epiworld

0.0-1

Generated by Doxygen 1.9.1

Chapter 1

Main Page

1.1 epiworld

This C++ library provides a general framework for epidemiologic simulation. The core principle of `epiworld` is fast epidemiological prototyping for building complex models quickly. Here are some of its main features:

- It only depends on the standard library (C++11 required.)
- It is a template library.
- It is header-only (`single file`).
- Models can have an arbitrary set of states.
- Viruses and tools (e.g., vaccines, mask-wearing) can be designed to have arbitrary features.
- Multiple tools and viruses can live in the same simulation.
- It is *FAST*: About 7.5 Million person/day simulations per second (see example below).

Various examples can be found in the examples folder.

1.1.1 Hello world

Here is a simple SIR model implemented with `epiworld`. The source code can be found `here`, and you can compile the code as follows:

```
g++ -std=c++17 -O2 readme.cpp -o readme.o
```

As you can see in `readme.cpp`, to use epiworld you only need to incorporate the single header file `epiworld.hpp`:

```
#include "epiworld.hpp"
using namespace epiworld;
int main()
{
    // Creating a model with three statuses:
    // - Susceptible: Status 0
    // - Infected: Status 1
    // - Recovered: Status 2
    Model<> model;
```

```
    model.add_status("Susceptible", default_update_susceptible<>);
    model.add_status("Infected", default_update_exposed<>);
    model.add_status("Recovered");
    // Desgining a virus: This virus will:
    // - Have a 90% transmission rate
    // - Have a 50% recovery rate
    // - Infected individuals become "Infected" (status 1)
    // - Recovered individuals become "Recovered" (status 2)
    // Only five individuals will have the virus from the beginning.
    Virus<> virus("covid 19");
    virus.set_prob_infecting(.9);
    virus.set_prob_recover(.5);

    virus.set_status(1, 2);
    model.add_virus_n(virus, 5);

    // Generating a random pop from a smallworld network
    model.population_smallworld(100000);
    // Initializing setting days and seed
    model.init(100, 123);
    // Running the model
    model.run();
    model.print();

}
```

And you should get something like the following:

```
Running the model...
_____________________________________________________________________________
|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||| done.
_____________________________________________________________________________
SIMULATION STUDY
Population size    : 100000
Days (duration)    : 100 (of 100)
Number of variants : 1
Last run elapsed t : 134.00ms
Rewiring           : off
Virus(es):
 - covid 19 (baseline prevalence: 5 seeds)
Tool(s):
 (none)
Model parameters:
 (none)
Distribution of the population at time 100:
- (0) Total Susceptible :  99995 -> 95466
- (1) Total Infected    :      5 -> 70
- (2) Total Recovered   :      0 -> 4464
```

Which took about 0.134 seconds ($\sim$ 7.5 million ppl x day / second).

1.1.2 Simulation Steps

The core logic of the model relies on user-defined statuses and their corresponding update functions. In particular, the model does not have a predefined set of statuses, e.g., susceptible, infected, recovered; it is the user who establishes them. This provides a great deal of flexibility as models in `epiworld` can have an arbitrary set of statuses.

Like most other ABM, `epiworld` simulates the evolution of a system in discrete steps. Each step represents a day in the system, and changes are reflected at the beginning of the following day. Therefore, agents can become recovered and transmit a virus on the same day. A single step of `epiworld` features the following procedures:

1. **Status update**: Agents are updated according to the status they are at.
2. (optional) **Execute global actions**: A call of user-defined functions affecting the system. These can make any type of change in the system.
3. (optional) **Apply rewiring algorithm**: When specified, the network is rewired according to a user-defined function.
4. **Lock the results**: The current date is incremented in one unit and the changes (exposition, new infections, recoveries, etc.) are recorded in the database.

5. (optional) **Mutate Variants**: When defined, variants can mutate, with the new variants appearing the next day.

To speed up computations, `epiworld` uses by default a queuing system that decides which agents will be active during each step and which will not. Agents are active when either they or at least one of their neighbors has a virus active. Agents' updates are triggered only for those who are in the queue, which in most cases accelerates the completion of the current step.

1.1.3 Agents

Agents carry two sets of important information: viruses and tools. Each agent can have multiple instances of them, meaning that multiple viruses and tools can coexist in a model. At each step of the simulation, an agent can face the following changes:

- **Acquire a virus (`add_virus()`)**: Become exposed to a particular virus+host.
- **Lose a virus (`rm_virus()`)**: Removing a virus from the agent. Losing a virus triggers a call to the virus's `postrecovery()` function, which can, for example, result in gaining immunity to that variant.
- **Change status (`change_status()`)**: An arbitrary change in the status of the agent. Examples of this are moving from "exposed" to "infected," from "infected" to "ICU," etc.
- **Become removed (`rm_agent_by_virus()`)**: An agent becomes inactive after its condition becoming worse. In such a case, all viruses attached to the agent are removed as well.

Any action in the model can trigger a change in its queuing system. By default, becoming exposed makes the agent (and its neighbors) active in the queuing system. Likewise, losing all viruses could make the agent and its neighbors inactive.

1.1.4 Contagion

Susceptible individuals can acquire a virus from any of their infected connections. The probability that susceptible individual `i` gets the virus `v` from individual `j` depends on how three things:

1. The transmissibility of the virus, ,
2. The contagion reduction factor of `i`, , and
3. The host's transmission reduction factor, .

The last two are computed from and 's tools. Ultimately, the probability of getting virus v from equals:

Nonetheless, the default behavior of the simulation model is to assume that individuals can acquire only one disease at a time, if any. This way, the actual probability is:

The latter is calculated using Bayes' rule

Where

This way, viruses with higher transmissibility will be more likely to be acquired when competing with other variants.

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

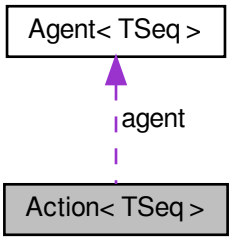
Class Documentation

3.1 Action< TSeq > Struct Template Reference

Action data for update an agent.

`#include <config.hpp>`

Collaboration diagram for Action< TSeq >:



Public Member Functions

- Action (Agent< TSeq > *agent_, VirusPtr< TSeq > virus_, ToolPtr< TSeq > tool_, epiworld_fast_int new_status_, epiworld_fast_int queue_, ActionFun< TSeq > call_)

  Construct a new Action object.

Public Attributes

- Agent< TSeq > * **agent**
- VirusPtr< TSeq > **virus**
- ToolPtr< TSeq > **tool**
- epiworld_fast_int **new_status**
- epiworld_fast_int **queue**
- ActionFun< TSeq > **call**

3.1.1 Detailed Description

template<typename TSeq>
struct Action< TSeq >

Action data for update an agent.

Template Parameters

TSeq	

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Action()

```
template<typename TSeq >
Action< TSeq >::Action (
            Agent< TSeq > * agent_,
            VirusPtr< TSeq > virus_,
            ToolPtr< TSeq > tool_,
            epiworld_fast_int new_status_,
            epiworld_fast_int queue_,
            ActionFun< TSeq > call_ )  [inline]
```

Construct a new Action object.

All the parameters are rather optional.

Parameters

*agent_*	Agent over who the action will happen
*virus_*	Virus to add
*tool_*	Tool to add
virus_idx	Index of virus to be removed (if needed)
tool_idx	Index of tool to be removed (if needed)
*new_↩ status_*	Next status
*queue_*	Efect on the queue
*call_*	The action call (if needed)

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

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

3.2 AdjList Class Reference

Public Member Functions

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

  Construct a new Adj List object.
- void read_edgelist (std::string fn, int size, int skip=0, bool directed=true)

  Read an edgelist.
- std::map< unsigned int, unsigned int > **operator()** (unsigned int i) const
- void **print** (unsigned int limit=20u) const
- size_t vcount () const

  Number of vertices/nodes in the network.
- size_t ecount () const

  Number of edges/arcs/ties in the network.
- std::vector< std::map< unsigned int, unsigned int > > & **get_dat** ()
- bool is_directed () const

  `true` *if the network is directed.*

3.2.1 Constructor & Destructor Documentation

3.2.1.1 AdjList()

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

Construct a new Adj List object.

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

Parameters

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

3.2.2 Member Function Documentation

3.2.2.1 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

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

3.3 Agent< TSeq > Class Template Reference

Agent (agents)

`#include <agent-bones.hpp>`

Public Member Functions

- **Agent** (const Agent< TSeq > &p)
- int get_id () const

  Id of the individual.
- std::mt19937 * **get_rand_endgine** ()
- Model< TSeq > * **get_model** ()
- VirusPtr< TSeq > & **get_virus** (int i)
- Viruses< TSeq > **get_viruses** ()
- const Viruses_const< TSeq > **get_viruses** () const
- size_t **get_n_viruses** () const noexcept
- ToolPtr< TSeq > & **get_tool** (int i)
- Tools< TSeq > **get_tools** ()
- const Tools_const< TSeq > **get_tools** () const
- size_t **get_n_tools** () const noexcept
- void **mutate_variant** ()
- void **add_neighbor** (Agent< TSeq > *p, bool check_source=true, bool check_target=true)
- std::vector< Agent< TSeq > * > & **get_neighbors** ()
- void **change_status** (epiworld_fast_uint new_status, epiworld_fast_int queue=0)
- const epiworld_fast_uint & **get_status** () const

- void **reset** ()
- bool **has_tool** (unsigned int t) const
- bool **has_tool** (std::string name) const
- bool **has_virus** (unsigned int t) const
- bool **has_virus** (std::string name) const

Add/Remove Virus/Tool

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

Parameters

tool	*Tool to add*
virus	*Virus to add*
status_new	*Status after the change*
queue	

- void **add_tool** (ToolPtr< TSeq > tool, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **add_tool** (Tool< TSeq > tool, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **add_virus** (VirusPtr< TSeq > virus, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **add_virus** (Virus< TSeq > virus, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_tool** (epiworld_fast_uint tool_idx, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_tool** (ToolPtr< TSeq > &tool, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_virus** (epiworld_fast_uint virus_idx, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void **rm_virus** (VirusPtr< TSeq > &virus, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)
- void rm_agent_by_virus (epiworld_fast_uint virus_idx, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)

  Agent removed by virus.
- void rm_agent_by_virus (VirusPtr< TSeq > &virus, epiworld_fast_int status_new=-99, epiworld_fast_int queue=-99)

  Agent removed by virus.

Get the rates (multipliers) for the agent

Parameters

v	*A pointer to a virus.*

Returns

epiworld_double

- epiworld_double **get_susceptibility_reduction** (VirusPtr< TSeq > v)
- epiworld_double **get_transmission_reduction** (VirusPtr< TSeq > v)
- epiworld_double **get_recovery_enhancer** (VirusPtr< TSeq > v)
- epiworld_double **get_death_reduction** (VirusPtr< TSeq > v)

Friends

- class **Model**< **TSeq** >
- class **Virus**< **TSeq** >

- class **Viruses**< **TSeq** >
- class **Viruses_const**< **TSeq** >
- class **Tool**< **TSeq** >
- class **Tools**< **TSeq** >
- class **Queue**< **TSeq** >
- void **default_add_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_add_tool** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_tool** (Action< TSeq > &a, Model< TSeq > *m)

3.3.1 Detailed Description

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

Agent (agents)

Template Parameters

TSeq	Sequence type (should match `TSeq` across the model)

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

- include/epiworld/agent-bones.hpp

3.4 DataBase< TSeq > Class Template Reference

Statistical data about the process.

`#include <database-bones.hpp>`

Public Member Functions

- **DataBase** (Model< TSeq > &m)
- void record_variant (Virus< TSeq > &v)

  Registering a new variant.
- void **record_tool** (Tool< TSeq > &t)
- void **set_seq_hasher** (std::function< std::vector< int >(TSeq)> fun)
- void **set_model** (Model< TSeq > &m)
- Model< TSeq > * **get_model** ()
- void **record** ()
- const std::vector< TSeq > & **get_sequence** () const
- const std::vector< int > & **get_nexposed** () const
- size_t **size** () const
- void **write_data** (std::string fn_variant_info, std::string fn_variant_hist, std::string fn_tool_info, std::string fn_tool_hist, std::string fn_total_hist, std::string fn_transmission, std::string fn_transition) const
- void **record_transmission** (int i, int j, int variant)
- size_t **get_n_variants** () const

- size_t **get_n_tools** () const
- void **reset** ()
- void **set_user_data** (std::vector< std::string > names)
- void **add_user_data** (std::vector< epiworld_double > x)
- void **add_user_data** (unsigned int j, epiworld_double x)
- UserData< TSeq > & **get_user_data** ()

Get recorded information from the model

Parameters

what	*std::string, The status, e.g., 0, 1, 2, ...*

Returns

In `get_today_total`, the current counts of `what`.

In `get_today_variant`, the current counts of `what` for each variant.

In `get_hist_total`, the time series of `what`

In `get_hist_variant`, the time series of what for each variant.

In `get_hist_total_date` and `get_hist_variant_date` the corresponding dates

- int **get_today_total** (std::string what) const
- int **get_today_total** (epiworld_fast_uint what) const
- void **get_today_total** (std::vector< std::string > *status=nullptr, std::vector< int > *counts=nullptr) const
- void **get_today_variant** (std::vector< std::string > &status, std::vector< int > &id, std::vector< int > &counts) const
- void **get_hist_total** (std::vector< int > *date, std::vector< std::string > *status, std::vector< int > *counts) const
- void **get_hist_variant** (std::vector< int > &date, std::vector< int > &id, std::vector< std::string > &status, std::vector< int > &counts) const

Friends

- class **Model< TSeq >**
- void **default_add_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_add_tool** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_tool** (Action< TSeq > &a, Model< TSeq > *m)

3.4.1 Detailed Description

template<typename TSeq>
class DataBase< TSeq >

Statistical data about the process.

Template Parameters

TSeq	

3.4.2 Member Function Documentation

3.4.2.1 record_variant()

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

Registering a new variant.

Parameters

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

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

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

3.5 Entity< TSeq > Class Template Reference

Public Member Functions

- void **add_agent** (Agent< TSeq > &p)
- void **add_agent** (Agent< TSeq > *p)
- void **rm_agent** (size_t idx)
- size_t **size** () const noexcept
- void **set_location** (std::vector< epiworld_double > loc)
- std::vector< epiworld_double > & **get_location** ()
- std::vector< Agent< TSeq > * >::iterator **begin** ()
- std::vector< Agent< TSeq > * >::iterator **end** ()
- std::vector< Agent< TSeq > * >::const_iterator **begin** () const
- std::vector< Agent< TSeq > * >::const_iterator **end** () const

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

- include/epiworld/entity-bones.hpp

3.6 LFMCMC< TData > Class Template Reference

Likelihood-Free Markov Chain Monte Carlo.

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

Public Member Functions

- void **run** (VEC(epiworld_double) param_init, size_t n_samples_, epiworld_double epsilon_)
- **LFMCMC** (TData &observed_data_)
- void **set_observed_data** (TData &observed_data_)
- void **set_proposal_fun** (FUN< void(VEC(epiworld_double)&, LFMCMC< TData > *)> fun)
- void **set_simulation_fun** (FUN< TData(VEC(epiworld_double)&, LFMCMC< TData > *)> fun)
- void **set_summary_fun** (FUN< VEC(epiworld_double)(TData &, LFMCMC< TData > *)> fun)
- void **set_kernel_fun** (FUN< epiworld_double(VEC(epiworld_double)&, epiworld_double, LFMCMC< TData > *)> fun)
- const size_t **get_n_samples** ()
- const size_t **get_n_statistics** ()
- const size_t **get_n_parameters** ()
- const epiworld_double **get_epsilon** ()
- const **VEC** (epiworld_double) &get_params_now()
- const **VEC** (epiworld_double) &get_params_prev()
- const **VEC** (epiworld_double) &get_params_init()
- const **VEC** (epiworld_double) &get_statistics_obs()
- const **VEC** (epiworld_double) &get_statistics_hist()
- const **VEC** (bool) &get_statistics_accepted()
- const **VEC** (epiworld_double) &get_posterior_lf_prob()
- const **VEC** (epiworld_double) &get_acceptance_prob()
- const **VEC** (epiworld_double) &get_drawn_prob()
- **VEC** (TData) *get_sampled_data()

Random number generation

Parameters

eng	

- void **set_rand_engine** (std::mt19937 &eng)
- std::mt19937 * **get_rand_endgine** ()
- void **seed** (unsigned int s)
- void **set_rand_gamma** (epiworld_double alpha, epiworld_double beta)
- epiworld_double **runif** ()
- epiworld_double **rnorm** ()
- epiworld_double **rnorm** (epiworld_double mean, epiworld_double sd)
- epiworld_double **rgamma** ()
- epiworld_double **rgamma** (epiworld_double alpha, epiworld_double beta)

3.6.1 Detailed Description

template<typename TData>
class LFMCMC< TData >

Likelihood-Free Markov Chain Monte Carlo.

Template Parameters

TData	Type of data that is generated

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

- include/epiworld/math/lfmcmc.hpp

3.7 Model< TSeq > Class Template Reference

Core class of epiworld.

`#include <model-bones.hpp>`

Public Member Functions

- DataBase< TSeq > & **get_db** ()
- epiworld_double & **operator()** (std::string pname)
- size_t **size** () const
- size_t **get_n_variants** () const
- size_t **get_n_tools** () const
- unsigned int **get_ndays** () const
- unsigned int **get_n_replicates** () const
- void **set_ndays** (unsigned int ndays)
- bool **get_verbose** () const
- void **verbose_off** ()
- void **verbose_on** ()
- int today () const

  The current time of the model.
- void write_data (std::string fn_variant_info, std::string fn_variant_hist, std::string fn_tool_info, std::string fn_tool_hist, std::string fn_total_hist, std::string fn_transmission, std::string fn_transition) const

  Wrapper of `DataBase::write_data`
- std::map< std::string, epiworld_double > & **params** ()
- void reset ()

  Reset the model.
- void **print** () const
- Model< TSeq > && **clone** () const
- void **get_elapsed** (std::string unit="auto", epiworld_double *last_elapsed=nullptr, epiworld_double *total_elapsed=nullptr, std::string *unit_abbr=nullptr, bool print=true) const
- void add_global_action (std::function< void(Model< TSeq > *)> fun, int date=-99)

  Set a global action.
- void **run_global_actions** ()
- void **clear_status_set** ()
- const std::vector< VirusPtr< TSeq > > & **get_viruses** () const
- const std::vector< ToolPtr< TSeq > > & **get_tools** () const

Set the backup object

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

- void **set_backup** ()
- void **restore_backup** ()

Random number generation

Parameters

eng	*Random number generator*
s	*Seed*

- void **set_rand_engine** (std::mt19937 &eng)
- std::mt19937 * **get_rand_endgine** ()
- void **seed** (unsigned int s)
- void **set_rand_gamma** (epiworld_double alpha, epiworld_double beta)
- epiworld_double **runif** ()
- epiworld_double **rnorm** ()
- epiworld_double **rnorm** (epiworld_double mean, epiworld_double sd)
- epiworld_double **rgamma** ()
- epiworld_double **rgamma** (epiworld_double alpha, epiworld_double beta)

Add Virus/Tool to the model

This is done before the model has been initialized.

Parameters

v	*Virus to be added*
t	*Tool 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, epiworld_double preval)
- void **add_virus_n** (Virus< TSeq > v, unsigned int preval)
- void **add_tool** (Tool< TSeq > t, epiworld_double preval)
- void **add_tool_n** (Tool< TSeq > t, unsigned int preval)

Accessing population of the model

Parameters

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

- void **agents_from_adjlist** (std::string fn, int size, int skip=0, bool directed=false)
- void **agents_from_adjlist** (AdjList al)
- bool **is_directed** () const
- std::vector< Agent< TSeq > > * **get_agents** ()
- void **agents_smallworld** (unsigned int n=1000, unsigned int k=5, bool d=false, epiworld_double p=.01)

Functions to run the model

Parameters

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

- void **init** (unsigned int ndays, unsigned int seed)
- void **update_status** ()
- void **mutate_variant** ()
- void **next** ()
- void run ()

  Runs the simulation (after initialization)
- void run_multiple (unsigned int nexperiments, std::function< void(size_t, Model< TSeq > *)> fun=save_run< TSeq >(), bool reset=true, bool verbose=true)

Rewire the network preserving the degree sequence.

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

Parameters

proportion	*Proportion of ties to be rewired.*

Returns

A rewired version of the network.

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

Export the network data in edgelist form

Parameters

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

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

- void **write_edgelist** (std::string fn) const
- void **write_edgelist** (std::vector< unsigned int > &source, std::vector< unsigned int > &target) const

Manage status (states) in the model

The functions `get_status` return the current values for the statuses included in the model.

Parameters

lab	`std::string` *Name of the status.*

Returns

`add_status*` *returns nothing.*

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

- void **add_status** (std::string lab, UpdateFun< TSeq > fun=nullptr)
- const std::vector< std::string > & **get_status** () const
- const std::vector< UpdateFun< TSeq > > & **get_status_fun** () const

- void **print_status_codes** () const

Set the user data object

Parameters

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

- void set_user_data (std::vector< std::string > names)
  [@
- void **add_user_data** (unsigned int j, epiworld_double x)
- void **add_user_data** (std::vector< epiworld_double > x)
- UserData< TSeq > & **get_user_data** ()

Queuing system

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

- void queuing_on ()
  Activates the queuing system (default.)
- void queuing_off ()
  Deactivates the queuing system.
- bool is_queuing_on () const
  Query if the queuing system is on.
- Queue< TSeq > & get_queue ()
  Retrieve the `Queue` object.

Get the susceptibility reduction object

Parameters

v	

Returns

epiworld_double

- void **set_susceptibility_reduction_mixer** (MixerFun< TSeq > fun)
- void **set_transmission_reduction_mixer** (MixerFun< TSeq > fun)
- void **set_recovery_enhancer_mixer** (MixerFun< TSeq > fun)
- void **set_death_reduction_mixer** (MixerFun< TSeq > fun)

Friends

- class **Agent**< **TSeq** >
- class **DataBase**< **TSeq** >
- class **Queue**< **TSeq** >

Tool Mixers

These functions combine the effects tools have to deliver a single effect. For example, wearing a mask, been vaccinated, and the immune system combine together to jointly reduce the susceptibility for a given virus.

- std::vector< epiworld_double > **array_double_tmp**
- std::vector< Virus< TSeq > * > **array_virus_tmp**
- **Model** ()
- **Model** (const Model< TSeq > &m)
- **Model** (Model< TSeq > &&m)
- Model< TSeq > & **operator=** (const Model< TSeq > &m)
- void **clone_population** (std::vector< Agent< TSeq > > &p, bool &d, Model< TSeq > *m=nullptr) const
- void **clone_population** (const Model< TSeq > &m)

Setting and accessing parameters from the model

Tools can incorporate parameters included in the model. Internally, parameters in the tool are stored as pointers to an std::map<> of parameters in the model. Using the `unsigned int` method directly fetches the parameters in the order these were added to the tool. Accessing parameters via the `std::string` method involves searching the parameter directly in the std::map<> member of the model (so it is not recommended.)

The function `set_param()` can be used when the parameter already exists in the model.

The `par()` function members are aliases for `get_param()`.

Parameters

initial_val	
pname	Name of the parameter to add or to fetch

Returns

The current value of the parameter in the model.

- epiworld_double * **p0**
- epiworld_double * **p1**
- epiworld_double * **p2**
- epiworld_double * **p3**
- epiworld_double * **p4**
- epiworld_double * **p5**
- epiworld_double * **p6**
- epiworld_double * **p7**
- epiworld_double * **p8**
- epiworld_double * **p9**
- epiworld_double * **p10**
- epiworld_double * **p11**
- epiworld_double * **p12**
- epiworld_double * **p13**
- epiworld_double * **p14**
- epiworld_double * **p15**
- epiworld_double * **p16**
- epiworld_double * **p17**
- epiworld_double * **p18**
- epiworld_double * **p19**
- epiworld_double * **p20**
- epiworld_double * **p21**
- epiworld_double * **p22**

- epiworld_double * **p23**
- epiworld_double * **p24**
- epiworld_double * **p25**
- epiworld_double * **p26**
- epiworld_double * **p27**
- epiworld_double * **p28**
- epiworld_double * **p29**
- epiworld_double * **p30**
- epiworld_double * **p31**
- epiworld_double * **p32**
- epiworld_double * **p33**
- epiworld_double * **p34**
- epiworld_double * **p35**
- epiworld_double * **p36**
- epiworld_double * **p37**
- epiworld_double * **p38**
- epiworld_double * **p39**
- unsigned int **npar_used** = 0u
- epiworld_double **add_param** (epiworld_double initial_val, std::string pname)
- epiworld_double **set_param** (std::string pname)
- epiworld_double **get_param** (unsigned int k)
- epiworld_double **get_param** (std::string pname)
- epiworld_double **par** (unsigned int k)
- epiworld_double **par** (std::string pname)

3.7.1 Detailed Description

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

Core class of epiworld.

The model class provides the wrapper that puts together `Agent`, `Virus`, and `Tools`.

Template Parameters

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

3.7.2 Member Function Documentation

3.7.2.1 add_global_action()

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

Set a global action.

Parameters

fun	A function to be called on the prescribed dates
date	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.

3.7.2.2 reset()

```
template<typename TSeq = int>
void Model< TSeq >::reset ( )
```

Reset the model.

Resetting the model will:

- clear the database
- restore the population (if `set_backup()` was called before)
- re-distribute tools
- re-distribute viruses
- set the date to 0

3.7.2.3 run_multiple()

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

Parameters

nexperiments	Multiple runs of the simulation

3.7.2.4 write_data()

```
template<typename TSeq = int>
void Model< TSeq >::write_data (
            std::string fn_variant_info,
```

```
            std::string fn_variant_hist,
            std::string fn_tool_info,
            std::string fn_tool_hist,
            std::string fn_total_hist,
            std::string fn_transmission,
            std::string fn_transition ) const
```

Wrapper of `DataBase::write_data`

Parameters

fn_variant_info	Filename. Information about the variant.
fn_variant_hist	Filename. History of the variant.
fn_tool_info	Filename. Information about the tool.
fn_tool_hist	Filename. History of the tool.
fn_total_hist	Filename. Aggregated history (status)
fn_transmission	Filename. Transmission history.
fn_transition	Filename. Markov transition history.

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

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

3.8 PersonTools< TSeq > Class Template Reference

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

- include/epiworld/config.hpp

3.9 Progress Class Reference

A simple progress bar.

`#include <progress.hpp>`

Public Member Functions

- **Progress** (int n_, int width_)
- void **start** ()
- void **next** ()
- void **end** ()

3.9.1 Detailed Description

A simple progress bar.

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

- include/epiworld/progress.hpp

3.10 Queue< TSeq > Class Template Reference

Controls which agents are verified at each step.

`#include <queue-bones.hpp>`

Public Member Functions

- void **operator+=** (Agent< TSeq > *p)
- void **operator-=** (Agent< TSeq > *p)
- epiworld_fast_int **operator[ ]** (unsigned int i) const
- void **set_model** (Model< TSeq > *m)

3.10.1 Detailed Description

template<typename TSeq = int>
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

TSeq	

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

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

3.11 RandGraph Class Reference

Public Member Functions

- **RandGraph** (int N_)

- void **init** (int s)
- void **set_rand_engine** (std::mt19937 &e)
- epiworld_double **runif** ()

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

- include/epiworld/random_graph.hpp

3.12 Tool< TSeq > Class Template Reference

Tools for defending the agent against the virus.

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

Public Member Functions

- **Tool** (std::string name="unknown tool")
- void **set_sequence** (TSeq d)
- void **set_sequence_unique** (TSeq d)
- void **set_sequence** (std::shared_ptr< TSeq > d)
- std::shared_ptr< TSeq > **get_sequence** ()
- TSeq & **get_sequence_unique** ()
- void **set_name** (std::string name)
- std::string **get_name** () const
- Agent< TSeq > * **get_agent** ()
- int **get_id** () const
- void **set_id** (int id)
- void **set_date** (int d)
- int **get_date** () const
- void **set_status** (epiworld_fast_int init, epiworld_fast_int post)
- void **set_queue** (epiworld_fast_int init, epiworld_fast_int post)
- void **get_status** (epiworld_fast_int *init, epiworld_fast_int *post)
- void **get_queue** (epiworld_fast_int *init, epiworld_fast_int *post)

Get and set the tool functions

Parameters

v	*The virus over which to operate*
fun	*the function to be used*

Returns

epiworld_double

- epiworld_double **get_susceptibility_reduction** (VirusPtr< TSeq > v)
- epiworld_double **get_transmission_reduction** (VirusPtr< TSeq > v)
- epiworld_double **get_recovery_enhancer** (VirusPtr< TSeq > v)
- epiworld_double **get_death_reduction** (VirusPtr< TSeq > v)
- void **set_susceptibility_reduction_fun** (ToolFun< TSeq > fun)

- void **set_transmission_reduction_fun** (ToolFun< TSeq > fun)
- void **set_recovery_enhancer_fun** (ToolFun< TSeq > fun)
- void **set_death_reduction_fun** (ToolFun< TSeq > fun)
- void **set_susceptibility_reduction** (epiworld_double *prob)
- void **set_transmission_reduction** (epiworld_double *prob)
- void **set_recovery_enhancer** (epiworld_double *prob)
- void **set_death_reduction** (epiworld_double *prob)
- void **set_susceptibility_reduction** (epiworld_double prob)
- void **set_transmission_reduction** (epiworld_double prob)
- void **set_recovery_enhancer** (epiworld_double prob)
- void **set_death_reduction** (epiworld_double prob)

Friends

- class **Agent**< **TSeq** >
- class **Model**< **TSeq** >
- void **default_add_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_add_tool** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_tool** (Action< TSeq > &a, Model< TSeq > *m)

3.12.1 Detailed Description

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

Tools for defending the agent against the virus.

Template Parameters

TSeq	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

3.13 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 >::iterator **begin** ()
- std::vector< TOOLPTR >::iterator **end** ()
- TOOLPTR & **operator()** (size_t i)
- TOOLPTR & **operator[ ]** (size_t i)
- size_t **size** () const noexcept

Friends

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

3.13.1 Detailed Description

template<**typename TSeq**>
class Tools< **TSeq** >

Set of tools (useful for building iterators)

Template Parameters

TSeq	

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

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

3.14 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 >::const_iterator **begin** ()
- std::vector< TOOLPTR >::const_iterator **end** ()
- const TOOLPTR & **operator()** (size_t i)
- const TOOLPTR & **operator[ ]** (size_t i)
- size_t **size** () const noexcept

Friends

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

3.14.1 Detailed Description

template<**typename TSeq**>
class Tools_const< **TSeq** >

Set of Tools (const) (useful for iterators)

Template Parameters

TSeq	

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

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

3.15 UserData< TSeq > Class Template Reference

Personalized data by the user.

`#include <userdata-bones.hpp>`

Public Member Functions

- **UserData** (Model< TSeq > &m)
- UserData (std::vector< std::string > names)

  Construct a new User Data object.
- std::vector< std::string > & **get_names** ()
- std::vector< int > & **get_dates** ()
- std::vector< epiworld_double > & **get_data** ()
- void **get_all** (std::vector< std::string > *names=nullptr, std::vector< int > *date=nullptr, std::vector< epiworld_double > *data=nullptr)
- unsigned int **nrow** () const
- unsigned int **ncol** () const
- void **write** (std::string fn)
- void **print** () const

Append data

Parameters

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

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

Access data

Parameters

i	*Row (0 through ndays - 1.)*
j	*Column (0 through `ncols()`).*

Returns

epiworld_double&

- epiworld_double & **operator()** (unsigned int i, unsigned int j)
- epiworld_double & **operator()** (unsigned int i, std::string name)

Friends

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

3.15.1 Detailed Description

template<**typename TSeq**>
class UserData< **TSeq** >

Personalized data by the user.

Template Parameters

TSeq	

3.15.2 Constructor & Destructor Documentation

3.15.2.1 UserData()

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

Construct a new User Data object.

Parameters

names	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

3.16 vecHasher< T > Struct Template Reference

Vector hasher.

`#include <misc.hpp>`

Public Member Functions

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

3.16.1 Detailed Description

template<typename T>
struct vecHasher< T >

Vector hasher.

Template Parameters

T	

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

- include/epiworld/misc.hpp

3.17 Virus< TSeq > Class Template Reference

Virus.

`#include <virus-bones.hpp>`

Public Member Functions

- **Virus** (std::string name="unknown virus")
- void **mutate** ()
- void **set_mutation** (MutFun< TSeq > fun)
- const TSeq * **get_sequence** ()
- void **set_sequence** (TSeq sequence)
- Agent< TSeq > * **get_agent** ()
- void **set_agent** (Agent< TSeq > *p, epiworld_fast_uint idx)
- Model< TSeq > * **get_model** ()
- void **set_date** (int d)
- int **get_date** () const
- void **set_id** (int idx)
- int **get_id** () const
- void **set_name** (std::string name)
- std::string **get_name** () const
- std::vector< epiworld_double > & **get_data** ()

Get and set the tool functions

Parameters

v	*The virus over which to operate*
fun	*the function to be used*

Returns

epiworld_double

- epiworld_double **get_prob_infecting** ()
- epiworld_double **get_prob_recovery** ()
- epiworld_double **get_prob_death** ()
- void **post_recovery** ()
- void **set_post_recovery** (PostRecoveryFun< TSeq > fun)
- void **set_post_immunity** (epiworld_double prob)
- void **set_post_immunity** (epiworld_double *prob)
- void **set_prob_infecting_fun** (VirusFun< TSeq > fun)
- void **set_prob_recovery_fun** (VirusFun< TSeq > fun)
- void **set_prob_death_fun** (VirusFun< TSeq > fun)
- void **set_prob_infecting** (epiworld_double *prob)
- void **set_prob_recovery** (epiworld_double *prob)
- void **set_prob_death** (epiworld_double *prob)
- void **set_prob_infecting** (epiworld_double prob)
- void **set_prob_recovery** (epiworld_double prob)
- void **set_prob_death** (epiworld_double prob)

Get and set the status and queue

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

Parameters

init	*After the virus/tool is added to the agent.*
end	*After the virus/tool is removed.*
removed	*After the agent (Agent) is removed.*

- void **set_status** (epiworld_fast_int init, epiworld_fast_int end, epiworld_fast_int removed=-99)
- void **set_queue** (epiworld_fast_int init, epiworld_fast_int end, epiworld_fast_int removed=-99)
- void **get_status** (epiworld_fast_int *init, epiworld_fast_int *end, epiworld_fast_int *removed=-99)
- void **get_queue** (epiworld_fast_int *init, epiworld_fast_int *end, epiworld_fast_int *removed=-99)

Friends

- class **Agent**< **TSeq** >
- class **Model**< **TSeq** >
- class **DataBase**< **TSeq** >
- void **default_add_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_add_tool** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_virus** (Action< TSeq > &a, Model< TSeq > *m)
- void **default_rm_tool** (Action< TSeq > &a, Model< TSeq > *m)

3.17.1 Detailed Description

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

Virus.

Template Parameters

TSeq	

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

3.18 Viruses< TSeq > Class Template Reference

Set of viruses (useful for building iterators)

`#include <viruses-bones.hpp>`

Public Member Functions

- **Viruses** (Agent< TSeq > &p)
- std::vector< VIRUSPTR >::iterator **begin** ()
- std::vector< VIRUSPTR >::iterator **end** ()
- VIRUSPTR & **operator()** (size_t i)
- VIRUSPTR & **operator[ ]** (size_t i)
- size_t **size** () const noexcept

Friends

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

3.18.1 Detailed Description

template<typename TSeq>
class Viruses< TSeq >

Set of viruses (useful for building iterators)

Template Parameters

TSeq	

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

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

3.19 Viruses_const< TSeq > Class Template Reference

Set of Viruses (const) (useful for iterators)

`#include <viruses-bones.hpp>`

Public Member Functions

- **Viruses_const** (const Agent< TSeq > &p)
- std::vector< VIRUSPTR >::const_iterator **begin** ()
- std::vector< VIRUSPTR >::const_iterator **end** ()
- const VIRUSPTR & **operator()** (size_t i)
- const VIRUSPTR & **operator[ ]** (size_t i)
- size_t **size** () const noexcept

Friends

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

3.19.1 Detailed Description

template<**typename TSeq**>
class Viruses_const< **TSeq** >

Set of Viruses (const) (useful for iterators)

Template Parameters

TSeq	

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

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

Index