETYPE Class Reference	110
15.35 Tool< TSeq > Class Template Reference	110
15.35.1 Detailed Description	111
15.36 ToolFunctions< TSeq > Class Template Reference	111

15.36.1 Detailed Description	112
15.37 Tools< TSeq > Class Template Reference	112
15.37.1 Detailed Description	112
15.38 Tools_const< TSeq > Class Template Reference	113
15.38.1 Detailed Description	113
15.39 UserData< TSeq > Class Template Reference	114
15.39.1 Detailed Description	115
15.39.2 Constructor & Destructor Documentation	115
15.39.2.1 UserData()	115
15.40 vecHasher< T > Struct Template Reference	115
15.40.1 Detailed Description	115
15.41 Virus< TSeq > Class Template Reference	116
15.41.1 Detailed Description	117
15.42 Viruses< TSeq > Class Template Reference	118
15.42.1 Detailed Description	118
15.43 Viruses_const< TSeq > Class Template Reference	119
15.43.1 Detailed Description	119
15.44 VirusFunctions< TSeq > Class Template Reference	119
16 File Documentation	121
16.1 include/epiworld/agent-meat-state.hpp File Reference	121
16.1.1 Detailed Description	122
Index	123

Chapter 1

Example: 00-hello-world

Output from the program:

```
Running the model...
|||||
SIMULATION STUDY
Name of the model      : (none)
Population size        : 10000
Agents' data           : (none)
Number of entities     : 0
Days (duration)        : 100 (of 100)
Number of viruses      : 1
Last run elapsed t     : 17.00ms
Last run speed         : 56.33 million agents x day / second
Rewiring               : off
Global events:
  (none)
Virus(es):
  - covid 19
Tool(s):
  - vaccine
Model parameters:
  (none)
Distribution of the population at time 100:
  - (0) Susceptible : 9950 -> 0
  - (1) Exposed     : 50 -> 0
  - (2) Recovered   : 0 -> 9399
  - (3) Removed     : 0 -> 601
Transition Probabilities:
  - Susceptible 0.92 0.08 - -
  - Exposed - 0.85 0.14 0.01
  - Recovered - - 1.00 -
  - Removed - - - 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 events (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", .01, true);
    covid19.set_infectiousness(.8);

    // Creating a tool
    epiworld::Tool<> vax("vaccine", .5, true);
    vax.set_contagion_reduction(.95);
    // Creating a model
    epiworld::Model<> model;
    // Adding the tool and virus
    model.add_virus(covid19);
    model.add_tool(vax);
    // 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

General parameters

The following are parameters used for both ABM and Compartmental models.

```
EPI_BETA <- 0.75
EPI_GAMMA <- 0.33
EPI_LATENCY <- 1/0.33
EPI_N <- 10000
EPI_0 <- 0.01
EPI_NDAYS <- 50
Sys.setenv( # nolint
  EPI_BETA = EPI_BETA,
  EPI_GAMMA = EPI_GAMMA,
  EPI_LATENCY = EPI_LATENCY,
  EPI_N = EPI_N,
  EPI_0 = EPI_0,
  EPI_NDAYS = EPI_NDAYS
)
```

5.1 Compartmental Models

5.1.1 SIR Model

```
library(deSolve)
library(ggplot2)
library(data.table)
# Code from
# Chapter 2: SIR
# Book "Epidemics: Models and Data using R."
# By: Ottar N. Bjørnstad
sirmod <- function(t, y, parms) {
  # Pull state variables from the vector y
  S = y[1]
  I = y[2]
  R = y[3]

  # Pull parameter values from parms vector
  beta = parms["beta"]
  mu = parms["mu"]
  gamma = parms["gamma"]
  N = parms["N"]

  # Define equations
  dS = mu * (N - S) - beta * S * I/N
  dI = beta * S * I/N - (mu + gamma) * I
  dR = gamma * I - mu * R
  res = c(dS, dI, dR)

  # Return list of gradients
  list(res)
}
# Initial parameters
times <- seq(0, EPI_NDAYS, by = 1)
parms <- c(mu = 0, N = EPI_N, beta = EPI_BETA, gamma = EPI_GAMMA)
start <- c(S = EPI_N * (1 - EPI_0), I = EPI_N * EPI_0, R = 0)
out <- ode(y = start, times = times, func = sirmod, parms = parms)
out <- as.data.frame(out)
out <- rbind(
```

```

with(out, data.table(date = time, state = "Susceptible", counts = S)),
with(out, data.table(date = time, state = "Infected", counts = I)),
with(out, data.table(date = time, state = "Recovered", counts = R))
)

```

Now we visualize the model

```

ggplot(out, aes(x = date, y = counts)) +
  geom_line(aes(colour = state)) +
  labs(title = "Compartmental SIR")

```

5.1.2 SEIR Model

```

# Code adapted from
# Chapter 2: SIR
# Book "Epidemics: Models and Data using R"
# By: Ottar N. Bjørnstad
seirmod <- function(t, y, parms) {
  # Pull state variables from y vector
  S = y[1]
  E = y[2]
  I = y[3]
  R = y[4]

  # Pull parameter values from parms vector
  beta = parms["beta"]
  mu = parms["mu"]
  alpha = parms["alpha"]
  gamma = parms["gamma"]
  N = parms["N"]

  # Define equations
  dS = mu * (N - S) - beta * S * I/N - mu * S
  dE = beta * S * I/N - (mu + alpha) * E
  dI = alpha * E - (mu + gamma) * I
  dR = gamma * I - mu * R
  res = c(dS, dE, dI, dR)

  # Return list of gradients
  list(res)
}
# Initial parameters
parms <- c(
  mu = 0, N = EPI_N, beta = EPI_BETA,
  alpha = 1/EPI_LATENCY, gamma = EPI_GAMMA
)
start <- c(S = EPI_N * (1 - EPI_0), E = EPI_N * EPI_0, I = 0, R = 0)
out_seir <- ode(y = start, times = times, func = seirmod, parms = parms)
out_seir <- as.data.frame(out_seir)
out_seir <- rbind(
  with(out_seir, data.table(date = time, state = "Susceptible", counts = S)),
  with(out_seir, data.table(date = time, state = "Exposed", counts = E)),
  with(out_seir, data.table(date = time, state = "Infected", counts = I)),
  with(out_seir, data.table(date = time, state = "Recovered", counts = R))
)

```

Now we visualize the model

```

ggplot(out_seir, aes(x = date, y = counts)) +
  geom_line(aes(colour = state)) +
  labs(title = "Compartmental SEIR")

```

5.2 Agent-Based Model Approach

Calculation of the expected number of days in state S when prob of changing state equals α is $1/\alpha$

```

set.seed(712)
a <- .3
R <- matrix(runif(2e5 * 50), ncol = 50)
dat <- apply(R, 1, \ (x) {
  which.max(x < a)
})
mean(dat) - 1 / a

```

```
[1] -0.01049333
```

5.2.1 Mathematical preliminaries

That agent i becomes infected can be computed as follows:

At the same time, the probability of not becoming infected equals to the probability of no infected agent transmitting the infection. The probability that agent j infects i equals

In this case, β is parametrized such that its values are within $(0,1)$. Since transmission from the i infected agents happens independently, we finally have the following:

With the above equation, we can now calculate the change in the number of susceptible agents. In this case, it equals the expected number of new infections:

With the same parametrization in the canonical SIR model (Kermack and McKendrick), the instantaneous change in the number of susceptible agents equals $\frac{dS}{dt} = -S \beta I$. Given S and I , we can show that, as $\beta \rightarrow 0$, i.e., the population grows, both rates converge to the same number. Formally:

The same can be shown for the change in the number recovered.

5.2.2 Simulation study

Now, what happens with `epiworld`.

```
system("./09-sir-connected.o -n $EPI_N -b $EPI_BETA -d $EPI_NDAYS -p $EPI_0 -r $EPI_GAMMA -i 1.0 -s555599")
library(ggplot2)
epiworld <- data.table::fread("total_hist.txt")
ggplot(epiworld, aes(x = date, y = counts)) +
  geom_line(aes(colour = state)) +
  labs(title = "ABM SIR")
system("./09-seir-connected.o -n $EPI_N -b $EPI_BETA -d $EPI_NDAYS -p $EPI_0 -r $EPI_GAMMA -i 1.0 -s555599
-l $EPI_LATENCY")
library(ggplot2)
epiworld <- data.table::fread("total_hist.txt")
ggplot(epiworld, aes(x = date, y = counts)) +
  geom_line(aes(colour = state)) +
  labs(title = "ABM SEIR")
```

5.3 Comparing ABM with Compartmental Models

To this end, we will compare the results of the first run of the Compartmental model with 100 runs of the ABM, compute the confidence interval, and see how likely is the compartmental model to fall within the trajectory of the ABM simulation.

5.3.1 SIR

```
system("./09-sir-connected.o -n $EPI_N -b $EPI_BETA -d $EPI_NDAYS -p $EPI_0 -r $EPI_GAMMA -i 1.0 -s555599 -e
100")
library(ggplot2)
library(data.table)
epiworld <- data.table::fread("09-sir-connected-experiments.csv")
epiworld <- epiworld[, .(
  min = quantile(counts, probs = .025),
  mean = mean(counts),
  max = quantile(counts, probs = .975)), by = .(date, state)]
# Merging Compartmental
epiworld <- merge(
  epiworld,
  out[, .(date = date, state = state, compartmental = counts)],
  by = c("date", "state")
)
setorder(epiworld, state, date)
ggplot(epiworld, aes(x = date, y = mean)) +
  geom_ribbon(aes(ymin = min, ymax = max, colour = state), alpha = 0.1) +
  geom_line(aes(x = date, y = compartmental, colour = sprintf("%s (compt)", state)))
```

It seems that, although both yield the same equilibria, compartmental models reach the highest point of the simulation earlier. This makes sense as within a single day of the ABM simulation, compartmental models have more events taking place. Nonetheless, as predicted, as $\beta \rightarrow 0$, the differences become lesser. Furthermore, we could use the fact that the transition rates are known to compute an adjustment.

5.3.2 SEIR

```
system("./09-seir-connected.o -n $EPI_N -b $EPI_BETA -d $EPI_NDAYS -p $EPI_0 -r $EPI_GAMMA -i 1.0 -s555599
-e 100 -l $EPI_LATENCY")
library(ggplot2)
library(data.table)
epiworld_seir <- data.table::fread("09-seir-connected-experiments.csv")
epiworld_seir <- epiworld_seir[, .(
  min = quantile(counts, probs = .025),
  mean = mean(counts),
  max = quantile(counts, probs = .975)), by = .(date, state)]
# Merging Compartmental
epiworld_seir <- merge(
  epiworld_seir,
  out_seir[, .(date = date, state = state, compartmental = counts)],
  by = c("date", "state")
)
setorder(epiworld_seir, state, date)
ggplot(epiworld_seir, aes(x = date, y = mean)) +
  geom_ribbon(aes(ymin = min, ymax = max, colour = state), alpha = 0.1) +
  geom_line(aes(x = date, y = compartmental, colour = sprintf("%s (compt)", state)))
```

5.3.3 Rates

```
S <- 1000
rate_comp <- function(I,B) S * B * I
rate_abm <- function(I,B) S * (1 - (1 - B)^I)
op <- par(mfrow = c(3, 2))
for (i in c(1, 10, 100)) {
  curve(rate_comp(i, x), from = .01, to = 0.05)
  curve(rate_abm(i, x), from = .01, to = 0.05, add = FALSE, lty = 2)
}
par(op)
```

Chapter 6

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 7

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 8

# Mixing probabilities in connected model

George G. Vega Yon, Ph.D. 2024-04-25

### 8.1 Case 1: No grouping

We will look into the probability of drawing infected individuals to simplify the algorithm. There are  $I$  infected individuals at any time in the simulation; thus, instead of drawing from  $\text{Bern}(c/N, N)$ , we will be drawing from  $\text{Bern}(c/N, I)$ . The next step is to check which infected individuals should be drawn. Let's compare the distributions using the hypergeometric as an example:

```
set.seed(132)
nsims <- 1e4
N <- 400
rate <- 5
p <- rate/N
I <- 10
sim_complex <- parallel::mclapply(1:nsims, \(i) {
 nsamples <- rbinom(N, N, p)
 sum(rbinom(N, size = nsamples, prob = I/N) > 0)
}, mc.cores = 4L) |> unlist()
sim_simple <- parallel::mclapply(1:nsims, \(i) {
 sum(rbinom(N, I, p) > 0)
}, mc.cores = 4L) |> unlist()
op <- par(mfrow = c(1,2))
MASS::truehist(sim_complex)
MASS::truehist(sim_simple)
par(op)
quantile(sim_complex)
```

0%	25%	50%	75%	100%
27	43	47	51	71

```
quantile(sim_simple)
```

0%	25%	50%	75%	100%
23	43	47	51	71

```
plotter(sim_complex, sim_simple)
```

These two approaches are equivalent, but the second one is more efficient from the computational perspective.

## 8.2 Case 2: Grouping

This explores the case when we have mixing across groups. The question is if we can replicate the effect at the group level.

```
set.seed(123133)
ngroups <- 3
mixing <- matrix(
 c(0.1, 0.2, 0.3, 0.2, 0.1, 0.2, 0.3, 0.2, 0.1),
 nrow = ngroups,
 ncol = ngroups
)
mixing <- mixing/rowSums(mixing)
mixing

 [,1] [,2] [,3]
[1,] 0.1666667 0.3333333 0.5000000
[2,] 0.4000000 0.2000000 0.4000000
[3,] 0.5000000 0.3333333 0.1666667

N <- 500
sizes <- c(100, 150, 250)
rate <- 5
p <- rate/N
I <- c(10, 30, 20)
ids <- rep.int(1:ngroups, times = sizes)
nsims <- 1e4
sim_complex <- parallel::mclapply(1:nsims, \(i) {
 # Sampling group first
 sapply(1:ngroups, \(g) {
 # How many each individual will sample from the groups
 ans <- rbinom(
 n = N, size = sizes[g], prob = mixing[ids,][,g] * p
) |> sum()
 # Sampling with replacement
 rbinom(ans, size = 1, prob = I[g]/sizes[g]) |> sum()
 }) |> sum()
}, mc.cores = 4L) |> unlist()
```

Using the alternative method in which we directly weight the probabilities:

```
sim_simple <- parallel::mclapply(1:nsims, \(i) {
 # Sampling group first
 sapply(1:ngroups, \(g) {
 rbinom(
 n = N, size = I[g], prob = mixing[cbind(ids,g)] * p
) |> sum()
 }) |> sum()
}, mc.cores = 4L) |> unlist()
op <- par(mfrow = c(1,2))
MASS::truehist(sim_complex)
MASS::truehist(sim_simple)
par(op)
quantile(sim_complex)
```

```
0% 25% 50% 75% 100%
57 88 94 101 131
```

```
quantile(sim_simple)
```

```
0% 25% 50% 75% 100%
58 87 94 101 135
```

```
plotter(sim_complex, sim_simple)
```

## Chapter 9

# EPI Simulator

### 9.1 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.

### 9.2 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.

### 9.3 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.

### 9.4 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

## 9.5 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.

### 9.5.1 Other parameters

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

## Chapter 10

# Namespace Index

### 10.1 Namespace List

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

<a href="#">sampler</a>	Functions for sampling viruses . . . . .	<a href="#">31</a>
-------------------------	------------------------------------------	--------------------





# Chapter 11

## Hierarchical Index

### 11.1 Class Hierarchy

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

AdjList . . . . .	35
Agent< TSeq > . . . . .	36
Agent< EPI_DEFAULT_TSEQ > . . . . .	36
AgentsSample< TSeq > . . . . .	40
DataBase< TSeq > . . . . .	42
Entities< TSeq > . . . . .	46
Entities_const< TSeq > . . . . .	47
Entity< TSeq > . . . . .	48
Entity< EPI_DEFAULT_TSEQ > . . . . .	48
Event< TSeq > . . . . .	50
GlobalEvent< TSeq > . . . . .	52
LFMCMC< TData > . . . . .	53
epiworld::Model	
ModelDiffNet< TSeq > . . . . .	69
ModelSEIR< TSeq > . . . . .	75
ModelSEIRCONN< TSeq > . . . . .	78
ModelSEIRD< TSeq > . . . . .	80
ModelSEIRDCONN< TSeq > . . . . .	83
ModelSEIRMixing< TSeq > . . . . .	85
ModelSIR< TSeq > . . . . .	88
ModelSIRCONN< TSeq > . . . . .	90
ModelSIRD< TSeq > . . . . .	93
ModelSIRDCONN< TSeq > . . . . .	95
ModelSIRLogit< TSeq > . . . . .	97
ModelSIRMixing< TSeq > . . . . .	100
ModelSIS< TSeq > . . . . .	102
ModelISD< TSeq > . . . . .	104
ModelSURV< TSeq > . . . . .	105
Model< TSeq > . . . . .	54
Model< EPI_DEFAULT_TSEQ > . . . . .	54
ModelMeaslesQuarantine< TSeq > . . . . .	70
ModelDiagram . . . . .	69
Network< Nettype, Nodetype, Edgetype > . . . . .	107
PersonTools< TSeq > . . . . .	108
Progress . . . . .	108

Queue< TSeq > . . . . .	108
RandGraph . . . . .	109
SAMPLETYPE . . . . .	110
Tool< TSeq > . . . . .	110
ToolFunctions< TSeq > . . . . .	111
Tools< TSeq > . . . . .	112
Tools_const< TSeq > . . . . .	113
UserData< TSeq > . . . . .	114
vecHasher< T > . . . . .	115
Virus< TSeq > . . . . .	116
Viruses< TSeq > . . . . .	118
Viruses_const< TSeq > . . . . .	119
VirusFunctions< TSeq > . . . . .	119

## Chapter 12

# Class Index

### 12.1 Class List

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

<a href="#">AdjList</a> . . . . .	35
<a href="#">Agent&lt; TSeq &gt;</a>	
<a href="#">Agent</a> (agents) . . . . .	36
<a href="#">AgentsSample&lt; TSeq &gt;</a>	
Sample of agents . . . . .	40
<a href="#">DataBase&lt; TSeq &gt;</a>	
Statistical data about the process . . . . .	42
<a href="#">Entities&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (useful for building iterators) . . . . .	46
<a href="#">Entities_const&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (const) (useful for iterators) . . . . .	47
<a href="#">Entity&lt; TSeq &gt;</a> . . . . .	48
<a href="#">Event&lt; TSeq &gt;</a>	
<a href="#">Event</a> data for update an agent . . . . .	50
<a href="#">GlobalEvent&lt; TSeq &gt;</a>	
Template for a Global <a href="#">Event</a> . . . . .	52
<a href="#">LFMCMC&lt; TData &gt;</a>	
Likelihood-Free Markov Chain Monte Carlo . . . . .	53
<a href="#">Model&lt; TSeq &gt;</a>	
Core class of epiworld . . . . .	54
<a href="#">ModelDiagram</a> . . . . .	69
<a href="#">ModelDiffNet&lt; TSeq &gt;</a>	
Template for a <a href="#">Network</a> Diffusion <a href="#">Model</a> . . . . .	69
<a href="#">ModelMeaslesQuarantine&lt; TSeq &gt;</a>	
Template for a Measles model with quarantine . . . . .	70
<a href="#">ModelSEIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed (SEIR) model . . . . .	75
<a href="#">ModelSEIRCONN&lt; TSeq &gt;</a> . . . . .	78
<a href="#">ModelSEIRD&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model . . . . .	80
<a href="#">ModelSEIRDCONN&lt; TSeq &gt;</a> . . . . .	83
<a href="#">ModelSEIRMixing&lt; TSeq &gt;</a> . . . . .	85
<a href="#">ModelSIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed (SIR) model . . . . .	88
<a href="#">ModelSIRCONN&lt; TSeq &gt;</a> . . . . .	90

<a href="#">ModelSIRD&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed-Deceased (SIRD) model	93
<a href="#">ModelSIRDCONN&lt; TSeq &gt;</a>	95
<a href="#">ModelSIRLogit&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed (SIR) model	97
<a href="#">ModelSIRMixing&lt; TSeq &gt;</a>	100
<a href="#">ModelSIS&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Susceptible (SIS) model	102
<a href="#">ModelSISD&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Susceptible-Deceased (SISD) model	104
<a href="#">ModelSURV&lt; TSeq &gt;</a>	105
<a href="#">Network&lt; Nettype, Nodetype, Edgetype &gt;</a>	107
<a href="#">PersonTools&lt; TSeq &gt;</a>	108
<a href="#">Progress</a>	
A simple progress bar	108
<a href="#">Queue&lt; TSeq &gt;</a>	
Controls which agents are verified at each step	108
<a href="#">RandGraph</a>	109
<a href="#">SAMPLETYPE</a>	110
<a href="#">Tool&lt; TSeq &gt;</a>	
Tools for defending the agent against the virus	110
<a href="#">ToolFunctions&lt; TSeq &gt;</a>	
Helper class to store the functions avoiding multiple shared_pointers (we have only one for the four of these)	111
<a href="#">Tools&lt; TSeq &gt;</a>	
Set of tools (useful for building iterators)	112
<a href="#">Tools_const&lt; TSeq &gt;</a>	
Set of <a href="#">Tools</a> (const) (useful for iterators)	113
<a href="#">UserData&lt; TSeq &gt;</a>	
Personalized data by the user	114
<a href="#">vecHasher&lt; T &gt;</a>	
Vector hasher	115
<a href="#">Virus&lt; TSeq &gt;</a>	
Virus	116
<a href="#">Viruses&lt; TSeq &gt;</a>	
Set of viruses (useful for building iterators)	118
<a href="#">Viruses_const&lt; TSeq &gt;</a>	
Set of <a href="#">Viruses</a> (const) (useful for iterators)	119
<a href="#">VirusFunctions&lt; TSeq &gt;</a>	119

## Chapter 13

# File Index

### 13.1 File List

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

<b>epiworld.hpp</b>	??
include/epiworld/ <b>adjlist-bones.hpp</b>	??
include/epiworld/ <b>adjlist-meat.hpp</b>	??
include/epiworld/ <b>agent-bones.hpp</b>	??
include/epiworld/ <b>agent-events-meat.hpp</b>	??
include/epiworld/ <b>agent-meat-state.hpp</b>	??
Sampling functions are getting big, so we keep them in a separate file	121
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-distribute-meat.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>globalevent-bones.hpp</b>	??
include/epiworld/ <b>globalevent-meat.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>modeldiagram-bones.hpp</b>	??
include/epiworld/ <b>modeldiagram-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-distribute-meat.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-distribute-meat.hpp</b>	??
include/epiworld/ <b>virus-meat.hpp</b>	??
include/epiworld/ <b>viruses-bones.hpp</b>	??
include/epiworld/math/ <b>distributions.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>diffnet.hpp</b>	??
include/epiworld/models/ <b>globalevents.hpp</b>	??
include/epiworld/models/ <b>init-functions.hpp</b>	??
include/epiworld/models/ <b>measlesquarantine.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>seird.hpp</b>	??
include/epiworld/models/ <b>seirdconnected.hpp</b>	??
include/epiworld/models/ <b>seirmixing.hpp</b>	??
include/epiworld/models/ <b>sir.hpp</b>	??
include/epiworld/models/ <b>sirconnected.hpp</b>	??
include/epiworld/models/ <b>sird.hpp</b>	??
include/epiworld/models/ <b>sirdconnected.hpp</b>	??
include/epiworld/models/ <b>sirlogit.hpp</b>	??
include/epiworld/models/ <b>sirmixing.hpp</b>	??
include/epiworld/models/ <b>sis.hpp</b>	??
include/epiworld/models/ <b>sisd.hpp</b>	??
include/epiworld/models/ <b>surveillance.hpp</b>	??
tests/ <b>tests.hpp</b>	??

## Chapter 14

# Namespace Documentation

### 14.1 sampler Namespace Reference

Functions for sampling viruses.

#### Functions

- `template<typename TSeq = EPI_DEFAULT_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 = EPI_DEFAULT_TSEQ>`  
`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 = EPI_DEFAULT_TSEQ>`  
`Virus< TSeq > * sample\_virus\_single (Agent< TSeq > *p, Model< TSeq > *m)`  
*Sample from neighbors pool of viruses (at most one)*

#### 14.1.1 Detailed Description

Functions for sampling viruses.

#### 14.1.2 Function Documentation

##### 14.1.2.1 [make\\_sample\\_virus\\_neighbors\(\)](#)

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

Make a function to sample from neighbors.

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

## Template Parameters

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

## Parameters

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

## Returns

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

## 14.1.2.2 make\_update\_susceptible()

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

Make a function to sample from neighbors.

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

## Template Parameters

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

## Parameters

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

## Returns

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

## 14.1.2.3 sample\_virus\_single()

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

Sample from neighbors pool of viruses (at most one)



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

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

**Template Parameters**

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

**Parameters**

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

**Returns**

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

## Chapter 15

# Class Documentation

### 15.1 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.*

#### 15.1.1 Constructor & Destructor Documentation

##### 15.1.1.1 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

## 15.1.2 Member Function Documentation

### 15.1.2.1 read\_edgelist()

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

Read an edgelist.

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

## Parameters

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

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

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

## 15.2 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](#) ()
- const VirusPtr< TSeq > & [get\\_virus](#) () const
- 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\\_virus](#) ()
- 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 unsigned int & [get\\_state](#) () const
- void [reset](#) ()
- bool [has\\_tool](#) (epiworld\_fast\_uint t) const
- bool [has\\_tool](#) (std::string name) const
- bool [has\\_tool](#) (const [Tool](#)< TSeq > &t) const
- bool [has\\_virus](#) (epiworld\_fast\_uint t) const
- bool [has\\_virus](#) (std::string name) const
- bool [has\\_virus](#) (const [Virus](#)< TSeq > &v) const
- bool [has\\_entity](#) (epiworld\_fast\_uint t) const
- bool [has\\_entity](#) (std::string name) const
- void [print](#) ([Model](#)< TSeq > \*model, bool compressed=false) const
- [Entities](#)< TSeq > [get\\_entities](#) ()
- 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
state_new	state after the change
queue	

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

- void **rm\_tool** (epiworld\_fast\_uint tool\_idx, [Model](#)< TSeq > \*model, epiworld\_fast\_int state\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_tool** (ToolPtr< TSeq > &tool, [Model](#)< TSeq > \*model, epiworld\_fast\_int state\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_virus** ([Model](#)< TSeq > \*model, epiworld\_fast\_int state\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_entity** (epiworld\_fast\_uint entity\_idx, [Model](#)< TSeq > \*model, epiworld\_fast\_int state\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_entity** ([Entity](#)< TSeq > &entity, [Model](#)< TSeq > \*model, epiworld\_fast\_int state\_new=-99, epiworld\_fast\_int queue=-99)
- void **rm\_agent\_by\_virus** ([Model](#)< TSeq > \*model, epiworld\_fast\_int state\_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)

- double & **operator()** (size\_t j)

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

- double & **operator[]** (size\_t j)
- double **operator()** (size\_t j) const
- double **operator[]** (size\_t j) const

### Friends

- class **Model**< TSeq >
- class **Virus**< TSeq >
- class **Tool**< TSeq >
- class **Tools**< TSeq >
- class **Tools\_const**< TSeq >
- class **Queue**< TSeq >
- class **Entities**< TSeq >
- class **AgentsSample**< TSeq >
- void **default\_add\_virus** ([Event](#)< TSeq > &a, [Model](#)< TSeq > \*m)
- void **default\_add\_tool** ([Event](#)< TSeq > &a, [Model](#)< TSeq > \*m)
- void **default\_add\_entity** ([Event](#)< TSeq > &a, [Model](#)< TSeq > \*m)
- void **default\_rm\_virus** ([Event](#)< TSeq > &a, [Model](#)< TSeq > \*m)
- void **default\_rm\_tool** ([Event](#)< TSeq > &a, [Model](#)< TSeq > \*m)
- void **default\_rm\_entity** ([Event](#)< TSeq > &a, [Model](#)< TSeq > \*m)
- void **default\_change\_state** ([Event](#)< TSeq > &a, [Model](#)< TSeq > \*m)

## 15.2.1 Detailed Description

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

[Agent](#) (agents)

Template Parameters

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

## 15.2.2 Member Function Documentation

### 15.2.2.1 operator()()

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

Access the j-th column of the agent.

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

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

Parameters

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

Returns

double&

### 15.2.2.2 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>	

## 15.2.3 Friends And Related Function Documentation

### 15.2.3.1 default\_rm\_entity

```
template<typename TSeq >
void default_rm_entity (
 Event< 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

## 15.3 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, std::vector< size\_t > states\_={}, bool truncate=false)
- **AgentsSample** ([Model](#)< TSeq > \*model, [Entity](#)< TSeq > &entity\_, size\_t n, std::vector< size\_t > states\_←  
\_={}, bool truncate=false)
- [AgentsSample](#) ([Model](#)< TSeq > \*model, [Agent](#)< TSeq > &agent\_, size\_t n, std::vector< size\_t > states\_←  
\_={}, 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



### 15.3.1 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>	
-------------	--

### 15.3.2 Constructor & Destructor Documentation

#### 15.3.2.1 AgentsSample()

```
template<typename TSeq >
AgentsSample< TSeq >::AgentsSample (
 Model< TSeq > * model,
 Agent< TSeq > & agent_,
 size_t n,
 std::vector< size_t > states_ = {},
 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

## 15.4 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\\_virus](#) ([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\_virus\_info, std::string fn\_virus\_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 virus, int i\_expo\_date)
- size\_t [get\\_n\\_viruses](#) () const  
*Get the number of viruses.*
- size\_t [get\\_n\\_tools](#) () const  
*Get the number of tools.*
- 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 > [get\\_transition\\_probability](#) (bool print=true, bool normalize=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

### 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\\_virus](#), the current counts of what for each virus.*

*In [get\\_hist\\_total](#), the time series of what*

*In [get\\_hist\\_virus](#), the time series of what for each virus.*

*In [get\\_hist\\_total\\_date](#) and [get\\_hist\\_virus\\_date](#) the corresponding date*

- 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\_virus** (std::vector< std::string > &state, std::vector< int > &id, std::vector< int > &counts) const
  - void **get\_today\_transition\_matrix** (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\_virus** (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
- 
- void **get\_transmissions** (std::vector< int > &date, std::vector< int > &source, std::vector< int > &target, std::vector< int > &virus, std::vector< int > &source\_exposure\_date) const  
*Get the transmissions object.*
  - void **get\_transmissions** (int \*date, int \*source, int \*target, int \*virus, int \*source\_exposure\_date) const
- 
- MapVec\_type< int, int > **get\_reproductive\_number** () const  
*Computes the reproductive number of each case.*
  - void **get\_reproductive\_number** (std::string fn) const
- 
- void **get\_generation\_time** (std::vector< int > &agent\_id, std::vector< int > &virus\_id, std::vector< int > &time, std::vector< int > &gentime) const  
*Get the generation time.*
  - void **get\_generation\_time** (std::string fn) const  
*Write the generation time to a file.*

## Friends

- class **Model**< TSeq >
- void **default\_add\_virus** (Event< TSeq > &a, Model< TSeq > \*m)
- void **default\_add\_tool** (Event< TSeq > &a, Model< TSeq > \*m)
- void **default\_rm\_virus** (Event< TSeq > &a, Model< TSeq > \*m)
- void **default\_rm\_tool** (Event< TSeq > &a, Model< TSeq > \*m)
- void **default\_change\_state** (Event< TSeq > &a, Model< TSeq > \*m)

### 15.4.1 Detailed Description

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

Statistical data about the process.

## Template Parameters

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

## 15.4.2 Member Function Documentation

### 15.4.2.1 `get_generation_time()`

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

Get the generation time.

Calculates the generating time

## Parameters

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

The generation time is the time between the infection of the source and the infection of the target.

### 15.4.2.2 `get_reproductive_number()`

```
template<typename TSeq >
MapVec_type< int, int > DataBase< TSeq >::get_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.
-----------	--------------------------------------------------

In the case of `MapVec_type<int, int>`, the key is a vector of 3 integers:

- [Virus](#) id
- Source id
- Date when the source was infected

### 15.4.2.3 get\_transition\_probability()

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

Calculates the transition probabilities.

#### Parameters

<i>print</i>	Print the transition matrix.
<i>normalize</i>	Normalize the transition matrix. Otherwise, it returns raw counts.

The transition matrix is the matrix of the counts of transitions from one state to another. So the ij-th element of the matrix is the number of transitions from state i to state j (when not normalized), or the probability of transitioning from state i to state j (when normalized).

#### Returns

std::vector< epiworld\_double >

### 15.4.2.4 get\_transmissions()

```
template<typename TSeq >
void DataBase< TSeq >::get_transmissions (
 std::vector< int > & date,
 std::vector< int > & source,
 std::vector< int > & target,
 std::vector< int > & virus,
 std::vector< int > & source_exposure_date) const [inline]
```

Get the transmissions object.

#### Parameters

<i>date</i>	
<i>source</i>	
<i>target</i>	
<i>virus</i>	
<i>source_exposure_date</i>	

### 15.4.2.5 operator==( ) [1/2]

```
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,

#### 15.4.2.6 operator==( ) [2/2]

```
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

#### 15.4.2.7 record\_virus()

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

Registering a new variant.

##### Parameters

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

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

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

## 15.5 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 >

### 15.5.1 Detailed Description

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

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

#### Template Parameters

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

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

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

## 15.6 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 >

### 15.6.1 Detailed Description

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

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

Template Parameters

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

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

- include/epiworld/entities-bones.hpp

## 15.7 Entity< TSeq > Class Template Reference

### Public Member Functions

- [Entity](#) (std::string name, EntityToAgentFun< TSeq > fun=nullptr)  
*Constructs an [Entity](#) object.*
- void **add\_agent** ([Agent](#)< TSeq > &p, [Model](#)< TSeq > \*model)
- void **add\_agent** ([Agent](#)< TSeq > \*p, [Model](#)< TSeq > \*model)
- void **rm\_agent** (size\_t idx, [Model](#)< TSeq > \*model)
- 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
- size\_t **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

#### Entity distribution

*These functions are used for distributing agents among entities. The idea is to have a flexible way of distributing agents among entities.*

- void **distribute** ([Model](#)< TSeq > \*model)
- std::vector< size\_t > & **get\_agents** ()
- void **print** () const
- void **set\_distribution** (EntityToAgentFun< TSeq > fun)



## Friends

- class **Agent**< TSeq >
- class **AgentsSample**< TSeq >
- class **Model**< TSeq >
- void **default\_add\_entity** (Event< TSeq > &a, Model< TSeq > \*m)
- void **default\_rm\_entity** (Event< TSeq > &a, Model< TSeq > \*m)

## 15.7.1 Constructor & Destructor Documentation

### 15.7.1.1 Entity()

```
template<typename TSeq >
Entity< TSeq >::Entity (
 std::string name,
 EntityToAgentFun< TSeq > fun = nullptr) [inline]
```

Constructs an [Entity](#) object.

This constructor initializes an [Entity](#) object with the specified parameters.

#### Parameters

<i>name</i>	The name of the entity.
<i>fun</i>	A function pointer to a function that maps the entity to an agent.

## 15.7.2 Friends And Related Function Documentation

### 15.7.2.1 default\_rm\_entity

```
template<typename TSeq >
void default_rm_entity (
 Event< 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

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

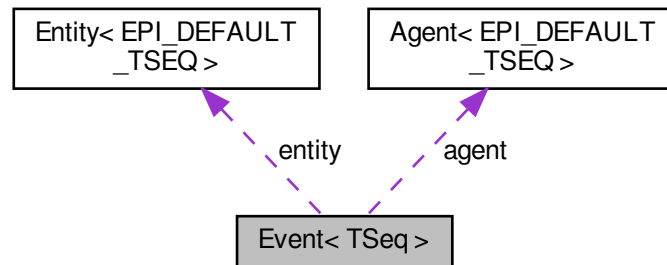
- include/epiworld/agent-bones.hpp
- include/epiworld/entity-bones.hpp
- include/epiworld/entity-meat.hpp

## 15.8 Event< TSeq > Struct Template Reference

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

```
#include <config.hpp>
```

Collaboration diagram for Event< TSeq >:



### Public Member Functions

- [Event](#) ([Agent](#)< TSeq > \*agent\_, VirusPtr< TSeq > virus\_, ToolPtr< TSeq > tool\_, [Entity](#)< TSeq > \*entity\_, epiworld\_fast\_int new\_state\_, epiworld\_fast\_int queue\_, EventFun< TSeq > call\_, int idx\_agent\_, int idx\_object\_)

Construct a new [Event](#) object.

### Public Attributes

- [Agent](#)< TSeq > \* **agent**
- VirusPtr< TSeq > **virus**
- ToolPtr< TSeq > **tool**
- [Entity](#)< TSeq > \* **entity**
- epiworld\_fast\_int **new\_state**
- epiworld\_fast\_int **queue**
- EventFun< TSeq > **call**
- int **idx\_agent**
- int **idx\_object**

#### 15.8.1 Detailed Description

```
template<typename TSeq = EPI_DEFAULT_TSEQ>
struct Event< TSeq >
```

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

## Template Parameters

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

## 15.8.2 Constructor &amp; Destructor Documentation

## 15.8.2.1 Event()

```
template<typename TSeq = EPI_DEFAULT_TSEQ>
Event< TSeq >::Event (
 Agent< TSeq > * agent_,
 VirusPtr< TSeq > virus_,
 ToolPtr< TSeq > tool_,
 Entity< TSeq > * entity_,
 epiworld_fast_int new_state_,
 epiworld_fast_int queue_,
 EventFun< TSeq > call_,
 int idx_agent_,
 int idx_object_) [inline]
```

Construct a new [Event](#) 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

## 15.9 GlobalEvent< TSeq > Class Template Reference

Template for a Global [Event](#).

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

### Public Member Functions

- [GlobalEvent](#) (GlobalFun< TSeq > fun, std::string name, int day=-99)  
*Construct a new Global [Event](#) object.*
- void **operator()** ([Model](#)< TSeq > \*m, int day)
- void **set\_name** (std::string name)
- std::string **get\_name** () const
- void **set\_day** (int day)
- int **get\_day** () const
- void **print** () const
- bool **operator==** (const [GlobalEvent](#)< TSeq > &other) const
- bool **operator!=** (const [GlobalEvent](#)< TSeq > &other) const

### 15.9.1 Detailed Description

```
template<typename TSeq>
class GlobalEvent< TSeq >
```

Template for a Global [Event](#).

Global events are functions that [Model](#)<TSeq> executes at the end of a day.

### 15.9.2 Constructor & Destructor Documentation

#### 15.9.2.1 GlobalEvent()

```
template<typename TSeq >
GlobalEvent< TSeq >::GlobalEvent (
 GlobalFun< TSeq > fun,
 std::string name,
 int day = -99) [inline]
```

Construct a new Global [Event](#) object.

#### Parameters

<i>fun</i>	A function that takes a <a href="#">Model</a> <TSeq> * as argument and returns void.
<i>name</i>	A descriptive name for the action.
<i>day</i>	The day when the action will be executed. If negative, it will be executed every day.

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

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

## 15.10 LFMCMC< TData > Class Template Reference

Likelihood-Free Markov Chain Monte Carlo.

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

### Public Member Functions

- void **run** (std::vector< epiworld\_double > params\_init\_, size\_t n\_samples\_, epiworld\_double epsilon\_, int seed=-1)
- **LFMCMC** (const TData &observed\_data\_)
- void **set\_observed\_data** (const 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)
- void **set\_params\_names** (std::vector< std::string > names)
- void **set\_stats\_names** (std::vector< std::string > names)
- size\_t **get\_n\_samples** () const
- size\_t **get\_n\_stats** () const
- size\_t **get\_n\_params** () const
- epiworld\_double **get\_epsilon** () const
- const std::vector< epiworld\_double > & **get\_initial\_params** () const
- const std::vector< epiworld\_double > & **get\_current\_proposed\_params** () const
- const std::vector< epiworld\_double > & **get\_current\_accepted\_params** () const
- const std::vector< epiworld\_double > & **get\_current\_proposed\_stats** () const
- const std::vector< epiworld\_double > & **get\_current\_accepted\_stats** () const
- const std::vector< epiworld\_double > & **get\_observed\_stats** () const
- const std::vector< epiworld\_double > & **get\_all\_sample\_params** () const
- const std::vector< epiworld\_double > & **get\_all\_sample\_stats** () const
- const std::vector< bool > & **get\_all\_sample\_acceptance** () const
- const std::vector< epiworld\_double > & **get\_all\_sample\_drawn\_prob** () const
- const std::vector< epiworld\_double > & **get\_all\_sample\_kernel\_scores** () const
- const std::vector< epiworld\_double > & **get\_all\_accepted\_params** () const
- const std::vector< epiworld\_double > & **get\_all\_accepted\_stats** () const
- const std::vector< epiworld\_double > & **get\_all\_accepted\_kernel\_scores** () const
- std::vector< TData > \* **get\_simulated\_data** () const
- std::vector< epiworld\_double > **get\_mean\_params** ()
- std::vector< epiworld\_double > **get\_mean\_stats** ()
- **LFMCMC**< TData > & **verbose\_off** ()
- **LFMCMC**< TData > & **verbose\_on** ()
- void **print** (size\_t burnin=0u) const

### Random number generation

*Parameters*

eng	
-----	--

- void **set\_rand\_engine** (std::shared\_ptr< std::mt19937 > &eng)
- std::shared\_ptr< 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)

**15.10.1 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:

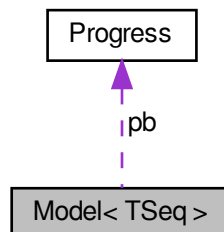
- include/epiworld/math/lfmcmc/lfmcmc-bones.hpp
- include/epiworld/math/lfmcmc/lfmcmc-meat-print.hpp
- include/epiworld/math/lfmcmc/lfmcmc-meat.hpp

**15.11 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** ()
- const [DataBase](#)< TSeq > & **get\_db** () const
- 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.*
- void **load\_agents\_entities\_ties** (const std::vector< int > &agents\_ids, const std::vector< int > &entities\_ids)  
*Associate agents-entities from data.*
- void **load\_agents\_entities\_ties** (const int \*agents\_id, const int \*entities\_id, size\_t n)
- size\_t **get\_n\_viruses** () const  
*Number of viruses in the model.*
- size\_t **get\_n\_tools** () const  
*Number of tools in the model.*
- epiworld\_fast\_uint **get\_ndays** () const
- epiworld\_fast\_uint **get\_n\_replicates** () const
- void **set\_ndays** (epiworld\_fast\_uint ndays)
- bool **get\_verbose** () const
- [Model](#)< TSeq > & **verbose\_off** ()
- [Model](#)< TSeq > & **verbose\_on** ()
- int **today** () const  
*The current time of the model.*
- void **write\_data** (std::string fn\_virus\_info, std::string fn\_virus\_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** ()
- virtual void **reset** ()  
*Reset the model.*
- const [Model](#)< TSeq > & **print** (bool lite=false) 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\_globalevent** (std::function< void([Model](#)< TSeq > \*)> fun, std::string name="A global action", int date=-99)  
*Set a global action.*
- void **add\_globalevent** ([GlobalEvent](#)< TSeq > action)
- [GlobalEvent](#)< TSeq > & **get\_globalevent** (std::string name)  
*Retrieve a global action by name.*
- [GlobalEvent](#)< TSeq > & **get\_globalevent** (size\_t i)  
*Retrieve a global action by index.*
- void **rm\_globalevent** (std::string name)  
*Remove a global action by name.*
- void **rm\_globalevent** (size\_t i)  
*Remove a global action by index.*
- void **run\_globalevents** ()
- void **clear\_state\_set** ()
- const std::vector< [VirusPtr](#)< TSeq > > & **get\_viruses** () const
- const std::vector< [ToolPtr](#)< TSeq > > & **get\_tools** () const
- [Virus](#)< TSeq > & **get\_virus** (size\_t id)
- [Tool](#)< TSeq > & **get\_tool** (size\_t id)
- void **set\_agents\_data** (double \*data\_, size\_t ncols\_)  
*Set the agents data object.*

- double \* **get\_agents\_data** ()
- size\_t **get\_agents\_data\_ncols** () const
- 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
- void **events\_run** ()  
*Executes the stored action.*
- void **draw** (const std::string &fn\_output="", bool self=false)  
*Draws a mermaid diagram of the model.*

### 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::shared\_ptr< std::mt19937 > &eng)
- std::shared\_ptr< 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)
- void **set\_rand\_binom** (int n, epiworld\_double p)
- void **set\_rand\_nbinom** (int n, epiworld\_double p)
- void **set\_rand\_geom** (epiworld\_double p)
- void **set\_rand\_pois** (epiworld\_double lambda)
- 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)
- int **rbinom** ()
- int **rbinom** (int n, epiworld\_double p)
- int **rnbinom** ()
- int **rnbinom** (int n, epiworld\_double p)
- int **rgeom** ()
- int **rgeom** (epiworld\_double p)
- int **rpois** ()
- int **rpois** (epiworld\_double lambda)

### Add Virus/Tool to the model

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



*Parameters*

v	<i>Virus to be added</i>
t	<i>Tool 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)
- void **add\_tool** (Tool< TSeq > &t)
- 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\_id)

**Accessing population of the model***Parameters*

fn	<i>std::string Filename of the edgelist file.</i>
skip	<i>int Number of lines to skip in fn.</i>
directed	<i>bool Whether the graph is directed or not.</i>
size	<i>Size of the network.</i>
al	<i>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** ()  
*Returns a reference to the vector of agents.*
- Agent< TSeq > & **get\_agent** (size\_t i)
- std::vector< epiworld\_fast\_uint > **get\_agents\_states** () const  
*Returns a vector with the states of the agents.*
- std::vector< Viruses\_const< TSeq > > & **get\_agents\_viruses** () const  
*Returns a const vector with the viruses of the agents.*
- std::vector< Viruses< TSeq > > & **get\_agents\_viruses** ()  
*Returns a vector with the viruses of the agents.*
- std::vector< Entity< TSeq > > & **get\_entities** ()
- Entity< TSeq > & **get\_entity** (size\_t entity\_id, int \*entity\_pos=nullptr)
- Model< TSeq > & **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 run_multiple, a function that is called after each experiment.</i>

- void **update\_state** ()

- void **mutate\_virus** ()
- void **next** ()
- virtual **Model**< TSeq > & **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	Proportion of ties to be rewired.
------------	-----------------------------------

#### Returns

A rewired version of the network.

- void **set\_rewire\_fun** (std::function< void(std::vector< **Agent**< TSeq >> \*, **Model**< TSeq > \*, epiworld\_double)> fun)
- void **set\_rewire\_prop** (epiworld\_double prop)
- epiworld\_double **get\_rewire\_prop** () const
- void **rewire** ()

### Export the network data in edgelist form

#### Parameters

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

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

- void **write\_edgelist** (std::string fn) const
- void **write\_edgelist** (std::vector< int > &source, std::vector< int > &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	std::string 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\_states** () const
- const std::vector< UpdateFun< TSeq > > & **get\_state\_fun** () const
- void **print\_state\_codes** () const

### Initial states

*These functions are called before the simulation starts.*

#### Parameters

proportions↔	Vector of proportions for each state.
—	
queue_	Vector of queue for each state.

- virtual [Model](#)< TSeq > & **initial\_states** (std::vector< double >, std::vector< int >)

### 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, `bool` overwrite=false)
- [Model](#)< TSeq > & **read\_params** (`std::string` fn, `bool` overwrite=false)
- `epiworld_double` **get\_param** (`epiworld_fast_uint` k)
- `epiworld_double` **get\_param** (`std::string` pname)
- `void` **set\_param** (`std::string` pname, `epiworld_double` val)
- `epiworld_double` **par** (`std::string` pname) const

### Set the user data object

#### Parameters

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

- `void` [set\\_user\\_data](#) (`std::vector< std::string >` names)
- `[@`
- `void` **add\_user\_data** (`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 [queueing\\_on](#) ()  
*Activates the queueing system (default.)*
- [Model](#)< TSeq > & [queueing\\_off](#) ()  
*Deactivates the queueing system.*
- bool [is\\_queueing\\_on](#) () const  
*Query if the queueing system is on.*
- [Queue](#)< TSeq > & [get\\_queue](#) ()  
*Retrieve the [Queue](#) object.*

### Get the susceptibility reduction object

#### Parameters

v	
---	--

#### Returns

*epiworld\_double*

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

### Protected Member Functions

- void [dist\\_tools](#) ()
- void [dist\\_virus](#) ()
- void [dist\\_entities](#) ()
- void [chrono\\_start](#) ()
- void [chrono\\_end](#) ()
- void [events\\_add](#) ([Agent](#)< TSeq > \*agent\_, VirusPtr< TSeq > virus\_, ToolPtr< TSeq > tool\_, [Entity](#)< TSeq > \*entity\_, epiworld\_fast\_int new\_state\_, epiworld\_fast\_int queue\_, EventFun< TSeq > call\_, int idx\_↵ agent\_, int idx\_object\_)  
*Construct a new [Event](#) object.*

### 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< ToolPtr< TSeq > > [tools](#) = {}
- std::vector< [Entity](#)< TSeq > > [entities](#) = {}
- std::vector< [Entity](#)< TSeq > > [entities\\_backup](#) = {}
- std::shared\_ptr< std::mt19937 > [engine](#) = std::make\_shared<std::mt19937>()
- std::uniform\_real\_distribution [runifd](#)

- `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::binomial_distribution` **rbinomd** = `std::binomial_distribution<>()`
- `std::negative_binomial_distribution` **rnbinomd**
- `std::geometric_distribution` **rgeomd** = `std::geometric_distribution<>()`
- `std::poisson_distribution` **rpoissd** = `std::poisson_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 >>` **state\_fun** = {}  
*Functions to update states.*
- `std::vector< std::string >` **states\_labels** = {}  
*Labels of the states.*
- `std::function< void(Model< TSeq > *)>` **initial\_states\_fun**
- `epiworld_fast_uint` **nstates** = 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< GlobalEvent< TSeq >>` **globalevents**
- `Queue< TSeq >` **queue**
- `bool` **use\_queueing** = true
- `std::vector< Event< TSeq >>` **events** = {}  
*Variables used to keep track of the events 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 *` **agents\_data** = nullptr
- `size_t` **agents\_data\_ncols** = 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**
- MixerFun< TSeq > **transmission\_reduction\_mixer**
- MixerFun< TSeq > **recovery\_enhancer\_mixer**
- 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)
- **Model** ([Model](#)< TSeq > &&m)
- [Model](#)< TSeq > & **operator=** (const [Model](#)< TSeq > &m)
- virtual ~**Model** ()

### 15.11.1 Detailed Description

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

Core class of epiworld.

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

#### Template Parameters

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

### 15.11.2 Member Function Documentation

#### 15.11.2.1 add\_globalevent()

```
template<typename TSeq >
void Model< TSeq >::add_globalevent (
 std::function< void(Model< TSeq > *)> fun,
 std::string name = "A global action",
 int date = -99) [inline]
```

Set a global action.

## Parameters

<i>fun</i>	A function to be called on the prescribed date
<i>name</i>	Name of the action.
<i>date</i>	Integer indicating when the function is called (see details)

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

## 15.11.2.2 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 [ModelMeaslesQuarantine< TSeq >](#).

## 15.11.2.3 draw()

```
template<typename TSeq >
void Model< TSeq >::draw (
 const std::string & fn_output = "",
 bool self = false) [inline]
```

Draws a mermaid diagram of the model.

## Parameters

<i>model</i>	The model to draw.
<i>fn_output</i>	The name of the file to write the diagram. If empty, the diagram will be printed to the standard output.
<i>self</i>	Whether to allow self-transitions.

## 15.11.2.4 events\_add()

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

```

Entity< TSeq > * entity_,
epiworld_fast_int new_state_,
epiworld_fast_int queue_,
EventFun< TSeq > call_,
int idx_agent_,
int idx_object_) [inline], [protected]

```

Construct a new [Event](#) 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.

#### 15.11.2.5 events\_run()

```

template<typename TSeq >
void Model< TSeq >::events_run [inline]

```

Executes the stored action.

#### Parameters

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

#### 15.11.2.6 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.

**15.11.2.7 reset()**

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

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

Reimplemented in [ModelMeaslesQuarantine< TSeq >](#).

**15.11.2.8 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
--------------	---------------------------------

**15.11.2.9 set\_agents\_data()**

```
template<typename TSeq >
```

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

Set the agents data object.

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

#### Parameters

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

#### 15.11.2.10 set\_name()

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

Set the name object.

#### Parameters

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

#### 15.11.2.11 write\_data()

```
template<typename TSeq >
void Model< TSeq >::write_data (
 std::string fn_virus_info,
 std::string fn_virus_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_virus_info</i>	Filename. Information about the virus.
<i>fn_virus_hist</i>	Filename. History of the virus.
<i>fn_tool_info</i>	Filename. Information about the tool.

## Parameters

<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

### 15.11.3 Member Data Documentation

#### 15.11.3.1 initial\_states\_fun

```
template<typename TSeq >
std::function<void (Model<TSeq>*)> Model< TSeq >::initial_states_fun [protected]
```

## Initial value:

```
=
 [] (Model<TSeq>*) -> void {}
```

Function to distribute states. Goes along with the function

#### 15.11.3.2 recovery\_enhancer\_mixer

```
template<typename TSeq >
MixerFun<TSeq> Model< TSeq >::recovery_enhancer_mixer [protected]
```

## Initial value:

```
=
 recovery_enhancer_mixer_default<TSeq>
```

#### 15.11.3.3 rnbinomd

```
template<typename TSeq >
std::negative_binomial_distribution Model< TSeq >::rnbinomd [protected]
```

## Initial value:

```
=
 std::negative_binomial_distribution<>()
```

#### 15.11.3.4 runifd

```
template<typename TSeq >
std::uniform_real_distribution Model< TSeq >::runifd [protected]
```

##### Initial value:

```
=
 std::uniform_real_distribution<>(0.0, 1.0)
```

#### 15.11.3.5 susceptibility\_reduction\_mixer

```
template<typename TSeq >
MixerFun<TSeq> Model< TSeq >::susceptibility_reduction_mixer [protected]
```

##### Initial value:

```
=
 susceptibility_reduction_mixer_default<TSeq>
```

#### 15.11.3.6 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()
```

#### 15.11.3.7 transmission\_reduction\_mixer

```
template<typename TSeq >
MixerFun<TSeq> Model< TSeq >::transmission_reduction_mixer [protected]
```

##### Initial value:

```
=
 transmission_reduction_mixer_default<TSeq>
```

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

## 15.12 ModelDiagram Class Reference

### Public Member Functions

- void **draw\_from\_data** (const std::vector< std::string > &states, const std::vector< epiworld\_double > &tprob, const std::string &fn\_output="", bool self=false)
- void **draw\_from\_file** (const std::string &fn\_transition, const std::string &fn\_output="", bool self=false)
- void **draw\_from\_files** (const std::vector< std::string > &fns\_transition, const std::string &fn\_output="", bool self=false)

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

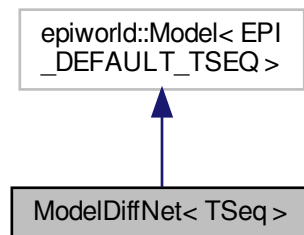
- include/epiworld/modeldiagram-bones.hpp
- include/epiworld/modeldiagram-meat.hpp

## 15.13 ModelDiffNet< TSeq > Class Template Reference

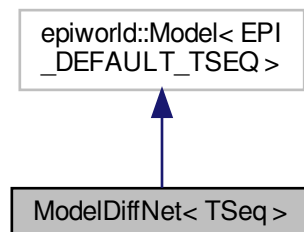
Template for a [Network](#) Diffusion [Model](#).

```
#include <diffnet.hpp>
```

Inheritance diagram for ModelDiffNet< TSeq >:



Collaboration diagram for ModelDiffNet< TSeq >:



## Public Member Functions

- **ModelDiffNet** ([ModelDiffNet](#)< TSeq > &model, const std::string &innovation\_name, epiworld\_double prevalence, epiworld\_double prob\_adopt, bool normalize\_exposure=true, double \*agents\_data=nullptr, size\_t data\_ncols=0u, std::vector< size\_t > data\_cols={}, std::vector< double > params={})
- **ModelDiffNet** (const std::string &innovation\_name, epiworld\_double prevalence, epiworld\_double prob\_adopt, bool normalize\_exposure=true, double \*agents\_data=nullptr, size\_t data\_ncols=0u, std::vector< size\_t > data\_cols={}, std::vector< double > params={})

## Public Attributes

- bool **normalize\_exposure** = true
- std::vector< size\_t > **data\_cols**
- std::vector< double > **params**

## Static Public Attributes

- static const int **NONADOPTER** = 0
- static const int **ADOPTER** = 1

### 15.13.1 Detailed Description

```
template<typename TSeq = EPI_DEFAULT_TSEQ>
class ModelDiffNet< TSeq >
```

Template for a [Network](#) Diffusion [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:

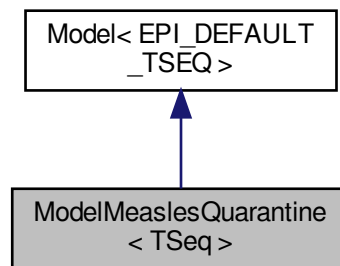
- include/epiworld/models/diffnet.hpp

## 15.14 ModelMeaslesQuarantine< TSeq > Class Template Reference

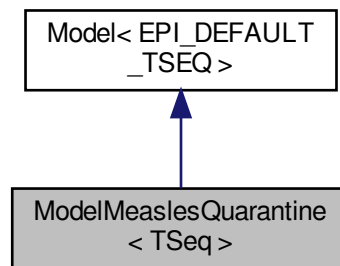
Template for a Measles model with quarantine.

```
#include <measlesquarantine.hpp>
```

Inheritance diagram for ModelMeaslesQuarantine< TSeq >:



Collaboration diagram for ModelMeaslesQuarantine< TSeq >:



## Public Member Functions

- void [quarantine\\_agents](#) ()  
*Quarantine agents that are in the system.*
- void [reset](#) ()  
*Reset the model.*
- void [update\\_infectious](#) ()
- [Model](#)< TSeq > \* [clone\\_ptr](#) ()  
*Advanced usage: Makes a copy of data and returns it as undeleted pointer.*
- [ModelMeaslesQuarantine](#) ([ModelMeaslesQuarantine](#)< TSeq > &model, epiworld\_fast\_uint n, epiworld\_double n\_exposed, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double vax\_efficacy, epiworld\_double vax\_reduction\_recovery\_rate, epiworld\_double incubation\_period, epiworld\_double prodromal\_period, epiworld\_double rash\_period, epiworld\_double days\_undetected, epiworld\_double hospitalization\_rate, epiworld\_double hospitalization\_period, epiworld\_double prop\_vaccinated, epiworld\_fast\_int quarantine\_period, epiworld\_double quarantine\_willingness, epiworld\_fast\_int isolation\_period)

- **ModelMeaslesQuarantine** (epiworld\_fast\_uint n, epiworld\_fast\_uint n\_exposed, epiworld\_double contact↔\_rate, epiworld\_double transmission\_rate, epiworld\_double vax\_efficacy, epiworld\_double vax\_reduction↔\_recovery\_rate, epiworld\_double incubation\_period, epiworld\_double prodromal\_period, epiworld\_double rash\_period, epiworld\_double days\_undetected, epiworld\_double hospitalization\_rate, epiworld\_double hospitalization\_period, epiworld\_double prop\_vaccinated, epiworld\_fast\_int quarantine\_period, epiworld\_double quarantine\_willingness, epiworld\_fast\_int isolation\_period)

## Public Attributes

- std::vector< [Agent](#)< TSeq > \* > [infectious](#)  
*Agents infectious for contact.*
- bool **system\_quarantine\_triggered** = false
- std::vector< int > [day\\_flagged](#)  
*Either detected or started quarantine.*
- std::vector< int > [day\\_rash\\_onset](#)  
*Day of rash onset.*

## Static Public Attributes

- static const epiworld\_fast\_uint **SUSCEPTIBLE** = 0u
- static const epiworld\_fast\_uint **EXPOSED** = 1u
- static const epiworld\_fast\_uint **PRODROMAL** = 2u
- static const epiworld\_fast\_uint **RASH** = 3u
- static const epiworld\_fast\_uint **ISOLATED** = 4u
- static const epiworld\_fast\_uint **ISOLATED\_RECOVERED** = 5u
- static const epiworld\_fast\_uint **DETECTED\_HOSPITALIZED** = 6u
- static const epiworld\_fast\_uint **QUARANTINED\_EXPOSED** = 7u
- static const epiworld\_fast\_uint **QUARANTINED\_SUSCEPTIBLE** = 8u
- static const epiworld\_fast\_uint **QUARANTINED\_PRODROMAL** = 9u
- static const epiworld\_fast\_uint **QUARANTINED\_RECOVERED** = 10u
- static const epiworld\_fast\_uint **HOSPITALIZED** = 11u
- static const epiworld\_fast\_uint **RECOVERED** = 12u

## Additional Inherited Members

### 15.14.1 Detailed Description

```
template<typename TSeq = EPI_DEFAULT_TSEQ>
class ModelMeaslesQuarantine< TSeq >
```

Template for a Measles model with quarantine.

#### Parameters

<i>TSeq</i>	The type of the sequence to be used.
-------------	--------------------------------------

This model can be described as a SEIHR model with isolation and quarantine. The infectious state is divided into prodromal and rash phases. Furthermore, the quarantine state includes exposed, susceptible, prodromal, and recovered states.



The quarantine process is triggered any time that an agent with rash is detected. The agent is then isolated and all agents who are unvaccinated are quarantined. Isolated agents then may be moved out of the isolation in `isolation_↵_period` days.

## 15.14.2 Constructor & Destructor Documentation

### 15.14.2.1 ModelMeaslesQuarantine()

```
template<typename TSeq >
ModelMeaslesQuarantine< TSeq >::ModelMeaslesQuarantine (
 ModelMeaslesQuarantine< TSeq > & model,
 epiworld_fast_uint n,
 epiworld_fast_uint n_exposed,
 epiworld_double contact_rate,
 epiworld_double transmission_rate,
 epiworld_double vax_efficacy,
 epiworld_double vax_reduction_recovery_rate,
 epiworld_double incubation_period,
 epiworld_double prodromal_period,
 epiworld_double rash_period,
 epiworld_double days_undetected,
 epiworld_double hospitalization_rate,
 epiworld_double hospitalization_period,
 epiworld_double prop_vaccinated,
 epiworld_fast_int quarantine_period,
 epiworld_double quarantine_willingness,
 epiworld_fast_int isolation_period) [inline]
```

#### Parameters

<i>n</i>	The number of agents in the system.
<i>n_exposed</i>	The number of exposed agents in the system.
<i>contact_rate</i>	The rate of contact between agents.
<i>transmission_rate</i>	The rate of transmission of the virus.
<i>vax_efficacy</i>	The efficacy of the vaccine.
<i>vax_reduction_recovery_rate</i>	The reduction in recovery rate due to the vaccine.
<i>incubation_period</i>	The incubation period of the virus.
<i>prodromal_period</i>	The prodromal period of the virus.
<i>rash_period</i>	The rash period of the virus.
<i>days_undetected</i>	The number of days the virus goes undetected.
<i>hospitalization_rate</i>	The rate of hospitalization.
<i>hospitalization_period</i>	The duration of hospitalization.
<i>prop_vaccinated</i>	The proportion of vaccinated agents.
<i>quarantine_period</i>	The number of days for quarantine.
<i>quarantine_willingness</i>	The willingness to be quarantined.
<i>isolation_period</i>	The number of days for isolation.

### 15.14.3 Member Function Documentation

#### 15.14.3.1 clone\_ptr()

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

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

##### Parameters

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

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

#### 15.14.3.2 quarantine\_agents()

```
template<typename TSeq >
void ModelMeaslesQuarantine< TSeq >::quarantine_agents [inline]
```

Quarantine agents that are in the system.

The flow should be:

- The function only runs if the quarantine status is active.
- Agents who are in quarantine, isolation, removed, or hospitalized are ignored.
- Agents who are in the RASH state are isolated.
- Vaccinated agents are ignored.
- Susceptible, Exposed, and Prodromal agents are moved to the QUARANTINED\_\* state.
- At the end of the function, the quarantine status is set false.

### 15.14.3.3 reset()

```
template<typename TSeq >
void ModelMeaslesQuarantine< TSeq >::reset [inline], [virtual]
```

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

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

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

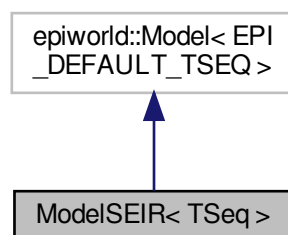
- `include/epiworld/models/measlesquarantine.hpp`

## 15.15 ModelSEIR< TSeq > Class Template Reference

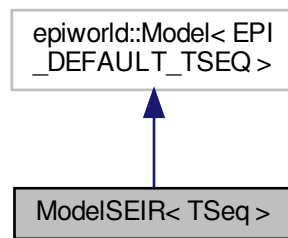
Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.

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

Inheritance diagram for ModelSEIR< TSeq >:



Collaboration diagram for ModelSEIR< TSeq >:



## Public Member Functions

- **ModelSEIR** ([ModelSEIR](#)< TSeq > &model, const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate)
- **ModelSEIR** (const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate)
- [ModelSEIR](#)< TSeq > & [initial\\_states](#) (std::vector< double > proportions\_, std::vector< int > queue\_={})  
*Set up the initial states of the model.*

## Public Attributes

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

## Static Public Attributes

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

### 15.15.1 Detailed Description

```
template<typename TSeq = EPI_DEFAULT_TSEQ>
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>prevalence</i>	epiworld_double Initial prevalence the immune system
<i>transmission_rate</i>	epiworld_double Transmission rate of the virus
<i>avg_incubation_days</i>	epiworld_double Average incubation days of the virus.
<i>recovery_rate</i>	epiworld_double Recovery rate of the virus.

## 15.15.2 Member Function Documentation

### 15.15.2.1 initial\_states()

```
template<typename TSeq >
ModelSEIR< TSeq > & ModelSEIR< TSeq >::initial_states (
 std::vector< double > proportions_,
 std::vector< int > queue_ = {}) [inline]
```

Set up the initial states of the model.

#### Parameters

<i>proportions_</i>	Double vector with the following values:
—	<ul style="list-style-type: none"> <li>• 0: Proportion of non-infected agents who are removed.</li> <li>• 1: Proportion of exposed agents to be set as infected.</li> </ul>

## 15.15.3 Member Data Documentation

### 15.15.3.1 update\_exposed\_seir

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

#### Initial value:

```
=
 [](epiworld::Agent<TSeq>* p, epiworld::Model<TSeq>* m) -> void {
 auto v = p->get_virus();
 if (m->runif() < 1.0 / (v->get_incubation(m)))
 p->change_state(m, ModelSEIR<TSeq>::INFECTED);
 return;
 }
```

### 15.15.3.2 update\_infected\_seir

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

#### Initial value:

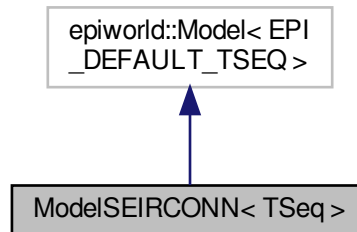
```
=
 [](epiworld::Agent<TSeq>* p, epiworld::Model<TSeq>* m) -> void {
 if (m->runif() < (m->par("Recovery rate")))
 p->rm_virus(m);
 return;
 }
```

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

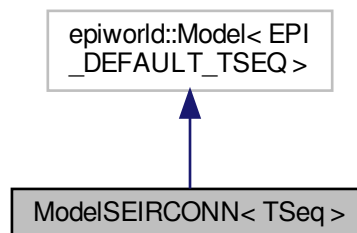
- include/epiworld/models/seir.hpp

## 15.16 ModelSEIRCONN< TSeq > Class Template Reference

Inheritance diagram for ModelSEIRCONN< TSeq >:



Collaboration diagram for ModelSEIRCONN< TSeq >:



### Public Member Functions

- [ModelSEIRCONN](#) ([ModelSEIRCONN](#)< TSeq > &model, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate)  
*Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.*
- **ModelSEIRCONN** (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate)
- [ModelSEIRCONN](#)< TSeq > & **run** (epiworld\_fast\_uint ndays, int seed=-1)
- void **reset** ()
- [Model](#)< TSeq > \* **clone\_ptr** ()
- [ModelSEIRCONN](#)< TSeq > & **initial\_states** (std::vector< double > proportions\_, std::vector< int > queue\_ = {})  
*Set the initial states of the model.*
- size\_t **get\_n\_infected** () const
- std::vector< double > **generation\_time\_expected** (int max\_days=200, int max\_contacts=200) const

## Static Public Attributes

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

## 15.16.1 Constructor & Destructor Documentation

### 15.16.1.1 ModelSEIRCONN()

```
template<typename TSeq >
ModelSEIRCONN< TSeq >::ModelSEIRCONN (
 ModelSEIRCONN< TSeq > & model,
 const std::string & vname,
 epiworld_uint n,
 epiworld_double prevalence,
 epiworld_double contact_rate,
 epiworld_double transmission_rate,
 epiworld_double avg_incubation_days,
 epiworld_double recovery_rate) [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>transmission_rate</i>	Probability of transmission
<i>recovery_rate</i>	Probability of recovery

## 15.16.2 Member Function Documentation

### 15.16.2.1 initial\_states()

```
template<typename TSeq >
ModelSEIRCONN< TSeq > & ModelSEIRCONN< TSeq >::initial_states (
 std::vector< double > proportions_,
 std::vector< int > queue_ = {}) [inline]
```

Set the initial states of the model.

## Parameters

<i>proportions</i> ↔	Double vector with a single element:
—	<ul style="list-style-type: none"> <li>The proportion of non-infected individuals who have recovered.</li> </ul>

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

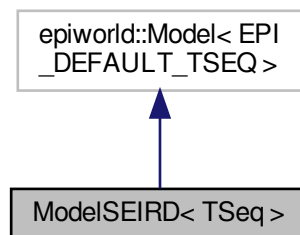
- include/epiworld/models/seirconnected.hpp

## 15.17 ModelSEIRD< TSeq > Class Template Reference

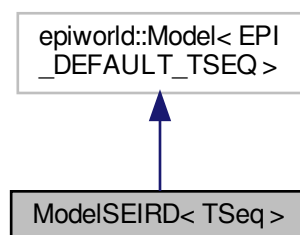
Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model.

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

Inheritance diagram for ModelSEIRD< TSeq >:



Collaboration diagram for ModelSEIRD< TSeq >:





## Public Member Functions

- [ModelSEIRD](#) ([ModelSEIRD](#)< TSeq > &model, const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, epiworld\_double death\_rate)  
*Constructor for the SEIRD model.*
- [ModelSEIRD](#) (const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, epiworld\_double death\_rate)  
*Constructor for the SEIRD model.*
- [ModelSEIRD](#)< TSeq > & **initial\_states** (std::vector< double > proportions\_, std::vector< int > queue\_={})

## Public Attributes

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

## Static Public Attributes

- static const int **SUSCEPTIBLE** = 0
- static const int **EXPOSED** = 1
- static const int **INFECTED** = 2
- static const int **REMOVED** = 3
- static const int **DECEASED** = 4

### 15.17.1 Detailed Description

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

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

### 15.17.2 Constructor & Destructor Documentation

#### 15.17.2.1 ModelSEIRD() [1/2]

```
template<typename TSeq >
ModelSEIRD< TSeq >::ModelSEIRD (
 ModelSEIRD< TSeq > & model,
 const std::string & vname,
 epiworld_double prevalence,
 epiworld_double transmission_rate,
 epiworld_double avg_incubation_days,
 epiworld_double recovery_rate,
 epiworld_double death_rate) [inline]
```

Constructor for the SEIRD model.

## Template Parameters

<i>TSeq</i>	Type of the sequence used in the model.
-------------	-----------------------------------------

## Parameters

<i>model</i>	Reference to the SEIRD model.
<i>vname</i>	Name of the model.
<i>prevalence</i>	Prevalence of the disease.
<i>transmission_rate</i>	Transmission rate of the disease.
<i>avg_incubation_days</i>	Average incubation period of the disease.
<i>recovery_rate</i>	Recovery rate of the disease.
<i>death_rate</i>	Death rate of the disease.

## 15.17.2.2 ModelSEIRD() [2/2]

```
template<typename TSeq >
ModelSEIRD< TSeq >::ModelSEIRD (
 const std::string & vname,
 epiworld_double prevalence,
 epiworld_double transmission_rate,
 epiworld_double avg_incubation_days,
 epiworld_double recovery_rate,
 epiworld_double death_rate) [inline]
```

Constructor for the SEIRD model.

## Parameters

<i>vname</i>	Name of the model.
<i>prevalence</i>	Initial prevalence of the disease.
<i>transmission_rate</i>	Transmission rate of the disease.
<i>avg_incubation_days</i>	Average incubation period of the disease.
<i>recovery_rate</i>	Recovery rate of the disease.
<i>death_rate</i>	Death rate of the disease.

## 15.17.3 Member Data Documentation

## 15.17.3.1 update\_exposed\_seir

```
template<typename TSeq = EPI_DEFAULT_TSEQ>
epiworld::UpdateFun<TSeq> ModelSEIRD< TSeq >::update_exposed_seir
```

**Initial value:**

```

=
 [](epiworld::Agent<TSeq>* p, epiworld::Model<TSeq>* m) -> void {
 auto v = p->get_virus();
 if (m->runif() < 1.0 / (v->get_incubation(m)))
 p->change_state(m, ModelSEIRD<TSeq>::INFECTED);
 return;
 }

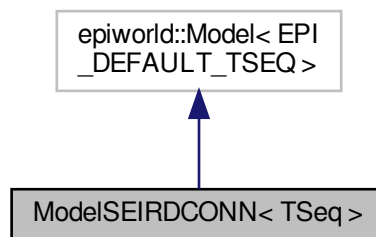
```

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

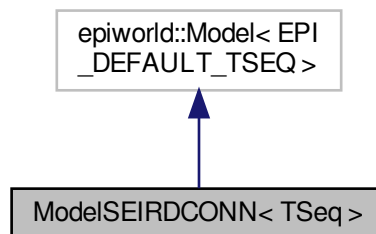
- include/epiworld/models/seird.hpp

## 15.18 ModelSEIRDCONN< TSeq > Class Template Reference

Inheritance diagram for ModelSEIRDCONN< TSeq >:



Collaboration diagram for ModelSEIRDCONN< TSeq >:



## Public Member Functions

- [ModelSEIRDCONN](#) ([ModelSEIRDCONN](#)< TSeq > &model, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, epiworld\_double death\_rate)  
*Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.*
- **ModelSEIRDCONN** (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, epiworld\_double death\_rate)
- [ModelSEIRDCONN](#)< TSeq > & **run** (epiworld\_fast\_uint ndays, int seed=-1)
- void **reset** ()
- [Model](#)< TSeq > \* **clone\_ptr** ()
- [ModelSEIRDCONN](#)< TSeq > & **initial\_states** (std::vector< double > proportions\_, std::vector< int > queue\_={})  
*Set up the initial states of the model.*
- size\_t **get\_n\_infected** () const

## Static Public Attributes

- static const int **SUSCEPTIBLE** = 0
- static const int **EXPOSED** = 1
- static const int **INFECTED** = 2
- static const int **REMOVED** = 3
- static const int **DECEASED** = 4

## 15.18.1 Constructor & Destructor Documentation

### 15.18.1.1 ModelSEIRDCONN()

```
template<typename TSeq >
ModelSEIRDCONN< TSeq >::ModelSEIRDCONN (
 ModelSEIRDCONN< TSeq > & model,
 const std::string & vname,
 epiworld_fast_uint n,
 epiworld_double prevalence,
 epiworld_double contact_rate,
 epiworld_double transmission_rate,
 epiworld_double avg_incubation_days,
 epiworld_double recovery_rate,
 epiworld_double death_rate) [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>transmission_rate</i>	Probability of transmission
<i>recovery_rate</i>	Probability of recovery
<i>death_rate</i>	Probability of death

## 15.18.2 Member Function Documentation

### 15.18.2.1 initial\_states()

```
template<typename TSeq >
ModelSEIRDCONN< TSeq > & ModelSEIRDCONN< TSeq >::initial_states (
 std::vector< double > proportions_,
 std::vector< int > queue_ = {}) [inline]
```

Set up the initial states of the model.

#### Parameters

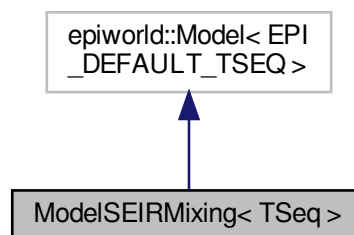
<i>proportions_</i> ↔	Double vector with the following values:
—	<ul style="list-style-type: none"> <li>• 0: Proportion of non-infected agents who are removed.</li> <li>• 1: Proportion of exposed agents to be set as infected.</li> </ul>

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

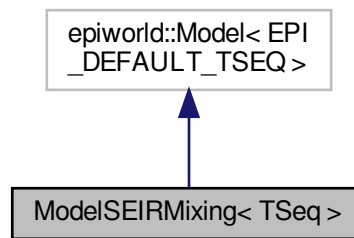
- include/epiworld/models/seirdconnected.hpp

## 15.19 ModelSEIRMixing< TSeq > Class Template Reference

Inheritance diagram for ModelSEIRMixing< TSeq >:



Collaboration diagram for ModelSEIRMixing< TSeq >:



## Public Member Functions

- [ModelSEIRMixing](#) ([ModelSEIRMixing](#)< TSeq > &model, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, std::vector< double > contact\_matrix)  
Constructs a [ModelSEIRMixing](#) object.
- [ModelSEIRMixing](#) (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, std::vector< double > contact\_matrix)  
Constructs a [ModelSEIRMixing](#) object.
- [ModelSEIRMixing](#)< TSeq > & **run** (epiworld\_fast\_uint ndays, int seed=-1)
- void **reset** ()
- [Model](#)< TSeq > \* **clone\_ptr** ()
- [ModelSEIRMixing](#)< TSeq > & **initial\_states** (std::vector< double > proportions\_, std::vector< int > queue\_ = {})  
Set the initial states of the model.
- void **set\_contact\_matrix** (std::vector< double > cmat)

## Static Public Attributes

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

### 15.19.1 Constructor & Destructor Documentation

## 15.19.1.1 ModelSEIRMixing() [1/2]

```
template<typename TSeq >
ModelSEIRMixing< TSeq >::ModelSEIRMixing (
 ModelSEIRMixing< TSeq > & model,
 const std::string & vname,
 epiworld_fast_uint n,
 epiworld_double prevalence,
 epiworld_double contact_rate,
 epiworld_double transmission_rate,
 epiworld_double avg_incubation_days,
 epiworld_double recovery_rate,
 std::vector< double > contact_matrix) [inline]
```

Constructs a [ModelSEIRMixing](#) object.

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

## Parameters

<i>model</i>	A reference to an existing <a href="#">ModelSEIRMixing</a> object.
<i>vname</i>	The name of the <a href="#">ModelSEIRMixing</a> object.
<i>n</i>	The number of entities in the model.
<i>prevalence</i>	The initial prevalence of the disease in the model.
<i>contact_rate</i>	The contact rate between entities in the model.
<i>transmission_rate</i>	The transmission rate of the disease in the model.
<i>avg_incubation_days</i>	The average incubation period of the disease in the model.
<i>recovery_rate</i>	The recovery rate of the disease in the model.
<i>contact_matrix</i>	The contact matrix between entities in the model. Specified in column-major order.
<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>transmission_rate</i>	Probability of transmission
<i>recovery_rate</i>	Probability of recovery

## 15.19.1.2 ModelSEIRMixing() [2/2]

```
template<typename TSeq >
ModelSEIRMixing< TSeq >::ModelSEIRMixing (
 const std::string & vname,
 epiworld_fast_uint n,
 epiworld_double prevalence,
 epiworld_double contact_rate,
 epiworld_double transmission_rate,
 epiworld_double avg_incubation_days,
 epiworld_double recovery_rate,
 std::vector< double > contact_matrix) [inline]
```

Constructs a [ModelSEIRMixing](#) object.

## Parameters

<i>vname</i>	The name of the <a href="#">ModelSEIRMixing</a> object.
<i>n</i>	The number of entities in the model.
<i>prevalence</i>	The initial prevalence of the disease in the model.
<i>contact_rate</i>	The contact rate between entities in the model.
<i>transmission_rate</i>	The transmission rate of the disease in the model.
<i>avg_incubation_days</i>	The average incubation period of the disease in the model.
<i>recovery_rate</i>	The recovery rate of the disease in the model.
<i>contact_matrix</i>	The contact matrix between entities in the model.

## 15.19.2 Member Function Documentation

### 15.19.2.1 initial\_states()

```
template<typename TSeq >
ModelSEIRMixing< TSeq > & ModelSEIRMixing< TSeq >::initial_states (
 std::vector< double > proportions_,
 std::vector< int > queue_ = {}) [inline]
```

Set the initial states of the model.

## Parameters

<i>proportions_↵</i>	Double vector with a single element:
—	<ul style="list-style-type: none"> <li>The proportion of non-infected individuals who have recovered.</li> </ul>

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

- include/epiworld/models/seirmixing.hpp

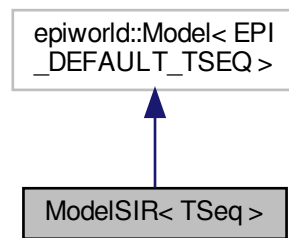
## 15.20 ModelSIR< TSeq > Class Template Reference

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

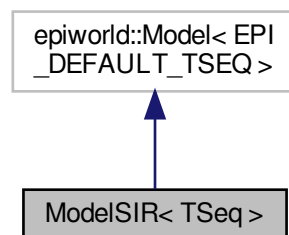
```
#include <sir.hpp>
```



Inheritance diagram for ModelSIR< TSeq >:



Collaboration diagram for ModelSIR< TSeq >:



## Public Member Functions

- **ModelSIR** ([ModelSIR](#)< TSeq > &model, const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate)
- **ModelSIR** (const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate)
- [ModelSIR](#)< TSeq > & [initial\\_states](#) (std::vector< double > proportions\_, std::vector< int > queue\_={})

*Set the initial states of the model.*

### 15.20.1 Detailed Description

```

template<typename TSeq = EPI_DEFAULT_TSEQ>
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_efficiency</i>	epiworld_double Initial susceptibility_reduction of the immune system
<i>initial_recovery</i>	epiworld_double Initial recovery_rate rate of the immune system

## 15.20.2 Member Function Documentation

### 15.20.2.1 initial\_states()

```
template<typename TSeq >
ModelSIR< TSeq > & ModelSIR< TSeq >::initial_states (
 std::vector< double > proportions_,
 std::vector< int > queue_ = {}) [inline]
```

Set the initial states of the model.

## Parameters

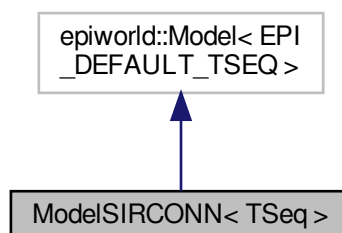
<i>proportions_</i>	Double vector with a single element:
—	<ul style="list-style-type: none"> <li>The proportion of non-infected individuals who have recovered.</li> </ul>

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

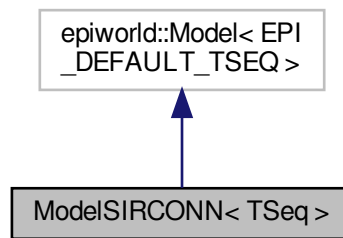
- include/epiworld/models/sir.hpp

## 15.21 ModelSIRCONN< TSeq > Class Template Reference

Inheritance diagram for ModelSIRCONN< TSeq >:



Collaboration diagram for ModelSIRCONN< TSeq >:



## Public Member Functions

- [ModelSIRCONN](#) ([ModelSIRCONN](#)< TSeq > &model, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate)  
*Template for a Susceptible-Infected-Removed (SIR) model.*
- **ModelSIRCONN** (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate)
- [ModelSIRCONN](#)< TSeq > & **run** (epiworld\_fast\_uint ndays, int seed=-1)
- void **reset** ()
- [Model](#)< TSeq > \* **clone\_ptr** ()
- [ModelSIRCONN](#)< TSeq > & **initial\_states** (std::vector< double > proportions\_, std::vector< int > queue\_ = {})  
*Set the initial states of the model.*
- size\_t **get\_n\_infected** () const  
*Get the infected individuals.*
- std::vector< double > **generation\_time\_expected** (int max\_days=200, int max\_contacts=200) const

## Static Public Attributes

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

### 15.21.1 Constructor & Destructor Documentation

### 15.21.1.1 ModelSIRCONN()

```
template<typename TSeq >
ModelSIRCONN< TSeq >::ModelSIRCONN (
 ModelSIRCONN< TSeq > & model,
 const std::string & vname,
 epiworld_uint n,
 epiworld_double prevalence,
 epiworld_double contact_rate,
 epiworld_double transmission_rate,
 epiworld_double recovery_rate) [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>transmission_rate</i>	Probability of transmission
<i>recovery_rate</i>	Probability of recovery

## 15.21.2 Member Function Documentation

### 15.21.2.1 get\_n\_infected()

```
template<typename TSeq = EPI_DEFAULT_TSEQ>
size_t ModelSIRCONN< TSeq >::get_n_infected () const [inline]
```

Get the infected individuals.

#### Returns

std::vector< epiworld::Agent<TSeq> \* >

### 15.21.2.2 initial\_states()

```
template<typename TSeq >
ModelSIRCONN< TSeq > & ModelSIRCONN< TSeq >::initial_states (
 std::vector< double > proportions_,
 std::vector< int > queue_ = {}) [inline]
```

Set the initial states of the model.

## Parameters

<i>proportions</i> ↔	Double vector with a single element:
—	<ul style="list-style-type: none"> <li>The proportion of non-infected individuals who have recovered.</li> </ul>

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

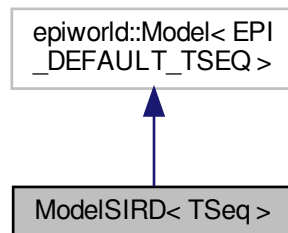
- include/epiworld/models/sirconnected.hpp

## 15.22 ModelSIRD< TSeq > Class Template Reference

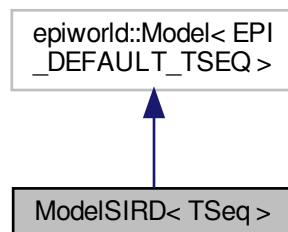
Template for a Susceptible-Infected-Removed-Deceased (SIRD) model.

```
#include <sird.hpp>
```

Inheritance diagram for ModelSIRD< TSeq >:



Collaboration diagram for ModelSIRD< TSeq >:



## Public Member Functions

- [ModelSIRD](#)< TSeq > & [initial\\_states](#) (std::vector< double > proportions\_, std::vector< int > queue\_={})  
*Set the initial states of the model.*
- [ModelSIRD](#) ([ModelSIRD](#)< TSeq > &model, const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double death\_rate)  
*Constructs a new SIRD model with the given parameters.*
- **ModelSIRD** (const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double death\_rate)

### 15.22.1 Detailed Description

```
template<typename TSeq = EPI_DEFAULT_TSEQ>
class ModelSIRD< TSeq >
```

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

### 15.22.2 Constructor & Destructor Documentation

#### 15.22.2.1 ModelSIRD()

```
template<typename TSeq >
ModelSIRD< TSeq >::ModelSIRD (
 ModelSIRD< TSeq > & model,
 const std::string & vname,
 epiworld_double prevalence,
 epiworld_double transmission_rate,
 epiworld_double recovery_rate,
 epiworld_double death_rate) [inline]
```

Constructs a new SIRD model with the given parameters.

#### Parameters

<i>model</i>	The SIRD model to copy from.
<i>vname</i>	The name of the vertex associated with this model.
<i>prevalence</i>	The initial prevalence of the disease in the population.
<i>transmission_rate</i>	The rate at which the disease spreads from infected to susceptible individuals.
<i>recovery_rate</i>	The rate at which infected individuals recover and become immune.
<i>death_rate</i>	The rate at which infected individuals die.

### 15.22.3 Member Function Documentation

#### 15.22.3.1 initial\_states()

```
template<typename TSeq >
ModelSIRD< TSeq > & ModelSIRD< TSeq >::initial_states (
 std::vector< double > proportions_,
 std::vector< int > queue_ = {}) [inline]
```

Set the initial states of the model.

##### Parameters

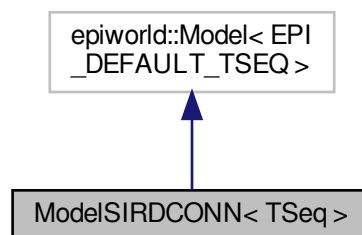
<i>proportions_</i>	Double vector with two elements:
—	<ul style="list-style-type: none"> <li>• The proportion of non-infected individuals who have recovered.</li> <li>• The proportion of non-infected individuals who have died.</li> </ul>

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

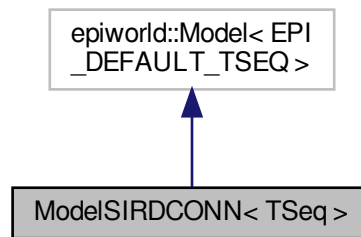
- include/epiworld/models/sird.hpp

## 15.23 ModelSIRDConn< TSeq > Class Template Reference

Inheritance diagram for ModelSIRDConn< TSeq >:



Collaboration diagram for ModelSIRDCONN< TSeq >:



## Public Member Functions

- [ModelSIRDCONN](#) ([ModelSIRDCONN](#) < TSeq > &model, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double death\_rate)  
*Template for a Susceptible-Infected-Removed (SIR) model.*
- **ModelSIRDCONN** (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double death\_rate)
- [ModelSIRDCONN](#) < TSeq > &run (epiworld\_fast\_uint ndays, int seed=-1)
- void **reset** ()
- [Model](#) < TSeq > \* **clone\_ptr** ()

## Static Public Attributes

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

## 15.23.1 Constructor & Destructor Documentation

### 15.23.1.1 ModelSIRDCONN()

```

template<typename TSeq >
ModelSIRDCONN< TSeq >::ModelSIRDCONN (
 ModelSIRDCONN< TSeq > & model,
 const std::string & vname,
 epiworld_fast_uint n,
 epiworld_double prevalence,
 epiworld_double contact_rate,
 epiworld_double transmission_rate,
 epiworld_double recovery_rate,
 epiworld_double death_rate) [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>transmission_rate</i>	Probability of transmission
<i>recovery_rate</i>	Probability of recovery
<i>death_rate</i>	Probability of death

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

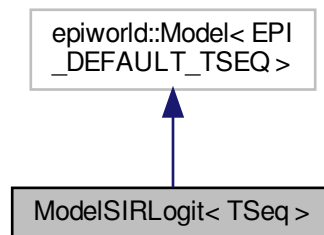
- include/epiworld/models/sirdconnected.hpp

## 15.24 ModelSIRLogit< TSeq > Class Template Reference

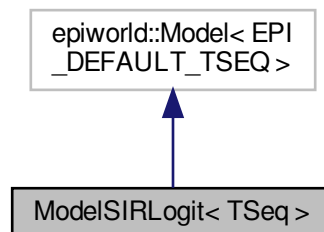
Template for a Susceptible-Infected-Removed (SIR) model.

```
#include <sirlogit.hpp>
```

Inheritance diagram for ModelSIRLogit< TSeq >:



Collaboration diagram for ModelSIRLogit< TSeq >:



## Public Member Functions

- [ModelSIRLogit](#) ([ModelSIRLogit](#)< TSeq > &model, const std::string &vname, double \*data, size\_t ncols, std::vector< double > coefs\_infect, std::vector< double > coefs\_recover, std::vector< size\_t > coef\_infect\_cols, std::vector< size\_t > coef\_recover\_cols, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double prevalence)

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

- **ModelSIRLogit** (const std::string &vname, double \*data, size\_t ncols, std::vector< double > coefs\_infect, std::vector< double > coefs\_recover, std::vector< size\_t > coef\_infect\_cols, std::vector< size\_t > coef\_recover\_cols, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double prevalence)
- [ModelSIRLogit](#)< TSeq > & **run** (epiworld\_fast\_uint ndays, int seed=-1)
- [Model](#)< TSeq > \* **clone\_ptr** ()
- void **reset** ()

## Public Attributes

- std::vector< double > **coefs\_infect**
- std::vector< double > **coefs\_recover**
- std::vector< size\_t > **coef\_infect\_cols**
- std::vector< size\_t > **coef\_recover\_cols**

### 15.24.1 Detailed Description

```
template<typename TSeq = EPI_DEFAULT_TSEQ>
class ModelSIRLogit< TSeq >
```

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

In this model, infection and recovery probabilities are computed using a logit model. Particularly, the probability of infection is computed as:

$$\frac{1}{1 + \exp(-(\beta_0 E_i + \sum_{i=1}^n \beta_i x_i))}$$

where  $\beta_0$  is the exposure coefficient and  $E_i$  is the exposure number,  $\beta_i$  are the coefficients for the features  $x_i$  of the agents, and  $n$  is the number of features. The probability of recovery is computed as:

$$\frac{1}{1 + \exp(-(\sum_{i=1}^n \beta_i x_i))}$$

where  $\beta_i$  are the coefficients for the features  $x_i$  of the agents, and  $n$  is the number of features.

#### Parameters

<i>TSeq</i>	Type of the sequence (e.g. std::vector, std::deque)
-------------	-----------------------------------------------------

## 15.24.2 Constructor & Destructor Documentation

### 15.24.2.1 ModelSIRLogit()

```
template<typename TSeq >
ModelSIRLogit< TSeq >::ModelSIRLogit (
 ModelSIRLogit< TSeq > & model,
 const std::string & vname,
 double * data,
 size_t ncols,
 std::vector< double > coefs_infect,
 std::vector< double > coefs_recover,
 std::vector< size_t > coef_infect_cols,
 std::vector< size_t > coef_recover_cols,
 epiworld_double transmission_rate,
 epiworld_double recovery_rate,
 epiworld_double prevalence) [inline]
```

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

#### Parameters

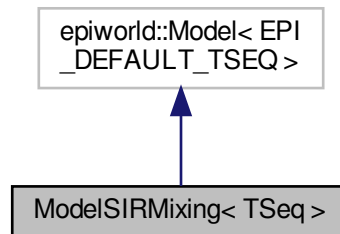
<i>vname</i>	Name of the virus.
<i>coefs_infect</i>	Double ptr. Infection coefficients.
<i>coefs_recover</i>	Double ptr. Recovery coefficients.
<i>ncoef_infect</i>	Unsigned int. Number of infection coefficients.
<i>ncoef_recover</i>	Unsigned int. Number of recovery coefficients.
<i>coef_infect_cols</i>	Vector<unsigned int>. Ids of infection vars.
<i>coef_recover_cols</i>	Vector<unsigned int>. Ids of recover vars.
<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

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

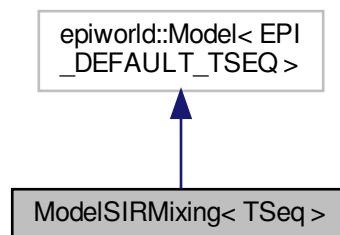
- include/epiworld/models/sirlogit.hpp

## 15.25 ModelSIRMixing< TSeq > Class Template Reference

Inheritance diagram for ModelSIRMixing< TSeq >:



Collaboration diagram for ModelSIRMixing< TSeq >:



### Public Member Functions

- [ModelSIRMixing](#) ([ModelSIRMixing](#)< TSeq > &model, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, std::vector< double > contact\_matrix)  
Constructs a [ModelSIRMixing](#) object.
- [ModelSIRMixing](#) (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, std::vector< double > contact\_matrix)  
Constructs a [ModelSIRMixing](#) object.
- [ModelSIRMixing](#)< TSeq > &run (epiworld\_fast\_uint ndays, int seed=-1)
- void **reset** ()
- [Model](#)< TSeq > \* **clone\_ptr** ()
- [ModelSIRMixing](#)< TSeq > & **initial\_states** (std::vector< double > proportions\_, std::vector< int > queue\_=  
\_={})  
Set the initial states of the model.
- size\_t **get\_n\_infected** (size\_t group) const
- void **set\_contact\_matrix** (std::vector< double > cmat)

## Static Public Attributes

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

## 15.25.1 Constructor & Destructor Documentation

### 15.25.1.1 ModelSIRMixing() [1/2]

```
template<typename TSeq >
ModelSIRMixing< TSeq >::ModelSIRMixing (
 ModelSIRMixing< TSeq > & model,
 const std::string & vname,
 epiworld_fast_uint n,
 epiworld_double prevalence,
 epiworld_double contact_rate,
 epiworld_double transmission_rate,
 epiworld_double recovery_rate,
 std::vector< double > contact_matrix) [inline]
```

Constructs a [ModelSIRMixing](#) object.

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

#### Parameters

<i>model</i>	A reference to an existing <a href="#">ModelSIRMixing</a> object.
<i>vname</i>	The name of the <a href="#">ModelSIRMixing</a> object.
<i>n</i>	The number of entities in the model.
<i>prevalence</i>	The initial prevalence of the disease in the model.
<i>contact_rate</i>	The contact rate between entities in the model.
<i>transmission_rate</i>	The transmission rate of the disease in the model.
<i>recovery_rate</i>	The recovery rate of the disease in the model.
<i>contact_matrix</i>	The contact matrix between entities in the model.
<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>transmission_rate</i>	Probability of transmission
<i>recovery_rate</i>	Probability of recovery

### 15.25.1.2 ModelSIRMixing() [2/2]

```
template<typename TSeq >
ModelSIRMixing< TSeq >::ModelSIRMixing (
```

```

const std::string & vname,
epiworld_fast_uint n,
epiworld_double prevalence,
epiworld_double contact_rate,
epiworld_double transmission_rate,
epiworld_double recovery_rate,
std::vector< double > contact_matrix) [inline]

```

Constructs a [ModelSIRMixing](#) object.

#### Parameters

<i>vname</i>	The name of the <a href="#">ModelSIRMixing</a> object.
<i>n</i>	The number of entities in the model.
<i>prevalence</i>	The initial prevalence of the disease in the model.
<i>contact_rate</i>	The contact rate between entities in the model.
<i>transmission_rate</i>	The transmission rate of the disease in the model.
<i>recovery_rate</i>	The recovery rate of the disease in the model.
<i>contact_matrix</i>	The contact matrix between entities in the model.

## 15.25.2 Member Function Documentation

### 15.25.2.1 initial\_states()

```

template<typename TSeq >
ModelSIRMixing< TSeq > & ModelSIRMixing< TSeq >::initial_states (
 std::vector< double > proportions_,
 std::vector< int > queue_ = {}) [inline]

```

Set the initial states of the model.

#### Parameters

<i>proportions_</i> ↔	Double vector with a single element:
—	<ul style="list-style-type: none"> <li>The proportion of non-infected individuals who have recovered.</li> </ul>

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

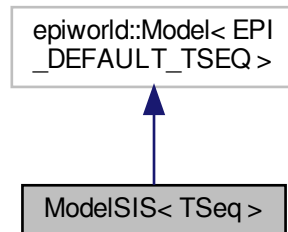
- include/epiworld/models/sirmixing.hpp

## 15.26 ModelSIS< TSeq > Class Template Reference

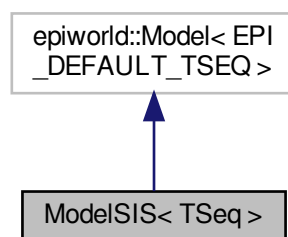
Template for a Susceptible-Infected-Susceptible (SIS) model.

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

Inheritance diagram for ModelSIS< TSeq >:



Collaboration diagram for ModelSIS< TSeq >:



## Public Member Functions

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

## Static Public Attributes

- static const int **SUSCEPTIBLE** = 0
- static const int **INFECTED** = 1

### 15.26.1 Detailed Description

```
template<typename TSeq = EPI_DEFAULT_TSEQ>
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_efficacy</i>	epiworld_double Initial susceptibility_reduction of the immune system
<i>initial_recovery</i>	epiworld_double Initial recovery_rate rate of the immune system

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

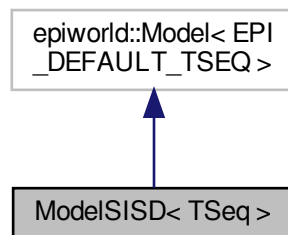
- include/epiworld/models/sis.hpp

## 15.27 ModelSISD< TSeq > Class Template Reference

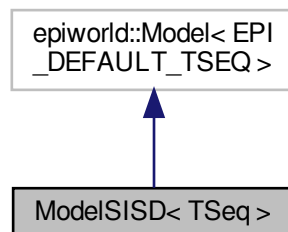
Template for a Susceptible-Infected-Susceptible-Deceased (SISD) model.

```
#include <sisd.hpp>
```

Inheritance diagram for ModelSISD< TSeq >:



Collaboration diagram for ModelSISD< TSeq >:





## Public Member Functions

- **ModelSISD** ([ModelSISD](#)< TSeq > &model, const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double death\_rate)
- **ModelSISD** (const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double death\_rate)

### 15.27.1 Detailed Description

```
template<typename TSeq = EPI_DEFAULT_TSEQ>
class ModelSISD< TSeq >
```

Template for a Susceptible-Infected-Susceptible-Deceased (SISD) model.

#### Parameters

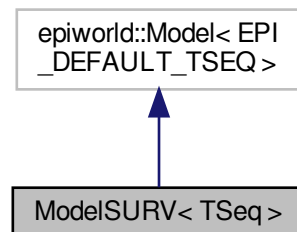
<i>vname</i>	std::string Name of the virus
<i>initial_prevalence</i>	epiworld_double Initial prevalence
<i>initial_efficacy</i>	epiworld_double Initial susceptibility_reduction of the immune system
<i>initial_recovery</i>	epiworld_double Initial recovery_rate rate of the immune system
<i>inital_death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

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

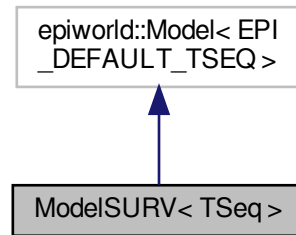
- include/epiworld/models/sisd.hpp

## 15.28 ModelSURV< TSeq > Class Template Reference

Inheritance diagram for ModelSURV< TSeq >:



Collaboration diagram for ModelSURV< TSeq >:



## Public Member Functions

- void `reset ()`

### Construct a new ModelSURV object

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

#### Parameters

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

#### Public Attributes

- std::vector< epiworld\_double > [days\\_latent\\_and\\_infectious](#)  
Vector of days spent in latent and infectious states A row-major matrix.

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

- include/epiworld/models/surveillance.hpp

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

#### Public Member Functions

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

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

- include/epiworld/network-bones.hpp

## 15.30 PersonTools< TSeq > Class Template Reference

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

- include/epiworld/config.hpp

## 15.31 Progress Class Reference

A simple progress bar.

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

### Public Member Functions

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

### 15.31.1 Detailed Description

A simple progress bar.

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

- include/epiworld/progress.hpp

## 15.32 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

## Static Public Attributes

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

## Friends

- class **Model**< TSeq >

### 15.32.1 Detailed Description

```
template<typename TSeq>
class Queue< TSeq >
```

Controls which agents are verified at each step.

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

#### Template Parameters

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

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

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

## 15.33 RandGraph Class Reference

### Public Member Functions

- **RandGraph** (int N\_)
- void **init** (int s)
- void **set\_rand\_engine** (std::shared\_ptr< std::mt19937 > &e)
- epiworld\_double **runif** ()

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

- include/epiworld/random\_graph.hpp

## 15.34 SAMPLETYPE Class Reference

### Static Public Attributes

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

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

- include/epiworld/agentssample-bones.hpp

## 15.35 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")
- **Tool** (std::string name, epiworld\_double prevalence, bool as\_proportion)
- void **set\_sequence** (TSeq d)
- void **set\_sequence** (std::shared\_ptr< TSeq > d)
- **EPI\_TYPENAME\_TRAITS** (TSeq, int) 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
- void **print** () const
- void **distribute** ([Model](#)< TSeq > \*model)
- void **set\_distribution** (ToolToAgentFun< TSeq > fun)
- void **set\_sequence** (int d)
- 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** ([Event](#)< TSeq > &a, [Model](#)< TSeq > \*m)
- void **default\_rm\_tool** ([Event](#)< TSeq > &a, [Model](#)< TSeq > \*m)

**15.35.1 Detailed Description**

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

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

**Template Parameters**

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

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

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

**15.36 ToolFunctions< TSeq > Class Template Reference**

Helper class to store the functions avoiding multiple shared\_pointers (we have only one for the four of these)

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

## Public Attributes

- ToolFun< TSeq > **susceptibility\_reduction** = nullptr
- ToolFun< TSeq > **transmission\_reduction** = nullptr
- ToolFun< TSeq > **recovery\_enhancer** = nullptr
- ToolFun< TSeq > **death\_reduction** = nullptr
- ToolToAgentFun< TSeq > **dist** = nullptr

### 15.36.1 Detailed Description

```
template<typename TSeq>
class ToolFunctions< TSeq >
```

Helper class to store the functions avoiding multiple shared\_pointers (we have only one for the four of these)

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

- include/epiworld/tool-bones.hpp

## 15.37 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
- void **print** () const noexcept

## Friends

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

### 15.37.1 Detailed Description

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

Set of tools (useful for building iterators)



## Template Parameters

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

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

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

## 15.38 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
- void **print** () const noexcept

### Friends

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

#### 15.38.1 Detailed Description

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

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

## Template Parameters

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

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

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

## 15.39 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

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()</code> - 1.

- 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</code> - 1.)
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 >

### 15.39.1 Detailed Description

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

Personalized data by the user.

Template Parameters

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

### 15.39.2 Constructor & Destructor Documentation

#### 15.39.2.1 UserData()

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

Construct a new User Data object.

Parameters

<i>names</i>	A vector of names. The length of the vector sets the number of columns to record.
--------------	-----------------------------------------------------------------------------------

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

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

## 15.40 vecHasher< T > Struct Template Reference

Vector hasher.

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

### Public Member Functions

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

#### 15.40.1 Detailed Description

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

Vector hasher.

## Template Parameters

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

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

- include/epiworld/misc.hpp

## 15.41 Virus< TSeq > Class Template Reference

[Virus.](#)

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

### Public Member Functions

- **Virus** (std::string name="unknown virus")
- **Virus** (std::string name, epiworld\_double prevalence, bool as\_proportion)
- void **mutate** ([Model](#)< TSeq > \*model)
- void **set\_mutation** (MutFun< TSeq > fun)
- **EPI\_TYPENAME\_TRAITS** (TSeq, int) get\_sequence()
- void **set\_sequence** (TSeq sequence)
- [Agent](#)< TSeq > \* **get\_agent** ()
- void **set\_agent** ([Agent](#)< TSeq > \*p)
- 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
- bool **operator==** (const [Virus](#)< TSeq > &other) const
- bool **operator!=** (const [Virus](#)< TSeq > &other) const
- void **print** () const
- void **set\_sequence** (int sequence)
- 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)

- `epiworld_double` **get\_incubation** ([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\_incubation\_fun** (VirusFun< TSeq > fun)
- `void` **set\_prob\_infecting** (`const epiworld_double` \*prob)
- `void` **set\_prob\_recovery** (`const epiworld_double` \*prob)
- `void` **set\_prob\_death** (`const epiworld_double` \*prob)
- `void` **set\_incubation** (`const epiworld_double` \*prob)
- `void` **set\_prob\_infecting** (`epiworld_double` prob)
- `void` **set\_prob\_recovery** (`epiworld_double` prob)
- `void` **set\_prob\_death** (`epiworld_double` prob)
- `void` **set\_incubation** (`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)

- `void` **distribute** ([Model](#)< TSeq > \*model)  
Get information about the prevalence of the virus.
- `void` **set\_distribution** (VirusToAgentFun< TSeq > fun)

### Friends

- `class` **Agent**< TSeq >
- `class` **Model**< TSeq >
- `class` **DataBase**< TSeq >
- `void` **default\_add\_virus** ([Event](#)< TSeq > &a, [Model](#)< TSeq > \*m)
- `void` **default\_rm\_virus** ([Event](#)< TSeq > &a, [Model](#)< TSeq > \*m)

### 15.41.1 Detailed Description

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

[Virus](#).

## Template Parameters

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

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

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

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

## 15.42 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
- void **print** () const noexcept

### Friends

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

### 15.42.1 Detailed Description

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

Set of viruses (useful for building iterators)

## Template Parameters

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

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

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

## 15.43 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
- void **print** () const noexcept

### Friends

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

### 15.43.1 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

## 15.44 VirusFunctions< TSeq > Class Template Reference

### Public Attributes

- MutFun< TSeq > **mutation** = nullptr

- PostRecoveryFun< TSeq > **post\_recovery** = nullptr
- VirusFun< TSeq > **probability\_of\_infecting** = nullptr
- VirusFun< TSeq > **probability\_of\_recovery** = nullptr
- VirusFun< TSeq > **probability\_of\_death** = nullptr
- VirusFun< TSeq > **incubation** = nullptr
- VirusToAgentFun< TSeq > **dist** = nullptr

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

- include/epiworld/virus-bones.hpp



## Chapter 16

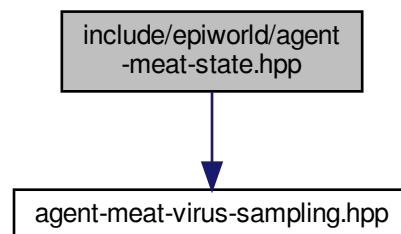
# File Documentation

### 16.1 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)`

### 16.1.1 Detailed Description

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

#### Author

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

#### Version

0.1

#### Date

2022-06-15

#### Copyright

Copyright (c) 2022

# Index

- add\_globlevent
  - Model< TSeq >, 62
- AdjList, 35
  - AdjList, 35
  - read\_edgelist, 36
- Agent< TSeq >, 36
  - default\_rm\_entity, 40
  - operator(), 39
  - swap\_neighbors, 39
- AgentsSample
  - AgentsSample< TSeq >, 41
- AgentsSample< TSeq >, 40
  - AgentsSample, 41
- clone\_ptr
  - Model< TSeq >, 63
  - ModelMeaslesQuarantine< TSeq >, 74
- DataBase< TSeq >, 42
  - get\_generation\_time, 44
  - get\_reproductive\_number, 44
  - get\_transition\_probability, 44
  - get\_transmissions, 45
  - operator==, 45, 46
  - record\_virus, 46
- default\_rm\_entity
  - Agent< TSeq >, 40
  - Entity< TSeq >, 49
- draw
  - Model< TSeq >, 63
- Entities< TSeq >, 46
- Entities\_const< TSeq >, 47
- Entity
  - Entity< TSeq >, 49
- Entity< TSeq >, 48
  - default\_rm\_entity, 49
  - Entity, 49
- Event
  - Event< TSeq >, 51
- Event< TSeq >, 50
  - Event, 51
- events\_add
  - Model< TSeq >, 63
- events\_run
  - Model< TSeq >, 64
- get\_generation\_time
  - DataBase< TSeq >, 44
- get\_n\_infected
  - ModelSIRCONN< TSeq >, 92
- get\_reproductive\_number
  - DataBase< TSeq >, 44
- get\_transition\_probability
  - DataBase< TSeq >, 44
- get\_transmissions
  - DataBase< TSeq >, 45
- GlobalEvent
  - GlobalEvent< TSeq >, 52
- GlobalEvent< TSeq >, 52
  - GlobalEvent, 52
- include/epiworld/agent-meat-state.hpp, 121
- initial\_states
  - ModelSEIR< TSeq >, 77
  - ModelSEIRCONN< TSeq >, 79
  - ModelSEIRDCONN< TSeq >, 85
  - ModelSEIRMixing< TSeq >, 88
  - ModelSIR< TSeq >, 90
  - ModelSIRCONN< TSeq >, 92
  - ModelSIRD< TSeq >, 95
  - ModelSIRMixing< TSeq >, 102
- initial\_states\_fun
  - Model< TSeq >, 67
- LFMCMC< TData >, 53
- load\_agents\_entities\_ties
  - Model< TSeq >, 64
- make\_sample\_virus\_neighbors
  - sampler, 31
- make\_update\_susceptible
  - sampler, 32
- Model< TSeq >, 54
  - add\_globlevent, 62
  - clone\_ptr, 63
  - draw, 63
  - events\_add, 63
  - events\_run, 64
  - initial\_states\_fun, 67
  - load\_agents\_entities\_ties, 64
  - recovery\_enhancer\_mixer, 67
  - reset, 65
  - rnbinomd, 67
  - run\_multiple, 65
  - runifd, 67
  - set\_agents\_data, 65
  - set\_name, 66
  - susceptibility\_reduction\_mixer, 68
  - time\_elapsed, 68

- transmission\_reduction\_mixer, 68
- write\_data, 66
- ModelDiagram, 69
- ModelDiffNet< TSeq >, 69
- ModelMeaslesQuarantine
  - ModelMeaslesQuarantine< TSeq >, 73
- ModelMeaslesQuarantine< TSeq >, 70
  - clone\_ptr, 74
  - ModelMeaslesQuarantine, 73
  - quarantine\_agents, 74
  - reset, 74
- ModelSEIR< TSeq >, 75
  - initial\_states, 77
  - update\_exposed\_seir, 77
  - update\_infected\_seir, 77
- ModelSEIRCONN
  - ModelSEIRCONN< TSeq >, 79
- ModelSEIRCONN< TSeq >, 78
  - initial\_states, 79
  - ModelSEIRCONN, 79
- ModelSEIRD
  - ModelSEIRD< TSeq >, 81, 82
- ModelSEIRD< TSeq >, 80
  - ModelSEIRD, 81, 82
  - update\_exposed\_seir, 82
- ModelSEIRDCONN
  - ModelSEIRDCONN< TSeq >, 84
- ModelSEIRDCONN< TSeq >, 83
  - initial\_states, 85
  - ModelSEIRDCONN, 84
- ModelSEIRMixing
  - ModelSEIRMixing< TSeq >, 86, 87
- ModelSEIRMixing< TSeq >, 85
  - initial\_states, 88
  - ModelSEIRMixing, 86, 87
- ModelSIR< TSeq >, 88
  - initial\_states, 90
- ModelSIRCONN
  - ModelSIRCONN< TSeq >, 91
- ModelSIRCONN< TSeq >, 90
  - get\_n\_infected, 92
  - initial\_states, 92
  - ModelSIRCONN, 91
- ModelSIRD
  - ModelSIRD< TSeq >, 94
- ModelSIRD< TSeq >, 93
  - initial\_states, 95
  - ModelSIRD, 94
- ModelSIRDCONN
  - ModelSIRDCONN< TSeq >, 96
- ModelSIRDCONN< TSeq >, 95
  - ModelSIRDCONN, 96
- ModelSIRLogit
  - ModelSIRLogit< TSeq >, 99
- ModelSIRLogit< TSeq >, 97
  - ModelSIRLogit, 99
- ModelSIRMixing
  - ModelSIRMixing< TSeq >, 101
- ModelSIRMixing< TSeq >, 100
  - initial\_states, 102
  - ModelSIRMixing, 101
- ModelSIS< TSeq >, 102
- ModelSISD< TSeq >, 104
- ModelSURV< TSeq >, 105
- Network< Nettype, Nodetype, Edgetype >, 107
- operator()
  - Agent< TSeq >, 39
- operator==
  - DataBase< TSeq >, 45, 46
- PersonTools< TSeq >, 108
- Progress, 108
- quarantine\_agents
  - ModelMeaslesQuarantine< TSeq >, 74
- Queue< TSeq >, 108
- RandGraph, 109
- read\_edgelist
  - AdjList, 36
- record\_virus
  - DataBase< TSeq >, 46
- recovery\_enhancer\_mixer
  - Model< TSeq >, 67
- reset
  - Model< TSeq >, 65
  - ModelMeaslesQuarantine< TSeq >, 74
- rnbinomd
  - Model< TSeq >, 67
- run\_multiple
  - Model< TSeq >, 65
- runifd
  - Model< TSeq >, 67
- sample\_virus\_single
  - sampler, 32
- sampler, 31
  - make\_sample\_virus\_neighbors, 31
  - make\_update\_susceptible, 32
  - sample\_virus\_single, 32
- SAMPLETYPE, 110
- set\_agents\_data
  - Model< TSeq >, 65
- set\_name
  - Model< TSeq >, 66
- susceptibility\_reduction\_mixer
  - Model< TSeq >, 68
- swap\_neighbors
  - Agent< TSeq >, 39
- time\_elapsed
  - Model< TSeq >, 68
- Tool< TSeq >, 110
- ToolFunctions< TSeq >, 111
- Tools< TSeq >, 112
- Tools\_const< TSeq >, 113

transmission\_reduction\_mixer  
    Model< TSeq >, [68](#)

update\_exposed\_seir  
    ModelSEIR< TSeq >, [77](#)  
    ModelSEIRD< TSeq >, [82](#)

update\_infected\_seir  
    ModelSEIR< TSeq >, [77](#)

UserData  
    UserData< TSeq >, [115](#)

UserData< TSeq >, [114](#)  
    UserData, [115](#)

vecHasher< T >, [115](#)

Virus< TSeq >, [116](#)

Viruses< TSeq >, [118](#)

Viruses\_const< TSeq >, [119](#)

VirusFunctions< TSeq >, [119](#)

write\_data  
    Model< TSeq >, [66](#)