

epiworld

0.0-1

Generated by Doxygen 1.9.1



<b>1 Example: 00-hello-world</b>	<b>1</b>
<b>2 Benchmarking</b>	<b>3</b>
<b>3 Contributor Code of Conduct</b>	<b>5</b>
<b>4 epiworld c++ template library</b>	<b>7</b>
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
<b>5 General parameters</b>	<b>11</b>
5.1 Compartmental Models . . . . .	11
5.1.1 SIR Model . . . . .	11
5.1.2 SEIR Model . . . . .	12
5.2 Agent-Based Model Approach . . . . .	12
5.2.1 Mathematical preliminaries . . . . .	13
5.2.2 Simulation study . . . . .	13
5.3 Comparing ABM with Compartmental Models . . . . .	13
5.3.1 SIR . . . . .	13
5.3.2 SEIR . . . . .	14
5.3.3 Rates . . . . .	14
<b>6 MIT License</b>	<b>15</b>
<b>7 model1</b>	<b>17</b>
<b>8 Mixing probabilities in connected model</b>	<b>19</b>
8.1 Case 1: No grouping . . . . .	19
8.2 Case 2: Grouping . . . . .	20
<b>9 EPI Simulator</b>	<b>21</b>
9.1 Disease dynamics . . . . .	21
9.2 Network dynamics . . . . .	21
9.3 Contagion dynamics . . . . .	21
9.4 Time dynamics . . . . .	21
9.5 Updating agent's status . . . . .	22
9.5.1 Other parameters . . . . .	22
<b>10 Namespace Index</b>	<b>23</b>
10.1 Namespace List . . . . .	23
<b>11 Hierarchical Index</b>	<b>25</b>

11.1 Class Hierarchy	25
<b>12 Class Index</b>	<b>27</b>
12.1 Class List	27
<b>13 File Index</b>	<b>31</b>
13.1 File List	31
<b>14 Namespace Documentation</b>	<b>33</b>
14.1 epiworld::sampler Namespace Reference	33
14.1.1 Detailed Description	33
14.1.2 Function Documentation	33
14.1.2.1 make_sample_virus_neighbors()	33
14.1.2.2 make_update_susceptible()	34
14.1.2.3 sample_virus_single()	34
14.2 sampler Namespace Reference	36
14.2.1 Detailed Description	36
14.2.2 Function Documentation	36
14.2.2.1 make_sample_virus_neighbors()	36
14.2.2.2 make_update_susceptible()	37
14.2.2.3 sample_virus_single()	37
<b>15 Class Documentation</b>	<b>41</b>
15.1 AdjList Class Reference	41
15.1.1 Constructor & Destructor Documentation	41
15.1.1.1 AdjList()	41
15.1.2 Member Function Documentation	42
15.1.2.1 read_edgelist()	42
15.2 epiworld::AdjList Class Reference	42
15.2.1 Constructor & Destructor Documentation	43
15.2.1.1 AdjList()	43
15.2.2 Member Function Documentation	43
15.2.2.1 read_edgelist()	43
15.3 Agent< TSeq > Class Template Reference	44
15.3.1 Detailed Description	46
15.3.2 Member Function Documentation	46
15.3.2.1 operator()()	46
15.3.2.2 swap_neighbors()	48
15.3.3 Friends And Related Function Documentation	48
15.3.3.1 default_rm_entity	48
15.4 epiworld::Agent< TSeq > Class Template Reference	49
15.4.1 Detailed Description	51
15.4.2 Member Function Documentation	51
15.4.2.1 operator()()	51

15.4.2.2 swap_neighbors()	52
15.4.3 Friends And Related Function Documentation	52
15.4.3.1 default_rm_entity	52
15.5 AgentsSample< TSeq > Class Template Reference	52
15.5.1 Detailed Description	53
15.5.2 Constructor & Destructor Documentation	53
15.5.2.1 AgentsSample()	53
15.6 epiworld::AgentsSample< TSeq > Class Template Reference	54
15.6.1 Detailed Description	54
15.6.2 Constructor & Destructor Documentation	55
15.6.2.1 AgentsSample()	55
15.7 DataBase< TSeq > Class Template Reference	55
15.7.1 Detailed Description	57
15.7.2 Member Function Documentation	58
15.7.2.1 generation_time()	58
15.7.2.2 get_transmissions()	58
15.7.2.3 operator==( ) [1/3]	58
15.7.2.4 operator==( ) [2/3]	59
15.7.2.5 operator==( ) [3/3]	59
15.7.2.6 record_virus()	59
15.7.2.7 reproductive_number()	60
15.7.2.8 transition_probability()	60
15.8 epiworld::DataBase< TSeq > Class Template Reference	60
15.8.1 Detailed Description	62
15.8.2 Member Function Documentation	63
15.8.2.1 generation_time()	63
15.8.2.2 get_transmissions()	63
15.8.2.3 operator==( )	63
15.8.2.4 record_virus()	64
15.8.2.5 reproductive_number()	64
15.8.2.6 transition_probability()	64
15.9 Entities< TSeq > Class Template Reference	65
15.9.1 Detailed Description	65
15.10 epiworld::Entities< TSeq > Class Template Reference	66
15.10.1 Detailed Description	66
15.11 Entities_const< TSeq > Class Template Reference	66
15.11.1 Detailed Description	67
15.12 epiworld::Entities_const< TSeq > Class Template Reference	67
15.12.1 Detailed Description	68
15.13 Entity< TSeq > Class Template Reference	68
15.13.1 Constructor & Destructor Documentation	69
15.13.1.1 Entity()	69

15.13.2 Friends And Related Function Documentation	69
15.13.2.1 default_rm_entity	69
15.14 epiworld::Entity< TSeq > Class Template Reference	70
15.14.1 Constructor & Destructor Documentation	70
15.14.1.1 Entity()	70
15.14.2 Friends And Related Function Documentation	71
15.14.2.1 default_rm_entity	71
15.15 epiworld::Event< TSeq > Struct Template Reference	71
15.15.1 Detailed Description	71
15.15.2 Constructor & Destructor Documentation	72
15.15.2.1 Event()	72
15.16 Event< TSeq > Struct Template Reference	72
15.16.1 Detailed Description	73
15.16.2 Constructor & Destructor Documentation	74
15.16.2.1 Event()	74
15.17 epiworld::GlobalEvent< TSeq > Class Template Reference	74
15.17.1 Detailed Description	75
15.17.2 Constructor & Destructor Documentation	75
15.17.2.1 GlobalEvent()	75
15.18 GlobalEvent< TSeq > Class Template Reference	76
15.18.1 Detailed Description	76
15.18.2 Constructor & Destructor Documentation	76
15.18.2.1 GlobalEvent()	76
15.19 epiworld::GroupSampler< TSeq > Class Template Reference	77
15.19.1 Detailed Description	77
15.20 GroupSampler< TSeq > Class Template Reference	77
15.20.1 Detailed Description	78
15.21 epiworld::LFMCMC< TData > Class Template Reference	78
15.21.1 Detailed Description	79
15.22 LFMCMC< TData > Class Template Reference	79
15.22.1 Detailed Description	80
15.23 epiworld::Model< TSeq > Class Template Reference	81
15.23.1 Detailed Description	88
15.23.2 Member Function Documentation	89
15.23.2.1 add_globlevent()	89
15.23.2.2 clone_ptr()	89
15.23.2.3 events_add()	90
15.23.2.4 events_run()	90
15.23.2.5 load_agents_entities_ties()	90
15.23.2.6 reset()	91
15.23.2.7 run_multiple()	91
15.23.2.8 set_agents_data()	92

15.23.2.9 set_name()	92
15.23.2.10 write_data()	92
15.23.3 Member Data Documentation	93
15.23.3.1 initial_states_fun	93
15.23.3.2 rbinomd	93
15.23.3.3 rexp	93
15.23.3.4 rgammad	94
15.23.3.5 rlognormald	94
15.23.3.6 rnormd	94
15.23.3.7 runifd	94
15.23.3.8 time_elapsed	94
15.24 Model< TSeq > Class Template Reference	95
15.24.1 Detailed Description	102
15.24.2 Member Function Documentation	103
15.24.2.1 add_globalevent()	103
15.24.2.2 clone_ptr()	103
15.24.2.3 events_add()	103
15.24.2.4 events_run()	104
15.24.2.5 load_agents_entities_ties()	104
15.24.2.6 reset()	105
15.24.2.7 run_multiple()	105
15.24.2.8 set_agents_data()	105
15.24.2.9 set_name()	106
15.24.2.10 write_data()	106
15.24.3 Member Data Documentation	107
15.24.3.1 initial_states_fun	107
15.24.3.2 rbinomd	107
15.24.3.3 rexp	107
15.24.3.4 rgammad	108
15.24.3.5 rlognormald	108
15.24.3.6 rnormd	108
15.24.3.7 runifd	108
15.24.3.8 time_elapsed	108
15.25 epiworld::epimodels::ModelDiffNet< TSeq > Class Template Reference	109
15.25.1 Detailed Description	110
15.26 ModelDiffNet< TSeq > Class Template Reference	110
15.26.1 Detailed Description	112
15.27 epiworld::epimodels::ModelSEIR< TSeq > Class Template Reference	112
15.27.1 Detailed Description	113
15.27.2 Member Function Documentation	114
15.27.2.1 initial_states()	114
15.27.3 Member Data Documentation	114

15.27.3.1 update_exposed_seir . . . . .	114
15.27.3.2 update_infected_seir . . . . .	115
15.28 ModelSEIR< TSeq > Class Template Reference . . . . .	115
15.28.1 Detailed Description . . . . .	116
15.28.2 Member Function Documentation . . . . .	116
15.28.2.1 initial_states() . . . . .	117
15.28.3 Member Data Documentation . . . . .	117
15.28.3.1 update_exposed_seir . . . . .	117
15.28.3.2 update_infected_seir . . . . .	117
15.29 epiworld::epimodels::ModelSEIRCONN< TSeq > Class Template Reference . . . . .	118
15.29.1 Constructor & Destructor Documentation . . . . .	119
15.29.1.1 ModelSEIRCONN() . . . . .	119
15.29.2 Member Function Documentation . . . . .	119
15.29.2.1 clone_ptr() . . . . .	120
15.29.2.2 initial_states() . . . . .	120
15.29.2.3 reset() . . . . .	120
15.30 ModelSEIRCONN< TSeq > Class Template Reference . . . . .	121
15.30.1 Constructor & Destructor Documentation . . . . .	122
15.30.1.1 ModelSEIRCONN() . . . . .	122
15.30.2 Member Function Documentation . . . . .	122
15.30.2.1 clone_ptr() . . . . .	123
15.30.2.2 initial_states() . . . . .	123
15.30.2.3 reset() . . . . .	123
15.31 epiworld::epimodels::ModelSEIRD< TSeq > Class Template Reference . . . . .	124
15.31.1 Detailed Description . . . . .	125
15.31.2 Constructor & Destructor Documentation . . . . .	125
15.31.2.1 ModelSEIRD() [1/2] . . . . .	125
15.31.2.2 ModelSEIRD() [2/2] . . . . .	126
15.31.3 Member Data Documentation . . . . .	126
15.31.3.1 update_exposed_seir . . . . .	126
15.32 ModelSEIRD< TSeq > Class Template Reference . . . . .	127
15.32.1 Detailed Description . . . . .	128
15.32.2 Constructor & Destructor Documentation . . . . .	128
15.32.2.1 ModelSEIRD() [1/2] . . . . .	128
15.32.2.2 ModelSEIRD() [2/2] . . . . .	129
15.32.3 Member Data Documentation . . . . .	129
15.32.3.1 update_exposed_seir . . . . .	129
15.33 epiworld::epimodels::ModelSEIRDCONN< TSeq > Class Template Reference . . . . .	130
15.33.1 Constructor & Destructor Documentation . . . . .	131
15.33.1.1 ModelSEIRDCONN() . . . . .	131
15.33.2 Member Function Documentation . . . . .	132
15.33.2.1 clone_ptr() . . . . .	132



15.33.2.2 initial_states()	132
15.33.2.3 reset()	132
15.34 ModelSEIRDCONN< TSeq > Class Template Reference	133
15.34.1 Constructor & Destructor Documentation	134
15.34.1.1 ModelSEIRDCONN()	134
15.34.2 Member Function Documentation	135
15.34.2.1 clone_ptr()	135
15.34.2.2 initial_states()	135
15.34.2.3 reset()	136
15.35 epiworld::epimodels::ModelSEIRMixing< TSeq > Class Template Reference	136
15.35.1 Constructor & Destructor Documentation	137
15.35.1.1 ModelSEIRMixing() [1/2]	138
15.35.1.2 ModelSEIRMixing() [2/2]	138
15.35.2 Member Function Documentation	139
15.35.2.1 clone_ptr()	139
15.35.2.2 initial_states()	139
15.35.2.3 reset()	140
15.36 ModelSEIRMixing< TSeq > Class Template Reference	140
15.36.1 Constructor & Destructor Documentation	141
15.36.1.1 ModelSEIRMixing() [1/2]	141
15.36.1.2 ModelSEIRMixing() [2/2]	142
15.36.2 Member Function Documentation	142
15.36.2.1 clone_ptr()	143
15.36.2.2 initial_states()	143
15.36.2.3 reset()	143
15.37 epiworld::epimodels::ModelSIR< TSeq > Class Template Reference	144
15.37.1 Detailed Description	145
15.37.2 Member Function Documentation	145
15.37.2.1 initial_states()	145
15.38 ModelSIR< TSeq > Class Template Reference	145
15.38.1 Detailed Description	146
15.38.2 Member Function Documentation	147
15.38.2.1 initial_states()	147
15.39 epiworld::epimodels::ModelSIRCONN< TSeq > Class Template Reference	148
15.39.1 Constructor & Destructor Documentation	149
15.39.1.1 ModelSIRCONN()	149
15.39.2 Member Function Documentation	149
15.39.2.1 clone_ptr()	150
15.39.2.2 get_n_infected()	150
15.39.2.3 initial_states()	150
15.39.2.4 reset()	150
15.40 ModelSIRCONN< TSeq > Class Template Reference	151

15.40.1 Constructor & Destructor Documentation	152
15.40.1.1 ModelSIRCONN()	152
15.40.2 Member Function Documentation	153
15.40.2.1 clone_ptr()	153
15.40.2.2 get_n_infected()	153
15.40.2.3 initial_states()	153
15.40.2.4 reset()	154
15.41 epiworld::epimodels::ModelSIRD< TSeq > Class Template Reference	154
15.41.1 Detailed Description	155
15.41.2 Constructor & Destructor Documentation	155
15.41.2.1 ModelSIRD()	155
15.41.3 Member Function Documentation	156
15.41.3.1 initial_states()	156
15.42 ModelSIRD< TSeq > Class Template Reference	156
15.42.1 Detailed Description	157
15.42.2 Constructor & Destructor Documentation	158
15.42.2.1 ModelSIRD()	158
15.42.3 Member Function Documentation	158
15.42.3.1 initial_states()	158
15.43 epiworld::epimodels::ModelSIRDCONN< TSeq > Class Template Reference	159
15.43.1 Constructor & Destructor Documentation	160
15.43.1.1 ModelSIRDCONN()	160
15.43.2 Member Function Documentation	161
15.43.2.1 clone_ptr()	161
15.43.2.2 reset()	161
15.44 ModelSIRDCONN< TSeq > Class Template Reference	162
15.44.1 Constructor & Destructor Documentation	163
15.44.1.1 ModelSIRDCONN()	163
15.44.2 Member Function Documentation	163
15.44.2.1 clone_ptr()	163
15.44.2.2 reset()	164
15.45 epiworld::epimodels::ModelSIRLogit< TSeq > Class Template Reference	164
15.45.1 Detailed Description	165
15.45.2 Constructor & Destructor Documentation	166
15.45.2.1 ModelSIRLogit()	166
15.45.3 Member Function Documentation	167
15.45.3.1 clone_ptr()	167
15.45.3.2 reset()	167
15.46 ModelSIRLogit< TSeq > Class Template Reference	168
15.46.1 Detailed Description	169
15.46.2 Constructor & Destructor Documentation	169
15.46.2.1 ModelSIRLogit()	169

15.46.3 Member Function Documentation	170
15.46.3.1 clone_ptr()	170
15.46.3.2 reset()	170
15.47 epiworld::epimodels::ModelSIRMixing< TSeq > Class Template Reference	171
15.47.1 Constructor & Destructor Documentation	172
15.47.1.1 ModelSIRMixing() [1/2]	172
15.47.1.2 ModelSIRMixing() [2/2]	173
15.47.2 Member Function Documentation	173
15.47.2.1 clone_ptr()	174
15.47.2.2 initial_states()	174
15.47.2.3 reset()	174
15.48 ModelSIRMixing< TSeq > Class Template Reference	175
15.48.1 Constructor & Destructor Documentation	176
15.48.1.1 ModelSIRMixing() [1/2]	176
15.48.1.2 ModelSIRMixing() [2/2]	177
15.48.2 Member Function Documentation	177
15.48.2.1 clone_ptr()	177
15.48.2.2 initial_states()	177
15.48.2.3 reset()	178
15.49 epiworld::epimodels::ModelSIS< TSeq > Class Template Reference	178
15.49.1 Detailed Description	179
15.50 ModelSIS< TSeq > Class Template Reference	180
15.50.1 Detailed Description	181
15.51 epiworld::epimodels::ModelSISD< TSeq > Class Template Reference	181
15.51.1 Detailed Description	182
15.52 ModelSISD< TSeq > Class Template Reference	183
15.52.1 Detailed Description	184
15.53 epiworld::epimodels::ModelSURV< TSeq > Class Template Reference	184
15.54 ModelSURV< TSeq > Class Template Reference	186
15.55 Network< Nettype, Nodetype, Edgetype > Class Template Reference	188
15.56 epiworld::PersonTools< TSeq > Class Template Reference	188
15.57 PersonTools< TSeq > Class Template Reference	189
15.58 epiworld::Progress Class Reference	189
15.58.1 Detailed Description	189
15.59 Progress Class Reference	189
15.59.1 Detailed Description	189
15.60 epiworld::Queue< TSeq > Class Template Reference	190
15.60.1 Detailed Description	190
15.61 Queue< TSeq > Class Template Reference	190
15.61.1 Detailed Description	191
15.62 RandGraph Class Reference	191
15.63 epiworld::SAMPLETYPE Class Reference	192

15.64 SAMPLETYPE Class Reference . . . . .	192
15.65 epiworld::Tool< TSeq > Class Template Reference . . . . .	192
15.65.1 Detailed Description . . . . .	193
15.66 Tool< TSeq > Class Template Reference . . . . .	193
15.66.1 Detailed Description . . . . .	195
15.67 epiworld::Tools< TSeq > Class Template Reference . . . . .	195
15.67.1 Detailed Description . . . . .	195
15.68 Tools< TSeq > Class Template Reference . . . . .	196
15.68.1 Detailed Description . . . . .	196
15.69 epiworld::Tools_const< TSeq > Class Template Reference . . . . .	197
15.69.1 Detailed Description . . . . .	197
15.70 Tools_const< TSeq > Class Template Reference . . . . .	197
15.70.1 Detailed Description . . . . .	198
15.71 epiworld::UserData< TSeq > Class Template Reference . . . . .	198
15.71.1 Detailed Description . . . . .	199
15.71.2 Constructor & Destructor Documentation . . . . .	200
15.71.2.1 UserData() . . . . .	200
15.72 UserData< TSeq > Class Template Reference . . . . .	200
15.72.1 Detailed Description . . . . .	201
15.72.2 Constructor & Destructor Documentation . . . . .	201
15.72.2.1 UserData() . . . . .	201
15.73 epiworld::vecHasher< T > Struct Template Reference . . . . .	202
15.73.1 Detailed Description . . . . .	202
15.74 vecHasher< T > Struct Template Reference . . . . .	202
15.74.1 Detailed Description . . . . .	202
15.75 epiworld::Virus< TSeq > Class Template Reference . . . . .	203
15.75.1 Detailed Description . . . . .	204
15.76 Virus< TSeq > Class Template Reference . . . . .	205
15.76.1 Detailed Description . . . . .	207
15.77 epiworld::Viruses< TSeq > Class Template Reference . . . . .	207
15.77.1 Detailed Description . . . . .	207
15.78 Viruses< TSeq > Class Template Reference . . . . .	208
15.78.1 Detailed Description . . . . .	208
15.79 epiworld::Viruses_const< TSeq > Class Template Reference . . . . .	209
15.79.1 Detailed Description . . . . .	209
15.80 Viruses_const< TSeq > Class Template Reference . . . . .	209
15.80.1 Detailed Description . . . . .	210
<b>16 File Documentation</b>	<b>211</b>
16.1 include/epiworld/agent-meat-state.hpp File Reference . . . . .	211
16.1.1 Detailed Description . . . . .	212
<b>Index</b>	<b>213</b>

# 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", .01, true);
    covid19.set_infectiousness(.8);

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

### 4.4 Surveillance simulation

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

### 4.4.1 Preliminary results

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

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

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

```
## - Covid19 (baseline prevalence: 100 seeds)
## Tool(s):
## - Vaccine (baseline prevalence: 25.00%)
##
## Model parameters:
## - Infect period      : 12.0000
## - Latent period      : 3.0000
## - Prob of symptoms   : 0.7000
## - Prob of transmission : 1.0000
## - Prob. death        : 0.0010
## - Prob. reinfect     : 0.1000
## - Surveillance prob. : 0.0020
## - Vax efficacy       : 0.9000
## - Vax redux transmission : 0.5000
##
## Distribution of the population at time 200:
## - Total susceptible (S)      : 19900 -> 2125
## - Total recovered (S)        : 0 -> 17325
## - Total latent (I)           : 100 -> 109
## - Total symptomatic (I)      : 0 -> 155
## - Total symptomatic isolated (I) : 0 -> 8
## - Total asymptomatic (I)     : 0 -> 76
## - Total asymptomatic isolated (I) : 0 -> 1
## - Total removed (R)          : 0 -> 201
##
## (S): Susceptible, (I): Infected, (R): Recovered
##
hist2 <- read.csv("07-surveillance_hist.txt", sep = " ")
surv2 <- read.csv("07-surveillance_user_data.txt", sep = " ")
hist_comb <- rbind(
  cbind(sim = as.character(s_levels[1]), hist1),
  cbind(sim = as.character(s_levels[2]), hist2)
)
ggplot(hist_comb, aes(x = date, y = counts + 1, colour = state, linetype=sim)) +
  geom_line() +
  # scale_y_log10() +
  labs(y = "Counts (log)")
```

#### 4.4.2 Cases detected

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

## Chapter 5

# General parameters

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

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

## 5.1 Compartmental Models

### 5.1.1 SIR Model

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

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

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

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

```

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

```

Now we visualize the model

```

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

```

## 5.1.2 SEIR Model

```

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

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

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

  # Return list of gradients
  list(res)
}

# Initial parameters
parms <- c(
  mu = 0, N = EPI_N, beta = EPI_BETA,
  alpha = 1/EPI_LATENCY, gamma = EPI_GAMMA
)

start <- c(S = EPI_N * (1 - EPI_0), E = EPI_N * EPI_0, I = 0, R = 0)
out_seir <- ode(y = start, times = times, func = seirmod, parms = parms)
out_seir <- as.data.frame(out_seir)
out_seir <- rbind(
  with(out_seir, data.table(date = time, state = "Susceptible", counts = S)),
  with(out_seir, data.table(date = time, state = "Exposed", counts = E)),
  with(out_seir, data.table(date = time, state = "Infected", counts = I)),
  with(out_seir, data.table(date = time, state = "Recovered", counts = R))
)

```

Now we visualize the model

```

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

```

## 5.2 Agent-Based Model Approach

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

```

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

```

```
[1] -0.01049333
```



### 5.2.1 Mathematical preliminaries

That agent  $i$  becomes infected can be computed as follows:

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

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

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

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

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

### 5.2.2 Simulation study

Now, what happens with `epiworld`.

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

## 5.3 Comparing ABM with Compartmental Models

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

### 5.3.1 SIR

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

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

### 5.3.2 SEIR

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

### 5.3.3 Rates

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

## Chapter 6

# MIT License

Copyright (c) 2021 George G. Vega Yon

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

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

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



## Chapter 7

### model1

The dynamics of the simulation process are:

1. Discrete Markov process.

2. The simulation has the following parameters:

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

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

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

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

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

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

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

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

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

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

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

3. The simulation process is as follows:

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

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

## Chapter 8

# Mixing probabilities in connected model

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

### 8.1 Case 1: No grouping

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

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

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

```
quantile(sim_simple)
```

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

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

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

## 8.2 Case 2: Grouping

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

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

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

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

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

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

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

```
quantile(sim_simple)
```

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

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



## Chapter 9

# EPI Simulator

### 9.1 Disease dynamics

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

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

### 9.2 Network dynamics

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

### 9.3 Contagion dynamics

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

### 9.4 Time dynamics

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

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

## 9.5 Updating agent's status

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

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

Where

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

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

### 9.5.1 Other parameters

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

## Chapter 10

# Namespace Index

### 10.1 Namespace List

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

<a href="#">epiworld::sampler</a>	Functions for sampling viruses . . . . .	<a href="#">33</a>
<a href="#">sampler</a>	Functions for sampling viruses . . . . .	<a href="#">36</a>



## Chapter 11

# Hierarchical Index

### 11.1 Class Hierarchy

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

AdjList . . . . .	41
epiworld::AdjList . . . . .	42
Agent< TSeq > . . . . .	44
epiworld::Agent< TSeq > . . . . .	49
Agent< EPI_DEFAULT_TSEQ > . . . . .	44
AgentsSample< TSeq > . . . . .	52
epiworld::AgentsSample< TSeq > . . . . .	54
DataBase< TSeq > . . . . .	55
epiworld::DataBase< TSeq > . . . . .	60
Entities< TSeq > . . . . .	65
epiworld::Entities< TSeq > . . . . .	66
Entities_const< TSeq > . . . . .	66
epiworld::Entities_const< TSeq > . . . . .	67
Entity< TSeq > . . . . .	68
epiworld::Entity< TSeq > . . . . .	70
Entity< EPI_DEFAULT_TSEQ > . . . . .	68
epiworld::Event< TSeq > . . . . .	71
Event< TSeq > . . . . .	72
epiworld::GlobalEvent< TSeq > . . . . .	74
GlobalEvent< TSeq > . . . . .	76
epiworld::GroupSampler< TSeq > . . . . .	77
GroupSampler< TSeq > . . . . .	77
epiworld::LFMCMC< TData > . . . . .	78
LFMCMC< TData > . . . . .	79
epiworld::Model< TSeq > . . . . .	81
Model< TSeq > . . . . .	95
epiworld::Model< EPI_DEFAULT_TSEQ > . . . . .	81
ModelSEIRCONN< TSeq > . . . . .	121
ModelSEIRDCONN< TSeq > . . . . .	133
ModelSEIRMixing< TSeq > . . . . .	140
ModelSIRCONN< TSeq > . . . . .	151
ModelSIRDCONN< TSeq > . . . . .	162
ModelSIRLogit< TSeq > . . . . .	168
ModelSIRMixing< TSeq > . . . . .	175
ModelSURV< TSeq > . . . . .	186

epiworld::epimodels::ModelSEIRCONN< TSeq > . . . . .	118
epiworld::epimodels::ModelSEIRDCONN< TSeq > . . . . .	130
epiworld::epimodels::ModelSEIRMixing< TSeq > . . . . .	136
epiworld::epimodels::ModelSIRCONN< TSeq > . . . . .	148
epiworld::epimodels::ModelSIRDCONN< TSeq > . . . . .	159
epiworld::epimodels::ModelSIRLogit< TSeq > . . . . .	164
epiworld::epimodels::ModelSIRMixing< TSeq > . . . . .	171
epiworld::epimodels::ModelSURV< TSeq > . . . . .	184
epiworld::Model< int > . . . . .	81
ModelDiffNet< TSeq > . . . . .	110
ModelSEIR< TSeq > . . . . .	115
ModelSEIRD< TSeq > . . . . .	127
ModelSIR< TSeq > . . . . .	145
ModelSIRD< TSeq > . . . . .	156
ModelSIS< TSeq > . . . . .	180
ModelSISD< TSeq > . . . . .	183
epiworld::epimodels::ModelDiffNet< TSeq > . . . . .	109
epiworld::epimodels::ModelSEIR< TSeq > . . . . .	112
epiworld::epimodels::ModelSEIRD< TSeq > . . . . .	124
epiworld::epimodels::ModelSIR< TSeq > . . . . .	144
epiworld::epimodels::ModelSIRD< TSeq > . . . . .	154
epiworld::epimodels::ModelSIS< TSeq > . . . . .	178
epiworld::epimodels::ModelSISD< TSeq > . . . . .	181
Network< Nettype, Nodetype, Edgetype > . . . . .	188
epiworld::PersonTools< TSeq > . . . . .	188
PersonTools< TSeq > . . . . .	189
epiworld::Progress . . . . .	189
Progress . . . . .	189
epiworld::Queue< TSeq > . . . . .	190
Queue< TSeq > . . . . .	190
RandGraph . . . . .	191
epiworld::SAMPLETYPE . . . . .	192
SAMPLETYPE . . . . .	192
epiworld::Tool< TSeq > . . . . .	192
Tool< TSeq > . . . . .	193
epiworld::Tools< TSeq > . . . . .	195
Tools< TSeq > . . . . .	196
epiworld::Tools_const< TSeq > . . . . .	197
Tools_const< TSeq > . . . . .	197
epiworld::UserData< TSeq > . . . . .	198
UserData< TSeq > . . . . .	200
epiworld::vecHasher< T > . . . . .	202
vecHasher< T > . . . . .	202
epiworld::Virus< TSeq > . . . . .	203
Virus< TSeq > . . . . .	205
epiworld::Viruses< TSeq > . . . . .	207
Viruses< TSeq > . . . . .	208
epiworld::Viruses_const< TSeq > . . . . .	209
Viruses_const< TSeq > . . . . .	209

## Chapter 12

# Class Index

### 12.1 Class List

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

<a href="#">AdjList</a> . . . . .	41
<a href="#">epiworld::AdjList</a> . . . . .	42
<a href="#">Agent&lt; TSeq &gt;</a>	
<a href="#">Agent</a> (agents) . . . . .	44
<a href="#">epiworld::Agent&lt; TSeq &gt;</a>	
<a href="#">Agent</a> (agents) . . . . .	49
<a href="#">AgentsSample&lt; TSeq &gt;</a>	
Sample of agents . . . . .	52
<a href="#">epiworld::AgentsSample&lt; TSeq &gt;</a>	
Sample of agents . . . . .	54
<a href="#">DataBase&lt; TSeq &gt;</a>	
Statistical data about the process . . . . .	55
<a href="#">epiworld::DataBase&lt; TSeq &gt;</a>	
Statistical data about the process . . . . .	60
<a href="#">Entities&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (useful for building iterators) . . . . .	65
<a href="#">epiworld::Entities&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (useful for building iterators) . . . . .	66
<a href="#">Entities_const&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (const) (useful for iterators) . . . . .	66
<a href="#">epiworld::Entities_const&lt; TSeq &gt;</a>	
Set of <a href="#">Entities</a> (const) (useful for iterators) . . . . .	67
<a href="#">Entity&lt; TSeq &gt;</a> . . . . .	68
<a href="#">epiworld::Entity&lt; TSeq &gt;</a> . . . . .	70
<a href="#">epiworld::Event&lt; TSeq &gt;</a>	
<a href="#">Event</a> data for update an agent . . . . .	71
<a href="#">Event&lt; TSeq &gt;</a>	
<a href="#">Event</a> data for update an agent . . . . .	72
<a href="#">epiworld::GlobalEvent&lt; TSeq &gt;</a>	
Template for a Global <a href="#">Event</a> . . . . .	74
<a href="#">GlobalEvent&lt; TSeq &gt;</a>	
Template for a Global <a href="#">Event</a> . . . . .	76
<a href="#">epiworld::GroupSampler&lt; TSeq &gt;</a>	
Weighted sampling of groups . . . . .	77
<a href="#">GroupSampler&lt; TSeq &gt;</a>	
Weighted sampling of groups . . . . .	77

<a href="#">epiworld::LFMCMC&lt; TData &gt;</a>	
Likelihood-Free Markov Chain Monte Carlo	78
<a href="#">LFMCMC&lt; TData &gt;</a>	
Likelihood-Free Markov Chain Monte Carlo	79
<a href="#">epiworld::Model&lt; TSeq &gt;</a>	
Core class of epiworld	81
<a href="#">Model&lt; TSeq &gt;</a>	
Core class of epiworld	95
<a href="#">epiworld::epimodels::ModelDiffNet&lt; TSeq &gt;</a>	
Template for a <a href="#">Network</a> Diffusion <a href="#">Model</a>	109
<a href="#">ModelDiffNet&lt; TSeq &gt;</a>	
Template for a <a href="#">Network</a> Diffusion <a href="#">Model</a>	110
<a href="#">epiworld::epimodels::ModelSEIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed (SEIR) model	112
<a href="#">ModelSEIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed (SEIR) model	115
<a href="#">epiworld::epimodels::ModelSEIRCONN&lt; TSeq &gt;</a>	118
<a href="#">ModelSEIRCONN&lt; TSeq &gt;</a>	121
<a href="#">epiworld::epimodels::ModelSEIRD&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model	124
<a href="#">ModelSEIRD&lt; TSeq &gt;</a>	
Template for a Susceptible-Exposed-Infected-Removed-Deceased (SEIRD) model	127
<a href="#">epiworld::epimodels::ModelSEIRDCONN&lt; TSeq &gt;</a>	130
<a href="#">ModelSEIRDCONN&lt; TSeq &gt;</a>	133
<a href="#">epiworld::epimodels::ModelSEIRMixing&lt; TSeq &gt;</a>	136
<a href="#">ModelSEIRMixing&lt; TSeq &gt;</a>	140
<a href="#">epiworld::epimodels::ModelSIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed (SIR) model	144
<a href="#">ModelSIR&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed (SIR) model	145
<a href="#">epiworld::epimodels::ModelSIRCONN&lt; TSeq &gt;</a>	148
<a href="#">ModelSIRCONN&lt; TSeq &gt;</a>	151
<a href="#">epiworld::epimodels::ModelSIRD&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed-Deceased (SIRD) model	154
<a href="#">ModelSIRD&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed-Deceased (SIRD) model	156
<a href="#">epiworld::epimodels::ModelSIRDCONN&lt; TSeq &gt;</a>	159
<a href="#">ModelSIRDCONN&lt; TSeq &gt;</a>	162
<a href="#">epiworld::epimodels::ModelSIRLogit&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed (SIR) model	164
<a href="#">ModelSIRLogit&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Removed (SIR) model	168
<a href="#">epiworld::epimodels::ModelSIRMixing&lt; TSeq &gt;</a>	171
<a href="#">ModelSIRMixing&lt; TSeq &gt;</a>	175
<a href="#">epiworld::epimodels::ModelSIS&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Susceptible (SIS) model	178
<a href="#">ModelSIS&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Susceptible (SIS) model	180
<a href="#">epiworld::epimodels::ModelSISD&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Susceptible-Deceased (SISD) model	181
<a href="#">ModelSISD&lt; TSeq &gt;</a>	
Template for a Susceptible-Infected-Susceptible-Deceased (SISD) model	183
<a href="#">epiworld::epimodels::ModelSURV&lt; TSeq &gt;</a>	184
<a href="#">ModelSURV&lt; TSeq &gt;</a>	186
<a href="#">Network&lt; Nettype, Nodetype, Edgetype &gt;</a>	188
<a href="#">epiworld::PersonTools&lt; TSeq &gt;</a>	188
<a href="#">PersonTools&lt; TSeq &gt;</a>	189



<a href="#">epiworld::Progress</a>	
A simple progress bar	189
<a href="#">Progress</a>	
A simple progress bar	189
<a href="#">epiworld::Queue&lt; TSeq &gt;</a>	
Controls which agents are verified at each step	190
<a href="#">Queue&lt; TSeq &gt;</a>	
Controls which agents are verified at each step	190
<a href="#">RandGraph</a>	191
<a href="#">epiworld::SAMPLETYPE</a>	192
<a href="#">SAMPLETYPE</a>	192
<a href="#">epiworld::Tool&lt; TSeq &gt;</a>	
Tools for defending the agent against the virus	192
<a href="#">Tool&lt; TSeq &gt;</a>	
Tools for defending the agent against the virus	193
<a href="#">epiworld::Tools&lt; TSeq &gt;</a>	
Set of tools (useful for building iterators)	195
<a href="#">Tools&lt; TSeq &gt;</a>	
Set of tools (useful for building iterators)	196
<a href="#">epiworld::Tools_const&lt; TSeq &gt;</a>	
Set of <a href="#">Tools</a> (const) (useful for iterators)	197
<a href="#">Tools_const&lt; TSeq &gt;</a>	
Set of <a href="#">Tools</a> (const) (useful for iterators)	197
<a href="#">epiworld::UserData&lt; TSeq &gt;</a>	
Personalized data by the user	198
<a href="#">UserData&lt; TSeq &gt;</a>	
Personalized data by the user	200
<a href="#">epiworld::vecHasher&lt; T &gt;</a>	
Vector hasher	202
<a href="#">vecHasher&lt; T &gt;</a>	
Vector hasher	202
<a href="#">epiworld::Virus&lt; TSeq &gt;</a>	
Virus	203
<a href="#">Virus&lt; TSeq &gt;</a>	
Virus	205
<a href="#">epiworld::Viruses&lt; TSeq &gt;</a>	
Set of viruses (useful for building iterators)	207
<a href="#">Viruses&lt; TSeq &gt;</a>	
Set of viruses (useful for building iterators)	208
<a href="#">epiworld::Viruses_const&lt; TSeq &gt;</a>	
Set of <a href="#">Viruses</a> (const) (useful for iterators)	209
<a href="#">Viruses_const&lt; TSeq &gt;</a>	
Set of <a href="#">Viruses</a> (const) (useful for iterators)	209



## Chapter 13

# File Index

### 13.1 File List

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

<b>epiworld.hpp</b>	??
include/epiworld/ <b>adjlist-bones.hpp</b>	??
include/epiworld/ <b>adjlist-meat.hpp</b>	??
include/epiworld/ <b>agent-bones.hpp</b>	??
include/epiworld/ <b>agent-events-meat.hpp</b>	??
include/epiworld/ <b>agent-meat-state.hpp</b>	??
Sampling functions are getting big, so we keep them in a separate file	211
include/epiworld/ <b>agent-meat-virus-sampling.hpp</b>	??
include/epiworld/ <b>agent-meat.hpp</b>	??
include/epiworld/ <b>agentssample-bones.hpp</b>	??
include/epiworld/ <b>config.hpp</b>	??
include/epiworld/ <b>database-bones.hpp</b>	??
include/epiworld/ <b>database-meat.hpp</b>	??
include/epiworld/ <b>entities-bones.hpp</b>	??
include/epiworld/ <b>entity-bones.hpp</b>	??
include/epiworld/ <b>entity-distribute-meat.hpp</b>	??
include/epiworld/ <b>entity-meat.hpp</b>	??
include/epiworld/ <b>epiworld-macros.hpp</b>	??
include/epiworld/ <b>epiworld.hpp</b>	??
include/epiworld/ <b>globalevent-bones.hpp</b>	??
include/epiworld/ <b>globalevent-meat.hpp</b>	??
include/epiworld/ <b>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-distribute-meat.hpp</b>	??

include/epiworld/ <b>tool-meat.hpp</b>	??
include/epiworld/ <b>tools-bones.hpp</b>	??
include/epiworld/ <b>userdata-bones.hpp</b>	??
include/epiworld/ <b>userdata-meat.hpp</b>	??
include/epiworld/ <b>virus-bones.hpp</b>	??
include/epiworld/ <b>virus-distribute-meat.hpp</b>	??
include/epiworld/ <b>virus-meat.hpp</b>	??
include/epiworld/ <b>viruses-bones.hpp</b>	??
include/epiworld/math/ <b>distributions.hpp</b>	??
include/epiworld/math/ <b>lfmcmc.hpp</b>	??
include/epiworld/math/lfmcmc/ <b>lfmcmc-bones.hpp</b>	??
include/epiworld/math/lfmcmc/ <b>lfmcmc-meat-print.hpp</b>	??
include/epiworld/math/lfmcmc/ <b>lfmcmc-meat.hpp</b>	??
include/epiworld/models/ <b>diffnet.hpp</b>	??
include/epiworld/models/ <b>globalevents.hpp</b>	??
include/epiworld/models/ <b>init-functions.hpp</b>	??
include/epiworld/models/ <b>models.hpp</b>	??
include/epiworld/models/ <b>seir.hpp</b>	??
include/epiworld/models/ <b>seirconnected.hpp</b>	??
include/epiworld/models/ <b>seird.hpp</b>	??
include/epiworld/models/ <b>seirdconnected.hpp</b>	??
include/epiworld/models/ <b>seirmixing.hpp</b>	??
include/epiworld/models/ <b>sir.hpp</b>	??
include/epiworld/models/ <b>sirconnected.hpp</b>	??
include/epiworld/models/ <b>sird.hpp</b>	??
include/epiworld/models/ <b>sirdconnected.hpp</b>	??
include/epiworld/models/ <b>sirlogit.hpp</b>	??
include/epiworld/models/ <b>sirmixing.hpp</b>	??
include/epiworld/models/ <b>sis.hpp</b>	??
include/epiworld/models/ <b>sisd.hpp</b>	??
include/epiworld/models/ <b>surveillance.hpp</b>	??
tests/ <b>tests.hpp</b>	??

## Chapter 14

# Namespace Documentation

### 14.1 epiworld::sampler Namespace Reference

Functions for sampling viruses.

#### Functions

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

#### 14.1.1 Detailed Description

Functions for sampling viruses.

#### 14.1.2 Function Documentation

##### 14.1.2.1 `make_sample_virus_neighbors()`

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

Make a function to sample from neighbors.

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

## Template Parameters

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

## Parameters

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

## Returns

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

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

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

Make a function to sample from neighbors.

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

## Template Parameters

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

## Parameters

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

## Returns

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

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

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

Sample from neighbors pool of viruses (at most one)

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

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

## Template Parameters

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

## Parameters

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

## Returns

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

## 14.2 sampler Namespace Reference

Functions for sampling viruses.

### Functions

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

### 14.2.1 Detailed Description

Functions for sampling viruses.

### 14.2.2 Function Documentation

#### 14.2.2.1 `make_sample_virus_neighbors()`

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

Make a function to sample from neighbors.

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



## Template Parameters

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

## Parameters

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

## Returns

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

## 14.2.2.2 make\_update\_susceptible()

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

Make a function to sample from neighbors.

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

## Template Parameters

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

## Parameters

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

## Returns

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

## 14.2.2.3 sample\_virus\_single()

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

Sample from neighbors pool of viruses (at most one)

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

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

## Template Parameters

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

## Parameters

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

## Returns

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



## Chapter 15

# Class Documentation

### 15.1 AdjList Class Reference

#### Public Member Functions

- [AdjList](#) (const std::vector< int > &source, const std::vector< int > &target, int size, bool directed)  
*Construct a new Adj List object.*
- **AdjList** ([AdjList](#) &&a)
- **AdjList** (const [AdjList](#) &a)
- [AdjList](#) & **operator=** (const [AdjList](#) &a)
- void [read\\_edgelist](#) (std::string fn, int size, int skip=0, bool directed=true)  
*Read an edgelist.*
- std::map< int, int > **operator()** (epiworld\_fast\_uint i) const
- void **print** (epiworld\_fast\_uint limit=20u) const
- size\_t [vcount](#) () const  
*Number of vertices/nodes in the network.*
- size\_t [ecount](#) () const  
*Number of edges/arcs/ties in the network.*
- std::vector< std::map< int, int > > & **get\_dat** ()
- bool [is\\_directed](#) () const  
*true if the network is directed.*

#### 15.1.1 Constructor & Destructor Documentation

##### 15.1.1.1 AdjList()

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

Construct a new Adj List object.

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

## Parameters

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

## 15.1.2 Member Function Documentation

### 15.1.2.1 read\_edgelist()

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

Read an edgelist.

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

## Parameters

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

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

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

## 15.2 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.*

## 15.2.1 Constructor & Destructor Documentation

### 15.2.1.1 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

## 15.2.2 Member Function Documentation

### 15.2.2.1 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`

## 15.3 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** ()
- const VirusPtr< TSeq > & **get\_virus** () const
- ToolPtr< TSeq > & **get\_tool** (int i)
- [Tools](#)< TSeq > **get\_tools** ()
- const [Tools\\_const](#)< TSeq > **get\_tools** () const
- size\_t **get\_n\_tools** () const noexcept
- void **mutate\_virus** ()
- void **add\_neighbor** ([Agent](#)< TSeq > &p, bool check\_source=true, bool check\_target=true)
- void **swap\_neighbors** ([Agent](#)< TSeq > &other, size\_t n\_this, size\_t n\_other)  
*Swaps neighbors between the current agent and agent other*
- std::vector< [Agent](#)< TSeq > \* > **get\_neighbors** ()
- size\_t **get\_n\_neighbors** () const
- void **change\_state** ([Model](#)< TSeq > \*model, epiworld\_fast\_uint new\_state, epiworld\_fast\_int queue=0)
- const 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
- bool **has\_entity** (epiworld\_fast\_uint t) const
- bool **has\_entity** (std::string name) const
- void **print** ([Model](#)< TSeq > \*model, bool compressed=false) const
- [Entities](#)< TSeq > **get\_entities** ()
- const [Entities\\_const](#)< TSeq > **get\_entities** () const



- const [Entity](#)< TSeq > & **get\_entity** (size\_t i) const
- [Entity](#)< TSeq > & **get\_entity** (size\_t i)
- size\_t **get\_n\_entities** () const
- bool **operator==** (const [Agent](#)< TSeq > &other) const
- bool **operator!=** (const [Agent](#)< TSeq > &other) const

### Add/Remove Virus/Tool

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

#### Parameters

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

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

[Agent](#) removed by virus.

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

#### Parameters

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

#### Returns

*epiworld\_double*

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

- double & [operator\(\)](#) (size\_t j)  
Access the j-th column of the agent.
- double & **operator[]** (size\_t j)
- double **operator()** (size\_t j) const
- double **operator[]** (size\_t j) const

## Friends

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

## 15.3.1 Detailed Description

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

[Agent](#) (agents)

Template Parameters

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

## 15.3.2 Member Function Documentation

### 15.3.2.1 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;

**15.3.2.2 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>	

**15.3.3 Friends And Related Function Documentation****15.3.3.1 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

## 15.4 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** ()
- const VirusPtr< TSeq > & **get\_virus** () const
- ToolPtr< TSeq > & **get\_tool** (int i)
- [Tools](#)< TSeq > **get\_tools** ()
- const [Tools\\_const](#)< TSeq > **get\_tools** () const
- size\_t **get\_n\_tools** () const noexcept
- void **mutate\_virus** ()
- void **add\_neighbor** ([Agent](#)< TSeq > &p, bool check\_source=true, bool check\_target=true)
- void **swap\_neighbors** ([Agent](#)< TSeq > &other, size\_t n\_this, size\_t n\_other)  
*Swaps neighbors between the current agent and agent other*
- std::vector< [Agent](#)< TSeq > \* > **get\_neighbors** ()
- size\_t **get\_n\_neighbors** () const
- void **change\_state** ([Model](#)< TSeq > \*model, epiworld\_fast\_uint new\_state, epiworld\_fast\_int queue=0)
- const 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
- bool **has\_entity** (epiworld\_fast\_uint t) const
- bool **has\_entity** (std::string name) const
- void **print** ([Model](#)< TSeq > \*model, bool compressed=false) const
- [Entities](#)< TSeq > **get\_entities** ()
- const [Entities\\_const](#)< TSeq > **get\_entities** () const
- const [Entity](#)< TSeq > & **get\_entity** (size\_t i) const
- [Entity](#)< TSeq > & **get\_entity** (size\_t i)
- size\_t **get\_n\_entities** () const
- bool **operator==** (const [Agent](#)< TSeq > &other) const
- bool **operator!=** (const [Agent](#)< TSeq > &other) const

### Add/Remove Virus/Tool

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

#### Parameters

tool	<a href="#">Tool</a> to add
virus	<a href="#">Virus</a> to add
state_new	state after the change
Generated by Doxygen	

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

[Agent](#) removed by virus.

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

#### Parameters

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

#### Returns

*epiworld\_double*

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

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

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

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

### Friends

- class **Model**< TSeq >
- class **Virus**< TSeq >
- class **Tool**< TSeq >
- class **Tools**< TSeq >
- class **Tools\_const**< TSeq >
- class **Queue**< TSeq >
- class **Entities**< TSeq >

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

### 15.4.1 Detailed Description

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

[Agent](#) (agents)

Template Parameters

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

### 15.4.2 Member Function Documentation

#### 15.4.2.1 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&

### 15.4.2.2 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>	

## 15.4.3 Friends And Related Function Documentation

### 15.4.3.1 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

## 15.5 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

### 15.5.1 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>	
-------------	--

### 15.5.2 Constructor & Destructor Documentation

#### 15.5.2.1 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

## 15.6 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

### 15.6.1 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>	
-------------	--

## 15.6.2 Constructor &amp; Destructor Documentation

## 15.6.2.1 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

## 15.7 DataBase&lt; TSeq &gt; Class Template Reference

Statistical data about the process.

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

## Public Member Functions

- **DataBase** ([Model](#)< TSeq > &m)
- **DataBase** (const [DataBase](#)< TSeq > &db)
- void [record\\_virus](#) ([Virus](#)< TSeq > &v)
  - Registering a new variant.*
- void [record\\_tool](#) ([Tool](#)< TSeq > &t)
- void [set\\_seq\\_hasher](#) (std::function< std::vector< int >(TSeq)> fun)
- void [reset](#) ()
- [Model](#)< TSeq > \* [get\\_model](#) ()
- void [record](#) ()
- const std::vector< TSeq > & [get\\_sequence](#) () const
- const std::vector< int > & [get\\_nexposed](#) () const
- size\_t [size](#) () const
- void [write\\_data](#) (std::string fn\_virus\_info, std::string fn\_virus\_hist, std::string fn\_tool\_info, std::string fn\_↵  
\_tool\_hist, std::string fn\_total\_hist, std::string fn\_transmission, std::string fn\_transition, std::string fn\_↵  
reproductive\_number, std::string fn\_generation\_time) const
- void [record\\_transmission](#) (int i, int j, int virus, int i\_expo\_date)
- size\_t [get\\_n\\_viruses](#) () const
  - Get the number of viruses.*
- size\_t [get\\_n\\_tools](#) () const
  - Get the number of tools.*
- void [set\\_user\\_data](#) (std::vector< std::string > names)
- void [add\\_user\\_data](#) (std::vector< epiworld\_double > x)
- void [add\\_user\\_data](#) (epiworld\_fast\_uint j, epiworld\_double x)
- [UserData](#)< TSeq > & [get\\_user\\_data](#) ()
- std::vector< epiworld\_double > [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\\_today\\_transition\\_matrix](#) (std::vector< int > &counts) const
- void [get\\_hist\\_total](#) (std::vector< int > \*date, std::vector< std::string > \*state, std::vector< int > \*counts) const

- void **get\_hist\_virus** (std::vector< int > &date, std::vector< int > &id, std::vector< std::string > &state, std::vector< int > &counts) const
  - void **get\_hist\_tool** (std::vector< int > &date, std::vector< int > &id, std::vector< std::string > &state, std::vector< int > &counts) const
  - void **get\_hist\_transition\_matrix** (std::vector< std::string > &state\_from, std::vector< std::string > &state\_to, std::vector< int > &date, std::vector< int > &counts, bool skip\_zeros) const
- 
- void **get\_transmissions** (std::vector< int > &date, std::vector< int > &source, std::vector< int > &target, std::vector< int > &virus, std::vector< int > &source\_exposure\_date) const  
*Get the transmissions object.*
  - void **get\_transmissions** (int \*date, int \*source, int \*target, int \*virus, int \*source\_exposure\_date) const
- 
- MapVec\_type< int, int > **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  
*Get the generation time.*
  - void **generation\_time** (std::string fn) const  
*Write the generation time to a file.*

## Friends

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

### 15.7.1 Detailed Description

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

Statistical data about the process.

#### Template Parameters

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

## 15.7.2 Member Function Documentation

### 15.7.2.1 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]
```

Get the generation time.

Calculates the generating time

#### Parameters

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

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

### 15.7.2.2 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>	

### 15.7.2.3 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,

#### 15.7.2.4 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,

#### 15.7.2.5 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

#### 15.7.2.6 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.
----------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**15.7.2.7 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.
-----------	--------------------------------------------------

**15.7.2.8 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

**15.8 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  
*Get the number of viruses.*
- size\_t [get\\_n\\_tools](#) () const  
*Get the number of tools.*
- void [set\\_user\\_data](#) (std::vector< std::string > names)
- void [add\\_user\\_data](#) (std::vector< epiworld\_double > x)
- void [add\\_user\\_data](#) (epiworld\_fast\_uint j, epiworld\_double x)
- [UserData](#)< TSeq > & [get\\_user\\_data](#) ()
- std::vector< epiworld\_double > [transition\\_probability](#) (bool print=true) const  
*Calculates the transition probabilities.*
- bool [operator==](#) (const [DataBase](#)< TSeq > &other) const
- bool [operator!=](#) (const [DataBase](#)< TSeq > &other) const

### Get recorded information from the model

#### Parameters

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

#### Returns

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

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

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

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

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

- int [get\\_today\\_total](#) (std::string what) const
- int [get\\_today\\_total](#) (epiworld\_fast\_uint what) const
- void [get\\_today\\_total](#) (std::vector< std::string > \*state=nullptr, std::vector< int > \*counts=nullptr) const
- void [get\\_today\\_virus](#) (std::vector< std::string > &state, std::vector< int > &id, std::vector< int > &counts) const
- void [get\\_today\\_transition\\_matrix](#) (std::vector< int > &counts) const
- void [get\\_hist\\_total](#) (std::vector< int > \*date, std::vector< std::string > \*state, std::vector< int > \*counts) const
- void [get\\_hist\\_virus](#) (std::vector< int > &date, std::vector< int > &id, std::vector< std::string > &state, std::vector< int > &counts) const

- void **get\_hist\_tool** (std::vector< int > &date, std::vector< int > &id, std::vector< std::string > &state, std::vector< int > &counts) const
  - void **get\_hist\_transition\_matrix** (std::vector< std::string > &state\_from, std::vector< std::string > &state\_to, std::vector< int > &date, std::vector< int > &counts, bool skip\_zeros) const
- 
- void **get\_transmissions** (std::vector< int > &date, std::vector< int > &source, std::vector< int > &target, std::vector< int > &virus, std::vector< int > &source\_exposure\_date) const  
*Get the transmissions object.*
  - void **get\_transmissions** (int \*date, int \*source, int \*target, int \*virus, int \*source\_exposure\_date) const
- 
- MapVec\_type< int, int > **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  
*Get the generation time.*
  - void **generation\_time** (std::string fn) const  
*Write the generation time to a file.*

## Friends

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

### 15.8.1 Detailed Description

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

Statistical data about the process.

#### Template Parameters

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

## 15.8.2 Member Function Documentation

### 15.8.2.1 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]
```

Get the generation time.

Calculates the generating time

#### Parameters

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

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

### 15.8.2.2 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>	

### 15.8.2.3 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

#### 15.8.2.4 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.
----------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

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

##### Parameters

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

#### 15.8.2.6 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

## 15.9 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 >

### 15.9.1 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

## 15.10 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 >

### 15.10.1 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

## 15.11 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 >

### 15.11.1 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

## 15.12 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 >

### 15.12.1 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

## 15.13 Entity< TSeq > Class Template Reference

### Public Member Functions

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

### Entity distribution

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

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



## Friends

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

## 15.13.1 Constructor & Destructor Documentation

### 15.13.1.1 Entity()

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

Constructs an [Entity](#) object.

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

#### Parameters

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

## 15.13.2 Friends And Related Function Documentation

### 15.13.2.1 default\_rm\_entity

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

< Last entity of the agent

< Last agent of the entity

< Last entity of the agent

< Last agent of the entity

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

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

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

### Public Member Functions

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

### Entity distribution

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

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

### Friends

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

## 15.14.1 Constructor & Destructor Documentation

### 15.14.1.1 Entity()

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

Constructs an [Entity](#) object.

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

## Parameters

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

## 15.14.2 Friends And Related Function Documentation

### 15.14.2.1 default\_rm\_entity

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

< Last entity of the agent

< Last agent of the entity

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

- epiworld.hpp

## 15.15 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\_, EventFun< TSeq > call\_, int idx\_agent\_, int idx\_object\_)  
Construct a new Event object.

### Public Attributes

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

### 15.15.1 Detailed Description

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

Event data for update an agent.

## Template Parameters

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

## 15.15.2 Constructor &amp; Destructor Documentation

## 15.15.2.1 Event()

```
template<typename TSeq = EPI_DEFAULT_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_,
    EventFun< TSeq > call_,
    int idx_agent_,
    int idx_object_ ) [inline]
```

Construct a new [Event](#) object.

All the parameters are rather optional.

## Parameters

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

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

- epiworld.hpp

## 15.16 Event&lt; TSeq &gt; Struct Template Reference

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

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

Collaboration diagram for Event< TSeq >:



## Public Member Functions

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

Construct a new [Event](#) object.

## Public Attributes

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

### 15.16.1 Detailed Description

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

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

#### Template Parameters

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

## 15.16.2 Constructor & Destructor Documentation

### 15.16.2.1 Event()

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

Construct a new [Event](#) object.

All the parameters are rather optional.

#### Parameters

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

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

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

## 15.17 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

### 15.17.1 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.

### 15.17.2 Constructor & Destructor Documentation

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

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

- epiworld.hpp

## 15.18 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

### 15.18.1 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.

### 15.18.2 Constructor & Destructor Documentation

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



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

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

## 15.19 epiworld::GroupSampler< TSeq > Class Template Reference

Weighted sampling of groups.

```
#include <epiworld.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)

### 15.19.1 Detailed Description

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

Weighted sampling of groups.

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

- epiworld.hpp

## 15.20 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)

### 15.20.1 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

## 15.21 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\_, int seed=-1)
- **LFMCMC** (const TData &observed\_data\_)
- void **set\_observed\_data** (const TData &observed\_data\_)
- void **set\_proposal\_fun** (LFMCMCProposalFun< TData > fun)
- void **set\_simulation\_fun** (LFMCMCSimFun< TData > fun)
- void **set\_summary\_fun** (LFMCMCSummaryFun< TData > fun)
- void **set\_kernel\_fun** (LFMCMCKernelFun< TData > fun)
- 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::shared\_ptr< std::mt19937 > &eng)
- std::shared\_ptr< std::mt19937 > & **get\_rand\_engine** ()
- void **seed** (epiworld\_fast\_uint s)
- void **set\_rand\_gamma** (epiworld\_double alpha, epiworld\_double beta)
- epiworld\_double **runif** ()
- epiworld\_double **rnorm** ()
- epiworld\_double **rgamma** ()
- epiworld\_double **runif** (epiworld\_double lb, epiworld\_double ub)
- epiworld\_double **rnorm** (epiworld\_double mean, epiworld\_double sd)
- epiworld\_double **rgamma** (epiworld\_double alpha, epiworld\_double beta)

**15.21.1 Detailed Description**

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

Likelihood-Free Markov Chain Monte Carlo.

*Template Parameters*

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

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

- epiworld.hpp

**15.22 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\_, int seed=-1)
- **LFMCMC** (const TData &observed\_data\_)
- void **set\_observed\_data** (const TData &observed\_data\_)
- void **set\_proposal\_fun** (LFMCMCProposalFun< TData > fun)
- void **set\_simulation\_fun** (LFMCMCSimFun< TData > fun)
- void **set\_summary\_fun** (LFMCMCSummaryFun< TData > fun)
- void **set\_kernel\_fun** (LFMCMCKernelFun< TData > fun)
- 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::shared_ptr< std::mt19937 >` &eng)
- `std::shared_ptr< std::mt19937 >` & **get\_rand\_engine** ()
- void **seed** (`epiworld_fast_uint` s)
- void **set\_rand\_gamma** (`epiworld_double` alpha, `epiworld_double` beta)
- `epiworld_double` **runif** ()
- `epiworld_double` **rnorm** ()
- `epiworld_double` **rgamma** ()
- `epiworld_double` **runif** (`epiworld_double` lb, `epiworld_double` ub)
- `epiworld_double` **rnorm** (`epiworld_double` mean, `epiworld_double` sd)
- `epiworld_double` **rgamma** (`epiworld_double` alpha, `epiworld_double` beta)

## 15.22.1 Detailed Description

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

Likelihood-Free Markov Chain Monte Carlo.

#### Template Parameters

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

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

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

## 15.23 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 ()`
- `const DataBase< TSeq > & get_db () const`
- `epiworld_double & operator() (std::string pname)`
- `size_t size () const`
- `void load_agents_entities_ties (std::string fn, int skip)`  
*Associate agents-entities from a file.*
- `void load_agents_entities_ties (const std::vector< int > &agents_ids, const std::vector< int > &entities_ids)`  
*Associate agents-entities from data.*
- `void load_agents_entities_ties (const int *agents_id, const int *entities_id, size_t n)`
- `size_t get_n_viruses () const`  
*Number of viruses in the model.*
- `size_t get_n_tools () const`  
*Number of tools in the model.*
- `epiworld_fast_uint get_ndays () const`
- `epiworld_fast_uint get_n_replicates () const`
- `void set_ndays (epiworld_fast_uint ndays)`
- `bool get_verbose () const`
- `Model< TSeq > & verbose_off ()`
- `Model< TSeq > & verbose_on ()`
- `int today () const`  
*The current time of the model.*
- `void write_data (std::string fn_virus_info, std::string fn_virus_hist, std::string fn_tool_info, std::string fn_tool_hist, std::string fn_total_hist, std::string fn_transmission, std::string fn_transition, std::string fn_reproductive_number, std::string fn_generation_time) const`  
*Wrapper of DataBase::write\_data*
- `std::map< std::string, epiworld_double > & params ()`
- `virtual void reset ()`

*Reset the model.*

- const **Model**< TSeq > & **print** (bool lite=false) const
- void **get\_elapsed** (std::string unit="auto", epiworld\_double \*last\_elapsed=nullptr, epiworld\_double \*total\_elapsed=nullptr, std::string \*unit\_abbr=nullptr, bool print=true) const
- void **add\_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< **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::shared\_ptr< std::mt19937 > &eng)
- std::shared\_ptr< std::mt19937 > & **get\_rand\_engine** ()
- void **seed** (size\_t s)
- void **set\_rand\_norm** (epiworld\_double mean, epiworld\_double sd)
- void **set\_rand\_unif** (epiworld\_double a, epiworld\_double b)

- void **set\_rand\_exp** (epiworld\_double lambda)
- void **set\_rand\_gamma** (epiworld\_double alpha, epiworld\_double beta)
- void **set\_rand\_lognormal** (epiworld\_double mean, epiworld\_double shape)
- void **set\_rand\_binom** (int n, epiworld\_double p)
- 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	<a href="#">Virus</a> to be added
t	<a href="#">Tool</a> to be added
preval	Initial prevalence (initial state.) It can be specified as a proportion (between zero and one,) or an integer indicating number of individuals.

- void **add\_virus** ([Virus](#)< TSeq > &v)
- void **add\_tool** ([Tool](#)< TSeq > &t)
- void **add\_entity** ([Entity](#)< TSeq > e)
- void **rm\_virus** (size\_t virus\_pos)
- void **rm\_tool** (size\_t tool\_pos)
- void **rm\_entity** (size\_t entity\_id)

### Accessing population of the model

#### Parameters

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

- void **agents\_from\_adjlist** (std::string fn, int size, int skip=0, bool directed=false)
- void **agents\_from\_edgelist** (const std::vector< int > &source, const std::vector< int > &target, int size, bool directed)
- void **agents\_from\_adjlist** ([AdjList](#) al)
- bool **is\_directed** () const
- std::vector< [Agent](#)< TSeq > > &**get\_agents** ()  
*Returns a reference to the vector of agents.*
- [Agent](#)< TSeq > &**get\_agent** (size\_t i)
- std::vector< epiworld\_fast\_uint > **get\_agents\_states** () const  
*Returns a vector with the states of the agents.*
- std::vector< [Viruses\\_const](#)< TSeq > > **get\_agents\_viruses** () const

- Returns a const vector with the viruses of the agents.*
- `std::vector< Viruses< TSeq > > get_agents_viruses ()`
- Returns a vector with the viruses of the agents.*
- `std::vector< Entity< TSeq > > & get_entities ()`
- `Entity< TSeq > & get_entity (size_t entity_id, int *entity_pos=nullptr)`
- `Model< TSeq > & agents_smallworld (epiworld_fast_uint n=1000, epiworld_fast_uint k=5, bool d=false, epiworld_double p=.01)`
- `void agents_empty_graph (epiworld_fast_uint n=1000)`

### Functions to run the model

#### Parameters

seed	<i>Seed to be used for Pseudo-RNG.</i>
ndays	<i>Number of days (steps) of the simulation.</i>
fun	<i>In the case of <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	<i>Proportion of ties to be rewired.</i>
------------	------------------------------------------

#### Returns

*A rewired version of the network.*

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

### Export the network data in edgelist form

#### Parameters

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

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



- void **write\_edgelist** (std::string fn) const
- void **write\_edgelist** (std::vector< 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) `const`

### Set the user data object

#### Parameters

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

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

### Queuing system

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

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

### Get the susceptibility reduction object

#### Parameters

v	
---	--

### Returns

*epiworld\_double*

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

### Protected Member Functions

- `void` **dist\_tools** ()
- `void` **dist\_virus** ()

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

Construct a new [Event](#) object.

## Protected Attributes

- std::string **name** = ""  
*Name of the model.*
- [DataBase](#)< TSeq > **db** = [DataBase](#)<TSeq>(\*this)
- std::vector< [Agent](#)< TSeq > > **population** = {}
- bool **using\_backup** = true
- std::vector< [Agent](#)< TSeq > > **population\_backup** = {}
- bool **directed** = false
- std::vector< [VirusPtr](#)< TSeq > > **viruses** = {}
- std::vector< [ToolPtr](#)< TSeq > > **tools** = {}
- std::vector< [Entity](#)< TSeq > > **entities** = {}
- std::vector< [Entity](#)< TSeq > > **entities\_backup** = {}
- std::shared\_ptr< std::mt19937 > **engine** = std::make\_shared< std::mt19937 >()
- std::uniform\_real\_distribution **runifd**
- std::normal\_distribution **rnormd**
- std::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** ()

## 15.23.1 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.)
-------------	-----------------------------------------------------------------------------------------------------------------------------------------

## 15.23.2 Member Function Documentation

## 15.23.2.1 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.

## 15.23.2.2 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>	
-------------	--

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

### 15.23.2.3 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_,
    EventFun< TSeq > call_,
    int idx_agent_,
    int idx_object_ ) [inline], [protected]
```

Construct a new [Event](#) object.

#### Parameters

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

### 15.23.2.4 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.
----------------------	-------------------------------------------------------

### 15.23.2.5 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.

### 15.23.2.6 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

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

### 15.23.2.7 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
--------------	---------------------------------

**15.23.2.8 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.

**15.23.2.9 set\_name()**

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

Set the name object.

## Parameters

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

**15.23.2.10 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

## 15.23.3 Member Data Documentation

### 15.23.3.1 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

### 15.23.3.2 rbinomd

```

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

```

#### Initial value:

```

=
    std::binomial_distribution<>()

```

### 15.23.3.3 rexp

```

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

```

#### Initial value:

```

=
    std::exponential_distribution<>()

```

### 15.23.3.4 rgammad

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

**Initial value:**

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

### 15.23.3.5 rlognormald

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

**Initial value:**

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

### 15.23.3.6 rnormd

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

**Initial value:**

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

### 15.23.3.7 runifd

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

**Initial value:**

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

### 15.23.3.8 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](#)

## 15.24 Model< TSeq > Class Template Reference

Core class of epiworld.

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

Collaboration diagram for Model< TSeq >:



### Public Member Functions

- [DataBase](#)< TSeq > & **get\_db** ()
- const [DataBase](#)< TSeq > & **get\_db** () const
- epiworld\_double & **operator()** (std::string pname)
- size\_t **size** () const
- void [load\\_agents\\_entities\\_ties](#) (std::string fn, int skip)  
*Associate agents-entities from a file.*
- void [load\\_agents\\_entities\\_ties](#) (const std::vector< int > &agents\_ids, const std::vector< int > &entities\_ids)  
*Associate agents-entities from data.*
- void **load\_agents\_entities\_ties** (const int \*agents\_id, const int \*entities\_id, size\_t n)
- size\_t [get\\_n\\_viruses](#) () const  
*Number of viruses in the model.*
- size\_t [get\\_n\\_tools](#) () const  
*Number of tools in the model.*
- epiworld\_fast\_uint **get\_ndays** () const
- epiworld\_fast\_uint **get\_n\_replicates** () const
- void **set\_ndays** (epiworld\_fast\_uint ndays)
- bool **get\_verbose** () const
- [Model](#)< TSeq > & **verbose\_off** ()
- [Model](#)< TSeq > & **verbose\_on** ()
- int [today](#) () const  
*The current time of the model.*
- void [write\\_data](#) (std::string fn\_virus\_info, std::string fn\_virus\_hist, std::string fn\_tool\_info, std::string fn↵  
\_tool\_hist, std::string fn\_total\_hist, std::string fn\_transmission, std::string fn\_transition, std::string fn\_↵  
reproductive\_number, std::string fn\_generation\_time) const  
*Wrapper of DataBase::write\_data*
- std::map< std::string, epiworld\_double > & **params** ()
- virtual void [reset](#) ()

*Reset the model.*

- const **Model**< TSeq > & **print** (bool lite=false) const
- void **get\_elapsed** (std::string unit="auto", epiworld\_double \*last\_elapsed=nullptr, epiworld\_double \*total\_elapsed=nullptr, std::string \*unit\_abbr=nullptr, bool print=true) const
- void **add\_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< 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::shared\_ptr< std::mt19937 > &eng)
- std::shared\_ptr< std::mt19937 > & **get\_rand\_engine** ()
- void **seed** (size\_t s)
- void **set\_rand\_norm** (epiworld\_double mean, epiworld\_double sd)
- void **set\_rand\_unif** (epiworld\_double a, epiworld\_double b)

- void **set\_rand\_exp** (epiworld\_double lambda)
- void **set\_rand\_gamma** (epiworld\_double alpha, epiworld\_double beta)
- void **set\_rand\_lognormal** (epiworld\_double mean, epiworld\_double shape)
- void **set\_rand\_binom** (int n, epiworld\_double p)
- 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	<a href="#">Virus</a> to be added
t	<a href="#">Tool</a> to be added
preval	Initial prevalence (initial state.) It can be specified as a proportion (between zero and one,) or an integer indicating number of individuals.

- void **add\_virus** ([Virus](#)< TSeq > &v)
- void **add\_tool** ([Tool](#)< TSeq > &t)
- void **add\_entity** ([Entity](#)< TSeq > e)
- void **rm\_virus** (size\_t virus\_pos)
- void **rm\_tool** (size\_t tool\_pos)
- void **rm\_entity** (size\_t entity\_id)

### Accessing population of the model

#### Parameters

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

- void **agents\_from\_adjlist** (std::string fn, int size, int skip=0, bool directed=false)
- void **agents\_from\_edgelist** (const std::vector< int > &source, const std::vector< int > &target, int size, bool directed)
- void **agents\_from\_adjlist** ([AdjList](#) al)
- bool **is\_directed** () const
- std::vector< [Agent](#)< TSeq > > &**get\_agents** ()  
*Returns a reference to the vector of agents.*
- [Agent](#)< TSeq > &**get\_agent** (size\_t i)
- std::vector< epiworld\_fast\_uint > **get\_agents\_states** () const  
*Returns a vector with the states of the agents.*
- std::vector< [Viruses\\_const](#)< TSeq > > **get\_agents\_viruses** () const

- Returns a const vector with the viruses of the agents.*
- `std::vector< Viruses< TSeq > > get_agents_viruses ()`
- Returns a vector with the viruses of the agents.*
- `std::vector< Entity< TSeq > > & get_entities ()`
- `Entity< TSeq > & get_entity (size_t entity_id, int *entity_pos=nullptr)`
- `Model< TSeq > & agents_smallworld (epiworld_fast_uint n=1000, epiworld_fast_uint k=5, bool d=false, epiworld_double p=.01)`
- `void agents_empty_graph (epiworld_fast_uint n=1000)`

### Functions to run the model

#### Parameters

seed	<i>Seed to be used for Pseudo-RNG.</i>
ndays	<i>Number of days (steps) of the simulation.</i>
fun	<i>In the case of <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	<i>Proportion of ties to be rewired.</i>
------------	------------------------------------------

#### Returns

*A rewired version of the network.*

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

### Export the network data in edgelist form

#### Parameters

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

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

- void **write\_edgelist** (std::string fn) const
- void **write\_edgelist** (std::vector< 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) `const`

### Set the user data object

#### Parameters

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

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

### Queuing system

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

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

### Get the susceptibility reduction object

#### Parameters

v	
---	--

### Returns

*epiworld\_double*

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

### Protected Member Functions

- `void` **dist\_tools** ()
- `void` **dist\_virus** ()



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

Construct a new [Event](#) object.

## Protected Attributes

- std::string **name** = ""  
*Name of the model.*
- DataBase< TSeq > **db** = DataBase<TSeq>(\*this)
- std::vector< Agent< TSeq > > **population** = {}
- bool **using\_backup** = true
- std::vector< Agent< TSeq > > **population\_backup** = {}
- bool **directed** = false
- std::vector< VirusPtr< TSeq > > **viruses** = {}
- std::vector< ToolPtr< TSeq > > **tools** = {}
- std::vector< Entity< TSeq > > **entities** = {}
- std::vector< Entity< TSeq > > **entities\_backup** = {}
- std::shared\_ptr< std::mt19937 > **engine** = std::make\_shared< std::mt19937 >()
- std::uniform\_real\_distribution **runifd**
- std::normal\_distribution **rnormd**
- std::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\_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::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** ()

## 15.24.1 Detailed Description

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

Core class of epiworld.

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

## Template Parameters

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

## 15.24.2 Member Function Documentation

## 15.24.2.1 add\_globalevent()

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

Set a global action.

## Parameters

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

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

## 15.24.2.2 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>	
-------------	--

## 15.24.2.3 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_,
EventFun< TSeq > call_,
int idx_agent_,
int idx_object_ ) [inline], [protected]

```

Construct a new [Event](#) object.

#### Parameters

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

#### 15.24.2.4 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.
----------------------	-------------------------------------------------------

#### 15.24.2.5 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.

**15.24.2.6 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

**15.24.2.7 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
--------------	---------------------------------

**15.24.2.8 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.

#### 15.24.2.9 set\_name()

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

Set the name object.

#### Parameters

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

#### 15.24.2.10 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.

## Parameters

<i>fn_tool_hist</i>	Filename. History of the tool.
<i>fn_total_hist</i>	Filename. Aggregated history (state)
<i>fn_transmission</i>	Filename. Transmission history.
<i>fn_transition</i>	Filename. Markov transition history.
<i>fn_reproductive_number</i>	Filename. Case by case reproductive number

## 15.24.3 Member Data Documentation

## 15.24.3.1 initial\_states\_fun

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

## Initial value:

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

Function to distribute states. Goes along with the function

## 15.24.3.2 rbinomd

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

## Initial value:

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

## 15.24.3.3 rexp

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

## Initial value:

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

#### 15.24.3.4 rgammad

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

##### Initial value:

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

#### 15.24.3.5 rlognormald

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

##### Initial value:

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

#### 15.24.3.6 rnormd

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

##### Initial value:

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

#### 15.24.3.7 runifd

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

##### Initial value:

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

#### 15.24.3.8 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



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

## Public Attributes

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

## Static Public Attributes

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

## Additional Inherited Members

### 15.25.1 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

## 15.26 ModelDiffNet< TSeq > Class Template Reference

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

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

Inheritance diagram for ModelDiffNet< TSeq >:



Collaboration diagram for ModelDiffNet< TSeq >:



## Public Member Functions

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

## Public Attributes

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

## Static Public Attributes

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

## Additional Inherited Members

### 15.26.1 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_efficiency</i>	epiworld_double Initial susceptibility_reduction of the immune system
<i>initial_recovery</i>	epiworld_double Initial recovery rate of the immune system

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

- include/epiworld/models/diffnet.hpp

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

## Public Attributes

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

## Static Public Attributes

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

## Additional Inherited Members

### 15.27.1 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.

## 15.27.2 Member Function Documentation

### 15.27.2.1 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 >](#).

## 15.27.3 Member Data Documentation

### 15.27.3.1 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;
}
```

## 15.27.3.2 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

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

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

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

Inheritance diagram for ModelSEIR< TSeq >:



Collaboration diagram for ModelSEIR< TSeq >:



## Public Member Functions

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

## Public Attributes

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

## Static Public Attributes

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

## Additional Inherited Members

### 15.28.1 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.

### 15.28.2 Member Function Documentation



### 15.28.2.1 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 >](#).

## 15.28.3 Member Data Documentation

### 15.28.3.1 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;
}
```

### 15.28.3.2 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

## 15.29 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, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate)  
*Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.*
- **ModelSEIRCONN** (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate)
- [ModelSEIRCONN](#)< TSeq > & [run](#) (epiworld\_fast\_uint ndays, int seed=-1)  
*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`
- `std::vector< double > generation_time_expected (int max_days=200, int max_contacts=200) const`

## Static Public Attributes

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

## Additional Inherited Members

### 15.29.1 Constructor & Destructor Documentation

#### 15.29.1.1 ModelSEIRCONN()

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

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

#### Parameters

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

### 15.29.2 Member Function Documentation

### 15.29.2.1 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

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

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

### 15.29.2.2 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 >](#).

### 15.29.2.3 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:

- `epiworld.hpp`

## 15.30 ModelSEIRCONN< TSeq > Class Template Reference

Inheritance diagram for ModelSEIRCONN< TSeq >:



Collaboration diagram for ModelSEIRCONN< TSeq >:



### Public Member Functions

- [ModelSEIRCONN](#) ([ModelSEIRCONN](#)< TSeq > &model, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate)  
*Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.*
- **ModelSEIRCONN** (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate)
- [ModelSEIRCONN](#)< TSeq > & [run](#) (epiworld\_fast\_uint ndays, int seed=-1)  
*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
- `std::vector< double > generation_time_expected` (`int max_days=200`, `int max_contacts=200`) const

## Static Public Attributes

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

## Additional Inherited Members

### 15.30.1 Constructor & Destructor Documentation

#### 15.30.1.1 ModelSEIRCONN()

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

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

#### Parameters

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

### 15.30.2 Member Function Documentation

### 15.30.2.1 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 >](#).

### 15.30.2.2 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 >](#).

### 15.30.2.3 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`

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



## Public Attributes

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

## Static Public Attributes

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

## Additional Inherited Members

### 15.31.1 Detailed Description

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

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

### 15.31.2 Constructor & Destructor Documentation

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

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

Constructor for the SEIRD model.

#### Template Parameters

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

#### Parameters

<i>model</i>	Reference to the SEIRD model.
<i>vname</i>	Name of the model.

## 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.

## 15.31.2.2 ModelSEIRD() [2/2]

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

Constructor for the SEIRD model.

## Parameters

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

## 15.31.3 Member Data Documentation

## 15.31.3.1 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`

## 15.32 ModelSEIRD< TSeq > Class Template Reference

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

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

Inheritance diagram for ModelSEIRD< TSeq >:



Collaboration diagram for ModelSEIRD< TSeq >:



### Public Member Functions

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

## Public Attributes

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

## Static Public Attributes

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

## Additional Inherited Members

### 15.32.1 Detailed Description

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

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

### 15.32.2 Constructor & Destructor Documentation

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

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

Constructor for the SEIRD model.

#### Template Parameters

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

#### Parameters

<i>model</i>	Reference to the SEIRD model.
<i>vname</i>	Name of the model.

## 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.

## 15.32.2.2 ModelSEIRD() [2/2]

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

Constructor for the SEIRD model.

## Parameters

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

## 15.32.3 Member Data Documentation

## 15.32.3.1 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`

### 15.33 `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, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, epiworld\_double death\_rate)  
*Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.*
- **ModelSEIRDCONN** (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, epiworld\_double death\_rate)
- [ModelSEIRDCONN](#)< TSeq > & [run](#) (epiworld\_fast\_uint ndays, int seed=-1)

- 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

### 15.33.1 Constructor & Destructor Documentation

#### 15.33.1.1 `ModelSEIRDCONN()`

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

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

#### Parameters

<i>model</i>	A <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
<i>death_rate</i>	Probability of death

## 15.33.2 Member Function Documentation

### 15.33.2.1 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 >](#).

### 15.33.2.2 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 >](#).

### 15.33.2.3 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:

- `epiworld.hpp`

## 15.34 ModelSEIRDCONN< TSeq > Class Template Reference

Inheritance diagram for ModelSEIRDCONN< TSeq >:



Collaboration diagram for ModelSEIRDCONN< TSeq >:



## Public Member Functions

- [ModelSEIRDCONN](#) ([ModelSEIRDCONN](#)< TSeq > &model, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, epiworld\_double death\_rate)  
*Template for a Susceptible-Exposed-Infected-Removed (SEIR) model.*
- **ModelSEIRDCONN** (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, epiworld\_double death\_rate)
- [ModelSEIRDCONN](#)< TSeq > & [run](#) (epiworld\_fast\_uint ndays, int seed=-1)  
*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

### 15.34.1 Constructor & Destructor Documentation

#### 15.34.1.1 ModelSEIRDCONN()

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

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

## Parameters

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

## 15.34.2 Member Function Documentation

## 15.34.2.1 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 >](#).

## 15.34.2.2 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 >](#).

### 15.34.2.3 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`

## 15.35 epiworld::epimodels::ModelSEIRMixing< TSeq > Class Template Reference

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



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



## Public Member Functions

- [ModelSEIRMixing](#) ([ModelSEIRMixing](#)< TSeq > &model, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, std::vector< double > contact\_matrix)  
*Constructs a [ModelSEIRMixing](#) object.*
- [ModelSEIRMixing](#) (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, std::vector< double > contact\_matrix)  
*Constructs a [ModelSEIRMixing](#) object.*
- [ModelSEIRMixing](#)< TSeq > &run (epiworld\_fast\_uint ndays, int seed=-1)  
*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

### 15.35.1 Constructor & Destructor Documentation

### 15.35.1.1 ModelSEIRMixing() [1/2]

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

Constructs a [ModelSEIRMixing](#) object.

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

#### Parameters

<i>model</i>	A reference to an existing <a href="#">ModelSEIRMixing</a> object.
<i>vname</i>	The name of the <a href="#">ModelSEIRMixing</a> object.
<i>n</i>	The number of entities in the model.
<i>prevalence</i>	The initial prevalence of the disease in the model.
<i>contact_rate</i>	The contact rate between entities in the model.
<i>transmission_rate</i>	The transmission rate of the disease in the model.
<i>avg_incubation_days</i>	The average incubation period of the disease in the model.
<i>recovery_rate</i>	The recovery rate of the disease in the model.
<i>contact_matrix</i>	The contact matrix between entities in the model. Specified in column-major order.
<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>prevalence</i>	Initial prevalence (proportion)
<i>contact_rate</i>	Average number of contacts (interactions) per step.
<i>transmission_rate</i>	Probability of transmission
<i>recovery_rate</i>	Probability of recovery

### 15.35.1.2 ModelSEIRMixing() [2/2]

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

Constructs a [ModelSEIRMixing](#) object.

## Parameters

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

## 15.35.2 Member Function Documentation

15.35.2.1 `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 >](#).

15.35.2.2 `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 >](#).

### 15.35.2.3 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:

- `epiworld.hpp`

## 15.36 ModelSEIRMixing< TSeq > Class Template Reference

Inheritance diagram for ModelSEIRMixing< TSeq >:



Collaboration diagram for ModelSEIRMixing< TSeq >:





## Public Member Functions

- [ModelSEIRMixing](#) ([ModelSEIRMixing](#)< TSeq > &model, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, std::vector< double > contact\_matrix)  
*Constructs a [ModelSEIRMixing](#) object.*
- [ModelSEIRMixing](#) (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double avg\_incubation\_days, epiworld\_double recovery\_rate, std::vector< double > contact\_matrix)  
*Constructs a [ModelSEIRMixing](#) object.*
- [ModelSEIRMixing](#)< TSeq > & [run](#) (epiworld\_fast\_uint ndays, int seed=-1)  
*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

### 15.36.1 Constructor & Destructor Documentation

#### 15.36.1.1 ModelSEIRMixing() [1/2]

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

Constructs a [ModelSEIRMixing](#) object.

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

## Parameters

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

15.36.1.2 [ModelSEIRMixing\(\)](#) [2/2]

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

Constructs a [ModelSEIRMixing](#) object.

## Parameters

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

## 15.36.2 Member Function Documentation

### 15.36.2.1 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 >](#).

### 15.36.2.2 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 >](#).

### 15.36.2.3 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`

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

## Additional Inherited Members

### 15.37.1 Detailed Description

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

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

#### Parameters

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

### 15.37.2 Member Function Documentation

#### 15.37.2.1 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

## 15.38 ModelSIR< TSeq > Class Template Reference

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

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

Inheritance diagram for ModelSIR< TSeq >:



Collaboration diagram for ModelSIR< TSeq >:



## Public Member Functions

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

*Set the initial states of the model.*

## Additional Inherited Members

### 15.38.1 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

## 15.38.2 Member Function Documentation

## 15.38.2.1 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

## 15.39 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, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate)  
*Template for a Susceptible-Infected-Removed (SIR) model.*
- **ModelSIRCONN** (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate)
- [ModelSIRCONN](#)< TSeq > & [run](#) (epiworld\_fast\_uint ndays, int seed=-1)  
*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.*
- `std::vector< double > generation_time_expected` (int `max_days=200`, int `max_contacts=200`) const

## Static Public Attributes

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

## Additional Inherited Members

### 15.39.1 Constructor & Destructor Documentation

#### 15.39.1.1 ModelSIRCONN()

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

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

#### Parameters

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

### 15.39.2 Member Function Documentation

### 15.39.2.1 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 >](#).

### 15.39.2.2 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> \* >

### 15.39.2.3 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 >](#).

### 15.39.2.4 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`

## 15.40 ModelSIRCONN< TSeq > Class Template Reference

Inheritance diagram for ModelSIRCONN< TSeq >:



Collaboration diagram for ModelSIRCONN< TSeq >:



## Public Member Functions

- [ModelSIRCONN](#) ([ModelSIRCONN](#)< TSeq > &model, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate)  
*Template for a Susceptible-Infected-Removed (SIR) model.*
- **ModelSIRCONN** (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate)
- [ModelSIRCONN](#)< TSeq > & [run](#) (epiworld\_fast\_uint ndays, int seed=-1)  
*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.*
- std::vector< double > **generation\_time\_expected** (int max\_days=200, int max\_contacts=200) const

## Static Public Attributes

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

## Additional Inherited Members

### 15.40.1 Constructor & Destructor Documentation

#### 15.40.1.1 ModelSIRCONN()

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

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

#### Parameters

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

## 15.40.2 Member Function Documentation

### 15.40.2.1 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 >](#).

### 15.40.2.2 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> * >`

### 15.40.2.3 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 >](#).

#### 15.40.2.4 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`

## 15.41 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, const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double death\_rate)  
*Constructs a new SIRD model with the given parameters.*
- **ModelSIRD** (const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double death\_rate)

## Additional Inherited Members

### 15.41.1 Detailed Description

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

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

### 15.41.2 Constructor & Destructor Documentation

#### 15.41.2.1 ModelSIRD()

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

Constructs a new SIRD model with the given parameters.

## Parameters

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

### 15.41.3 Member Function Documentation

#### 15.41.3.1 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](#)

## 15.42 ModelSIRD< TSeq > Class Template Reference

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

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



Inheritance diagram for ModelSIRD< TSeq >:



Collaboration diagram for ModelSIRD< TSeq >:



## Public Member Functions

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

## Additional Inherited Members

### 15.42.1 Detailed Description

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

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

## 15.42.2 Constructor & Destructor Documentation

### 15.42.2.1 ModelSIRD()

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

Constructs a new SIRD model with the given parameters.

#### Parameters

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

## 15.42.3 Member Function Documentation

### 15.42.3.1 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

## 15.43 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, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double death\_rate)

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

- **ModelSIRDCONN** (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, epiworld\_double death\_rate)
- **ModelSIRDCONN**< TSeq > & **run** (epiworld\_fast\_uint ndays, int seed=-1)  
*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

### 15.43.1 Constructor & Destructor Documentation

#### 15.43.1.1 ModelSIRDCONN()

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

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

#### Parameters

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

## 15.43.2 Member Function Documentation

### 15.43.2.1 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 >](#).

### 15.43.2.2 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

## 15.44 ModelSIRDConn< TSeq > Class Template Reference

Inheritance diagram for ModelSIRDConn< TSeq >:



Collaboration diagram for ModelSIRDConn< TSeq >:



### Public Member Functions

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

### 15.44.1 Constructor & Destructor Documentation

#### 15.44.1.1 ModelSIRDCONN()

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

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

#### Parameters

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

### 15.44.2 Member Function Documentation

#### 15.44.2.1 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 >](#).

**15.44.2.2 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`

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

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

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

### 15.45.1 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> )
-------------	---------------------------------------------------------------------------------

## 15.45.2 Constructor & Destructor Documentation

### 15.45.2.1 ModelSIRLogit()

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

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

#### Parameters

<i>vname</i>	Name of the virus.
<i>coefs_infect</i>	Double ptr. Infection coefficients.
<i>coefs_recover</i>	Double ptr. Recovery coefficients.
<i>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

## 15.45.3 Member Function Documentation

### 15.45.3.1 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 >](#).

### 15.45.3.2 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

## 15.46 ModelSIRLogit< TSeq > Class Template Reference

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

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

Inheritance diagram for ModelSIRLogit< TSeq >:



Collaboration diagram for ModelSIRLogit< TSeq >:



### Public Member Functions

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

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

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

### 15.46.1 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> )
-------------	---------------------------------------------------------------------------------

### 15.46.2 Constructor & Destructor Documentation

#### 15.46.2.1 ModelSIRLogit()

```
template<typename TSeq >
ModelSIRLogit< TSeq >::ModelSIRLogit (
    ModelSIRLogit< TSeq > & model,
    const std::string & vname,
    double * data,
    size_t ncols,
```

```

std::vector< double > coefs_infect,
std::vector< double > coefs_recover,
std::vector< size_t > coef_infect_cols,
std::vector< size_t > coef_recover_cols,
epiworld_double transmission_rate,
epiworld_double recovery_rate,
epiworld_double prevalence ) [inline]

```

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

#### Parameters

<i>vname</i>	Name of the virus.
<i>coefs_infect</i>	Double ptr. Infection coefficients.
<i>coefs_recover</i>	Double ptr. Recovery coefficients.
<i>ncoef_infect</i>	Unsigned int. Number of infection coefficients.
<i>ncoef_recover</i>	Unsigned int. Number of recovery coefficients.
<i>coef_infect_cols</i>	Vector<unsigned int>. Ids of infection vars.
<i>coef_recover_cols</i>	Vector<unsigned int>. Ids of recover vars.
<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>prevalence</i>	Initial prevalence (proportion)
<i>contact_rate</i>	Average number of contacts (interactions) per step.
<i>prob_transmission</i>	Probability of transmission
<i>prob_recovery</i>	Probability of recovery

## 15.46.3 Member Function Documentation

### 15.46.3.1 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 >](#).

### 15.46.3.2 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`

## 15.47 `epiworld::epimodels::ModelSIRMixing< TSeq >` Class Template Reference

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



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



## Public Member Functions

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

## Static Public Attributes

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

## Additional Inherited Members

### 15.47.1 Constructor & Destructor Documentation

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

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

Constructs a [ModelSIRMixing](#) object.

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



## Parameters

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

15.47.1.2 `ModelSIRMixing()` [2/2]

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

Constructs a [ModelSIRMixing](#) object.

## Parameters

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

## 15.47.2 Member Function Documentation

### 15.47.2.1 clone\_ptr()

```
template<typename TSeq >
Model< TSeq > * ModelSIRMixing< 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 >](#).

### 15.47.2.2 initial\_states()

```
template<typename TSeq >
ModelSIRMixing< TSeq > & ModelSIRMixing< 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 >](#).

### 15.47.2.3 reset()

```
template<typename TSeq >
void ModelSIRMixing< 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`

## 15.48 ModelSIRMixing< TSeq > Class Template Reference

Inheritance diagram for ModelSIRMixing< TSeq >:



Collaboration diagram for ModelSIRMixing< TSeq >:



### Public Member Functions

- [ModelSIRMixing](#) ([ModelSIRMixing](#)< TSeq > &model, const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, std::vector< double > contact\_matrix)  
*Constructs a [ModelSIRMixing](#) object.*
- [ModelSIRMixing](#) (const std::string &vname, epiworld\_fast\_uint n, epiworld\_double prevalence, epiworld\_double contact\_rate, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate, std::vector< double > contact\_matrix)  
*Constructs a [ModelSIRMixing](#) object.*
- [ModelSIRMixing](#)< TSeq > & [run](#) (epiworld\_fast\_uint ndays, int seed=-1)  
*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.*

- [ModelSIRMixing](#)< TSeq > & [initial\\_states](#) (std::vector< double > proportions\_, std::vector< int > queue\_ ← \_=({}))  
Set the initial states of the model.
- size\_t **get\_n\_infected** (size\_t group) const
- void **set\_contact\_matrix** (std::vector< double > cmat)

## Static Public Attributes

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

## Additional Inherited Members

### 15.48.1 Constructor & Destructor Documentation

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

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

Constructs a [ModelSIRMixing](#) object.

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

#### Parameters

<i>model</i>	A reference to an existing <a href="#">ModelSIRMixing</a> object.
<i>vname</i>	The name of the <a href="#">ModelSIRMixing</a> object.
<i>n</i>	The number of entities in the model.
<i>prevalence</i>	The initial prevalence of the disease in the model.
<i>contact_rate</i>	The contact rate between entities in the model.
<i>transmission_rate</i>	The transmission rate of the disease in the model.
<i>recovery_rate</i>	The recovery rate of the disease in the model.
<i>contact_matrix</i>	The contact matrix between entities in the model.
<i>model</i>	A Model<TSeq> object where to set up the SIR.
<i>vname</i>	std::string Name of the virus
<i>prevalence</i>	Initial prevalence (proportion)
<i>contact_rate</i>	Average number of contacts (interactions) per step.
<i>transmission_rate</i>	Probability of transmission
<i>recovery_rate</i>	Probability of recovery

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

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

Constructs a [ModelSIRMixing](#) object.

#### Parameters

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

## 15.48.2 Member Function Documentation

### 15.48.2.1 clone\_ptr()

```
template<typename TSeq >
Model< TSeq > * ModelSIRMixing< 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 >](#).

### 15.48.2.2 initial\_states()

```
template<typename TSeq >
ModelSIRMixing< TSeq > & ModelSIRMixing< 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 >](#).

### 15.48.2.3 reset()

```
template<typename TSeq >
void ModelSIRMixing< 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/sirmixing.hpp`

## 15.49 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, const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate)
- **ModelSIS** (const std::string &vname, epiworld\_double prevalence, epiworld\_double transmission\_rate, epiworld\_double recovery\_rate)

## Static Public Attributes

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

## Additional Inherited Members

### 15.49.1 Detailed Description

```

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

```

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

## Parameters

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

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

- epiworld.hpp

## 15.50 ModelSIS< TSeq > Class Template Reference

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

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

Inheritance diagram for ModelSIS< TSeq >:



Collaboration diagram for ModelSIS< TSeq >:





## Public Member Functions

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

## Static Public Attributes

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

## Additional Inherited Members

### 15.50.1 Detailed Description

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

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

#### Parameters

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

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

- include/epiworld/models/sis.hpp

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

## Additional Inherited Members

### 15.51.1 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_efficacy</i>	epiworld_double Initial susceptibility_reduction of the immune system
<i>initial_recovery</i>	epiworld_double Initial recovery_rate rate of the immune system
<i>inital_death</i>	epiworld_double Initial death_rate of the immune system

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

- epiworld.hpp

## 15.52 ModelSISD< TSeq > Class Template Reference

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

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

Inheritance diagram for ModelSISD< TSeq >:



Collaboration diagram for ModelSISD< TSeq >:



## Public Member Functions

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

## Additional Inherited Members

### 15.52.1 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

## 15.53 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

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

This model features the following states:

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

- *Removed*

#### Returns

An object of class `epiworld_surv`

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

### Additional Inherited Members

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

- `epiworld.hpp`

## 15.54 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	String. Name of the virus
prevalence	Integer. Number of initial cases of the virus.
efficacy_vax	Double. Efficacy of the vaccine ( $1 - P(\text{acquire the disease})$ ).
latent_period	Double. Shape parameter of a Gamma ( <code>latent_period</code> , 1) distribution. This coincides with the expected number of latent days.
infect_period	Double. Shape parameter of a Gamma ( <code>infected_period</code> , 1) distribution. This coincides with the expected number of infectious days.
prob_symptoms	Double. Probability of generating symptoms.
prop_vaccinated	Double. Probability of vaccination. Coincides with the initial prevalence of vaccinated individuals.
prop_vax_redux_transm	Double. Factor by which the vaccine reduces transmissibility.
prop_vax_redux_infect	Double. Factor by which the vaccine reduces the chances of becoming infected.
surveillance_prob	Double. Probability of testing an agent.
prob_transmission	Double. Raw transmission probability.
prob_death	Double. Raw probability of death for symptomatic individuals.
prob_noreinfect	Double. Probability of no re-infection.

This model features the following states:

- Susceptible
- Latent
- Symptomatic
- Symptomatic isolated
- Asymptomatic
- Asymptomatic isolated
- Recovered

- *Removed*

#### Returns

An object of class *epiworld\_surv*

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

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

### Public Member Functions

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

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

- include/epiworld/network-bones.hpp

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

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

- epiworld.hpp



## 15.57 PersonTools< TSeq > Class Template Reference

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

- include/epiworld/config.hpp

## 15.58 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** ()

#### 15.58.1 Detailed Description

A simple progress bar.

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

- epiworld.hpp

## 15.59 Progress Class Reference

A simple progress bar.

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

### Public Member Functions

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

#### 15.59.1 Detailed Description

A simple progress bar.

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

- include/epiworld/progress.hpp

## 15.60 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 >

### 15.60.1 Detailed Description

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

Controls which agents are verified at each step.

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

#### Template Parameters

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

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

- epiworld.hpp

## 15.61 Queue< TSeq > Class Template Reference

Controls which agents are verified at each step.

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

## Public Member Functions

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

## Static Public Attributes

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

## Friends

- class **Model**< TSeq >

### 15.61.1 Detailed Description

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

Controls which agents are verified at each step.

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

#### Template Parameters

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

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

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

## 15.62 RandGraph Class Reference

### Public Member Functions

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

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

- include/epiworld/random\_graph.hpp

## 15.63 epiworld::SAMPLETYPE Class Reference

### Static Public Attributes

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

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

- epiworld.hpp

## 15.64 SAMPLETYPE Class Reference

### Static Public Attributes

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

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

- include/epiworld/agentssample-bones.hpp

## 15.65 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")
- **Tool** (std::string name, epiworld\_double prevalence, bool as\_proportion)
- 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
- void **distribute** ([Model](#)< TSeq > \*model)
- void **set\_distribution** (ToolToAgentFun< TSeq > fun)

**Get and set the tool functions**

*Parameters*

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

*Returns**epiworld\_double*

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

**Friends**

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

**15.65.1 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

**15.66 Tool< TSeq > Class Template Reference**

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

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

## Public Member Functions

- **Tool** (std::string name="unknown tool")
- **Tool** (std::string name, epiworld\_double prevalence, bool as\_proportion)
- void **set\_sequence** (TSeq d)
- void **set\_sequence** (std::shared\_ptr< TSeq > d)
- 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
- void **distribute** ([Model](#)< TSeq > \*model)
- void **set\_distribution** (ToolToAgentFun< TSeq > fun)
- 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)

### 15.66.1 Detailed Description

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

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

#### Template Parameters

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

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

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

## 15.67 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 >

### 15.67.1 Detailed Description

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

Set of tools (useful for building iterators)

## Template Parameters

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

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

- epiworld.hpp

## 15.68 Tools< TSeq > Class Template Reference

Set of tools (useful for building iterators)

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

### Public Member Functions

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

### Friends

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

### 15.68.1 Detailed Description

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

Set of tools (useful for building iterators)

## Template Parameters

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

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

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



## 15.69 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 >

### 15.69.1 Detailed Description

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

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

#### Template Parameters

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

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

- epiworld.hpp

## 15.70 Tools\_const< TSeq > Class Template Reference

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

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

## Public Member Functions

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

## Friends

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

### 15.70.1 Detailed Description

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

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

#### Template Parameters

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

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

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

## 15.71 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 >

### 15.71.1 Detailed Description

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

Personalized data by the user.

#### Template Parameters

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

## 15.71.2 Constructor & Destructor Documentation

### 15.71.2.1 UserData()

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

Construct a new User Data object.

#### Parameters

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

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

- epiworld.hpp

## 15.72 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 >

## 15.72.1 Detailed Description

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

Personalized data by the user.

#### Template Parameters

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

## 15.72.2 Constructor & Destructor Documentation

### 15.72.2.1 UserData()

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

Construct a new User Data object.

#### Parameters

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

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

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

## 15.73 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

#### 15.73.1 Detailed Description

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

Vector hasher.

Template Parameters

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

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

- epiworld.hpp

## 15.74 vecHasher< T > Struct Template Reference

Vector hasher.

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

### Public Member Functions

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

#### 15.74.1 Detailed Description

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

Vector hasher.

## Template Parameters

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

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

- include/epiworld/misc.hpp

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

[Virus.](#)

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

### Public Member Functions

- **Virus** (std::string name="unknown virus")
- **Virus** (std::string name, epiworld\_double prevalence, bool as\_proportion)
- void **mutate** ([Model](#)< TSeq > \*model)
- void **set\_mutation** (MutFun< TSeq > fun)
- 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)

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

### Friends

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

## 15.75.1 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

## 15.76 Virus< TSeq > Class Template Reference

[Virus.](#)

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

### Public Member Functions

- **Virus** (std::string name="unknown virus")
- **Virus** (std::string name, epiworld\_double prevalence, bool as\_proportion)
- void **mutate** ([Model](#)< TSeq > \*model)
- void **set\_mutation** (MutFun< TSeq > fun)
- 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\_prob\_infecting** (epiworld\_double \*prob)
- void **set\_prob\_infecting\_fun** (VirusFun< TSeq > fun)
- void **set\_prob\_recovery\_fun** (VirusFun< TSeq > fun)
- void **set\_prob\_death\_fun** (VirusFun< TSeq > fun)
- void **set\_incubation\_fun** (VirusFun< TSeq > fun)
- void **set\_prob\_infecting** (const epiworld\_double \*prob)
- void **set\_prob\_recovery** (const epiworld\_double \*prob)
- void **set\_prob\_death** (const epiworld\_double \*prob)
- void **set\_incubation** (const epiworld\_double \*prob)
- void **set\_prob\_infecting** (epiworld\_double prob)
- void **set\_prob\_recovery** (epiworld\_double prob)
- void **set\_prob\_death** (epiworld\_double prob)
- void **set\_incubation** (epiworld\_double prob)

### Get and set the state and queue

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

### Parameters

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

- void **set\_state** (epiworld\_fast\_int init, epiworld\_fast\_int end, epiworld\_fast\_int removed=-99)
- void **set\_queue** (epiworld\_fast\_int init, epiworld\_fast\_int end, epiworld\_fast\_int removed=-99)
- void **get\_state** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*end, epiworld\_fast\_int \*removed=nullptr)
- void **get\_queue** (epiworld\_fast\_int \*init, epiworld\_fast\_int \*end, epiworld\_fast\_int \*removed=nullptr)

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

### Friends

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

### 15.76.1 Detailed Description

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

[Virus.](#)

Template Parameters

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

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

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

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

## 15.77 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 >

### 15.77.1 Detailed Description

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

Set of viruses (useful for building iterators)

## Template Parameters

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

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

- epiworld.hpp

## 15.78 Viruses< TSeq > Class Template Reference

Set of viruses (useful for building iterators)

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

### Public Member Functions

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

### Friends

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

### 15.78.1 Detailed Description

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

Set of viruses (useful for building iterators)

## Template Parameters

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

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

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

## 15.79 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 >

### 15.79.1 Detailed Description

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

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

#### Template Parameters

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

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

- epiworld.hpp

## 15.80 Viruses\_const< TSeq > Class Template Reference

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

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

## Public Member Functions

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

## Friends

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

### 15.80.1 Detailed Description

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

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

#### Template Parameters

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

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

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

## Chapter 16

# File Documentation

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

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

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

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



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



## Functions

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

### 16.1.1 Detailed Description

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

#### Author

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

#### Version

0.1

#### Date

2022-06-15

#### Copyright

Copyright (c) 2022



# Index

add\_globlevent  
  epiworld::Model< TSeq >, 89  
  Model< TSeq >, 103  
AdjList, 41  
  AdjList, 41  
  epiworld::AdjList, 43  
  read\_edgelist, 42  
Agent< TSeq >, 44  
  default\_rm\_entity, 48  
  operator(), 46  
  swap\_neighbors, 48  
AgentsSample  
  AgentsSample< TSeq >, 53  
  epiworld::AgentsSample< TSeq >, 55  
AgentsSample< TSeq >, 52  
  AgentsSample, 53  
clone\_ptr  
  epiworld::epimodels::ModelSEIRCONN< TSeq >, 119  
  epiworld::epimodels::ModelSEIRDCONN< TSeq >, 132  
  epiworld::epimodels::ModelSEIRMixing< TSeq >, 139  
  epiworld::epimodels::ModelSIRCONN< TSeq >, 149  
  epiworld::epimodels::ModelSIRDCONN< TSeq >, 161  
  epiworld::epimodels::ModelSIRLogit< TSeq >, 167  
  epiworld::epimodels::ModelSIRMixing< TSeq >, 173  
  epiworld::Model< TSeq >, 89  
  Model< TSeq >, 103  
  ModelSEIRCONN< TSeq >, 122  
  ModelSEIRDCONN< TSeq >, 135  
  ModelSEIRMixing< TSeq >, 142  
  ModelSIRCONN< TSeq >, 153  
  ModelSIRDCONN< TSeq >, 163  
  ModelSIRLogit< TSeq >, 170  
  ModelSIRMixing< TSeq >, 177  
DataBase< TSeq >, 55  
  generation\_time, 58  
  get\_transmissions, 58  
  operator==, 58, 59  
  record\_virus, 59  
  reproductive\_number, 60  
  transition\_probability, 60  
default\_rm\_entity  
  Agent< TSeq >, 48  
  Entity< TSeq >, 69  
  epiworld::Agent< TSeq >, 52  
  epiworld::Entity< TSeq >, 71  
Entities< TSeq >, 65  
Entities\_const< TSeq >, 66  
Entity  
  Entity< TSeq >, 69  
  epiworld::Entity< TSeq >, 70  
Entity< TSeq >, 68  
  default\_rm\_entity, 69  
  Entity, 69  
epiworld::AdjList, 42  
  AdjList, 43  
  read\_edgelist, 43  
epiworld::Agent< TSeq >, 49  
  default\_rm\_entity, 52  
  operator(), 51  
  swap\_neighbors, 51  
epiworld::AgentsSample< TSeq >, 54  
  AgentsSample, 55  
epiworld::DataBase< TSeq >, 60  
  generation\_time, 63  
  get\_transmissions, 63  
  operator==, 63  
  record\_virus, 64  
  reproductive\_number, 64  
  transition\_probability, 64  
epiworld::Entities< TSeq >, 66  
epiworld::Entities\_const< TSeq >, 67  
epiworld::Entity< TSeq >, 70  
  default\_rm\_entity, 71  
  Entity, 70  
epiworld::epimodels::ModelDiffNet< TSeq >, 109  
epiworld::epimodels::ModelSEIR< TSeq >, 112  
  initial\_states, 114  
  update\_exposed\_seir, 114  
  update\_infected\_seir, 114  
epiworld::epimodels::ModelSEIRCONN< TSeq >, 118  
  clone\_ptr, 119  
  initial\_states, 120  
  ModelSEIRCONN, 119  
  reset, 120  
epiworld::epimodels::ModelSEIRD< TSeq >, 124  
  ModelSEIRD, 125, 126  
  update\_exposed\_seir, 126  
epiworld::epimodels::ModelSEIRDCONN< TSeq >, 130  
  clone\_ptr, 132

- initial\_states, 132
- ModelSEIRDCONN, 131
- reset, 132
- epiworld::epimodels::ModelSEIRMixing< TSeq >, 136
  - clone\_ptr, 139
  - initial\_states, 139
  - ModelSEIRMixing, 137, 138
  - reset, 139
- epiworld::epimodels::ModelSIR< TSeq >, 144
  - initial\_states, 145
- epiworld::epimodels::ModelSIRCONN< TSeq >, 148
  - clone\_ptr, 149
  - get\_n\_infected, 150
  - initial\_states, 150
  - ModelSIRCONN, 149
  - reset, 150
- epiworld::epimodels::ModelSIRD< TSeq >, 154
  - initial\_states, 156
  - ModelSIRD, 155
- epiworld::epimodels::ModelSIRDCONN< TSeq >, 159
  - clone\_ptr, 161
  - ModelSIRDCONN, 160
  - reset, 161
- epiworld::epimodels::ModelSIRLogit< TSeq >, 164
  - clone\_ptr, 167
  - ModelSIRLogit, 166
  - reset, 167
- epiworld::epimodels::ModelSIRMixing< TSeq >, 171
  - clone\_ptr, 173
  - initial\_states, 174
  - ModelSIRMixing, 172, 173
  - reset, 174
- epiworld::epimodels::ModelSIS< TSeq >, 178
- epiworld::epimodels::ModelSISD< TSeq >, 181
- epiworld::epimodels::ModelSURV< TSeq >, 184
- epiworld::Event< TSeq >, 71
  - Event, 72
- epiworld::GlobalEvent< TSeq >, 74
  - GlobalEvent, 75
- epiworld::GroupSampler< TSeq >, 77
- epiworld::LFMCMC< TData >, 78
- epiworld::Model< TSeq >, 81
  - add\_globlevent, 89
  - clone\_ptr, 89
  - events\_add, 89
  - events\_run, 90
  - initial\_states\_fun, 93
  - load\_agents\_entities\_ties, 90
  - rbinomd, 93
  - reset, 91
  - rexp, 93
  - rgammad, 93
  - rlognormald, 94
  - rnormd, 94
  - run\_multiple, 91
  - runifd, 94
  - set\_agents\_data, 92
  - set\_name, 92
  - time\_elapsed, 94
  - write\_data, 92
- epiworld::PersonTools< TSeq >, 188
- epiworld::Progress, 189
- epiworld::Queue< TSeq >, 190
- epiworld::sampler, 33
  - make\_sample\_virus\_neighbors, 33
  - make\_update\_susceptible, 34
  - sample\_virus\_single, 34
- epiworld::SAMPLETYPE, 192
- epiworld::Tool< TSeq >, 192
- epiworld::Tools< TSeq >, 195
- epiworld::Tools\_const< TSeq >, 197
- epiworld::UserData< TSeq >, 198
  - UserData, 200
- epiworld::vecHasher< T >, 202
- epiworld::Virus< TSeq >, 203
- epiworld::Viruses< TSeq >, 207
- epiworld::Viruses\_const< TSeq >, 209
- Event
  - epiworld::Event< TSeq >, 72
  - Event< TSeq >, 74
- Event< TSeq >, 72
  - Event, 74
- events\_add
  - epiworld::Model< TSeq >, 89
  - Model< TSeq >, 103
- events\_run
  - epiworld::Model< TSeq >, 90
  - Model< TSeq >, 104
- generation\_time
  - DataBase< TSeq >, 58
  - epiworld::DataBase< TSeq >, 63
- get\_n\_infected
  - epiworld::epimodels::ModelSIRCONN< TSeq >, 150
  - ModelSIRCONN< TSeq >, 153
- get\_transmissions
  - DataBase< TSeq >, 58
  - epiworld::DataBase< TSeq >, 63
- GlobalEvent
  - epiworld::GlobalEvent< TSeq >, 75
  - GlobalEvent< TSeq >, 76
- GlobalEvent< TSeq >, 76
  - GlobalEvent, 76
- GroupSampler< TSeq >, 77
- include/epiworld/agent-meat-state.hpp, 211
- initial\_states
  - epiworld::epimodels::ModelSEIR< TSeq >, 114
  - epiworld::epimodels::ModelSEIRCONN< TSeq >, 120
  - epiworld::epimodels::ModelSEIRDCONN< TSeq >, 132
  - epiworld::epimodels::ModelSEIRMixing< TSeq >, 139
  - epiworld::epimodels::ModelSIR< TSeq >, 145

- epiworld::epimodels::ModelSIRCONN< TSeq >, 150
- epiworld::epimodels::ModelSIRD< TSeq >, 156
- epiworld::epimodels::ModelSIRMixing< TSeq >, 174
- ModelSEIR< TSeq >, 116
- ModelSEIRCONN< TSeq >, 123
- ModelSEIRDCONN< TSeq >, 135
- ModelSEIRMixing< TSeq >, 143
- ModelSIR< TSeq >, 147
- ModelSIRCONN< TSeq >, 153
- ModelSIRD< TSeq >, 158
- ModelSIRMixing< TSeq >, 177
- initial\_states\_fun
  - epiworld::Model< TSeq >, 93
  - Model< TSeq >, 107
- LFMCMC< TData >, 79
- load\_agents\_entities\_ties
  - epiworld::Model< TSeq >, 90
  - Model< TSeq >, 104
- make\_sample\_virus\_neighbors
  - epiworld::sampler, 33
  - sampler, 36
- make\_update\_susceptible
  - epiworld::sampler, 34
  - sampler, 37
- Model< TSeq >, 95
  - add\_globlevent, 103
  - clone\_ptr, 103
  - events\_add, 103
  - events\_run, 104
  - initial\_states\_fun, 107
  - load\_agents\_entities\_ties, 104
  - rbinomd, 107
  - reset, 105
  - rexp, 107
  - rgammad, 107
  - rlognormald, 108
  - rnormd, 108
  - run\_multiple, 105
  - runifd, 108
  - set\_agents\_data, 105
  - set\_name, 106
  - time\_elapsed, 108
  - write\_data, 106
- ModelDiffNet< TSeq >, 110
- ModelSEIR< TSeq >, 115
  - initial\_states, 116
  - update\_exposed\_seir, 117
  - update\_infected\_seir, 117
- ModelSEIRCONN
  - epiworld::epimodels::ModelSEIRCONN< TSeq >, 119
  - ModelSEIRCONN< TSeq >, 122
- ModelSEIRCONN< TSeq >, 121
  - clone\_ptr, 122
  - initial\_states, 123
- ModelSEIRCONN, 122
  - reset, 123
- ModelSEIRD
  - epiworld::epimodels::ModelSEIRD< TSeq >, 125, 126
  - ModelSEIRD< TSeq >, 128, 129
- ModelSEIRD< TSeq >, 127
  - ModelSEIRD, 128, 129
  - update\_exposed\_seir, 129
- ModelSEIRDCONN
  - epiworld::epimodels::ModelSEIRDCONN< TSeq >, 131
  - ModelSEIRDCONN< TSeq >, 134
- ModelSEIRDCONN< TSeq >, 133
  - clone\_ptr, 135
  - initial\_states, 135
  - ModelSEIRDCONN, 134
  - reset, 136
- ModelSEIRMixing
  - epiworld::epimodels::ModelSEIRMixing< TSeq >, 137, 138
  - ModelSEIRMixing< TSeq >, 141, 142
- ModelSEIRMixing< TSeq >, 140
  - clone\_ptr, 142
  - initial\_states, 143
  - ModelSEIRMixing, 141, 142
  - reset, 143
- ModelSIR< TSeq >, 145
  - initial\_states, 147
- ModelSIRCONN
  - epiworld::epimodels::ModelSIRCONN< TSeq >, 149
  - ModelSIRCONN< TSeq >, 152
- ModelSIRCONN< TSeq >, 151
  - clone\_ptr, 153
  - get\_n\_infected, 153
  - initial\_states, 153
  - ModelSIRCONN, 152
  - reset, 154
- ModelSIRD
  - epiworld::epimodels::ModelSIRD< TSeq >, 155
  - ModelSIRD< TSeq >, 158
- ModelSIRD< TSeq >, 156
  - initial\_states, 158
  - ModelSIRD, 158
- ModelSIRDCONN
  - epiworld::epimodels::ModelSIRDCONN< TSeq >, 160
  - ModelSIRDCONN< TSeq >, 163
- ModelSIRDCONN< TSeq >, 162
  - clone\_ptr, 163
  - ModelSIRDCONN, 163
  - reset, 164
- ModelSIRLogit
  - epiworld::epimodels::ModelSIRLogit< TSeq >, 166
  - ModelSIRLogit< TSeq >, 169
- ModelSIRLogit< TSeq >, 168

- clone\_ptr, 170
- ModelSIRLogit, 169
- reset, 170
- ModelSIRMixing
  - epiworld::epimodels::ModelSIRMixing< TSeq >, 172, 173
  - ModelSIRMixing< TSeq >, 176, 177
- ModelSIRMixing< TSeq >, 175
  - clone\_ptr, 177
  - initial\_states, 177
  - ModelSIRMixing, 176, 177
  - reset, 178
- ModelSIS< TSeq >, 180
- ModelSISD< TSeq >, 183
- ModelSURV< TSeq >, 186
- Network< Nettype, Nodetype, Edgetype >, 188
- operator()
  - Agent< TSeq >, 46
  - epiworld::Agent< TSeq >, 51
- operator==
  - DataBase< TSeq >, 58, 59
  - epiworld::DataBase< TSeq >, 63
- PersonTools< TSeq >, 189
- Progress, 189
- Queue< TSeq >, 190
- RandGraph, 191
- rbinomd
  - epiworld::Model< TSeq >, 93
  - Model< TSeq >, 107
- read\_edgelist
  - AdjList, 42
  - epiworld::AdjList, 43
- record\_virus
  - DataBase< TSeq >, 59
  - epiworld::DataBase< TSeq >, 64
- reproductive\_number
  - DataBase< TSeq >, 60
  - epiworld::DataBase< TSeq >, 64
- reset
  - epiworld::epimodels::ModelSEIRCONN< TSeq >, 120
  - epiworld::epimodels::ModelSEIRDCONN< TSeq >, 132
  - epiworld::epimodels::ModelSEIRMixing< TSeq >, 139
  - epiworld::epimodels::ModelSIRCONN< TSeq >, 150
  - epiworld::epimodels::ModelSIRDCONN< TSeq >, 161
  - epiworld::epimodels::ModelSIRLogit< TSeq >, 167
  - epiworld::epimodels::ModelSIRMixing< TSeq >, 174
  - epiworld::Model< TSeq >, 91
  - Model< TSeq >, 105
  - ModelSEIRCONN< TSeq >, 123
  - ModelSEIRDCONN< TSeq >, 136
  - ModelSEIRMixing< TSeq >, 143
  - ModelSIRCONN< TSeq >, 154
  - ModelSIRDCONN< TSeq >, 164
  - ModelSIRLogit< TSeq >, 170
  - ModelSIRMixing< TSeq >, 178
- rexp
  - epiworld::Model< TSeq >, 93
  - Model< TSeq >, 107
- rgammad
  - epiworld::Model< TSeq >, 93
  - Model< TSeq >, 107
- rlognormald
  - epiworld::Model< TSeq >, 94
  - Model< TSeq >, 108
- rnormd
  - epiworld::Model< TSeq >, 94
  - Model< TSeq >, 108
- run\_multiple
  - epiworld::Model< TSeq >, 91
  - Model< TSeq >, 105
- runifd
  - epiworld::Model< TSeq >, 94
  - Model< TSeq >, 108
- sample\_virus\_single
  - epiworld::sampler, 34
  - sampler, 37
- sampler, 36
  - make\_sample\_virus\_neighbors, 36
  - make\_update\_susceptible, 37
  - sample\_virus\_single, 37
- SAMPLETYPE, 192
- set\_agents\_data
  - epiworld::Model< TSeq >, 92
  - Model< TSeq >, 105
- set\_name
  - epiworld::Model< TSeq >, 92
  - Model< TSeq >, 106
- swap\_neighbors
  - Agent< TSeq >, 48
  - epiworld::Agent< TSeq >, 51
- time\_elapsed
  - epiworld::Model< TSeq >, 94
  - Model< TSeq >, 108
- Tool< TSeq >, 193
- Tools< TSeq >, 196
- Tools\_const< TSeq >, 197
- transition\_probability
  - DataBase< TSeq >, 60
  - epiworld::DataBase< TSeq >, 64
- update\_exposed\_seir
  - epiworld::epimodels::ModelSEIR< TSeq >, 114
  - epiworld::epimodels::ModelSEIRD< TSeq >, 126
  - ModelSEIR< TSeq >, 117

- ModelSEIRD< TSeq >, [129](#)
- update\_infected\_seir
  - epiworld::epimodels::ModelSEIR< TSeq >, [114](#)
  - ModelSEIR< TSeq >, [117](#)
- UserData
  - epiworld::UserData< TSeq >, [200](#)
  - UserData< TSeq >, [201](#)
- UserData< TSeq >, [200](#)
- UserData, [201](#)
- vecHasher< T >, [202](#)
- Virus< TSeq >, [205](#)
- Viruses< TSeq >, [208](#)
- Viruses\_const< TSeq >, [209](#)
- write\_data
  - epiworld::Model< TSeq >, [92](#)
  - Model< TSeq >, [106](#)