Evaluation of Healthcare Coverage Efficiency Using Agent-Based Simulation

Joe Dinius
University of Arizona
ECE508

October 13, 2009

Abstract

A hypothetical disease is developed and spread through an agent population via a simple interaction rule. Metrics for cost of infection, resource utilization, and time until epidemic ends are developed for agents with varying degrees of healthcare coverage. Results are presented showing efficiencies of public and private insurance options; in particular, emphasis is placed on the dichotomy between non-profit and for-profit insurance companies.

1 Introduction

Agent-based simulation has become a widespread technique for evaluating complex system behavior. In a typical agent-based simulation, complex and unpredictable behavior emerges from simple interactions between agents. It is normally assumed that agents have the following properties [6]:

- They can react timely and flexibly to the dynamic and unexpected changes in their environment.
- They have an autonomous and independent behavior, which is not controlled by any external entity.
- They can take the initiative and perform proactively actions that may help them to reach their goals.
- They can communicate with users or other agents. Thus, they can exchange information, engage in complex negotiations, and coordinate their activities to cooperate in the joint resolution of a problem.
- Agents usually have reasoning, planning and learning capabilities that allow them to display an intelligent behavior.

These properties make agent-based simulation a wonderful tool for evaluating problems in genetics, biology, energy production and distribution, and healthcare. This paper focuses on the last topic.

Healthcare in the United States is of particular concern nowadays. President Obama believes healthcare reform is a critical step in the recovery of our economy (Babington, AP/Washington Times, 10/4). While this may be true, little scientific evidence has been presented to support such a hypothesis. Data are compiled and published in the yearly **Health, United States** [5] compendium showing healthcare expenses, number of uninsured, privately- and publicly-insured American citizens and other metrics associated with public health and welfare (proliferation of sexually transmitted diseases, flu and other infectious diseases and the associated resource utilization for such infectious diseases). While these data are useful, the numbers are very misleading. Firstly, only averages (means) are presented for each of the metrics. Statistically, averages provide only integrated information concerning diverse populations. Therefore, what little information they do provide can only be used as a control variable in any experiment; the statistical "spread" of the data is not understood, therefore fix the parameter for which the averaged data is available and randomly vary other

quantities. This is unsatisfactory for many reasons; primarily, one is usually interested in the variation of all parameters so that he can establish a statistical meaning to the behavior that he is witnessing. This paper attempts to address the concern; what is "good" starting information to confirm or disconfirm Obama's hypothesis? A big part of addressing this issue will be the establishment of control variables.

The main control variable presented in this paper is a common hypothetical disease, X, requiring common treatment techniques (at a different cost for each level of insurance coverage; none, public, and private). Through such a development, it is possible to evaluate the efficiency of different combinations of healthcare options among a sample population of similar individuals, or agents. The focus of this paper will be on three metrics:

- What is the total cost of treatment?
- How many public resources are required to end an epidemic of X?
- How much time does it take for an epidemic of X to end?

Answering the above questions will provide important information about the efficiency of healthcare systems representative of both the current and proposed systems. The goal of this paper is to use agent-based simulation to answer these questions.

The agents for simulations presented in this paper are simple. They are autonomous, with a social network defined by their nearest neighbors on the grid. The only interaction modeled between agents is the transfer of disease X from an infected agent to an uninfected nearest neigbor. Through such a simple interaction, the study of proliferation of X can be simulated and understood statistically.

2 Review of the Literature

The body of available literature that is concerned with agent-based simulation applications to healthcare is primarily concerned with information management between different healthcare providers[6],[2]. This work is integral in the development of more systemic solutions to the provision of healthcare, however it is not useful for the purposes of this paper.

To address the concerns of this paper, it was necessary to go outside the body of work in agent-based simulation to find real-world examples of healthcare at work. Such examples are presented in [1],[3]. In [1], a case study of fee-for-service (FFS) vs. managed care is presented for a sample of low-income pregnant women in California. The paper presents cost and care statistics showing a negative correlation of managed care to neonatal death and birth weight; i.e. despite the availability of more care options, managed care had a negative impact on basic birth qualities. This paper provides statistics showing cost factors of private vs. public insurance with a concrete example. In [3], a case study of uninsured vs. privately insured persons involved in automobile accidents is presented. Several interesting factors are presented in this paper. Of particular interest to the aims of this paper are the conclusions that uninsured persons received less care, spent less time in the hospital and were billed more per day of treatment than their insured counterparts. Additionally, the uninsured have a higher mortality rate. Together, these two papers, along with [5], provide information for the development of the comparative model used in this paper. Cost and treatment structures are developed from empirical data and used as inputs to the simulation.

3 Hypotheses

The current problems with healthcare in America appear to stem from the growing percentage of the population that has no health insurance. Typically the uninsured make too much money to qualify for state- or federally-provided healthcare options, but too little to pay for premiums from private companies. From this information, it is expected that the higher the percentage of agents who are uninsured, the longer it

will take for an epidemic of X to end. From [3], it is expected that time-to-recovery of each agent will be longer for uninsured agents, and therefore the previous hypothesis seems justified. However, with more coverage comes more resource utilization and more cost. Therefore, it is expected in a population with a low percentage of uninsured agents that cost will be high as well as the number of days in treatment. It is expected that the optimal solution will, unfortunately, be developed using cost as a primary factor. In this case, the best solution is expected to be one where a small percentage of the population will remain uninsured. This would ensure lower cost, less resource utilization, and a manageable time to eliminate the X epidemic.

4 Description of the Model

The model is essentially a standard disease propagation model [4], with a few updates. The simulation is first initialized with a random number draw to determine the attributes of each agent. These attributes are

- Insurance status (tied to cost)
- Susceptibility to X
- Time to be cured of X
- Treatment time.

Then, a randomly selected agent on the grid is infected with X, this is at Day 0. This agent is infectious to his 8 nearest neighbors who either contract the disease or not. Whether or not they contract the disease at this step is determined by their susceptibility to the disease. After one day of infection, each infected agent is admitted to the hospital. When an agent is in the hospital, its social network is broken and they are no longer infectious to nearest neighbors. The amount of time each agent spends in the hospital is determined by the random number draw at initialization. If treatment time is greater than the amount of time needed to cure the agent of X, the treatment time is reset to the cure time. Otherwise, the agent is released when its treatment time is up. At this time, the agent is either cured of X when it goes back to its social network, meaning it is no longer infectious, or it remains infectious and its time to be cured increases by a scale factor multiplied by the number of additional treatment days needed. This process is repeated until the number of infected agents reaches 0. This represents one run. Each run is then repeated a number of times, with different random number seeds, to establish statistically relevant data to do hypothesis testing.

4.1 Pseudocode

The pseudocode is written as follows:

1. Initialize Monte Carlo parameters using a common random number seed
2. for $i = 1$, number of Monte Carlo runs
    - Initialize agent parameters
    - Introduce X to a randomly chosen agent on the grid
    - while (the number of agents infected is greater than 0)
        (a) Identify infected agents
        (b) for $j = 1$, number of infected agents
            – Identify if infected agents are in the hospital or not. If they are, go to the next infected agent, if not, continue.
            – Find nearest neighbors to infected agent (employs periodic boundary conditions)
            – for $k = 1$, number of nearest neighbors

i. Do random number draw to see if current neighbor becomes infected
ii. Reset current neighbor's susceptibility to X if they become infected

(c) for $l = 1$, number of infected agents
i. Check for admission of current agent to hospital
ii. If admitted, record the day admitted

(d) Identify agents who are in the hospital

(e) for $m = 1$, number of agents in the hospital
– Release agents from the hospital if their treatment time is up
– If agent released before cured, add time for the agent to be cured

(f) Calculate total number of infected agents

- end while

3. Iterate random number seed repeatably for next Monte Carlo run

4. end for

5 Parameters

The control variables for the simulation experiments are:

- Cost per day of treatment for uninsured, publicly insured and privately insured (both for-profit and non-profit options). Table 1 shows the values used.

Coverage	Treatment Cost per Day	Treatment Time μ	Treatment Time σ
Uninsured	1.05	6	4
Publicly Insured	1.00	9	2
Privately Insured, For-Profit	0.8284	10	3
Privately Insured, For-Profit	0.8011	10	4

Table 1: Cost Parameters

- The amount of time to be cured of X is normally distributed with mean 10 and standard deviation 6. There is a hard limit at 0 to ensure that cure time is non-negative.

- The susceptibility to X is normally distributed with mean 0.6 and standard deviation 0.12. Hard limits are imposed if the random number draw gives numbers that are negative or greater than 1. After infection, susceptibility to X goes to 0.

- The ratio of privately-insured agents remains fixed, at 0.6. The profit margin to reduce cost by for non-profit coverage is 3.3 percent [8].

- The scale factor that multiplies time remaining to be cured if released from hospital before fully treated remains fixed, at 2.

- The time period after infection but before hospital admission remains fixed, at 1.

The variational parameters used for the simulation experiments are:

- The ratio of uninsured vs. publicly-insured agents varies between 0.0 and 0.2 dependently. That is, the sum of the ratios of uninsured, publicly- and privately-insured agents must be 1.

6 Results

To understand how X propagates through the agent population, it is best to look at snapshots of grid points infected at different points in time. Figure 1 shows the initialization of a 100×100 grid.

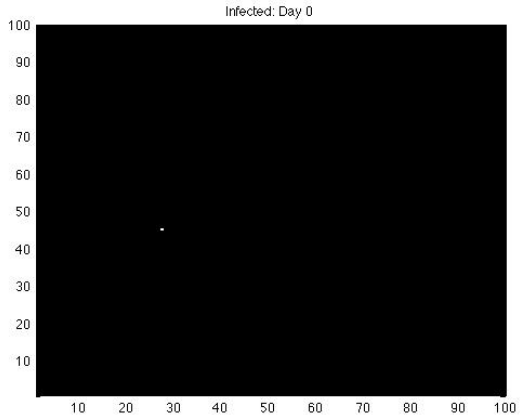

Figure 1: Infected Agents at Day 0

Seven days later, X is spread outward radially as seen in Figure 2.

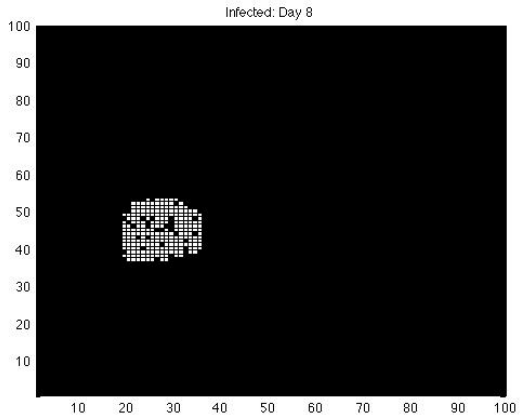

Figure 2: Infected Agents at Day 7

Evolution at later times is shown in Figures 3 and 4. At Day 20, the radial spread is still visible. Worth noting are the uninfected "islands" that emerge around the center of initial infection. These areas are a reflection of the different treatment rules applied to agents on the grid. At Day 53, the enforcement of the periodic boundary conditions is clear. At Days 81 and 98, the infection has been reduced to mostly single, unconnected agents. Because agents cannot be reinfected with X, and there is a high probability that each agent has a finite time to be cured of the disorder, the number of infected agents goes to 0 a few days later.

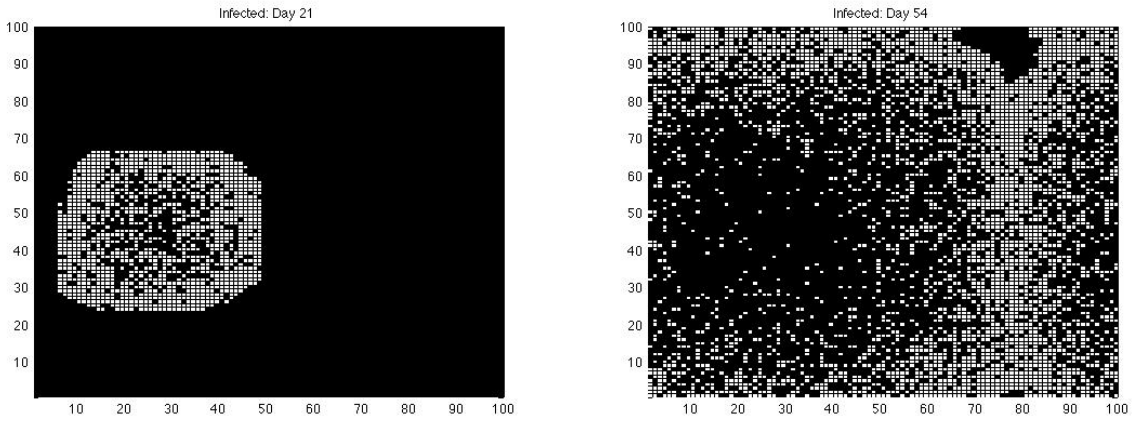

Figure 3: Infected Agents at Days 20 and 53

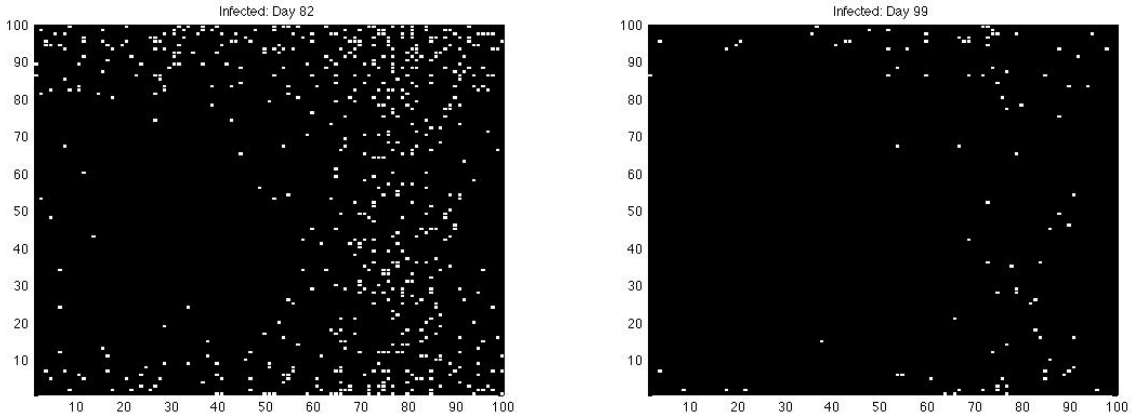

Figure 4: Infected Agents at Days 81 and 98

The examples shown above are for a 100×100 grid. Such a grid size is suitable for individual experiments, however the number of computations involved per run are very limiting for a suitable Monte Carlo analysis. For multiple runs, a more manageable 25×25 grid is used. Results using data reduced from 100 Monte Carlo runs are presented in the subsequent paragraphs.

From a statistical standpoint, it is expected that the parameters of interest (cost, treatment time, and epidemic duration) will be normal; that is, in the continuous limit (as the number of runs goes off to ∞) the output of the Monte Carlo analysis will be represented by Gaussian probability density functions. Figures 5 and 6 show sample histograms with comparison to the continuous normal distributions with μ and σ derived from sampled data. The two plots show the cost and cure statistics for each of the two cases of (i) 0.0 uninsured ratio and (ii) 0.16 uninsured ratio. Each of these cases assumes that 0.6 is privately insured by for-profit insurance.

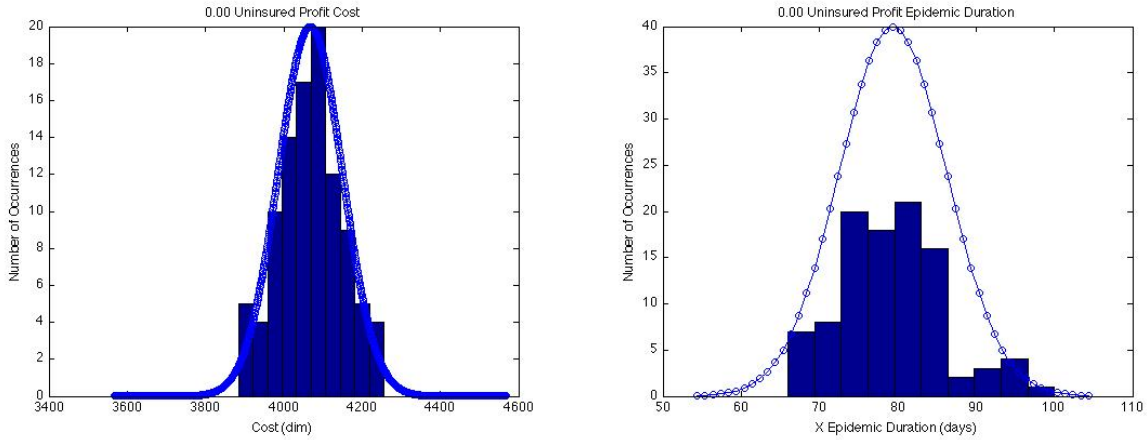

Figure 5: Cost and Cure Parameters: 0.00 Uninsured Ratio

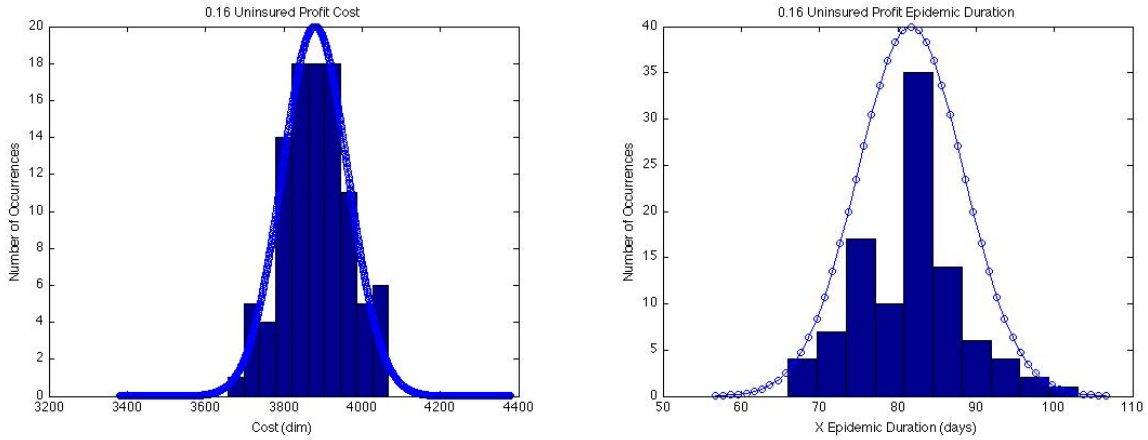

Figure 6: Cost and Cure Parameters: 0.16 Uninsured Ratio

Not surprisingly, the notions of cost and epidemic duration are closely linked, with a high-degree of correlation. Table 2 shows the results of the for-profit simulations.

	0.00	0.04	0.08	0.12	0.16	0.20
Cost Mean	4067.91	4019.41	3971.98	3926.50	3879.99	3834.07
Cost σ	81.96	81.82	82.11	84.34	84.13	82.28
Treatment Mean	4545.78	4491.36	4438.23	4386.00	4334.37	4282.58
Treatment σ	94.52	93.55	95.64	97.78	96.94	93.04
Epidemic Mean	79.37	80.34	80.80	80.73	81.66	81.77
Epidemic σ	6.87	7.24	7.25	6.94	6.78	6.78

Table 2: For-Profit Statistics

Quite surprisingly, the difference between an uninsured ratio of 0 and 0.2 is only about 2 days; in terms of relative difference, the difference is ≈ 0.025. That is, despite the widely varying treatment times for the different levels of agent coverage, the net result, for 625 agents, is only about 2 days of additional epidemic duration on average. Despite this negligible difference in epidemic duration, hospital utilization and cost drop by 0.05 to 0.07. In terms of sensitivity, the system seems to be fairly insensitive to a change in ratio

of uninsured to publicly-insured. An interesting comparison would be: how do the results compare for the better care-providing and less expensive non-profit insurance option? Table 3 shows these results.

	0.00	0.04	0.08	0.12	0.16	0.20
Cost Mean	3932.51	3883.75	3836.71	3790.04	3742.51	3693.89
Cost σ	85.91	86.00	87.16	88.59	87.38	86.62
Treatment Mean	4471.66	4416.71	4363.27	4310.78	4257.11	4202.38
Treatment σ	100.96	101.15	102.35	104.47	103.92	103.59
Epidemic Mean	80.19	81.50	82.14	82.72	83.18	83.42
Epidemic σ	6.62	6.88	7.14	7.11	6.95	6.81

Table 3: Non-Profit Statistics

The comparison of result from Tables 2 and 3 shows only a slight difference in cost (about 0.03), and a statistically-insignificant difference in treatment mean. An interesting difference is the larger epidemic duration mean. However, for the most part, differences in the two tables are negligible.

The output of the simulations show that, given the parameters for propagation of X, there are only slight differences between for-profit and non-profit coverage concerning cost and resource utilization. Additionally, there is little relative difference between epidemic duration for a spread of ratios of uninsured agents ranging from 0.0 and 0.2. These results are counterintuitive when considering the large differences in treatment time parameters for the different coverage options. However, it is reasonable to infer that the large transients in simulation runs are integrated out over several Monte Carlo runs leading to benign differences across different simulation experiments.

7 Verification/Validation

Verification tasks were performed to ensure the simulation functions as it is supposed to. These tasks were performed on a simple 4×4 grid. One agent on the grid, at position $(2, 2)$, is used as a paradigm. This agent is privately-insured.

- Check Day 1 spread of the disease as a result of the first agent. This step will verify proper spread of infection throughout the entire agent population.

  Result: The expected number of newly infected agents should be roughly the mean (0.6) susceptibility times the number of neighboring agents (8) which would nominally be 6. Fixing the susceptibility at 0.6 verifies this step, as 6 agents became infected. □

- Check that the broken social network as a result of being admitted to the hospital.

  Result: This step was verified using an if-else conditional. No new agents in an infected agent's network are infected if the particular agent is in the hospital. □

- Check for admission of agents to hospital.

  Result: The agent at grid position $(2, 2)$ was admitted to the hospital on Day 3, one day after it was infected on Day 2. □

- Check for proper discharge from hospital.

  Result: The agent at grid position $(2, 2)$ was released from the hospital 7 days after admission. This time was insufficient to cure this agent of the disease. Therefore, when returned to the grid, this agent was still infectious. □

- Check for proper cure of each agent.

  Result The agent at grid position $(2, 2)$ had a nominal cure time of 10 days. Its treatment time was 6 days. Therefore, according to the rule, the agent will be cured in 10 + 2*4 = 18 days. This number was observed in simulation. □

- Check for proper exit of simulation.

  Result When all agents have been cured, the infected agent counter decrements to 0 and the simulation exits. □

The model was developed to test a hypothetical disease propagation model; as such, no validation tasks were performed on this simulation. Data were compiled to create a logical model of cost and care for different levels of available healthcare coverage.

8 Sensitivity Analysis

The results previously presented in this paper seem to indicate that the model is insensitive to random variation in the Monte Carlo parameters and also to a change in the ratio of uninsured agents. As was mentioned previously, this comes as a slight surprise. However, the measures presented are integrated over a large array of agents, which eliminates the weight of low probability transients on the Monte Carlo experiments. However, on a run-to-run basis, there is still a question of sensitivity. This question is addressed by looking at the ratio of infected agents as a function of time for three runs with different initial seeds. Figure 7 shows these data: The data show that the maximum ratio of infected agents occurs within a small

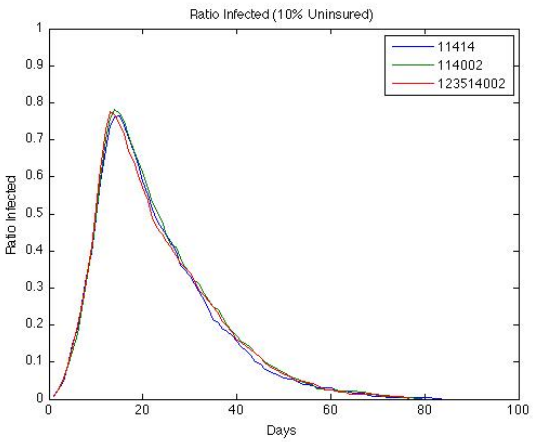

Figure 7: Sensitivity Analysis Plot

(1-3 day) window. There is small variation as the curves descend. Additionally, these two facts combine to show that the system is insensitive to random number seed as well. The spread and cure of the disease are only moderately stochastic, even though there are appreciably large differences in treatment options and associated times for agents to be cured.

9 Conclusion

A model was presented for evaluating efficiency of healthcare coverage options in a macroscopic sense using agent-based simulation techniques. Results were presented showing no statistically-significant difference in cost, resource utilization, or epidemic duration as a result of changing the ratio of uninsured agents; even despite the fact that there is significant difference in treatment statistics for these agents. For large grids of interacting agents, the lack of significant differences in collected statistics stems from the integration of large data sets. Simulations involving large groups of agents typically have transients with very low weights on the overall output. Simulations involving only a few (< 100) agents show a higher degree of stochasticity because low probability transients have a larger impact on the results when observed.

From results presented herein, it is reasonable to conclude that, in a macroscopic sense, Obama's hypothesis is incomplete at best and incorrect at worst. Certainly, the model presented in this paper is very simple; however, it models basic differences between the different healthcare options and how these differences would apply to a hypothetical epidemic. These basic differences, when simulated, show similar levels of containment of X in a macroscopic sense. For different disorders requiring different levels of treatment, the conclusions reached using this model are most definitely invalid. However, these results show that, perhaps, the considerations being made for public healthcare are taking a too microscopically-focused approach.

In the future, it would be interesting to investigate, using the same model, varying levels of susceptibility, numbers of privately-insured individuals and cost structures. Also, it would be interesting to look at competing epidemics, each with different cost, treatment and cure parameters.

10 Matlab Source Code

simExec.m

```
close all; clear all; clc
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%       script simExec                                                  %
%       Executive level run control for the HealthcareSim               %
%       function.  This script defines the statistical                  %
%       parameters used in HealthcareSim and iterates the               %
%       random number seed for repeatable Monte Carlo                   %
%       experiments. This script also records end-of-run                %
%       statistics for analysis.                                        %
%                                                                       %
%       outputs-                                                        %
%       cost         = total cost for the Nmc Monte Carlo               %
%                      experiments                                      %
%       daysPandemic = number of days until there are no                %
%                      infected agents left for the Nmc Monte           %
%                      Carlo experiments                                %
%       hospDays     = number of hospital days utilized                 %
%                      for the Nmc Monte Carlo experiments              %
%                                                                       %
%       Joe Dinius                                                      %
%       Prof: Miklos Szilagyi                                           %
%       ECE508 Project                                                  %
%       29 Sept 2009                                                    %
%%%%%%%%%%%%%%%%%%%%%%c%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%global simulation parameters
global treatmentCosts; %contains 4 costs for each of the insured
                       %categories and percentages of population at each
                       %cost level
global cureStats; %contains mean/sigma data for how long it takes to be
                  %cured of X
global treatmentStats; %contains mean/sigma data for how X is treated
global susceptibilityStats; %contains mean/sigma data for agents
                            %susceptibility to X
global nAgentRows; %number of rows on the agent grid
global nAgentCols; %number of columns on the agent grid
global incubationPeriod; %time after infection before going to hospital
global gamma; %multiplication factor for amount of time to be disease-free
              %if released before before cured


%set statistical parameters
treatmentCosts = [1 .8284 1.05 .8011;
                  .3 .6 .1 0];

cureStats = [10 6];
```

```

treatmentStats = [9 10 6 10;
                  2  3  4 4];

susceptibilityStats = [.6 .12];

%set array size and other important parameters
nAgentRows = 25;
nAgentCols = 25;
incubationPeriod = 1;
gamma = 2;

%Monte Carlo setup parameters
Nmc          = 100; %no. of Monte Carlo runs to perform
seed         = 11414;
daysPandemic = zeros(Nmc,1);
cost         = zeros(Nmc,1);
hospDays     = zeros(Nmc,1);

for i = 1:Nmc

    %call HealthcareSim
    [Agents daysPandemic(i)] = HealthcareSim(seed,0);

    %collect end-of-run statistics
    [cost(i) hospDays(i)]    = collectStats(Agents);

    %increment seed in a repeatable way
    rand('seed',seed);
    seed = round(seed + 1000*rand);

end
```

HealthcareSim.m

```
function [Agents day]= HealthcareSim(seed,movieFlag)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%       function [Agents day] = HealthcareSim(seed,movieFlag)  %
%       sets up a model for evaluating cost and resource       %
%       utilization for a hypothetical healthcare response     %
%       to a new disease X introduced to a population of       %
%       agents with different levels of healthcare coverage.   %
%                                                              %
%       inputs-                                                %
%       seed      = seed for random number generator           %
%       movieFlag = flag to invoke option of making movie      %
%                   (set to 1 to record movie of infected      %
%                    agents on the grid)                       %
%       outputs-                                               %
%       Agents    = structure representing key parameters for  %
%                   agents on the grid                         %
%       day       = number of days until there are no infected %
%                   agents left                                %
%                                                              %
%       Joe Dinius                                             %
%       Prof: Miklos Szilagyi                                  %
%       ECE508 Project                                         %
%       29 Sept 2009                                           %
%%%%%%%%%%%%%%%%%%%%%%c%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%global simulation parameters
global treatmentCosts; %contains 4 costs for each of the insured
                       %categories and percentages of population at each
                       %cost level
global cureStats; %contains mean/sigma data for how long it takes to be
                  %cured of X
```

```
global treatmentStats; %contains mean/sigma data for how X is treated
global susceptibilityStats; %contains mean/sigma data for agents
                            %susceptibility to X
global nAgentRows; %number of rows on the agent grid
global nAgentCols; %number of columns on the agent grid
global incubationPeriod; %time after infection before going to hospital
global gamma; %multiplication factor for amount of time to be disease-free
              %if released before before cured

%initialize the random number generators
rand('seed',seed);
randn('seed',seed);

%set attributes for each agent on the grid based on statistical parameters
%from calling routine
costAttributeDraw             = rand(nAgentRows,nAgentCols);
cureAttributeDraw             = cureStats(1) + cureStats(2) ...
                                *randn(nAgentRows,nAgentCols);
susceptibilityAttributeDraw   = susceptibilityStats(1) + ...
                                 susceptibilityStats(2) ...
                                *randn(nAgentRows,nAgentCols);

%initialize agent attributes to zero
for i = 1:nAgentRows

    for j = 1:nAgentCols

        Agents(i,j).isInfected     = 0;
        Agents(i,j).dayInfected    = 0;
        Agents(i,j).dayCured       = 0;
        Agents(i,j).inHospital     = 0;
        Agents(i,j).dayAdmitted    = 0;
        Agents(i,j).dayDischarged  = 0;
        Agents(i,j).costPerDay     = 0;
        Agents(i,j).susceptibility = 0;
        Agents(i,j).nomCureTime    = 0;
        Agents(i,j).treatmentTime  = 0;
        Agents(i,j).cureTime       = 0;

    end

end

%initialize agents based on random number draws from above
for i = 1:nAgentRows

    for j = 1:nAgentCols

        %initialize treatment time based upon differences due to healthcare
        %coverage
        sum = 0;
        for k = 1:length(treatmentCosts(1,:))
            if ( sum <= costAttributeDraw(i,j) && ...
                 costAttributeDraw(i,j) < sum + treatmentCosts(2,k) )
                Agents(i,j).costPerDay = treatmentCosts(1,k);
            end
            sum = sum + treatmentCosts(2,k);
        end

        if ( Agents(i,j).costPerDay == treatmentCosts(1,1) )
            Agents(i,j).treatmentTime = round(treatmentStats(1,1) + ...
                treatmentStats(2,1)*randn);
        elseif ( Agents(i,j).costPerDay == treatmentCosts(1,2) )
            Agents(i,j).treatmentTime = round(treatmentStats(1,2) + ...
                treatmentStats(2,2)*randn);
        elseif ( Agents(i,j).costPerDay == treatmentCosts(1,3) )
            Agents(i,j).treatmentTime = round(treatmentStats(1,3) + ...
                treatmentStats(2,3)*randn);
```

```
100         else
101             Agents(i,j).treatmentTime = round(treatmentStats(1,4) + ...
102                 treatmentStats(2,4)*randn);
103         end
104
105         %if random number draw is negative, zero it out
106         if (Agents(i,j).treatmentTime < 0)
107            Agents(i,j).treatmentTime = 0;
108         end
109
110         %nomCureTime represents amount of time to be cured of X, provided
111         %they are not released early from the hospital
112         Agents(i,j).nomCureTime    = round(cureAttributeDraw(i,j));
113
114         %if random number draw is negative, zero it out
115         if (Agents(i,j).nomCureTime < 0)
116            Agents(i,j).nomCureTime = 0;
117         end
118
119         %don't allow over-treatment of agents
120         if (Agents(i,j).treatmentTime > Agents(i,j).nomCureTime)
121             Agents(i,j).treatmentTime = Agents(i,j).nomCureTime;
122         end
123
124         %initialize actual cure time to be the nominal until simulation
125         %runs and identifies actual cure times
126         Agents(i,j).cureTime       = Agents(i,j).nomCureTime;
127
128         %initialize agents' susceptibility to X
129         Agents(i,j).susceptibility = susceptibilityAttributeDraw(i,j);
130
131         %although improbable that the random number draw gives an
132         %out-of-bounds number, scale susceptibility to be in [0 1] window
133         if (Agents(i,j).susceptibility < 0)
134             Agents(i,j).susceptibility = 0;
135         elseif (Agents(i,j).susceptibility > 1)
136             Agents(i,j).susceptibility = 1;
137         end
138
139     end
140
141 end
142
143 %initialize day count for start of X pandemic
144 day = 0;
145
146 %introduce X to a randomly selected agent on the grid
147 i = round(nAgentRows*rand);
148 j = round(nAgentCols*rand);
149
150 %guard agains 0 indices
151 if (i == 0)
152     i = 1;
153 end
154 if (j == 0)
155     j = 1;
156 end
157
158 Agents(i,j).isInfected = 1;
159 totalInfectedAgents = 1;
160
161 %record Day 0 infected plot and begin movie recording
162 if (movieFlag)
163     for i = 1:nAgentRows
164         for j = 1:nAgentCols
165             grid(i,j) = Agents(i,j).isInfected;
166         end
167     end
```

```
    figure(1);
    pcolor(grid)
    colormap(gray(2))
    title(['Infected: Day ' num2str(day)]);
    pause;
    M(day+1) = getframe;
end

%start simulation
while (totalInfectedAgents > 0)

    %iterate day count
    day = day + 1;

    %identify infected agents
    indices = findElem(Agents,'isInfected');

    %identify nearest neighbors to infected from above
    for i = 1:length(indices(:,1))
        %if agent is already in the hospital, they are not contagious to
        %neighboring agents
        if ( Agents(indices(i,1),indices(i,2)).inHospital ~= 1 )
            indexu = indices(i,1)-1;
            indexd = indices(i,1)+1;
            indexl = indices(i,2)-1;
            indexr = indices(i,2)+1;

            %apply periodic boundary conditions
            if (indexl == 0)
                indexl = nAgentCols;
            end
            if (indexr == nAgentCols + 1)
                indexr = 1;
            end
            if (indexu == 0)
                indexu = nAgentRows;
            end
            if (indexd == nAgentRows + 1)
                indexd = 1;
            end

            %store indices of nearest neighbors
            nbrIndices= [indexu       indexl;
                         indices(i,1) indexl;
                         indexd       indexl;
                         indexd       indices(i,2);
                         indexd       indexr;
                         indices(i,1) indexr;
                         indexu       indexr;
                         indexu       indices(i,2)];

            %loop over nearest neighbors
            for j = 1:length(nbrIndices(:,1))
                %perform random number draw to identify if neighbor becomes
                %infected and record the day the infection occurs
               if ( rand < ...
                Agents(nbrIndices(j,1),nbrIndices(j,2)).susceptibility )
                Agents(nbrIndices(j,1),nbrIndices(j,2)).isInfected  = 1;
                Agents(nbrIndices(j,1),nbrIndices(j,2)).dayInfected = day;

                %reset susceptibility after infection has already
                %occurred (agent cannot contract X more than once)
                Agents(nbrIndices(j,1),nbrIndices(j,2)).susceptibility = 0;
               end

            end
        end

```

```
236     end
237
238     %check for hospital admission
239     for i = 1:length(indices(:,1))
240         if ( Agents(indices(i,1),indices(i,2)).dayInfected == ...
241                 day - incubationPeriod )
242             Agents(indices(i,1),indices(i,2)).inHospital = 1;
243             Agents(indices(i,1),indices(i,2)).dayAdmitted = day;
244         end
245     end
246
247     %find agents who are in the the hospital
248     indices_hosp = findElem(Agents,'inHospital');
249
250     %put in code guard against indices_hosp being empty (i.e. make sure
251     %that there are agents in the hospital)
252     if (~isempty(indices_hosp))
253      for i = 1:length(indices_hosp(:,1))
254
255       %release treated patients from hospital if there treatment time
256       %is up
257       if (day > Agents(indices_hosp(i,1),indices_hosp(i,2)).treatmentTime ...
258               + Agents(indices_hosp(i,1),indices_hosp(i,2)).dayAdmitted)
259           Agents(indices_hosp(i,1),indices_hosp(i,2)).dayDischarged = day;
260           Agents(indices_hosp(i,1),indices_hosp(i,2)).inHospital = 0;
261       end
262
263       %add time to be cured if agent is released before fully cured
264       if (Agents(indices_hosp(i,1),indices_hosp(i,2)).nomCureTime > ...
265               Agents(indices_hosp(i,1),indices_hosp(i,2)).treatmentTime)
266           Agents(indices_hosp(i,1),indices_hosp(i,2)).cureTime = ...
267           gamma*(Agents(indices_hosp(i,1),indices_hosp(i,2)).nomCureTime ...
268            - Agents(indices_hosp(i,1),indices_hosp(i,2)).treatmentTime) ...
269            + Agents(indices_hosp(i,1),indices_hosp(i,2)).nomCureTime;
270             end
271
272         end
273     end
274
275     %check for when infected agents are cured
276     for i = 1:length(indices(:,1))
277         if (day > Agents(indices(i,1),indices(i,2)).cureTime ...
278                 + Agents(indices(i,1),indices(i,2)).dayInfected)
279            Agents(indices(i,1),indices(i,2)).isInfected = 0;
280            Agents(indices(i,1),indices(i,2)).dayCured = day;
281         end
282     end
283
284     %calculate total number of infected agents
285     indices_inf = findElem(Agents,'isInfected');
286     if (isempty(indices_inf))
287         totalInfectedAgents=0;
288     else
289         totalInfectedAgents = length(indices_inf(:,1));
290     end
291     infected(day) = totalInfectedAgents/(nAgentRows*nAgentCols);
292
293     %record figure of infected agents at the current day
294     if (movieFlag)
295         for i = 1:nAgentRows
296             for j = 1:nAgentCols
297                 grid(i,j) = Agents(i,j).isInfected;
298             end
299         end
300         figure(1);
301         pcolor(grid)
302         title(['Infected: Day ' num2str(day)]);
303         pause
```

```
        M(day+1) = getframe;
    end

end
save s123514002 day infected
plot(1:day,infected)
%render the object M into a movie file
if (movieFlag)
    movie2avi(M, 'agentsInfected.avi','fps',6,'compression','None');
end
```

collectStats.m

```
function [cost hospDays] = collectStats(structureName)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%       function indices = collectStats(structureName)         %
%       gather output from Healthcare sim                      %
%                                                              %
%       inputs—                                                %
%       structureName = name of structure                      %
%                                                              %
%       outputs—                                               %
%       cost     = total cost                                  %
%       hospDays = total number of days agents were in         %
%                  hospital                                    %
%                                                              %
%       Joe Dinius                                             %
%       Prof: Miklos Szilagyi                                  %
%       ECE508 Project                                         %
%       29 Sept 2009                                           %
%%%%%%%%%%%%%%%%%%%%%c%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%identify needed global parameters
global nAgentRows;
global nAgentCols;

%initialize sums
cost     = 0;
hospDays = 0;

for i = 1:nAgentRows

    for j = 1:nAgentCols

        cost     = cost     ...
        + structureName(i,j).costPerDay*structureName(i,j).treatmentTime;
        hospDays = hospDays + structureName(i,j).treatmentTime;

    end

end
```

findElem.m

```
function indices = findElem(structureName,elementName)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%       function indices = findElem(structureName,elementName)          %
%       find indices where structureName.elementName is equal           %
%       to 1                                                            %
%                                                                       %
%       inputs-                                                         %
%       structureName = name of structure                               %
%       elementName   = string of desired variable name for             %
%                       lookup                                          %
%                                                                       %
%       outputs-                                                        %
%       indices = indices where structureName.elementName               %
%                 is 1                                                  %
%                                                                       %
%       Joe Dinius                                                      %
%       Prof: Miklos Szilagyi                                           %
%       ECE508 Project                                                  %
%       29 Sept 2009                                                    %
%%%%%%%%%%%%%%%%%%%%%%c%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%initially, indices matrix is empty, and if no updates are made it will
%stay empty
indices = [];

%initialize array index
count = 1;

for i = 1:length(structureName(:,1))
    for j = 1:length(structureName(1,:))
        if ( eval(['structureName(' num2str(i) ...
                ',' num2str(j) ').' elementName ' == 1']) )
           indices(count,:) = [i,j];
           count = count + 1;
        end
    end
end
```

References

[1] Anna Aizer, Janet Currie and Enrico Moretti, The Review of Economics and Statistics **89**(**3**), 385 (2007).

[2] Martin Beer, Richard Hill and Andrew Sixsmith, in *Applications of Software Agent Technology in the Health Care Domain*, edited by Antonio Moreno and John Nealon (Whitestein Series in Software Agent Technologies, 2000), p. 19.

[3] Joseph J. Doyle, Jr., The Review of Economics and Statistics **87**(**2**), 256 (2005).

[4] Joshua M. Epstein and Robert Axtell. *Growing Artificial Societies: Social Science from the Bottom Up.* Brookings Institute Press, 1996.

[5] **Health, United States, 2008**, *Table 131 (page 1 of 3). Expenses for health care and prescribed medicine, by selected population characteristics: United States, selected years 1987-2005.*

[6] John Nealon and Antonio Moreno, in *Applications of Software Agent Technology in the Health Care Domain*, edited by Antonio Moreno and John Nealon (Whitestein Series in Software Agent Technologies, 2000), p. 3.

[7] William B. Rouse, Systems Research and Behavioral Science **26**, 573 (2009).

[8] http://freemarketmojo.wordpress.com/2009/08/13/profit-margin-health-insurance-industry-ranks-86/