On the Emergence of Spatial Patterns in an Epidemic Model by a Cross-Diffusion Term

With an approach of finite element analysis

ME20B087 Janmenjaya Panda
ME20B122 Nishant Sahoo

Instructor: Prof. Sundararajan Natarajan

ME5204 Finite Element Analysis
Department of Mechanical Engineering
Indian Institute of Technology Madras

Contents

[1] **S**usceptible - **I**nfectious - **R**emoved [2] Euler! My man. [3] **C**ourant-**F**riedrichs-**L**ewy

[4] The phase portrait of the system without diffusion terms

ABSTRACT

We present a numerical analysis of a two-dimensional epidemic model featuring a nonlinear cross-diffusion term, capturing the spatial dynamics of susceptible and infected populations as discussed in [1], [2]. The model incorporates non-linear self-diffusion and cross-diffusion terms[5] to represent psychological and sociological responses of the population to the disease. Utilizing a finite element method, we discretize the reaction-diffusion system and simulate both forward and backward Eulerian schemes. For the nonlinear systems, we simulate it through both Picard's iteration and the Newton-Raphson method. A comprehensive analysis of the model was conducted to study the influence of different model parameters such as birth rate, carrying capacity, rate of disease transmission, recovery rate etc. Our observations also highlight that nonlinear cross-diffusion results in the emergence of complex spatial structures, such as stripes, spots, and holes, influenced by initial conditions. Additionally, we analyze the stability and convergence of the numerical scheme both in time and in space, providing relevant error estimates.

1 INTRODUCTION

1.1 Modelling an epidemic: The SIR[6] model

Epidemics, the widespread occurrence of infectious diseases within a population, have a profound impact on public health and societal well-being. Understanding how these diseases spread is crucial in developing effective prevention and control strategies [3][5]. Modelling epidemics provides a quantitative framework to comprehend the dynamics of disease transmission. Among the various models, the SIR (Susceptible-Infected-Recovered) model stands as a fundamental category for describing epidemic evolution through ordinary differential equations. This model categorizes a population into three groups—susceptible (S), infected (I), and recovered (R)—and tracks the transitions between these subgroups. The pioneering work of Kermack and McKendrick introduced an early prototype of the SIR model, employing a system of ordinary differential equations to elucidate the population's progression throughout an epidemic [4], which is as follows:

$$\frac{dS}{dt} = -\alpha SI, \qquad \frac{dI}{dt} = \alpha SI - \beta I, \qquad \frac{dR}{dt} = \beta I$$

where $\alpha > 0$ is the infection rate and $\beta > 0$ is the recovery rate. Such models serve as invaluable tools in predicting disease spread, evaluating interventions, and guiding public health policies.

1.2 SIR in higher dimensions

Expanding the SIR model into higher dimensions, especially when incorporating cross-diffusion systems, fundamentally alters the understanding of disease propagation compared to its traditional 1D counterpart.

In one dimension, the SIR model assumes a linear flow of infection within a population. However, in higher dimensions with cross-diffusion, the model becomes more intricate. Cross-diffusion introduces spatial complexity, acknowledging that interactions between susceptible and infected individuals occur not just along a single axis but across multiple dimensions.

[5] There are two models. In the first one, the diffusion terms are independent of primary variables (essentially linear) and in model 2, the diffusion terms are non-linear. [6] **S**usceptible - **I**nfectious - **R**emoved

The cross-diffusion mechanism enables the infection to spread not only through direct contact but also via spatial gradients, considering how neighbouring areas influence each other. This spatial interdependence leads to non-local interactions, where the infection in one region affects the dynamics of adjacent regions.

Furthermore, in higher dimensions, cross-diffusion systems allow for a more comprehensive depiction of population heterogeneity, spatial connectivity, and intricate network structures. This accounts for diverse interactions, movement patterns, and contact rates among various subgroups or geographical regions within the population.

The inclusion of cross-diffusion in higher-dimensional SIR models provides a more realistic representation of disease transmission in spatially distributed populations. However, this complexity demands more sophisticated mathematical analyses and computational resources for accurate simulations and predictions. Overall, the higher-dimensional SIR model with cross-diffusion enriches our understanding of disease spread by considering spatial intricacies and non-local interactions within populations.

1.3 Significance of the cross and self-diffusion terms

Cross and self-diffusion terms within 2D epidemic models play a significant role in inducing pattern formations by considering spatial interactions and movements between distinct populations.

1. **Spatial Heterogeneity**: The self-diffusion term characterizes movement within each population group, leading to uneven distributions of susceptible and infected individuals across regions.

2. **Non-Uniform Spread**: Differential diffusion rates and interactions create spatial gradients, resulting in varying concentrations of infected individuals across different areas, thus generating patterns like clusters or waves of infections.

3. **Emergence of Structures**: Variations in diffusion rates and cross-interactions give rise to spatial arrangements within populations, leading to the emergence of structures or spatial patterns, reflecting localized interactions and varying transmission rates.

4. **Complex Spatial Dynamics**: Interplay between self and cross-diffusion terms results in complex spatial dynamics, where local interactions influence infection spread differently in various regions. This complexity leads to intricate patterns, such as waves, clusters, or spatial heterogeneity in disease prevalence.

In summary, the cross and self-diffusion terms induce pattern formations by accounting for spatial interactions and movement dynamics within and between population groups, resulting in non-uniform spread and the emergence of spatial structures in infectious disease models.

1.4 The Turing pattern formation

Turing pattern formation, a phenomenon named after mathematician Alan Turing, refers to the spontaneous generation of complex, repetitive patterns in systems where uniformity is expected. This process arises due to the interaction between two diffusing substances, often involving an activator and an inhibitor, under specific conditions. The diffusion terms, particularly cross-diffusion, play a pivotal role in Turing pattern formation by creating spatial instabilities that lead to the emergence of patterns. When the activator and inhibitor diffuse at different rates, it causes spatial heterogeneity, disrupting the uniformity and allowing for the formation of intricate patterns, such as spots, stripes, or waves. These diffusion-driven

instabilities enable the system to move from a homogeneous state to one exhibiting diverse spatial structures, showcasing the significance of diffusion processes in generating complex patterns observed in various natural and biological systems.

In the context of 2D epidemic modelling, Turing pattern formation, driven by diffusion terms, presents a mechanism for the emergence of intricate spatial patterns. These patterns arise from the interplay of diffusing substances, analogous to activator and inhibitor agents, representing different aspects of the epidemic dynamics. The cross-diffusion effect, where distinct populations interact and move across space at varying rates, introduces spatial instabilities within the model. These instabilities disrupt uniformity and pave the way for the spontaneous generation of complex patterns, such as localized clusters, waves, or spatial heterogeneity in disease spread. In essence, the diffusion-driven spatial instabilities, akin to Turing patterns, illustrate how variations in diffusion rates among different populations contribute to the formation of diverse and structured spatial patterns in 2D epidemic models.

In this project, we replicate some of the results on these pattern formation discussed in [2].

2 Differential equations & Boundary Condition

2.1 The strong form

The strong form of the spatial spread of an epidemic with the cross-diffusion, modelled by a reaction-diffusion system that operates in two dimensions, viz. $\Omega_T \triangleq \Omega \times (0, T)$ can be represented as follows:

$$\frac{\partial u}{\partial t} = f(u, v) + \nabla \cdot (a(u)\nabla u) + \nabla \cdot (c(u, v)\nabla v) \tag{1}$$

$$\frac{\partial v}{\partial t} = g(u, v) + \nabla \cdot (b(v)\nabla v) \tag{2}$$

where u and v denote the populations of susceptible and infected persons. The system ensures the segregation of phases as the susceptible species avoid contact with the infected ones due to a cross-diffusion term represented by $\nabla \cdot (c(u, v)\nabla v)$. This term causes the flow to move counter to the ∇v gradient. As the number of infected individuals rises, the susceptible agents instinctively move away from the direction of this increasing gradient.

Here $a(u)$, $b(u)$ and $c(u, v)$ are the diffusion coefficients. The reaction terms are defined followingly.

$$f(u, v) = ru(1 - \frac{u}{K}) - \beta\frac{uv}{u + v} \tag{3}$$

$$g(u, v) = \beta\frac{uv}{u + v} - kv \tag{4}$$

where K is the carrying capacity[7] of the susceptible species, r is the intrinsic birth rate, β is the rate of disease transmission and k stands for the recovery rate of the infected species.

2.1.1 Model 1

For Model 1, we assume linear diffusion functions, where the diffusion coefficients are chosen to be the constants

$$a(u) = a_0, \quad b(v) = b_0, \quad c(u, v) = c_0, \tag{5}$$

2.1.2 Model 2

In Model 2 we propose nonlinear model variants of the parametric functions. The self-diffusion terms are chosen to have the form

$$a(u) = a_0 u^m, \quad b(v) = b_0 v^m \quad c(u, v) = \max(0, c_0\, u\, v(c_1 - u - v)) \tag{6}$$

where, $c_0, c_1 > 0$,

2.1.2.1 Why Non-linear?

A nonlinear cross-diffusion function embodies a situation-specific inclination of the susceptible population to evade contact with the infected. This behaviour, dependent on the situation, mirrors an average psychological response. One method to model this disposition is as follows: regarding the susceptible population, the urge to avoid arises whenever a noticeable fraction of the population is

[7] the maximum population size that an environment can sustainably support over a specific period

infected. In scenarios with a small population, awareness might not be fully developed or temporarily inactive, as there's less urgency for self-protection. Conversely, in larger populations, selective detection becomes impractical, as the possibility of avoiding infection within the crowd diminishes or becomes non-existent. Population size significantly influences the conscious inclination towards avoidance. Thus, the cross-diffusion coefficient is tailored to be negligible in both small and large populations, guided by the above expression.

2.2 The boundary condition

As we have learned in the class, the boundary conditions can be distinguished among three mutually disjoint and exhaustive sets of boundary conditions, viz.

2.2.1 The Dirichlet boundary condition

We do not have any Dirichlet boundary condition due to the involvement of both mass and stiffness matrix on the LHS of the equation. This somewhat indicates that the model is self-sustained once it is initialized with the initial values and we need not to impose any specific boundary values.

2.2.2 The Neumann boundary condition

Since no external input is imposed, therefore on the physical domain boundary $\partial\Omega$, it holds the Neumann boundary condition given as follows

$$(a(u)\nabla u + c(u,v)\nabla v)\cdot\hat{n} = 0 \tag{7}$$
$$(b(v)\nabla u)\cdot\hat{n} = 0 \tag{8}$$

2.2.3 The Robin Boundary condition

We don't have any Robin boundary conditions.

2.3 The initial condition

The state of the system at $t = 0$ is given as follows:

$$u(x,y,0) = u^* + \rho_u\delta_u \tag{9}$$
$$v(x,y,0) = v^* + \rho_v\delta_v \tag{10}$$

where (u^*, v^*) represents the equilibrium point, viz. $(u(x,y,\infty), v(x,y,\infty))$. $\delta_w \in [0,1]$ is a uniformly distributed random variable for $w \in \{u,v\}$ and ρ_w is a scaling factor.

2.4 The weak form

Let's start with the weak-form formulation of the equation 1.

The strong form is as follows.

$$\frac{\partial u}{\partial t} = f(u,v) + \nabla\cdot(a(u)\nabla u) + \nabla\cdot(c(u,v)\nabla v)$$
$$\frac{\partial u}{\partial t} - f(u,v) - \nabla\cdot(a(u)\nabla u) - \nabla\cdot(c(u,v)\nabla v) = 0$$

Introduce variable w to the space of which we want to project it to minimize the L^2 error.

$$\int_\Omega w\left[\frac{\partial u}{\partial t} - f(u,v) - \nabla\cdot(a(u)\nabla u) - \nabla\cdot(c(u,v)\nabla v)\right] d\Omega = 0$$

$$\int_\Omega w\frac{\partial u}{\partial t}\, d\Omega - \int_\Omega wf\, d\Omega - \int_\Omega w\,\nabla\cdot(a\nabla u)\, d\Omega - \int_\Omega w\,\nabla\cdot(c\nabla v)\, d\Omega = 0$$

Now applying Green's first identity on the penultimate and the ultimate terms on the LHS, we obtain:

$$\int_\Omega w\frac{\partial u}{\partial t}\, d\Omega - \int_\Omega wf\, d\Omega - \left(\int_{\partial\Omega} w\; a\nabla u\cdot\hat{n}\, d\Omega - \int_\Omega \nabla w\cdot a\nabla u\, d\Omega\right) - \left(\int_{\partial\Omega} w\; c\nabla v\cdot\hat{n}\, d\Omega - \int_\Omega \nabla w\cdot c\nabla v\, d\Omega\right) = 0$$

Let's group the linear, bilinear[8] and boundary terms together:

$$\underbrace{\int_\Omega w\frac{\partial u}{\partial t}\, d\Omega}_{\text{Mass term}} - \underbrace{\int_\Omega wf\, d\Omega}_{\text{Linear term}} - \underbrace{\int_{\partial\Omega} w\left(a\nabla u + c\nabla v\right)\cdot\hat{n}\, d\Omega}_{\text{Neumann boundary term}} + \underbrace{\int_\Omega \nabla w\cdot\left(a\nabla u + c\nabla v\right) d\Omega}_{\text{Stiffness term}} = 0$$

The Neumann boundary condition from equation 7 yields the following:

$$\int_\Omega w\frac{\partial u}{\partial t}\, d\Omega + \int_\Omega \nabla w\cdot\left(a\nabla u + c\nabla v\right) d\Omega = \int_\Omega wf\, d\Omega \tag{11}$$

which is the required weak form representation for equation 1.

Let's formulate the weak form representation of equation 2

The strong form is given as follows:

$$\frac{\partial v}{\partial t} = g(u,v) + \nabla\cdot(b(v)\nabla v)$$

$$\frac{\partial v}{\partial t} - g(u,v) - \nabla\cdot(b(v)\nabla v) = 0$$

Introduce variable w to the space of which we want to project it to minimize the L^2 error.

$$\int_\Omega w\left[\frac{\partial v}{\partial t} - g(u,v) - \nabla\cdot(b(v)\nabla v)\right] d\Omega = 0$$

$$\int_\Omega w\frac{\partial v}{\partial t}\, d\Omega - \int_\Omega wg\, d\Omega - \int_\Omega w\,\nabla\cdot(b\nabla v)\, d\Omega = 0$$

Applying Green's first identity similarly as earlier, we obtain:

$$\underbrace{\int_\Omega w\frac{\partial v}{\partial t}\, d\Omega}_{\text{Mass term}} - \underbrace{\int_\Omega wg\, d\Omega}_{\text{Linear term}} - \left(\underbrace{\int_{\partial\Omega} w\, b\nabla v\cdot\hat{n}\, d\Omega}_{\text{Neumann boundary term}} - \underbrace{\int_\Omega \nabla w\cdot b\nabla v\, d\Omega}_{\text{Stiffness term}}\right) = 0$$

[8] the mass and stiffness terms

Again, apply the Neumann boundary condition as per equation 8 and obtain the final weak form representation of equation 2 as follows:

$$\int_\Omega w\frac{\partial v}{\partial t}\,d\Omega + \int_\Omega \nabla w \cdot b\nabla v\,d\Omega = \int_\Omega wg\,d\Omega \tag{12}$$

3 Mesh Choice

In finite element method (FEM), meshing is the process of dividing a geometric domain into smaller, simpler shapes called elements. The choice between triangular and quadrilateral elements depends on various factors, and each type has its advantages.

3.1 Triangular Elements

Triangular elements have various advantages like:

1. **Simplicity:** Triangular elements are simpler geometrically and computationally. They have fewer nodes, which makes the assembly and solution processes computationally less demanding.
2. **Mesh Flexibility:** Triangular elements can better adapt to complex geometries and irregular boundaries. They are more versatile in handling geometries with acute angles and curved boundaries.
3. **Conformity:** Triangular elements conform naturally to curved surfaces, which is advantageous in simulations involving curved geometries or irregular shapes.
4. **Reduced Numerical Smoothing:** Triangular elements can mitigate numerical smoothing effects, which may occur with quadrilateral elements, especially when dealing with highly distorted meshes.

3.2 Quadrilateral Elements

Advantages of quadrilateral elements are:

1. **Higher Interpolation Accuracy:** Quadrilateral elements often require fewer nodes for accurate interpolation, making them computationally more efficient in some cases.
2. **Aspect Ratio Robustness:** Quadrilateral elements are less sensitive to aspect ratio variations compared to triangular elements. They can better handle elements with disparate side lengths.

In this project, we have chosen triangular elements instead of quadrilateral elements, especially due to their simplicity and better computational efficiency over quadrilateral elements.

4 SHAPE FUNCTIONS IN THE PARAMETRIC SPACE

4.1 Shape functions and their derivatives

For discretization, we may assume the primary variable and the test variables can be expressed using the discrete nodal values with the linear Lagrange interpolant functions as follows:

For primary functions:

$$\begin{aligned} u(x,y) \approx u^h(x,y) &= \sum_I \phi_I u_I(x,y) \\ v(x,y) \approx v^h(x,y) &= \sum_I \phi_I v_I(x,y) \end{aligned} \tag{13}$$

We introduce a test function w:

$$w = \sum_I \phi_I w(x,y)$$

where ϕ_Is are linear Lagrange interpolant functions.

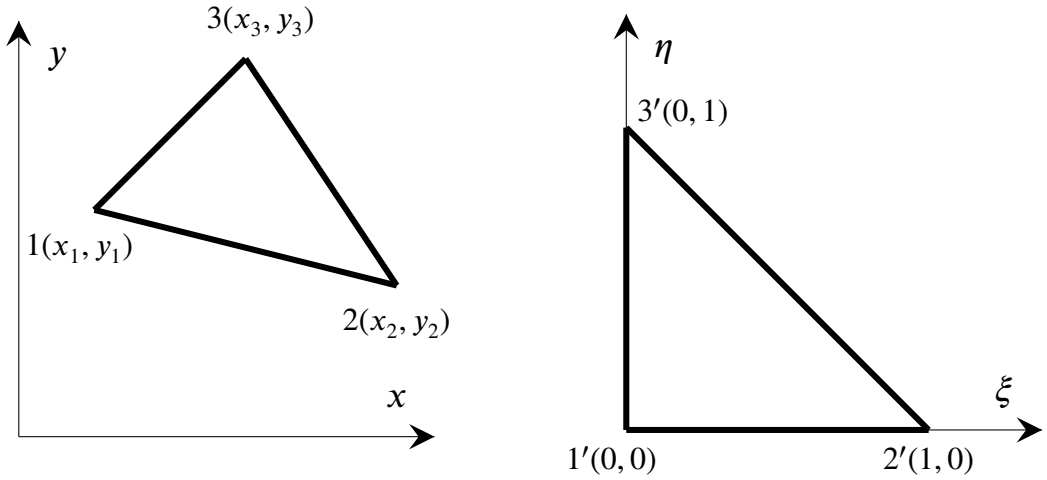
As far as the integrals over each are concerned, to make life easier, we converted each of the triangulated mesh elements of the domain in physical space to the standard triangle in the parametric space, which takes us from the (x, y) physical world to the (ξ, η) parametric world. Then we do the numerical integration of the required expression on the parametric space using the Gauss quadrature and the Jacobian ensures the equality of the integral values.
The linear shape functions and their derivatives are as follows:

$$\begin{array}{lll} \phi_1 = 1 - \xi - \eta & \dfrac{\partial \phi_1}{\partial \xi} = -1 & \dfrac{\partial \phi_1}{\partial \eta} = -1 \\ \phi_2 = \xi & \dfrac{\partial \phi_2}{\partial \xi} = 1 & \dfrac{\partial \phi_2}{\partial \eta} = 0 \\ \phi_3 = \eta & \dfrac{\partial \phi_3}{\partial \xi} = 0 & \dfrac{\partial \phi_3}{\partial \eta} = 1 \end{array}$$

4.2 The jacobian

$$\begin{pmatrix} \dfrac{\partial}{\partial \xi} \\ \dfrac{\partial}{\partial \eta} \end{pmatrix} = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix} \begin{pmatrix} \dfrac{\partial}{\partial x} \\ \dfrac{\partial}{\partial y} \end{pmatrix} \tag{14}$$

$$\text{Jacobian, } J = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix} \tag{15}$$



(a) Physical Space (b) Parametric Space

Figure 1: From the physical space to the parametric space

5 NUMERICAL INTEGRATION SCHEME

5.1 From integrals to summations

5.1.1 Discretizing bilinear terms:

$$\int_\Omega \nabla w \cdot a\nabla u \, d\Omega = \int_x \int_y \nabla w \cdot a\nabla u \, dy \, dx = \mathbf{w}^T \left(\int_\eta \int_\xi \nabla \phi_I^T \, a\nabla \phi_J \, |J| \, d\xi \, d\eta \right) \mathbf{u}$$

$$\int_\Omega \nabla w \cdot b\nabla v \, d\Omega = \int_x \int_y \nabla w \cdot b\nabla v \, dy \, dx = \mathbf{w}^T \left(\int_\eta \int_\xi \nabla \phi_I^T \, b\nabla \phi_J \, |J| \, d\xi \, d\eta \right) \mathbf{v}$$

$$\int_\Omega \nabla w \cdot c\nabla v \, d\Omega = \int_x \int_y \nabla w \cdot c\nabla v \, dy \, dx = \mathbf{w}^T \left(\int_\eta \int_\xi \nabla \phi_I^T \, c\nabla \phi_J \, |J| \, d\xi \, d\eta \right) \mathbf{v}$$

5.1.2 Discretizing linear terms:

$$\int_\Omega wf \, d\Omega = \mathbf{w}^T \int_\eta \int_\xi \phi_I f(u, v) \, |J| \, d\xi \, d\eta$$

$$\int_\Omega wg \, d\Omega = \mathbf{w}^T \int_\eta \int_\xi \phi_I g(u, v) \, |J| \, d\xi \, d\eta$$

5.1.3 Discretizing differential operator:

We have

$$\frac{\partial u}{\partial t} \approx \frac{\Delta u}{\Delta t} = \frac{u_{n+1} - u_n}{\Delta t} \quad \text{and} \quad \frac{\partial v}{\partial t} \approx \frac{\Delta v}{\Delta t} = \frac{v_{n+1} - v_n}{\Delta t}$$

$$\int_\Omega w\left(\frac{\partial u}{\partial t}\right)\,d\Omega \approx \int_x\int_y w\left(\frac{u_{n+1}-u_n}{\Delta t}\right)\,dy\,dx = \mathbf{w}^T\left(\int_\eta\int_\xi \phi_I^T\,\phi_J\,|J|\,d\xi\,d\eta\right)\left(\frac{\mathbf{u_{n+1}}-\mathbf{u_n}}{\Delta t}\right)$$
$$\int_\Omega w\left(\frac{\partial v}{\partial t}\right)\,d\Omega \approx \int_x\int_y w\left(\frac{v_{n+1}-v_n}{\Delta t}\right)\,dy\,dx = \mathbf{w}^T\left(\int_\eta\int_\xi \phi_I^T\,\phi_J\,|J|\,d\xi\,d\eta\right)\left(\frac{\mathbf{v_{n+1}}-\mathbf{v_n}}{\Delta t}\right)$$

5.1.4 Matrix representation

1. Stiffness Matrices:

$$\begin{aligned}\mathbf{K_1} &= \int_\eta\int_\xi \nabla\phi_I^T\,a\nabla\phi_J\,|J|\,d\xi\,d\eta\\ \mathbf{K_2} &= \int_\eta\int_\xi \nabla\phi_I^T\,b\nabla\phi_J\,|J|\,d\xi\,d\eta\\ \mathbf{K_3} &= \int_\eta\int_\xi \nabla\phi_I^T\,c\nabla\phi_J\,|J|\,d\xi\,d\eta\end{aligned} \tag{16}$$

2. Mass Matrix:

$$\mathbf{M} = \int_\eta\int_\xi \phi_I^T\,\phi_J\,|J|\,d\xi\,d\eta \tag{17}$$

3. Force vectors:

$$\begin{aligned}\mathbf{F} &= \int_\eta\int_\xi \phi_I f(u,v)\,|J|\,d\xi\,d\eta\\ \mathbf{G} &= \int_\eta\int_\xi \phi_I g(u,v)\,|J|\,d\xi\,d\eta\end{aligned} \tag{18}$$

4. Susceptibility equation:

$$\mathbf{M}\left(\frac{\Delta\mathbf{u}}{\Delta t}\right) + \mathbf{K_1 u} + \mathbf{K_3 v} = \mathbf{F} \tag{19}$$

5. Infectious equation:

$$\mathbf{M}\left(\frac{\Delta\mathbf{v}}{\Delta t}\right) + \mathbf{K_2 v} = \mathbf{G} \tag{20}$$

Note that, in Model 2, the stiffness matrices are also the functions of u and v as the diffusion coefficients are non-linear functions of u and v. On the other hand, in Model 1, the stiffness matrices are kind of constants just like the mass matrix. In the following section, we shall write $\mathbf{K}_n$ to denote the generic $\mathbf{K}$ of both Model 1 and 2. But it shall be understood that for Model !, once we fix, a_0, b_0, c_0, nodes and elements, the stiffness matrices are constants throughout the iterations.

5.2 Wherever you go, you will see his name[9]

Now we shall follow the Euler scheme to conquer the time dependency of the primary variable. Note that, here $\mathbf{u_n}$ denotes to the vector $\mathbf{u}$ corresponding to the nodal vector captured at the n^{th} time step denoted by t_n and similar notation is used for $\mathbf{v}$, $\mathbf{F}$ and $\mathbf{G}$ and all the stiffness matrices.

[9] Euler! My man.

5.2.1 The forward path

For the Forward Euler[10], we must consider the functional values of all the terms from the n^{th} iteration and this will give rise to the explicit formulation of u_{n+1} and v_{n+1} using u_n, v_n, F_n and G_n.

$$\mathbf{M}\left(\frac{\mathbf{u_{n+1}}-\mathbf{u_n}}{\Delta t}\right)+\mathbf{K_{1,n}u_n}+\mathbf{K_{3,n}v_n}=\mathbf{F_n}$$
$$\mathbf{M}\left(\frac{v_{n+1}-v_n}{\Delta t}\right)+\mathbf{K_{2,n}}v_n=\mathbf{G}_n$$

$$\begin{bmatrix}\mathbf{M} & \mathbf{0}\\ \mathbf{0} & \mathbf{M}\end{bmatrix}\begin{pmatrix}\mathbf{u_{n+1}}\\ \mathbf{v_{n+1}}\end{pmatrix}=\begin{bmatrix}\mathbf{M}-\Delta t\mathbf{K_{1,n}} & -\Delta t\mathbf{K_{3,n}}\\ \mathbf{0} & \mathbf{M}-\Delta t\mathbf{K_{2,n}}\end{bmatrix}\begin{pmatrix}\mathbf{u_n}\\ \mathbf{v_n}\end{pmatrix}+\Delta t\begin{pmatrix}\mathbf{F_n}\\ \mathbf{G_n}\end{pmatrix} \tag{21}$$

where $\mathbf{0}$ is the null matrix of order $N \times N$ where N is the number of nodes.

5.2.2 The backward path

For the Backward Euler[11], to calculate $\mathbf{u_{n+1}}$ and $\mathbf{v_{n+1}}$, we must substitute the functional value of all the terms as that of their $(n+1)^{th}$ iteration and this will give rise to an implicit formulation of $\mathbf{u_{n+1}}$ and $\mathbf{v_{n+1}}$ using $\mathbf{u_{n+1}}$, $\mathbf{v_{n+1}}$ $\mathbf{u_n}$, $\mathbf{v_n}$, $\mathbf{F_{n+1}}$ and $\mathbf{G_{n+1}}$, that shall be considering $\mathbf{u_{n+1}}$ and $\mathbf{v_{n+1}}$ as variables to obtain the required values at the $(n+1)^{th}$ iteration.

$$\mathbf{M}\left(\frac{\mathbf{u_{n+1}}-\mathbf{u_n}}{\Delta t}\right)+\mathbf{K_{1,n+1}u_{n+1}}+\mathbf{K_{3,n+1}v_{n+1}}=\mathbf{F_{n+1}}$$
$$\mathbf{M}\left(\frac{\mathbf{v_{n+1}}-\mathbf{v_n}}{\Delta t}\right)+\mathbf{K_{2,n+1}v_{n+1}}=\mathbf{G_{n+1}}$$

Combining the above equations in a matrix form produces the following equation.

$$\begin{bmatrix}\mathbf{M}+\Delta t\mathbf{K_{1,n+1}} & \Delta t\mathbf{K_{3,n+1}}\\ \mathbf{0} & \mathbf{M}+\Delta t\mathbf{K_{2,n+1}}\end{bmatrix}\begin{pmatrix}\mathbf{u_{n+1}}\\ \mathbf{v_{n+1}}\end{pmatrix}=\begin{bmatrix}\mathbf{M} & \mathbf{0}\\ \mathbf{0} & \mathbf{M}\end{bmatrix}\begin{pmatrix}\mathbf{u_n}\\ \mathbf{v_n}\end{pmatrix}+\Delta t\begin{pmatrix}\mathbf{F_{n+1}}\\ \mathbf{G_{n+1}}\end{pmatrix} \tag{22}$$

Again $\mathbf{0}$ is the null matrix of order $N \times N$, where N is the number of nodes. As you might notice, in order to calculate $\mathbf{u_{n+1}}$ and $\mathbf{v_{n+1}}$ we need $\mathbf{F_{n+1}}$ and $\mathbf{G_{n+1}}$. As you might recall from equation 18 representing the force vectors and the pair of equations 3 and 4 representing the reaction terms, we may conclude that $\mathbf{F_{n+1}}$ and $\mathbf{G_{n+1}}$ are non-linear functions (in fact, to be particular definite integrals over the domain involving non-linear functions of $\mathbf{u_{n+1}}$ and $\mathbf{v_{n+1}}$). Hence solving the implicit scheme is not as straightforward as that in the explicit scheme. We need to use some powerful hammers.

5.2.2.1 Picard's Iteration

Just to recall, in the backward Euler scheme within finite element methods, Picard's iteration is an iterative process used to solve nonlinear equations by linearizing them at each step, enabling convergence toward the solution. It involves updating the solution iteratively based on successive linear approximations until reaching a stable outcome within each time step.

Note that here everything (even $\mathbf{u_n}$ and $\mathbf{v_n}$) are known and we aim to find $\mathbf{u_{n+1}}$ and $\mathbf{v_{n+1}}$. Now we denote each iteration of the Picard's by the symbol θ and we start with a guess value for $\mathbf{u^0_{n+1}}$ and $\mathbf{v^0_{n+1}}$ (i.e. for $\theta = 0$), usually by $\mathbf{u_n}$ and $\mathbf{v_n}$ respectively. Now even $\mathbf{u^\theta_{n+1}}$ and $\mathbf{v^\theta_{n+1}}$ are known; the only unknown

[10] The explicit scheme [11] The implicit scheme

variables are $\mathbf{u}_{\mathbf{n+1}}^{\theta+1}$ and $\mathbf{v}_{\mathbf{n+1}}^{\theta+1}$. We iterate until the maximum of the successive difference of ($\mathbf{u}_{\mathbf{n+1}}^{\theta+1}$ and $\mathbf{u}_{\mathbf{n+1}}^{\theta}$) and ($\mathbf{v}_{\mathbf{n+1}}^{\theta+1}$ and $\mathbf{v}_{\mathbf{n+1}}^{\theta}$) under some norm[12] is less than certain threshold.

$$\begin{aligned}(\mathbf{M}+\Delta t\mathbf{K}_{\mathbf{1,n+1}}^{\theta})\cdot\mathbf{u}_{\mathbf{n+1}}^{\theta+1}&=\mathbf{M}\cdot\mathbf{u}_{\mathbf{n}}-\Delta t{\mathbf{K}_{\mathbf{3,n+1}}}^{\theta}\cdot\mathbf{v}_{\mathbf{n+1}}^{\theta}+\Delta t\mathbf{F}_{\mathbf{n+1}}^{\theta}\\(\mathbf{M}+\Delta t\mathbf{K}_{\mathbf{2,n+1}}^{\theta})\cdot\mathbf{v}_{\mathbf{n+1}}^{\theta+1}&=\mathbf{M}\cdot\mathbf{v}_{\mathbf{n}}+\Delta t\,\mathbf{G}_{\mathbf{n+1}}^{\theta}\end{aligned}\tag{23}$$

5.2.2.2 Newton- Raphson method

The error term may be defined as:

$$\begin{aligned}R_1(u,v)&=\int_\Omega wf\,d\Omega-\int_\Omega w\frac{\partial u}{\partial t}\,d\Omega-\int_\Omega\nabla w\cdot\left(a\nabla u+c\nabla v\right)d\Omega\\R_2(u,v)&=\int_\Omega wg\,d\Omega-\int_\Omega w\frac{\partial v}{\partial t}\,d\Omega-\int_\Omega\nabla w\cdot b\nabla v\,d\Omega\end{aligned}$$

We need to find the roots of both expressions.

Using the Taylor series expansion, we may represent the first order approximation of it as follows:

$$\begin{aligned}R_1(u,v)&=R_1(u^\theta,v^\theta)+\left.\frac{\partial R_1}{\partial u}\right|_{(u^\theta,v^\theta)}\delta u_\theta+\left.\frac{\partial R_1}{\partial v}\right|_{(u^\theta,v^\theta)}\delta v_\theta\\R_2(u,v)&=R_2(u^\theta,v^\theta)+\left.\frac{\partial R_2}{\partial u}\right|_{(u^\theta,v^\theta)}\delta u_\theta+\left.\frac{\partial R_2}{\partial v}\right|_{(u^\theta,v^\theta)}\delta v_\theta\end{aligned}$$

We aim to find the zeros of $R_1(u,v)$ and $R_2(u,v)$, hence we set those two terms on the LHS to be zero. Hence

$$\begin{aligned}-R_1(u^\theta,v^\theta)&=\left.\frac{\partial R_1}{\partial u}\right|_{(u^\theta,v^\theta)}\delta u_\theta+\left.\frac{\partial R_1}{\partial v}\right|_{(u^\theta,v^\theta)}\delta v_\theta\\-R_2(u^\theta,v^\theta)&=\left.\frac{\partial R_2}{\partial u}\right|_{(u^\theta,v^\theta)}\delta u_\theta+\left.\frac{\partial R_2}{\partial v}\right|_{(u^\theta,v^\theta)}\delta v_\theta\end{aligned}$$

Thus, we obtain the following matrix representation of the Newton-Raphson method.

$$-\begin{pmatrix}R_1(u^\theta,v^\theta)\\R_2(u^\theta,v^\theta)\end{pmatrix}=\begin{pmatrix}\frac{\partial R_1}{\partial u}&\frac{\partial R_1}{\partial v}\\\frac{\partial R_2}{\partial u}&\frac{\partial R_2}{\partial v}\end{pmatrix}\begin{pmatrix}\delta u_\theta\\\delta v_\theta\end{pmatrix}=\begin{pmatrix}\frac{\partial R_1}{\partial u}&\frac{\partial R_1}{\partial v}\\\frac{\partial R_2}{\partial u}&\frac{\partial R_2}{\partial v}\end{pmatrix}\begin{pmatrix}u_{\theta+1}-u_\theta\\v_{\theta+1}-v_\theta\end{pmatrix}$$

Hence,

$$\begin{pmatrix}u_{\theta+1}\\v_{\theta+1}\end{pmatrix}=\begin{pmatrix}u_\theta\\v_\theta\end{pmatrix}-\begin{pmatrix}\frac{\partial R_1}{\partial u}&\frac{\partial R_1}{\partial v}\\\frac{\partial R_2}{\partial u}&\frac{\partial R_2}{\partial v}\end{pmatrix}^{-1}\begin{pmatrix}R_1(u^\theta,v^\theta)\\R_2(u^\theta,v^\theta)\end{pmatrix}$$

... to be continued in future work.

[12] Usually Euclidean norm

5.2.3 CFL[13] condition

Just to recall, the CFL condition in finite element analysis, for both Forward and Backward Euler schemes, mandates that the chosen time step size must be small enough to maintain numerical stability. In Forward Euler, the time step should be less than or equal to a factor divided by the maximum element size squared, ensuring stability by capturing fast wave propagation. Backward Euler, being implicit, often allows for slightly larger time steps while ensuring stability. Adhering to the CFL condition ensures accurate and stable solutions in finite element analysis by balancing the time step size with the system's characteristics.

Here for the forward Euler scheme, suppose r is the characteristic length, which we will discuss soon. Our parameters are a_0, b_0 and c_0 for Model 1. It also involves c_1 in the case of Model 2. We have our rough estimate of CFL condition as follows

$$\Delta t \leq \frac{r^2}{2 \max\left(a_0, b_0, c_0, c_1\right)} \tag{24}$$

Now coming to the part, how we calculate r. To get a rough estimate of an element, divide the area of the domain by the number of elements. Now we get the average area of an element. Assume it to be an equilateral triangle. r is simply the incircle of this triangle.

[13] **C**ourant-**F**riedrichs-**L**ewy

6 IMPLEMENTATION

1. We define various parameters, the spatial domain and the time interval for the simulation.

2. We use *gmsh* to discretize the domain into triangular elements and assign nodal values for the primary variables (susceptible and infected populations).

3. We derive the weak form of the reaction-diffusion system by multiplying the strong form by a test function and integrating it over the domain (11, 12). Then, we apply the boundary and initial conditions.

4. We express the primary variables and the test function as linear combinations of shape functions (linear Lagrange interpolant) and nodal values (13).

5. We transform the physical domain into a parametric domain using a standard triangle. Jacobian matrix (15) is calculated to relate the derivatives in the two domains.

6. We implement the numerical implementation scheme by discretizing the terms and representing them in matrix format given in 16, 17 and 18. We assemble the global stiffness and mass matrices and the force vectors from the local element matrices and vectors. We use numerical integration (Gauss quadrature) to evaluate the integrals over the elements.

7. We discretize the time derivative using the forward Euler (21) or backward Euler (22) scheme. For the backward scheme, we use Picard's iteration (23) or Newton-Raphson method to solve the nonlinear system.

8. We solve the linear or nonlinear system of equations for the nodal values at each time step. We update the primary variables and the reaction terms accordingly.

9. Finally, we plot the spatial patterns of the susceptible and infected populations and analyze the effects of different model parameters and diffusion coefficients. We also analyse the space and time convergence of the mesh.

10. We do the above steps to simulate all the plots and results discussed in [2] along with a specific case which resembles a real-life scenario. For the details, please refer to the section 7.

7 RESULTS AND INTERPRETATIONS

Our study primarily encompasses two key segments. Firstly, we replicate the simulations detailed in [2]. In each scenario, we initialize both the susceptible and infected populations with slight deviations from their equilibrium values. This section comprises two subsections: one delves into Model 1, wherein the diffusion coefficients remain constants, while the other focuses on Model 2, where these coefficients become functions of u and v.

The replication phase predominantly involves initializing the susceptible and infected populations with closely proximate initial values. This deliberate choice allows us to examine the impact of cross-diffusion terms and the resulting patterns across most instances.

The second section of our study shifts focus to a real-world scenario, wherein, during the initial phase, the number of infected individuals remains notably smaller in comparison to the susceptible population. Here, we explore two cases: one involving a nonzero birth rate and another where the birth rate is zero, aiming to encapsulate scenarios related to a static population size.

7.1 Reproduced results

With all the hammers discussed in the previous section, we simulate the situations discussed in [2]. Let's deep dive into it.

Example 1

In Example 1, Model 1 is simulated, where the parameters are chosen according to [6] as given in [2]. The model is configured with specific parameter values: $K = 1000$, $\beta = 0.5$. The fixed self- and cross-diffusion coefficients are set as follows: $a_0 = 0.1$, $b_0 = 2$, $c_0 = 0.02$. For the backward Euler scheme, the reference tolerance stands at $\varepsilon = 1e{-}5$. Our initial data assumes a random density perturbation around the endemic stationary state (u^*, v^*). In essence, as we have discussed in section 2, the initial state of the system at time $t = 0$ is described as follows:

$$u, (x, y, 0) = u^* + \rho_u \delta_u(x, y)$$
$$v, (x, y, 0) = v^* + \rho_v \delta_v(x, y)$$

Here, (u^*, v^*) denotes the equilibrium point, specifically $(u, (x, y, \infty), v, (x, y, \infty))$. The variables $\delta_w(x, y) \in [0, 1]$, where w belongs to the set u, v, represent uniformly distributed random variables and the value corresponding to the coordinate (x, y), and ρ_w signifies a scaling factor, that is set to one in each of the cases in this subsection of Example 1.

In Examples 1A, 1B, and 1C, we fix parameters as $k = 0.25$ and $r = 0.27$, resulting in an equilibrium point of $(u^*, v^*) = (74.0741, 74.0741)$. The computational domain for Examples 1A, 1C, and 1D spans the region $\Omega = (0, L) \times (0, W)$, where $L = W = 200$. However, in Example 1B, we reduce the domain size to $(0, 50)^2$. This modification allows for a more detailed observation of the solution's behaviour.

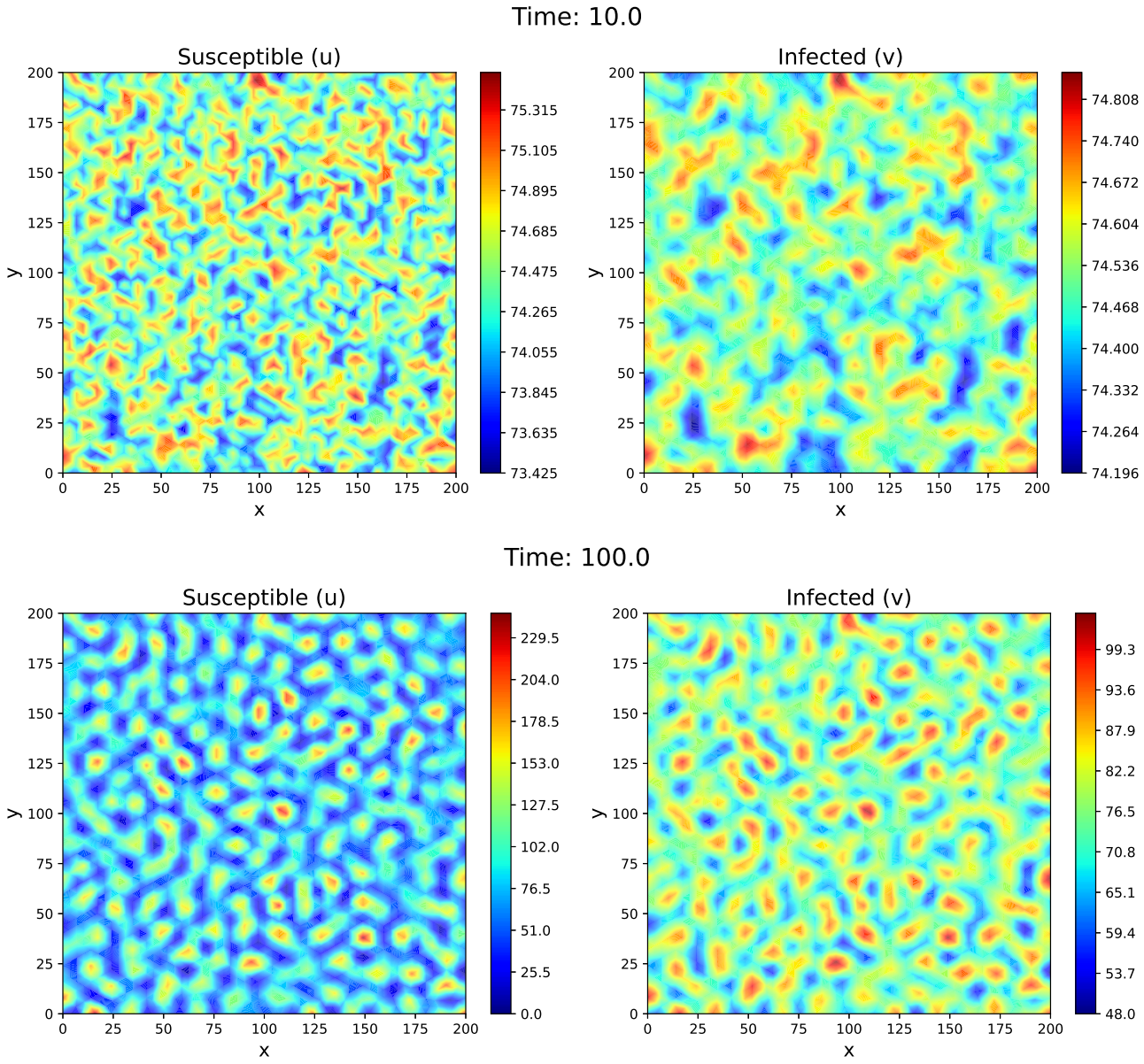
7.1.1 Example 1A

1. **Model parameters:**
   $r = 0.27$, $K = 1000$, $\beta = 0.5$, $k = 0.25$, $(u^*, v^*) = (74.0741, 74.0741)$,
   Model = 'Model 1', Euler = 'Forward', $a_0 = 0.1$, $b_0 = 2$, $c_0 = 0.02$,
   $\rho_u = \rho_v = 1$, $\delta_u(x, y) = \delta_v(x, y) \sim [0, 1]$, $L = W = 200$, mesh resolution $= 0.02$,

dt $= 0.2$, $t_0 = 0$, $t_\infty = 500$.

Number of nodes: 3018
Number of elements: 5834

2. **Plots:**



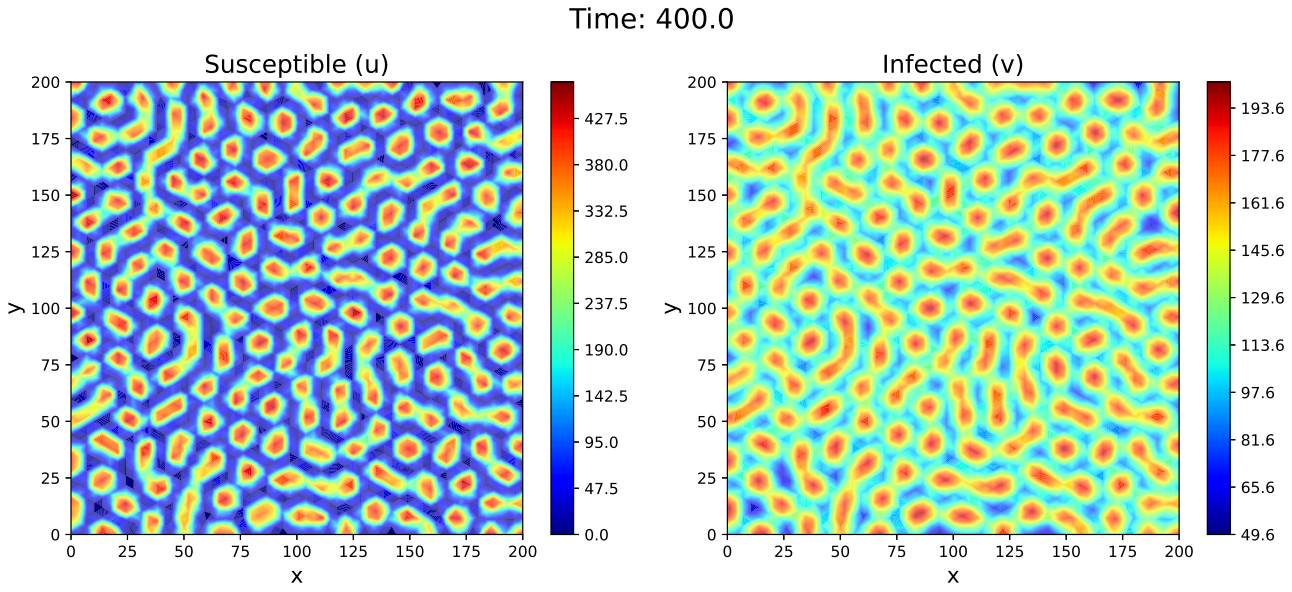


Figure 2: Example 1A Plots for Forward Euler

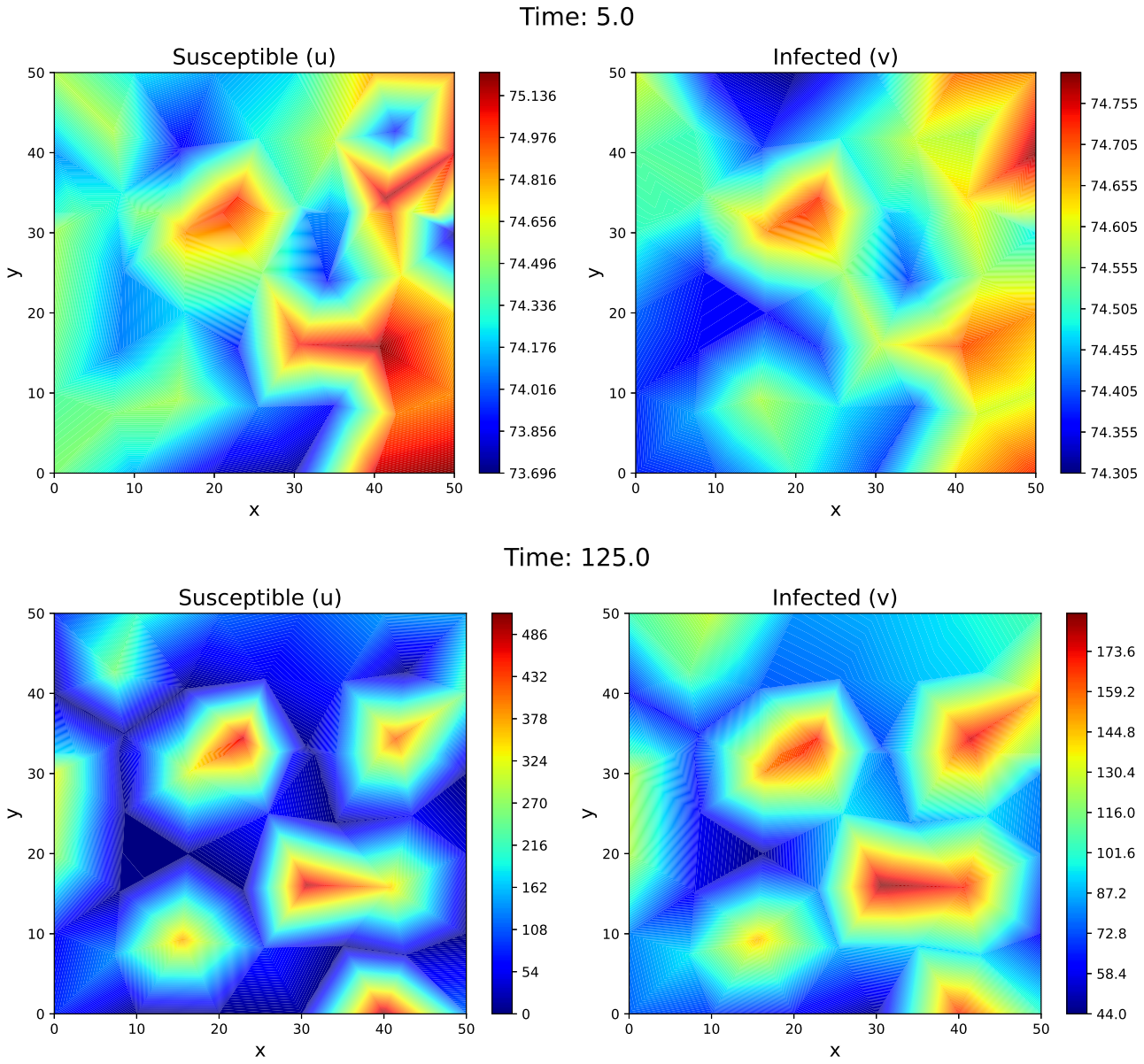
7.1.2 Example 1B

7.1.2.1 Forward Euler

1. **Model parameters:**
   $r = 0.27$, $K = 1000$, $\beta = 0.5$, $k = 0.25$, $(u^*, v^*) = (74.0741, 74.0741)$,
   Model = 'Model 1', Euler = 'Forward', $a_0 = 0.1$, $b_0 = 2$, $c_0 = 0.02$,
   $\rho_u = \rho_v = 1$, $\delta_u(x, y) = \delta_v(x, y) \sim [0, 1]$, $L = W = 50$, mesh resolution = 0.2,
   dt = 0.01, $t_0 = 0$, $t_\infty = 250$.

   Number of nodes: 64
   Number of elements: 104

2. **Plots:**


Time: 5.0
Susceptible (u)
Infected (v)
x
y
Time: 125.0
Susceptible (u)
Infected (v)
x
y

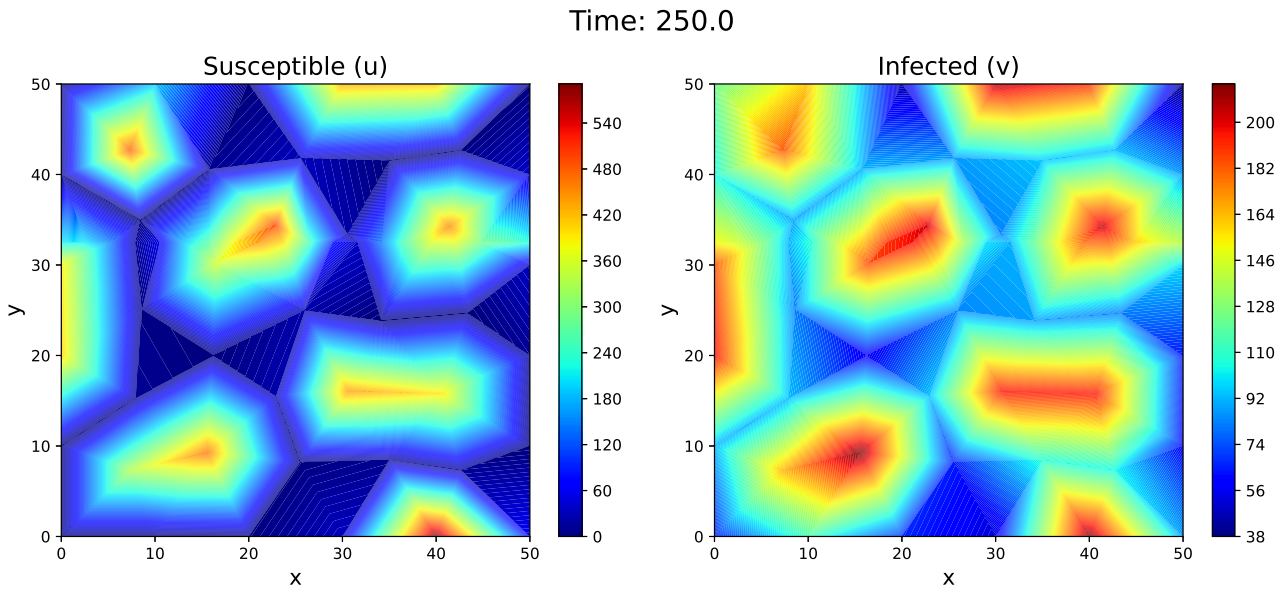


Figure 3: Example 1B Plots for Forward Euler

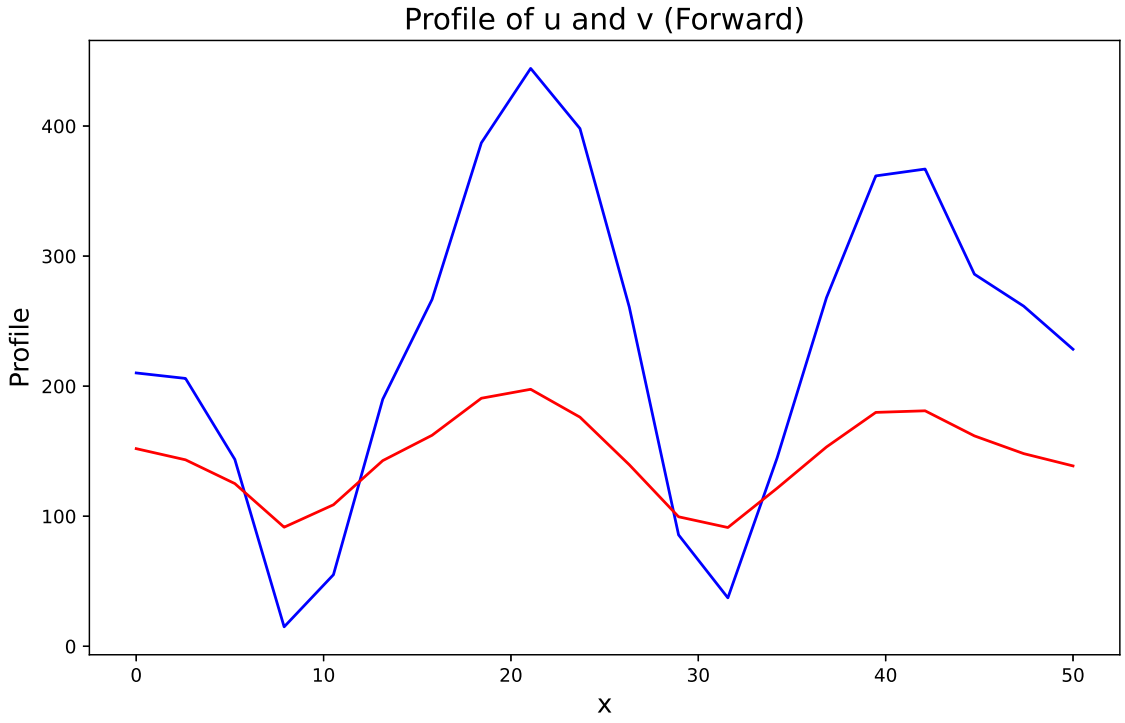
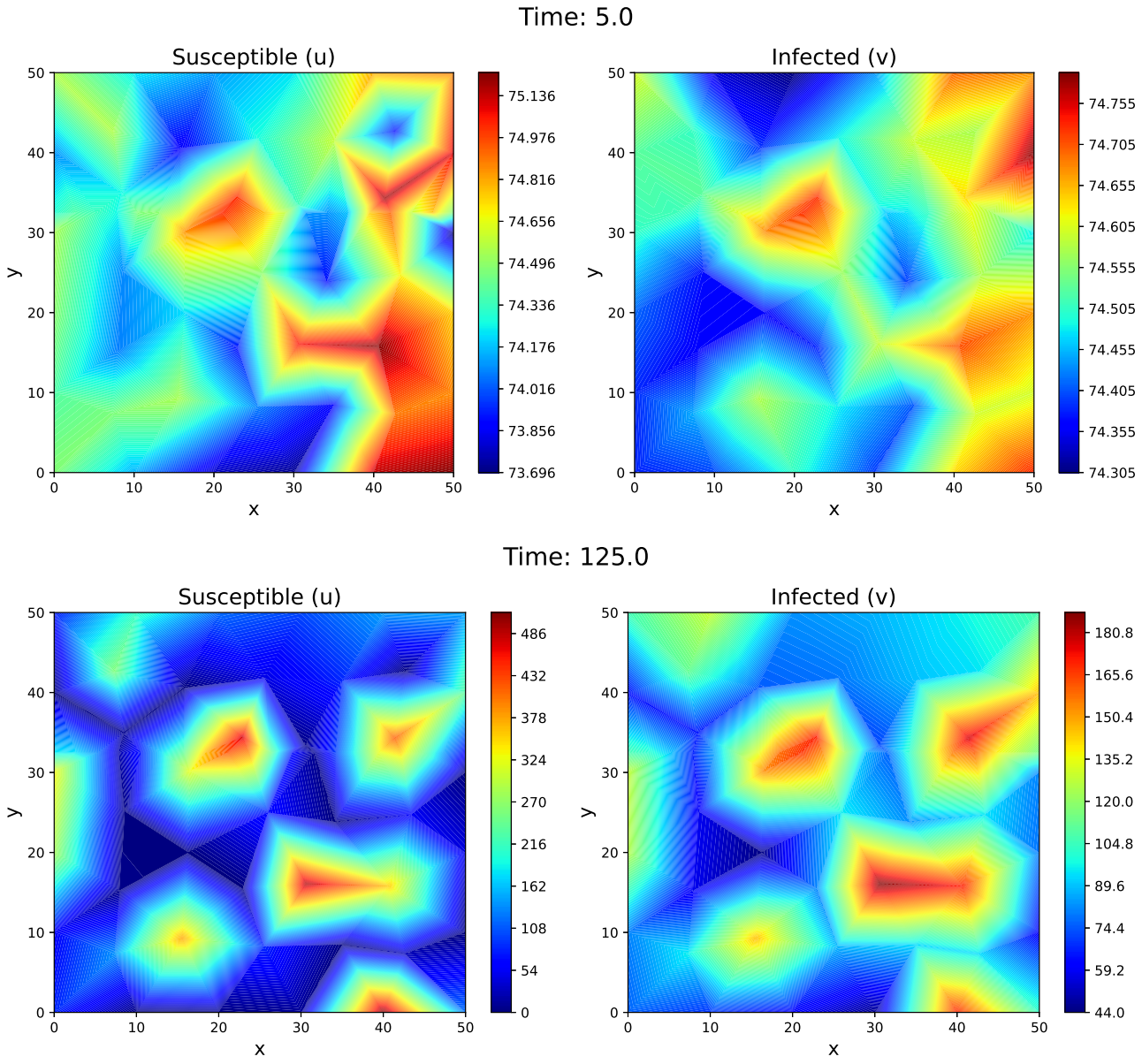


Figure 4: Profile of Forward Euler for u and v at $y = 32.5$ and time instance, $t = 250$

7.1.2.2 Backward Euler

1. **Model parameters:**
   All the parameters are the same as that of Example 1B-forward, except
   Euler = 'Backward', dt = 0.05, tolerance $\epsilon = 1e - 5$.

2. **Plots:**


Time: 5.0
Susceptible (u)
Infected (v)
x
y
Time: 125.0
Susceptible (u)
Infected (v)
x
y

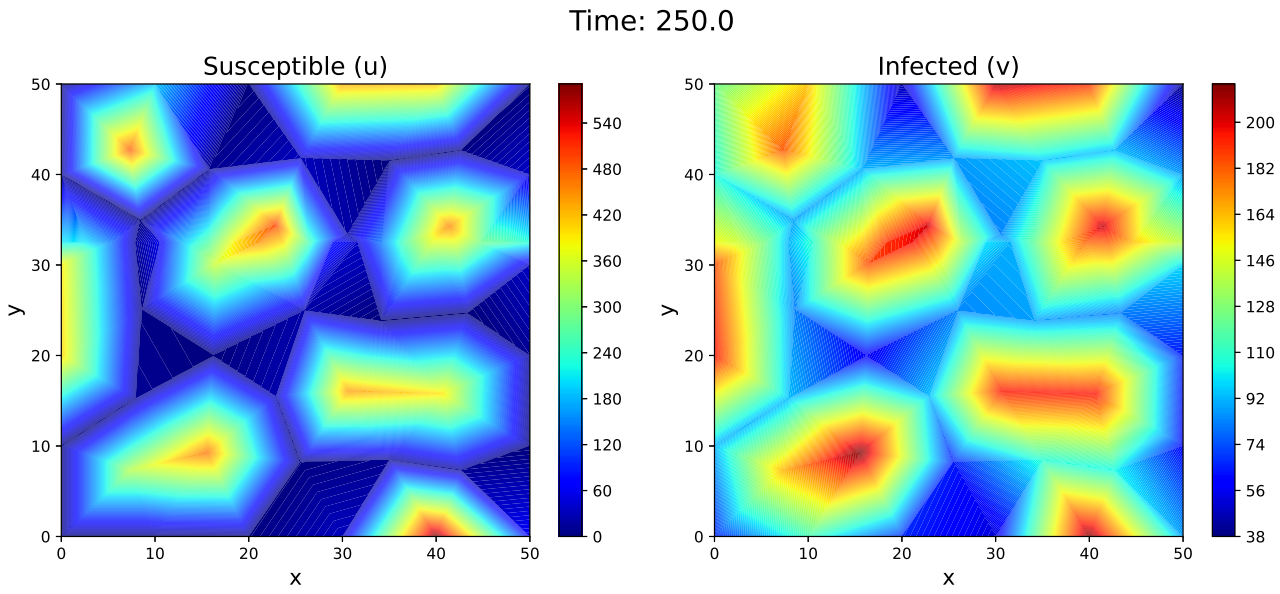


Figure 5: Example 1B Plots for Backward Euler

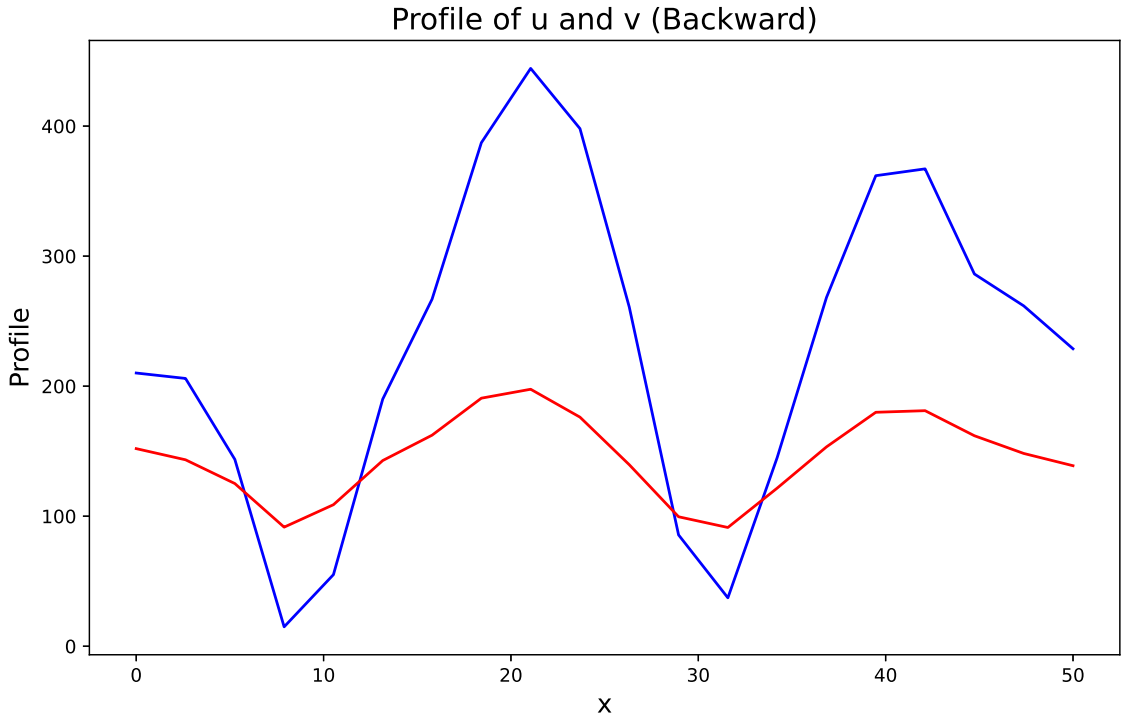


Figure 6: Profile of Backward Euler for u and v at $y = 32.5$ and time instance, $t = 250$

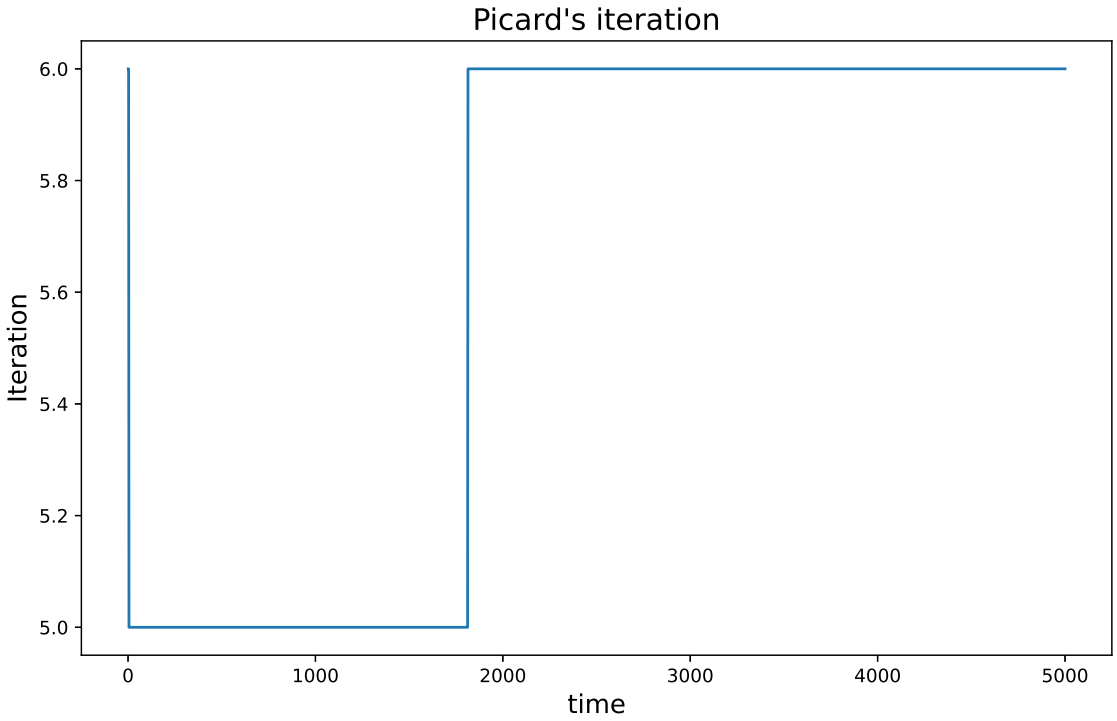


Figure 7: Picard's Iteration

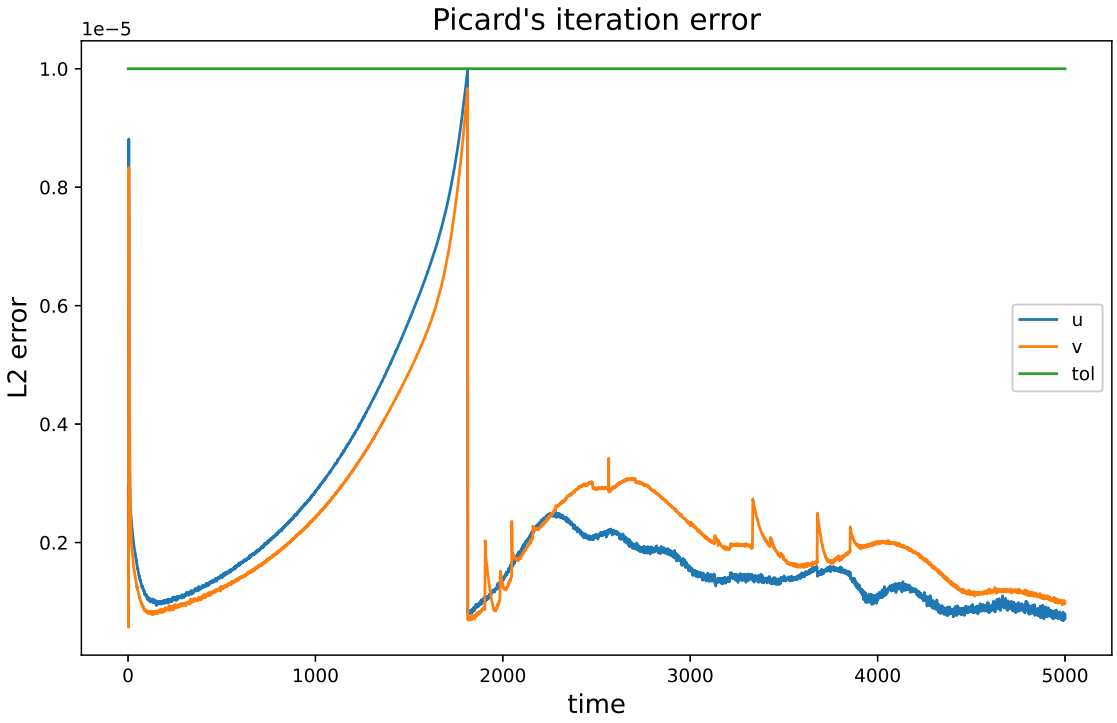


Figure 8: Picard's Iteration Error

7.1.3 Example 1C

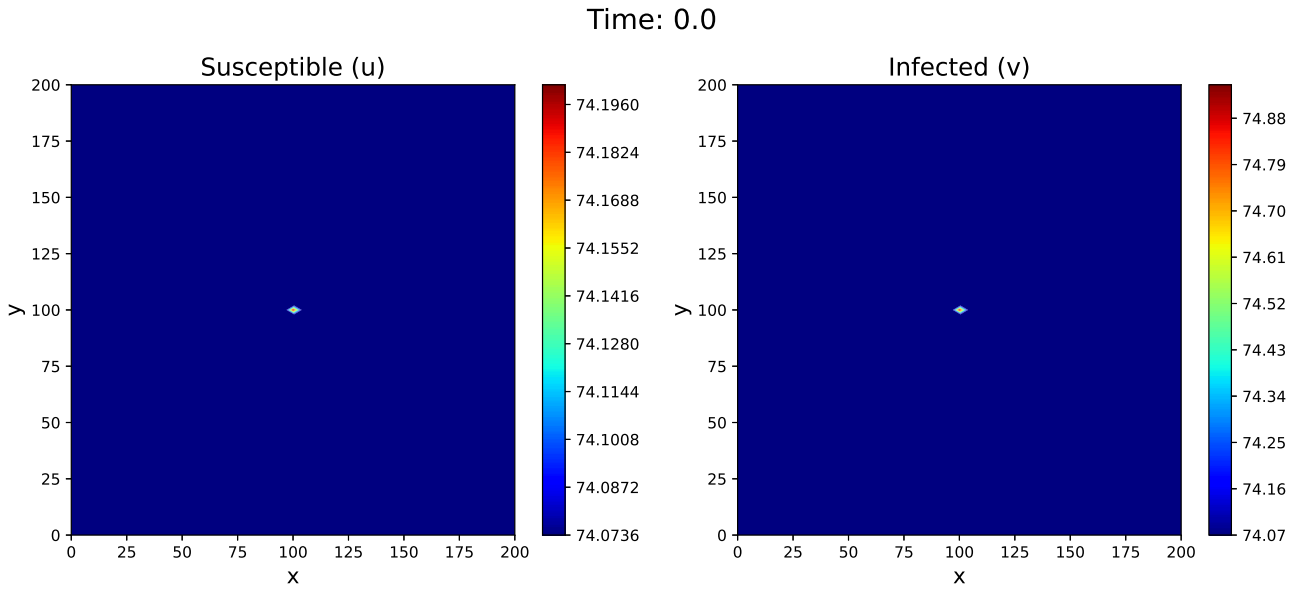
1. **Model parameters:**
   $r = 0.27$, $K = 1000$, $\beta = 0.5$, $k = 0.25$, $(u^*, v^*) = (74.0741, 74.0741)$,
   Model = 'Model 1', Euler = 'Forward', $a_0 = 0.1$, $b_0 = 2$, $c_0 = 0.02$,
   $\rho_u = \rho_v = 1$, $\delta_u(x, y) = \delta_v(x, y) \sim [0, 1]$, $L = W = 200$, mesh resolution = 0.02,
   dt = 0.01, $t_0 = 0$, $t_\infty = 500$.

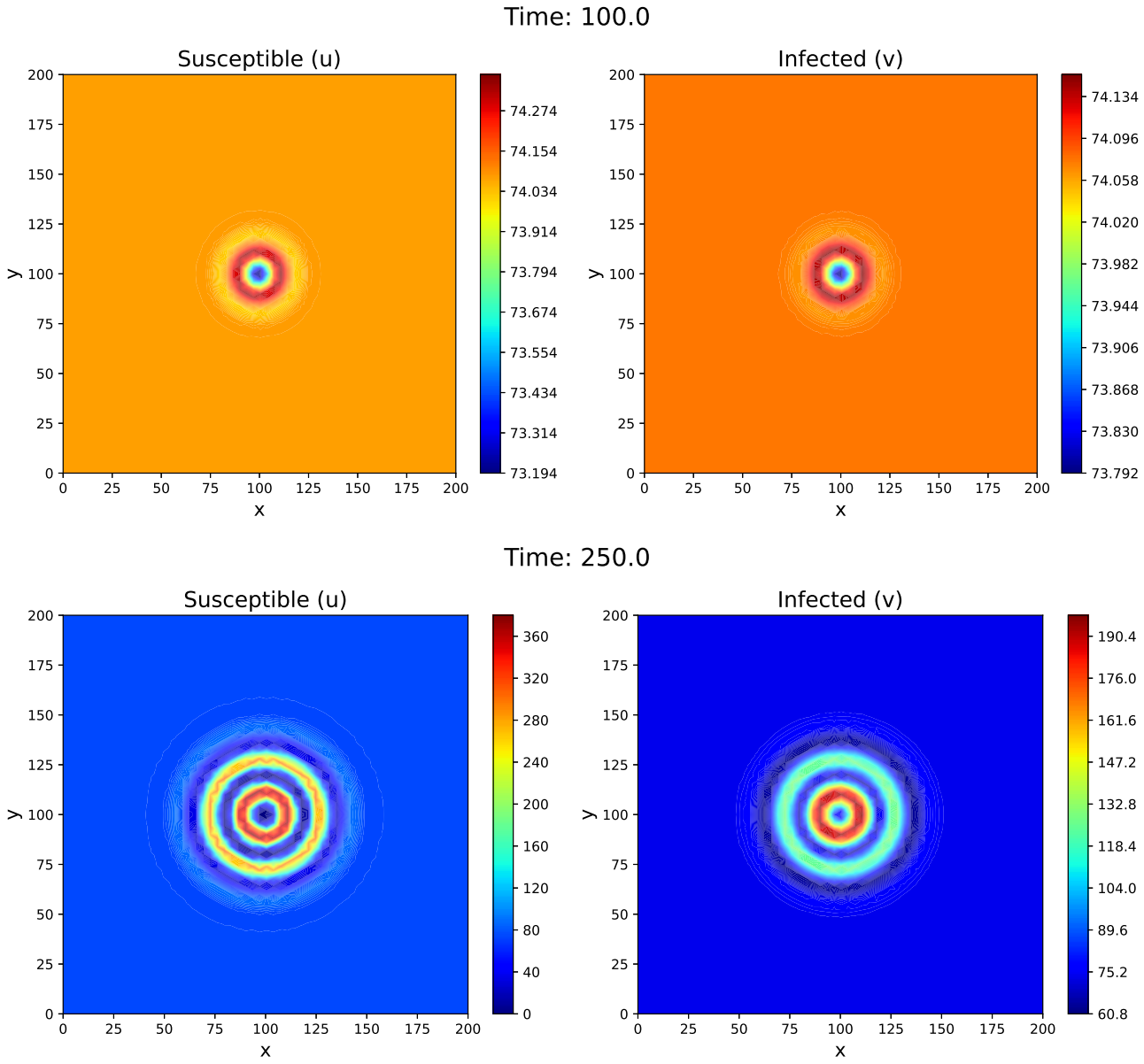
   Also, for Example 1C, the difference from Example 1A is that the initial data is now randomly distributed at only one spatial point, that is, we have

$$\delta_u(x, y) = \delta_v(x, y) = \begin{cases} \sim [0, 1] & \text{if } (x, y) = (L/2, W/2) \\ 0 & \text{otherwise} \end{cases}$$

   Number of nodes: 3019
   Number of elements: 5836

2. **Plots:**




Time: 100.0
Susceptible (u)
Infected (v)
Time: 250.0
Susceptible (u)
Infected (v)
x
y

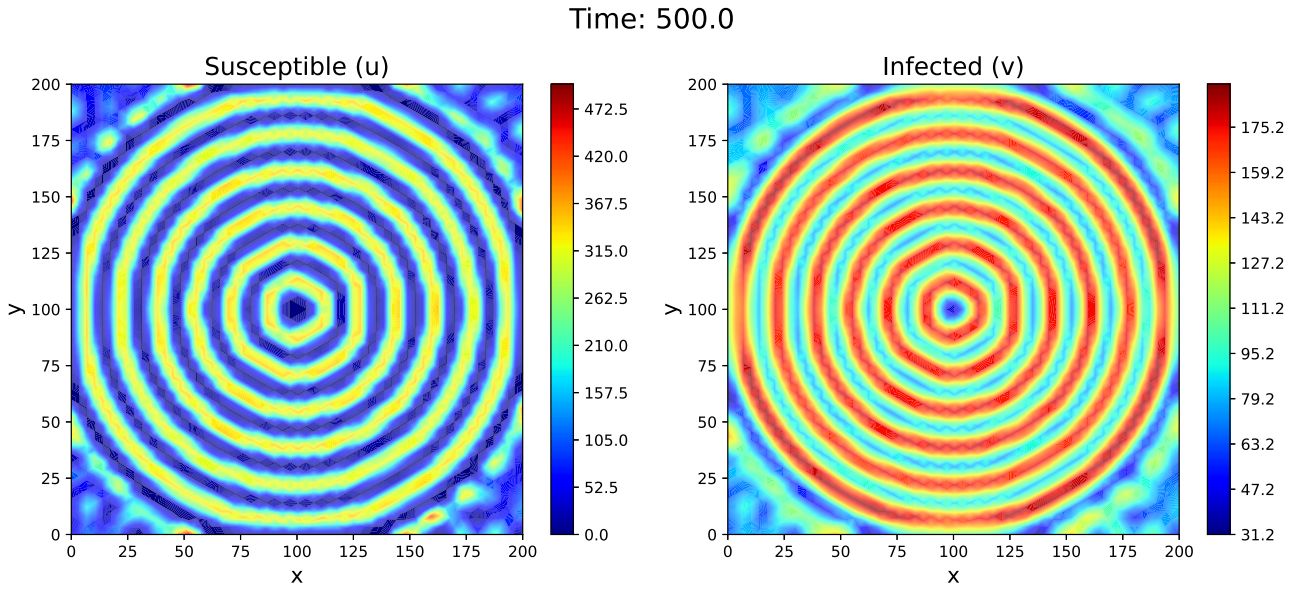


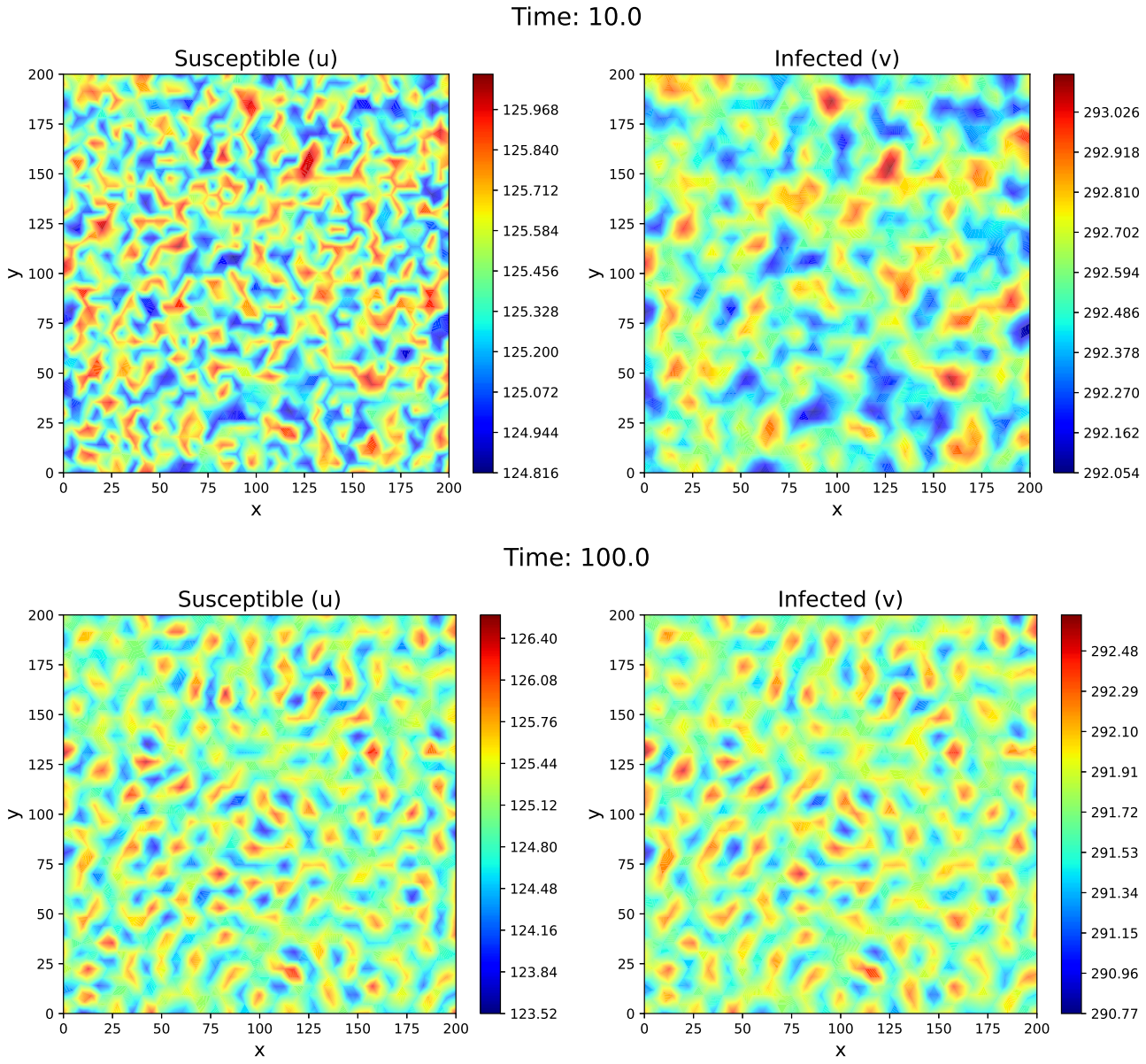
Figure 9: Example 1C Plots for Forward Euler

7.1.4 Example 1D

1. **Model parameters:**
   r = 0.4, $K = 1000$, $\beta = 0.5$, **k = 0.15**, $(u^*, v^*) = (125, 291.666)$,
   Model = 'Model 1', Euler = 'Forward', $a_0 = 0.1$, $b_0 = 2$, $c_0 = 0.02$,
   $\rho_u = \rho_v = 1$, $\delta_u(x, y) = \delta_v(x, y) \sim [0, 1]$, $L = W = 200$, mesh resolution = 0.025,
   dt = 0.1, $t_0 = 0$, $t_\infty = 500$.

   Number of nodes: 1939
   Number of elements: 3716

2. **Plots:**


Time: 10.0
Susceptible (u)
Infected (v)
y
x
0
25
50
75
100
125
150
175
200
125.968
125.840
125.712
125.584
125.456
125.328
125.200
125.072
124.944
124.816
293.026
292.918
292.810
292.702
292.594
292.486
292.378
292.270
292.162
292.054
Time: 100.0
Susceptible (u)
Infected (v)
126.40
126.08
125.76
125.44
125.12
124.80
124.48
124.16
123.84
123.52
292.48
292.29
292.10
291.91
291.72
291.53
291.34
291.15
290.96
290.77

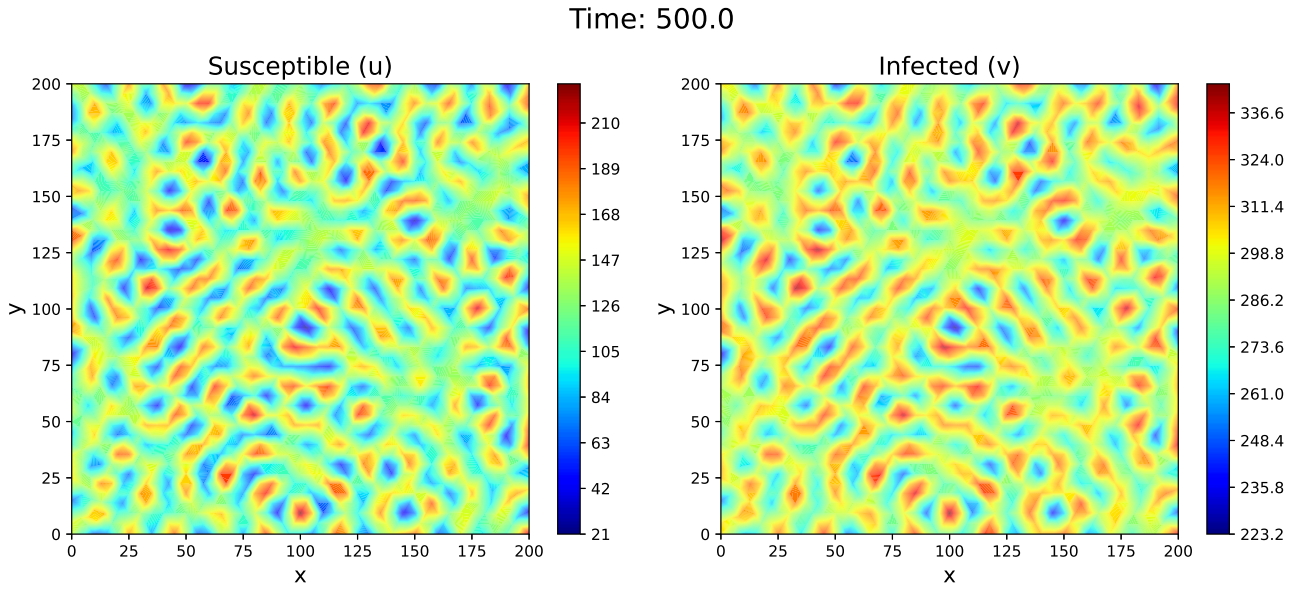


Figure 10: Example 1D Plots for Forward Euler

7.1.5 Inferences on Example 1

In the above subsections, we study and depict snapshots of the numerical solution u (the patterns of the species v coincide with those of u) for different time instants for Examples 1A, 1B, 1C, 1D, respectively. It can be seen how the initial random distribution of the densities evolves to reach a non- constant stationary state, that is, we observe the formation of spatial patterns induced by the Turing phenomenon. This pattern formation is remarkable since the data of Example 1 give the Jacobian matrix[14]

$$F(u^*, v^*) = \begin{pmatrix} 0.105 & -0.125 \\ 0.125 & -0.125 \end{pmatrix}$$

which has the eigenvalues $\lambda_{1,2} = -0.01 \pm \sqrt{0.0024}$. That means that without the cross-diffusion term, the solution would converge to a constant stationary state all over the two-dimensional domain.

In Examples 1A and 1B, there is an emergence of concentrated regions[15] of susceptible individuals, resembling 'islands'. This phenomenon signifies a phase separation, where the susceptible species actively segregate from the infected species.

In Example 1C, it's evident that perturbing a single point triggers pattern formation across the entire domain. Unlike other simulations, spatial patterns become distinctly visible at earlier time intervals due to deterministic perturbation in the constant initial data, ensuring reproducibility in the numerical solution. Figure 9 illustrates pattern reflections at the zero-flux but non-absorbing boundaries, disrupting the concentric structure. To generate solutions with more intricate spatial patterns, perturbations in two or three points would suffice.

Example 2

In Example 2, Model 2 is simulated, where the parameters for the reaction equation are the same as in Example 1A–C. The parameter of the nonlinear self-diffusion is $m = -1/2$. In Example 2A, the

[14] Please refer to the appendix to read more about it. [15] aka islands

self-diffusion coefficients are set to $a_0 = 1$, $b_0 = 6$. For the cross-diffusion, we put $c_0 = 12$, and the parameter c_1 is set to $c_1 = u^* + v^*$ (total population in the endemic stationary state). In the initial condition, we have now $\rho_u = \rho_v = 1e-4$.

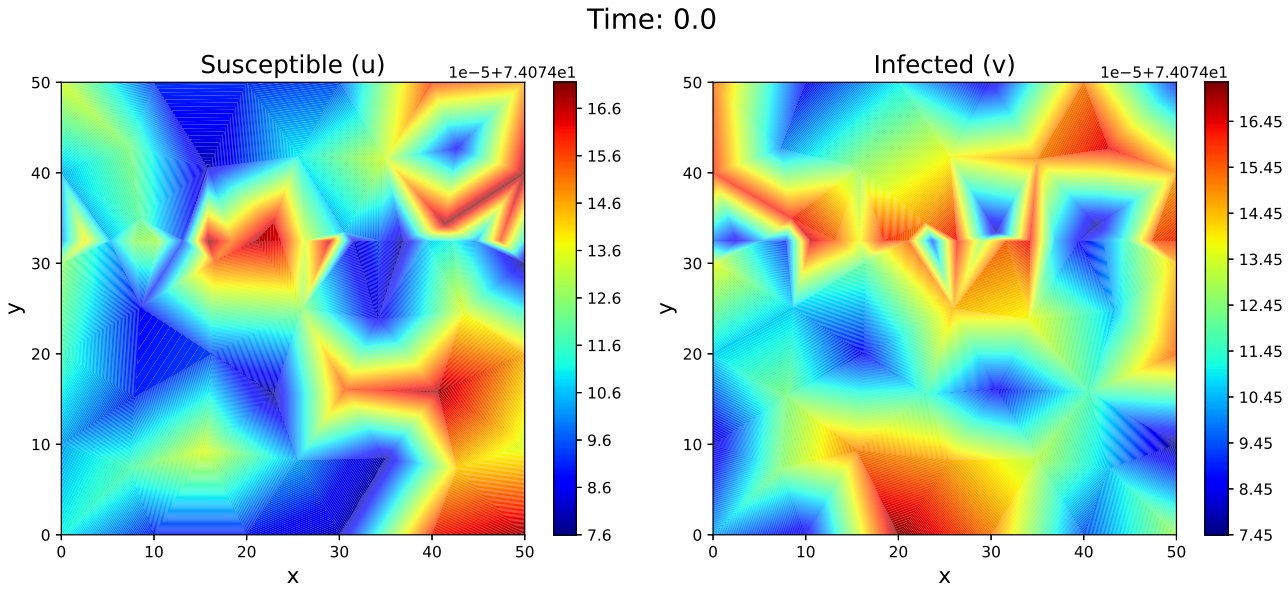
7.1.6 Example 2A

7.1.6.1 Forward Euler

1. **Model parameters:**
   $r = 0.27$, $K = 1000$, $\beta = 0.5$, $k = 0.25$, $(u^*, v^*) = (74.0741, 74.0741)$,
   Model = 'Model 2', Euler = 'Forward', $a_0 = 1$, $b_0 = 6$, $c_0 = 12$, $c_1 = u^* + v^*$, $m = -0.5$
   $\rho_u = \rho_v = 1e-4$, $\delta_u(x, y) = \delta_v(x, y) \sim [0, 1]$, $L = W = 50$, mesh resolution = 0.2,
   dt = 0.005, $t_0 = 0$, $t_\infty = 250$.

   Number of nodes: 64
   Number of elements: 104

2. **Plots:**



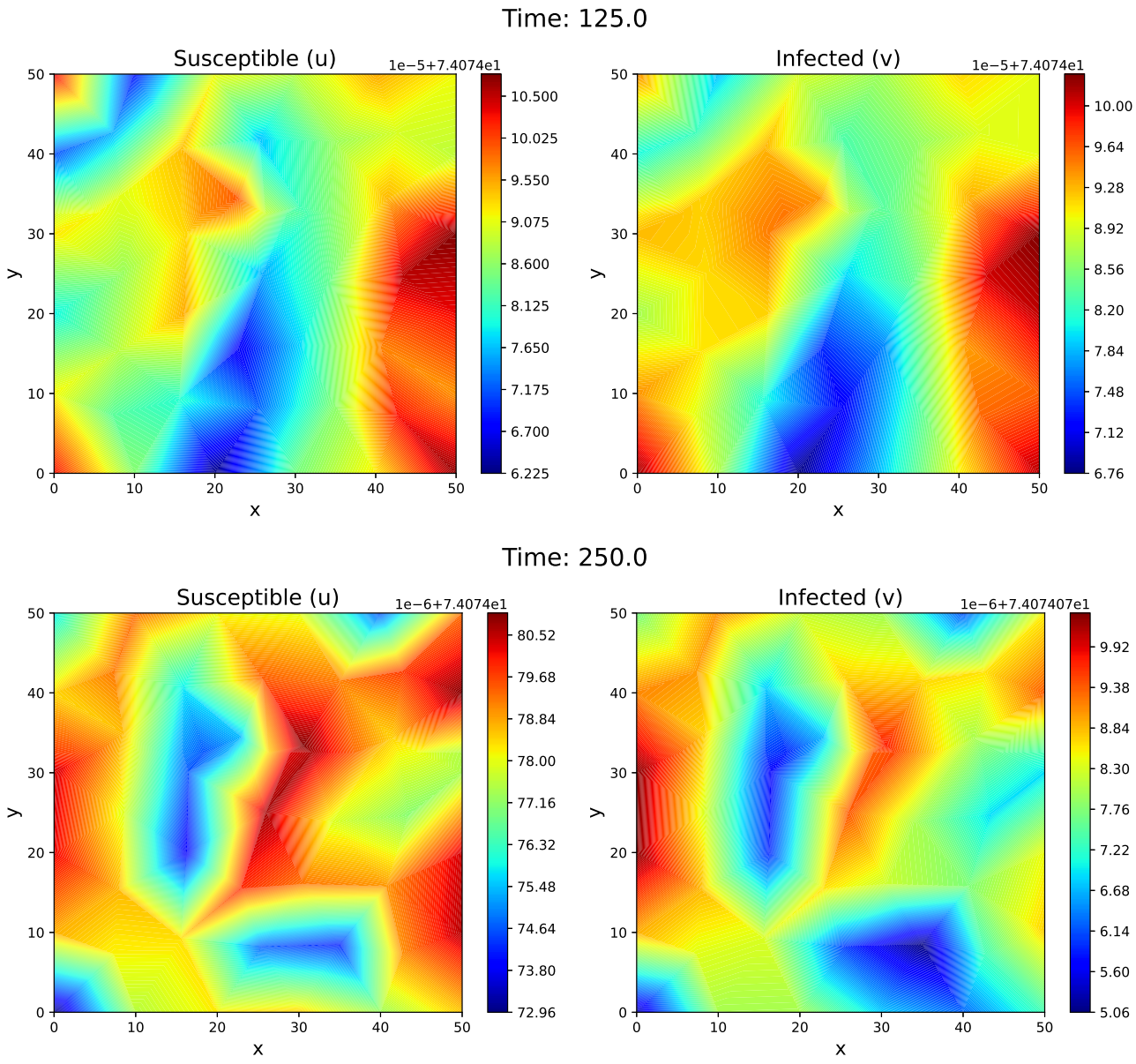


Figure 11: Example 2A Plots for Forward Euler

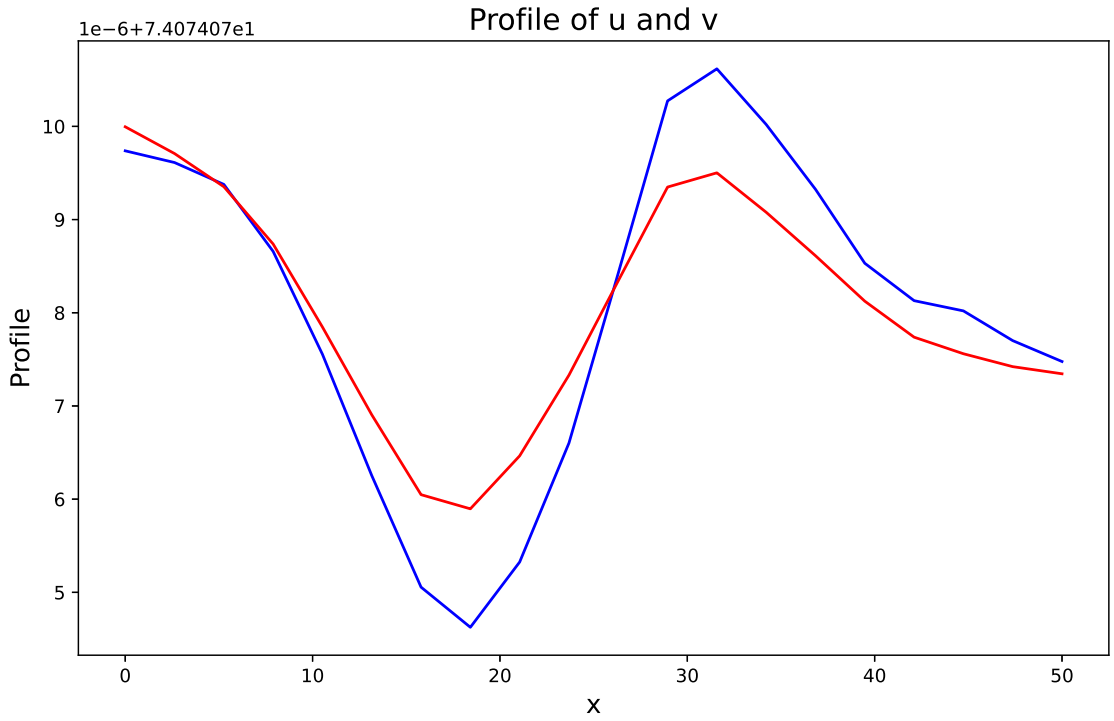
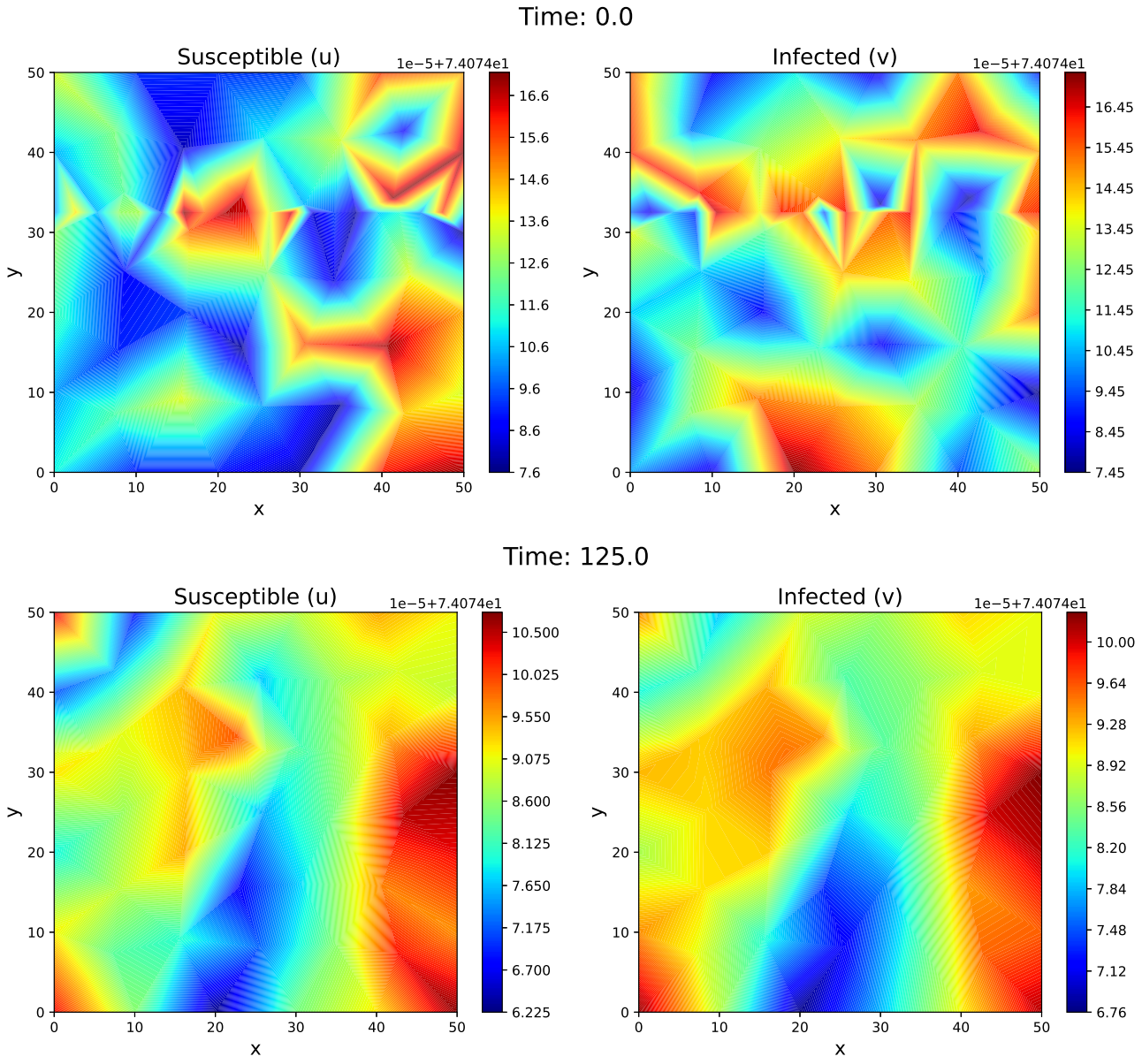


Figure 12: Profile of Forward Euler for u and v at $y = 32.5$ and time instance, $t = 250$

7.1.6.2 Backward Euler

1. **Model parameters:**
   All the parameters are the same as that of Example 2A-forward, except
   Euler = 'Backward', dt = 0.01, tolerance $\epsilon = 1e - 5$.

2. **Plots:**


Time: 0.0
Susceptible (u)
1e−5+7.4074e1
Infected (v)
1e−5+7.4074e1
Time: 125.0
Susceptible (u)
1e−5+7.4074e1
Infected (v)
1e−5+7.4074e1
x
y

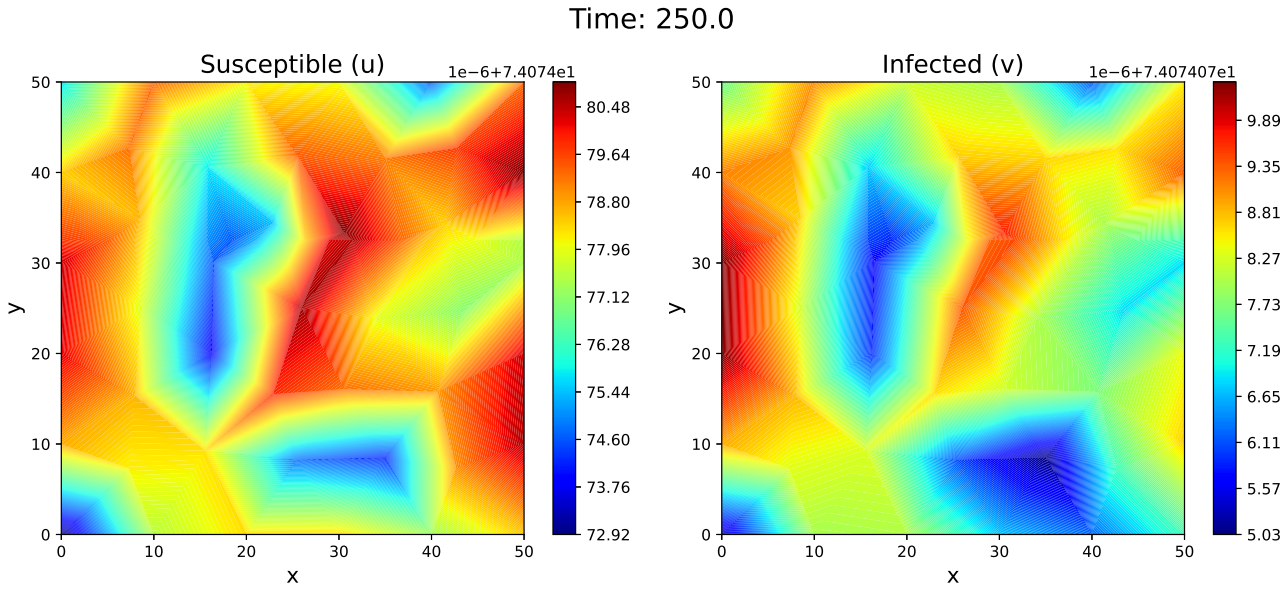


Figure 13: Example 2A Plots for Backward Euler

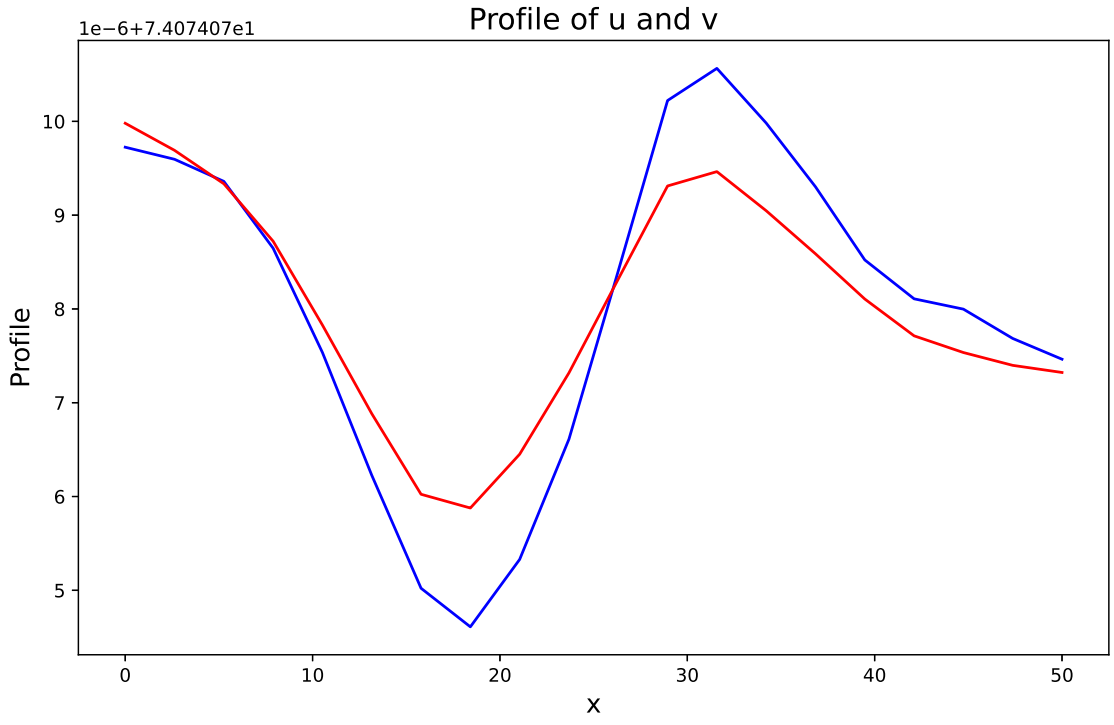


Figure 14: Profile of Backward Euler for u and v at $y = 32.5$ and time instance, $t = 250$

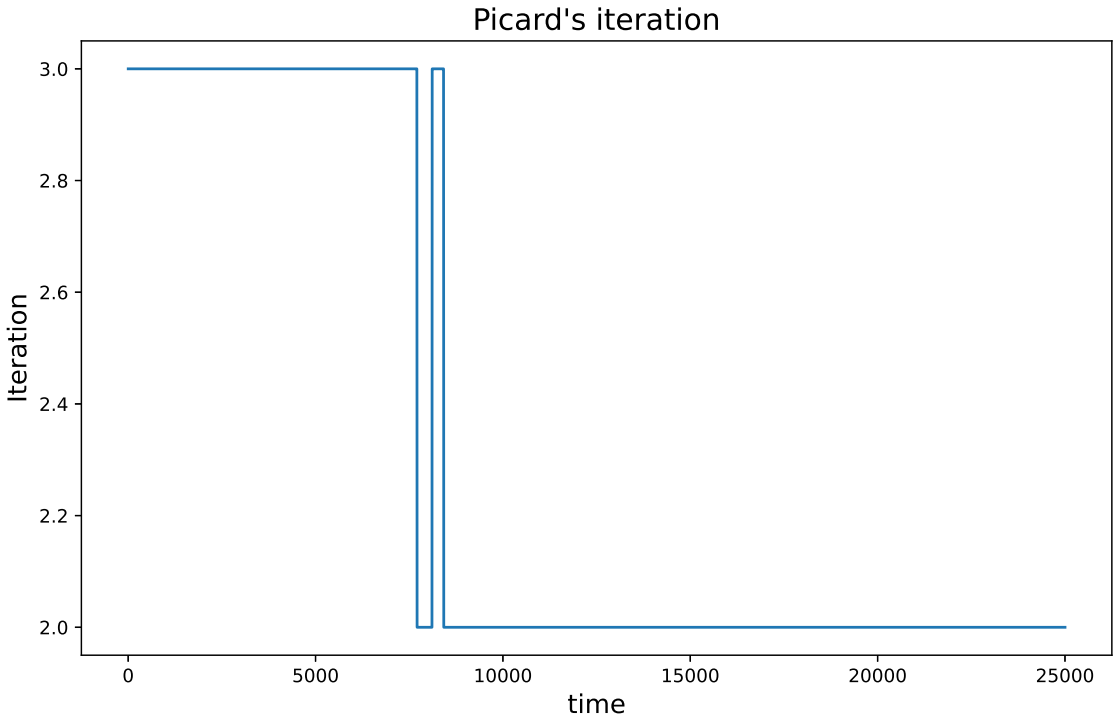


Figure 15: Picard's Iteration

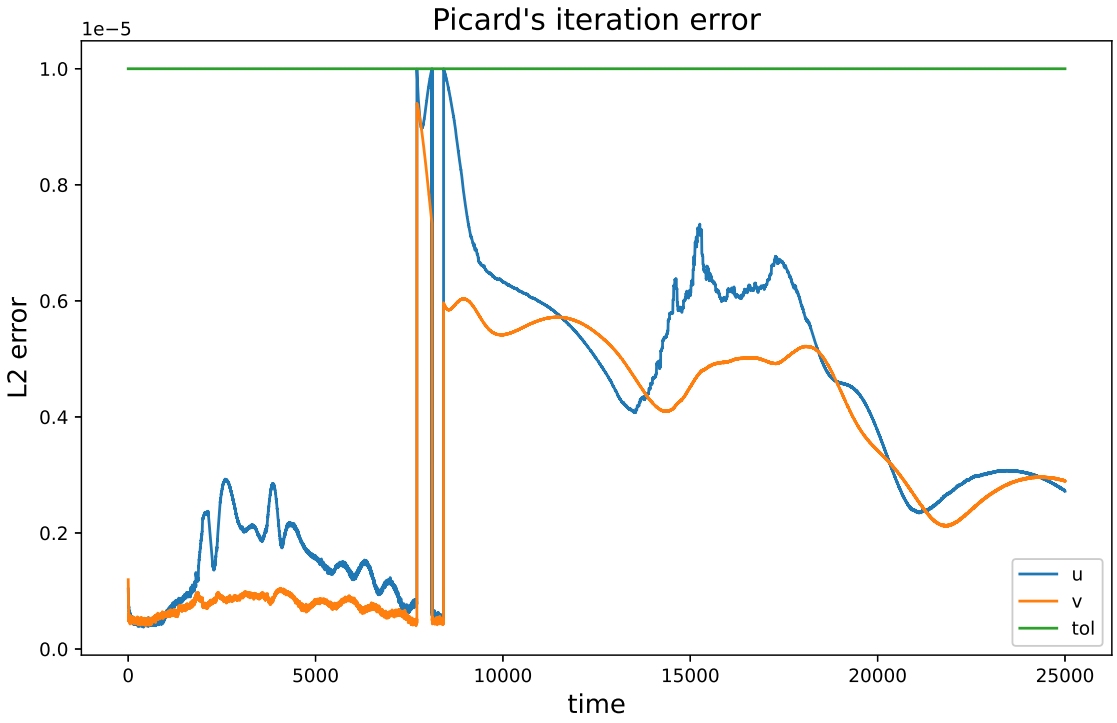


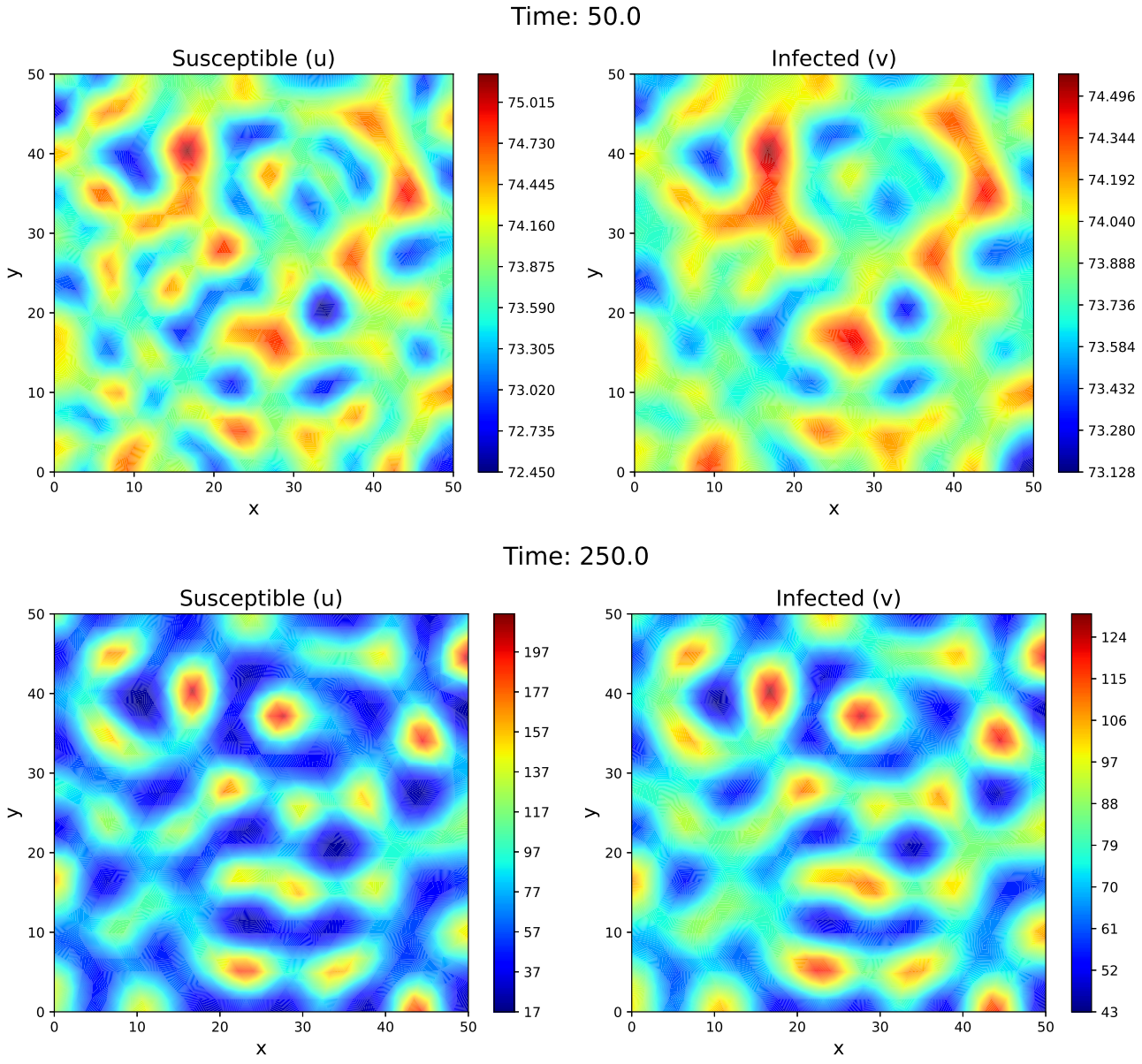
Figure 16: Picard's Iteration Error

7.1.7 Example 2B

1. **Model parameters:**
   $r = 0.27$, $K = 1000$, $\beta = 0.5$, $k = 0.25$, $(u^*, v^*) = (74.0741, 74.0741)$,
   Model = 'Model 2', Euler = 'Forward', $a_0 = 0.5$, $b_0 = 3$, $c_1 = 3u^*$, $c_0 = 0.02(u^*v^*(c_1 - u^* - v^*))^{-1}$,
   $m = -0.5$
   $\rho_u = \rho_v = 1$, $\delta_u(x, y) = \delta_v(x, y) \sim [0, 1]$, $L = W = 50$, mesh resolution = 0.0375,
   dt = 0.02, $t_0 = 0$, $t_\infty = 400$.

   Number of nodes: 919
   Number of elements: 1728

2. **Plots:**



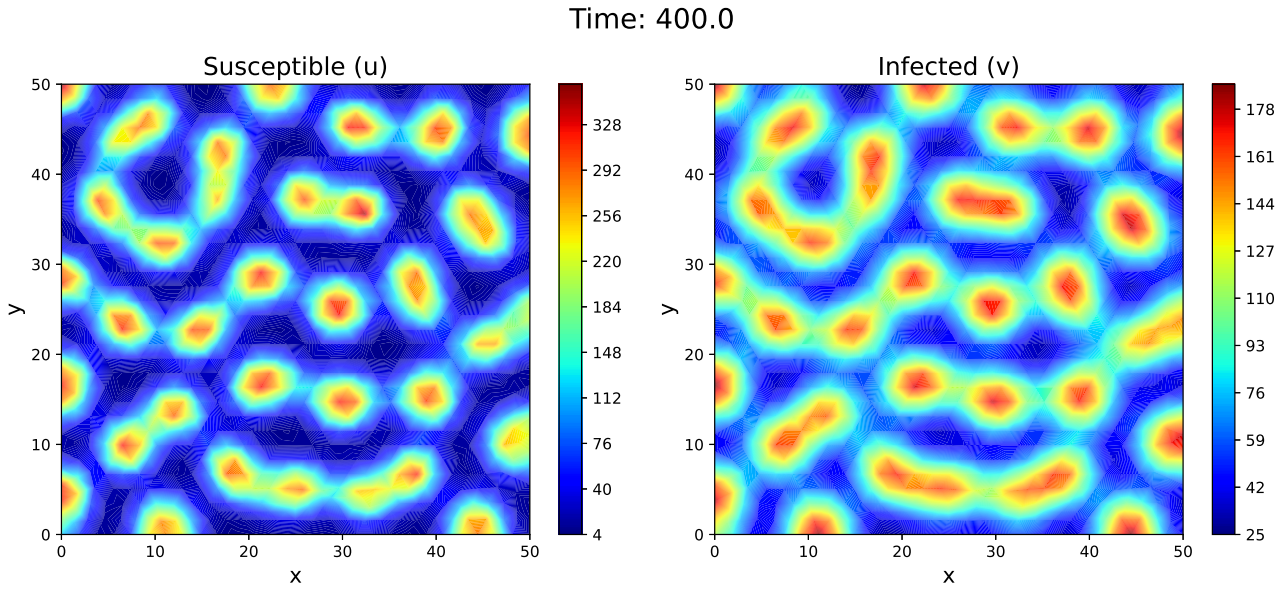


Figure 17: Example 2B Plots for Forward Euler

7.1.8 Inferences on Example 2 and comparison of Model 1 and 2

The result of a qualitative comparison between Schemes A and B for Model 2 is given in the above figures. For 2A, we have computed a numerical solution of a 2d problem using both forward and backward schemes with all the other parameters shown above. Even though the forward scheme is based on a higher number of discretizations of the time domain, one can see that the two solutions are almost indistinguishable. On the other hand, in terms of computational effort, forward is more efficient than backward. In this setting of 2A, we also observe the formation of spatial patterns. Notice, however, that in contrast with the results related to Model 1, here the "islands" of high concentration values of the susceptible species are surrounded by a layer of low concentration values. This behaviour is in accordance with the numerical simulation of cross-diffusion systems mentioned in [2].

When examining the profiles of Model 1 and Model 2, it becomes apparent that both exhibit the maximum value of species v at the same location as the maximum value of species u. The profile of the linear Model 1 appears sinusoidal, while the nonlinear Model 2 displays a more intricate pattern. Model 1 demonstrates species values spread across a wide range, while Model 2 maintains values closer to the equilibrium. However, adjusting parameters in Model 2 can readily modify this range due to the model's parametric flexibility, allowing for non-stationary solutions, as elaborated below.

To accurately estimate the scales of Model 1, the parameters for Model 2B undergo calibration based on the following guidelines:

1. The state of maximum avoidance $(\hat{u}, \hat{v})$ approximates the equilibrium (u^*, v^*) : $\hat{u} \approx u^*$, $\hat{v} \approx v^*$.

2. The coefficient c_0 is selected to ensure that the cross-diffusion coefficient at equilibrium, $c_0 uv(c_1 - u^* - v^*)$, surpasses the coefficient in the corresponding linear parametrization. This approach aims to maintain an average value for the nonlinear cross-diffusion coefficient comparable to the reference linear cross-diffusion coefficient.

The latter guideline can be implemented as follows: initially, ensuring that $c_1 \gg u^* + v^*$ guarantees that $c_1 - u^* + v^* \gg 0$ holds as a significant non-zero value. Subsequently, adjusting c_0 is necessary to achieve $c_0 uv(c_1 - u^* - v^*) \approx 0.02$, aligning Model 2 with the values observed in Model 1.

Consequently, in Example 2B (refer to Fig. 11), we select $a_0 = 0.5$, $b_0 = 3$, $c_1 = 3u^*$, $c_0 = 0.02(u^*v^*(c_1 - u^* - v^*))^{-1}$, and maintain the remaining parameters as in Example 1B. Analysis from Figure 17 reveals that the solution demonstrates a comparable scaling to that observed in Example 1A.

7.2 Real-Life infection dynamics

In this section, we depict a real-world scenario,

7.2.1 Nonzero Birth rate

7.2.1.1 Forward Euler

1. **Model parameters:**
   $r = 0.27$, $K = 1200$, $\beta = 0.5$, $k = 0.25$, $(u^*, v^*) = (950, 50)$,
   Model = 'Model 1', Euler = 'Forward', $a_0 = 0.1$, $b_0 = 2$, $c_0 = 0.02$,
   $\rho_u = 95$, $\rho_v = 5$, $\delta_u(x, y) = \delta_v(x, y) \sim [0, 1]$, $L = W = 200$,
   mesh resolution = 0.05, $dt = 0.05$, $t_0 = 0$, $t_\infty = 300$.

   Number of nodes: 515
   Number of elements: 948

2. **Plots:**

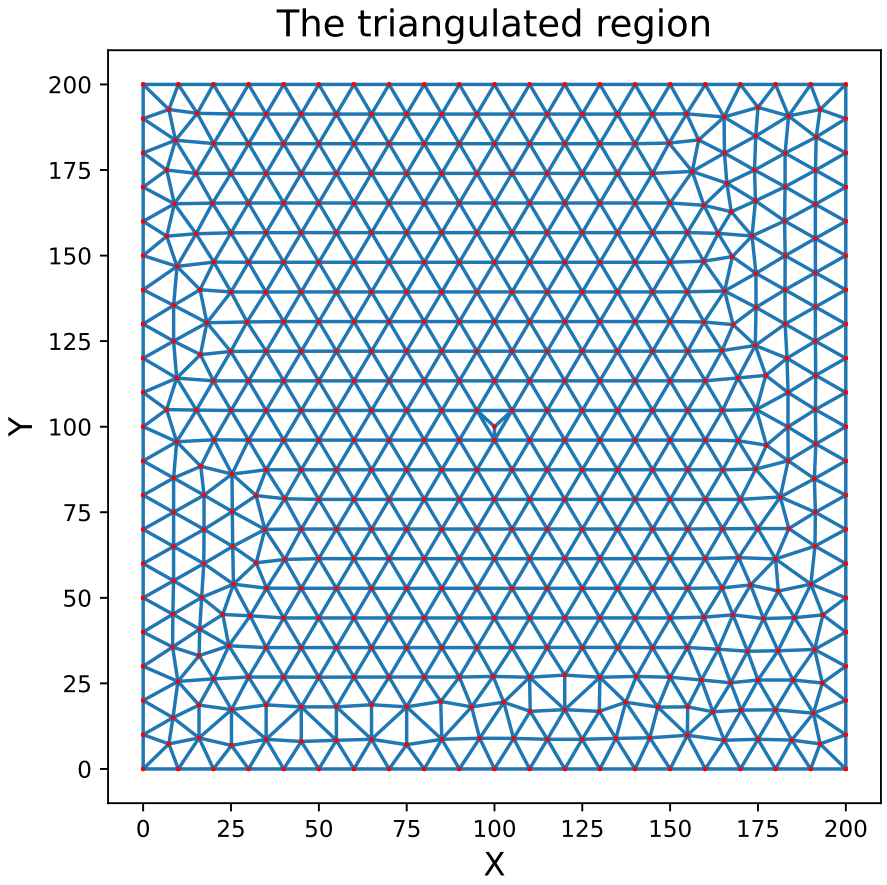
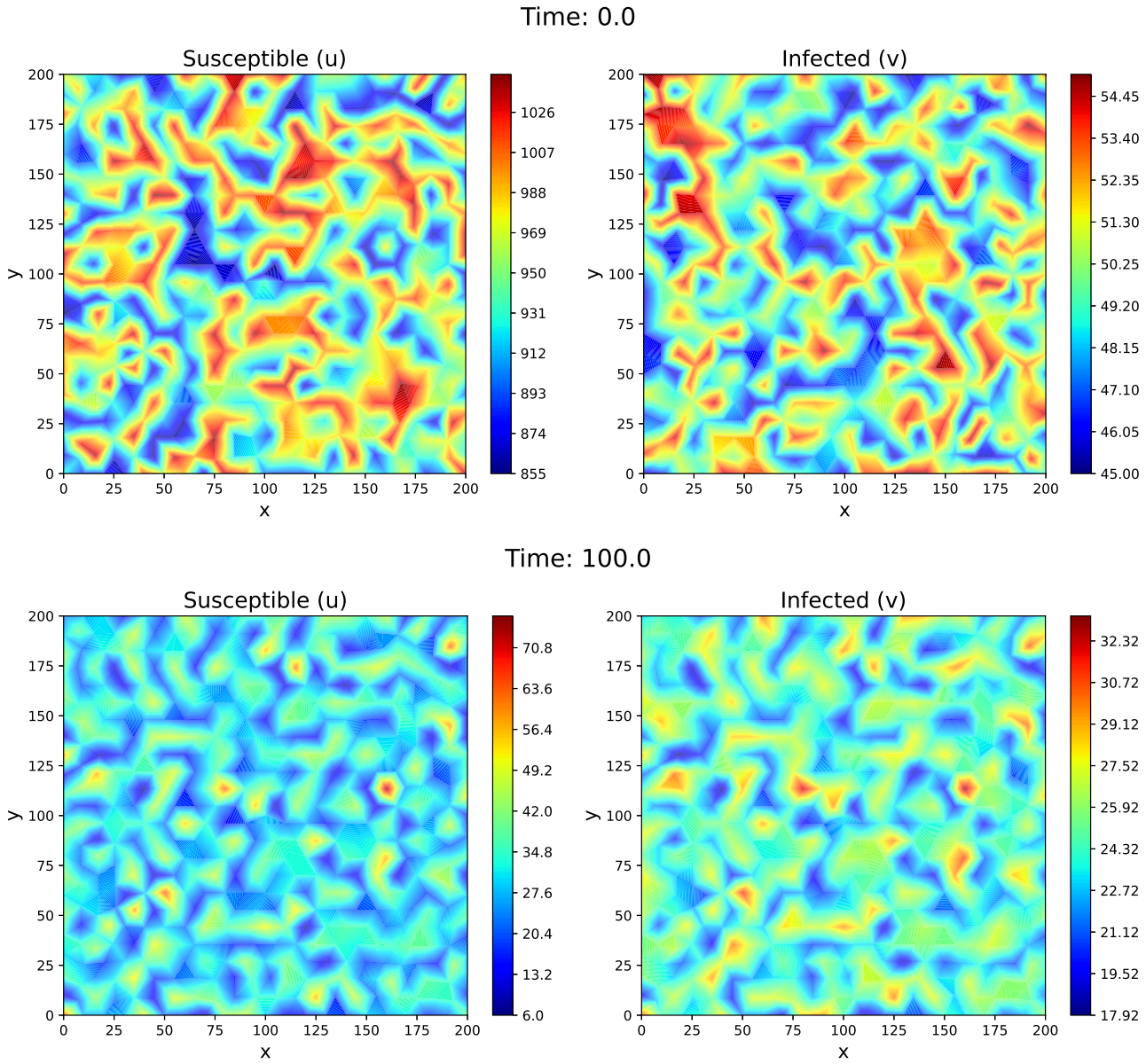


Figure 18: Triangulated Meshed Domain


Time: 0.0
Susceptible (u)
Infected (v)
Time: 100.0
Susceptible (u)
Infected (v)
x
y

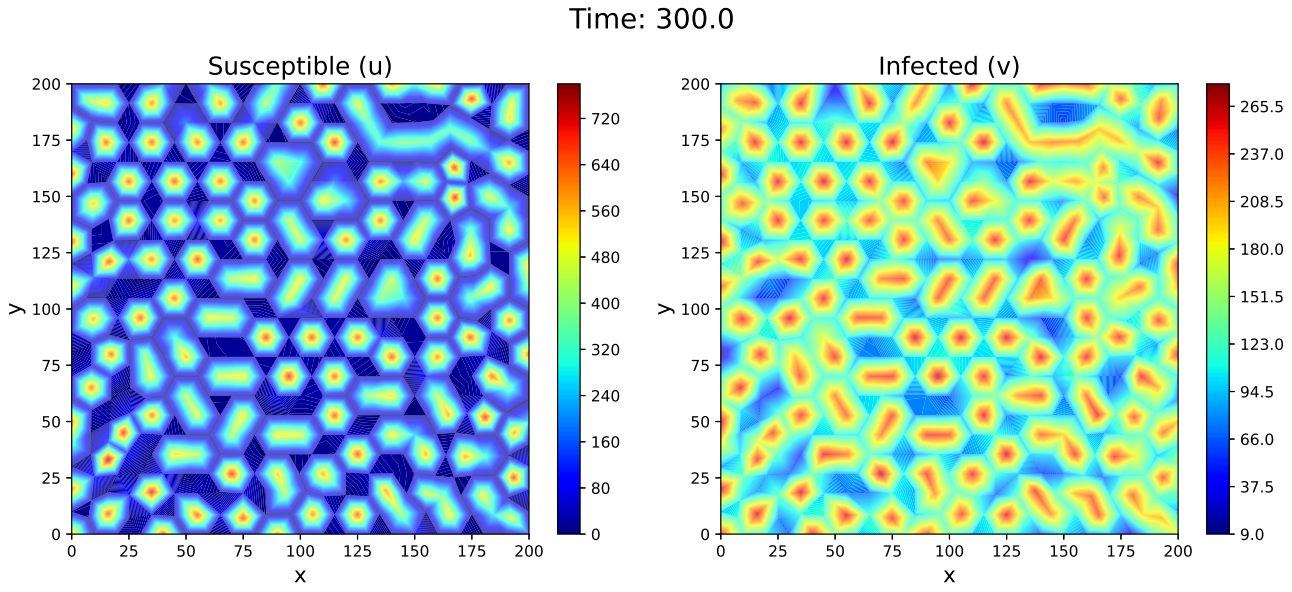


Figure 19: Plots for Forward Euler

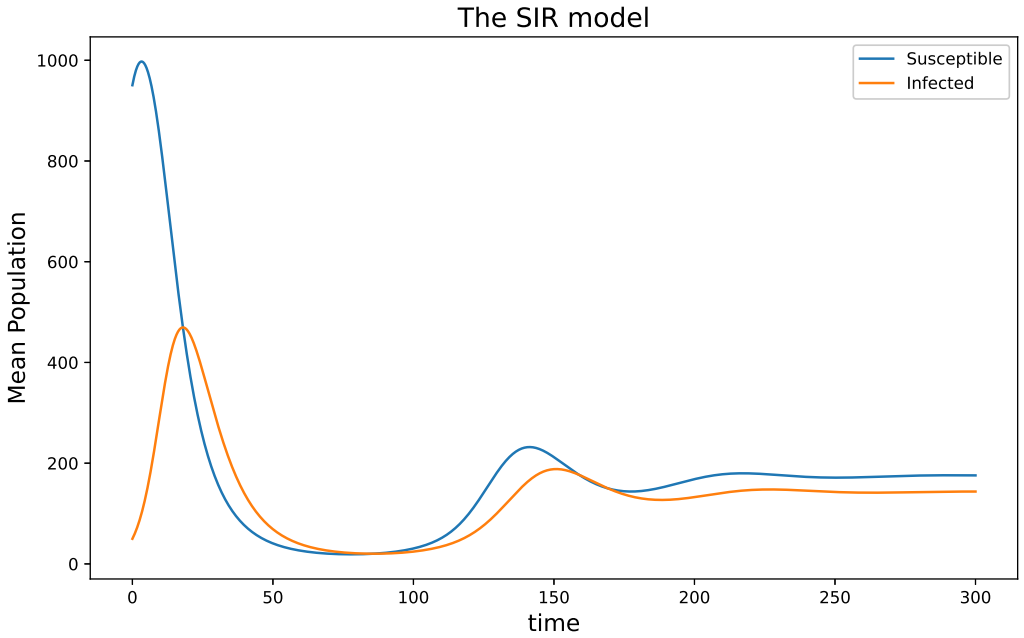


Figure 20: SIR Model (Mean Population)

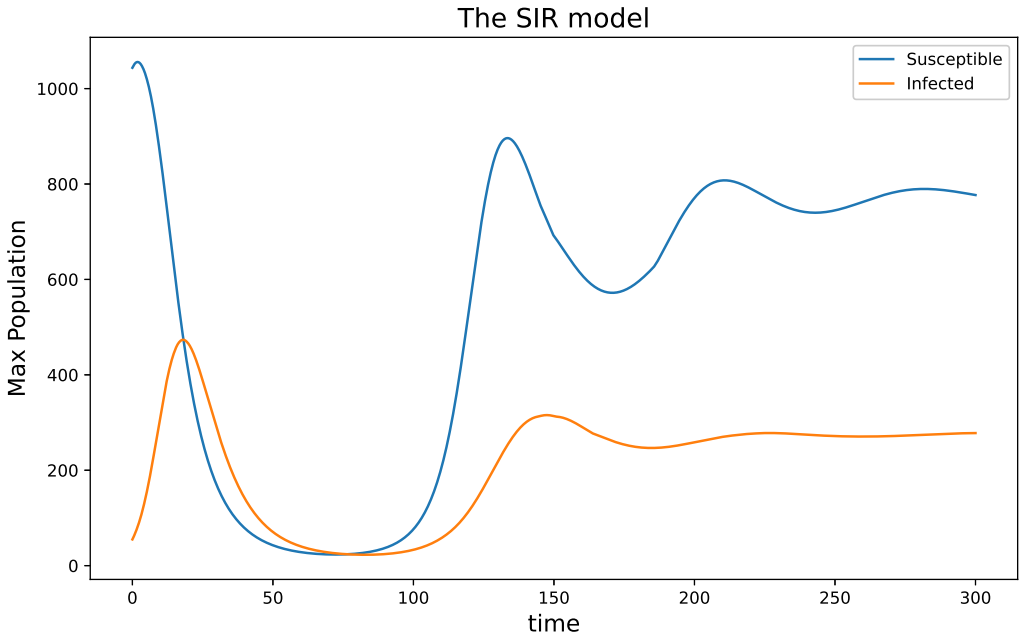


Figure 21: SIR Model (Max Population)

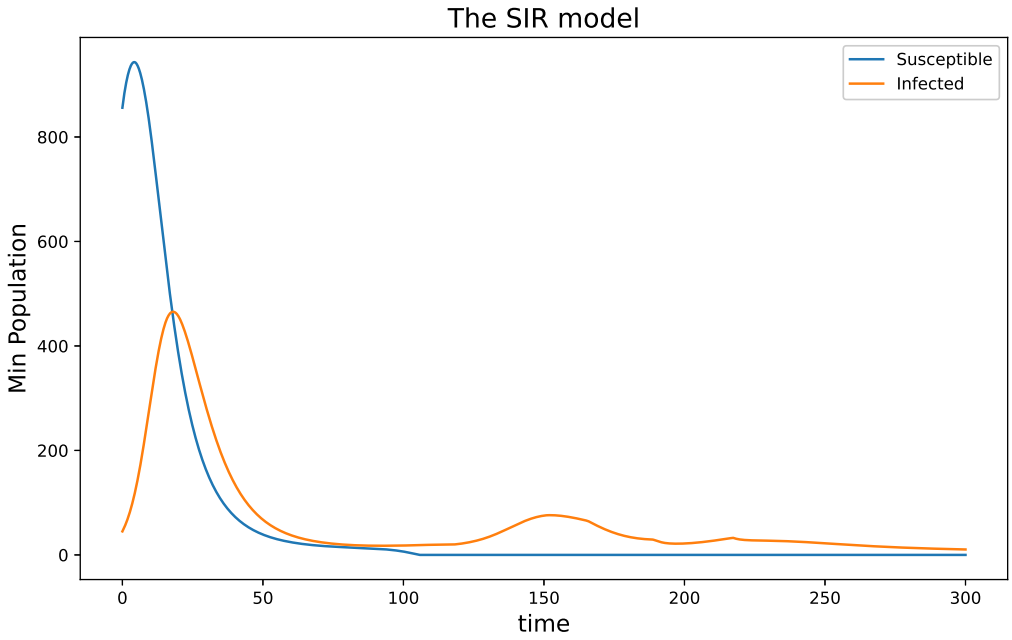


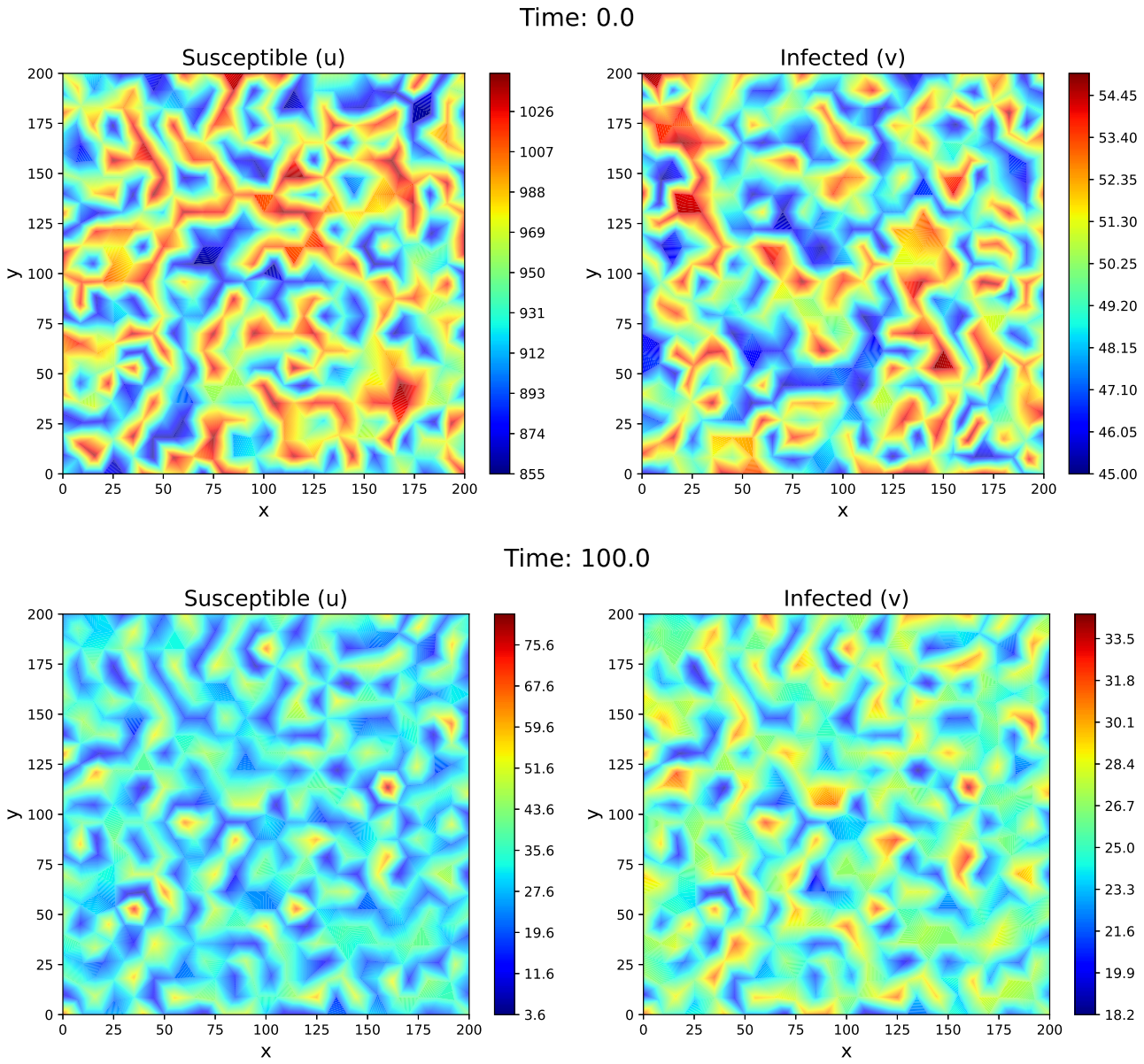
Figure 22: SIR Model (Min Population)

7.2.1.2 Backward Euler

1. **Model parameters:**

All the parameters are the same as that of Forward Euler, except Euler = 'Backward', dt = 0.2, tolerance $\epsilon = 1e - 5$.

2. **Plots:**



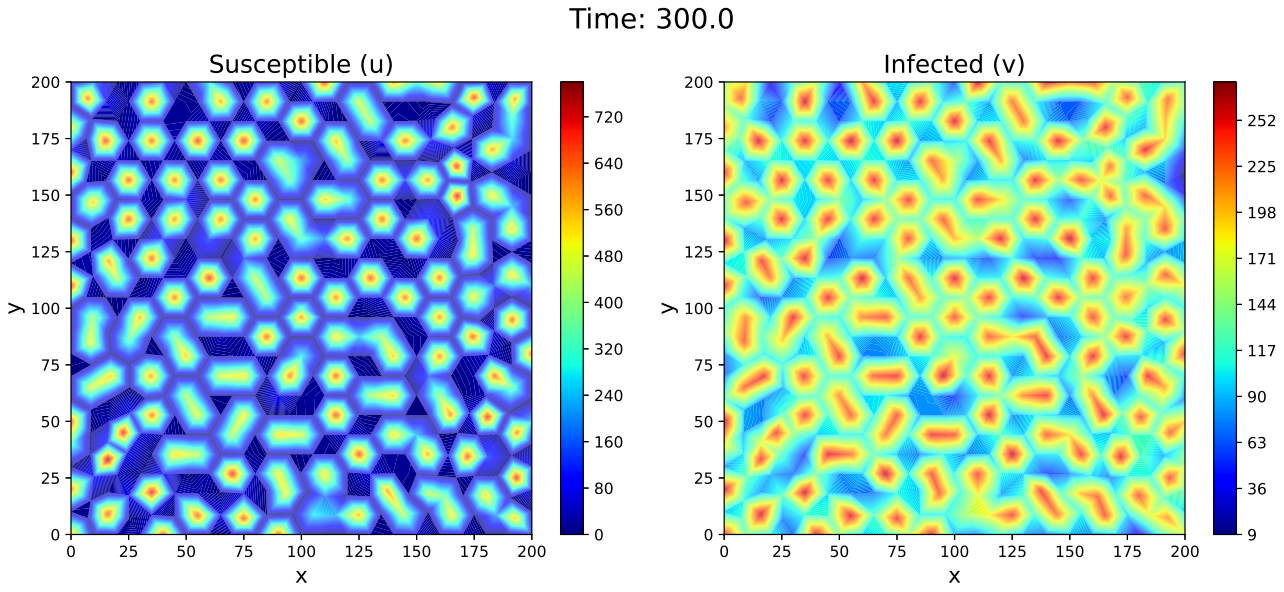


Figure 23: Plots for Backward Euler

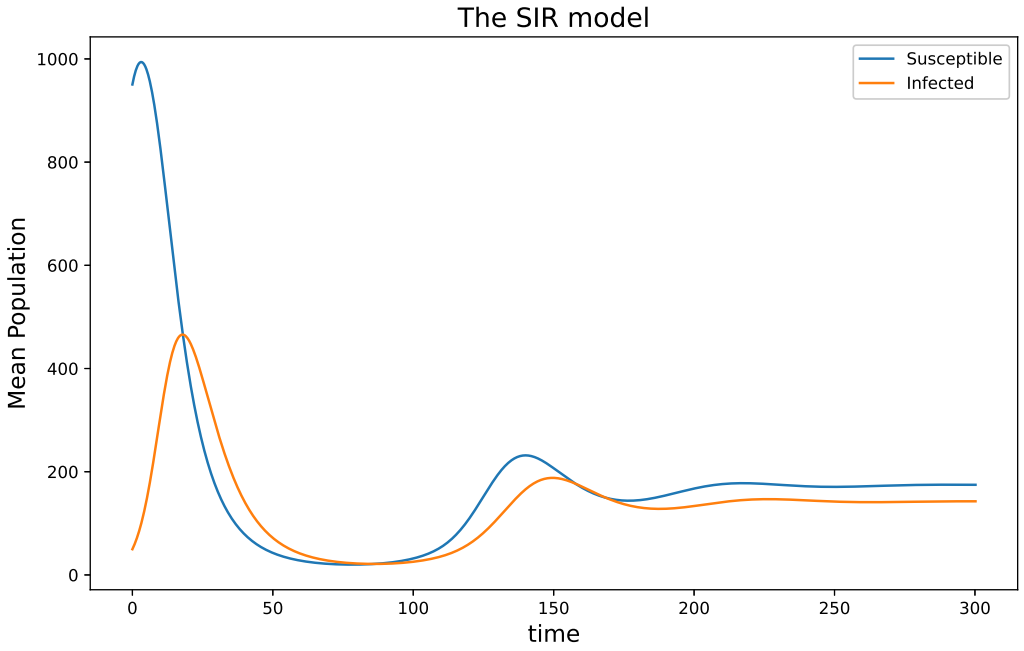


Figure 24: SIR Model (Mean Population)

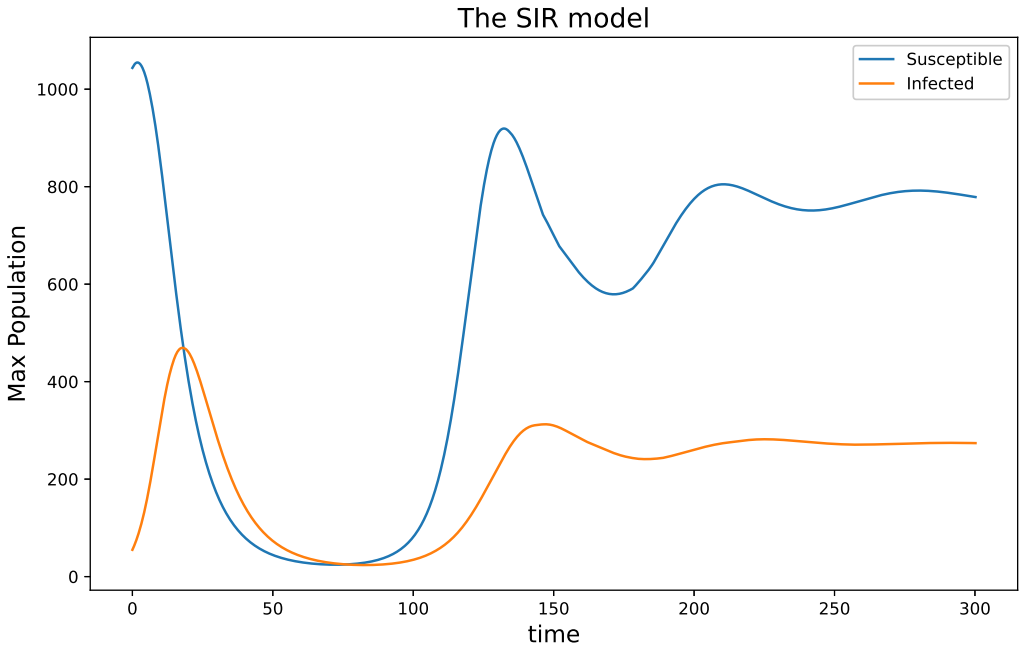


Figure 25: SIR Model (Max Population)

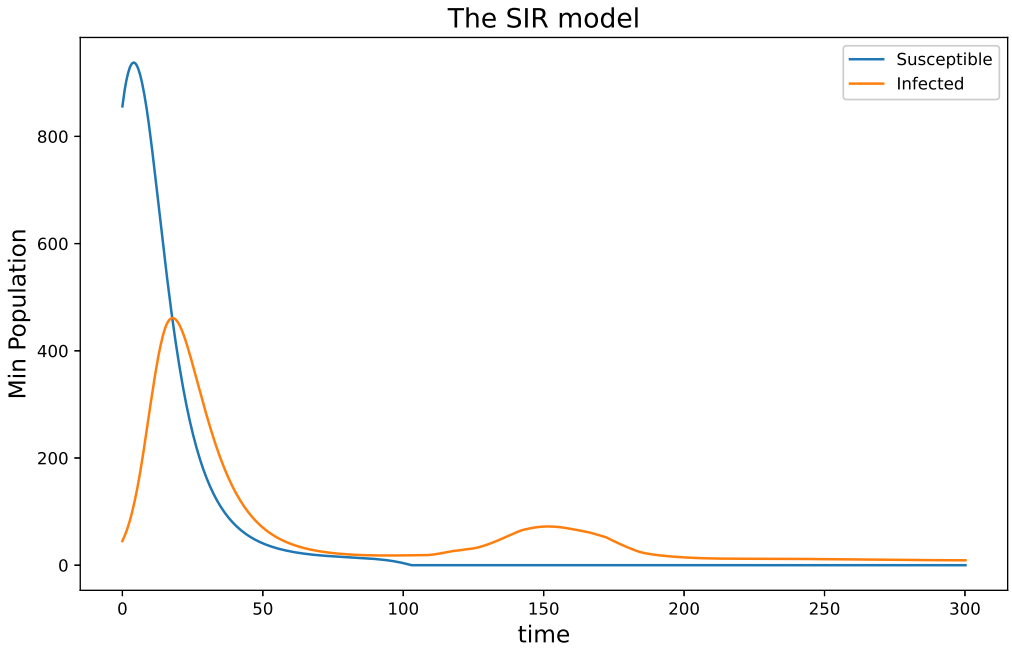


Figure 26: SIR Model (Min Population)

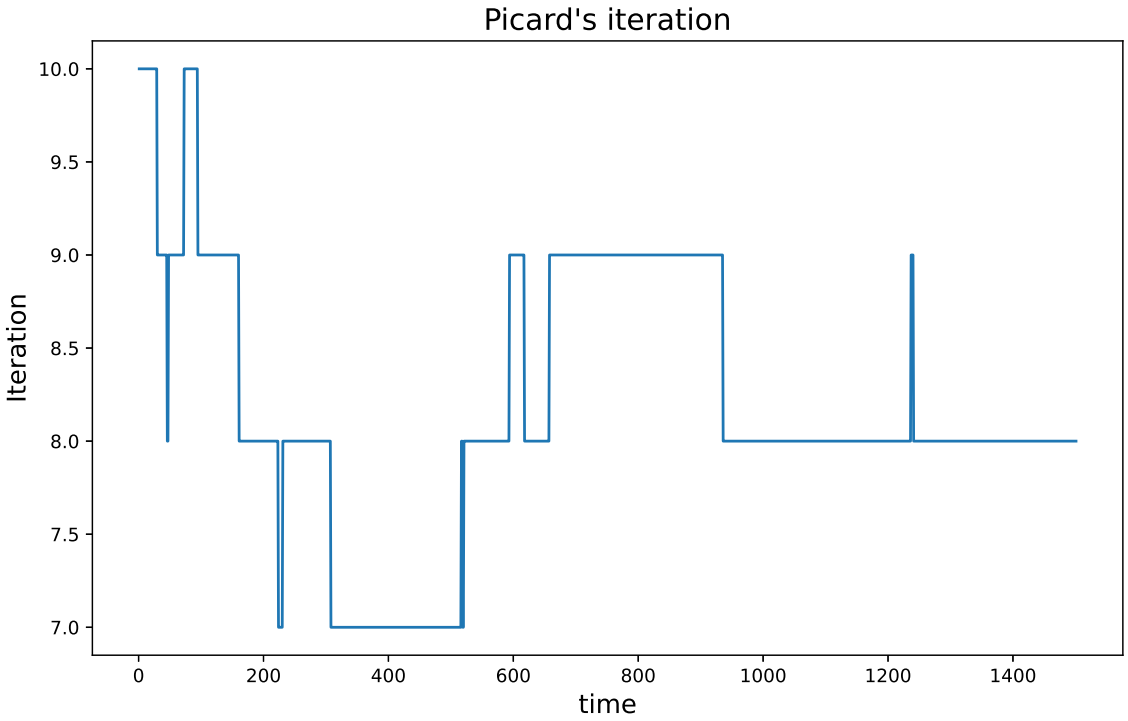


Figure 27: Picard's Iteration

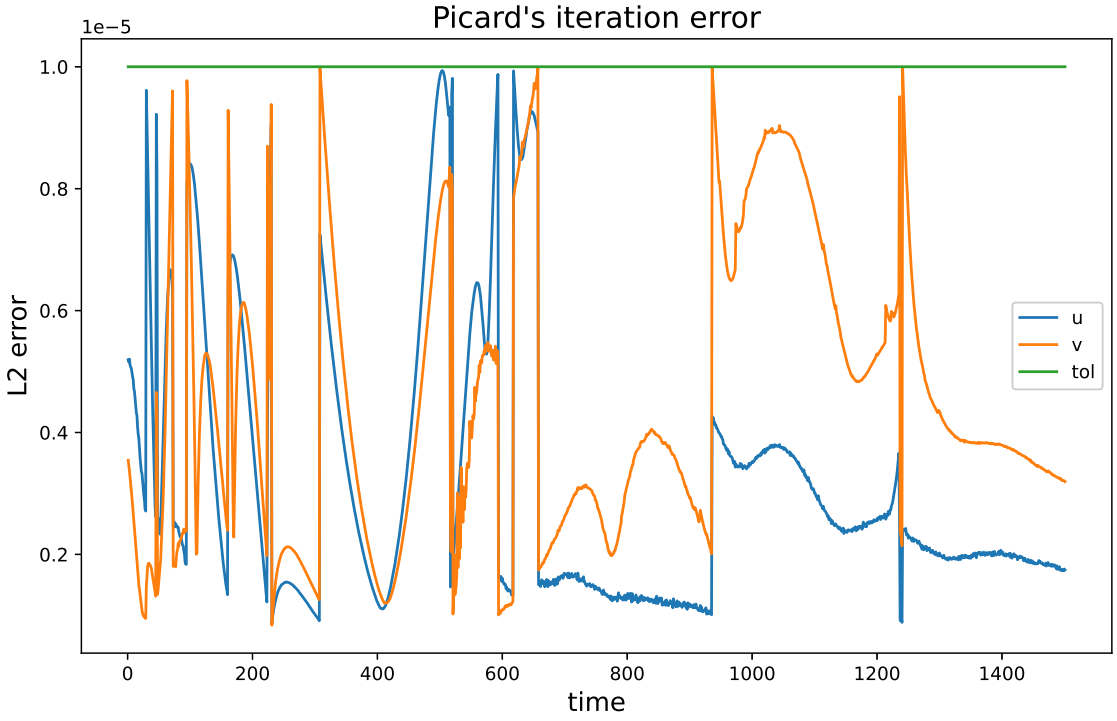


Figure 28: Picard's Iteration Error

7.2.2 Zero birth rate

1. **Model parameters:**
   All the parameters are the same as that of Forward Euler, except $r = 1e - 6$

2. **Plots:**

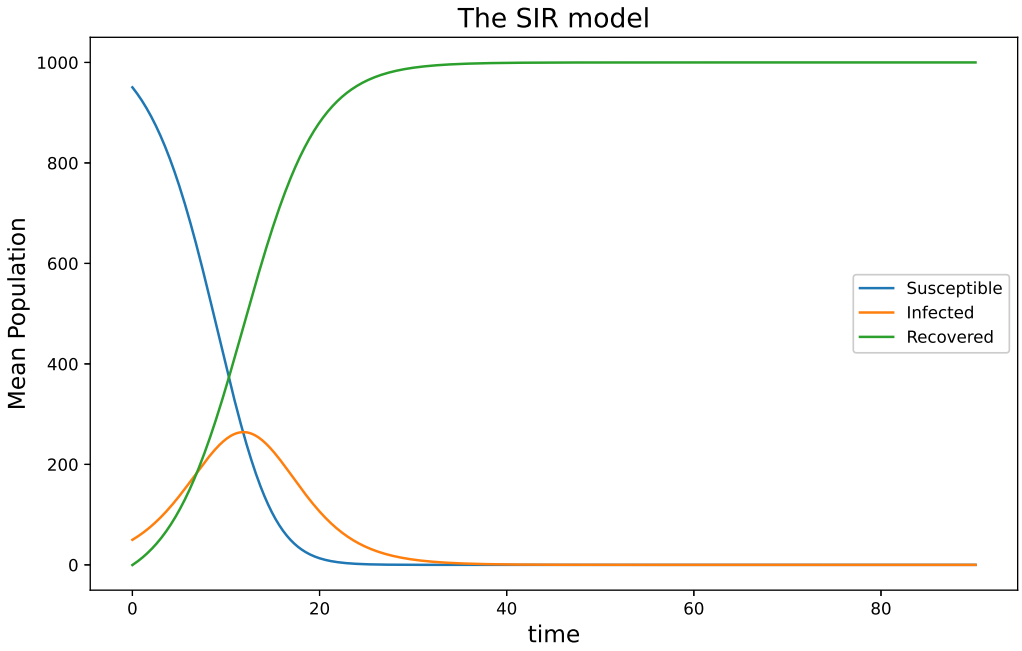


Figure 29: SIR Model (Mean Population)

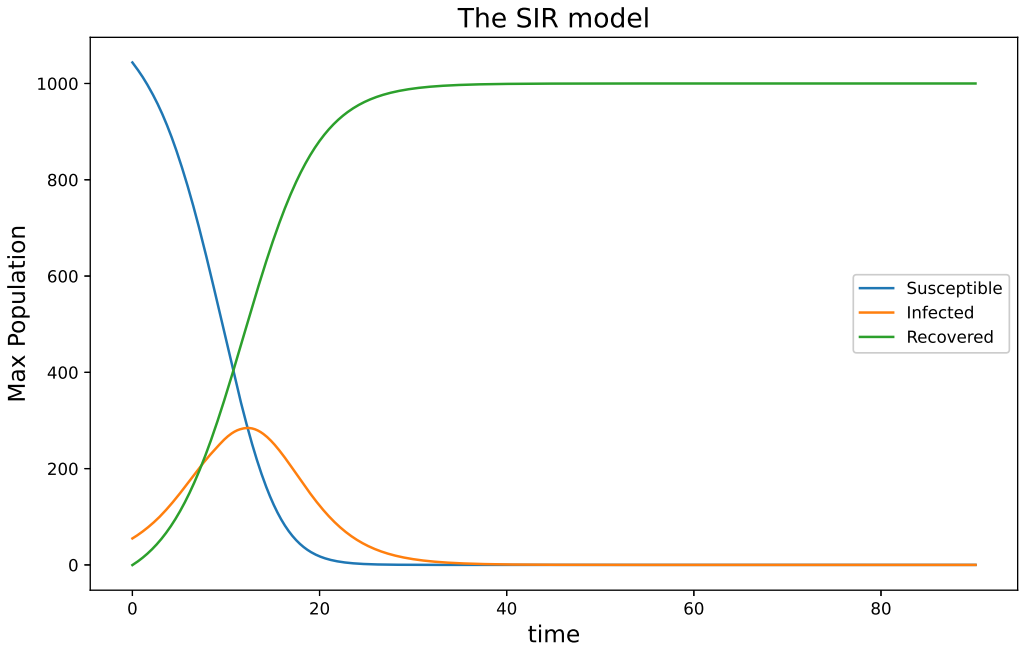


Figure 30: SIR Model (Max Population)

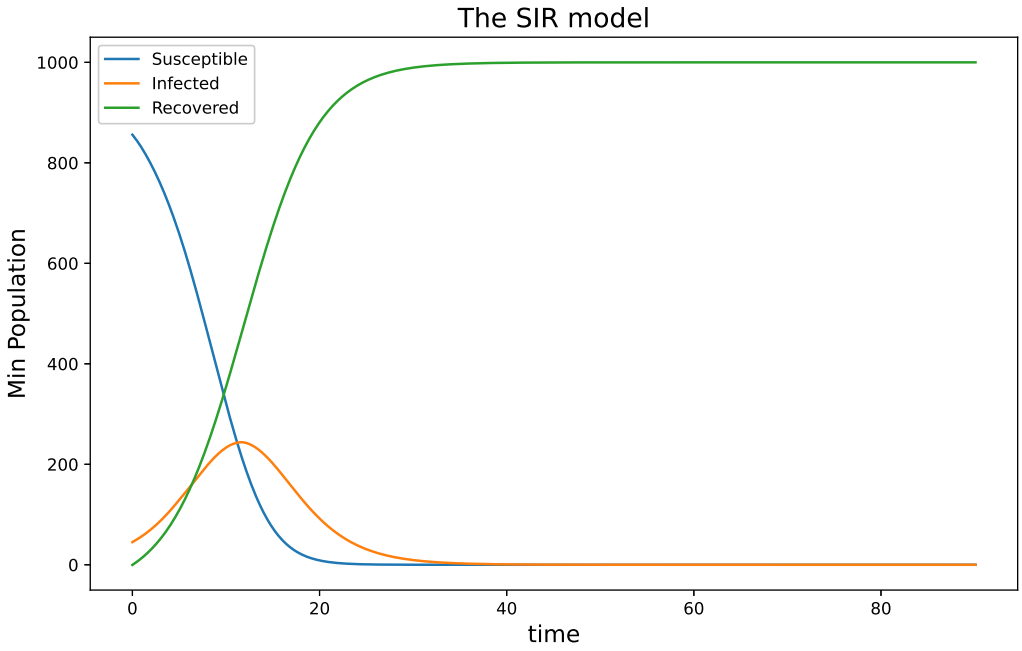


Figure 31: SIR Model (Min Population)

7.3 Inferences on real-life infection dynamics

This real-life infection dynamic model assumes the susceptible population to be 95% and the infected population as 5% of the total population. This model replicates Example 1A except that the initial

condition is different.

For the non-zero birth rate (20), u initially increases due to K[16] being more than 1000. We can observe that till $t = 90$, our graph exactly matches with 29 ($r = 0$). But, due to the non-zero intrinsic birth rate, we observe that the population of the susceptible and infected species starts to increase again. It oscillates for a small period of time with a smaller amplitude before achieving the equilibrium. The pattern formation in equilibrium (19) is very similar to that of Example 1A (2).

In addition, although the mean SIR model (29) oscillates and attains equilibrium, the population at each node oscillates between the max and min population as shown in (30) and (31) respectively.

For zero birth rate (29), u always decreases while v first increases and then decreases. Since, the total population is constant here, at equilibrium, susceptible and infected species tend to 0, while recovered species tend to be 100% of the population.

[16] carrying capacity

8 MESH CONVERGENCE

Note that, our governing differential equation is transient. Hence, we need to discuss convergence in space and convergence in time. For the convergence analysis, we used the real-life nonzero birth rate example.

8.1 Convergence in space

To study the mesh convergence in space, we fixed the equation at a particular time, viz. the equilibrium point. Hence we set the transient terms i.e. $\frac{\partial u}{\partial t}$ and $\frac{\partial v}{\partial t}$ to be zero.

The strong form equation 1 and 2 become:

$$f(u,v) + \nabla \cdot (a(u)\nabla u) + \nabla \cdot (c(u,v)\nabla v) = 0$$
$$g(u,v) + \nabla \cdot (b(v)\nabla v) = 0$$

The corresponding weak forms are as follows:

$$\int_\Omega \nabla w \cdot \Big(a\nabla u + c\nabla v \Big)\, d\Omega = \int_\Omega w f \, d\Omega$$
$$\int_\Omega \nabla w \cdot b\nabla v \, d\Omega = \int_\Omega w g \, d\Omega$$

But, we more care about the matrix form, which is represented as follows:

$$\mathbf{K_1} u + \mathbf{K_3} v = F$$
$$\mathbf{K_2} v = G$$

where, K_1, K_2, K_3 are the respective stiffness matrices, and F and G are the respective force vectors. In other words, the matrix representation of the static equilibrium state is given as follows:

$$\begin{bmatrix} \mathbf{K_1} & \mathbf{K_3} \\ \mathbf{0} & \mathbf{K_2} \end{bmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} F \\ G \end{pmatrix} \tag{25}$$

Here also, the $\mathbf{0}$ represents the null matrix of order $N \times N$, where N is the number of nodes. Here also, it can be observed that F and G are integrals over the domain Ω carrying the non-linear functional forms of u and v. Hence to compute u and v, we once again need Picard's iteration or Newton-Raphson method with some initial guesses of u and v.

The expression for Picard's iteration is given as follows:

$$\mathbf{K_1} \cdot u^{\theta+1} = -\mathbf{K_3} \cdot v^{\theta} + F^{\theta} \tag{26}$$
$$\mathbf{K_2} \cdot v^{\theta+1} = G^{\theta} \tag{27}$$

8.1.1 Convergence of the primary variable

8.1.1.1 Forward Euler

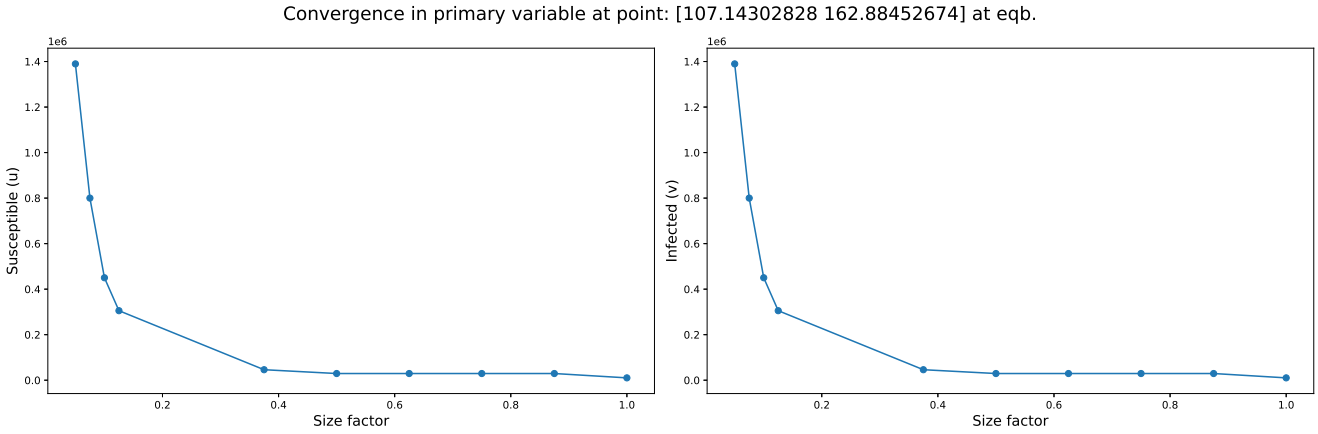


Figure 32: L^2 convergence (Forward)

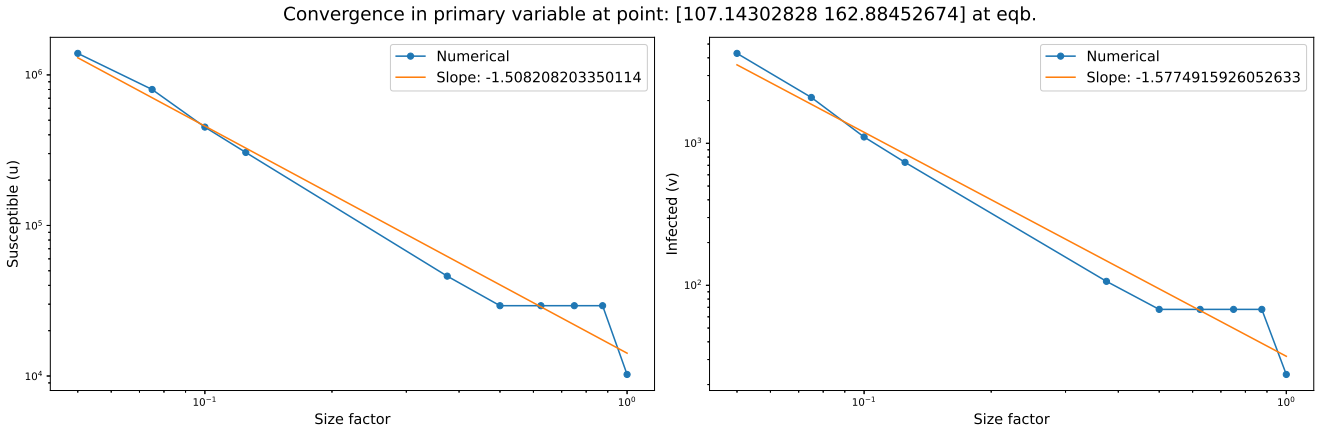


Figure 33: L^2 convergence (Forward) [log-log]

8.1.1.2 Backward Euler

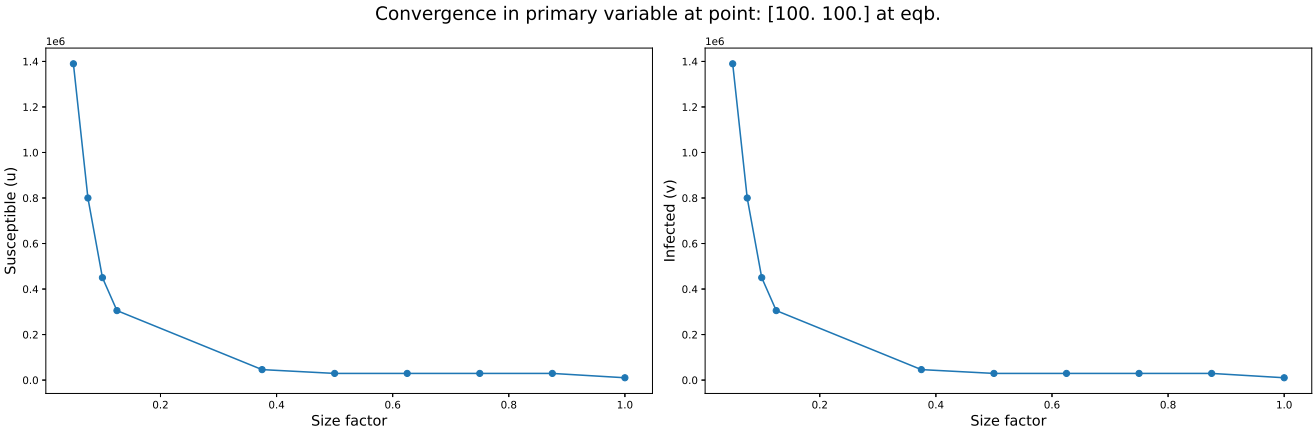


Figure 34: L^2 convergence (Backward)

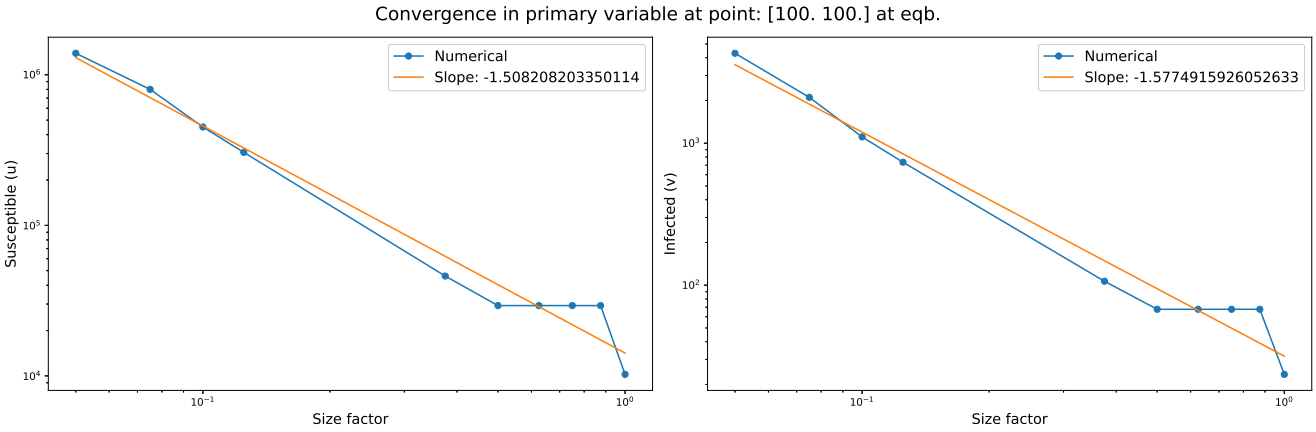


Figure 35: L^2 convergence (Backward) [log-log]

8.1.2 Convergence of the secondary variable

8.1.2.1 Forward Euler

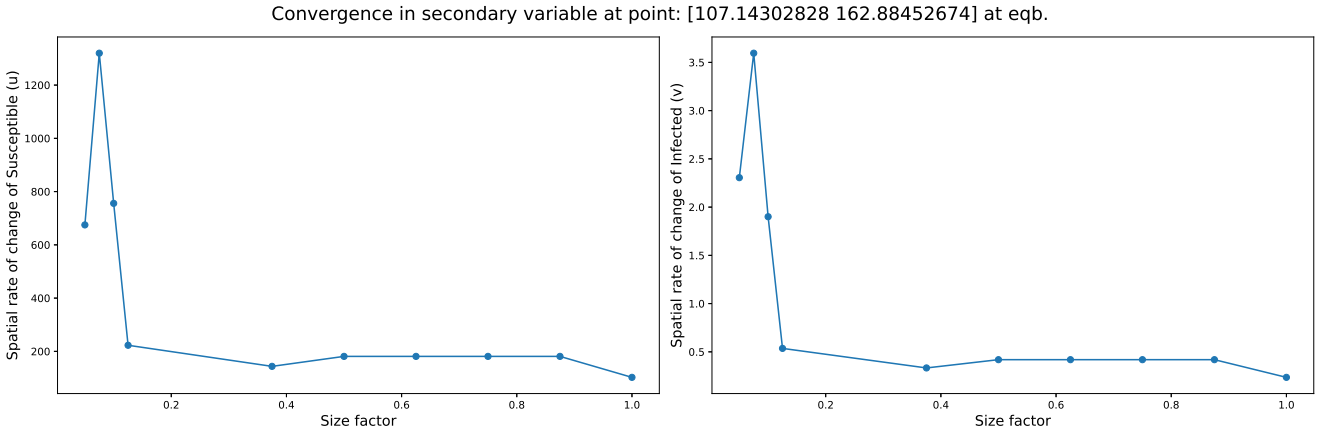


Figure 36: H^1 convergence (Forward)

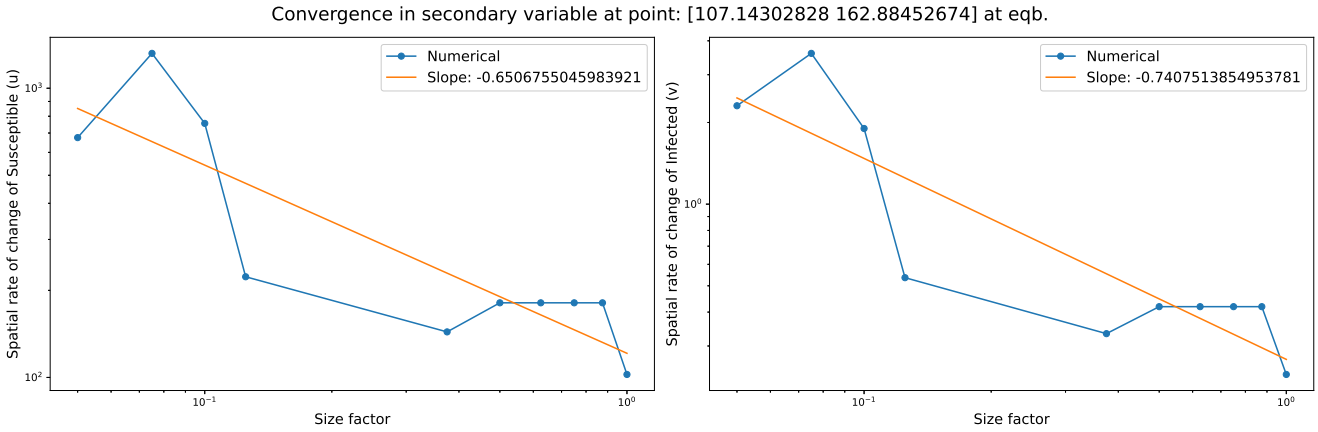


Figure 37: H^1 convergence (Forward) [log-log]

8.1.2.2 Backward Euler

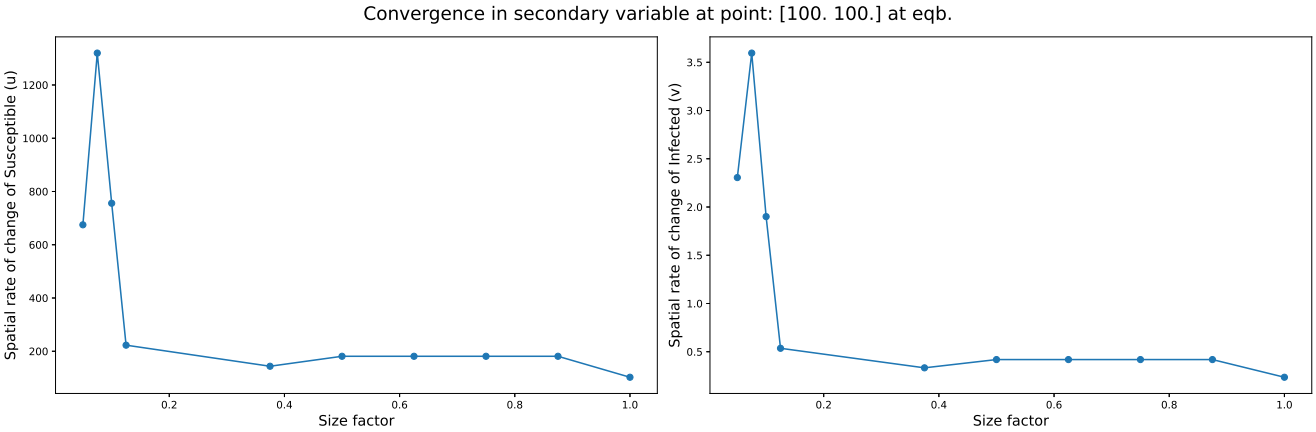


Figure 38: H^1 convergence (Backward)

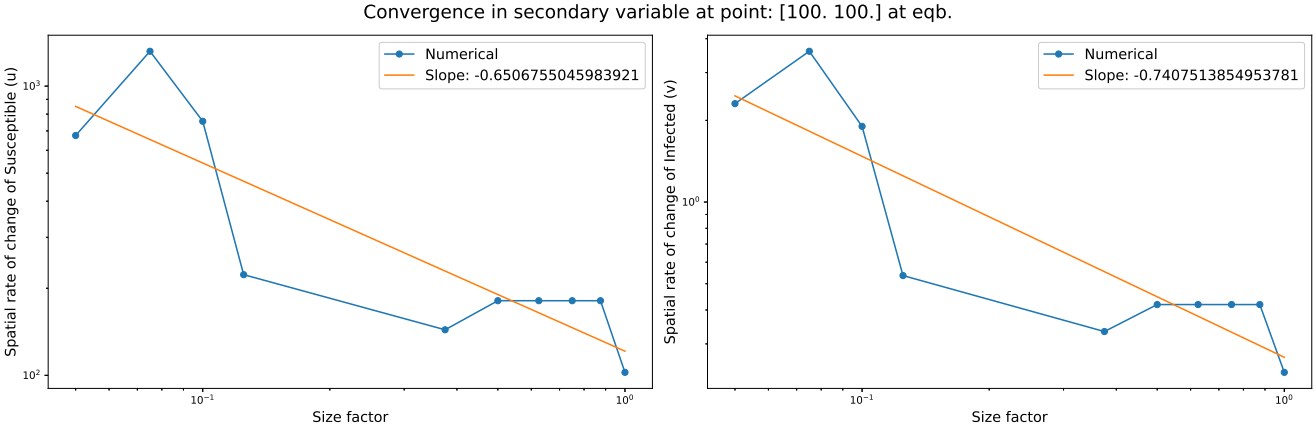


Figure 39: H^1 convergence (Backawrd) [log-log]

8.2 Convergence in time

For convergence in time, a particular mesh size, i.e. mesh resolution being 0.075, with the number of nodes 260 and the number of elements 462. We chose a random point in the mesh. And slowly increase the value of dt to capture the convergence in both forward and backward scheme.

8.2.1 Forward Euler

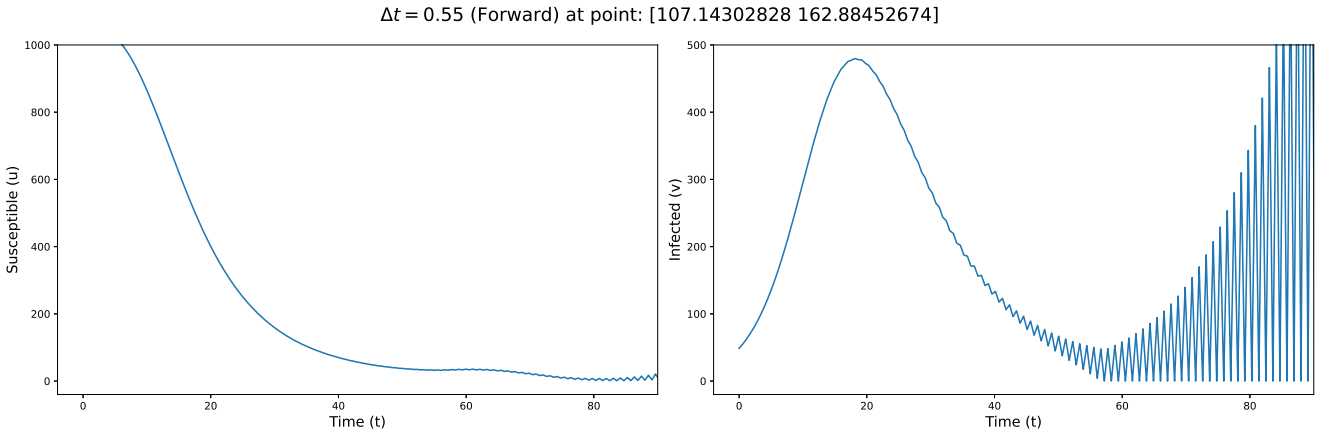


Figure 40: Convergence in time vs T for $dt = 0.55$ (diverges later)

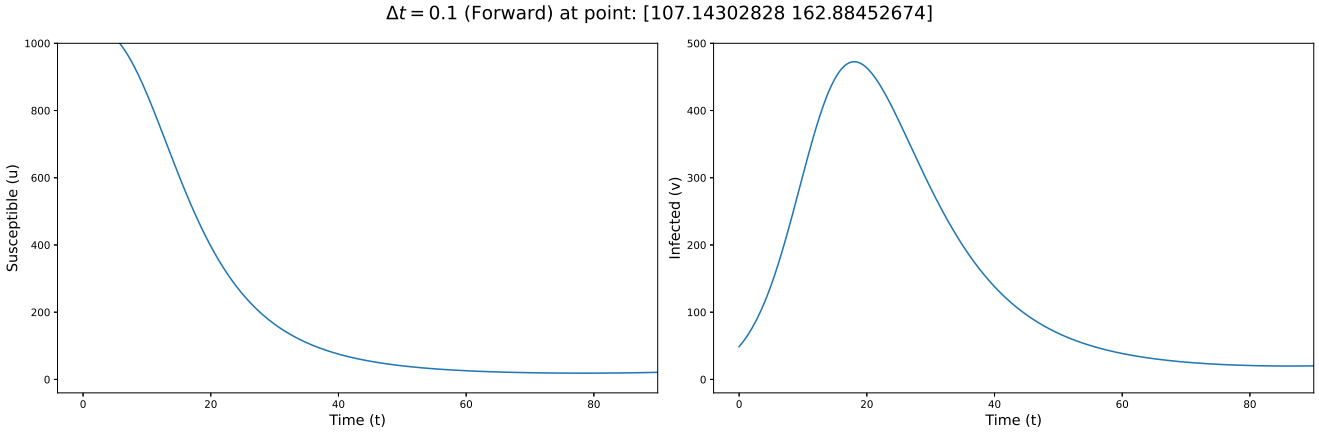


Figure 41: Convergence in time vs T for $dt = 0.1$

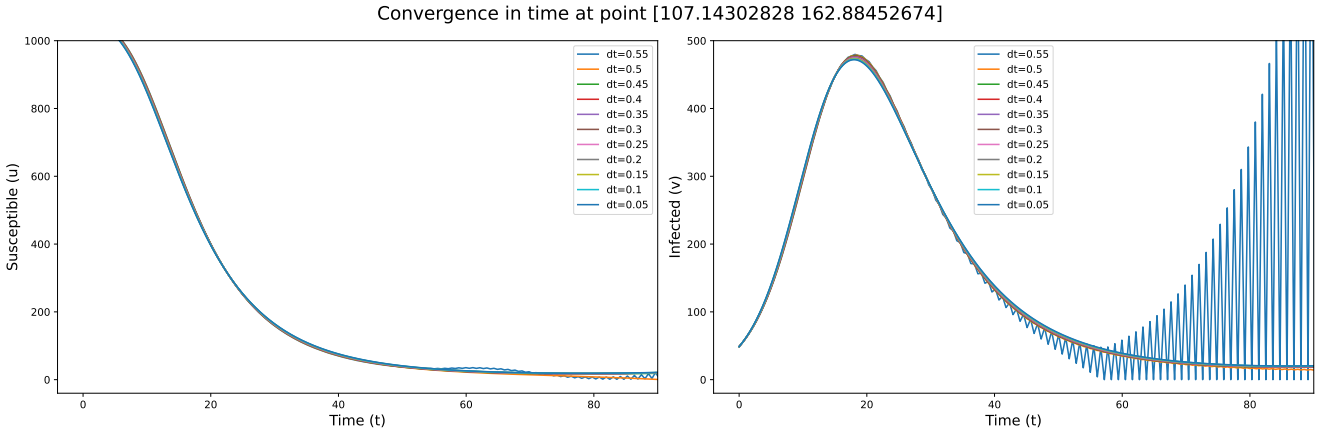


Figure 42: Convergence in time vs T for various dt

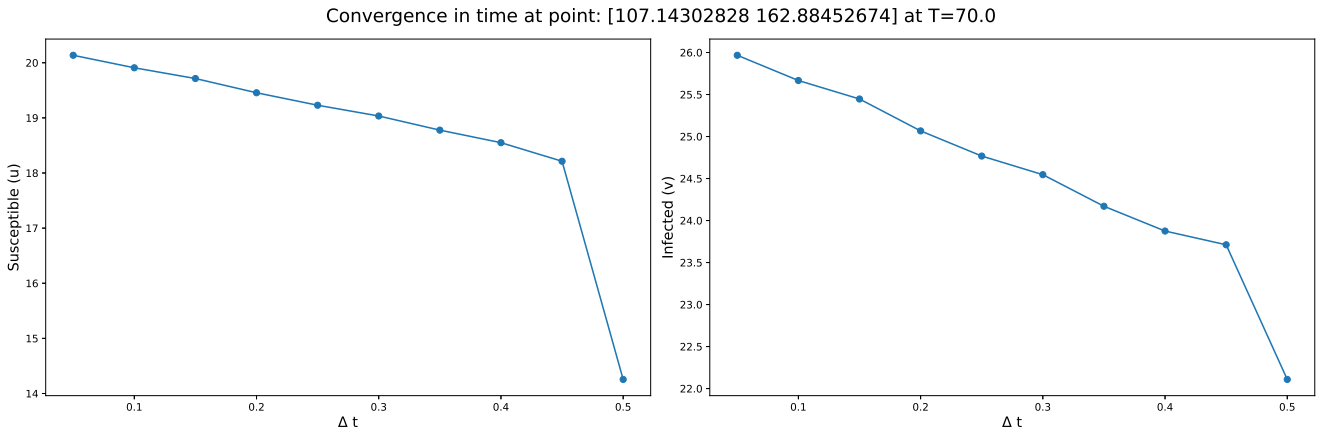


Figure 43: Convergence in time vs dt

8.2.2 Backward Euler

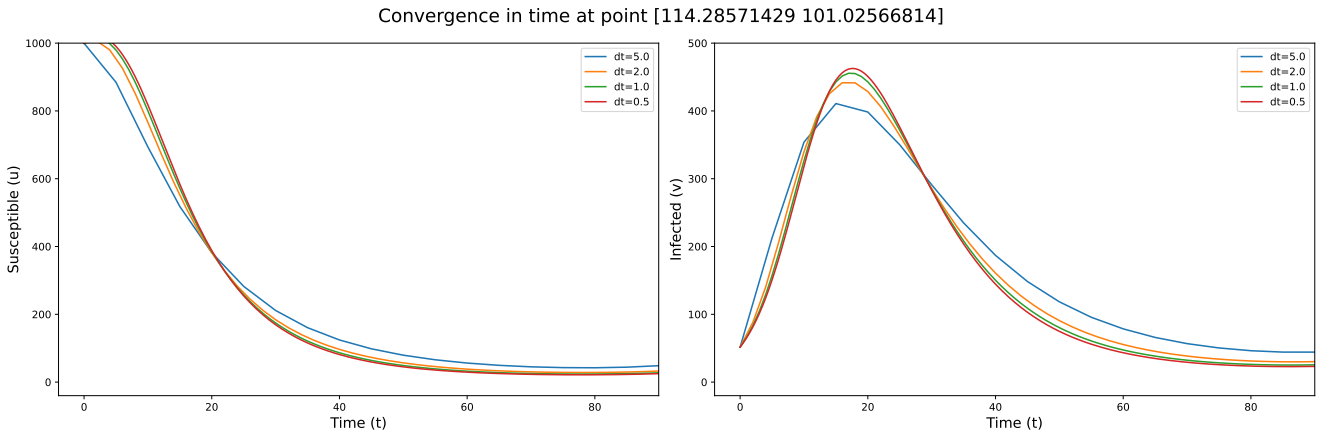


Figure 44: Convergence in time vs T for various dt

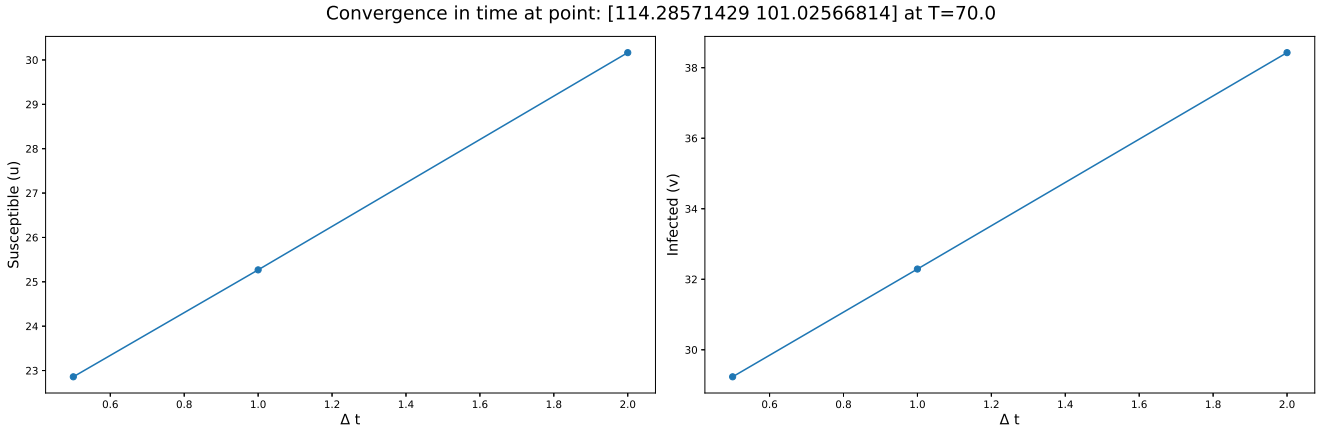


Figure 45: Convergence in time vs dt

8.2.3 Notes on forward and backward Euler

The main disadvantage of the forward model is it blew up even for $dt = 0.55$, which can be seen in the figure even though the calculations are simpler and do not involve techniques like Picard's iteration. For the values of dt which are in convergent region, the results are very similar to each other.

On the other hand, in backward, with the expense of a computation, we can model it for a bit coarse dt. The results vary for different values of dt but not too much. The good thing is they don't blow up even for $dt = 5$ unlike the forward.

9 CONCLUSION

9.1 Main Findings

In this report, we presented a numerical analysis of a two-dimensional epidemic model with nonlinear cross-diffusion, which captures the spatial dynamics of susceptible and infected populations. Using finite element method, we discretized the reaction-diffusion system and simulated both forward and backward Euler schemes. We also used Picard's iteration method to solve the nonlinear systems.

9.2 Significance of the results

Our results showed that the model can reproduce the emergence of complex spatial patterns, such as stripes, spots, and holes, influenced by the initial conditions and the model parameters. We also studied the effects of different factors, such as birth rate, carrying capacity, transmission rate, and recovery rate, on the evolution of the epidemic. Furthermore, we analyzed the stability and convergence of the numerical scheme both in time and space, providing relevant error estimates.

10 FUTURE SCOPES

The report has some limitations that can be addressed in future work.

1. The Newton-Raphson method mentioned in the report can be implemented because of its robustness (quadratic convergence), hence, converging much quicker than Picard's iteration.

2. The model assumes a homogeneous and isotropic diffusion, which may not be realistic for some scenarios.

3. The model does not consider the effects of spatial heterogeneity, such as environmental factors or population density, on the disease spread.

4. The model does not account for the possibility of reinfection or immunity. Future work can extend the model to incorporate these aspects and compare the results with real data.

11 STANDING ON THE SHOULDERS OF GIANTS

[1] S. Berres and J. Gonzalez-Marin. On epidemic models with nonlinear cross-diffusion. In *20th International Congress on Modelling and Simulation*, 2013.

[2] Stefan Berres and Ricardo Ruiz-Baier. A fully adaptive numerical approximation for a two-dimensional epidemic model with nonlinear cross-diffusion. *Nonlinear Analysis: Real World Applications*, 12:2888–2903, June 2011.

[3] Scott F Dowell. Seasonal variation in host susceptibility and cycles of certain infectious diseases. *Emerging infectious diseases*, 7(3):369, 2001.

[4] W.O. Kermack and A.G. McKendrick. A contribution to the mathematical theory of epidemics. *Proc. Roy. Soc. Lond. A*, 115(772):700–721, 1927.

[5] Alun L Lloyd and Robert M May. How viruses spread among computers and people. *Science*, 292(5520):1316–1317, 2001.

[6] Guiquan Sun, Zhen Jin, Qinxuan Liu, and Lili Li. Spatial pattern in an epidemic system with cross-diffusion of the susceptible. *Journal of Biological Systems*, 17(2):141–152, 2009.

12 APPENDIX

In this section, we discuss some extra results, that we regenrated/ played with.

12.1 The phase portrait[17]

Recall our strong form equation presented in equation 1 and 2 with the reaction terms given in equation 3 and 4.

The equilibrium points of these systems of non-linear transient differential equations are pairs (u, v) such that $f(u, v) = 0$ and $g(u, v) = 0$. For 3 and 4, the equilibrium points are $(0, 0)$ (trivial equilibrium point), $(K, 0)$[18], which corresponds to the disease-free point, and (u^*, v^*), which corresponds to an endemic stationary state that is explicitly given by:

$$(u^*, v^*) = \left(\frac{K(r - \beta + k)}{r}, \frac{K(r - \beta + k)(\beta - k)}{rk} \right) \tag{28}$$

This non-trivial equilibrium state is located in the first quadrant provided

$$k < \beta < k + r$$

The inequality $\beta > k$ implies that the transmission rate is greater than the recovery rate. Please refer to 3 and 4 to revisit the physical meaning of each of the terms in the above equation.

In figure 46, the phase portrait of the ODE system associated with equations 1, 2, 3, and 4 is presented. This system comprises equations 1 and 2, featuring the reaction terms $f(u, v)$ and $g(u, v)$, but **excluding the diffusion terms**. Various sets of initial data have been explored in this portrayal.

The system demonstrates asymptotic stability by converging towards the non-trivial equilibrium when initial data are selected in close proximity to it. This convergence to the equilibrium can be analytically proven by showcasing the behaviour of the Jacobian matrix, represented as follows:

$$F^* = F(u^*, v^*) = \begin{pmatrix} \frac{\partial f}{\partial u}(u^*, v^*) & \frac{\partial f}{\partial v}(u^*, v^*) \\ \frac{\partial g}{\partial u}(u^*, v^*) & \frac{\partial g}{\partial v}(u^*, v^*) \end{pmatrix}$$

which eigen values λ_1 and λ_2 with negative real parts. The Jacobian matrix corresponding to this system (i.e. without diffusion terms) in general can be represented as follows:

$$F(u, v) = \begin{pmatrix} r\left(1 - \frac{2u}{K}\right) - \frac{\beta v^2}{(u + v)^2} & -\frac{\beta u^2}{(u + v)^2} \\ \frac{\beta v^2}{(u + v)^2} & \frac{\beta u^2}{(u + v)^2} - k \end{pmatrix}$$

The above expression, at the endemic state (u^*, v^*) evaluates to the following expression:

$$F(u, v) = \begin{pmatrix} \beta - r - \frac{k^2}{\beta} & -\frac{\beta u^2}{(u + v)^2} \\ \frac{\beta v^2}{(u + v)^2} & \frac{\beta u^2}{(u + v)^2} - k \end{pmatrix}_{(u^\theta, v^\theta)}$$

[17] The phase portrait of the system without diffusion terms

[18] Recall K is the carrying capacity of the system

which turns out to be independent of K, the carrying capacity. This equilibrium point is stable if $\beta < r+k$.

On the other hand, the point $(K, 0)$ corresponds to a saddle point as the corresponding Jacobian has eigenvalues $\lambda_1 = -r < 0 < \beta - k = \lambda_2$.

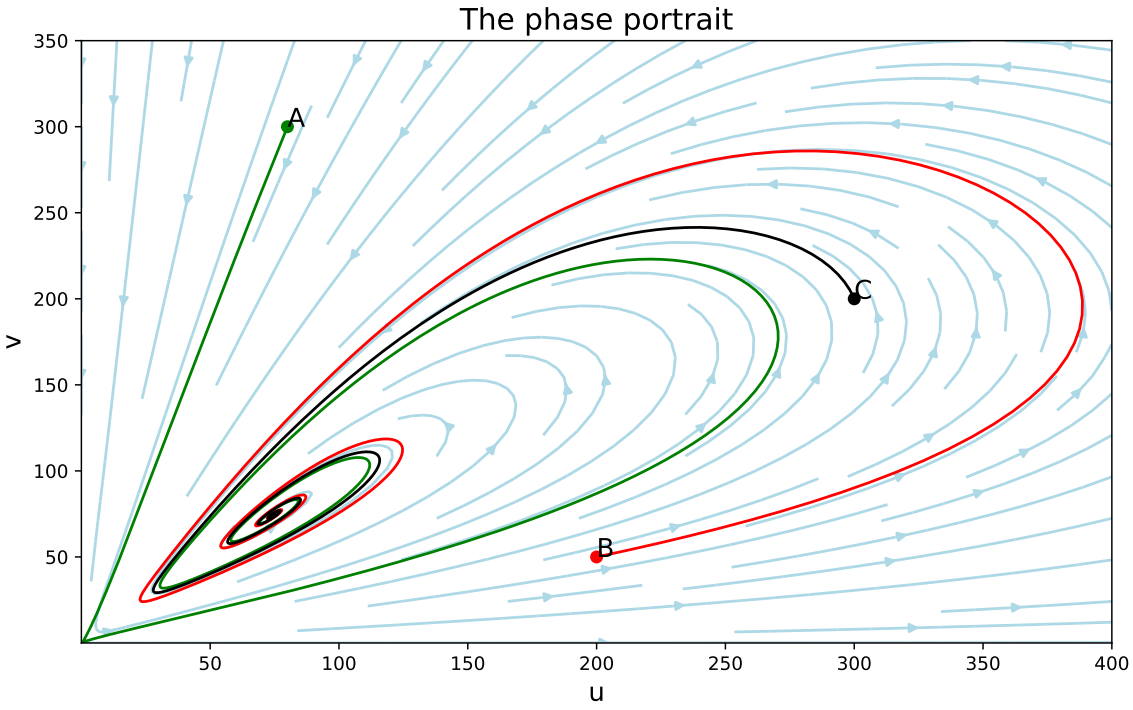


Figure 46: Phase portrait for the ODE system associated to 1, 2, 3 and 4. Three trajectories are displayed starting from the states $A = (80, 300)$, $B = (200, 50)$ and $C = (300, 200)$ and reaching the equilibrium point $(u^*, v^*) = (74.074, 74.074)$. The parameters are $(r, K, \beta, k) = (0.27, 1000, 0.5, 0.25)$.

12.2 Contribution and Acknowledgements

We individually reproduced the identical results to minimize the chances of error, ensuring a robust outcome. The report was a joint effort, with each of us making an equal contribution, ensuring an equitable distribution of the workload and responsibilities.

This project demanded significant computational resources from our perspective, involving approximately 49 hours of CPU running time dedicated solely to generating these plots. This duration does not include the numerous trial-and-error phases, rectifying incorrect codes, and other iterations. We extend our sincere gratitude to Archish S, Vivekanand S, Amit Kumar Mallik, Narayana D, and Aditya D for generously contributing their systems to facilitate our simulations.

Furthermore, we express our heartfelt appreciation to Prof. Sundararajan Natarajan for guiding us through this incredibly enriching journey. We're immensely grateful for the invaluable learnings, enhancement of coding skills, and the profoundly positive experiences gained during this course.