**Agent-based Virus Simulation in MATLAB**

**1. Description of model**

This Matlab programme is designed to simulate the spread of a virus throughout a population. The two agents which interact within the simulation are a healthy\_human and an infected\_human. Initial population sizes are user-defined and distributed randomly in the square environment, (dimensions = km, size = user-defined).

**1.1 Agent memory and functions**

The memory parameters of healthy humans are:

* **age** - age of agent in number of iterations
* **pos** - current position in cartesian coordinates [x,y]
* **speed** - number of units agent can migrate in 1 iteration
* **inf** - boolean value (1 or 0) for whether agent is infected
* **immunity** - boolean value (1 or 0) for whether agent has immunity

The model is run iteratively (1 iteration = 1 hour - fixed) and for each agent in turn, the following rule set is applied:

The memory parameters of infected humans are:

* **age** - age of agent in number of iterations
* **pos** - current position in cartesian coordinates [x,y]
* **speed** - number of units agent can migrate in 1 iteration
* **asymp** - boolean value (1 or 0) for whether agent is asymptomatic

The current rules can be summarised as:

**Migrate - specified in functions @healthy\_human/migrate.m, @infected\_human/migrate.m\***

SUMMARY OF HEALTHY\_HUMAN MIGRATE RULE

1. It will pick a random migration direction to move in
2. If it will leave the edge of the model, the direction is incremented by 45 degrees and it will try again (up to 8 times)

SUMMARY OF INFECTED\_HUMAN MIGRATE RULE

1. It will pick a random migration direction to move in
2. If it will leave the edge of the model, the direction is incremented by 45 degrees and it will try again (up to 8 times)

*\*credit to Dr Dawn Walker, author of the Ecolab user guide from which this excerpt was taken and adapted*

**Infected - specified in function @healthy\_human/infected.m**

SUMMARY OF HEALTHY\_HUMAN INFECTED RULE

1. Checks that agent doesn’t have immunity
2. Checks that there is at least one infected human within the search radius (if more than one infected human located at same distance then randomly pick one)
3. Probability that infected human will infect healthy human is a function of distance to the closest infected human (prob\_inf = 0.2/d)
4. If prob\_inf>rand (a randomly generated number between 0 and 1) then the healthy human will despawn and respawn as an infected human

**Recover - specified in function @infected\_human/recover.m**

SUMMARY OF INFECTED\_HUMAN RECOVERED RULE

1. Checks to see if the agent has been infected for the full duration of the disease
2. If the agent has been infected for the maximum length of the disease then update the message list and the relevant stats
3. If the agent has not been infected for long enough then increase age by one

**Spawn (healthy\_human) - specified in function @healthy\_human/spawn.m**

SUMMARY OF HEALTHY\_HUMAN SPAWN RULE

1. This function is called if a healthy agent becomes infected
2. An infected agent is spawned in place of the healthy agent
3. This infected agent has a 16% chance of being an asymptomatic carrier

**Spawn (infected\_human) - specified in function @infected\_human/spawn.m**

SUMMARY OF INFECTED\_HUMAN SPAWN RULE

1. This function is called if an infected agent recovers
2. A healthy agent is spawned in place of the infected agent
3. This healthy agent will have immunity from further infection

**1.2 Environment\***

The model environment is represented by a data structure containing the following fields:

* ENV\_DATA.shape - shape of environment - FIXED AS SQUARE
* ENV\_DATA.units - FIXED AS KM
* ENV\_DATA.bm\_size - length of environment edge in km

*\*credit to Dr Dawn Walker, author of the Ecolab user guide from which this excerpt was taken and adapted*

**1.3 Communication and timing\***

The global data structure MESSAGES contains the following fields:

MESSAGES.atype - n x 1 array listing the type of each agent in the model (1=healthy\_human, 2-infected\_human, 0=dead agent)

MESSAGES.pos - list of every agent position in [x y]. Dead agents have the ‘dummy’

position [-1 -1] (this is never accessed).

MESSAGES.dead - n x1 array containing ones for agents that have died in the current iteration

The model is run synchronously i.e. message lists are only updated at the end of each iteration (in update\_messages).

*\*credit to Dr Dawn Walker, author of the Ecolab user guide from which this excerpt was taken and adapted*

**1.4 Randomisation\***

Matlab’s random number generator is used to generate the initial locations of agents, and also within some functions (e.g. in migrate.m, in order to make decisions about which direction to migrate in). This means that if you run two simulations with the same numbers of agents, you may still get different results.

In order to override this, if you set the input into the random\_selection function to be zero (ecolab.m line 41), the same order of random numbers will be generated as for the previous simulation, and you should get the same results for the same initial agent number and environment size.

**Note – randomisation of agent order is NOT implemented in this model**

*\*credit to Dr Dawn Walker, author of the Ecolab user guide from which this excerpt was taken*

**1.5 Simulation control\***

The global structure CONTROL\_DATA contains data to enable “fast mode” to operate and to allow a pause flag to exist. This mode aims to speed up operation by reducing the amount of plotting that occurs as the number of live agents increases.

If fast mode is set to “true” it is also used to stop the program if the number of healthy humans or infected humans reaches zero.

CONTROL\_DATA.fmode\_display\_every – a value describing the iteration interval

between plotting if the fmode variable is set to “true”.

CONTROL\_DATA.fmode\_control – a 2x5 array containing agent no. thresholds and

associated iteration intervals to be used to set CONTROL\_DATA.fmode\_display\_every

CONTROL\_DATA.pause – a Boolean flag which is used to enable pausing within a while loop

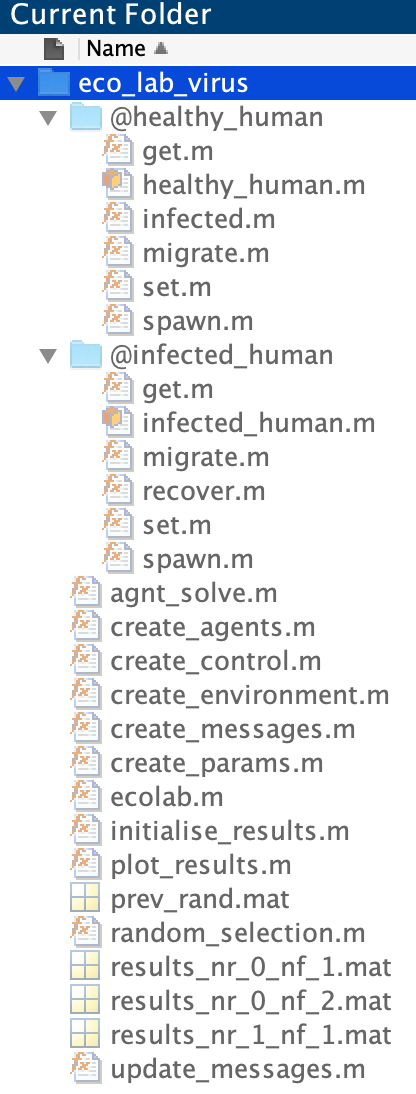
*\*credit to Dr Dawn Walker, author of the Ecolab user guide from which this excerpt was taken and adapted*

**2. File/directory structure\***

The programme files are under a directory called eco\_lab\_virus. The two class declaration

directories (for healthy humans and infected humans) are beneath this main directory:

Inside each class directory is the class declaration function (i.e. healthy\_human.m or infected\_human.m, the object set and get functions and the agent ‘rule’ functions:

****

*\*credit to Dr Dawn Walker, author of the Ecolab user guide from which this excerpt was taken and adapted*

**3. Running the model\***

Copy the zip file onto your PC. Unzip the file.

Open Matlab and change the working directory to eco\_lab\_virus

The function you need to call to run the model is called ecolab, with inputs in the

following form:

**ecolab(size, nh, ni, nsteps, fmode(optional),outImages(optional))**

Where:

*size is the size of the model environment*

*nh is the initial number of healthy humans*

*ni is the initial number of infected humans*

*\*credit to Dr Dawn Walker, author of the Ecolab user guide from which this excerpt was taken and adapted*