

epiworld

0.0-1

Generated by Doxygen 1.9.1

1 Example: 00-hello-world	1
2 Benchmarking	3
3 Contributor Code of Conduct	5
4 epiworld c++ template library	7
4.1 Main features	7
4.2 Algorithm	7
4.3 Hello world (C++)	8
4.4 Surveillance simulation	8
4.4.1 Preliminary results	9
4.4.2 Cases detected	10
5 MIT License	11
6 model1	13
7 Mixing probabilities in connected model	15
7.1 Case 1: No grouping	15
7.2 Case 2: Grouping	16
8 EPI Simulator	17
8.1 Disease dynamics	17
8.2 Network dynamics	17
8.3 Contagion dynamics	17
8.4 Time dynamics	17
8.5 Updating agent's status	18
8.5.1 Other parameters	18
9 Namespace Index	19
9.1 Namespace List	19
10 Hierarchical Index	21
10.1 Class Hierarchy	21
11 Class Index	23
11.1 Class List	23
12 File Index	27
12.1 File List	27
13 Namespace Documentation	29
13.1 epiworld::sampler Namespace Reference	29
13.1.1 Detailed Description	29
13.1.2 Function Documentation	29
13.1.2.1 make_sample_virus_neighbors()	29

13.1.2.2 make_update_susceptible()	30
13.1.2.3 sample_virus_single()	30
13.2 sampler Namespace Reference	32
13.2.1 Detailed Description	32
13.2.2 Function Documentation	32
13.2.2.1 make_sample_virus_neighbors()	32
13.2.2.2 make_update_susceptible()	33
13.2.2.3 sample_virus_single()	33
14 Class Documentation	37
14.1 AdjList Class Reference	37
14.1.1 Constructor & Destructor Documentation	37
14.1.1.1 AdjList()	37
14.1.2 Member Function Documentation	38
14.1.2.1 read_edgelist()	38
14.2 epiworld::AdjList Class Reference	38
14.2.1 Constructor & Destructor Documentation	39
14.2.1.1 AdjList()	39
14.2.2 Member Function Documentation	39
14.2.2.1 read_edgelist()	39
14.3 Agent< TSeq > Class Template Reference	40
14.3.1 Detailed Description	42
14.3.2 Member Function Documentation	42
14.3.2.1 operator>()	42
14.3.2.2 swap_neighbors()	43
14.3.3 Friends And Related Function Documentation	43
14.3.3.1 default_rm_entity	43
14.4 epiworld::Agent< TSeq > Class Template Reference	44
14.4.1 Detailed Description	46
14.4.2 Member Function Documentation	46
14.4.2.1 operator>()	46
14.4.2.2 swap_neighbors()	47
14.4.3 Friends And Related Function Documentation	47
14.4.3.1 default_rm_entity	47
14.5 AgentsSample< TSeq > Class Template Reference	47
14.5.1 Detailed Description	48
14.5.2 Constructor & Destructor Documentation	48
14.5.2.1 AgentsSample()	48
14.6 epiworld::AgentsSample< TSeq > Class Template Reference	49
14.6.1 Detailed Description	49
14.6.2 Constructor & Destructor Documentation	50
14.6.2.1 AgentsSample()	50

14.7 DataBase< TSeq > Class Template Reference	50
14.7.1 Detailed Description	52
14.7.2 Member Function Documentation	52
14.7.2.1 generation_time()	53
14.7.2.2 get_transmissions()	53
14.7.2.3 operator==() [1/3]	53
14.7.2.4 operator==() [2/3]	54
14.7.2.5 operator==() [3/3]	54
14.7.2.6 record_virus()	54
14.7.2.7 reproductive_number()	54
14.7.2.8 transition_probability()	55
14.8 epiworld::DataBase< TSeq > Class Template Reference	55
14.8.1 Detailed Description	57
14.8.2 Member Function Documentation	57
14.8.2.1 generation_time()	57
14.8.2.2 get_transmissions()	58
14.8.2.3 operator==()	58
14.8.2.4 record_virus()	58
14.8.2.5 reproductive_number()	59
14.8.2.6 transition_probability()	59
14.9 Entities< TSeq > Class Template Reference	59
14.9.1 Detailed Description	60
14.10 epiworld::Entities< TSeq > Class Template Reference	60
14.10.1 Detailed Description	61
14.11 Entities_const< TSeq > Class Template Reference	61
14.11.1 Detailed Description	61
14.12 epiworld::Entities_const< TSeq > Class Template Reference	62
14.12.1 Detailed Description	62
14.13 Entity< TSeq > Class Template Reference	63
14.13.1 Friends And Related Function Documentation	63
14.13.1.1 default_rm_entity	63
14.14 epiworld::Entity< TSeq > Class Template Reference	64
14.14.1 Friends And Related Function Documentation	64
14.14.1.1 default_rm_entity	64
14.15 epiworld::Event< TSeq > Struct Template Reference	65
14.15.1 Detailed Description	65
14.15.2 Constructor & Destructor Documentation	65
14.15.2.1 Event()	65
14.16 Event< TSeq > Struct Template Reference	66
14.16.1 Detailed Description	67
14.16.2 Constructor & Destructor Documentation	67
14.16.2.1 Event()	67

14.17 epiworld::GlobalEvent< TSeq > Class Template Reference	68
14.17.1 Detailed Description	68
14.17.2 Constructor & Destructor Documentation	69
14.17.2.1 GlobalEvent()	69
14.18 GlobalEvent< TSeq > Class Template Reference	69
14.18.1 Detailed Description	69
14.18.2 Constructor & Destructor Documentation	70
14.18.2.1 GlobalEvent()	70
14.19 GroupSampler< TSeq > Class Template Reference	70
14.19.1 Detailed Description	70
14.20 epiworld::LFMCMC< TData > Class Template Reference	71
14.20.1 Detailed Description	71
14.21 LFMCMC< TData > Class Template Reference	72
14.21.1 Detailed Description	73
14.22 epiworld::Model< TSeq > Class Template Reference	73
14.22.1 Detailed Description	81
14.22.2 Member Function Documentation	81
14.22.2.1 add_globlevent()	81
14.22.2.2 clone_ptr()	82
14.22.2.3 events_add()	82
14.22.2.4 events_run()	83
14.22.2.5 load_agents_entities_ties()	83
14.22.2.6 reset()	83
14.22.2.7 run_multiple()	84
14.22.2.8 set_agents_data()	84
14.22.2.9 set_name()	85
14.22.2.10 write_data()	85
14.22.3 Member Data Documentation	86
14.22.3.1 initial_states_fun	86
14.22.3.2 rbinomd	86
14.22.3.3 rexp	86
14.22.3.4 rgammad	86
14.22.3.5 rlognormald	87
14.22.3.6 rnormd	87
14.22.3.7 runifd	87
14.22.3.8 time_elapsed	87
14.23 Model< TSeq > Class Template Reference	88
14.23.1 Detailed Description	96
14.23.2 Member Function Documentation	96
14.23.2.1 add_globlevent()	96
14.23.2.2 clone_ptr()	96
14.23.2.3 events_add()	97

14.23.2.4 events_run()	97
14.23.2.5 load_agents_entities_ties()	98
14.23.2.6 reset()	98
14.23.2.7 run_multiple()	98
14.23.2.8 set_agents_data()	99
14.23.2.9 set_name()	99
14.23.2.10 write_data()	99
14.23.3 Member Data Documentation	100
14.23.3.1 initial_states_fun	100
14.23.3.2 rbinomd	100
14.23.3.3 rexp	100
14.23.3.4 rgammad	101
14.23.3.5 rlognormald	101
14.23.3.6 rnormd	101
14.23.3.7 runifd	101
14.23.3.8 time_elapsed	101
14.24 epiworld::epimodels::ModelDiffNet< TSeq > Class Template Reference	102
14.24.1 Detailed Description	103
14.25 ModelDiffNet< TSeq > Class Template Reference	103
14.25.1 Detailed Description	105
14.26 epiworld::epimodels::ModelSEIR< TSeq > Class Template Reference	105
14.26.1 Detailed Description	106
14.26.2 Member Function Documentation	107
14.26.2.1 initial_states()	107
14.26.3 Member Data Documentation	107
14.26.3.1 update_exposed_seir	107
14.26.3.2 update_infected_seir	108
14.27 ModelSEIR< TSeq > Class Template Reference	108
14.27.1 Detailed Description	109
14.27.2 Member Function Documentation	109
14.27.2.1 initial_states()	109
14.27.3 Member Data Documentation	110
14.27.3.1 update_exposed_seir	110
14.27.3.2 update_infected_seir	110
14.28 epiworld::epimodels::ModelSEIRCONN< TSeq > Class Template Reference	111
14.28.1 Constructor & Destructor Documentation	112
14.28.1.1 ModelSEIRCONN()	112
14.28.2 Member Function Documentation	112
14.28.2.1 clone_ptr()	113
14.28.2.2 initial_states()	113
14.28.2.3 reset()	113
14.29 ModelSEIRCONN< TSeq > Class Template Reference	114

14.29.1 Constructor & Destructor Documentation	115
14.29.1.1 ModelSEIRCONN()	115
14.29.2 Member Function Documentation	115
14.29.2.1 clone_ptr()	115
14.29.2.2 initial_states()	116
14.29.2.3 reset()	116
14.30 epiworld::epimodels::ModelSEIRD< TSeq > Class Template Reference	117
14.30.1 Detailed Description	118
14.30.2 Constructor & Destructor Documentation	118
14.30.2.1 ModelSEIRD() [1/2]	118
14.30.2.2 ModelSEIRD() [2/2]	119
14.30.3 Member Data Documentation	119
14.30.3.1 update_exposed_seir	119
14.31 ModelSEIRD< TSeq > Class Template Reference	120
14.31.1 Detailed Description	121
14.31.2 Constructor & Destructor Documentation	121
14.31.2.1 ModelSEIRD() [1/2]	121
14.31.2.2 ModelSEIRD() [2/2]	122
14.31.3 Member Data Documentation	122
14.31.3.1 update_exposed_seir	122
14.32 epiworld::epimodels::ModelSEIRDCONN< TSeq > Class Template Reference	123
14.32.1 Constructor & Destructor Documentation	124
14.32.1.1 ModelSEIRDCONN()	124
14.32.2 Member Function Documentation	125
14.32.2.1 clone_ptr()	125
14.32.2.2 initial_states()	125
14.32.2.3 reset()	125
14.33 ModelSEIRDCONN< TSeq > Class Template Reference	126
14.33.1 Constructor & Destructor Documentation	127
14.33.1.1 ModelSEIRDCONN()	127
14.33.2 Member Function Documentation	128
14.33.2.1 clone_ptr()	128
14.33.2.2 initial_states()	128
14.33.2.3 reset()	129
14.34 ModelSEIRMixing< TSeq > Class Template Reference	129
14.34.1 Constructor & Destructor Documentation	130
14.34.1.1 ModelSEIRMixing() [1/2]	131
14.34.1.2 ModelSEIRMixing() [2/2]	131
14.34.2 Member Function Documentation	132
14.34.2.1 clone_ptr()	132
14.34.2.2 initial_states()	132
14.34.2.3 reset()	133

14.35 epiworld::epimodels::ModelSIR< TSeq > Class Template Reference	133
14.35.1 Detailed Description	134
14.35.2 Member Function Documentation	134
14.35.2.1 <code>initial_states()</code>	135
14.36 ModelSIR< TSeq > Class Template Reference	135
14.36.1 Detailed Description	136
14.36.2 Member Function Documentation	136
14.36.2.1 <code>initial_states()</code>	137
14.37 epiworld::epimodels::ModelSIRCONN< TSeq > Class Template Reference	137
14.37.1 Constructor & Destructor Documentation	138
14.37.1.1 <code>ModelSIRCONN()</code>	139
14.37.2 Member Function Documentation	139
14.37.2.1 <code>clone_ptr()</code>	139
14.37.2.2 <code>get_n_infected()</code>	139
14.37.2.3 <code>initial_states()</code>	140
14.37.2.4 <code>reset()</code>	140
14.38 ModelSIRCONN< TSeq > Class Template Reference	141
14.38.1 Constructor & Destructor Documentation	142
14.38.1.1 <code>ModelSIRCONN()</code>	142
14.38.2 Member Function Documentation	142
14.38.2.1 <code>clone_ptr()</code>	142
14.38.2.2 <code>get_n_infected()</code>	143
14.38.2.3 <code>initial_states()</code>	143
14.38.2.4 <code>reset()</code>	143
14.39 epiworld::epimodels::ModelSIRD< TSeq > Class Template Reference	144
14.39.1 Detailed Description	145
14.39.2 Constructor & Destructor Documentation	145
14.39.2.1 <code>ModelSIRD()</code>	145
14.39.3 Member Function Documentation	146
14.39.3.1 <code>initial_states()</code>	146
14.40 ModelSIRD< TSeq > Class Template Reference	146
14.40.1 Detailed Description	147
14.40.2 Constructor & Destructor Documentation	147
14.40.2.1 <code>ModelSIRD()</code>	147
14.40.3 Member Function Documentation	148
14.40.3.1 <code>initial_states()</code>	148
14.41 epiworld::epimodels::ModelSIRDCONN< TSeq > Class Template Reference	149
14.41.1 Constructor & Destructor Documentation	150
14.41.1.1 <code>ModelSIRDCONN()</code>	150
14.41.2 Member Function Documentation	150
14.41.2.1 <code>clone_ptr()</code>	150
14.41.2.2 <code>reset()</code>	151

14.42 ModelSIRDConn< TSeq > Class Template Reference	151
14.42.1 Constructor & Destructor Documentation	152
14.42.1.1 ModelSIRDConn()	153
14.42.2 Member Function Documentation	153
14.42.2.1 clone_ptr()	153
14.42.2.2 reset()	153
14.43 epiworld::epimodels::ModelSIRLogit< TSeq > Class Template Reference	154
14.43.1 Detailed Description	155
14.43.2 Constructor & Destructor Documentation	156
14.43.2.1 ModelSIRLogit()	156
14.43.3 Member Function Documentation	157
14.43.3.1 clone_ptr()	157
14.43.3.2 reset()	157
14.44 ModelSIRLogit< TSeq > Class Template Reference	158
14.44.1 Detailed Description	159
14.44.2 Constructor & Destructor Documentation	159
14.44.2.1 ModelSIRLogit()	159
14.44.3 Member Function Documentation	160
14.44.3.1 clone_ptr()	160
14.44.3.2 reset()	160
14.45 epiworld::epimodels::ModelSIS< TSeq > Class Template Reference	161
14.45.1 Detailed Description	162
14.46 ModelSIS< TSeq > Class Template Reference	162
14.46.1 Detailed Description	163
14.47 epiworld::epimodels::ModelSISD< TSeq > Class Template Reference	164
14.47.1 Detailed Description	165
14.48 ModelSISD< TSeq > Class Template Reference	165
14.48.1 Detailed Description	166
14.49 epiworld::epimodels::ModelSURV< TSeq > Class Template Reference	167
14.50 ModelSURV< TSeq > Class Template Reference	169
14.51 Network< Nettype, Nodetype, Edgetype > Class Template Reference	171
14.52 epiworld::PersonTools< TSeq > Class Template Reference	171
14.53 PersonTools< TSeq > Class Template Reference	171
14.54 epiworld::Progress Class Reference	171
14.54.1 Detailed Description	172
14.55 Progress Class Reference	172
14.55.1 Detailed Description	172
14.56 epiworld::Queue< TSeq > Class Template Reference	172
14.56.1 Detailed Description	173
14.57 Queue< TSeq > Class Template Reference	173
14.57.1 Detailed Description	174
14.58 RandGraph Class Reference	174

14.59	epiworld::SAMPLETYPE Class Reference	174
14.60	SAMPLETYPE Class Reference	175
14.61	epiworld::Tool< TSeq > Class Template Reference	175
14.61.1	Detailed Description	176
14.62	Tool< TSeq > Class Template Reference	176
14.62.1	Detailed Description	178
14.63	epiworld::Tools< TSeq > Class Template Reference	178
14.63.1	Detailed Description	178
14.64	Tools< TSeq > Class Template Reference	179
14.64.1	Detailed Description	179
14.65	epiworld::Tools_const< TSeq > Class Template Reference	180
14.65.1	Detailed Description	180
14.66	Tools_const< TSeq > Class Template Reference	180
14.66.1	Detailed Description	181
14.67	epiworld::UserData< TSeq > Class Template Reference	181
14.67.1	Detailed Description	182
14.67.2	Constructor & Destructor Documentation	183
14.67.2.1	UserData()	183
14.68	UserData< TSeq > Class Template Reference	183
14.68.1	Detailed Description	184
14.68.2	Constructor & Destructor Documentation	184
14.68.2.1	UserData()	184
14.69	epiworld::vecHasher< T > Struct Template Reference	185
14.69.1	Detailed Description	185
14.70	vecHasher< T > Struct Template Reference	185
14.70.1	Detailed Description	185
14.71	epiworld::Virus< TSeq > Class Template Reference	186
14.71.1	Detailed Description	187
14.72	Virus< TSeq > Class Template Reference	188
14.72.1	Detailed Description	189
14.73	epiworld::Viruses< TSeq > Class Template Reference	190
14.73.1	Detailed Description	190
14.74	Viruses< TSeq > Class Template Reference	191
14.74.1	Detailed Description	191
14.75	epiworld::Viruses_const< TSeq > Class Template Reference	191
14.75.1	Detailed Description	192
14.76	Viruses_const< TSeq > Class Template Reference	192
14.76.1	Detailed Description	193
15	File Documentation	195
15.1	include/epiworld/agent-meat-state.hpp File Reference	195
15.1.1	Detailed Description	196

Chapter 1

Example: 00-hello-world

Output from the program:

```
Running the model...
||||| done.
done.
```

```
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     : 16.00ms
Last run speed         : 59.75 million agents x day / second
Rewiring               : off
Global events:
  (none)
Virus(es):
  - covid 19 (baseline prevalence: 50 seeds)
Tool(s):
  - vaccine (baseline prevalence: 50.00%)
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 0.87 0.13 0.00 0.00
  - Exposed      0.00 0.83 0.15 0.01
  - Recovered    0.00 0.00 1.00 0.00
  - Removed      0.00 0.00 0.00 1.00
```

Chapter 2

Benchmarking

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

Chapter 3

Contributor Code of Conduct

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

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

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

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

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

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

Chapter 4

epiworld c++ template library

4.1 Main features

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

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

4.2 Algorithm

Setup

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

Run

1. Distribute the tool(s) and virus(es)
2. For each t in 1 -> Duration:
 - Update state for susceptible/infected/removed(?)
 - Mutate virus(es) (each individual)
 - Run Global 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");
    covid19.set_infectiousness(.8);

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

4.4 Surveillance simulation

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

4.4.1 Preliminary results

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

## Running the model...
##
## | done.
##
##
## SIMULATION STUDY
##
## Population size      : 20000
## Days (duration)     : 200 (of 200)
## Number of variants  : 1
## Last run elapsed t   : 505.00ms
## Rewiring            : off
##
## Virus(es):
## - Covid19 (baseline prevalence: 100 seeds)
## Tool(s):
## - Vaccine (baseline prevalence: 25.00%)
##
## Model parameters:
## - Infect period      : 12.0000
## - Latent period      : 3.0000
## - Prob of symptoms   : 0.7000
## - Prob of transmission : 1.0000
## - Prob. death        : 0.0010
## - Prob. reinfect     : 0.1000
## - Surveillance prob. : 1.0e-04
## - Vax efficacy       : 0.9000
## - Vax redux transmission : 0.5000
##
## Distribution of the population at time 200:
## - Total susceptible (S) : 19900 -> 2106
## - Total recovered (S)   : 0 -> 17369
## - Total latent (I)      : 100 -> 109
## - Total symptomatic (I) : 0 -> 155
## - Total symptomatic isolated (I) : 0 -> 2
## - Total asymptomatic (I) : 0 -> 72
## - Total asymptomatic isolated (I) : 0 -> 0
## - Total removed (R)    : 0 -> 187
##
## (S): Susceptible, (I): Infected, (R): Recovered
##
hist1 <- read.csv("07-surveillance_hist.txt", sep = " ")
surv1 <- read.csv("07-surveillance_user_data.txt", sep = " ")
# With high surveillance
system(sprintf("./07-surveillance.o %i %i 100 %.04f 2>&1", pop_seed, pop_size, s_levels[2]), intern = TRUE)
|>
cat(sep = "\n")

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

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

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

4.4.2 Cases detected

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

Chapter 5

MIT License

Copyright (c) 2021 George G. Vega Yon

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

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

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

Chapter 6

model1

The dynamics of the simulation process are:

1. Discrete Markov process.

2. The simulation has the following parameters:

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

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

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

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

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

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

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

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

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

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

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

3. The simulation process is as follows:

- (a) Countries are initialized with a total population  $N(i)$ .
- (b) Variant zero initializes at a random location  $i$ , with an initial prevalence  $P(k, t) = N(i, t, k)$ .
- (c) For time  $t$  in  $(0, T)$  do:
  - a. Unvaccinated individuals can become sick of variant  $k$  with probability:  

$$\Pr(h \rightarrow s | i, t, k, u) \sim \sum(g \in \{u, v\}) (N(i, t-1, s, k | g) + \sum(j \neq i) F(i, j) * N(j, t-1, s, k | g)) * C(k) / (N(i) + \sum(j \neq i) N(j))$$
  - b. Vaccinated individuals can become sick of variant  $k$  with probability:  $\Pr(v \rightarrow s | i, t, k, v) \sim \Pr(h \rightarrow s | i, t, k) * (1 - E(v, k))$ .
  - b. Recovered individuals can become sick of variant  $k$  with probability:  $\Pr(v \rightarrow s | i, t, k, r) \sim \Pr(h \rightarrow s | i, t, k) * (1 - E(r, k))$ .
  - c. Sick individuals with variant  $k$  die with probability  $D(k)$  or recover with probability  $H(k)$ , otherwise they stay infected; with the rates depending on their vaccination status  $v$  or  $n$ .
  - d. Unvaccinated individuals vaccinate in country  $i$  with probability  $P(u \rightarrow v) \sim V(A(i, t), B(i))$ .
  - e. The country vaccine supply changes.

## Chapter 7

# Mixing probabilities in connected model

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

### 7.1 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.

## 7.2 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 8

# EPI Simulator

### 8.1 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.

### 8.2 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.

### 8.3 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.

### 8.4 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

## 8.5 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.

### 8.5.1 Other parameters

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

## Chapter 9

# Namespace Index

### 9.1 Namespace List

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

<a href="#">epiworld::sampler</a>	Functions for sampling viruses . . . . .	<a href="#">29</a>
<a href="#">sampler</a>	Functions for sampling viruses . . . . .	<a href="#">32</a>





## Chapter 10

# Hierarchical Index

### 10.1 Class Hierarchy

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

AdjList . . . . .	37
epiworld::AdjList . . . . .	38
Agent< TSeq > . . . . .	40
epiworld::Agent< TSeq > . . . . .	44
AgentsSample< TSeq > . . . . .	47
epiworld::AgentsSample< TSeq > . . . . .	49
DataBase< TSeq > . . . . .	50
epiworld::DataBase< TSeq > . . . . .	55
Entities< TSeq > . . . . .	59
epiworld::Entities< TSeq > . . . . .	60
Entities_const< TSeq > . . . . .	61
epiworld::Entities_const< TSeq > . . . . .	62
Entity< TSeq > . . . . .	63
epiworld::Entity< TSeq > . . . . .	64
epiworld::Event< TSeq > . . . . .	65
Event< TSeq > . . . . .	66
epiworld::GlobalEvent< TSeq > . . . . .	68
GlobalEvent< TSeq > . . . . .	69
GroupSampler< TSeq > . . . . .	70
epiworld::LFMCMC< TData > . . . . .	71
LFMCMC< TData > . . . . .	72
epiworld::Model< TSeq > . . . . .	73
Model< TSeq > . . . . .	88
epiworld::Model< EPI_DEFAULT_TSEQ > . . . . .	73
ModelSEIRCONN< TSeq > . . . . .	114
ModelSEIRDCONN< TSeq > . . . . .	126
ModelSEIRMixing< TSeq > . . . . .	129
ModelSIRCONN< TSeq > . . . . .	141
ModelSIRDCONN< TSeq > . . . . .	151
ModelSIRLogit< TSeq > . . . . .	158
ModelSURV< TSeq > . . . . .	169
epiworld::epimodels::ModelSEIRCONN< TSeq > . . . . .	111
epiworld::epimodels::ModelSEIRDCONN< TSeq > . . . . .	123
epiworld::epimodels::ModelSIRCONN< TSeq > . . . . .	137
epiworld::epimodels::ModelSIRDCONN< TSeq > . . . . .	149

epiworld::epimodels::ModelSIRLogit< TSeq > . . . . .	154
epiworld::epimodels::ModelSURV< TSeq > . . . . .	167
epiworld::Model< int > . . . . .	73
ModelDiffNet< TSeq > . . . . .	103
ModelSEIR< TSeq > . . . . .	108
ModelSEIRD< TSeq > . . . . .	120
ModelSIR< TSeq > . . . . .	135
ModelSIRD< TSeq > . . . . .	146
ModelSIS< TSeq > . . . . .	162
ModelSISD< TSeq > . . . . .	165
epiworld::epimodels::ModelDiffNet< TSeq > . . . . .	102
epiworld::epimodels::ModelSEIR< TSeq > . . . . .	105
epiworld::epimodels::ModelSEIRD< TSeq > . . . . .	117
epiworld::epimodels::ModelSIR< TSeq > . . . . .	133
epiworld::epimodels::ModelSIRD< TSeq > . . . . .	144
epiworld::epimodels::ModelSIS< TSeq > . . . . .	161
epiworld::epimodels::ModelSISD< TSeq > . . . . .	164
Network< Nettype, Nodetype, Edgetype > . . . . .	171
epiworld::PersonTools< TSeq > . . . . .	171
PersonTools< TSeq > . . . . .	171
epiworld::Progress . . . . .	171
Progress . . . . .	172
epiworld::Queue< TSeq > . . . . .	172
Queue< TSeq > . . . . .	173
RandGraph . . . . .	174
epiworld::SAMPLETYPE . . . . .	174
SAMPLETYPE . . . . .	175
epiworld::Tool< TSeq > . . . . .	175
Tool< TSeq > . . . . .	176
epiworld::Tools< TSeq > . . . . .	178
Tools< TSeq > . . . . .	179
epiworld::Tools_const< TSeq > . . . . .	180
Tools_const< TSeq > . . . . .	180
epiworld::UserData< TSeq > . . . . .	181
UserData< TSeq > . . . . .	183
epiworld::vecHasher< T > . . . . .	185
vecHasher< T > . . . . .	185
epiworld::Virus< TSeq > . . . . .	186
Virus< TSeq > . . . . .	188
epiworld::Viruses< TSeq > . . . . .	190
Viruses< TSeq > . . . . .	191
epiworld::Viruses_const< TSeq > . . . . .	191
Viruses_const< TSeq > . . . . .	192

## Chapter 11

# Class Index

### 11.1 Class List

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

<a href="#">AdjList</a> . . . . .	37
<a href="#">epiworld::AdjList</a> . . . . .	38
<a href="#">Agent&lt; TSeq &gt;</a>	
<a href="#">Agent</a> (agents) . . . . .	40
<a href="#">epiworld::Agent&lt; TSeq &gt;</a>	
<a href="#">Agent</a> (agents) . . . . .	44
<a href="#">AgentsSample&lt; TSeq &gt;</a>	
Sample of agents . . . . .	47
<a href="#">epiworld::AgentsSample&lt; TSeq &gt;</a>	
Sample of agents . . . . .	49
<a href="#">DataBase&lt; TSeq &gt;</a>	
Statistical data about the process . . . . .	50
<a href="#">epiworld::DataBase&lt; TSeq &gt;</a>	
Statistical data about the process . . . . .	55
<a href="#">Entities&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (useful for building iterators) . . . . .	59
<a href="#">epiworld::Entities&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (useful for building iterators) . . . . .	60
<a href="#">Entities_const&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (const) (useful for iterators) . . . . .	61
<a href="#">epiworld::Entities_const&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (const) (useful for iterators) . . . . .	62
<a href="#">Entity&lt; TSeq &gt;</a> . . . . .	63
<a href="#">epiworld::Entity&lt; TSeq &gt;</a> . . . . .	64
<a href="#">epiworld::Event&lt; TSeq &gt;</a>	
<a href="#">Event</a> data for update an agent . . . . .	65
<a href="#">Event&lt; TSeq &gt;</a>	
<a href="#">Event</a> data for update an agent . . . . .	66
<a href="#">epiworld::GlobalEvent&lt; TSeq &gt;</a>	
Template for a Global <a href="#">Event</a> . . . . .	68
<a href="#">GlobalEvent&lt; TSeq &gt;</a>	
Template for a Global <a href="#">Event</a> . . . . .	69
<a href="#">GroupSampler&lt; TSeq &gt;</a>	
Weighted sampling of groups . . . . .	70
<a href="#">epiworld::LFMCMC&lt; TData &gt;</a>	
Likelihood-Free Markov Chain Monte Carlo . . . . .	71

<a href="#">LFMCMC&lt; TData &gt;</a>	
Likelihood-Free Markov Chain Monte Carlo	72
<a href="#">epiworld::Model&lt; TSeq &gt;</a>	
Core class of epiworld	73
<a href="#">Model&lt; TSeq &gt;</a>	
Core class of epiworld	88
<a href="#">epiworld::epimodels::ModelDiffNet&lt; TSeq &gt;</a>	
Template for a <a href="#">Network</a> Diffusion <a href="#">Model</a>	102
<a href="#">ModelDiffNet&lt; TSeq &gt;</a>	
Template for a <a href="#">Network</a> Diffusion <a href="#">Model</a>	103
<a href="#">epiworld::epimodels::ModelSEIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed (SEIR) model	105
<a href="#">ModelSEIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed (SEIR) model	108
<a href="#">epiworld::epimodels::ModelSEIRCONN&lt; TSeq &gt;</a>	111
<a href="#">ModelSEIRCONN&lt; TSeq &gt;</a>	114
<a href="#">epiworld::epimodels::ModelSEIRD&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model	117
<a href="#">ModelSEIRD&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model	120
<a href="#">epiworld::epimodels::ModelSEIRDCONN&lt; TSeq &gt;</a>	123
<a href="#">ModelSEIRDCONN&lt; TSeq &gt;</a>	126
<a href="#">ModelSEIRMixing&lt; TSeq &gt;</a>	129
<a href="#">epiworld::epimodels::ModelSIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed (SIR) model	133
<a href="#">ModelSIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed (SIR) model	135
<a href="#">epiworld::epimodels::ModelSIRCONN&lt; TSeq &gt;</a>	137
<a href="#">ModelSIRCONN&lt; TSeq &gt;</a>	141
<a href="#">epiworld::epimodels::ModelSIRD&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed-Deceased (SIRD) model	144
<a href="#">ModelSIRD&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed-Deceased (SIRD) model	146
<a href="#">epiworld::epimodels::ModelSIRDCONN&lt; TSeq &gt;</a>	149
<a href="#">ModelSIRDCONN&lt; TSeq &gt;</a>	151
<a href="#">epiworld::epimodels::ModelSIRLogit&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed (SIR) model	154
<a href="#">ModelSIRLogit&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed (SIR) model	158
<a href="#">epiworld::epimodels::ModelSIS&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Susceptible (SIS) model	161
<a href="#">ModelSIS&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Susceptible (SIS) model	162
<a href="#">epiworld::epimodels::ModelSISD&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Susceptible-Deceased (SISD) model	164
<a href="#">ModelSISD&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Susceptible-Deceased (SISD) model	165
<a href="#">epiworld::epimodels::ModelSURV&lt; TSeq &gt;</a>	167
<a href="#">ModelSURV&lt; TSeq &gt;</a>	169
<a href="#">Network&lt; Nettype, Nodetype, Edgetype &gt;</a>	171
<a href="#">epiworld::PersonTools&lt; TSeq &gt;</a>	171
<a href="#">PersonTools&lt; TSeq &gt;</a>	171
<a href="#">epiworld::Progress</a>	
A simple progress bar	171
<a href="#">Progress</a>	
A simple progress bar	172
<a href="#">epiworld::Queue&lt; TSeq &gt;</a>	
Controls which agents are verified at each step	172

<a href="#">Queue&lt; TSeq &gt;</a>	
Controls which agents are verified at each step	173
<a href="#">RandGraph</a>	174
<a href="#">epiworld::SAMPLETYPE</a>	174
<a href="#">SAMPLETYPE</a>	175
<a href="#">epiworld::Tool&lt; TSeq &gt;</a>	
Tools for defending the agent against the virus	175
<a href="#">Tool&lt; TSeq &gt;</a>	
Tools for defending the agent against the virus	176
<a href="#">epiworld::Tools&lt; TSeq &gt;</a>	
Set of tools (useful for building iterators)	178
<a href="#">Tools&lt; TSeq &gt;</a>	
Set of tools (useful for building iterators)	179
<a href="#">epiworld::Tools_const&lt; TSeq &gt;</a>	
Set of <a href="#">Tools</a> (const) (useful for iterators)	180
<a href="#">Tools_const&lt; TSeq &gt;</a>	
Set of <a href="#">Tools</a> (const) (useful for iterators)	180
<a href="#">epiworld::UserData&lt; TSeq &gt;</a>	
Personalized data by the user	181
<a href="#">UserData&lt; TSeq &gt;</a>	
Personalized data by the user	183
<a href="#">epiworld::vecHasher&lt; T &gt;</a>	
Vector hasher	185
<a href="#">vecHasher&lt; T &gt;</a>	
Vector hasher	185
<a href="#">epiworld::Virus&lt; TSeq &gt;</a>	
Virus	186
<a href="#">Virus&lt; TSeq &gt;</a>	
Virus	188
<a href="#">epiworld::Viruses&lt; TSeq &gt;</a>	
Set of viruses (useful for building iterators)	190
<a href="#">Viruses&lt; TSeq &gt;</a>	
Set of viruses (useful for building iterators)	191
<a href="#">epiworld::Viruses_const&lt; TSeq &gt;</a>	
Set of <a href="#">Viruses</a> (const) (useful for iterators)	191
<a href="#">Viruses_const&lt; TSeq &gt;</a>	
Set of <a href="#">Viruses</a> (const) (useful for iterators)	192



## Chapter 12

# File Index

### 12.1 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	195
include/epiworld/ <b>agent-meat-virus-sampling.hpp</b>	??
include/epiworld/ <b>agent-meat.hpp</b>	??
include/epiworld/ <b>agentssample-bones.hpp</b>	??
include/epiworld/ <b>config.hpp</b>	??
include/epiworld/ <b>database-bones.hpp</b>	??
include/epiworld/ <b>database-meat.hpp</b>	??
include/epiworld/ <b>entities-bones.hpp</b>	??
include/epiworld/ <b>entity-bones.hpp</b>	??
include/epiworld/ <b>entity-meat.hpp</b>	??
include/epiworld/ <b>epiworld-macros.hpp</b>	??
include/epiworld/ <b>epiworld.hpp</b>	??
include/epiworld/ <b>globalevent-bones.hpp</b>	??
include/epiworld/ <b>globalevent-meat.hpp</b>	??
include/epiworld/ <b>groupsampler-bones.hpp</b>	??
include/epiworld/ <b>groupsampler-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>network-bones.hpp</b>	??
include/epiworld/ <b>progress.hpp</b>	??
include/epiworld/ <b>queue-bones.hpp</b>	??
include/epiworld/ <b>randgraph.hpp</b>	??
include/epiworld/ <b>random_graph.hpp</b>	??
include/epiworld/ <b>seq_processing.hpp</b>	??
include/epiworld/ <b>tool-bones.hpp</b>	??
include/epiworld/ <b>tool-meat.hpp</b>	??
include/epiworld/ <b>tools-bones.hpp</b>	??

include/epiworld/ <b>userdata-bones.hpp</b>	??
include/epiworld/ <b>userdata-meat.hpp</b>	??
include/epiworld/ <b>virus-bones.hpp</b>	??
include/epiworld/ <b>virus-meat.hpp</b>	??
include/epiworld/ <b>viruses-bones.hpp</b>	??
include/epiworld/math/ <b>lfmcmc.hpp</b>	??
include/epiworld/math/lfmcmc/ <b>lfmcmc-bones.hpp</b>	??
include/epiworld/math/lfmcmc/ <b>lfmcmc-meat-print.hpp</b>	??
include/epiworld/math/lfmcmc/ <b>lfmcmc-meat.hpp</b>	??
include/epiworld/models/ <b>diffnet.hpp</b>	??
include/epiworld/models/ <b>globalevents.hpp</b>	??
include/epiworld/models/ <b>init-functions.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>sis.hpp</b>	??
include/epiworld/models/ <b>sisd.hpp</b>	??
include/epiworld/models/ <b>surveillance.hpp</b>	??
tests/ <b>tests.hpp</b>	??



## Chapter 13

# Namespace Documentation

### 13.1 epiworld::sampler Namespace Reference

Functions for sampling viruses.

#### Functions

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

#### 13.1.1 Detailed Description

Functions for sampling viruses.

#### 13.1.2 Function Documentation

##### 13.1.2.1 `make_sample_virus_neighbors()`

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

Make a function to sample from neighbors.

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

## Template Parameters

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

## Parameters

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

## Returns

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

## 13.1.2.2 make\_update\_susceptible()

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

Make a function to sample from neighbors.

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

## Template Parameters

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

## Parameters

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

## Returns

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

## 13.1.2.3 sample\_virus\_single()

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

Sample from neighbors pool of viruses (at most one)

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

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

## Template Parameters

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

## Parameters

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

## Returns

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

## 13.2 sampler Namespace Reference

Functions for sampling viruses.

### Functions

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

### 13.2.1 Detailed Description

Functions for sampling viruses.

### 13.2.2 Function Documentation

#### 13.2.2.1 `make_sample_virus_neighbors()`

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

Make a function to sample from neighbors.

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

## Template Parameters

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

## Parameters

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

## Returns

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

## 13.2.2.2 make\_update\_susceptible()

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

Make a function to sample from neighbors.

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

## Template Parameters

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

## Parameters

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

## Returns

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

## 13.2.2.3 sample\_virus\_single()

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

Sample from neighbors pool of viruses (at most one)

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

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

## Template Parameters

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

## Parameters

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

## Returns

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





# Chapter 14

## Class Documentation

### 14.1 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.*

#### 14.1.1 Constructor & Destructor Documentation

##### 14.1.1.1 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

## 14.1.2 Member Function Documentation

### 14.1.2.1 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

## 14.2 epiworld::AdjList Class Reference

### Public Member Functions

- [AdjList](#) (const std::vector< int > &source, const std::vector< int > &target, int size, bool directed)  
*Construct a new Adj List object.*
- [AdjList](#) ([AdjList](#) &&a)
- [AdjList](#) (const [AdjList](#) &a)
- [AdjList](#) & [operator=](#) (const [AdjList](#) &a)
- void [read\\_edgelist](#) (std::string fn, int size, int skip=0, bool directed=true)  
*Read an edgelist.*

- `std::map< int, int > operator()` (`epiworld_fast_uint i`) `const`
- `void print` (`epiworld_fast_uint limit=20u`) `const`
- `size_t vcount` () `const`  
*Number of vertices/nodes in the network.*
- `size_t ecount` () `const`  
*Number of edges/arcs/ties in the network.*
- `std::vector< std::map< int, int > > & get_dat` ()
- `bool is_directed` () `const`  
*true if the network is directed.*

## 14.2.1 Constructor & Destructor Documentation

### 14.2.1.1 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

## 14.2.2 Member Function Documentation

### 14.2.2.1 read\_edgelist()

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

Read an edgelist.

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

## Parameters

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

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

- `epiworld.hpp`

## 14.3 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** ()
- 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 epiworld\_fast\_uint & **get\_state** () const
- void **reset** ()
- bool **has\_tool** (epiworld\_fast\_uint t) const
- bool **has\_tool** (std::string name) const
- bool **has\_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
- 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	<i>Tool to add</i>
virus	<i>Virus to add</i>
state_new	<i>state after the change</i>
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	<i>A pointer to a virus.</i>
---	------------------------------

*Returns*

*epiworld\_double*

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

- 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)

### 14.3.1 Detailed Description

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

[Agent](#) (agents)

Template Parameters

<a href="#">TSeq</a>	Sequence type (should match TSeq across the model)
----------------------	----------------------------------------------------

### 14.3.2 Member Function Documentation

#### 14.3.2.1 operator()

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

Access the j-th column of the agent.

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

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

## Parameters

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

## Returns

double&amp;

**14.3.2.2 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>	

**14.3.3 Friends And Related Function Documentation****14.3.3.1 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

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

[Agent](#) (agents)

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

### Public Member Functions

- **Agent** ([Agent](#)< TSeq > &&p)
- **Agent** (const [Agent](#)< TSeq > &p)
- [Agent](#)< TSeq > & **operator=** (const [Agent](#)< TSeq > &other\_agent)
- int **get\_id** () const  
*Id of the individual.*
- VirusPtr< TSeq > & **get\_virus** ()
- 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 epiworld\_fast\_uint & **get\_state** () const
- void **reset** ()
- bool **has\_tool** (epiworld\_fast\_uint t) const
- bool **has\_tool** (std::string name) const
- bool **has\_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
- 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	<i><a href="#">Tool</a> to add</i>
virus	<i><a href="#">Virus</a> to add</i>
state_new	<i>state after the change</i>
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)

### 14.4.1 Detailed Description

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

[Agent](#) (agents)

Template Parameters

<a href="#">TSeq</a>	Sequence type (should match TSeq across the model)
----------------------	----------------------------------------------------

### 14.4.2 Member Function Documentation

#### 14.4.2.1 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&

#### 14.4.2.2 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>	

### 14.4.3 Friends And Related Function Documentation

#### 14.4.3.1 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 file:

- epiworld.hpp

## 14.5 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

### 14.5.1 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>	
-------------	--

### 14.5.2 Constructor & Destructor Documentation

#### 14.5.2.1 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

## 14.6 epiworld::AgentsSample< TSeq > Class Template Reference

Sample of agents.

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

### Public Member Functions

- [AgentsSample](#) ()=delete  
*Default constructor.*
- [AgentsSample](#) (const [AgentsSample](#)< TSeq > &a)=delete  
*Copy constructor.*
- [AgentsSample](#) ([AgentsSample](#)< TSeq > &&a)=delete  
*Move constructor.*
- [AgentsSample](#) ([Model](#)< TSeq > &model\_, size\_t n, 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

### 14.6.1 Detailed Description

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

Sample of agents.

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

## Template Parameters

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

## 14.6.2 Constructor & Destructor Documentation

### 14.6.2.1 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:

- epiworld.hpp

## 14.7 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
- size\_t [get\\_n\\_tools](#) () const
- void [set\\_user\\_data](#) (std::vector< std::string > names)
- void [add\\_user\\_data](#) (std::vector< epiworld\_double > x)
- void [add\\_user\\_data](#) (epiworld\_fast\_uint j, epiworld\_double x)
- [UserData](#)< TSeq > & [get\\_user\\_data](#) ()
- std::vector< epiworld\_double > [transition\\_probability](#) (bool print=true) const
  - Calculates the transition probabilities.*
- bool [operator==](#) (const [DataBase](#)< TSeq > &other) const
- bool [operator!=](#) (const [DataBase](#)< TSeq > &other) const
- bool [operator==](#) (const [DataBase](#)< std::vector< int >> &other) const
- bool [operator==](#) (const [DataBase](#)< std::vector< int >> &other) const

### Get recorded information from the model

#### Parameters

what	<i>std::string, The state, e.g., 0, 1, 2, ...</i>
------	---------------------------------------------------

#### Returns

*In [get\\_today\\_total](#), the current counts of what.*

*In [get\\_today\\_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\\_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 > **reproductive\_number** () const  
*Computes the reproductive number of each case.*
- void **reproductive\_number** (std::string fn) const
- void **generation\_time** (std::vector< int > &agent\_id, std::vector< int > &virus\_id, std::vector< int > &time, std::vector< int > &gentime) const
- void **generation\_time** (std::string fn) const

## Friends

- class **Model**< TSeq >
- void **default\_add\_virus** (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)

### 14.7.1 Detailed Description

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

Statistical data about the process.

Template Parameters

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

### 14.7.2 Member Function Documentation



## 14.7.2.1 generation\_time()

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

Calculates the generating time

## Parameters

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

## 14.7.2.2 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>	

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

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

< Date of the transmission eve,

< Id of the sour,

< Id of the targ,

< Id of the varia,

< Date when the source acquired the varia,

**14.7.2.4 operator==( ) [2/3]**

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

< Date of the transmission eve,

< Id of the sour,

< Id of the targ,

< Id of the varia,

< Date when the source acquired the varia,

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

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

< Date of the transmission eve

< Id of the sour

< Id of the targ

< Id of the varia

< Date when the source acquired the varia

**14.7.2.6 record\_virus()**

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

Registering a new variant.

**Parameters**

<b>v</b>	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.
----------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**14.7.2.7 reproductive\_number()**

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

Computes the reproductive number of each case.

By definition, whereas it computes  $R_0$  (basic reproductive number) or  $R_t/R$  (the effective reproductive number) will depend on whether the virus is allowed to circulate naïvely or not, respectively.

#### Parameters

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

#### 14.7.2.8 transition\_probability()

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

Calculates the transition probabilities.

#### Returns

`std::vector< epiworld_double >`

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

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

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

Statistical data about the process.

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

### Public Member Functions

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

### Get recorded information from the model

#### Parameters

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

#### 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\_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 > [reproductive\\_number](#) () const

*Computes the reproductive number of each case.*

- void **reproductive\_number** (std::string fn) const
- void **generation\_time** (std::vector< int > &agent\_id, std::vector< int > &virus\_id, std::vector< int > &time, std::vector< int > &gentime) const
- void **generation\_time** (std::string fn) const

## Friends

- class **Model**< TSeq >
- void **default\_add\_virus** (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)

## 14.8.1 Detailed Description

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

Statistical data about the process.

### Template Parameters

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

## 14.8.2 Member Function Documentation

### 14.8.2.1 generation\_time()

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

Calculates the generating time

### Parameters

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

### 14.8.2.2 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>	

### 14.8.2.3 operator==( )

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

< Date of the transmission eve

< Id of the sour

< Id of the targ

< Id of the varia

< Date when the source acquired the varia

### 14.8.2.4 record\_virus()

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

Registering a new variant.

## Parameters

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

**14.8.2.5 reproductive\_number()**

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

Computes the reproductive number of each case.

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

## Parameters

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

**14.8.2.6 transition\_probability()**

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

Calculates the transition probabilities.

## Returns

`std::vector< epiworld_double >`

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

- epiworld.hpp

**14.9 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 >

### 14.9.1 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

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

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

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

## Public Member Functions

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



## Friends

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

### 14.10.1 Detailed Description

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

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

Template Parameters

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

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

- epiworld.hpp

## 14.11 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 >

### 14.11.1 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

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

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

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

### Public Member Functions

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

### Friends

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

### 14.12.1 Detailed Description

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

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

## Template Parameters

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

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

- epiworld.hpp

## 14.13 Entity< TSeq > Class Template Reference

### Public Member Functions

- **Entity** (std::string name)
- void **add\_agent** ([Agent](#)< TSeq > &p, [Model](#)< TSeq > \*model)
- void **add\_agent** ([Agent](#)< TSeq > \*p, [Model](#)< TSeq > \*model)
- void **rm\_agent** (size\_t idx)
- size\_t **size** () const noexcept
- void **set\_location** (std::vector< epiworld\_double > loc)
- std::vector< epiworld\_double > & **get\_location** ()
- std::vector< [Agent](#)< TSeq > \* >::iterator **begin** ()
- std::vector< [Agent](#)< TSeq > \* >::iterator **end** ()
- std::vector< [Agent](#)< TSeq > \* >::const\_iterator **begin** () const
- std::vector< [Agent](#)< TSeq > \* >::const\_iterator **end** () const
- [Agent](#)< TSeq > \* **operator[]** (size\_t i)
- int **get\_id** () const noexcept
- const std::string & **get\_name** () const noexcept
- void **set\_state** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **set\_queue** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **get\_state** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- void **get\_queue** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- void **reset** ()
- bool **operator==** (const [Entity](#)< TSeq > &other) const
- bool **operator!=** (const [Entity](#)< TSeq > &other) const

### Friends

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

### 14.13.1 Friends And Related Function Documentation

#### 14.13.1.1 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

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

### Public Member Functions

- **Entity** (std::string name)
- void **add\_agent** ([Agent](#)< TSeq > &p, [Model](#)< TSeq > \*model)
- void **add\_agent** ([Agent](#)< TSeq > \*p, [Model](#)< TSeq > \*model)
- void **rm\_agent** (size\_t idx)
- size\_t **size** () const noexcept
- void **set\_location** (std::vector< epiworld\_double > loc)
- std::vector< epiworld\_double > & **get\_location** ()
- std::vector< [Agent](#)< TSeq > \* >::iterator **begin** ()
- std::vector< [Agent](#)< TSeq > \* >::iterator **end** ()
- std::vector< [Agent](#)< TSeq > \* >::const\_iterator **begin** () const
- std::vector< [Agent](#)< TSeq > \* >::const\_iterator **end** () const
- [Agent](#)< TSeq > \* **operator[]** (size\_t i)
- int **get\_id** () const noexcept
- const std::string & **get\_name** () const noexcept
- void **set\_state** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **set\_queue** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **get\_state** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- void **get\_queue** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- void **reset** ()
- bool **operator==** (const [Entity](#)< TSeq > &other) const
- bool **operator!=** (const [Entity](#)< TSeq > &other) const

### Friends

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

### 14.14.1 Friends And Related Function Documentation

#### 14.14.1.1 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 file:

- epiworld.hpp

## 14.15 epiworld::Event< TSeq > Struct Template Reference

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

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

### 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\_, ActionFun< 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**
- ActionFun< TSeq > **call**
- int **idx\_agent**
- int **idx\_object**

### 14.15.1 Detailed Description

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

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

Template Parameters

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

### 14.15.2 Constructor & Destructor Documentation

#### 14.15.2.1 Event()

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

```

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

```

Construct a new [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 file:

- epiworld.hpp

## 14.16 [Event](#)< TSeq > Struct Template Reference

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

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

### 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\_, ActionFun< 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**
- ActionFun< TSeq > **call**
- int **idx\_agent**
- int **idx\_object**

### 14.16.1 Detailed Description

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

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

Template Parameters

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

### 14.16.2 Constructor & Destructor Documentation

#### 14.16.2.1 Event()

```
template<typename 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_,
 ActionFun< 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

## Parameters

<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

## 14.17 epiworld::GlobalEvent< TSeq > Class Template Reference

Template for a Global [Event](#).

```
#include <epiworld.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

### 14.17.1 Detailed Description

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

Template for a Global [Event](#).

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



## 14.17.2 Constructor & Destructor Documentation

### 14.17.2.1 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 Model<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 file:

- epiworld.hpp

## 14.18 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

### 14.18.1 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.

## 14.18.2 Constructor & Destructor Documentation

### 14.18.2.1 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 <code>Model&lt;TSeq&gt; *</code> 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

## 14.19 GroupSampler< TSeq > Class Template Reference

Weighted sampling of groups.

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

### Public Member Functions

- **GroupSampler** (const std::vector< double > &contact\_matrix\_, const std::vector< size\_t > &group\_sizes\_↵\_, bool normalize=true)
- int **sample\_1** ([Model](#)< TSeq > \*model, const int origin\_group)
- void **sample\_n** ([Model](#)< TSeq > \*model, std::vector< int > &sample, const int origin\_group, const int nsamples)

### 14.19.1 Detailed Description

```
template<typename TSeq>
class GroupSampler< TSeq >
```

Weighted sampling of groups.

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

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

## 14.20 epiworld::LFMCMC< TData > Class Template Reference

Likelihood-Free Markov Chain Monte Carlo.

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

### Public Member Functions

- void **run** (std::vector< epiworld\_double > param\_init, size\_t n\_samples\_, epiworld\_double epsilon\_)
- **LFMCMC** (TData &observed\_data\_)
- void **set\_observed\_data** (TData &observed\_data\_)
- void **set\_proposal\_fun** (LFMCMCProposalFun< TData > fun)
- void **set\_simulation\_fun** (LFMCMCSimFun< TData > fun)
- void **set\_summary\_fun** (LFMCMCSummaryFun< TData > fun)
- void **set\_kernel\_fun** (LFMCMCKernelFun< TData > fun)
- size\_t **get\_n\_samples** () const
- size\_t **get\_n\_statistics** () const
- size\_t **get\_n\_parameters** () const
- epiworld\_double **get\_epsilon** () const
- const std::vector< epiworld\_double > & **get\_params\_now** ()
- const std::vector< epiworld\_double > & **get\_params\_prev** ()
- const std::vector< epiworld\_double > & **get\_params\_init** ()
- const std::vector< epiworld\_double > & **get\_statistics\_obs** ()
- const std::vector< epiworld\_double > & **get\_statistics\_hist** ()
- const std::vector< bool > & **get\_statistics\_accepted** ()
- const std::vector< epiworld\_double > & **get\_posterior\_if\_prob** ()
- const std::vector< epiworld\_double > & **get\_drawn\_prob** ()
- std::vector< TData > \* **get\_sampled\_data** ()
- void **set\_par\_names** (std::vector< std::string > names)
- void **set\_stats\_names** (std::vector< std::string > names)
- std::vector< epiworld\_double > **get\_params\_mean** ()
- std::vector< epiworld\_double > **get\_stats\_mean** ()
- void **print** ()

### Random number generation

#### Parameters

eng	
-----	--

- void **set\_rand\_engine** (std::mt19937 &eng)
- std::mt19937 & **get\_rand\_engine** ()
- void **seed** (epiworld\_fast\_uint s)
- void **set\_rand\_gamma** (epiworld\_double alpha, epiworld\_double beta)
- epiworld\_double **runif** ()
- epiworld\_double **rnorm** ()
- epiworld\_double **rgamma** ()
- epiworld\_double **runif** (epiworld\_double lb, epiworld\_double ub)
- epiworld\_double **rnorm** (epiworld\_double mean, epiworld\_double sd)
- epiworld\_double **rgamma** (epiworld\_double alpha, epiworld\_double beta)

### 14.20.1 Detailed Description

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

Likelihood-Free Markov Chain Monte Carlo.

#### Template Parameters

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

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

- epiworld.hpp

## 14.21 LFMCMC< TData > Class Template Reference

Likelihood-Free Markov Chain Monte Carlo.

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

### Public Member Functions

- void **run** (std::vector< epiworld\_double > param\_init, size\_t n\_samples\_, epiworld\_double epsilon\_)
- **LFMCMC** (TData &observed\_data\_)
- void **set\_observed\_data** (TData &observed\_data\_)
- void **set\_proposal\_fun** (LFMCMCProposalFun< TData > fun)
- void **set\_simulation\_fun** (LFMCMCSimFun< TData > fun)
- void **set\_summary\_fun** (LFMCMCSummaryFun< TData > fun)
- void **set\_kernel\_fun** (LFMCMCKernelFun< TData > fun)
- size\_t **get\_n\_samples** () const
- size\_t **get\_n\_statistics** () const
- size\_t **get\_n\_parameters** () const
- epiworld\_double **get\_epsilon** () const
- const std::vector< epiworld\_double > & **get\_params\_now** ()
- const std::vector< epiworld\_double > & **get\_params\_prev** ()
- const std::vector< epiworld\_double > & **get\_params\_init** ()
- const std::vector< epiworld\_double > & **get\_statistics\_obs** ()
- const std::vector< epiworld\_double > & **get\_statistics\_hist** ()
- const std::vector< bool > & **get\_statistics\_accepted** ()
- const std::vector< epiworld\_double > & **get\_posterior\_if\_prob** ()
- const std::vector< epiworld\_double > & **get\_drawn\_prob** ()
- std::vector< TData > \* **get\_sampled\_data** ()
- void **set\_par\_names** (std::vector< std::string > names)
- void **set\_stats\_names** (std::vector< std::string > names)
- std::vector< epiworld\_double > **get\_params\_mean** ()
- std::vector< epiworld\_double > **get\_stats\_mean** ()
- void **print** ()

### Random number generation

*Parameters*

eng	
-----	--

- void **set\_rand\_engine** (std::mt19937 &eng)
- std::mt19937 & **get\_rand\_engine** ()
- void **seed** (epiworld\_fast\_uint s)
- void **set\_rand\_gamma** (epiworld\_double alpha, epiworld\_double beta)
- epiworld\_double **runif** ()
- epiworld\_double **rnorm** ()
- epiworld\_double **rgamma** ()
- epiworld\_double **runif** (epiworld\_double lb, epiworld\_double ub)
- epiworld\_double **rnorm** (epiworld\_double mean, epiworld\_double sd)
- epiworld\_double **rgamma** (epiworld\_double alpha, epiworld\_double beta)

**14.21.1 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:

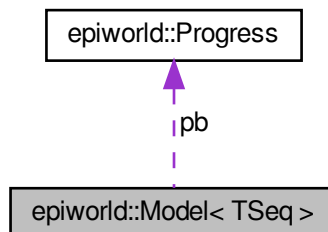
- include/epiworld/math/lfmcmc/lfmcmc-bones.hpp
- include/epiworld/math/lfmcmc/lfmcmc-meat-print.hpp
- include/epiworld/math/lfmcmc/lfmcmc-meat.hpp

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

Core class of epiworld.

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

Collaboration diagram for epiworld::Model< TSeq >:



## Public Member Functions

- [DataBase](#)< TSeq > & **get\_db** ()
- epiworld\_double & **operator()** (std::string pname)
- size\_t **size** () const
- void **load\_agents\_entities\_ties** (std::string fn, int skip)  
*Associate agents-entities from a file.*
- size\_t **get\_n\_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
- [Model](#)< TSeq > && **clone** () const
- void **get\_elapsed** (std::string unit="auto", epiworld\_double \*last\_elapsed=nullptr, epiworld\_double \*total\_↵  
 elapsed=nullptr, std::string \*unit\_abbr=nullptr, bool print=true) const
- void **add\_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< epiworld\_double > & **get\_prevalence\_virus** () const
- const std::vector< bool > & **get\_prevalence\_virus\_as\_proportion** () 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.*

### Set the backup object

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

- `void set_backup ()`

### Random number generation

#### Parameters

eng	Random number generator
s	Seed

- `void set_rand_engine (std::mt19937 &eng)`
- `std::mt19937 & get_rand_engine ()`
- `void seed (size_t s)`
- `void set_rand_norm (epiworld_double mean, epiworld_double sd)`
- `void set_rand_unif (epiworld_double a, epiworld_double b)`
- `void set_rand_exp (epiworld_double lambda)`
- `void set_rand_gamma (epiworld_double alpha, epiworld_double beta)`
- `void set_rand_lognormal (epiworld_double mean, epiworld_double shape)`
- `void set_rand_binom (int n, epiworld_double p)`
- `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)`

### 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, epiworld\_double preval)
- void **add\_virus\_n** ([Virus](#)< TSeq > &v, epiworld\_fast\_uint preval)
- void **add\_virus\_fun** ([Virus](#)< TSeq > &v, VirusToAgentFun< TSeq > fun)
- void **add\_tool** ([Tool](#)< TSeq > &t, epiworld\_double preval)
- void **add\_tool\_n** ([Tool](#)< TSeq > &t, epiworld\_fast\_uint preval)
- void **add\_tool\_fun** ([Tool](#)< TSeq > &t, ToolToAgentFun< TSeq > fun)
- void **add\_entity** ([Entity](#)< TSeq > e)
- void **rm\_virus** (size\_t virus\_pos)
- void **rm\_tool** (size\_t tool\_pos)
- void **rm\_entity** (size\_t entity\_pos)

### Accessing population of the model

#### Parameters

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

- void **agents\_from\_adjlist** (std::string fn, int size, int skip=0, bool directed=false)
- void **agents\_from\_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.*
- 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** ()
- [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 <code>run_multiple</code>, 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)
- void **read\_params** (std::string fn)
- `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)

### Set the user data object

#### Parameters

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

- void [set\\_user\\_data](#) (std::vector< std::string > names)
- [\[@\]](#)
- void **add\_user\_data** (`epiworld_fast_uint` j, `epiworld_double` x)
- void **add\_user\_data** (std::vector< `epiworld_double` > x)
- [UserData](#)< TSeq > & **get\_user\_data** ()

### Queuing system

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

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

Retrieve the [Queue](#) object.

### Get the susceptibility reduction object

#### Parameters

v	
---	--

#### Returns

*epiworld\_double*

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

## Protected Member Functions

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

Construct a new [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< epiworld\_double > **prevalence\_virus** = {}  
Initial prevalence\_virus of each virus.
- std::vector< bool > **prevalence\_virus\_as\_proportion** = {}
- std::vector< [VirusToAgentFun](#)< TSeq > > **viruses\_dist\_funs** = {}
- std::vector< [ToolPtr](#)< TSeq > > **tools** = {}
- std::vector< epiworld\_double > **prevalence\_tool** = {}
- std::vector< bool > **prevalence\_tool\_as\_proportion** = {}
- std::vector< [ToolToAgentFun](#)< TSeq > > **tools\_dist\_funs** = {}
- std::vector< [Entity](#)< TSeq > > **entities** = {}
- std::vector< [Entity](#)< TSeq > > **entities\_backup** = {}
- std::mt19937 **engine**
- std::uniform\_real\_distribution **runifd**
- std::normal\_distribution **rnormd**
- std::gamma\_distribution **rgammad**
- std::lognormal\_distribution **rlognormald**

- `std::exponential_distribution` **rexp**
- `std::binomial_distribution` **rbinom**
- `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\_queuing** = 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::shared_ptr< std::vector< Agent< TSeq > * > >` **sampled\_population**
- `size_t` **sampled\_population\_n** = 0u
- `std::shared_ptr< 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** = susceptibility\_reduction\_mixer\_default<TSeq>
  - MixerFun< TSeq > **transmission\_reduction\_mixer** = transmission\_reduction\_mixer\_default<TSeq>
  - MixerFun< TSeq > **recovery\_enhancer\_mixer** = recovery\_enhancer\_mixer\_default<TSeq>
  - MixerFun< TSeq > **death\_reduction\_mixer** = death\_reduction\_mixer\_default<TSeq>
  - std::vector< epiworld\_double > **array\_double\_tmp**
  - std::vector< [Virus](#)< TSeq > \* > **array\_virus\_tmp**
  - virtual [Model](#)< TSeq > \* **clone\_ptr** ()
- Advanced usage: Makes a copy of data and returns it as undeleted pointer.*
- **Model** ()
  - **Model** (const [Model](#)< TSeq > &m)
  - **Model** ([Model](#)< TSeq > &m)
  - **Model** ([Model](#)< TSeq > &&m)
  - [Model](#)< TSeq > & **operator=** (const [Model](#)< TSeq > &m)
  - virtual ~**Model** ()
  - void **clone\_population** (std::vector< [Agent](#)< TSeq > > &other\_population, std::vector< [Entity](#)< TSeq > > &other\_entities, [Model](#)< TSeq > \*other\_model, bool &other\_directed) const
  - void **clone\_population** (const [Model](#)< TSeq > &other\_model)

### 14.22.1 Detailed Description

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

Core class of epiworld.

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

#### Template Parameters

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

### 14.22.2 Member Function Documentation

#### 14.22.2.1 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.

**14.22.2.2 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 [ModelSIRLogit< TSeq >](#), [ModelSIRDConn< TSeq >](#), [ModelSIRConn< TSeq >](#), [ModelSEIRMixing< TSeq >](#), [ModelSEIRDConn< TSeq >](#), [ModelSEIRConn< TSeq >](#), [epiworld::epimodels::ModelSIRLogit< TSeq >](#), [epiworld::epimodels::ModelSEIRDConn< TSeq >](#), [epiworld::epimodels::ModelSIRDConn< TSeq >](#), [epiworld::epimodels::ModelSIRConn< TSeq >](#) and [epiworld::epimodels::ModelSIRConn< TSeq >](#).

**14.22.2.3 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_,
 ActionFun< 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

## Parameters

<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.

**14.22.2.4 events\_run()**

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

Executes the stored action.

## Parameters

<i>model↔ _</i>	<b>Model</b> over which it will be executed.
---------------------	----------------------------------------------

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

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

Associate agents-entities from a file.

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

## Parameters

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

**14.22.2.6 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 [ModelSIRLogit< TSeq >](#), [ModelSIRDCONN< TSeq >](#), [ModelSIRCONN< TSeq >](#), [ModelSEIRMixing< TSeq >](#), [ModelSEIRDCONN< TSeq >](#), [ModelSEIRCONN< TSeq >](#), [epiworld::epimodels::ModelSIRLogit< TSeq >](#), [epiworld::epimodels::ModelSEIRDCONN< TSeq >](#), [epiworld::epimodels::ModelSIRDCONN< TSeq >](#), [epiworld::epimodels::ModelSIRCONN< TSeq >](#).

#### 14.22.2.7 run\_multiple()

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

##### Parameters

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

#### 14.22.2.8 set\_agents\_data()

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

Set the agents data object.

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



## Parameters

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

## 14.22.2.9 set\_name()

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

Set the name object.

## Parameters

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

## 14.22.2.10 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.
<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

## 14.22.3 Member Data Documentation

### 14.22.3.1 initial\_states\_fun

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

**Initial value:**

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

Function to distribute states. Goes along with the function

### 14.22.3.2 rbinomd

```
template<typename TSeq >
std::binomial_distribution epiworld::Model< TSeq >::rbinomd [protected]
```

**Initial value:**

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

### 14.22.3.3 rexp

```
template<typename TSeq >
std::exponential_distribution epiworld::Model< TSeq >::rexp [protected]
```

**Initial value:**

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

### 14.22.3.4 rgammad

```
template<typename TSeq >
std::gamma_distribution epiworld::Model< TSeq >::rgammad [protected]
```

**Initial value:**

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

### 14.22.3.5 rlognormald

```
template<typename TSeq >
std::lognormal_distribution epiworld::Model< TSeq >::rlognormald [protected]
```

**Initial value:**

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

### 14.22.3.6 rnormd

```
template<typename TSeq >
std::normal_distribution epiworld::Model< TSeq >::rnormd [protected]
```

**Initial value:**

```
=
 std::normal_distribution<>(0.0)
```

### 14.22.3.7 runifd

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

**Initial value:**

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

### 14.22.3.8 time\_elapsed

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

**Initial value:**

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

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

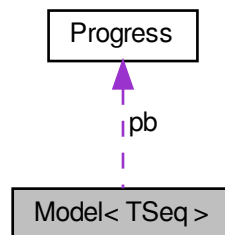
- epiworld.hpp

## 14.23 Model< TSeq > Class Template Reference

Core class of epiworld.

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

Collaboration diagram for Model< TSeq >:



### Public Member Functions

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

- **Model**< TSeq > && **clone** () const
- void **get\_elapsed** (std::string unit="auto", epiworld\_double \*last\_elapsed=nullptr, epiworld\_double \*total\_elapsed=nullptr, std::string \*unit\_abbr=nullptr, bool print=true) const
- void **add\_globlevent** (std::function< void(**Model**< TSeq > \*)> fun, std::string **name**="A global action", int date=-99)  
*Set a global action.*
- void **add\_globlevent** (**GlobalEvent**< TSeq > action)
- **GlobalEvent**< TSeq > & **get\_globlevent** (std::string **name**)  
*Retrieve a global action by name.*
- **GlobalEvent**< TSeq > & **get\_globlevent** (size\_t i)  
*Retrieve a global action by index.*
- void **rm\_globlevent** (std::string **name**)  
*Remove a global action by name.*
- void **rm\_globlevent** (size\_t i)  
*Remove a global action by index.*
- void **run\_globlevents** ()
- void **clear\_state\_set** ()
- const std::vector< **VirusPtr**< TSeq > > & **get\_viruses** () const
- const std::vector< epiworld\_double > & **get\_prevalence\_virus** () const
- const std::vector< bool > & **get\_prevalence\_virus\_as\_proportion** () 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.*

### Set the backup object

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

- void **set\_backup** ()

### Random number generation

#### Parameters

eng	Random number generator
s	Seed

- void **set\_rand\_engine** (std::mt19937 &eng)
- std::mt19937 & **get\_rand\_engine** ()
- void **seed** (size\_t s)
- void **set\_rand\_norm** (epiworld\_double mean, epiworld\_double sd)

- void **set\_rand\_unif** (epiworld\_double a, epiworld\_double b)
- void **set\_rand\_exp** (epiworld\_double lambda)
- void **set\_rand\_gamma** (epiworld\_double alpha, epiworld\_double beta)
- void **set\_rand\_lognormal** (epiworld\_double mean, epiworld\_double shape)
- void **set\_rand\_binom** (int n, epiworld\_double p)
- 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)

### 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, epiworld\_double preval)
- void **add\_virus\_n** ([Virus](#)< TSeq > &v, epiworld\_fast\_uint preval)
- void **add\_virus\_fun** ([Virus](#)< TSeq > &v, VirusToAgentFun< TSeq > fun)
- void **add\_tool** ([Tool](#)< TSeq > &t, epiworld\_double preval)
- void **add\_tool\_n** ([Tool](#)< TSeq > &t, epiworld\_fast\_uint preval)
- void **add\_tool\_fun** ([Tool](#)< TSeq > &t, ToolToAgentFun< TSeq > fun)
- void **add\_entity** ([Entity](#)< TSeq > e)
- void **add\_entity\_n** ([Entity](#)< TSeq > e, epiworld\_fast\_uint preval)
- void **add\_entity\_fun** ([Entity](#)< TSeq > e, EntityToAgentFun< TSeq > fun)
- void **rm\_virus** (size\_t virus\_pos)
- void **rm\_tool** (size\_t tool\_pos)
- void **rm\_entity** (size\_t entity\_pos)

### Accessing population of the model

#### Parameters

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

- void **agents\_from\_adjlist** (std::string fn, int size, int skip=0, bool directed=false)
- void **agents\_from\_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.*
- `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 ()`
- `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	Seed to be used for Pseudo-RNG.
ndays	Number of days (steps) of the simulation.
fun	In the case of <i>run_multiple</i> , a function that is called after each experiment.

- `void 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	<i>std::string. File name.</i>
source	<i>Integer vector</i>
target	<i>Integer vector</i>

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

- void **write\_edgelist** (std::string fn) const
- void **write\_edgelist** (std::vector< 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	<code>std::string</code> Name of the state.
-----	---------------------------------------------

#### Returns

`add_state*` returns nothing.

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

- void **add\_state** (std::string lab, UpdateFun< TSeq > fun=nullptr)
- const std::vector< std::string > & **get\_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)
- `void` **read\_params** (`std::string` fn)
- `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)

**Set the user data object****Parameters**

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

- `void` **set\_user\_data** (`std::vector< std::string >` names)
- `[@`
- `void` **add\_user\_data** (`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\_, [ActionFun](#)< 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](#)< [epiworld\\_double](#) > **prevalence\_virus** = {}  
*Initial prevalence\_virus of each virus.*
- [std::vector](#)< [bool](#) > **prevalence\_virus\_as\_proportion** = {}
- [std::vector](#)< [VirusToAgentFun](#)< TSeq > > **viruses\_dist\_funs** = {}
- [std::vector](#)< [ToolPtr](#)< TSeq > > **tools** = {}
- [std::vector](#)< [epiworld\\_double](#) > **prevalence\_tool** = {}
- [std::vector](#)< [bool](#) > **prevalence\_tool\_as\_proportion** = {}
- [std::vector](#)< [ToolToAgentFun](#)< TSeq > > **tools\_dist\_funs** = {}
- [std::vector](#)< [Entity](#)< TSeq > > **entities** = {}
- [std::vector](#)< [epiworld\\_double](#) > **prevalence\_entity** = {}
- [std::vector](#)< [bool](#) > **prevalence\_entity\_as\_proportion** = {}
- [std::vector](#)< [EntityToAgentFun](#)< TSeq > > **entities\_dist\_funs** = {}
- [std::vector](#)< [Entity](#)< TSeq > > **entities\_backup** = {}
- [std::mt19937](#) **engine**
- [std::uniform\\_real\\_distribution](#) **runifd**
- [std::normal\\_distribution](#) **rnormd**
- [std::gamma\\_distribution](#) **rgammad**
- [std::lognormal\\_distribution](#) **rlognormald**
- [std::exponential\\_distribution](#) **rexp**
- [std::binomial\\_distribution](#) **rbinomd**
- [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** = `susceptibility_reduction_mixer_default<TSeq>`
- `MixerFun< TSeq >` **transmission\_reduction\_mixer** = `transmission_reduction_mixer_default<TSeq>`
- `MixerFun< TSeq >` **recovery\_enhancer\_mixer** = `recovery_enhancer_mixer_default<TSeq>`
- `MixerFun< TSeq >` **death\_reduction\_mixer** = `death_reduction_mixer_default<TSeq>`
- `std::vector< epiworld_double >` **array\_double\_tmp**
- `std::vector< Virus< TSeq > * >` **array\_virus\_tmp**
- `std::vector< int >` **array\_int\_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` ()
- `void` **clone\_population** (`std::vector< Agent< TSeq > > &other_population`, `std::vector< Entity< TSeq > > &other_entities`, `Model< TSeq > *other_model`, `bool &other_directed`) const
- `void` **clone\_population** (const `Model< TSeq > &other_model`)

### 14.23.1 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.)
-------------	-----------------------------------------------------------------------------------------------------------------------------------------

### 14.23.2 Member Function Documentation

#### 14.23.2.1 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.

#### 14.23.2.2 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>	
-------------	--

### 14.23.2.3 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_,
 ActionFun< 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.

### 14.23.2.4 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.
----------------------	-------------------------------------------------------

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

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

Associate agents-entities from a file.

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

#### Parameters

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

### 14.23.2.6 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

### 14.23.2.7 run\_multiple()

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

## Parameters

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

**14.23.2.8 set\_agents\_data()**

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

Set the agents data object.

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

## Parameters

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

**14.23.2.9 set\_name()**

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

Set the name object.

## Parameters

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

**14.23.2.10 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.
<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

## 14.23.3 Member Data Documentation

### 14.23.3.1 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

### 14.23.3.2 rbinomd

```

template<typename TSeq >
std::binomial_distribution Model< TSeq >::rbinomd [protected]

```

#### Initial value:

```

=
 std::binomial_distribution<>()

```

### 14.23.3.3 rexp

```

template<typename TSeq >
std::exponential_distribution Model< TSeq >::rexp [protected]

```

#### Initial value:

```

=
 std::exponential_distribution<>()

```



#### 14.23.3.4 rgammad

```
template<typename TSeq >
std::gamma_distribution Model< TSeq >::rgammad [protected]
```

**Initial value:**

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

#### 14.23.3.5 rlognormald

```
template<typename TSeq >
std::lognormal_distribution Model< TSeq >::rlognormald [protected]
```

**Initial value:**

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

#### 14.23.3.6 rnormd

```
template<typename TSeq >
std::normal_distribution Model< TSeq >::rnormd [protected]
```

**Initial value:**

```
=
 std::normal_distribution<>(0.0)
```

#### 14.23.3.7 runifd

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

**Initial value:**

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

#### 14.23.3.8 time\_elapsed

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

**Initial value:**

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

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

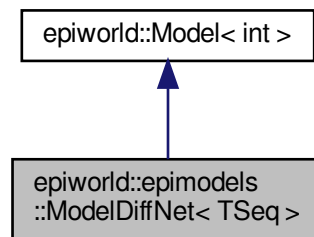
- include/epiworld/agent-bones.hpp
- include/epiworld/model-bones.hpp
- include/epiworld/model-meat-print.hpp
- include/epiworld/model-meat.hpp

## 14.24 epiworld::epimodels::ModelDiffNet< TSeq > Class Template Reference

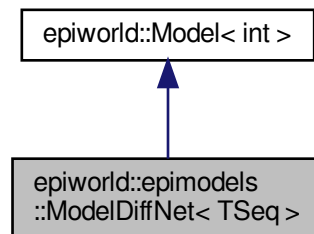
Template for a [Network](#) Diffusion [Model](#).

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

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



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



### Public Member Functions

- **ModelDiffNet** ([ModelDiffNet](#)< TSeq > &model, 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** (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

## Additional Inherited Members

### 14.24.1 Detailed Description

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

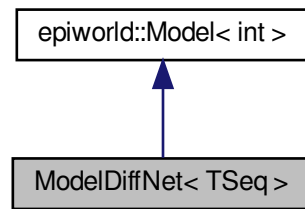
- epiworld.hpp

## 14.25 ModelDiffNet< TSeq > Class Template Reference

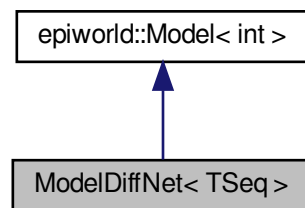
Template for a [Network](#) Diffusion [Model](#).

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

Inheritance diagram for ModelDiffNet< TSeq >:



Collaboration diagram for ModelDiffNet< TSeq >:



## Public Member Functions

- **ModelDiffNet** ([ModelDiffNet](#)< TSeq > &model, std::string innovation\_name, epiworld\_double prevalence, epiworld\_double probb\_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** (std::string innovation\_name, epiworld\_double prevalence, epiworld\_double probb\_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

## Additional Inherited Members

### 14.25.1 Detailed Description

```
template<typename TSeq = int>
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:

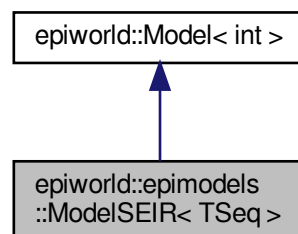
- include/epiworld/models/diffnet.hpp

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

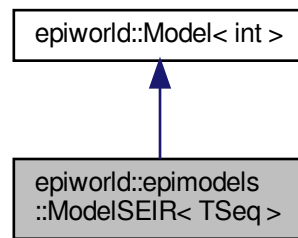
Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.

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

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



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



## Public Member Functions

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

## Additional Inherited Members

### 14.26.1 Detailed Description

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

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

## Parameters

<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>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.

## 14.26.2 Member Function Documentation

### 14.26.2.1 initial\_states()

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

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>

Reimplemented from [epiworld::Model< int >](#).

## 14.26.3 Member Data Documentation

### 14.26.3.1 update\_exposed\_seir

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

## Initial value:

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

### 14.26.3.2 update\_infected\_seir

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

#### Initial value:

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

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

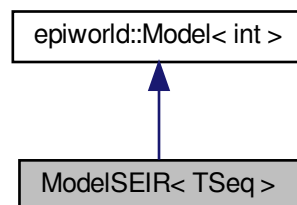
- epiworld.hpp

## 14.27 ModelSEIR< TSeq > Class Template Reference

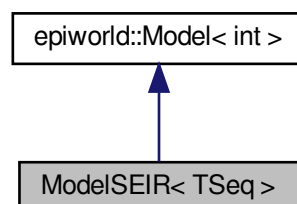
Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.

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

Inheritance diagram for ModelSEIR< TSeq >:



Collaboration diagram for ModelSEIR< TSeq >:





## Public Member Functions

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

## Additional Inherited Members

### 14.27.1 Detailed Description

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

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

#### Parameters

<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>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.

### 14.27.2 Member Function Documentation

#### 14.27.2.1 initial\_states()

```
template<typename TSeq >
ModelSEIR< TSeq > & ModelSEIR< TSeq >::initial_states (
```

```
std::vector< double > proportions_,
std::vector< int > queue_ = {}) [inline], [virtual]
```

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>

Reimplemented from [epiworld::Model< int >](#).

## 14.27.3 Member Data Documentation

### 14.27.3.1 update\_exposed\_seir

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

#### Initial value:

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

### 14.27.3.2 update\_infected\_seir

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

#### Initial value:

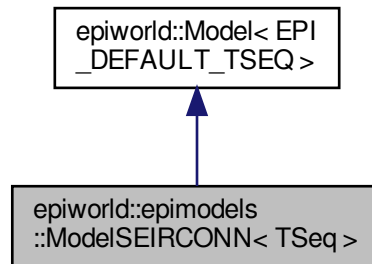
```
= [] (
 epiworld::Agent<TSeq> * p,
 epiworld::Model<TSeq> * m
) -> void {
 if (m->runif() < (m->par("Recovery rate")))
 p->rm_virus(m);
 return;
}
```

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

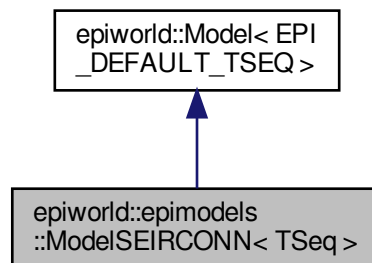
- include/epiworld/models/seir.hpp

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

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



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



### Public Member Functions

- [ModelSEIRCONN](#) ([ModelSEIRCONN](#)< TSeq > &model, std::string vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate)  
*Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.*
- **ModelSEIRCONN** (std::string vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate)
- [ModelSEIRCONN](#)< TSeq > & [run](#) (epiworld\_fast\_uint ndays, int seed=-1)  
*Runs the simulation (after initialization)*
- void [reset](#) ()  
*Reset the model.*

- `Model< TSeq > * clone_ptr ()`  
*Advanced usage: Makes a copy of data and returns it as undeleted pointer.*
- `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`

## Static Public Attributes

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

## Additional Inherited Members

### 14.28.1 Constructor & Destructor Documentation

#### 14.28.1.1 ModelSEIRCONN()

```
template<typename TSeq >
ModelSEIRCONN< TSeq >::ModelSEIRCONN (
 ModelSEIRCONN< TSeq > & model,
 std::string vname,
 epiworld_fast_uint n,
 epiworld_double prevalence,
 epiworld_double contact_rate,
 epiworld_double 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

### 14.28.2 Member Function Documentation

### 14.28.2.1 `clone_ptr()`

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

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

#### Parameters

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

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

### 14.28.2.2 `initial_states()`

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

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>

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

### 14.28.2.3 `reset()`

```
template<typename TSeq >
void ModelSEIRCONN< 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

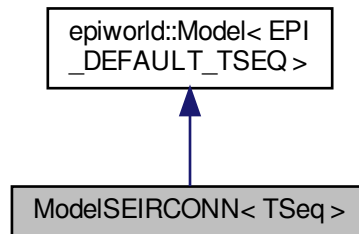
Reimplemented from [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

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

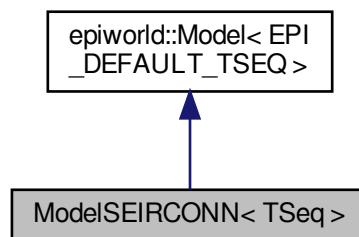
- `epiworld.hpp`

## 14.29 ModelSEIRCONN< TSeq > Class Template Reference

Inheritance diagram for ModelSEIRCONN< TSeq >:



Collaboration diagram for ModelSEIRCONN< TSeq >:



### Public Member Functions

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

## Static Public Attributes

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

## Additional Inherited Members

### 14.29.1 Constructor & Destructor Documentation

#### 14.29.1.1 ModelSEIRCONN()

```
template<typename TSeq >
ModelSEIRCONN< TSeq >::ModelSEIRCONN (
 ModelSEIRCONN< TSeq > & model,
 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

### 14.29.2 Member Function Documentation

#### 14.29.2.1 clone\_ptr()

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

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

## Parameters

copy	
------	--

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

**14.29.2.2 initial\_states()**

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

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>

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

**14.29.2.3 reset()**

```
template<typename TSeq >
void ModelSEIRCONN< 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 [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

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

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

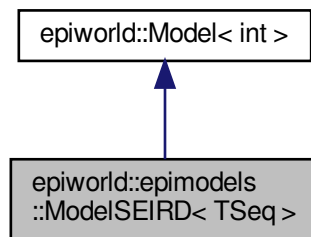


## 14.30 epiworld::epimodels::ModelSEIRD< TSeq > Class Template Reference

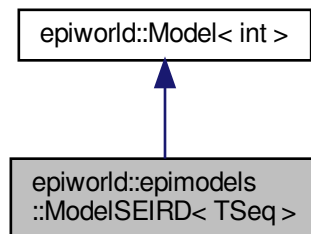
Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model.

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

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



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



### Public Member Functions

- [ModelSEIRD](#) ([ModelSEIRD](#)< TSeq > &model, std::string vname, 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](#) (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

## Additional Inherited Members

### 14.30.1 Detailed Description

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

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

### 14.30.2 Constructor & Destructor Documentation

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

```
template<typename TSeq >
ModelSEIRD< TSeq >::ModelSEIRD (
 ModelSEIRD< TSeq > & model,
 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.

## Parameters

<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.

## 14.30.2.2 ModelSEIRD() [2/2]

```
template<typename TSeq >
ModelSEIRD< TSeq >::ModelSEIRD (
 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.

## 14.30.3 Member Data Documentation

## 14.30.3.1 update\_exposed\_seir

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

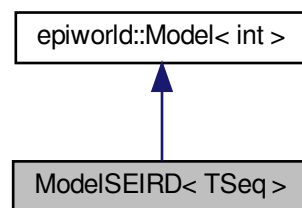
- `epiworld.hpp`

### 14.31 `ModelSEIRD< TSeq >` Class Template Reference

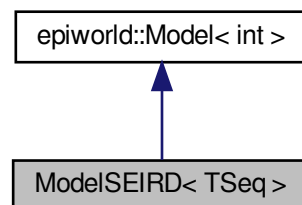
Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model.

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

Inheritance diagram for `ModelSEIRD< TSeq >`:



Collaboration diagram for `ModelSEIRD< TSeq >`:



#### Public Member Functions

- `ModelSEIRD` (`ModelSEIRD< TSeq > &model`, `std::string vname`, `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` (`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

## Additional Inherited Members

### 14.31.1 Detailed Description

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

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

### 14.31.2 Constructor & Destructor Documentation

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

```
template<typename TSeq >
ModelSEIRD< TSeq >::ModelSEIRD (
 ModelSEIRD< TSeq > & model,
 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.

## Parameters

<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.

**14.31.2.2 ModelSEIRD()** [2/2]

```
template<typename TSeq >
ModelSEIRD< TSeq >::ModelSEIRD (
 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.

**14.31.3 Member Data Documentation****14.31.3.1 update\_exposed\_seir**

```
template<typename TSeq = int>
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:

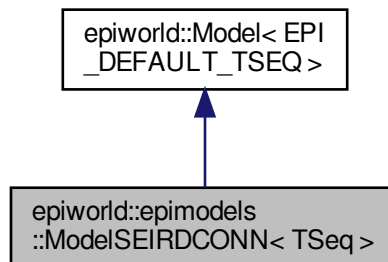
- include/epiworld/models/seird.hpp

## 14.32 epiworld::epimodels::ModelSEIRDCONN< TSeq > Class Template Reference

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



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



### Public Member Functions

- [ModelSEIRDCONN](#) ([ModelSEIRDCONN](#)< TSeq > &model, 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** (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)

- Runs the simulation (after initialization)*
- void `reset ()`  
*Reset the model.*
- `Model< TSeq > * clone_ptr ()`  
*Advanced usage: Makes a copy of data and returns it as undeleted pointer.*
- `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

## Additional Inherited Members

### 14.32.1 Constructor & Destructor Documentation

#### 14.32.1.1 ModelSEIRDCONN()

```
template<typename TSeq >
ModelSEIRDCONN< TSeq >::ModelSEIRDCONN (
 ModelSEIRDCONN< TSeq > & model,
 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



## 14.32.2 Member Function Documentation

### 14.32.2.1 `clone_ptr()`

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

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

#### Parameters

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

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

### 14.32.2.2 `initial_states()`

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

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>

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

### 14.32.2.3 `reset()`

```
template<typename TSeq >
void ModelSEIRDCONN< 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

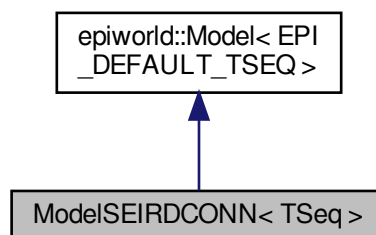
Reimplemented from [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

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

- `epiworld.hpp`

### 14.33 ModelSEIRDCONN< TSeq > Class Template Reference

Inheritance diagram for ModelSEIRDCONN< TSeq >:



Collaboration diagram for ModelSEIRDCONN< TSeq >:



## Public Member Functions

- [ModelSEIRDCONN](#) ([ModelSEIRDCONN](#)< TSeq > &model, 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](#) (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)  
*Runs the simulation (after initialization)*
- void [reset](#) ()  
*Reset the model.*
- [Model](#)< TSeq > \* [clone\\_ptr](#) ()  
*Advanced usage: Makes a copy of data and returns it as undeleted pointer.*
- [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

## Additional Inherited Members

### 14.33.1 Constructor & Destructor Documentation

#### 14.33.1.1 ModelSEIRDCONN()

```
template<typename TSeq >
ModelSEIRDCONN< TSeq >::ModelSEIRDCONN (
 ModelSEIRDCONN< TSeq > & model,
 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

## 14.33.2 Member Function Documentation

### 14.33.2.1 clone\_ptr()

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

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

## Parameters

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

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

### 14.33.2.2 initial\_states()

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

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>

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

### 14.33.2.3 reset()

```
template<typename TSeq >
void ModelSEIRDConn< 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

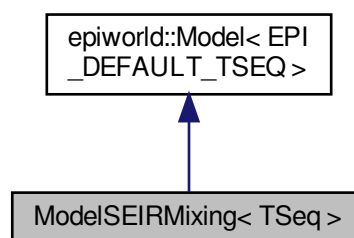
Reimplemented from [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

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

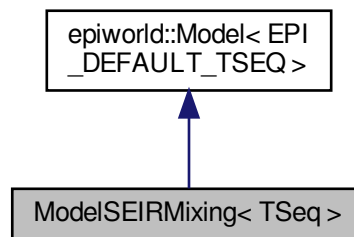
- `include/epiworld/models/seirdconnected.hpp`

## 14.34 ModelSEIRMixing< TSeq > Class Template Reference

Inheritance diagram for ModelSEIRMixing< TSeq >:



Collaboration diagram for ModelSEIRMixing< TSeq >:



## Public Member Functions

- [ModelSEIRMixing](#) ([ModelSEIRMixing](#)< TSeq > &model, 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](#) (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)  
*Runs the simulation (after initialization)*
- void [reset](#) ()  
*Reset the model.*
- [Model](#)< TSeq > \* [clone\\_ptr](#) ()  
*Advanced usage: Makes a copy of data and returns it as undeleted pointer.*
- [ModelSEIRMixing](#)< 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 **EXPOSED** = 1
- static const int **INFECTED** = 2
- static const int **RECOVERED** = 3

## Additional Inherited Members

### 14.34.1 Constructor & Destructor Documentation

**14.34.1.1 ModelSEIRMixing() [1/2]**

```
template<typename TSeq >
ModelSEIRMixing< TSeq >::ModelSEIRMixing (
 ModelSEIRMixing< TSeq > & model,
 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.
<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

**14.34.1.2 ModelSEIRMixing() [2/2]**

```
template<typename TSeq >
ModelSEIRMixing< TSeq >::ModelSEIRMixing (
 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.

## 14.34.2 Member Function Documentation

### 14.34.2.1 clone\_ptr()

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

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

## Parameters

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

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

### 14.34.2.2 initial\_states()

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

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>

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



**14.34.2.3 reset()**

```
template<typename TSeq >
void ModelSEIRMixing< 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 [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

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

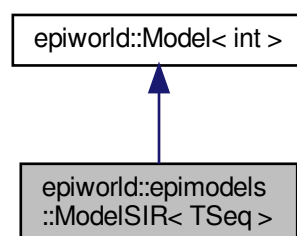
- `include/epiworld/models/seirmixing.hpp`

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

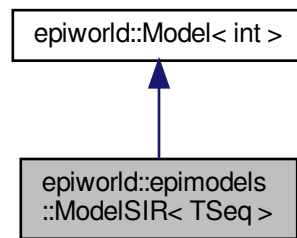
Template for a Susceptible-Infected-Removed (SIR) model.

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

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



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



## Public Member Functions

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

## Additional Inherited Members

### 14.35.1 Detailed Description

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

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

#### Parameters

<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>initial_prevalence</i>	epiworld_double Initial prevalence
<i>initial_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

### 14.35.2 Member Function Documentation

## 14.35.2.1 initial\_states()

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

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>

Reimplemented from [epiworld::Model< int >](#).

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

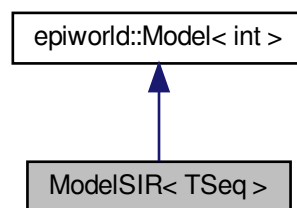
- [epiworld.hpp](#)

## 14.36 ModelSIR&lt; TSeq &gt; Class Template Reference

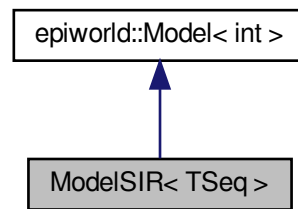
Template for a Susceptible-Infected-Removed (SIR) model.

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

Inheritance diagram for ModelSIR< TSeq >:



Collaboration diagram for ModelSIR< TSeq >:



## Public Member Functions

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

## Additional Inherited Members

### 14.36.1 Detailed Description

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

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

#### Parameters

<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>initial_prevalence</i>	epiworld_double Initial prevalence
<i>initial_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

### 14.36.2 Member Function Documentation

### 14.36.2.1 initial\_states()

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

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>

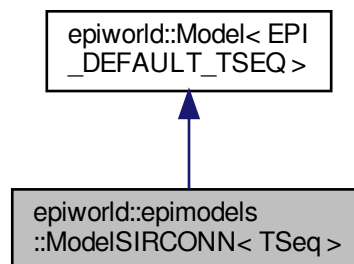
Reimplemented from [epiworld::Model< int >](#).

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

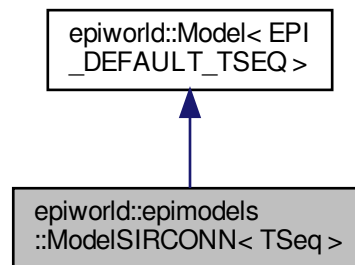
- include/epiworld/models/sir.hpp

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

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



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



## Public Member Functions

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

## Static Public Attributes

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

## Additional Inherited Members

### 14.37.1 Constructor & Destructor Documentation

### 14.37.1.1 `ModelSIRCONN()`

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

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

#### Parameters

<i>model</i>	A <code>Model&lt;TSeq&gt;</code> object where to set up the SIR.
<i>vname</i>	<code>std::string</code> 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

## 14.37.2 Member Function Documentation

### 14.37.2.1 `clone_ptr()`

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

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

#### Parameters

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

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

### 14.37.2.2 `get_n_infected()`

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

Get the infected individuals.

**Returns**

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

**14.37.2.3 initial\_states()**

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

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>

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

**14.37.2.4 reset()**

```
template<typename TSeq >
void ModelSIRCONN< 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 [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

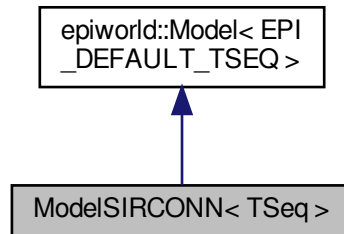
The documentation for this class was generated from the following file:

- `epiworld.hpp`

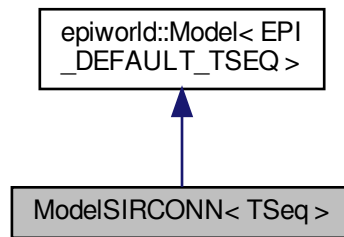


## 14.38 ModelSIRCONN< TSeq > Class Template Reference

Inheritance diagram for ModelSIRCONN< TSeq >:



Collaboration diagram for ModelSIRCONN< TSeq >:



### Public Member Functions

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

## Static Public Attributes

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

## Additional Inherited Members

### 14.38.1 Constructor & Destructor Documentation

#### 14.38.1.1 ModelSIRCONN()

```
template<typename TSeq >
ModelSIRCONN< TSeq >::ModelSIRCONN (
 ModelSIRCONN< TSeq > & model,
 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

### 14.38.2 Member Function Documentation

#### 14.38.2.1 clone\_ptr()

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

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

## Parameters

copy	
------	--

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

**14.38.2.2 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> * >
```

**14.38.2.3 initial\_states()**

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

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>

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

**14.38.2.4 reset()**

```
template<typename TSeq >
void ModelSIRCONN< 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 [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

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

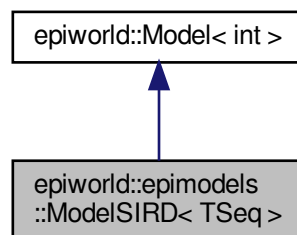
- `include/epiworld/models/sirconnected.hpp`

## 14.39 epiworld::epimodels::ModelSIRD< TSeq > Class Template Reference

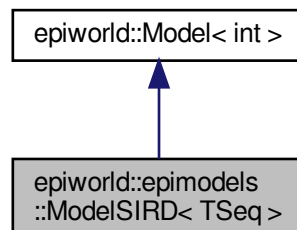
Template for a Susceptible-Infected-Removed-Deceased (SIRD) model.

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

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



Collaboration diagram for `epiworld::epimodels::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, 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** (std::string vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double death\_rate)

## Additional Inherited Members

### 14.39.1 Detailed Description

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

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

### 14.39.2 Constructor & Destructor Documentation

#### 14.39.2.1 ModelSIRD()

```
template<typename TSeq >
ModelSIRD< TSeq >::ModelSIRD (
 ModelSIRD< TSeq > & model,
 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.

### 14.39.3 Member Function Documentation

#### 14.39.3.1 initial\_states()

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

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>

Reimplemented from [epiworld::Model< int >](#).

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

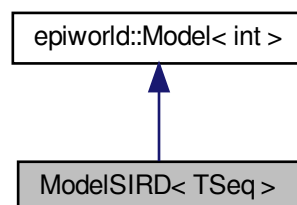
- [epiworld.hpp](#)

## 14.40 ModelSIRD< TSeq > Class Template Reference

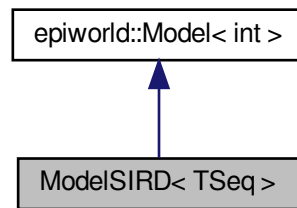
Template for a Susceptible-Infected-Removed-Deceased (SIRD) model.

```
#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`, `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` (`std::string vname`, `epiworld_double prevalence`, `epiworld_double transmission_rate`, `epiworld_double recovery_rate`, `epiworld_double death_rate`)

## Additional Inherited Members

### 14.40.1 Detailed Description

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

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

### 14.40.2 Constructor & Destructor Documentation

#### 14.40.2.1 ModelSIRD()

```
template<typename TSeq >
ModelSIRD< TSeq >::ModelSIRD (
 ModelSIRD< TSeq > & model,
 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.

### 14.40.3 Member Function Documentation

#### 14.40.3.1 `initial_states()`

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

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>

Reimplemented from [epiworld::Model< int >](#).

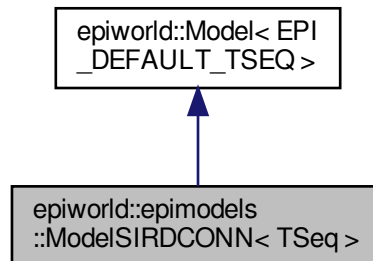
The documentation for this class was generated from the following file:

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

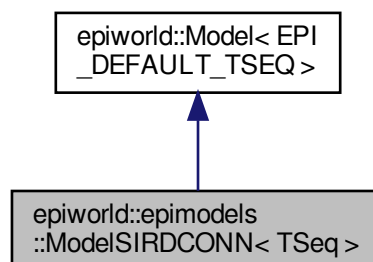


## 14.41 epiworld::epimodels::ModelSIRDCONN< TSeq > Class Template Reference

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



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



### Public Member Functions

- [ModelSIRDCONN](#) ([ModelSIRDCONN](#)< TSeq > &model, 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** (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)  
*Runs the simulation (after initialization)*
- void [reset](#) ()  
*Reset the model.*
- [Model](#)< TSeq > \* [clone\\_ptr](#) ()  
*Advanced usage: Makes a copy of data and returns it as undeleted pointer.*

## Static Public Attributes

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

## Additional Inherited Members

### 14.41.1 Constructor & Destructor Documentation

#### 14.41.1.1 ModelSIRDCONN()

```
template<typename TSeq >
ModelSIRDCONN< TSeq >::ModelSIRDCONN (
 ModelSIRDCONN< TSeq > & model,
 std::string vname,
 epiworld_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

### 14.41.2 Member Function Documentation

#### 14.41.2.1 clone\_ptr()

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

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

## Parameters

copy	
------	--

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

### 14.41.2.2 reset()

```
template<typename TSeq >
void ModelSIRDConn< 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

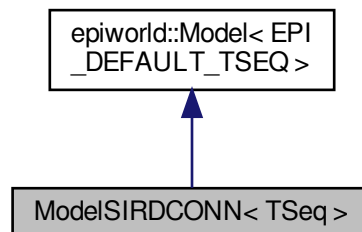
Reimplemented from [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

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

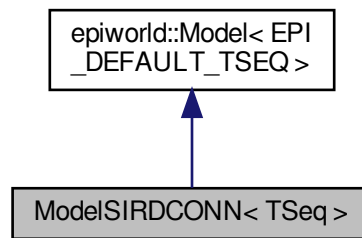
- `epiworld.hpp`

## 14.42 ModelSIRDConn< TSeq > Class Template Reference

Inheritance diagram for ModelSIRDConn< TSeq >:



Collaboration diagram for ModelSIRDCONN< TSeq >:



## Public Member Functions

- [ModelSIRDCONN](#) ([ModelSIRDCONN](#)< TSeq > &model, 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** (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)  
*Runs the simulation (after initialization)*
- [ModelSIRDCONN](#)< TSeq > & [run](#) (epiworld\_fast\_uint ndays, int seed=-1)  
*Reset the model.*
- void [reset](#) ()  
*Advanced usage: Makes a copy of data and returns it as undeleted pointer.*
- [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

## Additional Inherited Members

### 14.42.1 Constructor & Destructor Documentation

### 14.42.1.1 ModelSIRDCONN()

```
template<typename TSeq >
ModelSIRDCONN< TSeq >::ModelSIRDCONN (
 ModelSIRDCONN< TSeq > & model,
 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

## 14.42.2 Member Function Documentation

### 14.42.2.1 clone\_ptr()

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

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

#### Parameters

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

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

### 14.42.2.2 reset()

```
template<typename TSeq >
void ModelSIRDCONN< 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 [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

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

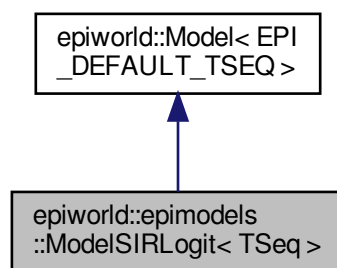
- `include/epiworld/models/sirdconnected.hpp`

## 14.43 epiworld::epimodels::ModelSIRLogit< TSeq > Class Template Reference

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

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

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



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



## Public Member Functions

- [ModelSIRLogit](#) ([ModelSIRLogit](#)< TSeq > &model, 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** (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)  
*Runs the simulation (after initialization)*
- [Model](#)< TSeq > \* [clone\\_ptr](#) ()  
*Advanced usage: Makes a copy of data and returns it as undeleted pointer.*
- void [reset](#) ()  
*Reset the model.*

## 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**

## Additional Inherited Members

### 14.43.1 Detailed Description

```
template<typename TSeq = EPI_DEFAULT_TSEQ>
class epiworld::epimodels::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. <code>std::vector</code> , <code>std::deque</code> )
-------------	---------------------------------------------------------------------------------

## 14.43.2 Constructor & Destructor Documentation

### 14.43.2.1 ModelSIRLogit()

```
template<typename TSeq >
ModelSIRLogit< TSeq >::ModelSIRLogit (
 ModelSIRLogit< TSeq > & model,
 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>ncof_infect</i>	Unsigned int. Number of infection coefficients.
<i>ncof_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.



## Parameters

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

### 14.43.3 Member Function Documentation

#### 14.43.3.1 clone\_ptr()

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

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

## Parameters

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

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

#### 14.43.3.2 reset()

```
template<typename TSeq >
void ModelSIRLogit< 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 [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

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

- epiworld.hpp

## 14.44 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, 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** (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)  
*Runs the simulation (after initialization)*
- [Model](#)< TSeq > \* [clone\\_ptr](#) ()  
*Advanced usage: Makes a copy of data and returns it as undeleted pointer.*
- void [reset](#) ()  
*Reset the model.*

## 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`

## Additional Inherited Members

### 14.44.1 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. <code>std::vector</code> , <code>std::deque</code> )
-------------	---------------------------------------------------------------------------------

### 14.44.2 Constructor & Destructor Documentation

#### 14.44.2.1 ModelSIRLogit()

```
template<typename TSeq >
ModelSIRLogit< TSeq >::ModelSIRLogit (
 ModelSIRLogit< TSeq > & model,
 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>nccoef_infect</i>	Unsigned int. Number of infection coefficients.
<i>nccoef_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

## 14.44.3 Member Function Documentation

### 14.44.3.1 clone\_ptr()

```

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

```

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

#### Parameters

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

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

### 14.44.3.2 reset()

```

template<typename TSeq >
void ModelSIRLogit< 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 [epiworld::Model< EPI\\_DEFAULT\\_TSEQ >](#).

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

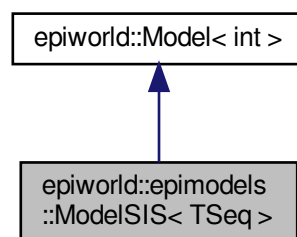
- `include/epiworld/models/sirlogit.hpp`

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

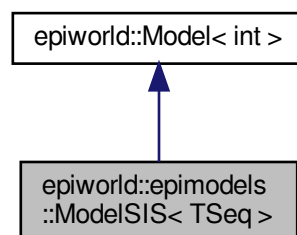
Template for a Susceptible-Infected-Susceptible (SIS) model.

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

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



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



## Public Member Functions

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

## Additional Inherited Members

### 14.45.1 Detailed Description

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

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

#### Parameters

<i>vname</i>	std::string Name of the virus
<i>initial_prevalence</i>	epiworld_double Initial prevalence
<i>initial_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

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

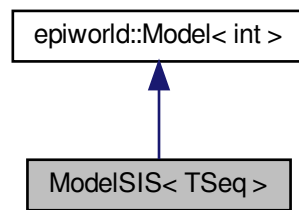
- epiworld.hpp

## 14.46 ModelSIS< TSeq > Class Template Reference

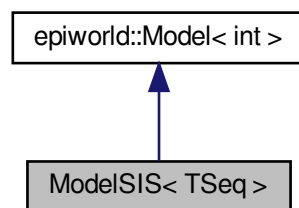
Template for a Susceptible-Infected-Susceptible (SIS) model.

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

Inheritance diagram for ModelSIS< TSeq >:



Collaboration diagram for ModelSIS< TSeq >:



## Public Member Functions

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

## Additional Inherited Members

### 14.46.1 Detailed Description

```

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

```

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

## Parameters

<i>vname</i>	std::string Name of the virus
<i>initial_prevalence</i>	epiworld_double Initial prevalence
<i>initial_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:

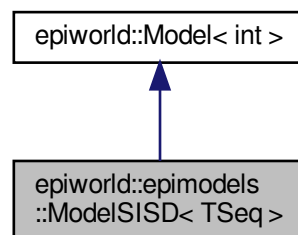
- include/epiworld/models/sis.hpp

## 14.47 epiworld::epimodels::ModelSISD< TSeq > Class Template Reference

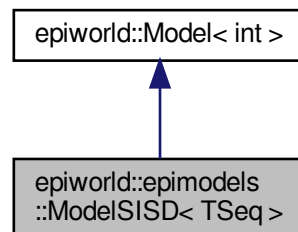
Template for a Susceptible-Infected-Susceptible-Deceased (SISD) model.

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

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



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





## Public Member Functions

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

## Additional Inherited Members

### 14.47.1 Detailed Description

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

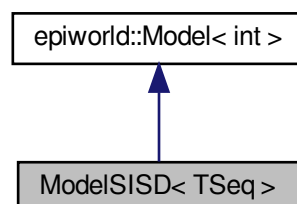
- epiworld.hpp

## 14.48 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, std::string vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double death\_rate)
- **ModelSISD** (std::string vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double death\_rate)

## Additional Inherited Members

### 14.48.1 Detailed Description

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

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

#### Parameters

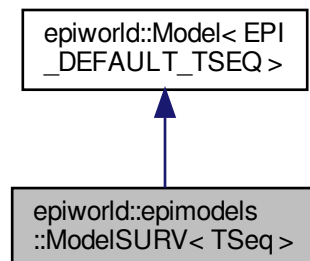
<i>vname</i>	std::string Name of the virus
<i>initial_prevalence</i>	epiworld_double Initial prevalence
<i>initial_efficiency</i>	epiworld_double Initial susceptibility_reduction of the immune system
<i>initial_recovery</i>	epiworld_double Initial recovery_rate rate of the immune system
<i>initial_death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

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

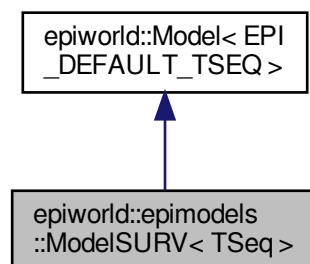
- include/epiworld/models/sisd.hpp

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

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



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



### Public Member Functions

#### Construct a new ModelSURV object

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

#### Parameters

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

**Parameters**

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

*This model features the following states:*

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

**Returns**

*An object of class `epiworld_surv`*

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

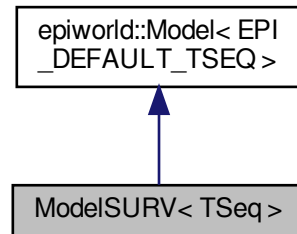
**Additional Inherited Members**

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

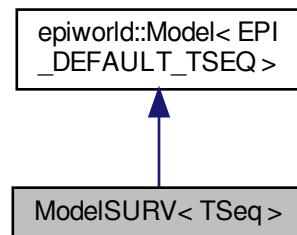
- `epiworld.hpp`

## 14.50 ModelSURV< TSeq > Class Template Reference

Inheritance diagram for ModelSURV< TSeq >:



Collaboration diagram for ModelSURV< TSeq >:



### Public Member Functions

#### Construct a new ModelSURV object

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

#### Parameters

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

### Parameters

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

*This model features the following states:*

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

### Returns

*An object of class `epiworld_surv`*

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

## Additional Inherited Members

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

- include/epiworld/models/surveillance.hpp

## 14.51 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

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

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

- epiworld.hpp

## 14.53 PersonTools< TSeq > Class Template Reference

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

- include/epiworld/config.hpp

## 14.54 epiworld::Progress Class Reference

A simple progress bar.

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

### Public Member Functions

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

### 14.54.1 Detailed Description

A simple progress bar.

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

- epiworld.hpp

## 14.55 Progress Class Reference

A simple progress bar.

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

### Public Member Functions

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

### 14.55.1 Detailed Description

A simple progress bar.

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

- include/epiworld/progress.hpp

## 14.56 epiworld::Queue< TSeq > Class Template Reference

Controls which agents are verified at each step.

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

### Public Member Functions

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



## Static Public Attributes

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

## Friends

- class **Model**< TSeq >

### 14.56.1 Detailed Description

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

Controls which agents are verified at each step.

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

#### Template Parameters

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

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

- epiworld.hpp

## 14.57 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 >

### 14.57.1 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

## 14.58 RandGraph Class Reference

### Public Member Functions

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

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

- include/epiworld/random\_graph.hpp

## 14.59 epiworld::SAMPLETYPE Class Reference

### Static Public Attributes

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

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

- epiworld.hpp

## 14.60 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

## 14.61 epiworld::Tool< TSeq > Class Template Reference

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

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

### Public Member Functions

- **Tool** (std::string name="unknown tool")
- void **set\_sequence** (TSeq d)
- void **set\_sequence** (std::shared\_ptr< TSeq > d)
- std::shared\_ptr< TSeq > **get\_sequence** ()
- void **set\_name** (std::string name)
- std::string **get\_name** () const
- [Agent](#)< TSeq > \* **get\_agent** ()
- int **get\_id** () const
- void **set\_id** (int id)
- void **set\_date** (int d)
- int **get\_date** () const
- void **set\_state** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **set\_queue** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **get\_state** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- void **get\_queue** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- bool **operator==** (const [Tool](#)< TSeq > &other) const
- bool **operator!=** (const [Tool](#)< TSeq > &other) const
- void **print** () const

### Get and set the tool functions

#### Parameters

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

### Returns

*epiworld\_double*

- *epiworld\_double* **get\_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)

## 14.61.1 Detailed Description

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

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

#### Template Parameters

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

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

- epiworld.hpp

## 14.62 Tool< TSeq > Class Template Reference

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

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

## Public Member Functions

- **Tool** (std::string name="unknown tool")
- void **set\_sequence** (TSeq d)
- void **set\_sequence** (std::shared\_ptr< TSeq > d)
- std::shared\_ptr< TSeq > **get\_sequence** ()
- void **set\_name** (std::string name)
- std::string **get\_name** () const
- [Agent](#)< TSeq > \* **get\_agent** ()
- int **get\_id** () const
- void **set\_id** (int id)
- void **set\_date** (int d)
- int **get\_date** () const
- void **set\_state** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **set\_queue** (epiworld\_fast\_int init, epiworld\_fast\_int post)
- void **get\_state** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- void **get\_queue** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*post)
- bool **operator==** (const [Tool](#)< TSeq > &other) const
- bool **operator!=** (const [Tool](#)< TSeq > &other) const
- void **print** () const
- bool **operator==** (const [Tool](#)< std::vector< int >> &other) const
- bool **operator==** (const [Tool](#)< std::vector< int >> &other) const

### Get and set the tool functions

#### Parameters

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

#### Returns

*epiworld\_double*

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

## Friends

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

### 14.62.1 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

## 14.63 epiworld::Tools< TSeq > Class Template Reference

Set of tools (useful for building iterators)

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

### Public Member Functions

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

### Friends

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

### 14.63.1 Detailed Description

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

Set of tools (useful for building iterators)

## Template Parameters

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

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

- epiworld.hpp

## 14.64 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 >

#### 14.64.1 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

## 14.65 epiworld::Tools\_const< TSeq > Class Template Reference

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

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

### Public Member Functions

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

### Friends

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

### 14.65.1 Detailed Description

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

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

#### Template Parameters

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

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

- epiworld.hpp

## 14.66 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 >

### 14.66.1 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

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

Personalized data by the user.

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

## Public Member Functions

- **UserData** ([Model](#)< TSeq > &m)
- **UserData** ([Model](#)< TSeq > \*m)
- **UserData** (std::vector< std::string > names)  
Construct a new User Data object.
- std::vector< std::string > & **get\_names** ()
- std::vector< int > & **get\_dates** ()
- std::vector< epiworld\_double > & **get\_data** ()

- void **get\_all** (std::vector< std::string > \*names=nullptr, std::vector< int > \*date=nullptr, std::vector< epiworld\_double > \*data=nullptr)
- epiworld\_fast\_uint **nrow** () const
- epiworld\_fast\_uint **ncol** () const
- void **write** (std::string fn)
- void **print** () const

### Append data

#### Parameters

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

### 14.67.1 Detailed Description

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

Personalized data by the user.

#### Template Parameters

TSeq	
------	--

## 14.67.2 Constructor & Destructor Documentation

### 14.67.2.1 UserData()

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

Construct a new User Data object.

#### Parameters

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

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

- epiworld.hpp

## 14.68 UserData< TSeq > Class Template Reference

Personalized data by the user.

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

### Public Member Functions

- **UserData** ([Model](#)< TSeq > &m)
- **UserData** ([Model](#)< TSeq > \*m)
- **UserData** (std::vector< std::string > names)  
*Construct a new User Data object.*
- std::vector< std::string > & **get\_names** ()
- std::vector< int > & **get\_dates** ()
- std::vector< epiworld\_double > & **get\_data** ()
- void **get\_all** (std::vector< std::string > \*names=nullptr, std::vector< int > \*date=nullptr, std::vector< epiworld\_double > \*data=nullptr)
- epiworld\_fast\_uint **nrow** () const
- epiworld\_fast\_uint **ncol** () const
- void **write** (std::string fn)
- void **print** () const

### Append data

#### Parameters

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

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

### Access data

#### Parameters

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

#### Returns

*epiworld\_double*&

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

### Friends

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

## 14.68.1 Detailed Description

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

Personalized data by the user.

#### Template Parameters

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

## 14.68.2 Constructor & Destructor Documentation

### 14.68.2.1 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

## 14.69 epiworld::vecHasher< T > Struct Template Reference

Vector hasher.

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

### Public Member Functions

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

#### 14.69.1 Detailed Description

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

Vector hasher.

Template Parameters

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

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

- epiworld.hpp

## 14.70 vecHasher< T > Struct Template Reference

Vector hasher.

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

### Public Member Functions

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

#### 14.70.1 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

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

[Virus.](#)

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

### Public Member Functions

- **Virus** (std::string name="unknown virus")
- void **mutate** ([Model](#)< TSeq > \*model)
- void **set\_mutation** (MutFun< TSeq > fun)
- std::shared\_ptr< TSeq > **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
- std::vector< epiworld\_double > & **get\_data** ()
- bool **operator==** (const [Virus](#)< TSeq > &other) const
- bool **operator!=** (const [Virus](#)< TSeq > &other) const
- void **print** () const

### Get and set the tool functions

#### Parameters

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

#### Returns

*epiworld\_double*

- epiworld\_double **get\_prob\_infecting** ([Model](#)< TSeq > \*model)
- epiworld\_double **get\_prob\_recovery** ([Model](#)< TSeq > \*model)
- epiworld\_double **get\_prob\_death** ([Model](#)< TSeq > \*model)
- 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)

## 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)

### 14.71.1 Detailed Description

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

[Virus](#).

#### Template Parameters

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

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

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

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

- epiworld.hpp

## 14.72 Virus< TSeq > Class Template Reference

[Virus.](#)

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

### Public Member Functions

- **Virus** (std::string name="unknown virus")
- void **mutate** ([Model](#)< TSeq > \*model)
- void **set\_mutation** (MutFun< TSeq > fun)
- std::shared\_ptr< TSeq > **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
- std::vector< epiworld\_double > & **get\_data** ()
- bool **operator==** (const [Virus](#)< TSeq > &other) const
- bool **operator!=** (const [Virus](#)< TSeq > &other) const
- void **print** () const
- bool **operator==** (const [Virus](#)< std::vector< int >> &other) const
- bool **operator==** (const [Virus](#)< std::vector< int >> &other) const

### Get and set the tool functions

#### Parameters

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

#### Returns

*epiworld\_double*

- epiworld\_double **get\_prob\_infecting** ([Model](#)< TSeq > \*model)
- epiworld\_double **get\_prob\_recovery** ([Model](#)< TSeq > \*model)
- epiworld\_double **get\_prob\_death** ([Model](#)< TSeq > \*model)
- 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)

## 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)

### 14.72.1 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

## 14.73 epiworld::Viruses< TSeq > Class Template Reference

Set of viruses (useful for building iterators)

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

### Public Member Functions

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

### Friends

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

### 14.73.1 Detailed Description

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

Set of viruses (useful for building iterators)

#### Template Parameters

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

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

- epiworld.hpp

## 14.74 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 >

### 14.74.1 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

## 14.75 epiworld::Viruses\_const< TSeq > Class Template Reference

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

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

## Public Member Functions

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

## Friends

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

### 14.75.1 Detailed Description

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

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

#### Template Parameters

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

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

- epiworld.hpp

## 14.76 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 >

### 14.76.1 Detailed Description

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

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

#### Template Parameters

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

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

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



## Chapter 15

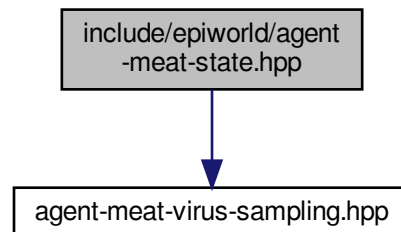
# File Documentation

### 15.1 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)`

### 15.1.1 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  
  epiworld::Model< TSeq >, 81  
  Model< TSeq >, 96  
AdjList, 37  
  AdjList, 37  
  epiworld::AdjList, 39  
  read\_edgelist, 38  
Agent< TSeq >, 40  
  default\_rm\_entity, 43  
  operator(), 42  
  swap\_neighbors, 43  
AgentsSample  
  AgentsSample< TSeq >, 48  
  epiworld::AgentsSample< TSeq >, 50  
AgentsSample< TSeq >, 47  
  AgentsSample, 48  
clone\_ptr  
  epiworld::epimodels::ModelSEIRCONN< TSeq >, 112  
  epiworld::epimodels::ModelSEIRDCONN< TSeq >, 125  
  epiworld::epimodels::ModelSIRCONN< TSeq >, 139  
  epiworld::epimodels::ModelSIRDCONN< TSeq >, 150  
  epiworld::epimodels::ModelSIRLogit< TSeq >, 157  
  epiworld::Model< TSeq >, 82  
  Model< TSeq >, 96  
  ModelSEIRCONN< TSeq >, 115  
  ModelSEIRDCONN< TSeq >, 128  
  ModelSEIRMixing< TSeq >, 132  
  ModelSIRCONN< TSeq >, 142  
  ModelSIRDCONN< TSeq >, 153  
  ModelSIRLogit< TSeq >, 160  
DataBase< TSeq >, 50  
  generation\_time, 52  
  get\_transmissions, 53  
  operator==, 53, 54  
  record\_virus, 54  
  reproductive\_number, 54  
  transition\_probability, 55  
default\_rm\_entity  
  Agent< TSeq >, 43  
  Entity< TSeq >, 63  
  epiworld::Agent< TSeq >, 47  
  epiworld::Entity< TSeq >, 64  
Entities< TSeq >, 59  
Entities\_const< TSeq >, 61  
Entity< TSeq >, 63  
  default\_rm\_entity, 63  
epiworld::AdjList, 38  
  AdjList, 39  
  read\_edgelist, 39  
epiworld::Agent< TSeq >, 44  
  default\_rm\_entity, 47  
  operator(), 46  
  swap\_neighbors, 46  
epiworld::AgentsSample< TSeq >, 49  
  AgentsSample, 50  
epiworld::DataBase< TSeq >, 55  
  generation\_time, 57  
  get\_transmissions, 58  
  operator==, 58  
  record\_virus, 58  
  reproductive\_number, 59  
  transition\_probability, 59  
epiworld::Entities< TSeq >, 60  
epiworld::Entities\_const< TSeq >, 62  
epiworld::Entity< TSeq >, 64  
  default\_rm\_entity, 64  
epiworld::epimodels::ModelDiffNet< TSeq >, 102  
epiworld::epimodels::ModelSEIR< TSeq >, 105  
  initial\_states, 107  
  update\_exposed\_seir, 107  
  update\_infected\_seir, 107  
epiworld::epimodels::ModelSEIRCONN< TSeq >, 111  
  clone\_ptr, 112  
  initial\_states, 113  
  ModelSEIRCONN, 112  
  reset, 113  
epiworld::epimodels::ModelSEIRD< TSeq >, 117  
  ModelSEIRD, 118, 119  
  update\_exposed\_seir, 119  
epiworld::epimodels::ModelSEIRDCONN< TSeq >, 123  
  clone\_ptr, 125  
  initial\_states, 125  
  ModelSEIRDCONN, 124  
  reset, 125  
epiworld::epimodels::ModelSIR< TSeq >, 133  
  initial\_states, 134  
epiworld::epimodels::ModelSIRCONN< TSeq >, 137  
  clone\_ptr, 139  
  get\_n\_infected, 139  
  initial\_states, 140

- ModelSIRCONN, 138
- reset, 140
- epiworld::epimodels::ModelSIRD< TSeq >, 144
  - initial\_states, 146
  - ModelSIRD, 145
- epiworld::epimodels::ModelSIRDCONN< TSeq >, 149
  - clone\_ptr, 150
  - ModelSIRDCONN, 150
  - reset, 151
- epiworld::epimodels::ModelSIRLogit< TSeq >, 154
  - clone\_ptr, 157
  - ModelSIRLogit, 156
  - reset, 157
- epiworld::epimodels::ModelSIS< TSeq >, 161
- epiworld::epimodels::ModelSISD< TSeq >, 164
- epiworld::epimodels::ModelSURV< TSeq >, 167
- epiworld::Event< TSeq >, 65
  - Event, 65
- epiworld::GlobalEvent< TSeq >, 68
  - GlobalEvent, 69
- epiworld::LFMCMC< TData >, 71
- epiworld::Model< TSeq >, 73
  - add\_globalevent, 81
  - clone\_ptr, 82
  - events\_add, 82
  - events\_run, 83
  - initial\_states\_fun, 86
  - load\_agents\_entities\_ties, 83
  - rbinomd, 86
  - reset, 83
  - rexp, 86
  - rgammad, 86
  - rlognormald, 86
  - rnormd, 87
  - run\_multiple, 84
  - runifd, 87
  - set\_agents\_data, 84
  - set\_name, 85
  - time\_elapsed, 87
  - write\_data, 85
- epiworld::PersonTools< TSeq >, 171
- epiworld::Progress, 171
- epiworld::Queue< TSeq >, 172
- epiworld::sampler, 29
  - make\_sample\_virus\_neighbors, 29
  - make\_update\_susceptible, 30
  - sample\_virus\_single, 30
- epiworld::SAMPLETYPE, 174
- epiworld::Tool< TSeq >, 175
- epiworld::Tools< TSeq >, 178
- epiworld::Tools\_const< TSeq >, 180
- epiworld::UserData< TSeq >, 181
  - UserData, 183
- epiworld::vecHasher< T >, 185
- epiworld::Virus< TSeq >, 186
- epiworld::Viruses< TSeq >, 190
- epiworld::Viruses\_const< TSeq >, 191
- Event
  - epiworld::Event< TSeq >, 65
  - Event< TSeq >, 67
- Event< TSeq >, 66
  - Event, 67
- events\_add
  - epiworld::Model< TSeq >, 82
  - Model< TSeq >, 97
- events\_run
  - epiworld::Model< TSeq >, 83
  - Model< TSeq >, 97
- generation\_time
  - DataBase< TSeq >, 52
  - epiworld::DataBase< TSeq >, 57
- get\_n\_infected
  - epiworld::epimodels::ModelSIRCONN< TSeq >, 139
  - ModelSIRCONN< TSeq >, 143
- get\_transmissions
  - DataBase< TSeq >, 53
  - epiworld::DataBase< TSeq >, 58
- GlobalEvent
  - epiworld::GlobalEvent< TSeq >, 69
  - GlobalEvent< TSeq >, 70
- GlobalEvent< TSeq >, 69
  - GlobalEvent, 70
- GroupSampler< TSeq >, 70
- include/epiworld/agent-meat-state.hpp, 195
- initial\_states
  - epiworld::epimodels::ModelSEIR< TSeq >, 107
  - epiworld::epimodels::ModelSEIRCONN< TSeq >, 113
  - epiworld::epimodels::ModelSEIRDCONN< TSeq >, 125
  - epiworld::epimodels::ModelSIR< TSeq >, 134
  - epiworld::epimodels::ModelSIRCONN< TSeq >, 140
  - epiworld::epimodels::ModelSIRD< TSeq >, 146
  - ModelSEIR< TSeq >, 109
  - ModelSEIRCONN< TSeq >, 116
  - ModelSEIRDCONN< TSeq >, 128
  - ModelSEIRMixing< TSeq >, 132
  - ModelSIR< TSeq >, 136
  - ModelSIRCONN< TSeq >, 143
  - ModelSIRD< TSeq >, 148
- initial\_states\_fun
  - epiworld::Model< TSeq >, 86
  - Model< TSeq >, 100
- LFMCMC< TData >, 72
- load\_agents\_entities\_ties
  - epiworld::Model< TSeq >, 83
  - Model< TSeq >, 97
- make\_sample\_virus\_neighbors
  - epiworld::sampler, 29
  - sampler, 32
- make\_update\_susceptible

- epiworld::sampler, 30
- sampler, 33
- Model< TSeq >, 88
  - add\_globlevent, 96
  - clone\_ptr, 96
  - events\_add, 97
  - events\_run, 97
  - initial\_states\_fun, 100
  - load\_agents\_entities\_ties, 97
  - rbinomd, 100
  - reset, 98
  - rexp, 100
  - rgammad, 100
  - rlognormald, 101
  - rnormd, 101
  - run\_multiple, 98
  - runifd, 101
  - set\_agents\_data, 99
  - set\_name, 99
  - time\_elapsed, 101
  - write\_data, 99
- ModelDiffNet< TSeq >, 103
- ModelSEIR< TSeq >, 108
  - initial\_states, 109
  - update\_exposed\_seir, 110
  - update\_infected\_seir, 110
- ModelSEIRCONN
  - epiworld::epimodels::ModelSEIRCONN< TSeq >, 112
  - ModelSEIRCONN< TSeq >, 115
- ModelSEIRCONN< TSeq >, 114
  - clone\_ptr, 115
  - initial\_states, 116
  - ModelSEIRCONN, 115
  - reset, 116
- ModelSEIRD
  - epiworld::epimodels::ModelSEIRD< TSeq >, 118, 119
  - ModelSEIRD< TSeq >, 121, 122
- ModelSEIRD< TSeq >, 120
  - ModelSEIRD, 121, 122
  - update\_exposed\_seir, 122
- ModelSEIRDCONN
  - epiworld::epimodels::ModelSEIRDCONN< TSeq >, 124
  - ModelSEIRDCONN< TSeq >, 127
- ModelSEIRDCONN< TSeq >, 126
  - clone\_ptr, 128
  - initial\_states, 128
  - ModelSEIRDCONN, 127
  - reset, 129
- ModelSEIRMixing
  - ModelSEIRMixing< TSeq >, 130, 131
- ModelSEIRMixing< TSeq >, 129
  - clone\_ptr, 132
  - initial\_states, 132
  - ModelSEIRMixing, 130, 131
  - reset, 132
- ModelSIR< TSeq >, 135
  - initial\_states, 136
- ModelSIRCONN
  - epiworld::epimodels::ModelSIRCONN< TSeq >, 138
  - ModelSIRCONN< TSeq >, 142
- ModelSIRCONN< TSeq >, 141
  - clone\_ptr, 142
  - get\_n\_infected, 143
  - initial\_states, 143
  - ModelSIRCONN, 142
  - reset, 143
- ModelSIRD
  - epiworld::epimodels::ModelSIRD< TSeq >, 145
  - ModelSIRD< TSeq >, 147
- ModelSIRD< TSeq >, 146
  - initial\_states, 148
  - ModelSIRD, 147
- ModelSIRDCONN
  - epiworld::epimodels::ModelSIRDCONN< TSeq >, 150
  - ModelSIRDCONN< TSeq >, 152
- ModelSIRDCONN< TSeq >, 151
  - clone\_ptr, 153
  - ModelSIRDCONN, 152
  - reset, 153
- ModelSIRLogit
  - epiworld::epimodels::ModelSIRLogit< TSeq >, 156
  - ModelSIRLogit< TSeq >, 159
- ModelSIRLogit< TSeq >, 158
  - clone\_ptr, 160
  - ModelSIRLogit, 159
  - reset, 160
- ModelSIS< TSeq >, 162
- ModelSISD< TSeq >, 165
- ModelSURV< TSeq >, 169
- Network< Nettype, Nodetype, Edgetype >, 171
- operator()
  - Agent< TSeq >, 42
  - epiworld::Agent< TSeq >, 46
- operator==
  - DataBase< TSeq >, 53, 54
  - epiworld::DataBase< TSeq >, 58
- PersonTools< TSeq >, 171
- Progress, 172
- Queue< TSeq >, 173
- RandGraph, 174
- rbinomd
  - epiworld::Model< TSeq >, 86
  - Model< TSeq >, 100
- read\_edgelist
  - AdjList, 38
  - epiworld::AdjList, 39

- record\_virus
  - DataBase< TSeq >, [54](#)
  - epiworld::DataBase< TSeq >, [58](#)
- reproductive\_number
  - DataBase< TSeq >, [54](#)
  - epiworld::DataBase< TSeq >, [59](#)
- reset
  - epiworld::epimodels::ModelSEIRCONN< TSeq >, [113](#)
  - epiworld::epimodels::ModelSEIRDCONN< TSeq >, [125](#)
  - epiworld::epimodels::ModelSIRCONN< TSeq >, [140](#)
  - epiworld::epimodels::ModelSIRDCONN< TSeq >, [151](#)
  - epiworld::epimodels::ModelSIRLogit< TSeq >, [157](#)
  - epiworld::Model< TSeq >, [83](#)
  - Model< TSeq >, [98](#)
  - ModelSEIRCONN< TSeq >, [116](#)
  - ModelSEIRDCONN< TSeq >, [129](#)
  - ModelSEIRMixing< TSeq >, [132](#)
  - ModelSIRCONN< TSeq >, [143](#)
  - ModelSIRDCONN< TSeq >, [153](#)
  - ModelSIRLogit< TSeq >, [160](#)
- rexp
  - epiworld::Model< TSeq >, [86](#)
  - Model< TSeq >, [100](#)
- rgammad
  - epiworld::Model< TSeq >, [86](#)
  - Model< TSeq >, [100](#)
- rlognormald
  - epiworld::Model< TSeq >, [86](#)
  - Model< TSeq >, [101](#)
- rnornd
  - epiworld::Model< TSeq >, [87](#)
  - Model< TSeq >, [101](#)
- run\_multiple
  - epiworld::Model< TSeq >, [84](#)
  - Model< TSeq >, [98](#)
- runifd
  - epiworld::Model< TSeq >, [87](#)
  - Model< TSeq >, [101](#)
- sample\_virus\_single
  - epiworld::sampler, [30](#)
  - sampler, [33](#)
- sampler, [32](#)
  - make\_sample\_virus\_neighbors, [32](#)
  - make\_update\_susceptible, [33](#)
  - sample\_virus\_single, [33](#)
- SAMPLETYPE, [175](#)
- set\_agents\_data
  - epiworld::Model< TSeq >, [84](#)
  - Model< TSeq >, [99](#)
- set\_name
  - epiworld::Model< TSeq >, [85](#)
  - Model< TSeq >, [99](#)
- swap\_neighbors
  - Agent< TSeq >, [43](#)
  - epiworld::Agent< TSeq >, [46](#)
- time\_elapsed
  - epiworld::Model< TSeq >, [87](#)
  - Model< TSeq >, [101](#)
  - Tool< TSeq >, [176](#)
  - Tools< TSeq >, [179](#)
  - Tools\_const< TSeq >, [180](#)
- transition\_probability
  - DataBase< TSeq >, [55](#)
  - epiworld::DataBase< TSeq >, [59](#)
- update\_exposed\_seir
  - epiworld::epimodels::ModelSEIR< TSeq >, [107](#)
  - epiworld::epimodels::ModelSEIRD< TSeq >, [119](#)
  - ModelSEIR< TSeq >, [110](#)
  - ModelSEIRD< TSeq >, [122](#)
- update\_infected\_seir
  - epiworld::epimodels::ModelSEIR< TSeq >, [107](#)
  - ModelSEIR< TSeq >, [110](#)
- UserData
  - epiworld::UserData< TSeq >, [183](#)
  - UserData< TSeq >, [184](#)
- UserData< TSeq >, [183](#)
  - UserData, [184](#)
- vecHasher< T >, [185](#)
- Virus< TSeq >, [188](#)
- Viruses< TSeq >, [191](#)
- Viruses\_const< TSeq >, [192](#)
- write\_data
  - epiworld::Model< TSeq >, [85](#)
  - Model< TSeq >, [99](#)