12-2020

On Coupled Reaction Diffusion Equations and Their Applications

Juan J. Huerta
The University of Texas Rio Grande Valley

ON COUPLED REACTION DIFFUSION EQUATIONS AND THEIR APPLICATIONS

A Thesis

by

JUAN J. HUERTA

Submitted to the Graduate College of
The University of Texas Rio Grande Valley
In partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2020

Major Subject: Mathematics

ON COUPLED REACTION DIFFUSION EQUATIONS AND THEIR APPLICATIONS

A Thesis
by
JUAN J. HUERTA

COMMITTEE MEMBERS

Dr. Erwin Suazo
Chair of Committee

Dr. Tamer Oraby
Co-Chair of Committee

Dr. Zhijun Qiao
Committee Member

Dr. Josef Sifuentes
Committee Member

December 2020

ABSTRACT

Huerta, Juan J., On coupled reaction diffusion equations and their applications. Master of Science (MS), December, 2020, 65 pp., 37 figures, 8 references.

Reaction-diffusion equations are nonlinear partial differential equations that have been used extensively in mathematical modeling. An interesting case in this type of equation is the Fisher-Kolmogorov system, which has been used to study a low-grade glioma, a group of primary brain tumors. In the first part of this thesis, a stochastic version of the Fisher-Kolmogorov system will be studied, and exact and numerical solutions will be presented.

The second part of this thesis will show how the speed of information propagation affects disease spread and vaccination uptake through networks in epidemics. In this model, the information reaches different people at different distances from the center of information containing the health data. The Fisher-Kolmogorov equations are used to depict the vaccine and disease information propagation on a network embedded into a straight line. The Fisher-Kolmogorov equations are coupled equations with the SIR (Susceptible-Infected-Recovered) model to examine the anticipated mutual influence. Numerical simulations of the model are presented. It is shown how the propagation of information about the disease impacts the probability of vaccination and, consequently, the vaccination rate.

DEDICATION

To my family and friends that supported me throughout this journey.

ACKNOWLEDGMENTS

I would like to acknowledge Dr. Erwin Suazo and Dr. Tamer Oraby; they both helped me so much through this thesis. It was fun working with them.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

CHAPTER I

INTRODUCTION

In this chapter, we are going to explain the Lotka-Volterra equations, also known as the predator-prey equations. They are pair of nonlinear differential equations, mainly used to describe the dynamics of biological systems between two species. The Lotka-Volterra system is a Kolmogorov model used for competition, diseases, mutualism, etc. The competition exclusion, also known as Gause's law, is a proposition that when two species compete for the same limited resources both cannot coexist. One of the competitors will always have an ever so slight advantage over the other which in the long run will lead to extinction. If the weaker competitor does not go extinct, it will lead to an evolutionary shift to a different ecological niche. As a consequence, competing species often evolve to coexist. This also helps the population of each species to maintain the slight advantage over the competing species since the population changes over time.

1.1 Lotka-Volterra equations

In a scenario of J.D. Murray [6], we study two or more species which compete for the same limited food or in some way prevent each other's growth. The species compete with each other for territory, which relates to resources such as food, water, etc. If the species both fight for the same resource, as a result one of the species will become extinct as two or more complete competitors cannot coexist. If the weaker competitor does not become extinct, it shifts and adapts to other resources to avoid competition or the two species adapt to share the resources with one another in order to coexist.

Let us consider the Lotka-Volterra competition model with two species N_1 and N_2,

$$\frac{dN_1}{dt} = r_1 N_1[1 - \frac{N_1}{K_1} - b_{12}\frac{N_2}{K_1}], \tag{1.1}$$

$$\frac{dN_2}{dt} = r_2 N_2[1 - \frac{N_2}{K_2} - b_{21}\frac{N_1}{K_2}], \tag{1.2}$$

where r is the linear birth rate and K is the carrying capacity, where it is the maximum population size of the species that the environment can sustain given the resources available. b_{12} is the competitive effect of N_2 on N_1, and b_{21} is the competitive effect of N_1 on N_2. They are all positive constants.

We can simplify this model if we nondimensionalize it by

$$\begin{aligned} u_1 &= \frac{N_1}{K_1}, & u_2 &= \frac{N_2}{K_2}, \\ \tau &= r_1 t, & \rho &= \frac{r_2}{r_1}, \\ a_{12} &= b_{12}\frac{K_2}{K_1}, & a_{21} &= b_{21}\frac{K_1}{K_2}. \end{aligned} \tag{1.3}$$

Thus, equation (1.1) and (1.2) change to

$$\frac{du_1}{d\tau} = u_1[1 - u_1 - a_{12}u_2] = f_1(u_1, u_2), \tag{1.4}$$

$$\frac{du_2}{d\tau} = \rho u_2[1 - u_2 - a_{21}u_1] = f_2(u_1, u_2). \tag{1.5}$$

The steady states and phase plane singularities u_1^*, u_2^* are solutions to $f_1(u_1, u_2) = f_2(u_1, u_2) = 0$. Hence, the solutions we get are

1. $(u_1^*, u_2^*) = (0, 0)$
2. $(u_1^*, u_2^*) = (1, 0)$
3. $(u_1^*, u_2^*) = (0, 1)$
4. $(u_1^*, u_2^*) = (\frac{1-a_{12}}{1-a_{12}a_{21}}, \frac{1-a_{21}}{1-a_{12}a_{21}})$.

The last solution is only of relevance if $u_1^*, u_2^* \geq 0$ are finite in which $a_{12}a_{21} \neq 1$. The null clines of (1.4) and (1.5) are the straight lines

$$f_1 = 1 - u_1 - a_{12}u_2 = 0, \quad f_2 = 1 - u2 - a_{21}u_1 = 0,$$

which are going to be graphs in u_1, u_2 phase plane.

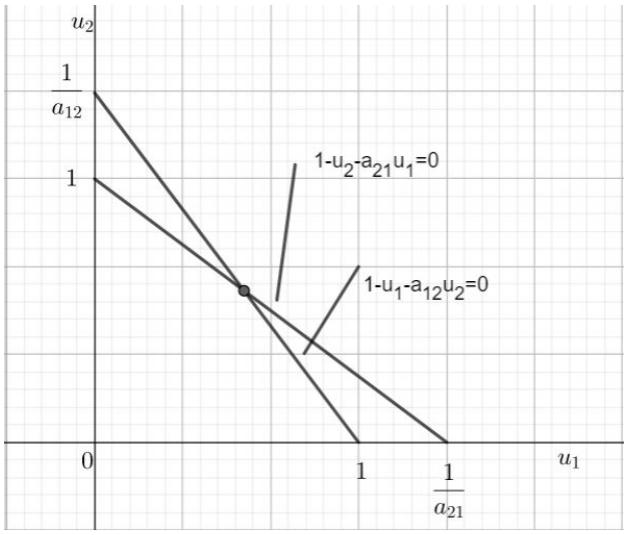

Figure 1.1: Null clines (a) of the competition model

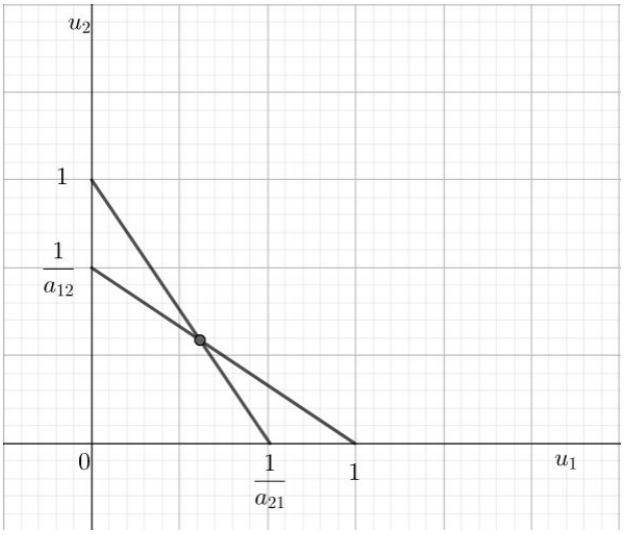

Figure 1.2: Null clines (b) of the competition model

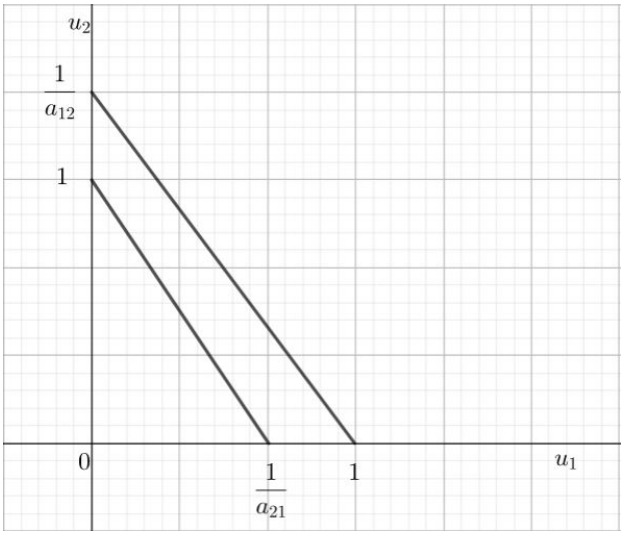

Figure 1.3: Null clines (c) of the competition model

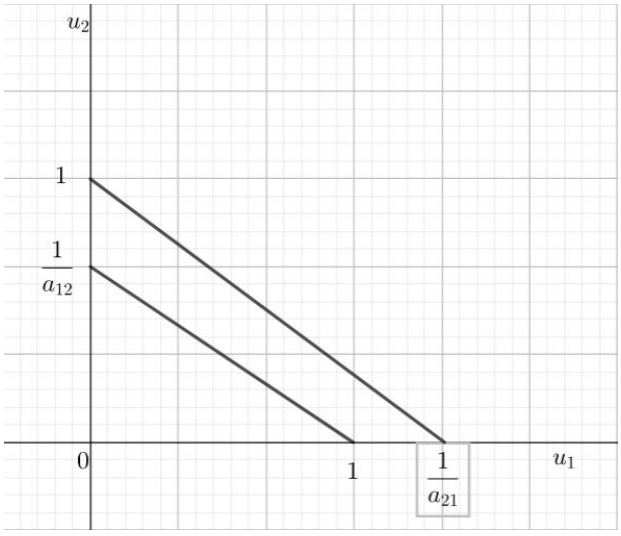

Figure 1.4: Null clines (d) of the competition model

To see the stability of the solutions of (1.4) and (1.5) we will use the community matrix (Jacobian). Therefore, we have

$$A = \begin{pmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} \end{pmatrix} \tag{1.6}$$

$$= \begin{pmatrix} 1-2u_1-a_{12}u_2 & -a_{12}u_2 \\ -\rho a_{21}u_2 & \rho(1-2u_2-a_{21}u_1) \end{pmatrix}_{u_1^*,u_2^*}.$$

The first steady state is $(0,0)$, and by finding the eigenvalues, λ, of the community matrix (1.6), we get

$$|A-\lambda I| = \begin{vmatrix} 1-\lambda & 0 \\ 0 & \rho-\lambda \end{vmatrix} = 0$$

$$\implies \lambda_1 = 1,\ \lambda_2 = \rho.$$

Now, since the eigenvalues are positive the first steady stable is unstable. For the second steady state, namely, $(1,0)$, we obtain

$$|A-\lambda I| = \begin{vmatrix} -1-\lambda & -a_{12} \\ 0 & \rho(1-a_{21})-\lambda \end{vmatrix} = 0$$

$$\implies \lambda_1 = -1,\ \lambda_2 = \rho(1-a_{21}).$$

Thus,

$$u_1^* = 1, u_2^* = 0 \text{ is } \begin{cases} \text{stable} & \text{if } \begin{cases} a_{21} < 1 \\ a_{21} > 1. \end{cases} \\ \text{unstable} \end{cases}$$

For the third steady state, $(0,1)$, the eigenvalues are

$$|A-\lambda I| = \begin{vmatrix} -1-a_{12}-\lambda & 0 \\ -\rho a_{21} & \rho(-1)-\lambda \end{vmatrix} = 0$$

$$\implies \lambda_1 = -\rho, \;\; \lambda_2 = \rho(1-a_{12}).$$

$$u_1^* = 0, u_2^* = 1 \;\text{ is } \begin{cases} \text{stable} & \text{if } \begin{cases} a_{12} > 1 \\ \\ a_{12} < 1. \end{cases} \\ \\ \text{unstable} \end{cases}$$

If $a_{12} < 1$, we get a positive eigenvalue, making the steady state unstable. For the last steady state, the eigenvalues are

$$|A-\lambda I| = \begin{vmatrix} \frac{a_{12}-1}{1-a_{12}a_{21}} - \lambda & \frac{a_{12}(a_{12}-1)}{1-a_{12}a_{21}} \\ \\ \frac{\rho a_{21}(a_{21}-1)}{1-a_{12}a_{21}} & \frac{\rho(a_{21}-1)}{1-a_{12}a_{21}} - \lambda \end{vmatrix} = 0$$

$$\implies \lambda_1, \lambda_2 = [2(1-a_{12}a_{21})]^{-1}[(a_{12}-1)+\rho(a_{21}-1)$$
$$\pm \{[(a_{12}-1)+\rho(a_{21}-1)]^2 - 4\rho(1-a_{12}a_{21})(a_{12}-1)(a_{21}-1)\}^{\frac{1}{2}}].$$

The stability for this steady state depends on ρ, a_{12}, a_{21} for the Re λ if complex. For the last steady state we have various cases, which are

1. $a_{12} < 1$ and $a_{21} < 1$
2. $a_{12} > 1$ and $a_{21} > 1$
3. $a_{12} < 1$ and $a_{21} > 1$
4. $a_{12} > 1$ and $a_{21} < 1$.

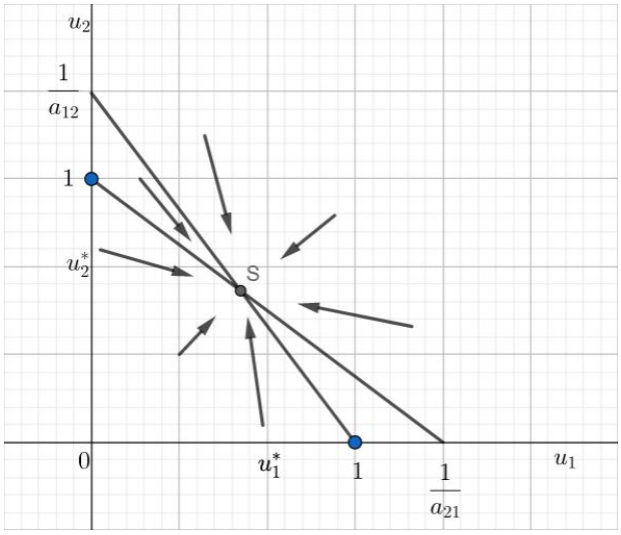

Figure 1.5: Schematic phase trajectories near the steady states (a)

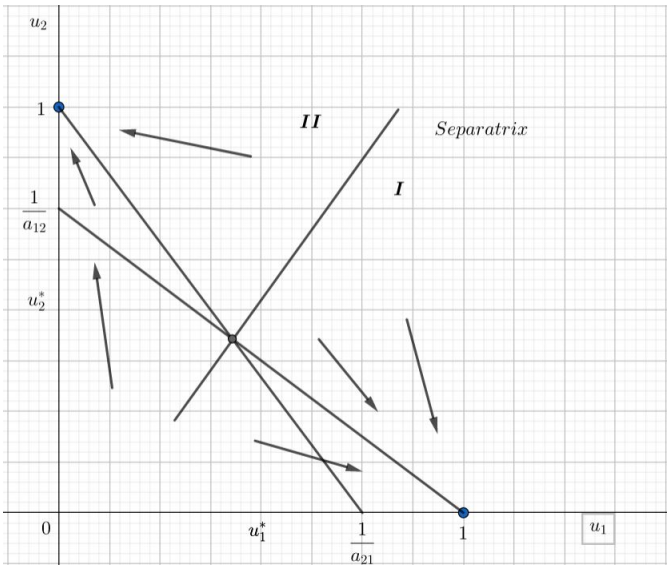

Figure 1.6: Schematic phase trajectories near the steady states (b)

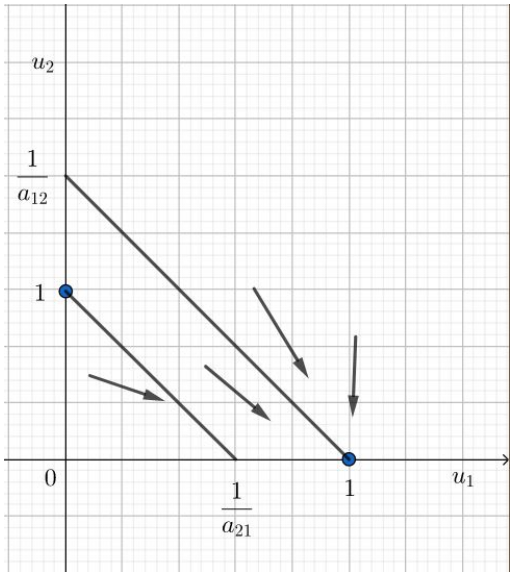

Figure 1.7: Schematic phase trajectories near the steady states (c)

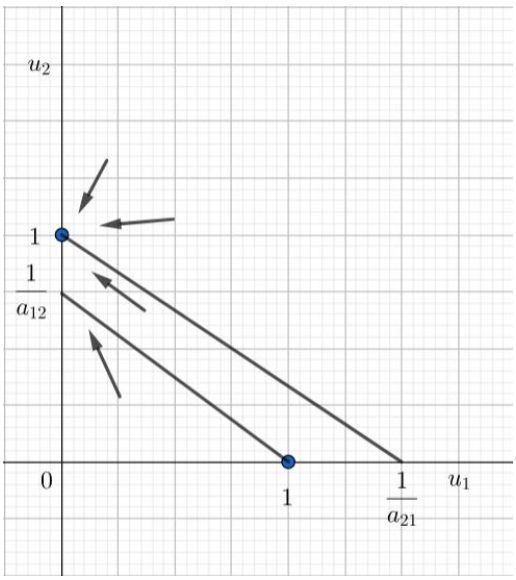

Figure 1.8: Schematic phase trajectories near the steady states (d)

The null clines figures: (1.1) to (1.4) and the steady states figures: (1.5) to (1.8) are related to the steady state cases (1) to (4). Now, (1.5) is the only positive steady state that is stable and all trajectories tend to it. Both species go to the steady state so they can coexist. For (1.6), (1,0) and (0,1) are the stable steady states, and each has a domain of attraction that is separated by a separatrix. The separatrix passes through the steady state which is a saddle point. Now, (1.7) there is only one steady state which is (1,0) where u_1 dominates causing u_2 to go extinct. (1.8) is similar to (1.7), yet the steady state is (0,1) and u_2 dominates and u_1 goes extinct. In cases (3) and (4) one of the species will go extinct due to natural fluctuations in the populations levels; this is the principle of competitive exclusion.

CHAPTER II

STOCHASTIC PARTIAL DIFFERENTIAL EQUATIONS

In this chapter, we study a general Fisher-Kolmogorov system with stochastic noise in time. Many mathematical models involve Fisher-Kolmogorov equations such as the following [1],

$$\frac{\partial C}{\partial t} = D\Delta C + \rho(1 - C - C_d)C \tag{2.1}$$

$$\frac{\partial C_d}{\partial t} = D\Delta C_d - \frac{\rho}{k}(1 - C_d - C)C_d. \tag{2.2}$$

The Fisher-Kolmogorov equations (2.1) and (2.2) are used to study a low-grade glioma which are a group of primary brain tumors. These tumors grow slowly with a median of survival time of more than 5 years. They are more common in children and adults under the age of 40. Radiation therapy is known to have a relevant effect on the survival, but in some cases it will be deferred to avoid side effects while maintaining its beneficial effect. A system can be modeled for the density of the tumor cells, C, and for the density of the damaged tumor cells, C_d, [1], [2], and [8].

In this section, we study solutions for the following stochastic Fisher-Kolmogorov System with random noise:

$$dm = (D(t)m_{xx} + \gamma(t)m_x + \beta(t)(1 - m - n)m - \mu(t)m)dt + \sigma(t)m_x dW_t \tag{2.3}$$

$$dn = (D(t)n_{xx} + \gamma(t)n_x + \beta(t)(1 - m - n)n - \mu(t)n)dt + \sigma(t)n_x dW_t. \tag{2.4}$$

2.1 Solutions for a System of Stochastic Fisher-Kolmogorov

2.1.1 Preliminaries and Results

m

Consider the probability space $(\Omega, \mathscr{F}, \mathbf{P})$ for which the Brownian motion $\{W_t, t \geq 0\}$ is defined and $E(W_s W_t) = \min(s,t)$ for all $s,t \geq 0$. Also consider the filtration $\mathscr{F}_t := \sigma(W_s : s \leq t)$ being the smallest $\sigma-$algebra to which W_s is measurable for $s \leq t$.

Then consider the stochastic differential equation (SDE) with variable coefficients [3]

$$dX_t = \alpha(t, X_t)dt + \beta(t, X_t)dW_t, \tag{2.5}$$

with initial state X_{t_0} and for $t \in [t_0, T]$. The SDE in (2.5) has a general solution given by

$$X_t = X_{t_0} + \int_{t_0}^{t} \alpha(s, X_s)ds + \int_{t_0}^{t} \beta(s, X_s)dW_s$$

for $t \leq T$. If $\alpha(t) := \alpha(t, X_t)$ and $\beta(t) := \beta(t, X_t)$, then equation (2.5) has a general solution given by

$$X_t = X_{t_0} + \int_{t_0}^{t} \alpha(s)ds + \int_{t_0}^{t} \beta(s)dW_s$$

for $t \leq T$. The process $\{W_t; t \geq 0\}$ is a Wiener process with respect to a filtration $\{\mathscr{F}_t; t \geq 0\}$. The initial state X_{t_0} is $\mathscr{F}_{t_0}$ and the functions $\alpha(t)$ and $\beta(t)$ are Lebesgue measurable and bounded on $[t_0, T]$. The latter implies both the global Lipschitz and linearity growth conditions required to ensure the existence and (path-wise) uniqueness of a strong solution to (2.5), [3].

Let X_t and Y_t be any two diffusion processes like those defined by the solution of equation (2.5). If $F(x,y)$ is a differentiable function that works as a transformation for two processes X_t and Y_t, then the general bi-variate Itô formula [4] gives

$$\begin{aligned} dF(X_t, Y_t) &= \partial_x F(X_t, Y_t)dX_t + \partial_y F(X_t, Y_t)dY_t + \frac{1}{2}\partial_{xx} F(X_t, Y_t)(dX_t)^2 \\ &\quad + \frac{1}{2}\partial_{yy} F(X_t, Y_t)(dY_t)^2 + \partial_{xy} F(X_t, Y_t)dX_t dY_t. \end{aligned} \tag{2.6}$$

$F(t,y)$ is a differentiable function [4].

We will use the following particular version of Ito's formula

$$dF_1 = f_1(t,Y_t)dt + g_1(t,Y_t)dW_t \tag{2.7}$$

$$dF_2 = f_2(t,Y_t)dt + g_2(t,Y_t)dW_t \tag{2.8}$$

related to the stochastic differential equation

$$dX_t = \alpha(t,X_t)dt + \beta(t,X_t)dW_t \tag{2.9}$$

where $\alpha : [0,T] \times \mathbb{R}^2 \to \mathbb{R}^2$, $b : [0,T] \times \mathbb{R}^2 \to \mathbb{R}^2$, $X_t = (X_t^1, X_t^2)$. Then

$$f_1(t,x) = \partial_t F_1 + \alpha_1(t,x)\partial_x F_1 + \frac{1}{2}\beta_1^2(t,x)\partial_{xx}F_1 \tag{2.10}$$

$$f_2(t,x) = \partial_t F_2 + \alpha_1(t,x)\partial_x F_2 + \frac{1}{2}\beta_2^2(t,x)\partial_{xx}F_2 \tag{2.11}$$

and

$$g_1(t,x) = \beta_1(t,x)\partial_x U(t,x) \tag{2.12}$$

$$g_2(t,x) = \beta_2(t,x)\partial_x U(t,x). \tag{2.13}$$

We consider a 2-dimensional Wiener process $W = \{W_t, t \geq 0\}$ with components W_t^1 and W_t^2 which are independent scalar Wiener processes with respect to a common family of $\sigma-$algebras $\{A_t, t \geq 0\}$.

Theorem 1. *Let D, γ, β, μ and $\sigma \in C^b([t_0,T])$ be bounded continuous functions on $[t_0,T]$. Then the stochastic system (2.3)-(2.4) has a solution $(m(t,z),n(t,z)) = (M(t,X_t),N(t,X_t))$ where*

$(M(t,x),N(t,x))$ is the solution of

$$\partial_t M = \left(D(t)-\frac{1}{2}\sigma^2(t)\right)\partial_{xx}M+\beta(t)(1-M-N)M-\mu(t)M \tag{2.14}$$

$$\partial_t N = \left(D(t)-\frac{1}{2}\sigma^2(t)\right)\partial_{xx}N+\beta(t)(1-M-N)N-\mu(t)N \tag{2.15}$$

with $M(x,0)=\phi_1(x)$ and $N(x,0)=\phi_2(x)$. Also X_t is the solution of

$$dX_t = \begin{pmatrix} \gamma(t) \\ \gamma(t) \end{pmatrix} dt + \begin{pmatrix} \sigma(t) & 0 \\ 0 & \sigma(t) \end{pmatrix} \begin{pmatrix} dW_t^1 \\ dW_t^2 \end{pmatrix}$$

with $X_t=(X_t^1,X_t^2)$.

Proof. Applying the Ito formula for the two dimensional case with

$$\begin{aligned} f_1(t,x) &= \partial_t M+\gamma(t)\partial_x M+\frac{1}{2}\sigma^2(t)\partial_{xx}M \\ f_2(t,x) &= \partial_t N+\gamma(t)\partial_x N+\frac{1}{2}\sigma^2(t)\partial_{xx}N \end{aligned}$$

using the expressions (2.14)-(2.15) we get

$$\begin{aligned} f_1(t,x) &= D(t)\partial_{xx}M+\beta(t)(1-M-N)M-\mu(t)M+\gamma(t)\partial_x M \\ f_2(t,x) &= D(t)\partial_{xx}N+\beta(t)(1-M-N)N-\mu(t)N+\gamma(t)\partial_x N. \end{aligned}$$

We also obtain

$$g_1(t,x) = \sigma(t)\partial_x M(t,x) \tag{2.16}$$

$$g_2(t,x) = \sigma(t)\partial_x N(t,x), \tag{2.17}$$

which proves the statement. □

2.2 Traveling waves for a particular case of the Fisher-Kolmogorov

In this section, we look for the exact solutions of the coupled Fisher-Kolmogorov equations in the model (3.3) without the boundary conditions. A solution is obtained by simplifying the model, which is done by nondimensionalizing it and using special coefficients.

Proposition 1. *The following coupled Fisher-Kolmogorov system,*

$$\frac{\partial M_I}{\partial t} = D_I \frac{\partial^2 M_I}{\partial x^2} + \gamma_I (1 - M_I - \beta_{VI} M_V) M_I + C_I \frac{\partial M_I}{\partial x} \tag{2.18}$$

$$\frac{\partial M_V}{\partial t} = D_V \frac{\partial^2 M_V}{\partial x^2} + \gamma_V (1 - M_V - \beta_{IV} M_I) M_V + C_V \frac{\partial M_V}{\partial x}, \tag{2.19}$$

admits explicit traveling wave solutions under the following conditions:

i) $M_I = 1, M_V = 0$ *at* $z = -\infty$ *and* $M_I = 0, M_V = 1$ *at* $z = \infty$

ii) $\beta_{VI} + \beta_{IV} = 2$.

Proof. We are going to simplify the equations by letting $D = \frac{D_V}{D_I}, \gamma = \frac{\gamma_V}{\gamma_I}$

$$\frac{\partial M_I}{\partial t} = \frac{\partial^2 M_I}{\partial x^2} + (1 - M_I - \beta_{VI} M_V) M_I + C_I \frac{\partial M_I}{\partial x} \tag{2.20}$$

$$\frac{\partial M_V}{\partial t} = D \frac{\partial^2 M_V}{\partial x^2} + \gamma (1 - M_V - \beta_{IV} M_I) M_V + C_V \frac{\partial M_V}{\partial x}. \tag{2.21}$$

Now, we are looking for a solution for $M_I(x,t) = f(z)$ and $M_V = g(z)$, where $z = x - wt$. We have the boundary conditions $M_I = 1, M_v = 0$ at $z = -\infty$ and $M_I = 0, M_v = 1$ at $z = \infty$. By substituting $M_I(x,t) = f(x - w_1 t)$ and $M_V = g(x - w_2 t)$ the pdes change to

$$-w_1 \frac{\partial f}{\partial z} = \frac{\partial^2 f}{\partial z^2} + (1 - f - g\beta_{VI}) f + C_I \frac{\partial f}{\partial z} \tag{2.22}$$

$$-w_2 \frac{\partial g}{\partial z} = D \frac{\partial^2 g}{\partial z^2} + \gamma (1 - g - \beta_{IV} f) g + C_V \frac{\partial g}{\partial z}. \tag{2.23}$$

By a special case we let $D = \gamma = 1$ and $\beta_{VI} + \beta_{IV} = 2$. If we let the equations (2.22) and (2.23) equal to zero by moving everything to one side and then adding the equations together where $h = f + g$,

we then acquire the following equation

$$\frac{\partial^2 h}{\partial z^2} + w\frac{\partial h}{\partial z} + h(1-h) + C\frac{\partial h}{\partial z} = 0,$$

and by combining like terms, we get

$$\frac{\partial^2 h}{\partial z^2} + (w+C)\frac{\partial h}{\partial z} + h(1-h) = 0.$$

The boundary conditions for this Fisher-Kolmogoroff are different with $h = 1$ at $z = \pm\infty$, and which suggest that for all z, $h = 1 \implies f + g = 1$. Now, if we substitute $f + g = 1$ into the equation (2.22) we get

$$-w_1\frac{\partial f}{\partial z} = \frac{\partial^2 f}{\partial z^2} + (1 - f - \beta_{VI}g)f + C_I\frac{\partial f}{\partial z},$$

and by combining like terms and factoring, we obtain

$$\frac{\partial^2 f}{\partial z^2} + (w_1 + C_I)\frac{\partial f}{\partial z} + (1-\beta_{VI})f(1-f) = 0 \tag{2.24}$$

with the wavefront speed

$$w_1 \geq w_{min} = 2(1-\beta_{VI})^{1/2}, \beta_{VI} < 1.$$

Similarly, for equation (2.23), we have

$$-w_2\frac{\partial g}{\partial z} = D\frac{\partial^2 g}{\partial z^2} + \gamma(1 - g - \beta_{IV}f)g + C_V\frac{\partial g}{\partial z}.$$

Then by combining like terms and factoring, the following equation is obtained:

$$\frac{\partial^2 g}{\partial z^2} + (w_2 + C_V)\frac{\partial g}{\partial z} + (\beta_{IV} - 1)g(1-g) = 0 \tag{2.25}$$

with the wavefront speed

$$w_{min} = 2(\beta_{IV} - 1)^{1/2}, \beta_{IV} > 1.$$

By Ablowitz-Zappatella on equation (2.24), we look for a solution of the following form:

$$f(z) = (1 + me^{nz})^{-r} \quad m, n, r > 0.$$

By deriving $f(z)$ and getting each term of equation (2.24), we obtain

$$\begin{aligned}
\frac{df}{dz} &= -r(1 + me^{nz})^{-r-1}(nme^{nz}) \\
\frac{d^2f}{dz^2} &= -r(-r-1)(1 + me^{nz})^{-r-2}(nme^{nz})(nme^{nz}) \\
&\quad - r(1 + me^{nz})^{-r-1}(n^2me^{nz}) \\
&= r(r+1)(1 + me^{nz})^{-r-2}(n^2m^2e^{2nz}) - r(1 + me^{nz})^{-r-1}(n^2me^{nz}) \\
f^p + 1 &= (1 + me^{nz})^{-r(p+1)}.
\end{aligned}$$

By substituting each term in equation (2.24),

$$\begin{aligned}
\frac{\partial^2 f}{\partial z^2} + (w_1 + C_I)\frac{\partial f}{\partial z} + (1 - \beta_{VI})f(1 - f) &= \frac{\partial^2 f}{\partial z^2} + (w_1 + C_I)\frac{\partial f}{\partial z} + f - \beta_{VI}f - f^2 + \beta_{VI}f^2 \\
&= r(r+1)(1 + me^{nz})^{-r-2}(n^2m^2e^{2nz}) \\
&\quad - r(1 + me^{nz})^{-r-1}(n^2me^{nz}) \\
&\quad - r(w_1 + C_I)(1 + me^{nz})^{-r-1}(nme^{nz}) \\
&\quad + (1 + me^{nz})^{-r} - \beta_{VI}(1 + me^{nz})^{-r} \\
&\quad - (1 + me^{nz})^{-2r} + \beta_{VI}(1 + me^{nz})^{-2r}.
\end{aligned}$$

By factoring $(1+me^{nz})^{-r-2}$, we get the following

$$\begin{aligned}\frac{\partial^2 f}{\partial z^2}+(w_1+C_I)\frac{\partial f}{\partial z}+f-\beta_{VI}f-f^2+\beta_{VI}f^2 =&(1+me^{nz})^{-r-2}[r(r+1)(n^2m^2e^{2nz})\\ &-r(1+me^{nz})(n^2me^{nz})\\ &-r(w_1+C_I)(1+me^{nz})(nme^{nz})\\ &+(1+me^{nz})^2-\beta_{VI}(1+me^{nz})^2\\ &-(1+me^{nz})^{-r+2}+\beta_{VI}(1+me^{nz})^{-r+2}]\\ =&(1+me^{nz})^{-r-2}[r(r+1)n^2m^2e^{2nz}\\ &-rn^2me^{nz}-rn^2m^2e^{2nz}-C_Irnme^{nz}-C_Irnm^2e^{2nz}\\ &-rw_1nme^{nz}-rw_1nm^2e^{2nz}+1+2me^{nz}\\ &+m^2e^{2nz}-\beta_{VI}-2\beta_{VI}me^{nz}-\beta_{VI}m^2e^{2nz}\\ &-(1+me^{nz})^{-r+2}+\beta_{VI}(1+me^{nz})^{-r+2}].\end{aligned}$$

Now we can factor m^2e^{2nz} and me^{nz},

$$\begin{aligned}(1+me^{nz})^{-r-2}[&m^2e^{2nz}(r(r+1)n^2-rn^2-C_Irn-rnw_1+1-\beta_{VI})\\ &me^{nz}(-rn^2-C_Irn-rnw_1+2-2\beta_{VI})\\ &+1-\beta_{VI}-(1+me^{nz})^{-r+2}+\beta_{VI}(1+me^{nz})^{-r+2}].\end{aligned}$$

Now by solving, we have

$$r(r+1)n^2-rn^2-C_Irn-rnw_1+1-\beta_{VI}=0 \tag{2.26}$$

$$-rn^2-C_Irn-rnw_1+1-\beta_{VI}=-1+\beta_{VI}. \tag{2.27}$$

Then we multiply (2.27) by -1

$$\begin{aligned} r(r+1)n^2 - rn^2 - C_I rn + rnw_1 + 1 - \beta_{VI} &= 0 \\ + rn^2 + C_I rn - rnw_1 - 1 + \beta_{VI} &= 1 - \beta_{VI}, \end{aligned}$$

and by adding the equations together, we get

$$r(r+1)n^2 - 1 + \beta_{VI} = 0, \tag{2.28}$$

where

$$\begin{aligned} -rp + 2 &= 0 \\ \implies r &= \frac{2}{p} \qquad \text{for} \qquad p = 1. \\ \implies r &= 2. \end{aligned}$$

We can now substitute r in equation (2.28) and solve for n:

$$\begin{aligned} r(r+1)n^2 - 1 + \beta_{VI} &= 0 \\ 2(2+1)n^2 - 1 + \beta_{VI} &= 0 \\ 6n^2 - 1 + \beta_{VI} &= 0 \\ 6n^2 &= 1 - \beta_{VI} \\ n &= \pm\sqrt{\frac{1-\beta_{VI}}{6}}. \end{aligned}$$

Going back to equations (2.26) and (2.27), we substitute for n and solve for w_1, so that

$$\begin{aligned}
r(r+1)n^2 - rn^2 - C_I rn - rnw_1 + 1 - \beta_{VI} &= 0 \\
6n^2 - 2n^2 - 2C_I n - 2nw_1 + 1 - \beta_{VI} &= 0 \\
\frac{5}{3}(1-\beta_{VI}) - 2C_I \pm \sqrt{\frac{1-\beta_{VI}}{6}} - 2w_1 \pm \sqrt{\frac{1-\beta_{VI}}{6}} &= 0 \\
C_I \pm \sqrt{\frac{1-\beta_{VI}}{6}} + w_1 \pm \sqrt{\frac{1-\beta_{VI}}{6}} &= \frac{5}{6}(1-\beta_{VI}) \\
\pm\sqrt{\frac{1-\beta_{VI}}{6}}(C_I + w_1) &= \frac{5}{6}(1-\beta_{VI}) \\
(C_I + w_1) &= \frac{5}{6}(1-\beta_{VI})(\pm\sqrt{\frac{6}{1-\beta_{VI}}}) \\
w_1 &= \pm 5\sqrt{\frac{1-\beta_{VI}}{6}} - C_I
\end{aligned}$$

and

$$\begin{aligned}
-rn^2 - C_I rn - rnw_1 + 2 - 2\beta_{VI} &= 0 \\
-2n^2 - 2C_I n - 2nw_1 + 2 - 2\beta_{VI} &= 0 \\
\frac{5}{3}(1-\beta_{VI}) - 2C_I \pm \sqrt{\frac{1-\beta_{VI}}{6}} - 2 \pm \sqrt{\frac{1-\beta_{VI}}{6}} w_1 &= 0 \\
\frac{5}{3}(1-\beta_{VI}) - 2 \pm \sqrt{\frac{1-\beta_{VI}}{6}}(w_1 + C_I) &= 0 \\
\pm\sqrt{\frac{1-\beta_{VI}}{6}}(w_1 + C_I) &= \frac{5}{6}(1-\beta_{VI}) \\
w_1 &= \pm 5\sqrt{\frac{1-\beta_{VI}}{6}} - C_I.
\end{aligned}$$

Therefore, the solution of equation (2.24) is

$$f(z) = (1 + me^{nz})^{-2} \quad \text{where} \quad z = x - wt.$$

Now, by substituting the solution into (2.22), we check that the solution satisfies the equation. Recall

that

$$\begin{aligned}\frac{\partial^2 f}{\partial z^2}+(w_1+C_I)\frac{\partial f}{\partial z}+(1-\beta_{VI})f(1-f) =&(1+me^{nz})^{-r-2}\\ &[m^2e^{2nz}(r(r+1)n^2-rn^2\\ &-C_Irn+rnw_1+1-\beta_{VI})\\ &me^{nz}(-rn^2-C_Irn\\ &-rnw_1+2-2\beta_{VI})\\ &+1-\beta_{VI}-(1+me^{nz})^{-r+2}\\ &+\beta_{VI}(1+me^{nz})^{-r+2}]\end{aligned}$$

and by substituting r, w and n we get

$$\begin{aligned}\frac{\partial^2 f}{\partial z^2}+(w_1+1)\frac{\partial f}{\partial z}+(1-\beta_{VI})f(1-f) =&(1+me^{nz})^{-4}\\ &[m^2e^{2nz}((1-\beta_{VI})(1-\frac{1}{3}-\frac{5}{3}+1)\\ &+(-2C_I+2C_I)(\pm\sqrt{\frac{1-\beta_{VI}}{6}}))\\ &me^{nz}((1-\beta_{VI})(-\frac{1}{3}-\frac{5}{3}+2)\\ &+(-2C_I+2C_I)(\pm\sqrt{\frac{1-\beta_{VI}}{6}}))\\ &+1-\beta_{VI}-1+\beta_{VI}]\\ =&0.\end{aligned}$$

Since we get 0, then it is a solution. Now, for equation (2.25), we have

$$\begin{aligned}\frac{\partial^2 g}{\partial z^2}+(w_2+C_V)\frac{\partial g}{\partial z}+(\beta_{IV}-1)g(1-g)=0\\ \frac{\partial^2 g}{\partial z^2}+(w_2+C_V)\frac{\partial g}{\partial z}-g+\beta_{IV}g+g^2-\beta_{IV}g^2=0.\end{aligned}$$

Recall that if $f+g=1$ then $g=1-f$, so the solution is

$$\begin{aligned} g(z) &= 1-f(z) \\ &= 1-(1+me^{nz})^{-2}, \quad \text{where} \quad z=x-wt. \end{aligned}$$

□

2.2.1 Exact solutions

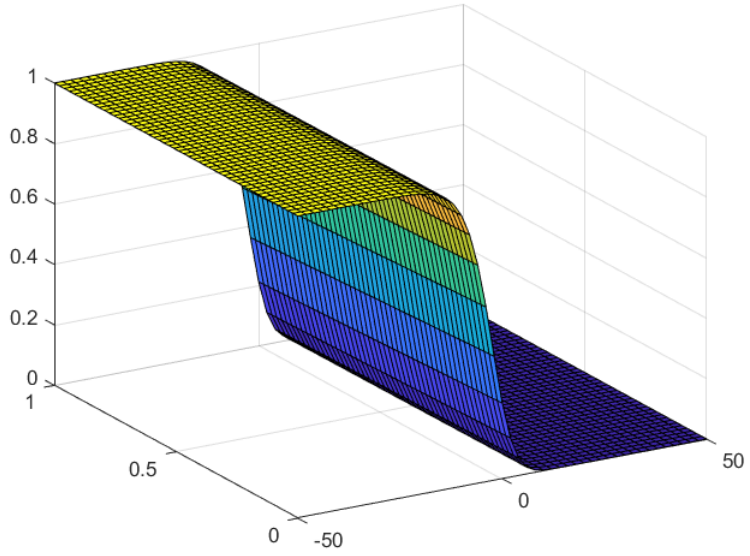

Figure 2.1: Exact solution for M_I

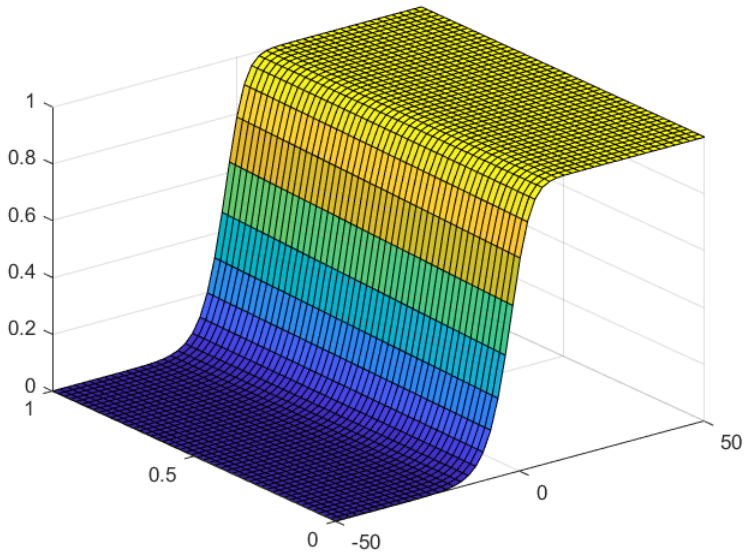

Figure 2.2: Exact solution for M_V

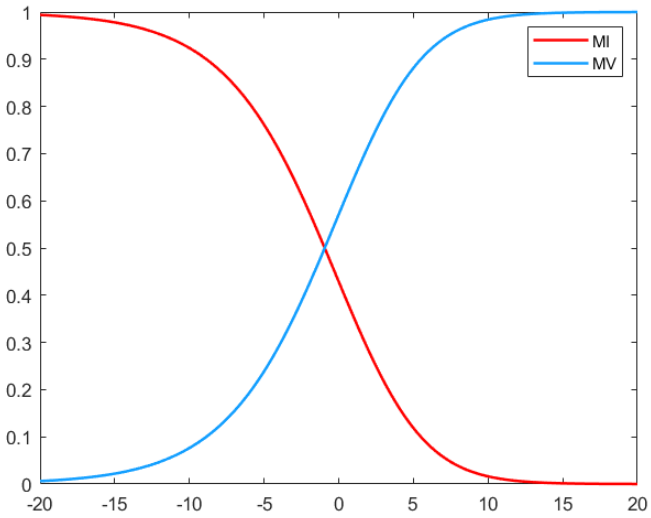

Figure 2.3: Exact solutions for M_I and M_V

2.2.2 Numerical Model 1

Consider the following stochastic Fisher-Kolmogorov system ($D = 3$, $\gamma = 1$, $\beta_1 + \beta_2 = 2$, $C = \sqrt{3}$, $\sigma = 1$):

$$dm = (2.5m_{xx} + 2\sqrt{3}m_x + (1 - m - \beta_1 n)m)dt + m_x dW_t^1 \tag{2.29}$$

$$dn = (2.5n_{xx} + 2\sqrt{3}n_x + (1 - n - \beta_2 m)n)dt + n_x dW_t^2 \tag{2.30}$$

where

$$m(0,z) = \left(1 + k\exp\left(n_1 x/\sqrt{3}\right)\right)^{-2} \tag{2.31}$$

$$n(0,z) = 1 - m(0,z), \tag{2.32}$$

with $n_1 = \sqrt{(1-\beta_1)/6}$ for $t \in [0,1]$. By Theorem 1 equation (2.29)-(2.30) has a solution $m(t,z) = M(t,X_t^1)$ and $n(t,z) = N(t,X_t^2)$ such that $M(t,x)$ and $N(t,x)$ are explicit solutions of the deterministic equations

$$\partial_t M = 3\partial_{xx}M + (1 - M - \beta_1 N)M + \sqrt{3}\partial_x M \tag{2.33}$$

$$\partial_t N = 3\partial_{xx}N + (1 - N - \beta_2 M)N + \sqrt{3}\partial_x N \tag{2.34}$$

given by

$$M(t,z) = \left(1 + k\exp\left(n_1\left(x/\sqrt{3} - w_1 t\right)\right)\right)^{-2} \tag{2.35}$$

$$N(t,z) = 1 - M(t,z), \tag{2.36}$$

with $w_1 = -5\sqrt{(1-\beta_1)/6} - \sqrt{3}$. Also X_t^1 and X_t^2 are solutions of

$$dX_t^1 = \sqrt{3}dt + dW_t^1 \tag{2.37}$$

$$dX_t^2 = \sqrt{3}dt + dW_t^2 \tag{2.38}$$

with initial state $X_0^1 = z$ and $X_0^2 = z$ and for $t \in [0,1]$.

The solutions are given by $X_t^1 = z + \sqrt{3}t + W_t^1$ and $X_t^2 = z + \sqrt{3}t + W_t^2$ for $t \in [0,1]$. Therefore, a particular solution of the stochastic system is

$$m(t,z) = \left(1 + k\exp\left(n_1\left(\left(z + \sqrt{3}t + W_t^1\right)/\sqrt{3} - w_1 t\right)\right)\right)^{-2} \tag{2.39}$$

$$n(t,z) = 1 - \left(1 + k\exp\left(n_1\left(\left(z + \sqrt{3}t + W_t^2\right)/\sqrt{3} - w_1 t\right)\right)\right)^{-2}. \tag{2.40}$$

See figure (2.4)- (2.10) for the graph results.

In the following algorithm, we discretize the time and spatial interval and create a stochastic mesh. We then use the central difference on the deterministic and add boundary conditions. Using ode45 we solve the model in one function. We then compare the numerical solutions to the exact solutions and get the error.

Algorithm: Figures (2.4)-(2.10) Main script

Write down all the parameters for both equations and solution

Discretize the time interval [0,T]. Similarly discretize, for the space interval [a,b].

Write down the stochastic mesh and plot it

Write down both functions and boundary conditions for each

Write the number of entries for initial condition of both functions

and combine them into one array

Use a finite (central) difference for time and space for the equations

Use ode45 to solve for the model

Separate the solutions for each equation by using the correct indices

Plot each of the solutions using surf plot

Using the **for** command write down an iteration method to solve for each of the exact solutions

Plot each by using a surf plot

Get the error of the numerical solutions

Plot the error using surf plot

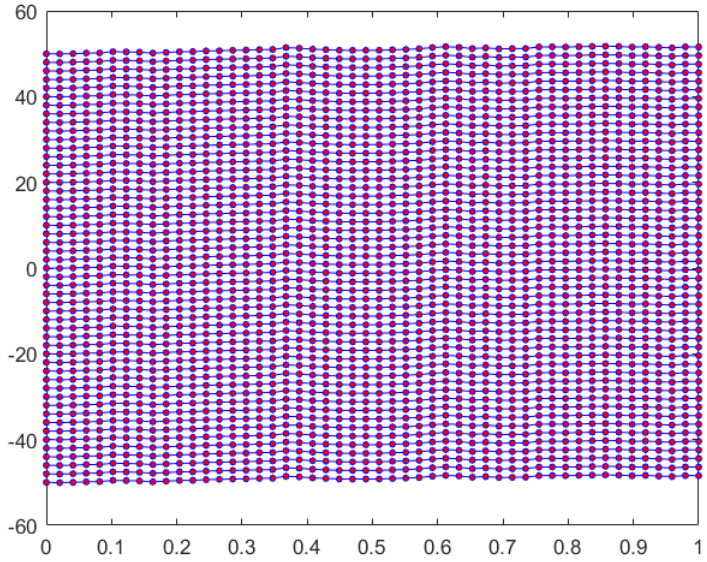

Figure 2.4: Stochastic Mesh

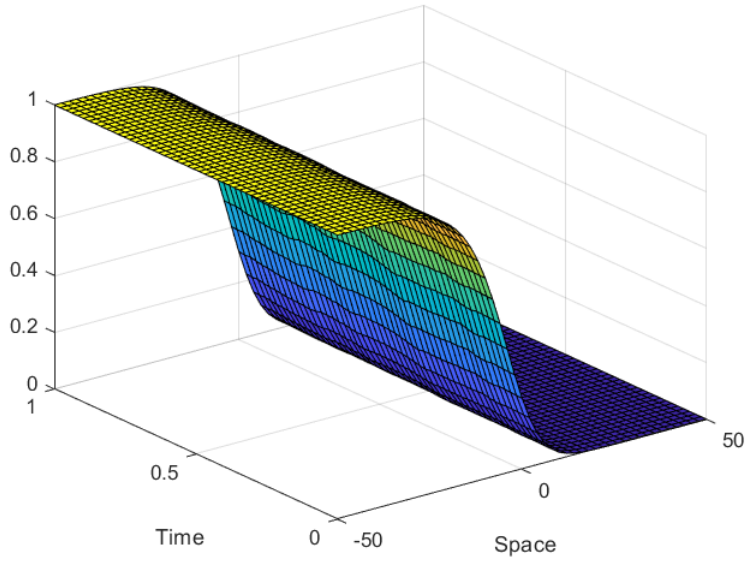

Figure 2.5: Numerical solution for M

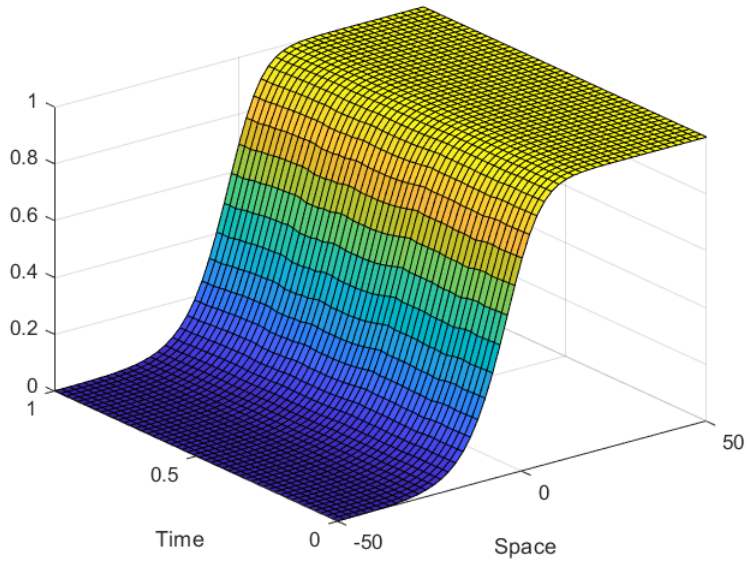

Figure 2.6: Numerical solution for N

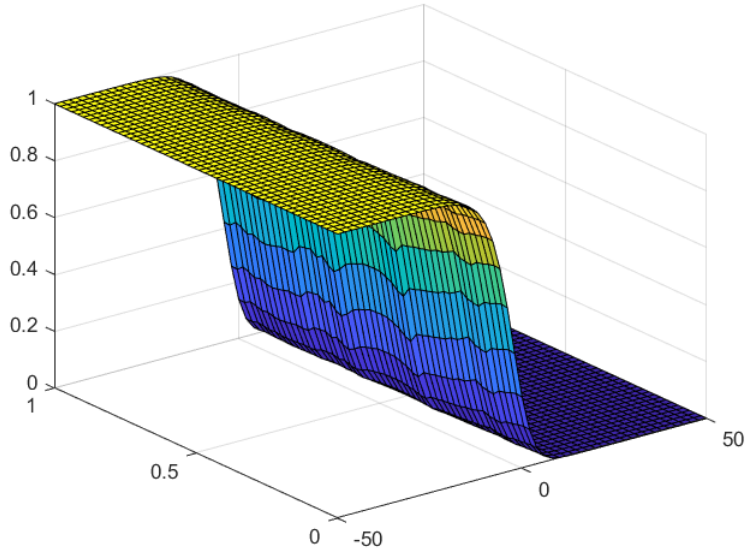

Figure 2.7: Exact solution for M

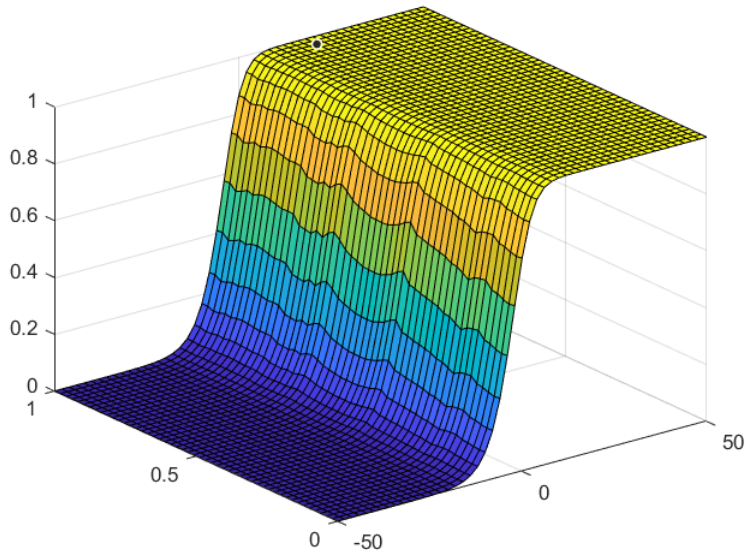

Figure 2.8: Exact solution for N

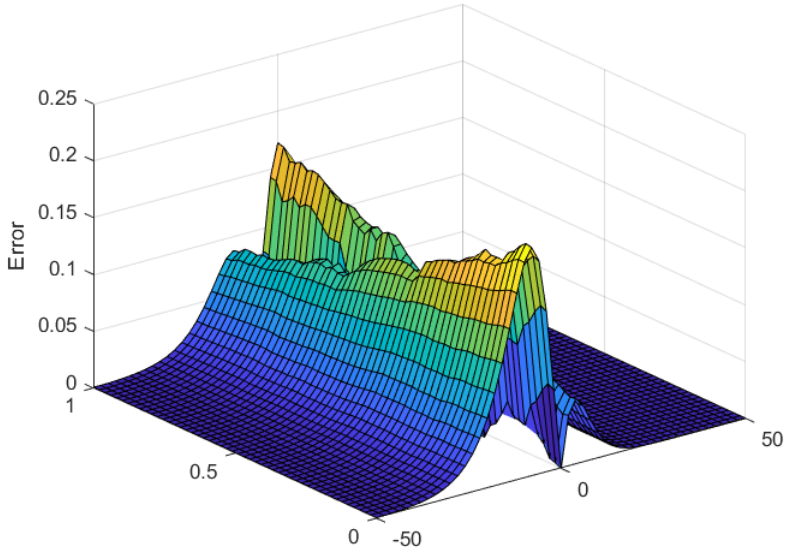

Figure 2.9: Error for M

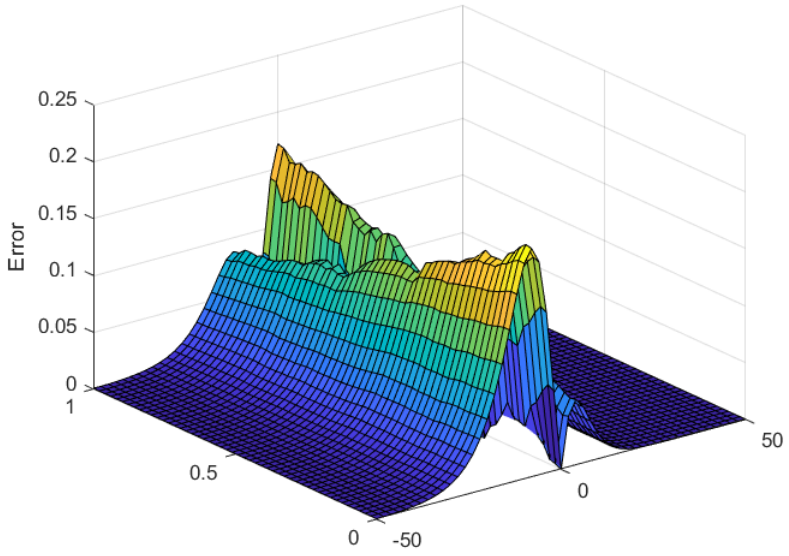

Figure 2.10: Error for N

CHAPTER III

SIR WITH NETWORKING

In this chapter, we add diffusion to the Lokta-voltera equations from chapter 1, obtaining Fisher-Kolmogorov equations. Using the principle of competitive exclusion and Fisher-Kolmogorov equations, the disease and and the vaccine will compete until one of them goes extinct. We will then coupled the standard SIR model with the information provided by a network. The network information is rendered by a couple system of Fisher-Kolmogorov equations. We solve the Fisher-Kolmogorov system by finding the exact solutions using Ablowitz Zappatella. The solutions will be employed in MATLAB and compared to each other. We can see different situations of the SIR model and how the disease and vaccine react to one another by changing the parameters.

3.1 Classical SIR

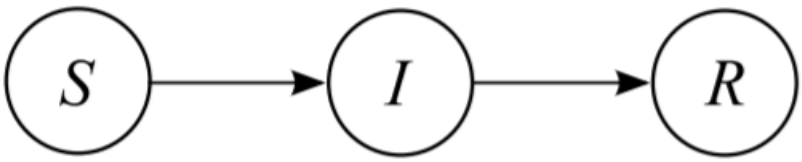

Figure 3.1: Flow of SIR model

In figure (3.1), we see the flow of the SIR model, where the three different populations interact: susceptible population S, the infected population I and the removed population R. A susceptible person can only get infected and once infected, the person can be removed by quarantine, vaccinated, or death. The variables of interest can be consider as proportions such that $S+I+R=1$.

Yet, if we consider each as actual populations then

$$S+I+R=N,$$

where N is a constant that is defined as a closed population. By figure (3.1), we let $S(t)$ denote the number of individuals who are susceptible to the disease. $I(t)$ denotes the number of infected individuals, assumed infectious and able to spread the disease by contact with a susceptible. $R(t)$ denotes the number of individuals who have been infected and then removed from the possibility of being infected again or of spreading infection. An individual can go from susceptible to infected by a force of infection. If there is interaction between the susceptible people and the infected ones, a fraction of the population will become infected. We will call this effect β, the transmission rate. Now, from I to R, there is the rate at which people become cured or removed rate which will be denoted as γ. We always regard β and γ as positive quantities. With all this information we can see a more clear diagram as in figure (3.2).

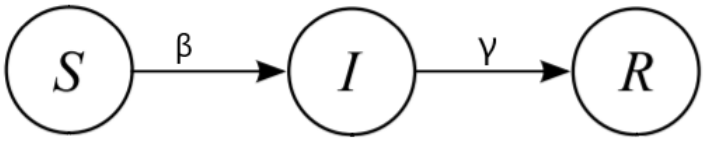

Figure 3.2: Classical SIR model

An important note is that R is a variable that is redundant. In most of the cases we will ignore the R variable. Thus, we will mainly analyze the differential equations form S and I. From figure (3.2), we can also construct a model, for the time-rate of change of $S(t), I(t)$ and $R(t)$. Which

is a set of ordinary differential equations:

$$\begin{aligned} \frac{dS}{dt} &= -\beta SI \\ \frac{dI}{dt} &= \beta SI - \gamma I \\ \frac{dR}{dt} &= \gamma I \end{aligned} \tag{3.1}$$

By using the following algorithm, we created a code in MATLAB for the SIR model. We graphed the model with a fixed population, fixed removed rate γ and random rate of of infection β. In the function script, we use the function command to combine the set of ordinary differential equations in (3.1) together in one array.

Algorithm 1: figure (3.3) function script

function

Since all equations are under the same variable, we separate them:

Write the equation for $\frac{dS}{dt}$

Write the equation for $\frac{dI}{dt}$

Write the equation for $\frac{dR}{dt}$

Make the variable equal to the derivative, so it can put them all together.

end

For the main script we have the initial values for each rate of change equation. Constants parameters are used and the time vector is discretized. Using ode45 the model is solved and then it is plotted to show each population.

Algorithm 2: figure (3.3) Main script

Create an initial value for S,I and R

Initialize the parameters

Discretize the time vector

Use ode45 to solve the model

Plot each equation

Create a legend for the equations and label the x and y axis appropriately.

From the equations in (3.1), we get the following graph:

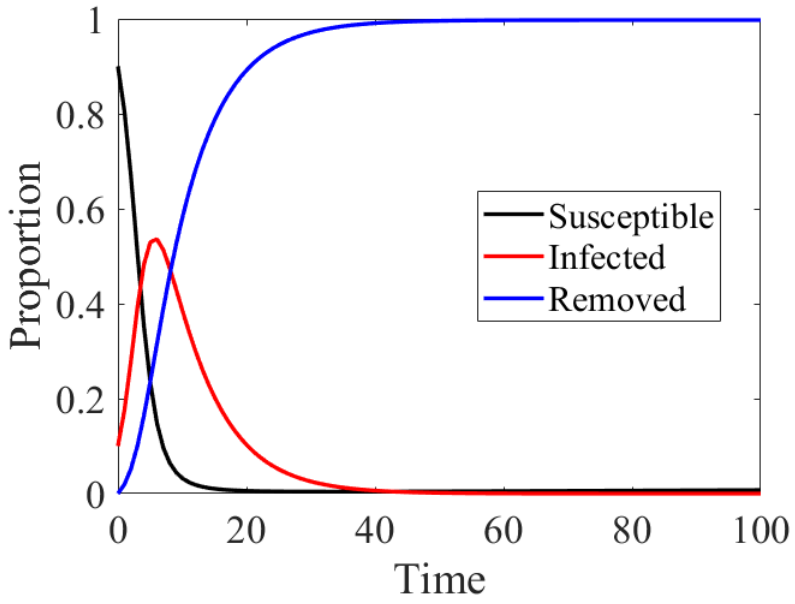

Figure 3.3: SIR graph model

3.2 SIR Model with networking

In this section, we introduce the SIR with networking:

$$\begin{aligned}
\frac{\partial S(x,t)}{\partial t} &= -\beta(1-p(x,t))S(x,t)I(t) - p(x,t)S(x,t) \\
\frac{dI(t)}{dt} &= \beta I(t)\int_0^\infty (1-p(x,t))S(x,t)dx - \gamma I(t) \\
\frac{dR(t)}{dt} &= \int_0^\infty p(x,t)S(x,t)dx + \gamma I(t).
\end{aligned} \tag{3.2}$$

We start by considering $S(x,t)$, the density of the susceptible individuals at a distance x from the source 0, which is the center of information.

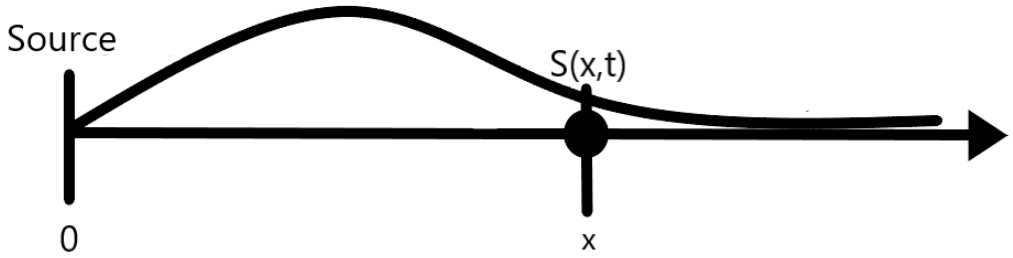

Figure 3.4: Susceptible with x and t.

Thus, we get

$$S(t) = \int_0^\infty S(x,t)dx.$$

$I(t)$ is the fraction of infected in the population and $R(t)$ is the fraction removed in the population. As well, we have $p(x,t)$ which is the probability an individual at a distance x is going to vaccinate at a time t. the rate of change of the assembled vaccinated individuals at a time t at distance x is given by

$$\frac{dV(t)}{dt} = \int_0^\infty p(x,t)S(x,t)dx.$$

The classical SIR model diagram is now updated to the following:

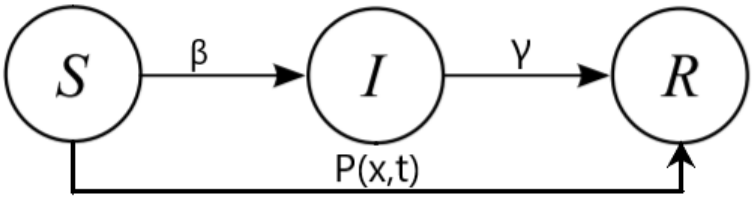

Figure 3.5: SIR with $p(x,t)$

There is a probability that is a common practice for competition of information, where people can make the decision to vaccinate [7]. We see the exponential difference between $M_I(x,t)$ and $M_V(x,t)$ in the probability as follows:

$$p(x,t) = \frac{1}{1+e^{-(C_1 M_I(x,t) - C_2 M_V(x,t))}}.$$

The information about the disease and vaccine at time t and distance x are M_I and M_V. Here, the disease and the vaccine will compete until one of them goes extinct. Using the principle of competitive exclusion and Fishers-Komogorov equations, we consider the network of information modeled as:

$$\begin{aligned}
\frac{\partial M_I}{\partial t} &= D_I \frac{\partial^2 M_I}{\partial x^2} + \gamma_I (1 - M_I - \beta_{VI} M_V) M_I + C_I \frac{\partial M_I}{\partial x} \\
\frac{\partial M_V}{\partial t} &= D_V \frac{\partial^2 M_V}{\partial x^2} + \gamma_V (1 - M_V - \beta_{IV} M_I) M_V + C_V \frac{\partial M_V}{\partial x}
\end{aligned}$$

$$\begin{aligned}
M_I(x,0) &= \varepsilon_I I(0) \delta_0(x) \\
M_V(x,0) &= 0 \\
M_I(0,t) &= \varepsilon_I I(t) \\
M_V(0,t) &= \varepsilon_V V(t).
\end{aligned} \tag{3.3}$$

3.2.1 SIR Model with Networking

In the previous chapter, in proposition 1, we found the exact solutions for $M_I(x,t)$ and $M_V(x,t)$ with constant coefficients. The exact solutions are then applied on the probability since the probability was the exponential difference between $M_I(x,t)$ and $M_V(x,t)$. Using MATLAB, we will apply the network information into the probability. The probability will then be used in the SIR model (3.2), and by ode45 we will solve the model with the exact solutions in two MATLAB scripts.

For the next algorithm, we imply the exact solutions of $M_I(x,t)$ and $M_V(x,t)$. Then they are applied into the probability $p(x,t)$. The probability is then applied to the SIR model, where it is then solved in the main script.

Algorithm 1 figures (2.23) and (2.24): function script

function

Globalize the parameters use in both scripts

Create an array for n points plus one

Put the exact solutions for $M_I(x,t)$ and $M_V(x,t)$ and write a fixed $p(x,t)$

for 1:n

Compute the derivative of S(x,t) with correct indices

end

Compute the derivative of I(t) for n+1 as well as using trapezoid rule to integrate for x

end

In the second script, we write down all our parameters and the time and space discretization, plot the graphs and use Ode45 to calculate the derivatives of the function script.

Algorithm 2 figures (2.23) and (2.24): Main script

function

Globalize the parameters use in both scripts and initialize the rest of the parameters

Create time and space vectors for the discretization

Create an initial condition for M_I and M_V

Use Ode45 to solve the model and plot I

for i=1:m

Use trapezoid function to solve for S(i)

end

Plot the graph for S

We get the following two graphs showing the data for the S and I, where the parameters are fixed and using the special case solutions along with a fixed p. The speed of the equations were changed to get better results.

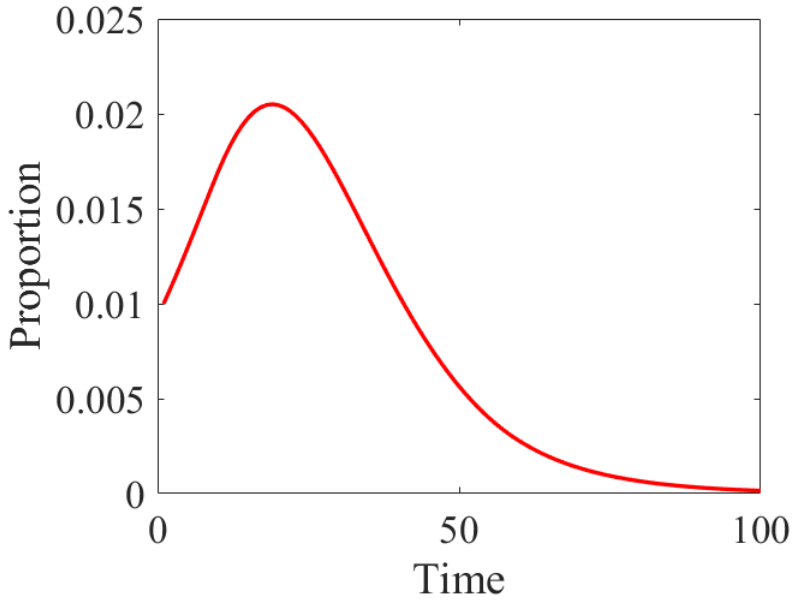

Figure 3.6: Proportion of Infected-from exact

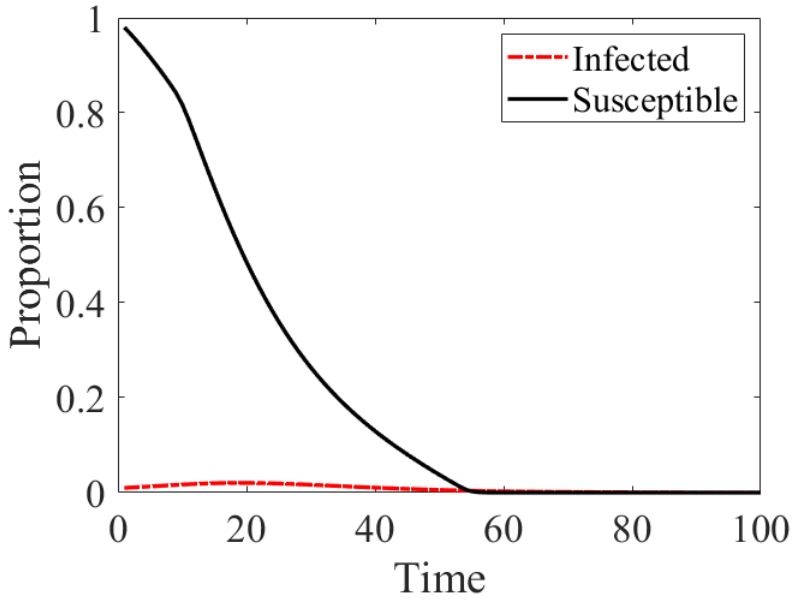

Figure 3.7: With Susceptible

3.3 Numerical Results

We are going to numerically solve the coupled system (2.18) and (2.19) by approximating the derivatives with finite differences. By discretization of the spatial domain and time interval (if applicable) we will use the boundaries and initial conditions to solve values or nearby points. Using finite difference, we can put this coupled system into matrices which can then be solved by using MATLAB to get numerical solutions. This procedure will require the use of Taylor's expansion and in some integrals of the model will be solved by the trapezoid rule, which is already integrated in MATLAB.

3.3.1 Numerical solutions for the Fisher-Kolmogorov system

We will use finite difference on the coupled equations (2.18) and (2.19) to obtain numerical solutions. Using the central difference over the space variable, we will approximate the right side of the equations (2.18) and (2.19).

The bi-variate Taylor expansion of $U(t+k,x)$ about (t,x) gives us the following approxima-

tions:

$$U(t+k,x)-U(t,x)\approx k\partial_t U(t,x)$$

and

$$U(t,x\pm h)-U(t,x)\approx \pm h\partial_x U(t,x)+\frac{1}{2}h^2\partial_{xx}U(t,x),$$

which is also equivalent to

$$\begin{bmatrix} h & \frac{h^2}{2} & 0 \\ -h & \frac{h^2}{2} & 0 \\ 0 & 0 & k \end{bmatrix}\begin{bmatrix} \partial_x U(t,x) \\ \partial_{xx}u(t,x) \\ \partial_t U(t,x) \end{bmatrix}=\begin{bmatrix} U(t,x+h)-U(t,x) \\ U(t,x-h)-U(t,x) \\ U(t+k,x)-U(t,x) \end{bmatrix},$$

Thus, for $\partial_x U(t,x)$ and $\partial_{xx}U(t,x)$ we get

$$\partial_x U(t,x)=\frac{U(t,x+h)-U(t,x-h)}{2h} \tag{3.4}$$

$$\partial_{xx}U(t,x)=\frac{U(t,x+h)-2U(t,x)+U(t,x-h)}{h^2} \tag{3.5}$$

and for $\partial_t U(t,x)$

$$\partial_t U(t,x)=\frac{U(t+k,x)-U(t,x)}{k}. \tag{3.6}$$

Similarly, we do the same for $V(t,x)$. We can now substitute (3.4)-(3.6) into equations (2.18) and (2.19) by letting $U(t,x)=M_I(x,t)$ and $V(t,x)=M_V(x,t)$, we get the following:

$$\partial_t U(t,x) = D_I \left(\frac{U(t,x+h) - 2U(t,x) + U(t,x-h)}{h^2} \right)$$
$$+ \gamma_I (1 - U(t,x) - \beta_{VI} V(t,x)) U(t,x)$$
$$+ C_I \left(\frac{U(t,x+h) - U(t,x-h)}{2h} \right)$$
$$\partial_t V(t,x) = D_V \left(\frac{V(t,x+h) - 2V(t,x) + V(t,x-h)}{h^2} \right)$$
$$+ \gamma_V (1 - V(t,x) - \beta_{IV} U(t,x)) V(t,x)$$
$$+ C_V \left(\frac{V(t,x+h) - V(t,x-h)}{2h} \right).$$

To solve for the model (3.3) we will be using finite difference and putting them as matrices in MATLAB. As a result we obtain the following algorithms:

Algorithm 1 function script for figures (3.8) - (3.13)

function

Globalize the parameters used in both scripts

Create an array for n-1 points for M_I, M_V and S

Write the boundary conditions for both M_I and M_V

Write down the matrix for the discretization

Multiply the appropriate parts of the matrix to the appropriate part of the array with M_I,M_V and S

Write down the probability and compute the equations for M_I and M_V

Compute the derivative of S(x,t) with correct indices

Compute the derivative of I(t) with the trapezoid rule on S(x,t) with correct indices.

Compute the rate of the vaccine with correct indices

Put all of the equations together in one array

end

In the second script, we give values to the parameters and discretize the time and spatial vectors. We then use ode45 to solve for the function script. Plots are then used to show the numerical solutions.

Algorithm 2: Plot script for figures (3.8) - (3.13)

Globalize the parameters used in both scripts

Initialize the rest of the parameters

Initialize both time and space vectors for the discretization

Write down the initial conditions for M_I, M_V, S, I and for $V(vaccine)$ in one vector

Use ode45 to solve for the function script

Put the boundary conditions with the correct indices

Separate the array to get the solution for M_I, M_V, S, I and V

Plot each solution

By putting the same values that we put for the exact solution, we get a similar graph to figure (3.6). The difference is that this graph has different boundary conditions, so there is a small difference between the graphs.

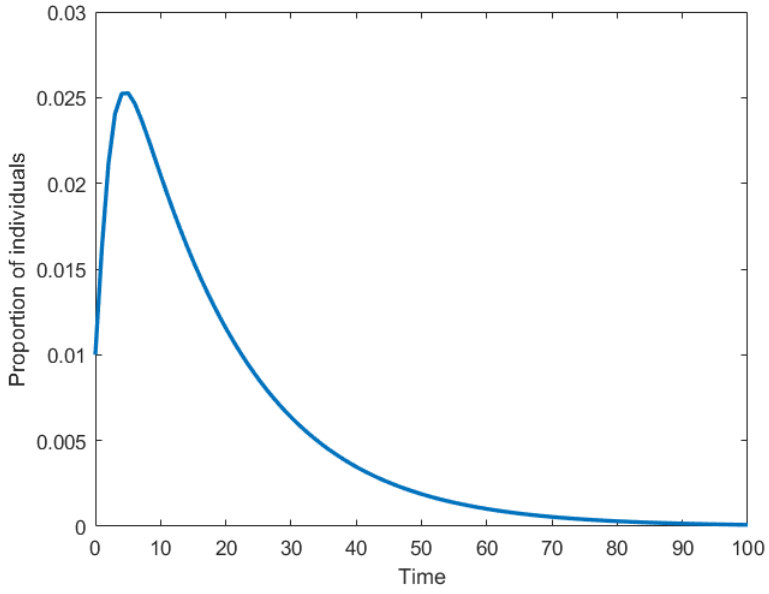

Figure 3.8: Proportion of Infected-comparison to exact

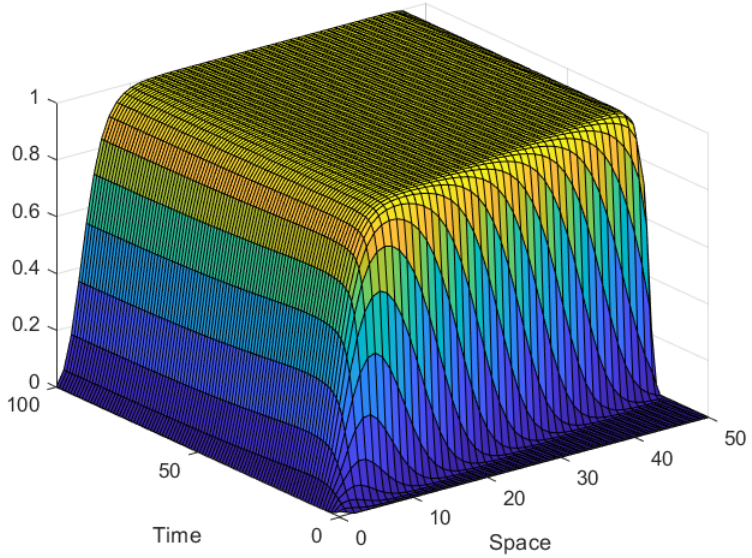

Figure 3.9: Surf of M_I

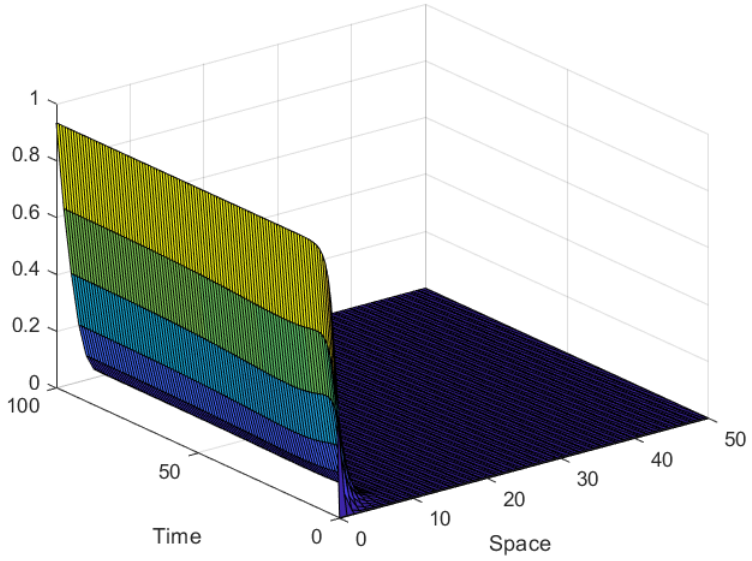

Figure 3.10: Surf of M_V

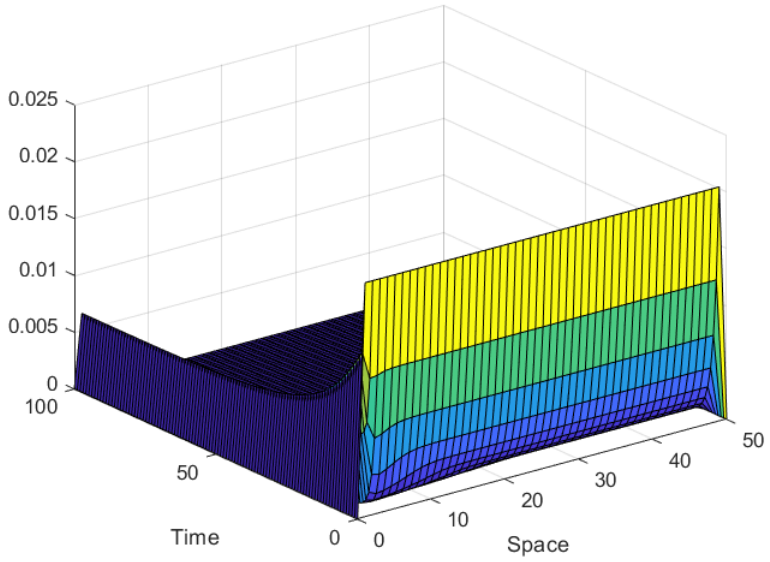

Figure 3.11: Surf of $S(x,t)$

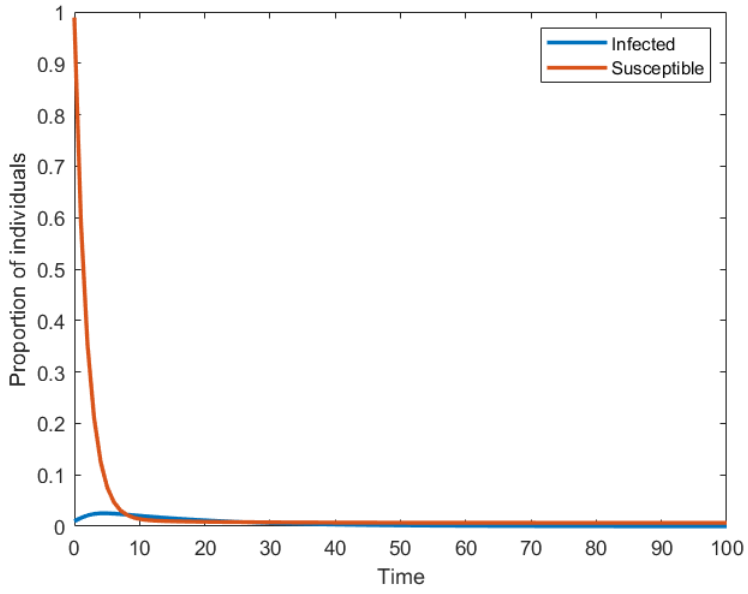

Figure 3.12: Proportion of Susceptible-Infected

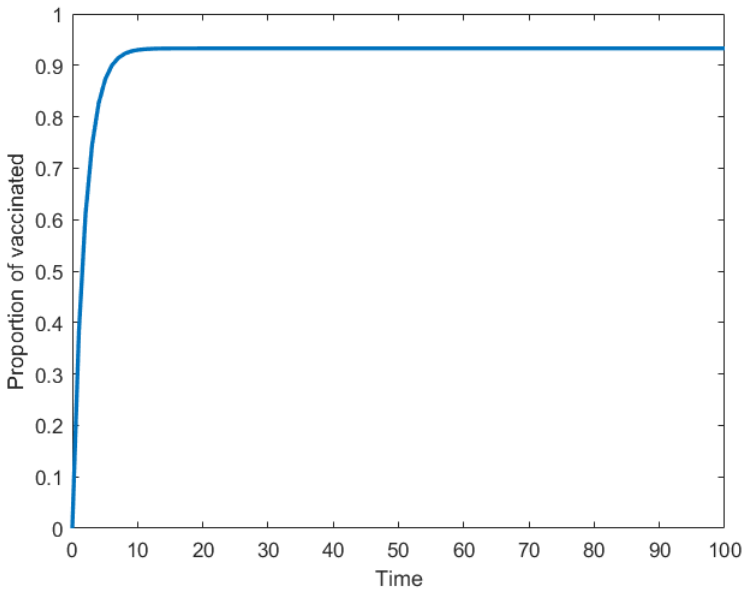

Figure 3.13: Proportion of Vaccinated

3.3.2 Computational Simulations

We are going to plot several solutions where all the parameters are one except the diffusion coefficient and the parameters used for the probability. For the time and space discretization, we consider $n = 49$, $m = 100$, $x_0 = 0$, $x_n = 50$, $t_0 = 0$ and $t_m = 100$. The three cases we will use for the diffusion are $D_I = 500D_V, D_I = D_V$ and $500D_I = D_V$. For figure (3.14) and figure (3.15), the parameters used for the probability are $C_I = 50$ and $C_V = 50$, and the parameters used for the SIR model are $\beta = \gamma = 1$.

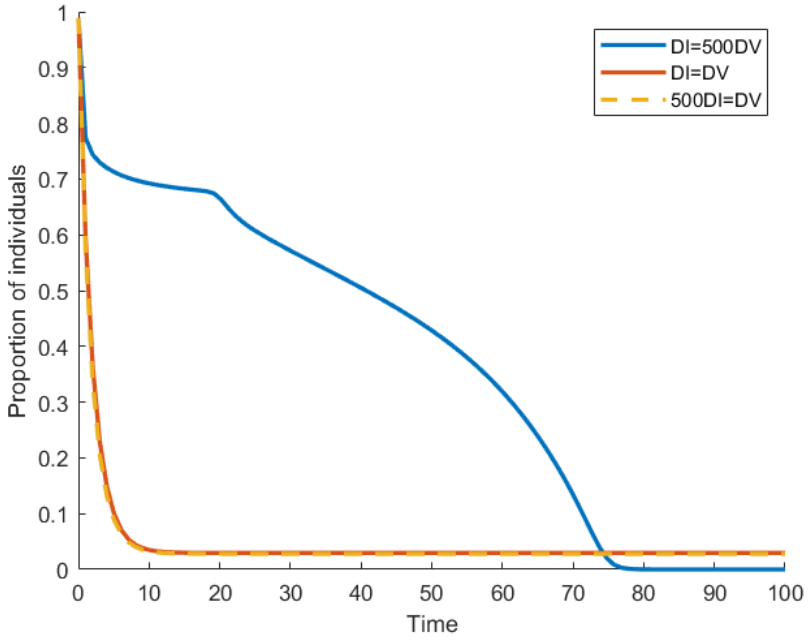

Figure 3.14: Solution for S ($\beta = \gamma = 1$)

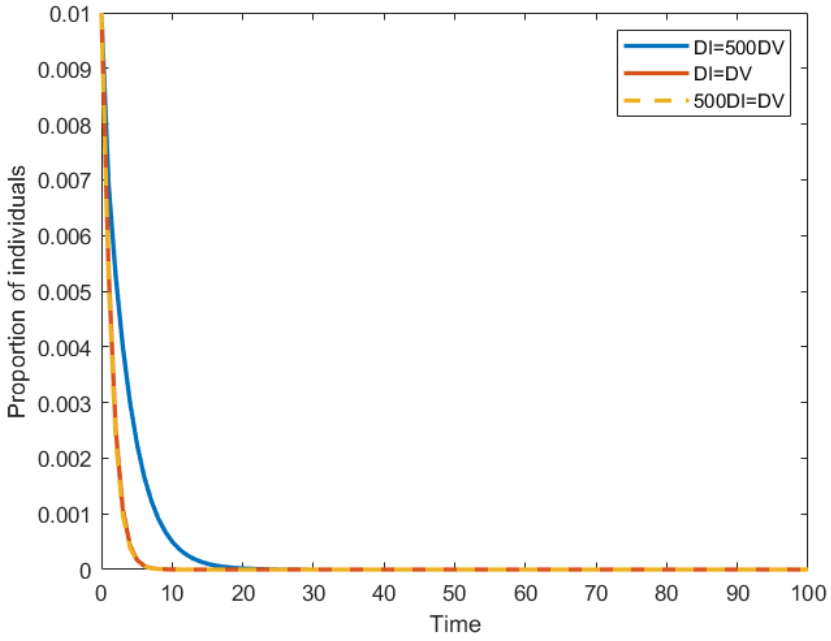

Figure 3.15: Solution for I ($\beta = \gamma = 1$)

In the next simulation we are going to use the same parameters from the past simulation except that we are going to change the SIR model parameters. For figures (3.16) and (3.17), we will

let $\beta = 10$ and $\gamma = 1$.

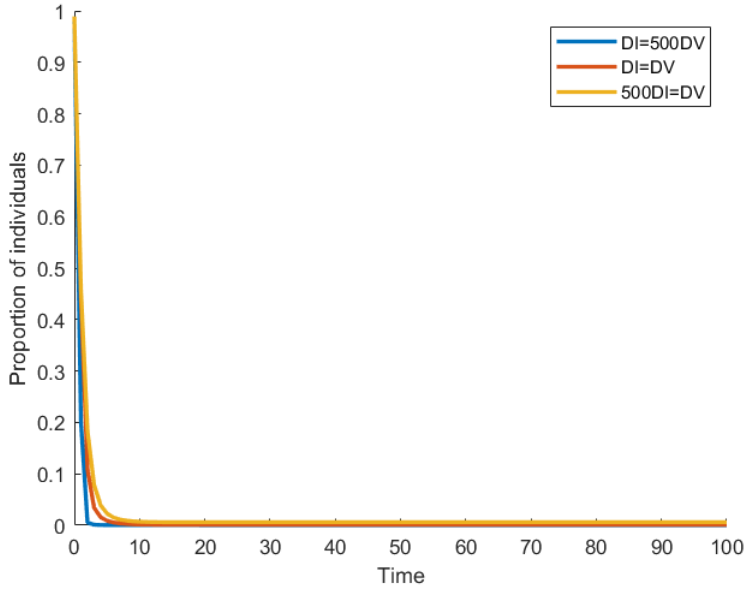

Figure 3.16: Solution for S $(\beta = 10, \gamma = 1)$

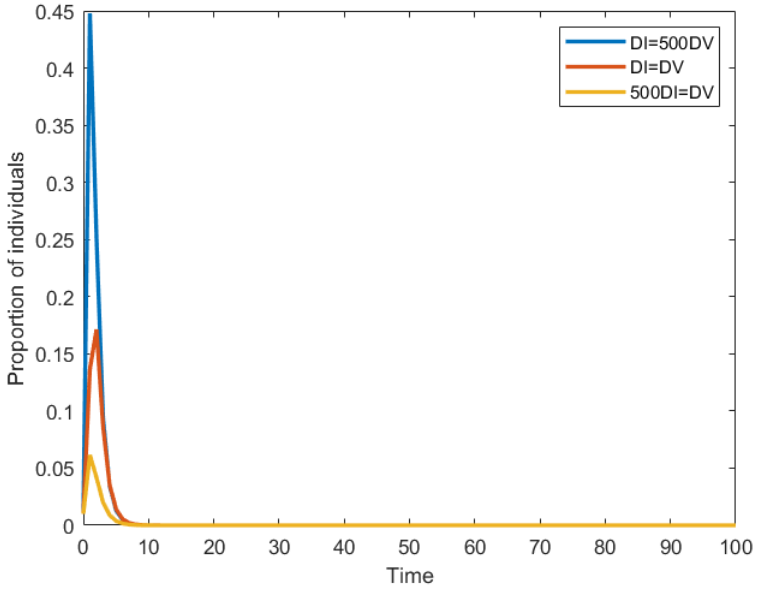

Figure 3.17: Solution for I $(\beta = 10, \gamma = 1)$

Similarly, in this next simulation we will leave the probability the same and just change the parameters from the SIR model. For figure (3.18) and figure (3.19), we will let $\beta = 1$ and $\gamma = 10$.

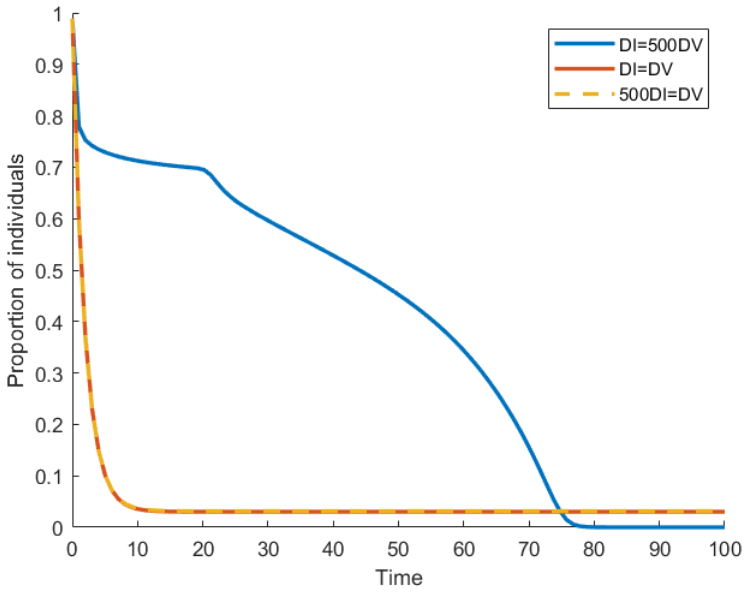

Figure 3.18: Solution for S ($\beta = 1, \gamma = 10$)

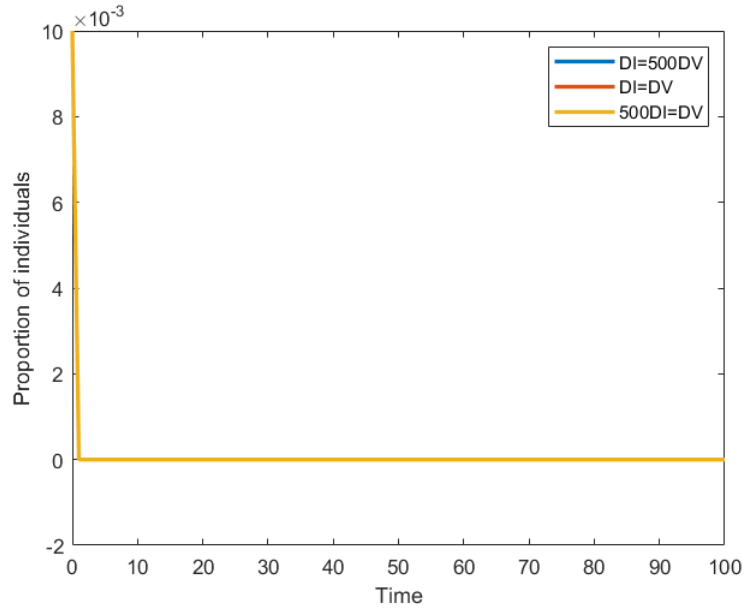

Figure 3.19: Solution for I ($\beta = 1, \gamma = 10$)

CHAPTER IV

CONCLUSION AND FUTURE RESEARCH

Fisher-Kolmogorov equations are reaction diffusion equations which have been used extensively in mathematical modeling, such as with low-grade glioma, biology and more. In this thesis, explicit solutions and numerical simulations for a stochastic Fisher-Kolmogorov system with random noise were presented. Furthermore, the coupled Fisher-Kolmogorov equations are incorporated with the SIR (Susceptible-Infected-Recovered) model to examine the anticipated mutual influence: The disease and the vaccine will compete until one of them goes extinct. Using the principle of competitive exclusion incorporated in Fisher-Kolmogorov equations, we considered the network of information modeled by equation (3.3). The results showed how the propagation of information about the disease impacts the probability of vaccination and, consequently, the vaccination rate.

As part of future research the analysis of spreading speeds and traveling wave speeds of the Fisher-Kolmogorov equation can be extended to include spatial- or time-dependent heterogeneity in the growth rate and diffusion coefficients, see [5].

BIBLIOGRAPHY

[1] J. BELMONTE-BEITIA, *Existence of travelling wave solutions for a fisher–kolmogorov system with biomedical applications*, Communications in Nonlinear Science and Numerical Simulation, 36 (2016), pp. 14–20.

[2] J. BELMONTE-BEITIA, G. F. CALVO, AND V. M. PEREZ-GARCIA, *Effective particle methods for fisher–kolmogorov equations: theory and applications to brain tumor dynamics*, Communications in Nonlinear Science and Numerical Simulation, 19 (2014), pp. 3267–3283.

[3] P. E. KLOEDEN AND E. PLATEN, *Higher-order implicit strong numerical schemes for stochastic differential equations*, Journal of statistical physics, 66 (1992), pp. 283–314.

[4] ——, *Numerical solution of stochastic differential equations*, vol. 23, Springer Science & Business Media, 2013.

[5] M. A. LEWIS, S. V. PETROVSKII, AND J. R. POTTS, *The mathematics behind biological invasions*, vol. 44, Springer, 2016.

[6] J. D. MURRAY, *Mathematical biology: I. An introduction*, vol. 17, Springer Science & Business Media, 2007.

[7] T. ORABY AND C. T. BAUCH, *Bounded rationality alters the dynamics of paediatric immunization acceptance*, Scientific reports, 5 (2015), pp. 1–12.

[8] V. M. PÉREZ-GARCÍA, M. BOGDANSKA, A. MARTÍNEZ-GONZÁLEZ, J. BELMONTE-BEITIA, P. SCHUCHT, AND L. A. PÉREZ-ROMASANTA, *Delay effects in the response of low-grade gliomas to radiotherapy: a mathematical model and its therapeutical implications*, Mathematical medicine and biology: A journal of the IMA, 32 (2015), pp. 307–329.

code1

APPENDIX A

APPENDIX A

MATLAB CODE

Figure 3.3: SIR Graph Model function code

```
1 function yprime=sirodes1(~,y,mu,beta,gamma)
2 % y(1) is S and y(2) is I
3 %yprime(1) is S' and yprime(2) is I'
4 yprime(1)=mu-beta*y(1)*y(2)-mu*y(1);
5 yprime(2)=beta*y(1)*y(2)-(mu+gamma)*y(2);
6 yprime(3) = gamma*y(2);
7 yprime=yprime';
8 end
```

Figure 3.3SIR Graph Model plot code

```
1 initial=[0.9;0.1;0];
2 t=0;
3 T=100;
4 tspan=linspace(t,T,100);
5 mu = 1/(365*50);
6 beta = rand;
7 gamma = 1/7;
8 [T,Y]=ode45(@sirodes1,tspan,initial,[],mu,beta,gamma);
9
10 figure1 = figure;
```

```
11
12 % Create axes
13 axes1 = axes('Parent',figure1);
14 hold(axes1,'on');
15
16 % Create multiple lines using matrix input to plot
17 plot1 = plot(T,Y,'LineWidth',2,'Parent',axes1);
18 set(plot1(1),'DisplayName','Susceptible','Color',[0 0 0]);
19 set(plot1(2),'DisplayName','Infected','Color',[1 0 0]);
20 set(plot1(3),'DisplayName','Recovered','Color',[0 0 1]);
21
22 % Create ylabel
23 ylabel('Proportion');
24
25 % Create xlabel
26 xlabel('Time');
27
28 box(axes1,'on');
29 % Set the remaining axes properties
30 set(axes1,'FontName','Times','FontSize',20);
31 % Create legend
32 legend1 = legend(axes1,'show');
33 set(legend1,'Location','best');
```

Figure 3.6 function code

```
1 function yprime=sirodes10(t,y)
2 % y1 is S and y2 is I while yprime(1) is S' and yprime(2) is I'
3 global n dx vx n1 w1 beta gamma
```

```
 4 yprime=zeros(n+1,1);
 5 MI=@(x,t)(1+exp(n1*(x-w1*t))).^(-2);
 6 MV=@(x,t)(1-MI(x,t));
 7 P=@(x,t)(1./(1+exp(50*MV(x,t)-40*MI(x,t))));
 8 for i=1:n
 9 yprime(i)=-beta*y(n+1)*(1-P(vx(i),t)).*y(i)-P(vx(i),t)*y(i);
10 end
11 yprime(n+1)=beta*y(n+1)*dx*trapz((1-P(vx,t)).*y(1:n)')-gamma*y(n
       +1);
12 end
```

Figure 3.6 plot code

```
 1 clear all
 2 clc
 3 global n dx vx n1 n2 w1 w2 beta gamma
 4
 5 %time and space discretization
 6 n=100;
 7 m=50;
 8 x0=0;
 9 xn=10;
10 t0=0;
11 tm=100;
12 vx=linspace(0,20,n); h=vx(2)-vx(1);
13 vt=linspace(t0,tm,m); k=vt(2)-vt(1);
14 x1=zeros(n+1,1);
15 t1=zeros(m+1,1);
16 dx =(xn-x0)/n;
```

```
17
18 %parameters for MI and MV
19
20 B1=.5;
21 c1=1;
22 n1= -sqrt((1-B1)/6);
23 w1= 5*sqrt((1-B1)/6)-c1;
24
25 %parameters for SIR
26 gamma = 1/14;
27 beta = 20*gamma;
28
29 initial=[ones(n,1)*.99/n;.01];
30
31 [T,Y]=ode45(@sirodes10,vt,initial,[]);
32
33 figure1 = figure;
34
35 % Create axes
36 axes1 = axes('Parent',figure1);
37 hold(axes1,'on');
38
39 % Create multiple lines using matrix input to plot
40 plot1 = plot(Y(:,end),'LineWidth',2,'Parent',axes1);
41 set(plot1,'DisplayName','Infected','Color',[1 0 0]);
42
43 % Create ylabel
```

```
44 ylabel('Proportion');
45
46 % Create xlabel
47 xlabel('Time');
48
49 box(axes1,'on');
50 % Set the remaining axes properties
51 set(axes1,'FontName','Times','FontSize',20);
52 S=zeros(0,1);
53 for i=1:m
54 S(i)=trapz(Y(i,1:n));
55 end
56 hold on
57 plot2=plot(S,'LineWidth',2);
58 set(plot2,'DisplayName','Susceptible','Color',[0 0 0]);
59 legend
```

Figure 3.8 Function code

```
1 function uprime=sirodes13(t,u)
2 % u(1:n-1) is MI, u(n:(2n-2)) is MV,
3 %u(2n-1:end-2) is S and u(end-1) is I
4 %u(end) is V
5
6 global n h beta gamma Cv Ci mi mv D1 D2 gamma1 gamma2 B1 B2 c1 c2
7
8 uprime=zeros(3*n-1,1);
9
10 ul=mi*u(end-1);
```

```
11 ur=0;
12 vl=mv*u(end);
13 vr=0;
14
15 E_p=diag(-1*ones(n-2,1),-1)+diag(1*ones(n-2,1),1);
16 E0=blkdiag(D1*eye(n-1),D2*eye(n-1));
17 E1=(((1/h^2)).*(diag(-2.*ones(n-1,1))+diag(ones(n-2,1),1)+diag(
       ones(n-2,1),-1)));
18 E1=blkdiag(E1,E1);
19 E2=((1/(h^2)).*[ul;zeros(n-3,1);ur;vl;zeros(n-3,1);vr]);
20 E3=[eye(n-1) B1*eye(n-1);B2*eye(n-1) eye(n-1)];
21 E4=blkdiag(gamma1*eye(n-1),gamma2*eye(n-1));
22 E5=(1/(2*h))*blkdiag(c1*E_p,c2*E_p);
23 E6=((1/(2*h)).*[-1*c1*ul;zeros(n-3,1);c1*ur;-1*c2*vl;zeros(n-3,1)
       ;c2*vr]);
24
25 F=@(t,u)(E0*(E1*u(1:(2*n-2))+E2)+E4*u(1:(2*n-2)).*(1-E3*u(1:(2*n
       -2)))+E5*u(1:(2*n-2))+E6); %this u has both systems joint
       together by the E's above.
26
27 P=(1./(1+exp(Cv*u(n:(2*n-2))-Ci*u(1:(n-1)))));
28
29 uprime1=F(t,u);
30 temp1=-beta*u(end-1)*(1-P).*u((2*n-1):(end-2));
31 temp2=P.*u((2*n-1):(end-2));
32 uprime2=temp1-temp2;
33 uprime3=beta*u(end-1)*h*trapz((1-P).*u((2*n-1):(end-2)))
```

```
34              -gamma*u(end-1);
35 uprime4=trapz(P.*u((2*n-1):(end-2)));
36 uprime=[uprime1;uprime2;uprime3;uprime4];
37 end
```

Figure 3.8 plot code

```
 1 clear all
 2 clc
 3
 4 global n h beta gamma Cv Ci mi mv D1 D2 gamma1 gamma2 B1 B2 c1 c2
 5
 6 %space and time discretization
 7 n=100;
 8 T=50;
 9 m=50;
10 a=0;
11 b=20;
12 vx=linspace(a,b,n+1); h=vx(2)-vx(1);
13 vt=linspace(0,T,m);    k=vt(2)-vt(1);
14 dx = (b-a)/(n+1);
15
16 beta=20/14;
17 gamma=1/14;
18 Cv=50;
19 Ci=40;
20 mi=1;
21 mv=1;
22 D1=1;
```

```
23 D2=1;
24 gamma1=1;
25 gamma2=1;
26 B1=.5;
27 B2=2-B1;
28 c1=2;
29 c2=-2;
30
31 initial=[mi*.01;zeros((n-2),1); zeros((n-1),1)
32             ; ones(n-1,1)*.99/(n-1);.01;0];
33
34 [vT,solut]=ode45(@sirodes13,vt,initial);
35 u=solut;
36 ul=mi*u(:,(end-2));
37 ur=zeros(length(vT),1);
38 vl=mv*u(:,end);
39 vr=zeros(length(vT),1);
40 u_final=[ul, solut(:,1:(n-1)), ur]; %getting solution for u
41 v_final=[vl, solut(:,n:(2*n-2)), vr]; %getting solution for v
42 S=u(:,(2*n-1:end-2));
43 I=u(:,(end-1));
44 V=u(:,end);
45
46 %plots
47 figure(1)
48 surf(vx,vT,u_final)
49 xlabel('Space')
```

```
50 ylabel('Time')
51
52 figure(2)
53 surf(vx,vT,v_final)
54 xlabel('Space')
55 ylabel('Time')
56
57 figure(4)
58 surf(vx,vT,[zeros(m,1) S zeros(m,1)])
59 xlabel('Space')
60 ylabel('Time')
61
62 figure(5)
63 plot(vT,I,'Linewidth',2)
64 hold on
65 plot(vT,h*trapz(S,2),'Linewidth',2)
66 xlabel('Time')
67 ylabel('Proportion of individuals')
68 legend('Infected','Susceptible')
69
70 figure(6)
71 plot(vT,V,'Linewidth',2)
72 xlabel('Time')
73 ylabel('Proportion of vaccinated')
```

Figures 2.4-2.10: code

```
1 clear all
2 clc
```

```
3
4 %constants parameters
5 D1=3;
6 D2=3;
7 gamma1=1;
8 gamma2=1;
9 B1=.5;
10 B2=2-B1;
11 c1=sqrt(3);
12 %c2=2*sqrt(3);
13 sigma1=1;
14 %sigma2=1;
15 n1= -sqrt((1-B1)/6);
16 %n2= sqrt((B2-1)/6);
17 w1= -5*sqrt((1-B1)/6)-1;
18 %w2= -5*sqrt((B2-1)/6)-1;
19 m0 = 1;
20
21 %space and time discretization
22 n=50;
23 t0=0;
24 T=1;
25 m=50;
26 a=50;
27 b=-50;
28 vx=linspace(a,b,n+1); h=vx(2)-vx(1);
29 vt=linspace(t0,T,m);    k=vt(2)-vt(1);
```

```
30
31 f=@(t,x)(c1);
32 g=@(t,x)(sigma1);
33 N=1;
34 dw=sqrt(k).*randn(N,m-1);
35 BM=[zeros(N,1) cumsum(dw,2)];
36 dz=.5*k*(dw+sqrt(k/3).*randn(N,m-1));% Another indepedent BM
37 vrk=vx';%z
38 for j=2:m
39     Lp=vrk(:,j-1)+k*f(vt(j-1),vrk(:,j-1))+sqrt(k)*g(vt(j-1),vrk
    (:,j-1));
40     Lm=vrk(:,j-1)+k*f(vt(j-1),vrk(:,j-1))-sqrt(k)*g(vt(j-1),vrk
    (:,j-1));
41     Hp=Lp+sqrt(k)*g(vt(j-1),Lp);
42     Hm=Lp-sqrt(k)*g(vt(j-1),Lp);
43       vrk(:,j)=vrk(:,j-1)+k*f(vt(j-1),vrk(:,j-1))+g(vt(j-1),vrk(:,
    j-1)).*dw(:,j-1)...
44           +(1/(2*sqrt(k)))*(f(vt(j-1),Lp)-f(vt(j-1),Lm)).*dz(:,j-1)
    ...
45           +.25*k*(f(vt(j-1),Lp)-2*f(vt(j-1),vrk(:,j-1))+f(vt(j-1),
    Lm))...
46           +(1/(4*sqrt(k)))*(g(vt(j-1),Lp)-g(vt(j-1),Lm)).*(dw(:,j
    -1).^2-k)...
47           +(1/(2*k))*(g(vt(j-1),Lp)-2*g(vt(j-1),vrk(:,j-1))+g(vt(j
    -1),Lm)).*(k*dw(:,j-1)-dz(:,j-1))...
48           +(1/(4*k))*(g(vt(j-1),Hp)-g(vt(j-1),Hm)-(g(vt(j-1),Lp)-g(
    vt(j-1),Lm))).*((1/3)*(dw(:,j-1).^2)-k).*dw(:,j-1)...
```

```
49              +(k/2)*(f(vt(j),vrk(:,j-1))-f(vt(j-1),vrk(:,j-1))) ...
50              +(1/k)*(g(vt(j),vrk(:,j-1))-g(vt(j-1),vrk(:,j-1))).*(k*dw
        (:,j-1)-dz(:,j-1));
51 end
52 figure(1)
53 plot(vt,vrk,'b-o','MarkerSize',3,'MarkerFaceColor',[1,0,0])
54
55 L=@(t) (interp1(vt,vrk(1,:),t)-(t~=vt(1))*vrk(1,max([1 find(vt<t
        ,1,'last')])));
56
57 %A=(1/h^2)*(diag(-2*ones(n-1,1),0)+diag(ones(n-2,1),-1)+diag(ones
        (n-2,1),1));
58 %c1=1/h^2;
59 %c2=1/(4*h);
60 uu1=@(t,x)(1+m0*exp(n1*(((x/sqrt(3))-w1*t)))).^(-2);   %function 1
61 uu2=@(t,x)(1-(uu1(t,x)));   %function 2
62 ul=@(t)(uu1(t,interp1(vt,vrk(1,:),t)));      %u lower bc
63 ur=@(t)(uu1(t,interp1(vt,vrk(end,:),t)));   %u upper bc
64 vl=@(t)(uu2(t,interp1(vt,vrk(1,:),t)));      %v lower bc
65 vr=@(t)(uu2(t,interp1(vt,vrk(end,:),t)));   %v upper bc
66
67 vvx=vx(2:end-1);
68 u_initial=uu1(0,vvx'); %n-1 entries
69 v_initial=uu2(0,vvx'); %n-1 entries
70 initial=[u_initial; v_initial];
71
72
```

```
73 E0=@(t)[(D1 - (1/2)*sigma1 + L(t)./(2*k))*eye(n-1) zeros(n-1,n-1)
       ;zeros(n-1,n-1) (D2-(1/2)*sigma1+L(t)./(2*k))*eye(n-1)];
74 E1=((((1/h^2))).*(diag(-2.*ones(n-1,1))+diag(ones(n-2,1),1)+diag(
       ones(n-2,1),-1)));
75 E1=[E1 zeros(size(E1));zeros(size(E1)) E1];
76 E2=@(t)(((1/(h^2))).*[ul(t);zeros(n-3,1);ur(t);vl(t);zeros(n-3,1)
       ;vr(t)]);
77 E3=[eye(n-1) B1*eye(n-1);B2*eye(n-1) eye(n-1)];
78 E4=[gamma1*eye(n-1) zeros(n-1,n-1);zeros(n-1,n-1) gamma2*eye(n-1)
       ];
79
80 F=@(t,w)(E0(t)*(E1*w+E2(t))+E4*w.*(1-E3*w)+((1/(2*h))).*(1/k).*L(t
       ).*([w(2:n-1);ur(t);w(n+1:end);vr(t)]-[ul(t);w(1:(n-2));vl(t);
       w(n:(end-1))])));  %this u has both systems joint together by
       the E's above.
81
82 %ode45 solving the system
83 [vT,solut]=ode45(F, vt, initial);
84 m_final=[ul(vT), solut(:,1:(n-1)), ur(vT)]; %getting solution for
        u
85 n_final=[vl(vT), solut(:,n:end), vr(vT)]; %getting solution for v
86 Total=m_final+n_final;
87 %%
88 %plots
89 figure(2)
90 surf(vx,vT,m_final)
91 xlabel('Space')
```

```
92 ylabel('Time')
93
94 figure(3)
95 surf(vx,vT,n_final)
96 xlabel('Space')
97 ylabel('Time')
98
99 %% Exact
100 m_exactsto=@(t,x,BM) (1+m0*exp(n1.*((x+c1.*t+sigma1.*BM)-w1*t)))
        .^(-2);
101 for i=1:m
102 for j=1:n+1
103 xexact_1(i,j)=m_exactsto(vt(i),vx(j),BM(i));
104 end
105 end
106 figure(4)
107 surf(vx,vt,xexact_1)
108
109 n_exactsto=@(t,x,BM) (1-m_exactsto(t,x,BM));
110 for i=1:m
111 for j=1:n+1
112 xexact_2(i,j)=n_exactsto(vt(i),vx(j),BM(i));
113 end
114 end
115 figure(5)
116 surf(vx,vt,xexact_2)
117
```

```
118 %% Errors
119 err1 = abs(m_final - xexact_1);
120 figure(6)
121 surf(vx,vt,err1);
122 zlabel('Error')
123
124 err2 = abs(n_final - xexact_2);
125 figure(7)
126 surf(vx,vt,err2);
127 zlabel('Error')
```

BIOGRAPHICAL SKETCH

Juan Jesus Huerta, born and raised in Weslaco, Texas. Graduated from Weslaco High school at age 18 in 2015. He then graduated as first class in 2018 in the University of Texas Rio Grande Valley with a bachelor's in science in Mathematics. In spring 2019, he then started to pursue a Master's in science in applied mathematics. He received a Master of Science in Mathematics from The University of Texas Rio Grande Valley in December 2020.

Email: juan.huerta01@utrgv.edu