



Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Lecture with Computer Exercises:
Modelling and Simulating Social Systems with MATLAB

Project Report

**Zombie Outbreak:
The Effect of Inter-State Collaboration
on the Survival of Humanity**

by

Matthieu G. MOTTET
Basile I. M. WICKY

Zurich
December 2012

Agreement for free-download

We hereby agree to make our source code for this project freely available for download from the web pages of the SOMS chair. Furthermore, we assure that all source code is written by ourselves and is not violating any copyright restrictions.

Matthieu G. MOTTET

Basile I. M. WICKY

Contents

1	Abstract	5
2	Individual contributions	5
3	Acknowledgments	5
4	Introduction and Motivations	6
5	Description of the Model	8
6	Implementation	10
7	Simulation Results and Discussion	10
8	Summary and Outlook	11
9	References	11
10	Appendix	11
10.1	outbreak.m	11
10.2	update.m	16
10.3	plotResults.m	20

1 Abstract

This part needs updating once the rest of the report will be finished

Investigation of the application of the SIR model to a zombie outbreak has already been studied, raising the fear of dark days for humanity. However, we would like to deepen this investigation to a multi-state system to see how interactions between subpopulation may brighten the future of the human race. Moreover, we are interested in seeing to what extent the different paradigms of international politics, Realpolitik, Liberalism and Neoconservatism as defined by Daniel W. Drezner in Theories of International Politics and zombies may lead to different outcomes.

2 Individual contributions

M.G.M. and B.I.M.W. formulated the question in mathematical terms and discussed the implementation in MATLAB. M.G.M. wrote the code. M.G.M. and B.I.M.W. analysed and discussed the results and B.I.M.W. wrote the report.

3 Acknowledgments

We wish to thank Karsten Donnay and Stefano Balietti for their support in our work, fruitful discussions and an open mind to accept such a project. We would also like to thank the Chair of Sociology for the computational support provided with simulation time on the ETH cluster Brutus.

4 Introduction and Motivations

While models of international relationship have already been studied under different pressure components, the effect of a zombie outbreak on international collaboration and equilibrium is a question that has been underestimated and was never addressed to the best of our knowledge. It is remarkable that the effect of such an intense event has not been looked at, although the fear of zombies and the threat they represent is vivid for many of us as reflected by the importance of the zombie culture. Zombies, compared to other unnatural creatures such as vampires or aliens, have the very peculiar property not to be a minority inside the human civilization but rather to be in a way a part of it, just not quite as it was, i.e. zombified. Accordingly, zombies cause a much more deep-rooted fear as they not only threaten our lives but also our very sense of identity as it questions our notion of what humanity is. The psychological effect of such a non-standard threat as well as the repercussion on the behaviour of large population systems such as states should be far from trivial. Accordingly, we decided to simulate an inter-state collaboration models in order to see the outcome on a large scale of such an extreme event. While some people might question the validity of such a study (no zombie has been observed so far), we think that the applicability of such a reasoning could extend to a more probable large-scale epidemiological disaster or simply, give a line of reasoning to cope with what former U.S. Secretary of Defense Donald Rumsfeld referred as the "unknown unknowns" of international security. Zombies might not be real, but the threat and stress they could impose on current world politics is.

Epidemiological models have been widely studied to see the evolution dynamics and spread of a disease in a population [REFERENCE](#). They are based on a mass action model where interaction between susceptibles and infected together with an infection constant define the rate of infection. In the simplest case of such models, infected people can move to the immunized category also following a mass action law. These models have been shown to work in numerous cases and when corrections are added [REFERENCE](#), they can very well represent the evolution and spread of an infectious disease in a population. However, an epidemiological model of communicating population has not been tested to the best of our knowledge.

The origin of the word "zombie" as well as its initial meaning is quite remote from modern pop culture description. The word itself is said to have originated in the voodoo language in the Caribbean [7]. The original description is that of a ritual where the wizard of a tribe would start controlling another person. The fact that this person doesn't have a soul anymore and that it is now under the control of another human being, i.e. it has no longer a free-will, makes it a zombie [7].

Some studies suggest that those "zombified" members of the tribe were actually administered a cocktail of two natural drugs, one being a neurotoxin and the other a hallucinogenic drug [REFERENCE](#). In fact, the neurotoxin damages the brain and turns the person into a vegetative state.

Zombies in popular culture differ a lot from this original etymological definition. The canon of the zombie literature have had numerous description and hypothesis on how they may emerge in a human population, as well as what might characterize them. Since Romero's "night of the living dead" [REFERENCE](#), where zombies were said to have raised from some pseudo-magical event, the depiction and origin of zombies has considerably evolved. Recent zombie stories usually describe the genesis of flesh-eating monsters in an epidemiological sense, typically some sort of virus. Recent examples in the zombie culture are numerous and include (non-exhaustively) "Resident Evil", "28 Days Later". For simplicity, we will treat the emergence and zombification events as linked to an infectious disease of some sort, allowing the treatment of the problem with a modified epidemiological model. The big difference with a standard infectious disease where people usually get immunized, being transformed into a zombie is a on way process. The only way out is death and therefore, humans (susceptibles) can only decrease in our system.

It is interesting to notice that most zombie canon predict a very bleak outcome concerning the fate of humanity. Indeed, most movies/film describe an almost total disappearance of humans and the few survivors are rarely in a position that seem to be about to brighten up. While some might argue that the disappearance of humanity might not be such regretful event and might actually benefit our planet [REFERENCE](#), we decided to see if the usual outcome and fate of the human race in case of a zombie outbreak might differ from those classical scenarios, and if so, under which set of particular conditions.

Mathematical modeling of a zombie outbreak in a single population has previously been simulated [6] but showed very little hope for humans in the case of such an unlikely event. The primary reason for the annihilation of humans in all of the presented scenarios lies in their models. In contrast to "classical" epidemiological models where infected people can recover and although having changed population statues (going from susceptible to removed in a immunized sense of the term), they do not actually die. This is very different for the zombie scenario, where now "removed" is no longer synonym of "immunized", but is actually a very nice way of putting "dead". Accordingly, under the considerations of the model presented, humans can only day and this eventually happens in every case (some set of parameters can give reprieve the inevitable fate).

We rationalized that a more state-based description of the world population

might actually help in brightening the outcome of a zombie outbreak. In fact, the world is divided into states and nations that apply their own laws and restrictions in terms of immigration. If immigration applies to humans, it might as well apply to zombies. In such a case, one might envision that in a given state under the threat of a zombie epidemiological disaster, flux of incoming susceptibles to help them kill the zombies or on the other hand the emigration of the survivors to a non-infected state might lead various outcome. For example, it is imaginable to see the emergence of a zombie-only state where all the remaining survivor would have found shelter in another state. Or even better, that the help of susceptible from another state might help eradicate the new-coming zombie threat.

For those reason we decided to simulate a model of interacting sub-populations, each under an epidemiological-like treatment. The inner-state epidemiological model describes the emergences of zombies from a spreading disease point of view, while the populations fluxes between states would represent immigration/emigration of the populations under concern (humans and/or zombies).

5 Description of the Model

$$\Delta S_i^{micro} = -\alpha S_i Z_i - \gamma S_i Z_i = -(\alpha + \gamma) S_i Z_i \quad (1)$$

$$\Delta Z_i^{micro} = +\alpha S_i Z_i - \beta S_i Z_i = (\alpha - \beta) S_i Z_i \quad (2)$$

$$\Delta R_i^{micro} = +\beta S_i Z_i + \gamma S_i Z_i = (\beta + \gamma) S_i Z_i \quad (3)$$

$$\Delta S_i^{macro} = -\nu S_i \Delta S_i + \sum_{j \neq i} \nu S_j \Delta S_j \quad (4)$$

$$\Delta Z_i^{macro} = -\eta Z_i \tanh\left(\frac{Z_i}{S_i}\right) + \sum_{j \neq i} \eta Z_j \tanh\left(\frac{Z_j}{S_j}\right) \quad (5)$$

For each state (microstate), we define a SZR model that evaluates the evolution of the different populations under studies: susceptibles (S), zombies (Z) and removed (R). Epidemiological-like transfer of populations between the states (at the macrostate level) also occurs as defined above and models the refugees and zombie transfer across states. Those transfers are parameters dependants, which depend on the cost-hypothesis as defined by the Game-Theoretical paradigm implemented. The apparition of zombies in one state will start the game. Each state will then evolve on the domestic and international level. The domestic level will follow a standard SZR model, whereas the international level will introduce exchange in the population of susceptible and zombie between the states. These

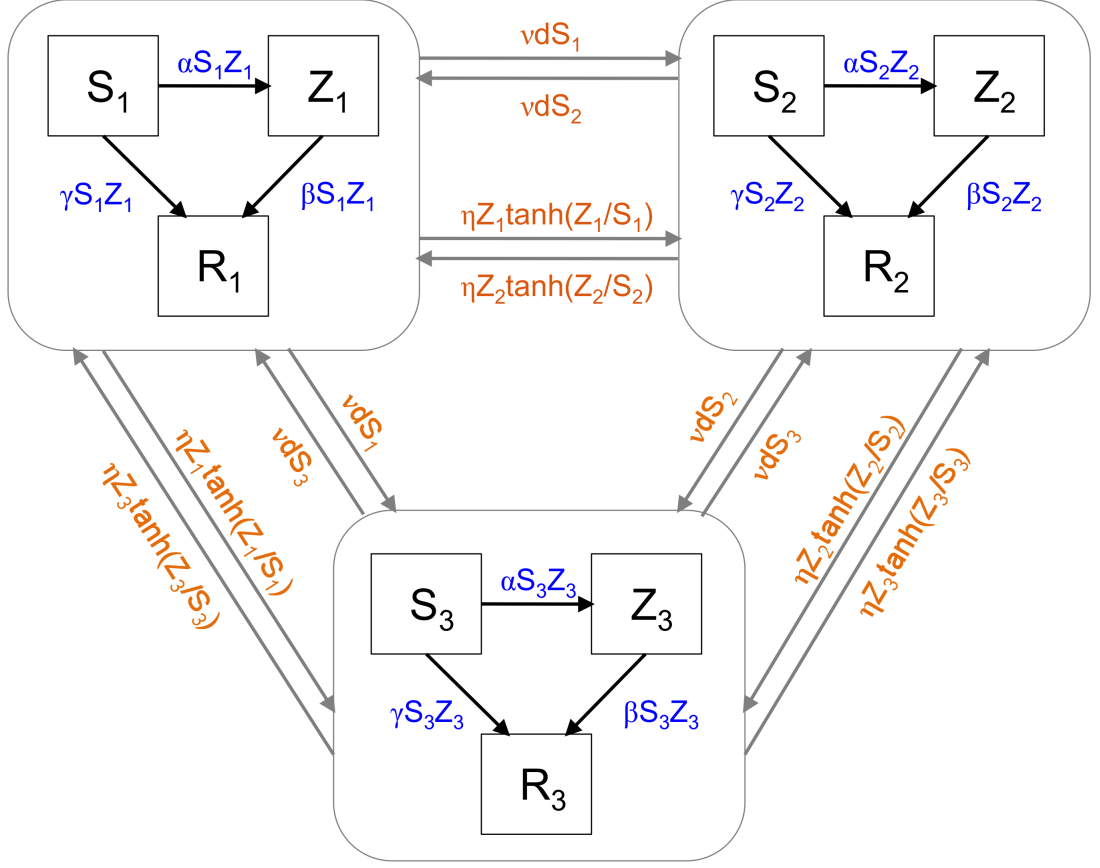


Figure 1: Schematic of our model including all variables. The parts in blue represent the fluxes at the micro-state level while the parts in orange represent the macro-state population fluxes. This schematic represent our simulated system where the interaction of three states (micro-states) is modeled

exchanges will be influenced by the state decisions on foreign policies such as humanitarian or military actions determined by our game theory framework. The Game-Theoretical framework is defined as the possibility of undertaking military action of foreign soil (exporting S) or changing the refugee politics by modifying the mu parameter (allowing more S to come into one's state, and with a collateral cost of having more zombies crossing as well). Each action will be defined with a specific payoff, which in turn will depend on the international cooperation system under scrutiny. For simplicity, we will only model homogenous systems, i.e. all the states will adopt the same international politics paradigm. Finally, we will also introduce a "feedback" loop on the payoff depending on the success of a previously undertaken action (positive or negative affectation of the payoffs). This effect models the psychological effect of a successful or unsuccessful action

on future action, for example the effectiveness of a military attack. This effect will be made as to converge after a certain time to model the wearing out of the psychological effect over time. The system will be implemented as a step-based update. This implies the ignorance of the actors (the states) of the action of the other actors. This rationalisation comes as the idea that the outbreak would occur over a short period of time, forcing for rapid decision-making and therefore not allow a reaction-based decision-making process.

6 Implementation

Research Methods

We would like to tackle this question by using a two-level model. Intra-state populations would be modelled by a standard SIR (Kermack-McKendrick) model or an evolution of it that might include quarantined populations and more evolved parameters. On the next level, the inter-state relationships would be modelled using Game-Theory under different cost-hypothesis related to the main paradigms of international relationships as previously defined. Other

The material necessary for defining our models of zombies and 'zombification' parameters will be extracted from the canon of the zombie popular culture such as World War Z (Brooks, 2006), 28 Days Later (Boyle, 2002), The Night of the Living Dead (Romero, 1968), Zombieland (Fleischer, 2009), Resident Evil (Capcom, 1996), etc.

7 Simulation Results and Discussion

Fundamental Questions

Investigation of the application of the SIR model to a zombie outbreak has already been studied, raising the fear of dark days for humanity. However, we would like to deepen this investigation to a multi-state system to see how interactions between subpopulation may brighten the future of the human race. Moreover, we are interested in seeing to what extent the different paradigms of international politics, Realpolitik, Liberalism and Neoconservatism as defined by Daniel W. Drezner in Theories of International Politics and zombies may lead to different outcomes. Expected Results

As describe in Drezner's book, we expect different equilibrium outcomes depending on the paradigm under consideration. He postulates the possible appearance of zombie states under Realpolitik and Liberalism paradigms while Neoconservatism would not allow such an outcome.

8 Summary and Outlook

blablabla[1]

9 References

- [1] T.C. Reluga. An sis epidemiology game with two subpopulations. *Journal of Biological Dynamics*, 3(5):515–531, 2009.
- [2] P.G. Bennett. Modelling decisions in international relations: game theory and beyond. *Mershon International Studies Review*, pages 19–52, 1995.
- [3] D. Balcan and A. Vespignani. Phase transitions in contagion processes mediated by recurrent mobility patterns. *Nature physics*, 7(7):581–586, 2011.
- [4] S. Funk, M. Salathé, and V.A.A. Jansen. Modelling the influence of human behaviour on the spread of infectious diseases: a review. *Journal of The Royal Society Interface*, 7(50):1247–1256, 2010.
- [5] T.C. Reluga. Game theory of social distancing in response to an epidemic. *PLoS computational biology*, 6(5):e1000793, 2010.
- [6] P. Munz, I. Hudea, J. Imad, and R.J. Smith. When zombies attack!: mathematical modelling of an outbreak of zombie infection. *Infectious Disease Modelling Research Progress. Hauppauge NY: Nova Science Publishers*, pages 133–150, 2009.
- [7] D.W. Drezner. *Theories of international politics and zombies*. Princeton University Press, 2011.

10 Appendix

10.1 outbreak.m

```
1 function dump = outbreak( varargin )
2
3     %
4     % OUTBREAK runs the simulation of a zombie outbreak in a 3 states
5     % system.
6     %
7     % DUMP = OUTBREAK( ARGS )
8     %     ARGS are name-value pairs. Possible arguments are :
9     %     params: structure containing rate parameters (alpha, beta,
10    %             gamma, eta, nu) as arrays (3x1 for alpha, beta and
11    %             gamma or 3x2 for eta and nu).
```

```

12      %      paramfile: path to a mat-file storing the parameter
13      %      structure.
14      %      zombies: 3x1 array containing the number of initial zombie
15      %      per state (default: 0, 0, 0).
16      %      population: 3x1 array containing the number of initial
17      %      susceptible population per state (default: 1000,
18      %      1000, 1000).
19      %      steps: maximal number of steps for the simulation (default:
20      %      1e8).
21      %      silent: level of output for the simulation:
22      %      0 = all outputs
23      %      1 = all outputs excepts graphs
24      %      2 = no outputs
25      %      dslength: size of the sliding window used for the
26      %      calculation of the mean of dS (used as weight
27      %      factor for the inter-state susceptible transfer).
28      %
29      %      DUMP is a structure containing the following fields:
30      %      S: 4xn array containing the evolution of the susceptible
31      %      population.
32      %      dS: 4xn array containing the evolution of the susceptible
33      %      population's variation.
34      %      Z: 4xn array containing the evolution of the zombie
35      %      population.
36      %      dZ: 4xn array containing the evolution of the zombie
37      %      population's variation.
38      %      R: 4xn array containing the evolution of the removed
39      %      population.
40      %      dR: 4xn array containing the evolution of the removed
41      %      population's variation.
42      %      step: number of steps performed.
43      %      alpha, beta, gamma, eta, nu: transfer rate parameters.
44      %      time: simulation time.
45      %
46
47
48      %% Variable initialization
49      silent = 0;
50      dslength = 10;
51      maxSteps = 1e8;
52      rates = struct( 'alpha', [ 0.00095 ; 0.00095 ; 0.00095 ], 'beta', ...
53          [ 0.00025 ; 0.00025 ; 0.00025 ], 'gamma', [ 0.00005 ; 0.00005 ; ...
54          0.00005 ], 'mu', [ 0.00000005 , 0.00000005 ; 0.00000005 , ...
55          0.00000005 ; 0.00000005 , 0.00000005 ], 'nu', [ 0.00000005 , ...
56          0.00000005 ; 0.00000005 , 0.00000005 ; 0.00000005 , 0.00000005 ], ...
57          'eta', [ 0.00000005 , 0.00000005 ; 0.00000005 , 0.00000005 ; ...
58          0.00000005 , 0.00000005 ] );
59      zombies = [ 0 ; 0 ; 0 ; 0 ];
60      populations = [ 1e3 ; 1e3 ; 1e3 ; 3e3 ];
61
62      % Structure storing populations and populations variation
63      states = struct( 'pop', zeros( 4, 3 ), 'dpop', zeros( 4, 3 ) );

```

```

59     % See exit condition at the end of the loop.
60     exitThreshold = 1e-1;
61
62     p = [ 'alpha' ; 'beta ' ; 'gamma' ; 'eta ' ; 'mu ' ; 'nu ' ];
63     reverseStr = '';
64
65
66     %% Argument parsing
67     % All argument are optional.
68     for i = 1:2:nargin
69
70         % Argument are passed in name-value pairs. If i + 1 does not exist,
71         % then the value is absent and the program stops.
72         if( nargin < i + 1 )
73
74             error( [ 'Missing argument for parameter "' varargin{ i } ...
75                     '","' ] );
76
77         end
78
79         switch varargin{ i }
80
81             % params expects a structure containing the rates values for the
82             % epidemiologic model. Not all fields have to be present. If a
83             % field is missing, the default value is used.
84             case 'params'
85                 for j = 1:6
86
87                     if isfield( varargin{ i + 1 }, strtrim( p( j, : ) ) )
88
89                         rates.( strtrim( p( j, : ) ) ) = varargin{ i + 1 ...
90                             }. ( strtrim( p( j, : ) ) );
91
92                     end
93                 end
94
95             % paramfile expects the path to a mat-file containing a
96             % parameter structure (refer to parms).
97             case 'paramfile'
98                 pfile = load( varargin{ i + 1 }, '-mat' );
99                 for j = 1:6
100
101                     if isfield( pfile{ i + 1 }, strtrim( p( j, : ) ) )
102
103                         rates.( strtrim( p( j ) ) ) = pfile{ i + 1 }. ( ...
104                             strtrim( p( j ) ) );
105
106                     end
107                 end
108
109             % zombies expects an array containing the initial zombie
110             % populations in each state.
111             case 'zombies'
112                 temp = varargin{ i + 1 };
113                 for j = 1:3

```

```

109         zombies( j ) = temp( j );
110         zombies( 4 ) = zombies( 4 ) + temp( j );
111     end
112
113     % zombies expects an array containing the initial susceptible
114     % population in each state.
115     case 'population'
116         temp = varargin{ i + 1 };
117         populations( 4 ) = 0;
118         for j = 1:3
119
120             populations( j ) = temp( j );
121             populations( 4 ) = populations( 4 ) + populations( j );
122         end
123
124     % steps expects the maximum number of steps allowed for the
125     % simulation.
126     case 'steps'
127         maxSteps = varargin{ i + 1 };
128
129     % silent except a int defining the output type :
130     % 2 : No output
131     % 1 : All outputs but the graphs
132     % 0 : Normal output
133     case 'silent'
134         silent = varargin{ i + 1 };
135
136     % dslength expects the size of the sliding window used for the
137     % calculation of the mean of dS (used as weight factor for the
138     % inter-state susceptible transfer).
139     case 'dslength'
140         dslength = varargin{ i + 1 };
141
142     otherwise
143         error( [ 'Unknown parameter "', varargin{ i }, "'. ' ] );
144     end
145 end
146
147
148
149 %% Initialization of simulation
150
151 % Setting the initial populations.
152 states.pop( :, 1 ) = populations;
153 states.pop( :, 2 ) = zombies;
154
155 % Initialization of the sliding window for inter-state population
156 % transfer.
157 dshistory = zeros( 3, dslength );
158
159 % Setting up the initial dump structure. Large matrices are used to
160 % avoid the need of dynamic memory allocation to improve code
161 % efficiency.

```

```

162 d = zeros( 4, .001 * maxSteps );
163 dump = struct( 'S', d, 'dS', d, 'Z', d, 'dZ', d, 'R', d, 'dR', d, ...
    'step', 1, 'alpha', mean( rates.alpha ), 'beta', mean( rates.beta ...
    ), 'gamma', mean( rates.gamma ), 'eta', mean( mean( rates.eta ) ...
    ), 'nu', mean( mean( rates.nu ) ), 'time', 0 );
164 dump.S( :, 1 ) = states.pop( 1:4, 1 );
165 dump.Z( :, 1 ) = states.pop( 1:4, 2 );
166 dump.R( :, 1 ) = states.pop( 1:4, 3 );
167
168 % Initialize timer.
169 tic;
170
171 %% Update loop.
172 for i = 1:maxSteps
173
174     % Output of step # and current population count.
175     if silent < 2
176
177         msg = sprintf('Processing step %d... Human population %d, ...
            Zombie population %d\n', i, dump.S( 4, dump.step ), ...
            dump.Z( 4, dump.step ) );
178         fprintf([ reverseStr, msg]);
179
180         reverseStr = repmat(sprintf('\b'), 1, length(msg));
181     end
182
183     % Update function (see update.m for details).
184     [ states, rates, dshistory ] = update( states, rates, dshistory );
185
186     % Dumping of the latest state of the simulation.
187     % Update of the step number.
188     dump.step = i + 1;
189
190     % Dumping of the various population and population variation.
191     dump.S( :, dump.step ) = states.pop( :, 1 );
192     dump.dS( :, dump.step ) = states.dpop( :, 1 );
193     dump.Z( :, dump.step ) = states.pop( :, 2 );
194     dump.dZ( :, dump.step ) = states.dpop( :, 2 );
195     dump.R( :, dump.step ) = states.pop( :, 3 );
196     dump.dR( :, dump.step ) = states.dpop( :, 3 );
197
198     % Stops simulation if S is equal to 0.
199     if states.pop( 4, 1 ) == 0
200
201         if silent < 2
202
203             disp( ' ' );
204             disp( 'End of the human race.' );
205         end
206         break;
207     end
208
209     % Stops simulation if Z is equal to 0.

```

```

210         if states.pop( 4, 2 ) == 0
211             if silent < 2
212                 disp( ' ' );
213                 disp( 'Humanity survived.' );
214             end
215             break;
216         end
217     end
218
219     % Stops simulation if the average fluctuation of both dS and dZ
220     % over a sliding window is lower than the threshold (simulation
221     % reached equilibrium in the limit of our time frame).
222     if i > 100 && mean( abs( dump.dS( 4, ( i - 100 ):i ) ) ) < ...
223         exitThreshold && mean( abs( dump.dZ( 4, ( i - 100 ):i ) ) ) < ...
224         exitThreshold
225
226         if silent < 2
227             disp( ' ' );
228             disp( 'Equilibrium reached.' );
229         end
230         break;
231     end
232 end
233
234 % Store simulation time.
235 dump.time = toc;
236
237 % Resizing of the dump matrices before plotting.
238 dump.S = dump.S( :, 1:dump.step );
239 dump.dS = dump.dS( :, 1:dump.step );
240 dump.Z = dump.Z( :, 1:dump.step );
241 dump.dZ = dump.dZ( :, 1:dump.step );
242 dump.R = dump.R( :, 1:dump.step );
243 dump.dR = dump.dR( :, 1:dump.step );
244
245 if ~silent
246     % Plot the simulation history (see plotResults.m for details).
247     plotResults( dump );
248 end
249
250 end
251 end

```

10.2 update.m

```

1 function [ states, rates, dshistory ] = update( states, rates, ...
        dshistory )

```

```

2
3 %
4 % UPDATE compute the population evolution for one step of the
5 % epidemiological model.
6 %
7 % [ STATES, RATES, DSHISTORY, DUMP ] = UPDATE( STATES, RATES, ...
   DSHISTORY, DUMP )
8 %     STATES is the structure storing the populations and population
9 %     variations as defined in the OUTBREAK function.
10 %     RATES is the structure containing the epidemiological transfer
11 %     rates as defined in the OUTBREAK function.
12 %     DSHISTORY is the array containing the sliding window for
13 %     inter-state susceptible transfer.
14 %
15
16 %% Initialization of parameters :
17
18 s = states.pop( 1:3, 1 );
19 z = states.pop( 1:3, 2 );
20
21 alpha = rates.alpha;
22 beta = rates.beta;
23 gamma = rates.gamma;
24 nu = rates.nu;
25 eta = rates.eta;
26
27 % Permutation matrix used to set the input inter-state flux from the
28 % inter-state output flux to avoid redundant calculations. See flux
29 % update.
30 permutation = [ 0 0 1 ; 1 0 0 ; 0 1 0 ];
31
32 % Matrix containing the coordinates of the input inter-state flux from
33 % each state (line) to the other two (block of two colons). See flux
34 % correction.
35 iFluxCoor = [ 2 2 3 1 ; 3 2 1 1 ; 1 2 2 1 ];
36
37 %% Update of the susceptible population.
38
39 % Depletion of the susceptible population goes according to :
40 %
41 %   S  -> Z  = - alpha * s * z
42 %   S  -> R  = - gamma * s * z
43 %   S1 -> S2 = nu * dSmean
44 %   S1 -> S3 = nu * dSmean
45 %
46 % Immigration is the only source of susceptible population.
47 %
48 % dS is a 3x6 matrix containing for each state :
49 %
50 %   Row 1 : n( S  -> Z  )
51 %   Row 2 : n( S  -> R  )
52 %   Row 3 : n( S1 -> S2 ) | n( S2 -> S3 ) | n( S3 -> S1 )
53 %   Row 4 : n( S1 -> S3 ) | n( S2 -> S1 ) | n( S3 -> S2 )

```



```

54 % Row 5 : n( S2 -> S1 ) | n( S3 -> S2 ) | n( S1 -> S3 )
55 % Row 6 : n( S3 -> S1 ) | n( S1 -> S2 ) | n( S2 -> S3 )
56 %
57 % The sum of each line gives the population variation for the state.
58
59 % Mean variation of the susceptible in the sliding window.
60 dsmean = mean( dshistory, 2 );
61
62 dS = zeros( 3, 6 );
63 dS( :, 1 ) = - alpha .* s .* z;
64 dS( :, 2 ) = - gamma .* s .* z;
65 dS( :, 3 ) = min( nu( :, 1 ) .* dsmean, 0 );
66 dS( :, 4 ) = min( nu( :, 2 ) .* dsmean, 0 );
67 dS( :, 5 ) = - permutation * permutation * dS( :, 4 );
68 dS( :, 6 ) = - permutation * dS( :, 3 );
69
70 % There is the possibility that any of the |-dSi| is larger than the
71 % actual population in the state (negative population fluctuation larger
72 % than the actual population). If so, a correction is applied to
73 % avoid negative population.
74 % As long as any state falls into this category:
75 while sum( sum( dS( :, 1:4 ), 2 ) + s + sum( dS( :, 5:6 ), 2 ) < ...
76         -1e-4 )
77     for i = 1:3
78
79         % Correct the state that falls into this category:
80         if( sum( dS( i, 1:4 ), 2 ) + ( s( i ) + sum( dS( i, 5:6 ), 2 ...
81             ) ) < -1e-4 )
82
83             % Population is considered equal to the sum of previous
84             % population and the inter-state input transfer. The
85             % negative contributions to the dS are reduced so that it
86             % equals this population.
87             dS( i, 1:4 ) = dS( i, 1:4 ) / abs( sum( dS( i, 1:4 ) ) ) ...
88                 * ( sum( dS( i, 5:6 ) ) + s( i ) );
89
90             % The effect of the correction on the output flux is
91             % applied to the other state input flux.
92             dS( iFluxCoor( i, 1 ), 4 + iFluxCoor( i, 2 ) ) = - dS( i, ...
93                 3 );
94             dS( iFluxCoor( i, 3 ), 4 + iFluxCoor( i, 4 ) ) = - dS( i, ...
95                 4 );
96         end
97     end
98     % Note that the output flux can only be lower than the original flux.
99     % Therefore, the input flux for the two other states can only
100     % be smaller than before the correction. Accordingly, the
101     % process has to be repeated until all states have a population
102     % differential such that it doesn't lead to a negative population.
103 end
104
105 %% Update of the zombie population.

```

```

102
103 % Depletion of the zombie populations goes according to :
104 %
105 % S -> Z = alpha * s * z
106 % Z -> R = - beta * s * z
107 % Z1 -> Z2 = - eta * z * tanh( z / s )
108 % Z1 -> Z3 = - eta * z * tanh( z / s )
109 %
110 % Conversion from susceptible and "immigration" are the two sources of
111 % zombies.
112 %
113 % dZ is a 3x6 matrix containing for each states :
114 %
115 % Row 1 : n( S -> Z )
116 % Row 2 : n( Z -> R )
117 % Row 3 : n( Z1 -> Z2 ) | n( Z2 -> Z3 ) | n( Z3 -> Z1 )
118 % Row 4 : n( Z1 -> Z3 ) | n( Z2 -> Z1 ) | n( Z3 -> Z2 )
119 % Row 5 : n( Z2 -> Z1 ) | n( Z3 -> Z2 ) | n( Z1 -> Z3 )
120 % Row 6 : n( Z3 -> Z1 ) | n( Z1 -> Z2 ) | n( Z2 -> Z3 )
121 %
122
123 dZ = zeros( 3, 6 );
124 dZ( :, 1 ) = - dS( :, 1 );
125 dZ( :, 2 ) = - beta .* s .* z;
126 dZ( :, 3 ) = - eta( :, 1 ) .* z .* tanh( z ./ s );
127 dZ( :, 4 ) = - eta( :, 2 ) .* z .* tanh( z ./ s );
128 dZ( :, 5 ) = - permutation * permutation * dZ( :, 4 );
129 dZ( :, 6 ) = - permutation * dZ( :, 3 );
130
131 % The same control procedure as for the susceptible population is
132 % applied to avoid negative population of zombies. See susceptible
133 % correction for details on the procedure.
134 while sum( ( dZ( :, 1 ) + sum( dZ( :, 5:6 ), 2 ) + z ) + sum( dZ( :, ...
135     2:4 ), 2 ) < - 1e-4 )
136
137     for i = 1:3
138
139         if( ( dZ( i, 1 ) + sum( dZ( i, 5:6 ), 2 ) + z( i ) ) + sum( ...
140             dZ( i, 2:4 ) ) < - 1e-4 )
141
142             dZ( iFluxCoor( i, 1 ), 4 + iFluxCoor( i, 2 ) ) = - dZ( i, ...
143                 3 );
144             dZ( iFluxCoor( i, 3 ), 4 + iFluxCoor( i, 4 ) ) = - dZ( i, ...
145                 4 );
146         end
147     end
148
149 %% Update of the removed population

```

```

150 % Variation in the removed population is strictly positive and comes
151 % from both the susceptible and zombie population:
152 %
153 % S -> R = alpha * s * z
154 % Z -> R = beta * s * z
155 %
156 % The value obtained for dS and dZ are used:
157
158 dR = - [ dS( :, 2 ), dZ( :, 2 ) ];
159
160 %% Update each population
161
162 % Compilation of the populations variations
163 states.dpop( 1:3, : ) = [ sum( dS, 2 ), sum( dZ, 2 ), sum( dR, 2 ) ];
164
165 % Update of the sliding window matrix for the susceptible.
166 [ ~, dsSize ] = size( dshistory );
167 dshistory( :, 2:dsSize ) = dshistory( :, 1:( dsSize - 1 ) );
168 dshistory( :, 1 ) = states.dpop( 1:3, 1 );
169
170 % Update of the current populations
171 states.pop( 1:3, : ) = states.pop( 1:3, : ) + states.dpop( 1:3, : );
172
173 % Update of the total populations
174 states.pop( 4, : ) = sum( states.pop( 1:3, : ) );
175 states.dpop( 4, : ) = sum( states.dpop( 1:3, : ) );
176 end

```

10.3 plotResults.m

```

1 function plotResults( dump )
2
3 % PLOTRESULTS plots the populations' evolution during the simulation. It
4 % plots the different populations (S, Z, R) for each state as well as
5 % the population variation. Furthermore the global populations and
6 % population variations are also displayed.
7 %
8 % PLOTRESULTS( DUMP )
9 % DUMP is the dumping structure as defined in the OUTBREAK function.
10 %
11
12 x = 0:( length( dump.S ) - 1 );
13
14 % For each of the three states, plot the populations and
15 % populations' variations.
16 for i = 1:3
17
18     subplot( 4, 4, ( i * 4 - 3 ):( i * 4 - 2 ) );
19     plot( x, dump.S( i, : ), 'g', x, dump.Z( i, : ), 'r', x, dump.R( ...
        i, : ), 'k' );

```

```

20     ylim( [ 0 dump.S( 4 ) ] );
21     xlabel( 'Step' );
22     ylabel( 'Population' );
23     title( [ 'State ', int2str( i ) ], 'fontweight', 'b' );
24     if i == 1
25
26         legend( 'Susceptibles', 'Zombies', 'Removed' );
27     end
28
29     subplot( 4, 4, ( i * 4 - 1 ):( i * 4 ) );
30     plot( x, dump.dS( i, : ), 'g', x, dump.dZ( i, : ), 'r', x, ...
31           dump.dR( i, : ), 'k' );
32     xlabel( 'Step' );
33     ylabel( 'Population Variation' );
34     title( [ 'State ', int2str( i ) ], 'fontweight', 'b' );
35 end
36
37 % Plot the total populations and total populations' variations.
38 subplot( 4, 4, 13:14 );
39 plot( x, dump.S( 4, : ), 'g', x, dump.Z( 4, : ), 'r', x, dump.R( 4, : ...
40       ), 'k' );
41 ylim( [ 0 dump.S( 4 ) ] );
42 xlabel( 'Step' );
43 ylabel( 'Population' );
44 title( 'World population', 'fontweight', 'b' );
45
46 subplot( 4, 4, 15:16 );
47 plot( x, dump.dS( 4, : ), 'g', x, dump.dZ( 4, : ), 'r', x, dump.dR( ...
48       4, : ), 'k' );
49 xlabel( 'Step' );
50 ylabel( 'Population Variation' );
51 title( 'World population', 'fontweight', 'b' );
52 end

```