OSTI Phase 2: A Multi Scale Model of Influenza

Jackie Ang
Jonathan Brooks-Bartlett
Alexander Erlich
James Mbewu
Robert Ross

January 2013
University of Oxford

Contents

1 Introduction **1**

1.1 Infection and the Immune Response . . . 1

1.2 Influenza Epidemiology and Interventions . . . 2

2 What We Inherited **3**

2.1 The Hancioglu Model . . . 3

2.2 Software, GUI and Documentation . . . 4

2.3 Praise and Blame are All the Same . . . 5

3 What We Did **7**

3.1 Technical Improvements . . . 7

3.2 Biological and Model improvements . . . 8

3.2.1 Quarantining . . . 8

3.2.2 Treatment . . . 8

3.2.3 Vaccination . . . 9

3.2.4 Stochastic Susceptibility and Quantified Interaction . . . 9

3.3 Implementation of Stochastic Differential Equations . . . 9

3.3.1 Euler-Maruyama . . . 10

3.3.2 Predictor-Corrector Euler–Maruyama . . . 10

3.3.3 Predictor-Corrector Fourth Order Runge-Kutta . . . 11

4 Results **12**

4.1 The Initial Infection . . . 12

4.2 Randomisation of Spread of Infection . . . 12

4.3 Susceptability . . . 13

4.4 Interventions . . . 14

4.5 Stochasticity . . . 17

5 Discussion **19**

Chapter 1

Introduction

Influenza is a viral disease caused by the influenza virus family of viruses. The most common of these viruses is the influenza A virus, which has caused 3 major pandemics in the 20th century and most recently caused the H1N1 pandemic in 2009. These pandemics have had very high mortality, with the 1918-1919 “Spanish Flu”pandemic killing an estimated 50-100 million people and the 1968-1969 “Hong Kong Flu”outbreak killing an estimated 1 million people.

Influenza viruses are RNA viruses which are about 80-120nm in diameter and are spherical in shape. They contain a membranous envelope made up of the proteins haemagglutinin and neuraminidase, both of which have several subtypes and are essential to the naming of the viral strain. For example, H1N1 stands for haemagglutinin subtype 1, neuraminidase subtype 1 influenza A virus. These proteins also act as antigens for the human immune system to work against.

Human symptoms of infection with the influenza virus include fever, cough, nasal congestion, body aches, fatigue and headaches.[4] These symptoms often lead to the loss of the ability to work and a reduction in economic output of infected people in addition to possible mortality. Thus, the study of influenza as well as the modeling of its epidemiology is of importance to us.

Our predecessors have taken a model of host immunity towards influenza by Hancioglu et al.[2] and built a spatial population model based on it. These models will be discussed later within this report. In addition, we have built on both of these models to incorporate responses towards the spread of influenza in a population as well as introduced stochasticity into some of the processes in the model.

1.1 Infection and the Immune Response

The influenza virus spreads via contact between people or via aerosols such as expelled mucus produced by sneezing. It attacks the epithelial cells in the human respiratory tract[3] and hijacks their replication and transcription machinery to produce more viral particles, in the end leading to the death of the host cell and the release of the viral particles to the bloodstream.

The haemagglutinin also acts as an antigen and triggers the immune response in humans. Antigen presenting cells take up viral antigens and presents them to T-helper cells by forming a major histocompatibility complex(MHC) and displaying these antigens. This leads to three major immune responses. The first is the production of interferons, which cause the fever response and binds to epithelial cells, making them resistant to viral infection by interfering with the viral replication process. Interferons also upregulate the production of MHC molecules and thus hastens the immune response.[5]

Antigen presenting cells are also able to stimulate the production of antibodies by lymphocytes and can also stimulate effector cells such as cytotoxic T cells, which in turn stimulate macrophages and other phagocytes which clear the viral particles.

1.2 Influenza Epidemiology and Interventions

The spread of influenza has been a target for many epidemic models, which are mainly based on the SIR model for infectious disease modeling.[1] Public health responses by governments have been included in these models to explore the effect of certain forms of intervention on the spread of the epidemic. Our model explores the effects of vaccination, treatment and self-imposed quarantine on the spread of influenza across a field of evenly spaced-out grid elements, each representing an individual with the disease kinetics modeled using the Hancioglu model.

Vaccination is the introduction of non-pathogenic antigenic material into an organism. It has the effect of stimulating the adaptive immune system of an individual and is often used to prevent the spread of infectious diseases such as smallpox, measles and chickenpox. Influenza vaccines are currently in widespread use as a countermeasure to annual seasonal influenza epidemics and are renewed every year as the strain dominant in each epidemic is different. The effectiveness of the influenza vaccine has been estimated at 50-70%, where it is defined that people who received this vaccine are 50-70% less likely to require medical attention when infected with the influenza virus.

The treatment of influenza is another important countermeasure which governments use to prevent the spread of influenza. The current frontline treatment for influenza are neuraminidase inhibitors such as Tamiflu(oseltamivir). These drugs prevent the spread of influenza virus between cells by preventing the lysis of cells, a process mediated by oseltamivir.

Lastly, another barrier to the spread of influenza is a self-imposed or government-imposed quarantine of infected individuals. As influenza causes symptoms which lead to the reduction of mobility of infected individuals, they often go into self-imposed quarantine when symptoms appear and remove themselves from this quarantine when the symptoms disappear.

Chapter 2

What We Inherited

2.1 The Hancioglu Model

The Hancioglu model simulates disease kinetics within an individual. The influenza virus V infects healthy cells H and turns them into infected cells I. These infected cells die after the viral life cycle progresses and newly formed viruses break out of the human cell. Dead cells D stimulate antigen presenting cells M to exert their effects, which include the production of interferons F and antibodies A . These interferons bind to other healthy cells and make them resistant R to further viral invasion. The full model reads:

$$\frac{\mathrm{d}V}{\mathrm{d}t} = \gamma_V I - \gamma_{VA} SAV - \gamma_{VH} HV - a_V V - \frac{a_{V1} V}{1 + a_{V2} V} \tag{2.1}$$

$$\frac{\mathrm{d}H}{\mathrm{d}t} = b_{HD} D(H + R) + a_R R - \gamma_{HV} VH - b_{HF} FH \tag{2.2}$$

$$\frac{\mathrm{d}I}{\mathrm{d}t} = \gamma_{HV} VH - b_{IE} EI - a_I I \tag{2.3}$$

$$\frac{\mathrm{d}M}{\mathrm{d}t} = (b_{MD} D + b_{MV} V)(1 - M) - a_M M \tag{2.4}$$

$$\frac{\mathrm{d}F}{\mathrm{d}t} = b_F M + c_F I - b_{FH} HF - a_F F \tag{2.5}$$

$$\frac{\mathrm{d}R}{\mathrm{d}t} = b_{HF} FH - a_R R \tag{2.6}$$

$$\frac{\mathrm{d}E}{\mathrm{d}t} = b_{EM} ME - b_{EI} IE + a_E(1 - E) \tag{2.7}$$

$$\frac{\mathrm{d}P}{\mathrm{d}t} = b_{PM} MP + a_P(1 - P) \tag{2.8}$$

$$\frac{\mathrm{d}A}{\mathrm{d}t} = b_A P - \gamma_{AV} SAV - a_A A \tag{2.9}$$

$$\frac{\mathrm{d}S}{\mathrm{d}t} = rP(1 - S) \tag{2.10}$$

$$D = 1 - H - R - I \tag{2.11}$$

where

- $V > 0$ is the viral load per epithelial cell,

- $0 < H < 1$ is the proportion of healthy cells,
- $0 < I < 1$ is the proportion of infected cells,
- $0 < M < 1$ is the activated antigen presenting cells per homeostatic level,
- $F > 0$ are the interferons per homeostatic level of macrophages,
- $0 < R < 1$ is the proportion of resistant cells,
- $E > 0$ is the effector cells per homeostatic level,
- $P > 0$ is the plasma cells per homeostatic level,
- $A > 0$ are the antibodies per homeostatic level,
- $S > 0$ is the antigen distance .

and the lower case variables are different rate constants (more details in [2]).

An epithelial cell can either be healthy, infected, dead, or resistant (i.e. a cell which was infected and then recovered). Therefore, (2.11) describes cell conservation:

$$H + I + D + R = 1\,. \tag{2.12}$$

2.2 Software, GUI and Documentation

The code we inherited is meaningfully split into Matlab functions: There is a core computational routine (`Coupled_Multiscale` and its derivatives `_fast` and `_parallelised`) which calls ODE related functions (`SolveODESystem` which, in turn, calls `ODE_function`). The core routine is called from the GUI (`gui2` script). A number of the files are duplicates (e.g. `_old.m` files) or derivatives.

As for the core algorithm (`Coupled_Multiscale`), the idea is to solve the system (2.1) – (2.11) for times $0 \leq t \leq 1$ which basically corresponds to the human body reacting to a virus over the course of a day. The solution is for a grid of $N \times N$ individuals. The numerical algorithm is `ode45` and some alternative treatments of the differential equations are provided (`Coupled_Multiscale_parallelised` attempts a `parfor` solution, `Coupled_Multiscale_Fast` uses a precomputed solution for V).

Next, the spread of infection between an individual and its neighbors is considered. This is achieved with a convolution: The viral load V (also an $N \times N$ matrix) is convoluted with a kernel

$$\texttt{infectionKernel} = \frac{1}{8}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} \tag{2.13}$$

using the Matlab line

$$\texttt{transmitV} = \texttt{convn}(\texttt{variable.V}(:,:,\texttt{indextime}), \texttt{infectionKernel},'\texttt{same}'); \tag{2.14}$$

which is actually a very elegant solution: The current (i.e. `indextime`) $N \times N$ state matrix V (a struct) is convoluted with the 3×3 sized `infectionKernel`, returning an $N \times N$ matrix `transmitV` (this is made sure by the `"same"` property.

These two operations (iterating ODEs and convoluting with a kernel) correspond to a day. The code loops through several days, thus simulating viral spread.

Additionally, a GUI is provided, allowing to configure grid size and number of days to be simulated and giving fast access to two numerical approaches: `Coupled_Multiscale` and `Coupled_Multiscale_Fast`. The initial conditions are set by point-and-clicking.

Documentation is also provided, giving an outline of both the scientific and the computational problem.

2.3 Praise and Blame are All the Same

This section reviews the strengths and weaknesses of the work on which we expanded. On the plus side

- The report is generally well written. In particular, the introduction is quite clear. It would have been even better if variable names relating to the mathematical model were added to the introductory discussion but the introduction is nevertheless very helpful.
- The glossary is just awesome.
- The idea of using Matlab's convolution function (see line 2.14) is very good use of Matlab's capabilities. To be honest, `conv()` could have made our phase 1 code (constructing the inversion matrix D, see maths section) somewhat easier.
- The results section is interesting. In particular, the analysis of the program runtime made us debate about optimisation (MEX it or not?) and we decided that it is more rewarding to study and implement SDEs instead. In addition, we discovered that the fast version of the computational core (`Coupled_Multiscale_Fast`) uses a precomputed solution of the ODE system 2.1 – 2.11 which turned out to be an interesting approach. The nonlinearity and stiffness of the equations makes this perturbational approach somewhat questionable and there were also some implementational issues – e.g. that a person could not ever get healthy after infection – but the idea is interesting and made us understand the system better.
- The code is sufficiently commented and its structure was fairly easy to grasp. The code is meaningfully split into Matlab functions: There is a core computational routine (`Coupled_Multiscale` and its derivatives `_fast` and `_parallelised`) which calls ODE related functions (`SolveODESystem` which, in turn, calls `ODE_function`). The core routine is called from the GUI (`gui2` script). A number of the files are duplicates (e.g. `_old.m` files) or derivatives and they can be identified easily as such.
- The GUI makes sense and works well. In particular, the selection of initial conditions by clicking is intuitive (It is not intuitive, however, in which order to use the elements of the gui, see below). The GUI code is clear, too.
- Licensing makes sense: BB–CY–SA for documentation, GPL for code. We used that too.

On the minus side,

- We foud the description of the Hancioglu model (section 2 in inherited paper) rather unhelpful. The description in the original paper [2] is actually more straightforward and our discussion in section 2.1 follows their style.
- On a more practical note, it would have been useful if there were a more detailed description of how to run the GUI: Either as a more extended section in the report or (even better) as a static text in the actual GUI. The elements of the GUI, even though they work well, must be operated in a certain order and unitl the user has discovered this order by trial and error he will experience a good share of Matlab's red warm greetings and beeps.

Chapter 3

What We Did

3.1 Technical Improvements

The existing code was generally very well structured and only one major improvement to the performance of the code was implemented. Because the system of ODEs used to model the influenza virus and the immune response are stiff, a stiff solver was used to solve the differential equations. In the matlab code this meant replacing

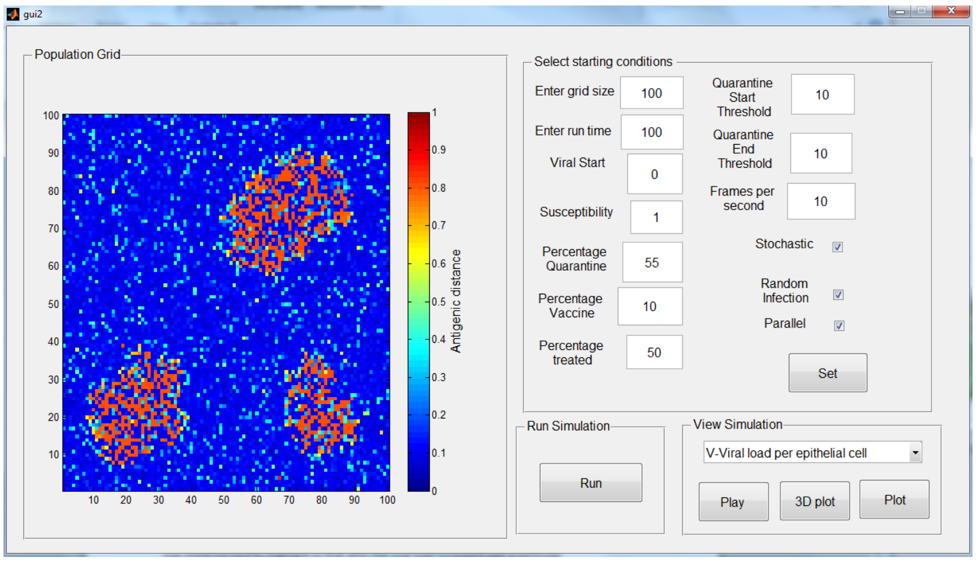
$$\texttt{ode45} \longrightarrow \texttt{ode15s} \tag{3.1}$$

and this single modification reduced the solving time drastically to 0.084556 seconds.

The use of a significantly faster solver made it possible to simulate a grid of people with the influenza virus and their effect on the people surrounding them. Previously this grid was implemented using a predefined viral load in each grid element. In our simulation the system of equations was solved for each grid element, using the previous viral load and received viral load from the eight neighbouring grid elements as initial conditions for the system. The simulation was iterated each day.

Because the system needed be solved at each iteration, parallelism (as previously implemented) was extremely useful. To implement this fully it was necessary to change the struct of parameter values to an array of parameter values that could be accessed from within a parfor loop. Parallelising the code showed significant improvement in computational time as will be shown below. This made it possible to simulate large populations of 10 000 individuals over 100 days on a 4 core processor. On a parallel cluster it would be possible to potentially simulate millions of individuals increasing the biological relevance of the model.

The GUI was improved by a adding a number of parameters for the additions noted below (see figure 3.1). Output for all of the state variables of the system of ODEs was also added. This greatly aided visualisation of both the viral load (indicating when a person is symptomatic) and the antigenic distance (indicating which people had been infected and whether they had been treated or not).



Enter grid size	100
Enter run time	100
Viral Start	0
Susceptibility	1
Percentage Quarantine	55
Percentage Vaccine	10
Percentage treated	50
Quarantine Start Threshold	10
Quarantine End Threshold	10
Frames per second	10
Stochastic	✓
Random Infection	✓
Parallel	✓

V-Viral load per epithelial cell

Figure 3.1: State of the GUI at the end of phase 2.

3.2 Biological and Model improvements

In addition to these technical improvements, a number of biological elements were added to both the system of equations and the grid simulation of a population of people.

3.2.1 Quarantining

Generally, when a person gets influenza and they begin showing symptoms, they stay at home in bed or don't go to work and do not interact with many people; they are in effect quarantined. This was implemented by making it so that after the viral load increased past a particular threshold, the person was quarantined and could not pass on or receive viral load from any of the neighbouring grid elements. After the viral load had reduced past a threshold, the person was released from quarantine and was able to transmit and receive viral load from neighbouring grid elements. This is biologically valid as viral load is a good indicator of a person being symptomatic. The threshold value can be chosen in the GUI and a value of 10×10^{10} virus particles per ml respiratory epithelial cells was selected (approximately 10% of the maximal viral load that is achieved using the Hancioglu model.

3.2.2 Treatment

We amended the Hancioglu model by including the effect of treatment. This was achieved by including an extra linear term in the $\frac{dV}{dt}$ equation that reduced the rate of viral load increase. The $\frac{dV}{dt}$ equation 2.1 then becomes

$$\frac{\mathrm{d}V}{\mathrm{d}t} = \cdots - \alpha_T V. \tag{3.2}$$

This models a class of antiviral treatments called neuraminidase inhibitors (such as Tamiflu) that reduce viral load increase by inhibiting the function of viral neuraminidase protein, which helps the virus break out or bud from the cell it has infected. In terms of implementation this involved adding the term to the ODE_function.m file as well as adding a parameter that controls the strength of action of the drug in reducing viral load. It was found that a parameter value $\alpha_T = 5.0$ was sufficient to qualitatively model the action of this class of drugs.

3.2.3 Vaccination

Every year, many people get vaccination against the new strain of influenza. This was modelled by increasing the initial antigenic distance of an individual. Antigenic distance, being a measure of the immune memory of the individual, correctly models the immune memory that results from being vaccinated. It was found that a value of $S(0) = 0.35$ produced qualitatively correct results.

3.2.4 Stochastic Susceptibility and Quantified Interaction

Because no individual makes the same amount of interactions with his/her environment, we implemented stochastic susceptibility. This amounted to randomly varying the amount of viral load an individual could receive from its neighbouring grid elements. To describe how this was done, it is important to describe how the interaction between individuals and their neighbouring elements was implemented. A 3×3 matrix was produced filled with randomly generated number from a normal distribution with mean 1 and variance 3. The central value of the matrix was set to zero and the values normalised such that the sum of terms in the matrix was equal to one. This matrix was then scaled by a value called the susceptibility s,

$$s = \frac{V_{threshold}}{\frac{n \times V_{max}}{8}} \tag{3.3}$$

Where $V_{threshold}$ is the viral load needed for a viral outbreak to occur (found to be $V_{threshold} \approx 0.0023$), V_max is the maximal viral load (found to be approximately $V_{max} \approx 80 \times 10^{10}$) and n is the number of surrounding individuals at maximum viral loads needed to create a virus outbreak in one time step. The parameter n can be tuned in the GUI.

3.3 Implementation of Stochastic Differential Equations

Obviously, individuals respond differently to infections as their immune systems are different. Modelling this individuality in immune response goes beyond just applying some randomness to the initial conditions: It should be intrinsic to each individual, i.e. expressed in the ODE system 2.1 – 2.11. Potentially, it would be possible to add stochastic terms to either of these differential equation. However, the most representative variable regulating the immune response is the antigen distance S. It represents the immune memory of each individual. Some thoughts and implementations on Stochastic Differential Equations (SDEs) will be shared in this section.

3.3.1 Euler-Maruyama

The most simple algorithm for SDEs is Euler-Maruyama:

$$\mathbf{y}^{t+\Delta t} = \mathbf{y}^{t} + \Delta t\, \mathbf{f}\left(t_i,\, \mathbf{y}^{t}\right) + \sqrt{\Delta t}\mathbf{g}\left(t_i,\, \mathbf{y}^{t}\right) \cdot \boldsymbol{\xi} \tag{3.4}$$

where the initial conditions $\mathbf{y}^0$ are given.

This is an iterating algorithm which computes the values of a vector $\mathbf{y}$ at time $t + \Delta t$ from the values of the vector at time t. While the superscript denotes time the subscript denotes the vector component $(\mathbf{y})_i = y_i$. The same applies to the vector valued functions $\mathbf{f}$ and $\mathbf{g}$. Usually, $\mathbf{f}$ is called drift rate function and $\mathbf{g}$ is called the diffusion rate function. The time step is Δt.

The random variable ξ_i is is generated by a normal distribution. For all indices i^* in which we want to include a random variable we generate it using `randn(1)`. Therefore,

$$\xi_i = \delta_{i,\,i^*}\texttt{randn}(1) \tag{3.5}$$

and $g_i = y_i$.

In the case of our model,

$$y_i^t = \left(V^t,\, H^t,\, I^t,\, M^t,\, F^t,\, R^t,\, E^t,\, P^t,\, A^t,\, S^t\right)^T \tag{3.6}$$

where the values of our model variables are also denoted with a superscript t. We decided to include a stochastic term only at the antigen variable S (corresponding to $i^* = 10$) so the diffusion rate function reads

$$g_i^t = \left(0, 0, 0, 0, 0, 0, 0, 0, 0, c\, S^t\right)^T \tag{3.7}$$

We employ a scaling factor c for consistency.

Our implementation (`Coupled_Multiscale_EM`) of the forward Euler–Maruyama method showed very poor convergence for this system of 10 coupled stiff ODEs. After extensive debugging and experimentation with stepsizes etc. we decided we will not get anywhere without stabilisation: This calls for a predictor-corrector method.

3.3.2 Predictor-Corrector Euler–Maruyama

For a stabilisation of the simple forward Euler algorithm which Euler–Maruyama generalises for SDEs, we implemented a predictor-corrector version of Euler–Maruyama. This includes a simple predictor Euler step $\tilde{y}_i^{t+\Delta t}$ and a corrector step $y_i^{t+\Delta t}$ incorporating the predictor:

$$\begin{aligned}
&\text{predictor} && \tilde{\mathbf{y}}^{t+\Delta t} = \mathbf{y}^{t} + \Delta t\, \mathbf{f}\left(t,\, \mathbf{y}^{t}\right) \\
&\text{corrector} && \mathbf{y}^{t+\Delta t} = \mathbf{y}^{t} + \frac{\Delta t}{2}\left(\mathbf{f}\left(t, \mathbf{y}^{t}\right) + \mathbf{f}\left(t + \Delta t,\, \tilde{\mathbf{y}}^{t+\Delta t}\right)\right) + \sqrt{\Delta t}\mathbf{g}\left(t_i,\, \mathbf{y}^{t}\right) \cdot \boldsymbol{\xi}
\end{aligned}$$

Predictor-corrector algorithms *always* have a stabilising effect, suggesting themselves for stiff problems. This algorithm converges much better than the simple forward Euler–Maruyama and was eventually employed in our simulation (see fig. 3.2). It was only stable for the first 7 days, thus in the grid simulation the stochastic solver was only used for the first five timesteps. This still introduced a large degree of randomness into the solutions found.

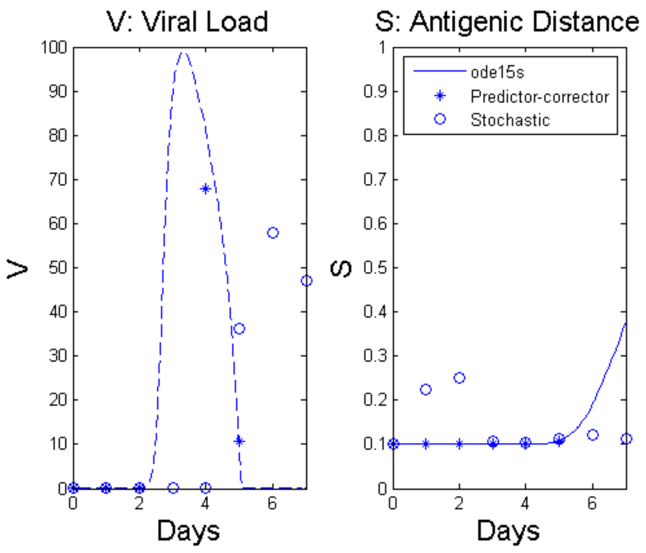


Figure 3.2: An implementation of Predictor-Corrector Euler–Maruyama compared to Matlab `ode15s` results. The stochasticity causes the viral load to soar after the `ode15s` solution.

3.3.3 Predictor-Corrector Fourth Order Runge-Kutta

As a further improvement, we tried a predictor-corrector version of fourth order Runge Kutta. This is implemented as such:

$$\begin{aligned}
\text{predictor} \qquad \tilde{\mathbf{y}}^{t+\Delta t} &= \mathbf{y}^t + \Delta t\, \mathbf{f}\left(t,\, \mathbf{y}^t\right) \\
\text{corrector} \qquad \tilde{\mathbf{k}}_1 &= \mathbf{f}\left(t + \Delta t,\, \tilde{\mathbf{y}}^{t+\Delta t}\right) \\
\tilde{\mathbf{k}}_2 &= \mathbf{f}\left(t + \frac{3}{2}\Delta t,\, \tilde{\mathbf{y}}^{t+\Delta t} + \frac{1}{2}\tilde{\mathbf{k}}_1\right) \\
\tilde{\mathbf{k}}_3 &= \mathbf{f}\left(t + \frac{3}{2}\Delta t,\, \tilde{\mathbf{y}}^{t+\Delta t} + \frac{1}{2}\tilde{\mathbf{k}}_2\right) \\
\tilde{\mathbf{k}}_4 &= \mathbf{f}\left(t + 2\Delta t,\, \tilde{\mathbf{y}}^{t+\Delta t} + \frac{1}{2}\tilde{\mathbf{k}}_3\right) \\
\mathbf{y}^{t+\Delta t} &= \mathbf{y}^t + \frac{\Delta t}{6}\left(\tilde{\mathbf{k}}_1 + 2\tilde{\mathbf{k}}_2 + 2\tilde{\mathbf{k}}_3 + \tilde{\mathbf{k}}_4\right)
\end{aligned}$$

The convergence of our predictor-corrector RK4 (see `SDERK4solver`) implementation is actually worse than for predictor-corrector Euler–Maruyama, crashing our population into an `Inf` and `NaN` apocalypse after roughly five days. For this reason we chose the predictor-corrector Euler–Maruyama for our simulation.

Chapter 4

Results

4.1 The Initial Infection

As discussed earlier, we inherited a simple model of an epidemic spreading through a population with a fixed infection kernel. This was done with an initial viral load of 0.1 and with 15 time steps for individual grid spaces. This results in a viral load and antigenic distance plot shown in Figure 4.1. Antigenic distance was found to be the best indicator of a symptomatic infection at any time point as it increases to 1 and stays at 1 even after the infection subsides and the viral load returns to 0. All other variables are not shown as these two are the most indicative of the progress of the infection.

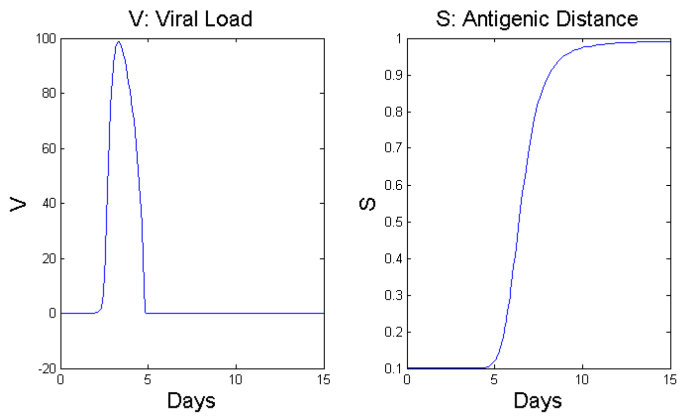


Figure 4.1: Timeline of an Infection

However, not all initial viral loads produce such a response. If the initial viral load is below 0.025, it does not increase at all and instead decreases to 0. Such a simulation with initial viral load at 0.001 is shown in Figure 4.2

4.2 Randomisation of Spread of Infection

We also inherited a basic population model for the spread of influenza from the previous group. This model assumes spread of influenza from any element to all 8 of its neighbours equally using a convolution matrix discussed above. The result of such a simulation of 20

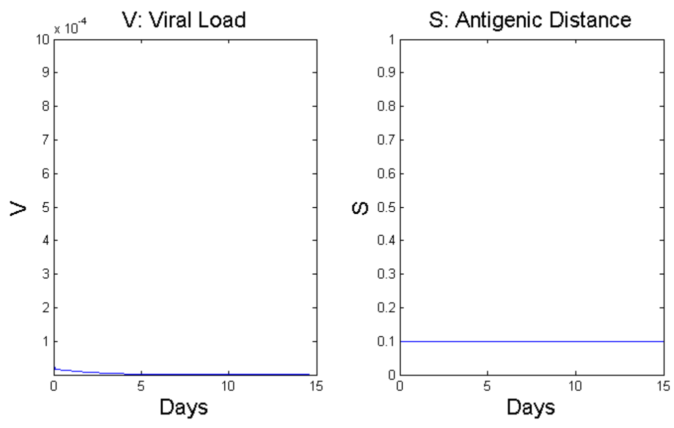


Figure 4.2: Timeline of an Infection with Insufficient Virus

time points assuming an initial viral load of 0.1 infecting a person in the middle of a 10 by 10 grid can be seen in Figure 4.3. The antigenic distance plot is seen as this covers all the individuals who have been infected within the time of running the simulation.

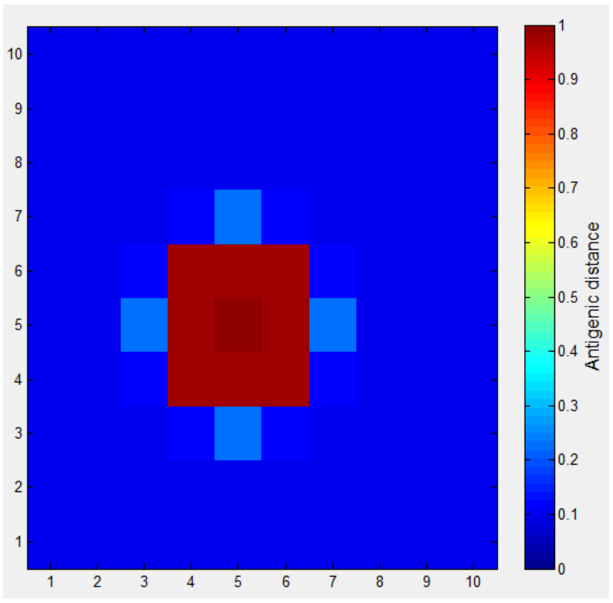


Figure 4.3: Simulation of Spread of Influenza with Nonrandom Infection

A plot with the same initialisation but a randomised infection kernel is seen in Figure 4.4. This is more biologically relavant as the spread of influenza depends on the intensity of contact between individuals and a randomisation of the infection kernel reflects daily life in that people interact with others to different extents even though they are in the same social circle.

4.3 Susceptability

Our simulation allows a user to define susceptability, which is a measure of the infectiousness of the virus. Susceptability is defined by the program as the number of neighbours at a viral load of 80 or above required to infect any given element. Hence, a virus with a susceptability of 1, which is the default taken by the program is actually less infectious

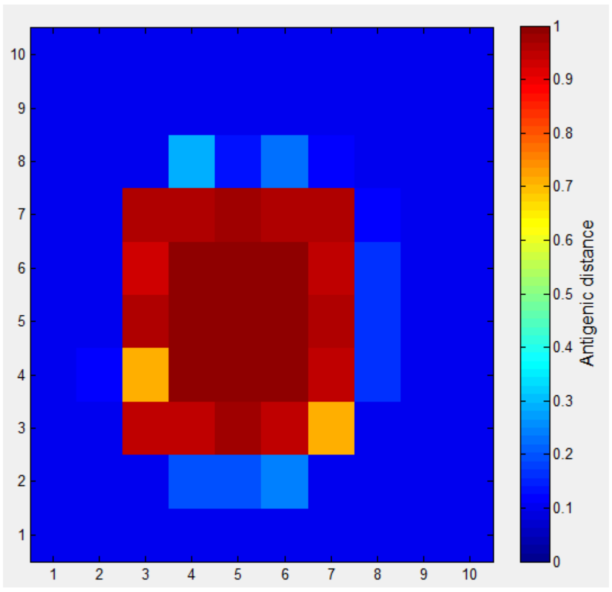


Figure 4.4: Simulation of Spread of Influenza with Random Infection

than that with a susceptability of 0.2. A simulation of 20 time points of a epidemic with the same initialisation as those above but with a susceptability of 0.2 is shown in Figure 4.5

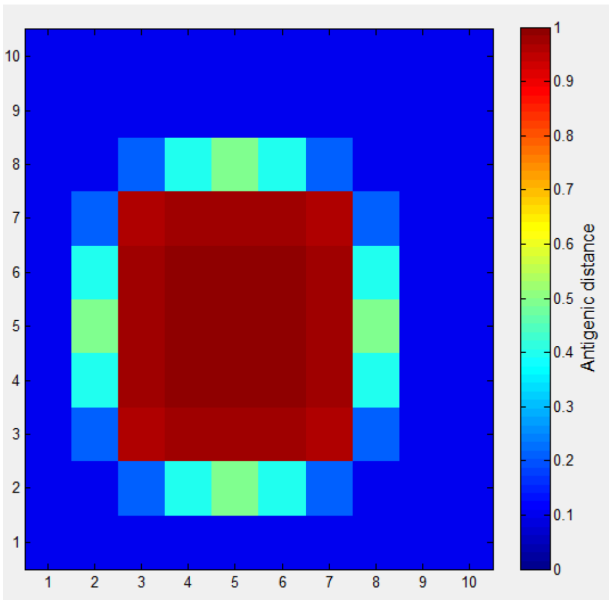


Figure 4.5: Simulation of Spread of Influenza with Nonrandom Infection and a Susceptability of 0.2

4.4 Interventions

As discussed in the previous sections, there are 3 main public health interventions against an influenza epidemic. The effects of these interventions have been simulated in isolation as well as in combination. At the individual level, vaccination has the effect of making a person immune to influenza with an initial viral load of 0.1. This is shown in Figure 4.6, where the initial antigenic distance parameter is set to 0.35 instead of 0.1, mimicking the effect of vaccination as discussed earlier.

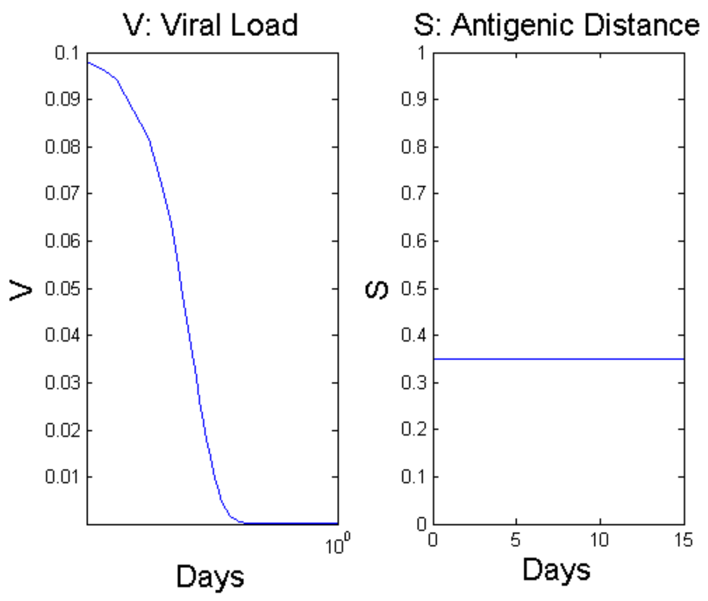


Figure 4.6: Timeline of an Infection on a Vaccinated Individual

Vaccination also has the effect of slowing the spread of the epidemic and containing it, as shown in the simulation in Figure 4.7. This simulation has the same initialisation as that described in Figure 4.3, except for a change in the initial vaccination parameter, where 20% of the individuals have been vaccinated against influenza and the simulation was run for 30 time cycles for more visible results.

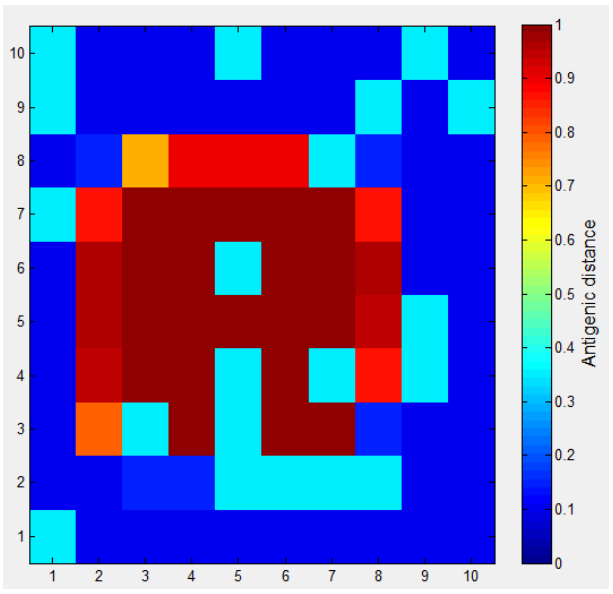


Figure 4.7: Simulation of Spread of Influenza with Nonrandom Infection and 20% of Population Vaccinated - The Cyan Pixels Represent Vaccinated Individuals

Another possible measure taken against the spread of epidemics is the quarantine of infected individuals. This has no effect on the disease kinetics but has a very marked effect on the spread of epidemics. The effect of quarantine is an effective prevention of spread of infection in certain directions where there exists a set of quarantined individuals, as seen in Figure 4.8. This simulation was run with similar initialisation parameters as Figure 4.3, but for 50 time cycles to make the effect more visible.

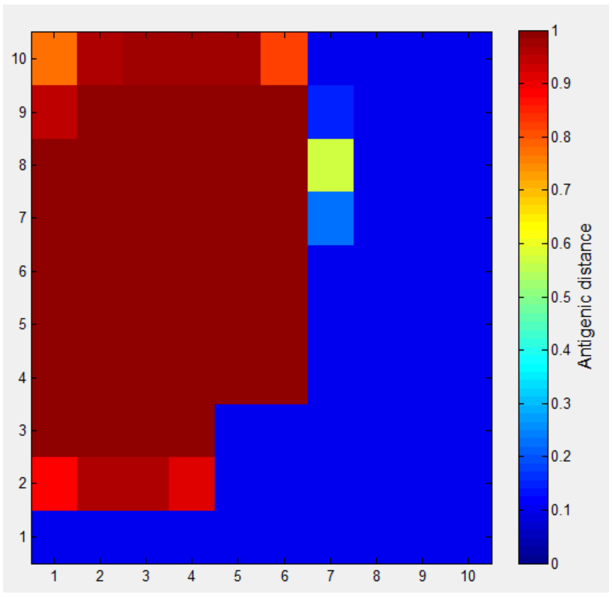


Figure 4.8: Simulation of Spread of Influenza with Nonrandom Infection and 55% of Population Quarantined after Infection

Treatment after infection is another common prevention measure for the spread of epidemics. The effect of treatment on disease kinetics is to reduce the viral load at all time points after initialisation. While the virus is still able to reproduce to some extent, it is greatly reduced compared to if no treatment was used. This however, also has the effect of neutering the increase in antigenic distance and leaves individuals susceptable to future infections if more virus was introduced midway through the simulation. While the introduction of virus within the simulations was not performed, the effect of treatment on individuals can been seen in Figure 4.9.

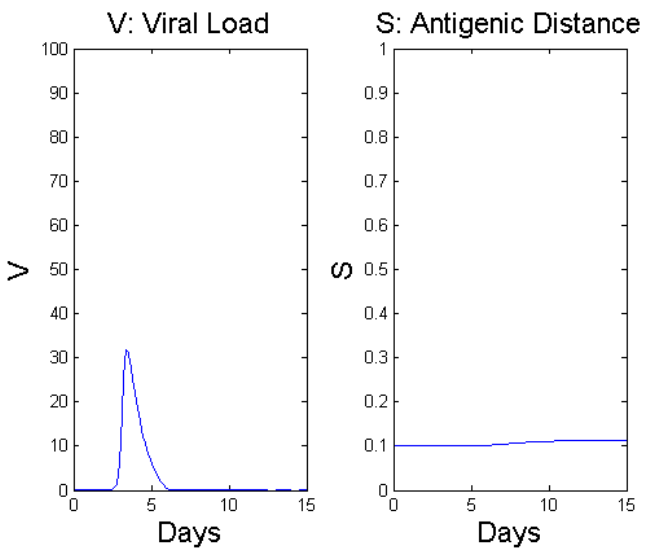


Figure 4.9: Timeline of an Infection with Treatment after Onset

Treatment has the effect of slowing down the spread of an epidemic by making treated individuals less or non-infectious due to reducing their viral load. This effect can be seen in this simulation shown in Figure 4.10. This simulation was carried out with the same initialisation parameters as Figure 4.3, but with 50% of all affected individuals treated and for 50 time cycles in order to make the effect more visible.

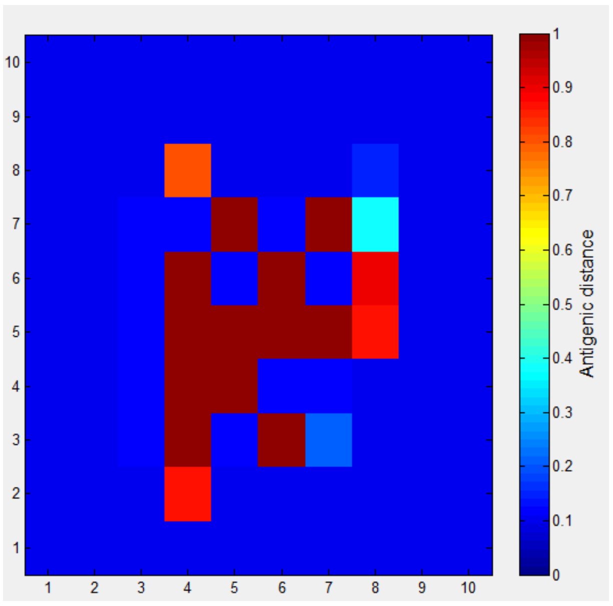


Figure 4.10: Simulation of Spread of Influenza with Nonrandom Infection and 50% of Population Treated after Infection- Blue Pixels in Centre Areas Represent Treated Individuals

Public health initiatives often advocate the use of such interventions in combination and the effects of these are often cumulative. In order to show the effect of these interventions in tandem, we have performed a simulation with a 100 by 100 grid of individuals with an initial infection of 10 selected individuals in all parts of the grid, roughly evenly spaced out from one another. This simulation also assumes random infection, a susceptability of 1 and is run for 100 time cycles. In this simulation, 10% of all individuals are vaccinated prior to starting the simulation, 55% of individuals go into quarantine after infection and 50% of individuals are treated after infection. This is roughly equal to the percentages that exist in society as quoted by epidemic simulations [1]. The results of this simulation are shown in Figure 4.11

As can be seen from this simulation, only 1 of the individuals initially infected ended up spreading the disease. However, if given enough time, even this 1 individual would be enough for influenza to become a pandemic within the grid. Hence, even these interventions are not enough to completely prevent the spread of an epidemic.

4.5 Stochasticity

All the simulations above were performed with a set of deterministic differential equations. A separate simulation was run with a stochastic differential equation model instead and the results are shown below in Figure 4.12. This simulation was run with the same initialisation as that of Figure 4.4 except for the use of the SDE model. The antigenic distance still increases after infection, but not to 1 and this is due to the stochasticity in the model and is perhaps more biologically accurate as antibodies do not match exactly

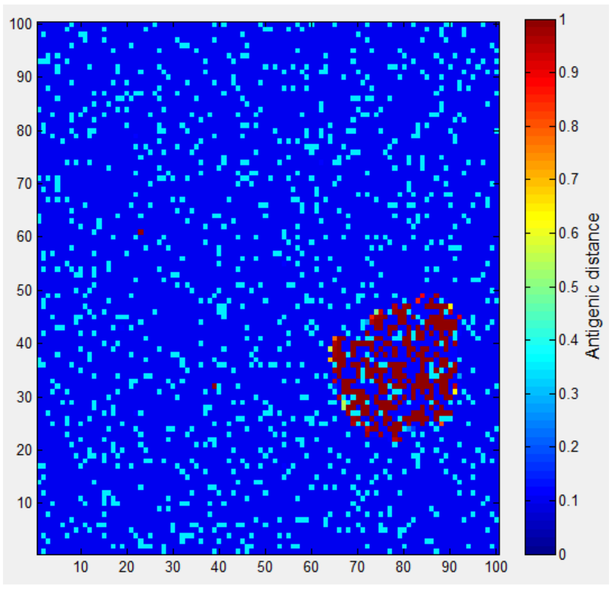


Figure 4.11: Big Grid Simulation of Spread of Influenza with Random Infection

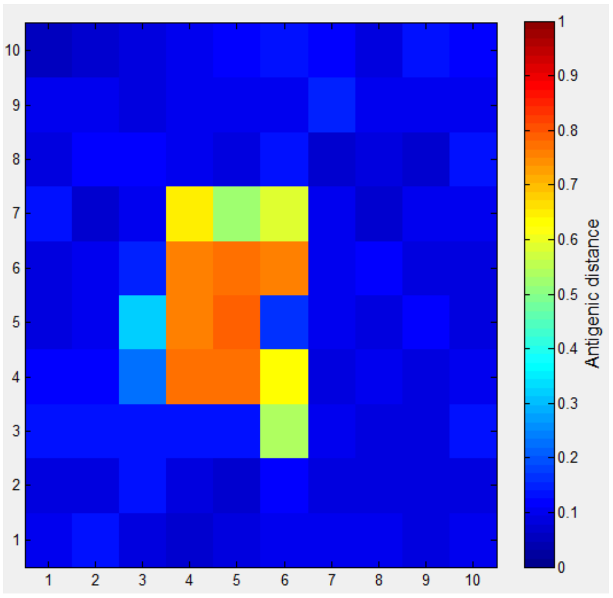


Figure 4.12: Stochastic Simulation of Spread of Influenza with Random Infection

to antigens, only enough to cause immunity in biological systems. In all other aspects, it is similar to the deterministic model in the spread of the epidemic.

Chapter 5

Discussion

We inherited a well-defined and easy to understand project, and consequently were able to implement various improvements that increased the project's applicability to infection in the real-world.

As outlined in the results, the implementation of vaccination, quarantine, treatment and stochastic processes allow the model to considered more realistic. The implementation of a stochastic nature to both the convolution that distributed viral loads, and the underlying ODE system by a conversion to an Euler-Maruyama system, greatly increased the models applicability to the real world. Although it is a moot point as whether nature has inherent randomness, or whether stochastics is more a useful framework in which to frame things not well understood, it does not limit the use of randomness to mimic biological systems unpredictability. By methods of introducing randomness into the system, both in the fraction of viral load distributed, and the first 5 days of viral load > 0 for each cell added realism and moved away from the predictable nature of the inherited model. The different vaccination, quarantine and treatment schemes also allowed more to be gauged from the model. Most importantly, it removed the symmetry from the viral load wave front as present in the inherited model. This is especially evident in figures 2.10 and 2.12, where the spread of influenza is irregular representing the inhomogeneity of a population. Finally, a key success of the project was optimising the code through parallelisation and the use of a stiff ODE solver. In similar simulations this improved the models running time by over two-fold. The inherited project had been slow to solve the underlying system, and so the increase in speed achieved greatly increased the projects usefulness and flexibility of use.

Despite the improvements made to the inherited project, we were well aware that serious flaws still existed with the project and therefore its applicability.

The use of convolution to spread viral load was an intelligent and low effort way to represent how infection could spread through a community. However, it meant that interaction between moving individuals was only represented abstractly, if at all. An obvious improvement, while still maintaining the cellular automata format, would be to have a cell's viral load distributed to eight randomly selected cells from the entire grid, as opposed to the eight immediately adjacent ones. This could mimic interaction between people that moved, and so improve the models applicability.

Another obvious difficulty with the model is that each cell has the same ODE system representing its immune system. Human beings are not identical, and so a plausible way to introduce this to the model, beyond personalising the ODE system for each cell, would be to assign different starting parameters to the cells. While over a long simulation the cells starting differences may disappear, in a shorter simulation these variations in parameters could act to mimic individual variability.

Bibliography

[1] D. L. Chao, M. E. Halloran, V. J. Obenchain, and I. M. Longini. FluTE, a publicly available stochastic influenza epidemic simulation model. *PLoS Comput. Biol.*, 6(1):e1000656, Jan 2010. 1.2, 4.4

[2] B. Hancioglu, D. Swigon, and G. Clermont. A dynamical model of human immune response to influenza A virus infection. *J. Theor. Biol.*, 246(1):70–86, May 2007. 1, 2.1, 2.3

[3] M. N. Matrosovich, T. Y. Matrosovich, T. Gray, N. A. Roberts, and H. D. Klenk. Human and avian influenza viruses target different cell types in cultures of human airway epithelium. *Proc. Natl. Acad. Sci. U.S.A.*, 101(13):4620–4624, Mar 2004. 1.1

[4] A. S. Monto, S. Gravenstein, M. Elliott, M. Colopy, and J. Schweinle. Clinical signs and symptoms predicting influenza infection. *Arch. Intern. Med.*, 160(21):3243–3247, Nov 2000. 1

[5] I. Yang, T. J. Kremen, A. J. Giovannone, E. Paik, S. K. Odesa, R. M. Prins, and L. M. Liau. Modulation of major histocompatibility complex Class I molecules and major histocompatibility complex-bound immunogenic peptides induced by interferon-alpha and interferon-gamma treatment of human glioblastoma multiforme. *J. Neurosurg.*, 100(2):310–319, Feb 2004. 1.1