

A Numerical Comparison of Petri Net and Ordinary Differential Equation SIR Component Models

Trevor Reckell¹, Beckett Sterner², Petar Jevtić¹, and Reggie Davidrajuh³

¹School of Mathematical and Statistical Sciences, Arizona State University, 901 S. Palm Walk, Tempe, AZ 85287-1804, USA

²School of Life Sciences, Arizona State University, Tempe, USA

³Faculty of Science and Technology, University of Stavanger, Stavanger, Norway

July 18, 2024

Abstract

Petri nets are a promising modeling framework for epidemiology, including the spread of disease across populations or within an individual. In particular, the Susceptible-Infectious-Recovered (SIR) compartment model is foundational for population epidemiological modeling and has been implemented in several prior Petri net studies. However, the SIR model is generally stated as a system of ordinary differential equations (ODEs) with continuous time and variables, while Petri nets are discrete event simulations. To our knowledge, no prior study has investigated the numerical equivalence of Petri net SIR models to the classical ODE formulation. We introduce crucial numerical techniques for implementing SIR models in the GPenSim package for Petri net simulations. We show that these techniques are critical for Petri net SIR models and show a relative root mean squared error of less than 1% compared to ODE simulations for biologically relevant parameter ranges. We conclude that Petri nets provide a valid framework for modeling SIR-type dynamics using biologically relevant parameter values, provided that the other PN structures we outline are also implemented.

keywords: Petri nets, Differential Equations, ODE, Epidemiology, Modeling, RRMSE, RMSE

1 Introduction

1.1 Prior Work

Petri nets (PNs), a discrete event mathematical formalism, have become an increasingly popular modeling tool in epidemiology, with the number of papers on the subject steadily increasing. From 1990-2000, there were 11 papers on the subject published; from 2000-2010, there were 107 papers published; from 2010-2020, there were 375 papers published; from 2020-June 2024, there have been already 317 papers published, as found through Google Scholar and Scopus aggregate filtered search results. Petri nets have the upside of potentially modeling diverse phenomena across different types of disease propagation and intriguing framework due to the ease of visualization and comprehension of the PN model for those without a strong background in mathematics. With PNs, it is also straightforward to alter logical statements to fit the dynamics of a disease with various standard PN structures. Furthermore, Petri nets can easily scale up and be made non-deterministic, a mathematically intensive process with ordinary differential equations. Not surprisingly, ordinary differential equation (ODE) compartment models, a cornerstone in numerous epidemiological modeling studies with over 16,000 published articles on the subject from 1990-2024, are used as the mathematical structure from which many biological PN models are derived. However, the traditional ODE models exist as a system of equations operating in continuous time with continuous population levels, while Petri nets operate through discrete event simulations with discrete token levels. Moreover, there is not a one-to-one relationship between ODEs and PNs. With many PNs in epidemiology being transformed from ODEs, the comparative dynamics are essential for the epidemiological community. To our knowledge, the numerical equivalence between Petri net-based models and their classical ODE counterparts has not been explored, nor has Petri nets' fundamental dynamical limitations been extensively scrutinized in this context.

In particular, several criteria and critical properties of Petri nets have yet to be adequately scrutinized to ensure that a PN model's mathematical, biological, and computer science properties are appropriate for epidemiology applications. From papers talking about bio-chemical interactions [1] with over 800 citations to textbooks on modeling systems biology [2] with over 1400 citations, and within no other paper on Petri nets in epidemiology that we could find, are the practical application issues

of rounding errors in discrete token space, discrete time steps versus continuous time, or Petri net firing dynamics in relation to biological dynamics appropriately addressed. A SIR derivative PN model is proposed [3] with a method for parameter fitting, another practical application issue that needs addressing. Unfortunately, the structure of the PN model in the paper means that the parameters in the PN model are set to represent concepts different from those in the corresponding ODE mode, and the method of parameter fitting does not find a global minimum for error. The theme of less than rigorous validation is continued in other works where the dynamics of a PN model are not proven across the range of the parameters, the parameters themselves have different fundamental meanings compared to the ODE models they are replicating, and the dynamics of the PN model happen in a way that is impractical for a biological system. Similarly, the fitting and forecasting abilities in the Petri nets compared against ODEs are unvalidated in [4], [5], [6], and [7].

The contribution of this work is in addressing the two crucial issues of rounding errors of variable arc weights and the PN time steps per time unit. These issues, which have not been discussed in all previously mentioned papers, nor in any other paper we could find on the subject, are addressed here in a way where the parameters are connected to their real life and ODE definition. This connection allows the two modeling techniques to be more directly compared, with practical implications for disease modeling applications. These issues, applied to a fundamental model we outline in this paper, will allow the publication of our code and subsequent use by us and others to apply Petri nets in epidemiology further. Before getting to all this, though, we establish the basics of Petri nets, the basics of an ODE model in epidemiology, PN setup options for corresponding ODEs, and software for PNs.

The paper is organized first by establishing some basics about PNs, how PNs are set up in relation to corresponding ODEs, and steps to ensure PN dynamics correspond with fundamental biological dynamics. Then we get into how we will be addressing the issues of rounding error and PN time steps per time unit.

One note before continuing on the software we will be using: a considerable variety of Petri net simulation and software tools are available, all with different advantages and disadvantages [8][9]. The popular ones like GPenSIM, Snoopy, CPNTools, and Spike can handle various Petri net types. Nevertheless, the software GPenSIM was chosen for this work due to its ability to modify PN systems for deep analysis easily, run variable arc weight PN models, and run them in MATLAB. Beyond MATLAB's ability to quickly transfer into running with C, C++, and Python programs, MATLAB has many prebuilt functions that allow for expanding the scope of GPenSIM to fit more applications in disease modeling in the future.

2 Methodology – Preliminaries

Understanding the basics of ODE and PN models is crucial, as these basics play a huge role in the overall dynamics of a model and, later, the mathematical analysis of the results.

2.1 Fundamentals of Petri Nets and ODE models

A Petri net graph, or Petri net structure, is a weighted bipartite graph [10] defined as n-tuple (P, T, A, w, x) where,

- P is the finite set of places (one type of node in the graph).
- T is the finite set of transitions (the other type of node in the graph).
- A is the set of arcs from places to transitions and from transitions to places in the graph $A \subset (P \times T) \cup (T \times P)$.
- M_0 is the initial state (*aka* marking), $M_0 = [m_1, m_2, \dots, m_n]$, where m_i is the number of tokens in place p_i .
- $w : A \rightarrow \{1, 2, 3, \dots\}$ is the weight function on the arcs.
- x is a marking of the set of places P ; $x = [x(p_1), x(p_2), \dots, x(p_n)] \in N^n$ is the row vector associated with x .

A transition $t_j \in T$ in a Petri net is said to be enabled if $x(p_i) \geq w(p_i, t_j)$ for all $p_i \in I(t_j)$. This allows us to define the state transition function, $f : N^n \times T \rightarrow N^n$, of Petri net (P, T, A, w, x) is defined for transition $t_j \in T$ if and only if $x(p_i) \geq w(p_i, t_j)$ for all $p_i \in I(t_j)$. If $f(x, t_j)$ is defined, then we set $x' = f(x, t_j)$, where $x'(p_i) = x(p_i) - w(p_i, t_j) + w(t_j, p_i), i = 1, \dots, n$. In simple terms, a transition is enabled if the number of tokens in all places connected to that transition via an incoming arc is greater than or equal to the arc weight for the respective arc connected to the transition.

Tokens are assigned to places, with the initial assignment being the initial marking. By definition, a Petri net can utilize topological analysis. It is important to note that the number of tokens assigned to a place is an arbitrary non-negative integer but does not necessarily have an upper bound.

In Section 3.2, we go into more detail about the idea of PN time steps per unit time, but here we define them as the number of firings each transition is allowed per one unit of time, defined as τ . The graph then can be defined as (P, T, A, w, x, τ) . Similar concepts were referred to in other works such as [11], where $\frac{1}{\tau}$ is called sampling frequency. Here, we leave τ as defined to indicate the continuity of the sequence of execution.

Petri nets often have corresponding diagrams that can give all the information on the mathematical workings of the system. While these can get quite intricate, we keep ours simple and utilize the legend for Petri nets model formalism elements, as seen in Figure 1.

Ordinary differential equation models are the standard model used in epidemiology, drug pharmacokinetics, and even cell population growth in diseases like cancer [12], [13], [14], and [15]. As such, it would be imperative to validate that Petri nets can mimic the dynamics of ODEs. The SIR model is the most commonly used and seen model in epidemiology, as it is the basis on which most other models are built. Derived initially by Kermack-McKendrick [16] in 1927, With S defined as the susceptible population, I being the infected population, and R the recovered population. In the system of Equations 1-3, the parameter β is the rate at which susceptible populations become infected. The parameter γ is the rate at which the infected population recovers per unit of time.

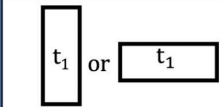

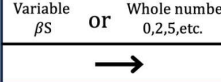
	transition
	place
Variable βS or Whole number 0,2,5,etc.	arc weight
	arc

Figure 1: Petri nets model formalism elements

$$\frac{dS}{dt} = -\beta SI \quad (1)$$

$$\frac{dI}{dt} = \beta SI - \gamma I \quad (2)$$

$$\frac{dR}{dt} = \gamma I \quad (3)$$

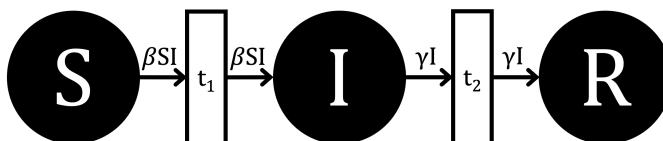
The classical SIR model assumes that there is no birth, no death, no immigration, no emigration, homogeneous mixing, that new infections are not dependent on other factors besides βSI , and that there is an exponential waiting time for events to happen in each compartment. In practice today, the base SIR model is typically used as a foundation for constructing more complex models that include disease-specific dynamics and various ways of treating or preventing the disease.

2.2 SIR Model in Petri Net Framework

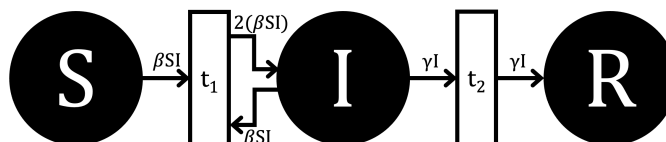
The model we focus on first is the SIR model, with only three equations (Equations 1-3), and relatively few parameters in comparison to other epidemiology models.

The corresponding PN model to the SIR ODE model can be laid out to preserve the ODE model’s assumptions (such as waiting times and closed population) and preserve biological dynamics (such as positive populations and susceptible populations always able to be infected). An example of this is seen in the differences between Layout 1 and 2 in Figure 2 the SIR Model, transitioning from the ODE equations.

With the exclusion/addition of the arc from place I to transition t_1 the two layout options are presented. This arc is included in many PN structures when transitioning from ODE to PN [17],[18]. However, this arc is not mandated by a mathematical principle and introduces a hindrance to the number of susceptible being infected at a given time step as the number of tokens in place S and place I have to be greater than βSI in order for transition t_1 to fire.



(a) SIR Layout 1



(b) SIR Layout 2

Figure 2: Two Petri net layout options for SIR model

Eliminating this additional arc and setting the arc weights to a level with the same net flow of tokens as shown in Layout 1 reduces the number of disabled transition situations that do not have sound biological principles. This hindrance arc from place I to transition t_1 could be considered biologically plausible in specific scenarios but not for the purposes of comparing to an ODE. The code for all of the models can be found in the GitHub ¹. The code contains the model laid out as PN, and the corresponding ODE is defined as a function.

The Petri net structure and ODEs can also be seen in the Segovia paper [17] Figure 2 and Yang paper [19] Figure 1. The SIR model can easily be adapted to the SIRS model by adding the term δR , representing the rate of re-susceptibility per unit of time. By adding δR to Equation 1 and subtracting it from Equation 3, we

¹<https://github.com/trevorreckell/Numerical-Comparison-of-PN-vs-ODE-for-SIR>

get Equations 4-6.

$$\frac{dS}{dt} = \delta R - \beta SI \quad (4)$$

$$\frac{dI}{dt} = \beta SI - \gamma I \quad (5)$$

$$\frac{dR}{dt} = \gamma I - \delta R. \quad (6)$$

Then using the same technique used to create Figure 2a, we apply Equations 4-6 to yield Figure 3. The resulting PN model structure is utilized throughout the various tests done on particular aspects of the Petri net. Note that if $\delta = 0$, the SIRS model will perform the same as the SIR model in the code.

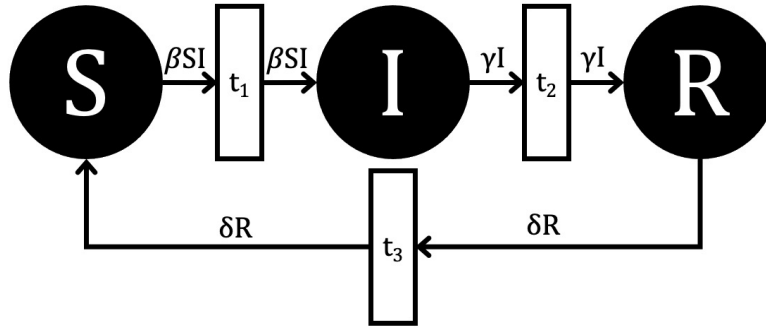


Figure 3: SIRS model

2.3 Dealing with extreme values in discrete systems

Although there are Continuous Petri nets (CPN) and Continuous Timed Petri nets (CTPN) [4], most software systems currently do not allow for either the construction of these continuous systems or for the development of code to make such Petri nets useful for analysis or forecasting. With no realistic software options for these types of PNs, we narrowed our scope to discrete PNs, where discrete refers to both time and token values.

One impactful drawback of these discrete PNs is dealing with relatively low token values, which disable the firing of transitions.

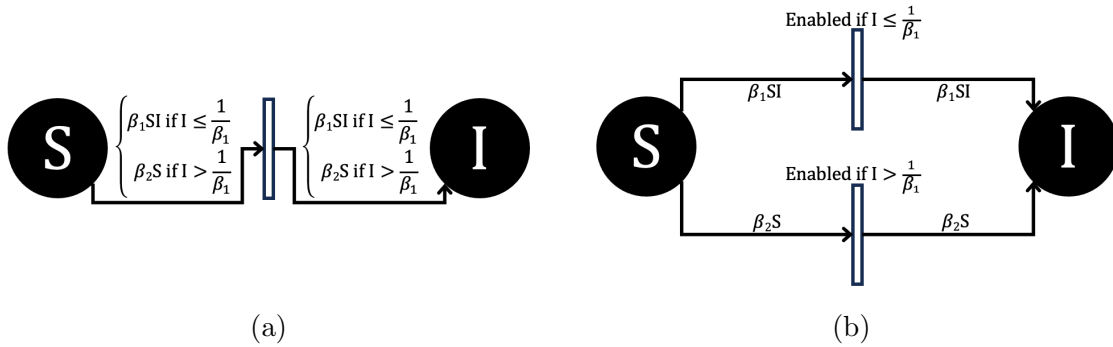


Figure 4: Petri net layout options for dealing with low population values. In Figure 4a, this can be achieved with a simple "if else" logic statement. In 4b an additional transition needs to be installed.

For example, using the layout options outlined in Figure 4, we can allow for the transition to fire even if the arc weight is initially higher than the place has tokens, specifically if the system is calling for a number of tokens to be moved from place S (susceptible population) to place I (infected population), which is larger than the number of tokens that place S has. The transition would normally not fire until enough tokens have built up in place S (susceptible population) or the arc weight has changed to a lower value. Alarmingly, this does not follow biological dynamics. Thus, we can adapt either 4a or 4b implementations to allow a firing to occur still. In figure 4a, we allow the formula for the arc weight to switch when we transition t_1 is disabled, where this transition being disabled corresponds biologically to not being able to infect the susceptible population. The formula goes from $\beta_1 SI$ to $\beta_2 S$ where $\beta_2 = \frac{1}{I+1}$. If $\beta_2 \leq \frac{1}{7}$, then transition t_1 will be enabled, but the formula for β_2 could be set to anything in the range of $[0, \frac{1}{7}]$. The formula will depend on the system being modeled and the desired dynamics. The method laid out in figure 4b is similar, just with a different transition being enabled if $I > \frac{1}{\beta_1}$, rather than the formula for the arc weight changing for the same transition. Another method of dealing with these extreme values is outlined in the PN time steps per unit time in Section 3.2.

2.4 GPenSIM

General-purpose Petri Net Simulator (GPenSIM) is a new tool for the MATLAB platform. GPenSIM is for modeling, simulation, and performance analysis of discrete systems. One of the authors of this paper developed GPenSIM. The main reasons for developing GPenSIM were [20]:

- **Easiness:** GPenSIM should be easy to learn and use.
- **Interoperability:** GPenSIM should be able to model discrete systems in different domains such as Production and Mechanical Engineering, Industrial Engineering, Computer science and engineering, etc., taking the full potential of MATLAB's numerous functions in diverse toolboxes.
- **Extensibility:** GPenSIM should be simple, possessing a core engine and an interface so that users can use the interface to extend GPenSIM functions or create new functions to model their systems.
- **Industrial Applications:** GPenSIM should be able to solve industrial problems in addition to academic (teaching and research) usage.

The original version of GPenSIM was developed for P/T Petri nets for simple analysis [20]. Later, coloring capability (Colored Petri nets) was added so that real-life systems could be modeled [21]. Also, Petri nets pose a huge problem: even for small systems, their Petri net models tend to become huge. Hence, for modeling large real-life systems, modular modeling capability was added to GPenSIM [22].

Though Petri nets implemented by GPenSIM are fundamentally deterministic, GPenSIM allows static Petri nets with non-varying arc weights. At the start of the simulation, a static Petri net graph must be defined in a separate file (Petri net Definition File (PDF)), and this file will be used throughout the simulation. However, with an iterative approach, this weakness can be overcome. In the iterative approach, we start with a static Petri net graph with some initial arc weights (initial PDF file). At the end of the simulation, the newer arc weights are found, and these arc weights are assigned to the corresponding arcs to create a new PDF file for the next iteration.

The iterations are run until the arc weights reach the maximum time interval and can converge, similar to an ODE, with particular parameter sets to equilibria corresponding to ODE equilibria. With this iterative approach, we can also use GPenSIM to model and simulate stochastic Petri nets. However, the iterative approach can take time, and hence, timing (slowness in simulation) can become an issue when using GPenSIM for stochastic applications.

3 Methodology – Petri Net Modeling Issues

As mentioned earlier, multiple aspects should be considered when validating Petri nets as an epidemiological modeling tool. However, we will focus on the significant issues involving the rounding scheme and time steps.

3.1 Rounding Variable Arc Weights in Discrete Petri Nets

Petri net simulation software, as a whole, does not have a standardized method for rounding variable arc weights, also called dynamic arc weights or arc expressions in some software documentation [23][24]. Rounding of the arc weight is necessary to have the arc weights be whole numbers. Otherwise, continuous weights and token values would need to be utilized. If using the standard PN approach of a whole number of arc weights proves insufficient, another approach is to use continuous values for the arc weights. Unfortunately, the software options for continuous value arc weights are minimal. Amongst variable arc weight PN software with discrete arc weight values, there is no standardized rounding system for the variable arc weights. With this in mind, the question arises about what rounding system should be used. The initial functions we will define are the ceiling function (weights are rounded up to the nearest whole number), floor function (weights are rounded down to the nearest whole number), and standard rounding (weights are rounded down when the tenths place is less than five and up when the tenths place is greater than or equal to five).

We propose another rounding method that utilizes standard rounding with the addition of carrying the residual of the rounding process to the next time step. In the next time step, the residual is added to the same arc's weight before that weight is rounded again. This process is repeated each time the variable arc weight is recalculated. This process could be done with other rounding functions, like ceiling or floor functions, and we will compare this "standard + residual" method with many other possible rounding methods more rigorously in future work. These rounding methods will each likely have their own drawbacks in terms of computation time and memory, but crucially selecting one to allow the dynamics of a PN to mimic that of an ODE closer is the goal.

A rounding scheme can only change a PN model so much. To improve the PN model further, we must explore time steps, a classic problem with discrete-time systems.

3.2 Time Steps of Petri Net Models

A Petri net can be set up in various ways, such as with a set number of firings allowed in a given time interval or with each firing taking a set amount of time, of course, with firings allowed to happen if the transition is enabled. The best approach depends on the system being modeled. For our purposes of examining an epidemiological SIR ODE model, we have allowed for one firing per PN time step. Additionally, we allow multiple PN time steps to occur for every "unit of time," which could be seconds, hours, or weeks, depending on the disease being modeled. This means we

could allow for one PN time step for one unit of time used by the ODE or arbitrary PN time steps, depending on the PN model and desired dynamics. This PN time step per unit of time τ could be seen as a parameter that could be fit to a data set with the other parameters but also could be set at an arbitrarily high level to allow for rapid dynamic to occur and then have the same number of fit parameters as the corresponding ODE system. It should be noted that similar dynamics could be done by assigning a set amount of time for each transition to fire. The benefit of this system composition is that each transition could have an independent firing time. However, combining this dynamic with the variable arc weights would be challenging to set up, and given that we are considering consistent firing times, we decided to stay with altering the PN time steps per unit time for this work.

When changing τ , the other parameters implemented are dependent on a respective time unit, so they need to be scaled to ensure they are applied appropriately. For instance, if the susceptible population becomes infected at a rate of 5%, $\beta = 0.05$, and we are changing the PN time step to twenty-time steps per one unit of time, then we need to scale β by $\frac{1}{20}$ giving $\beta = 0.05 \cdot \frac{1}{20} = 0.0025$. With the lower parameter value, this method also has the benefit of avoiding the situation of not allowing the transition to fire as frequently as laid out in Section 2.3.

With either method, varying time steps or firing times, there will be a significant trade off of computation time. As such, we will calculate the mean computation time when running the model for various time steps to get an idea of what times step level is necessary given the desired dynamic level and computational power available.

4 Results

The methodology laid out will be implemented, starting with looking at the rounding methods to determine which best mimics ODE dynamics when applied to a PN. From this, the best rounding method will be applied to the PN for all further simulations, to determine the effect of PN time steps per unit time (τ) on the PN model compared to the ODE model when parameter values are the same. However, first, we need a numerical comparison approach between ODEs and PNs.

4.1 Numerical comparison of models

For the purposes of ease of following the associated code², we follow the notation used in MATLAB documentation for root mean square error. We denote the observed

²<https://github.com/trevorreckell/Numerical-Comparison-of-PN-vs-ODE-for-SIR>

data vector as A , where A_i is a single vector entry indexed by the time point i . The forecasted data is denoted as F , where F_i is a single forecasted vector entry again indexed by the time point i . Finally, n represents the total number of time points being compared. Specifically The approach for comparison is relative root mean square error (RRMSE) as defined in

$$RRMSE = 100 \cdot \sqrt{\frac{\frac{1}{n} \sum_{i=1}^n |A_i - F_i|^2}{\sum_{i=1}^n |A_i|^2}}$$

where A_i is the ODE model data, and F_i is the forecasted PN model data in our application.

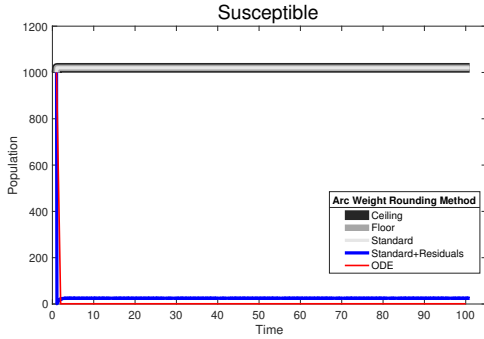
This comparison will inspect the comparison given the parameters β , γ , and δ , across their respective ranges of $[0,1]$. Using as an example the arc weight and ODE rate of βSI . A level of $\beta = 0$ means that an infected individual infects and/or interacts with 0% of all susceptible individuals, and $\beta = 1$ means that an infected individual infects and interacts with 100% of all susceptible individuals. In a continuous ODE system, this happens with exponential waiting time in each compartment, and measuring true values for the parameters is not realistic, so they are often estimated based on other factors. However, the value of one is likely far beyond the biologically plausible level for β , γ , and δ . Let us look at the most extreme case for each of the parameters. First, if we look at the measles (rubeola) with a maximum theorized R_0 value of 18 [25]. Within the SIR ODE framework $R_0 = \frac{\beta}{\gamma}$. This would mean even with $\beta = 1$ at maximum theoretical value, γ maximum value would be $\frac{1}{18} \approx 0.05556$ for measles. For δ with a minimum time of reinfection of 90 days, even for respiratory diseases, it leads to a δ value of 0.25. The parameter β represents a combination of the percent of susceptible population that an infected person interacts with infects as well as the interaction rate. Suppose we inspected β further and assumed that infected population interacted with unrealistically high values of 70% of susceptible population and the disease infected unrealistically high values of 70% of susceptible population with every interaction. In that case, it leads to $\beta = 0.49$. Current ODE modeling efforts where the model is applied to real diseases have more compartments and often more mechanisms for various things like vaccines, age, immigration, etc. With these come additional equations and parameters that split apart what the parameters of a reduced model represent. Looking at models of diseases like COVID-19 [19], measles [26], and influenza [27] combined with the examples given above, the realistic range for each parameter is still far less than the theoretical range of $[0,1]$. Thus, from these same sources, we define the biologically realistic parameter range for each respective parameter of $[0,0.5]$.

4.2 Rounding Variable Arc Weights in Discrete Petri Nets

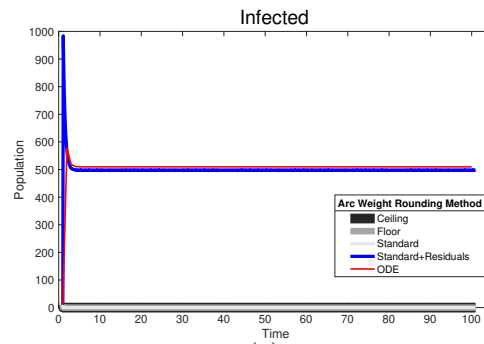
First, it is necessary to address which rounding method is best, as the rounding method selected will then be utilized in further experiments. Thus, we tested the various rounding methods for parameters β , γ , and δ as laid out in the SIR model for ODE and PN at extreme values, including (0,0,0) and (1,1,1) and various biologically plausible values. From here, we use RRMSE to compare the ODE to the PN models and visually observe the system's dynamics. Visual observation, although somewhat subjective, is essential when determining model performance, as large spiking or missing in inconsistent ways would lead to not wanting to put too much weight on a model for important decisions.

Error Method Comparison

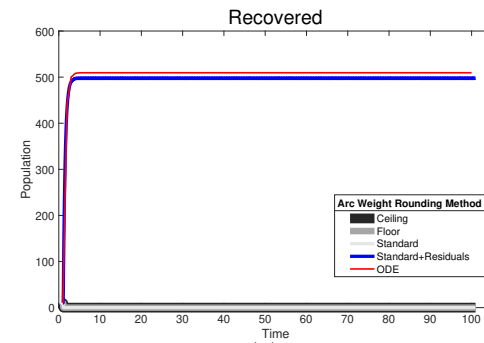
Parameters: $\beta = 1, \gamma = 1, \delta = 1$



(a)

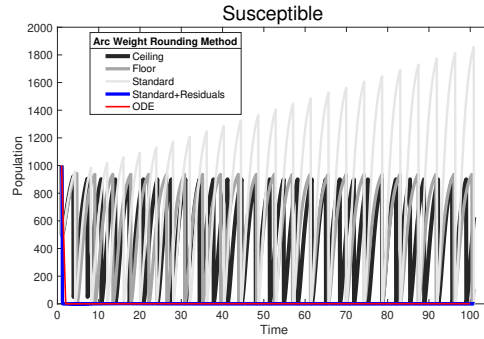


(c)

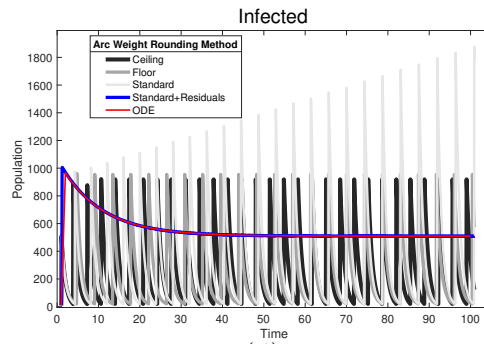


(e)

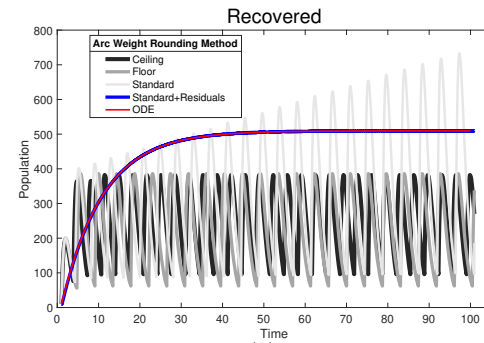
Parameters: $\beta = 0.05, \gamma = 0.05, \delta = 0.05$



(b)



(d)



(f)

Figure 5: This is a comparison of different rounding methods for when variables are $\beta = 1, \gamma = 1, \delta = 1$ for the figures 5a, 5c, and 5e on the left side and $\beta = 0.05, \gamma = 0.05, \delta = 0.05$ for the figures 5b, 5d, and 5f on the right side. The top row is the susceptible population, the middle is the infected population, and the bottom is the recovered population. All rounding method comparisons were conducted at a time step of 20 PN time steps per unit time per 1 ODE time interval.

In all of the Figure 5's subfigures, the PN time steps per unit time (τ) is set to 20. Initially, when viewing figures 5a, 5c, and 5e, we can see that the non-residual rounding methods are unable to capture the dynamics of the ODE at all. Granted, this is for the most extreme case for the parameter values, but it is important that the PN model is capable of capturing the dynamics for all values $[0,1]$ for β, γ , and δ . When looking at more biologically plausible values of $\beta, \gamma, \delta = 0.05$ in figures 5b, 5d, and 5f we can see that the nature of PN firing only when transitions are enabled combined with the extreme value situation as laid out in section 2.3, the dynamics behave like a 2-cycle. Without the extra logic statements laid out in section 2.3, the dynamics are even less biologically plausible, though, with no new infected even with high levels of infected and susceptible, simply with $S < \beta SI$.

4.3 Time Steps of Petri Net Models

The PN model, as seen in Figure 3, with different PN time steps per unit time (τ) are compared to the ODE model Equations 4-6 with the same parameters values using RRMSE. The parameter grid chosen was a linearly spaced grid of size ten between $[0,1]$ for parameters β and γ . These are the x-axis and y-axis, respectively, for all of the sub-figures in Figures 6-10. Then for δ there is a logarithmic spaced grid of size five between $[0,1]$ going from the top to bottom row of Figures 6-10.

One PN Time Step Per Unit of Time, $\tau = 1$

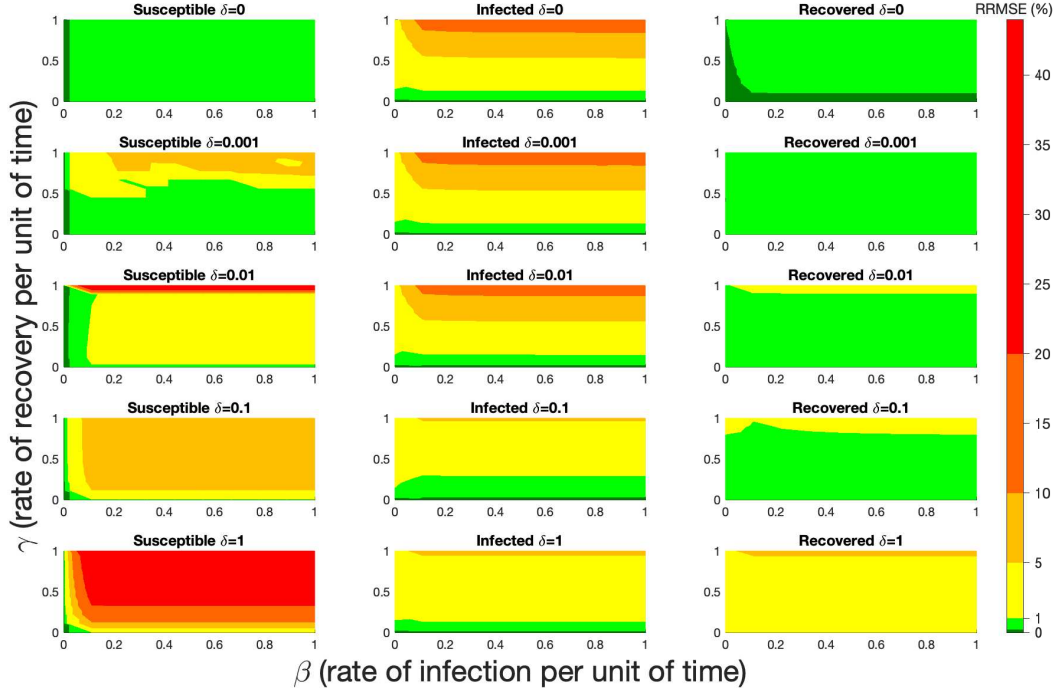


Figure 6: RRMSE percentage across parameter ranges of $[0, 1]$ for each respective parameter with γ the y-axis of each subfigure, β the x-axis of each subfigure, and δ set at a different fixed value for each subfigure. PN Time Step Per Unit of Time parameter $\tau = 1$. Note that red is RRMSE $\leq 44\%$ (43.75% being the max observed RRMSE across all simulations), dark orange is RRMSE $\leq 20\%$, light orange is RRMSE $\leq 10\%$, yellow is RRMSE $\leq 5\%$, light green is RRMSE $\leq 1\%$, and dark green is RRMSE $\leq .1\%$

When the $\tau = 1$, in Figure 6 the RRMSE performance is relatively poor, especially for higher values of β , γ , and δ . This low τ value displays the problem of comparing a discrete versus a continuous time system with large swings in population happening instantly at the time step, not allowing the dynamics of the PN to come close to matching that of an ODE continuous system. These sharp dynamics combined with the additional firing mechanisms of PN means the RRMSE values produced are relatively large.

Twenty PN Time Steps Per Unit of Time, $\tau = 20$

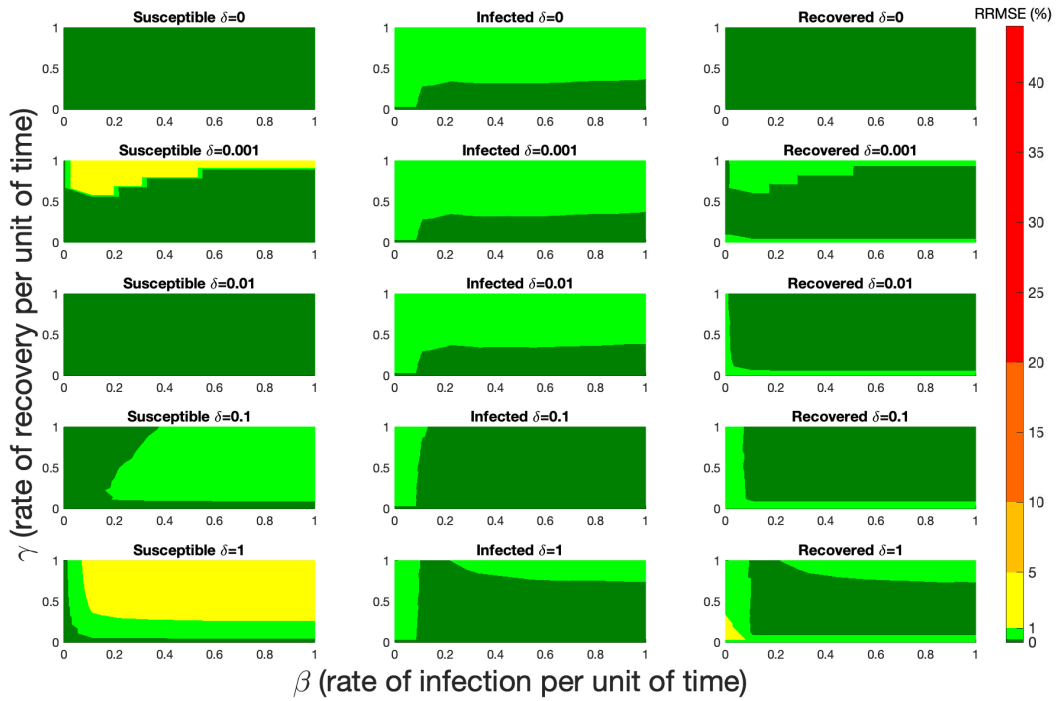


Figure 7: RRMSE percentage across parameter ranges of $[0, 1]$ for each respective parameter with γ the y-axis of each subfigure, β the x-axis of each subfigure, and δ set at a different fixed value for each subfigure. PN Time Step Per Unit of Time parameter $\tau = 20$. Note that yellow is $\text{RRMSE} \leq 5\%$, light green is $\text{RRMSE} \leq 1\%$, and dark green is $\text{RRMSE} \leq 0.1\%$.

Figure 7 shows the vast improvement in RRMSE with utilizing higher PN time steps per unit time. With this one change, the maximum RRMSE becomes ≈ 4.3 and the visual improvement at extreme values is immense.

Forty PN Time Steps Per Unit of Time, $\tau = 40$

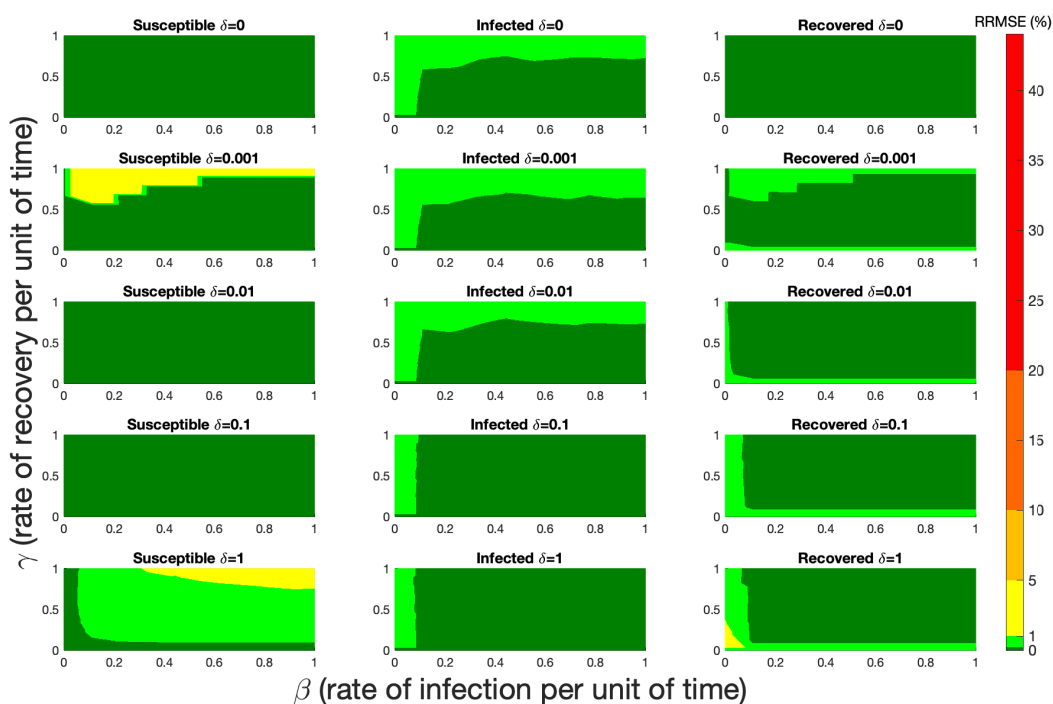


Figure 8: RRMSE percentage across parameter ranges of $[0, 1]$ for each respective parameter with γ the y-axis of each subfigure, β the x-axis of each subfigure, and δ set at a different fixed value for each subfigure. PN Time Step Per Unit of Time parameter $\tau = 40$. Note that yellow is $\text{RRMSE} \leq 5\%$, light green is $\text{RRMSE} \leq 1\%$, and dark green is $\text{RRMSE} \leq 0.1\%$.

Sixty PN Time Steps Per Unit of Time, $\tau = 60$

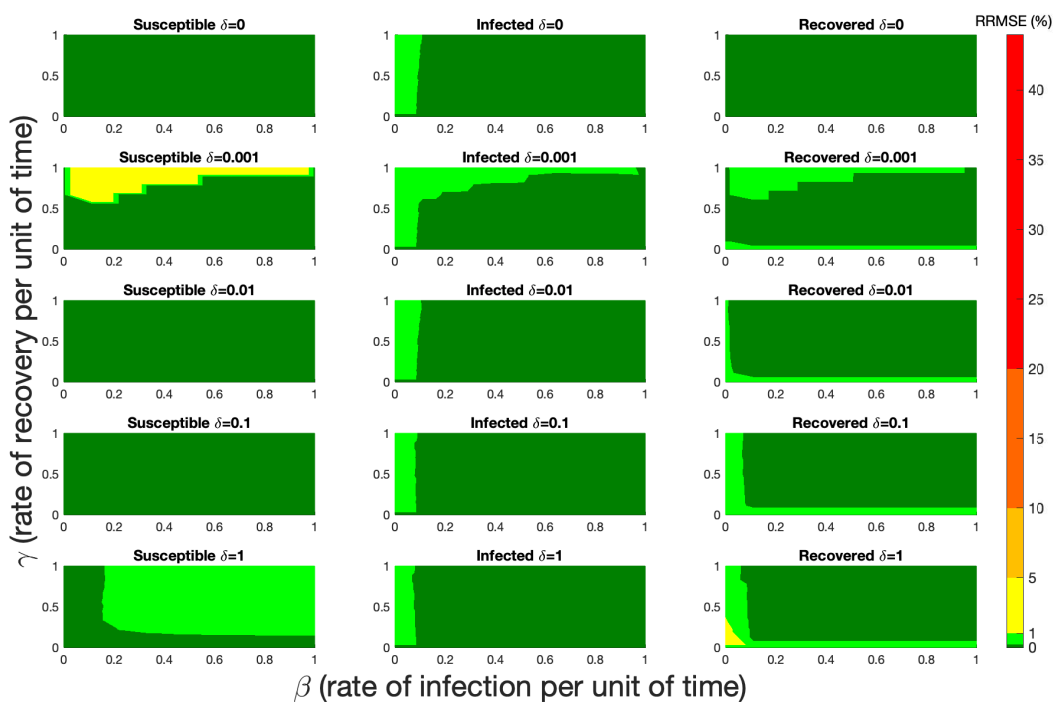


Figure 9: RRMSE percentage across parameter ranges of $[0, 1]$ for each respective parameter with γ the y-axis of each subfigure, β the x-axis of each subfigure, and δ set at a different fixed value for each subfigure. PN Time Step Per Unit of Time parameter $\tau = 60$. Note that yellow is $\text{RRMSE} \leq 5\%$, light green is $\text{RRMSE} \leq 1\%$, and dark green is $\text{RRMSE} \leq 0.1\%$.

Eighty PN Time Steps Per Unit of Time, $\tau = 80$

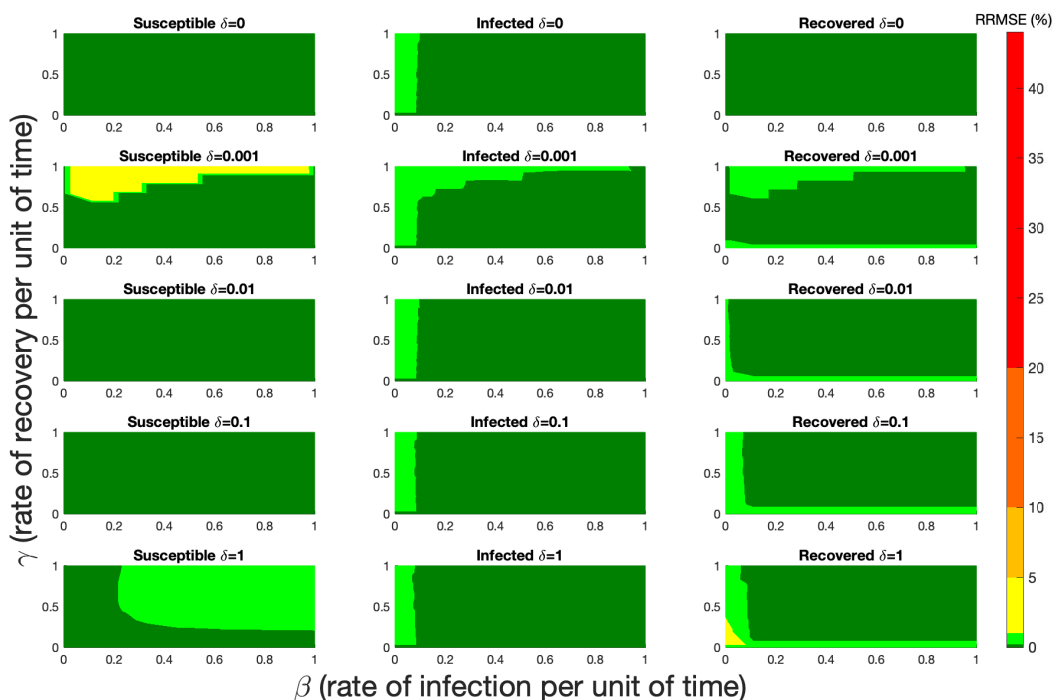


Figure 10: RRMSE percentage across parameter ranges of $[0, 1]$ for each respective parameter with γ the y-axis of each subfigure, β the x-axis of each subfigure, and δ set at a different fixed value for each subfigure. PN Time Step Per Unit of Time parameter $\tau = 80$. Note that yellow is $\text{RRMSE} \leq 5\%$, light green is $\text{RRMSE} \leq 1\%$, and dark green is $\text{RRMSE} \leq 0.1\%$.

In figures 6-10, there is an overall reduction of RRMSE with each subsequent increase of the PN time steps per unit time, which can be more clearly seen in Figure 11.

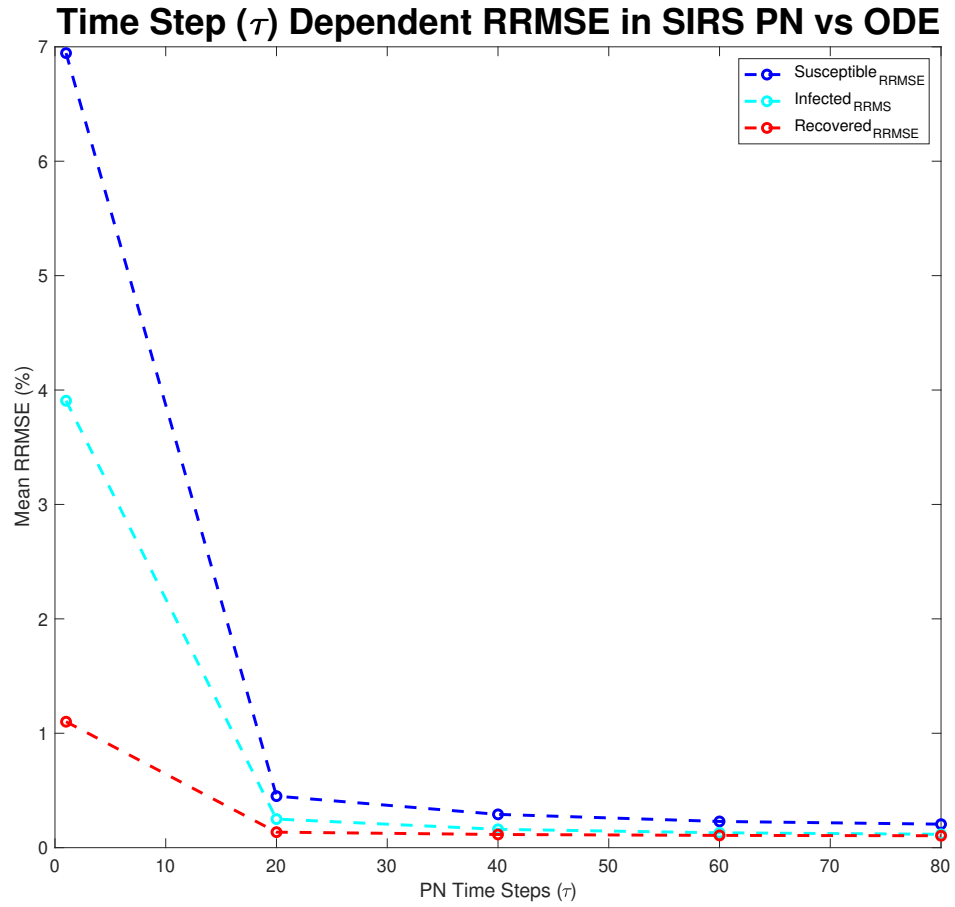


Figure 11: Mean RRMSE in percentage across parameter ranges of gamma (γ) $[0,1]$ with 10 linearly spaced points, beta (β) $[0,1]$ with 10 linearly spaced points, and delta (δ) $[0,1]$ with 5 logarithmically spaced points.

Computation Time for PN depending on Times Steps (τ)

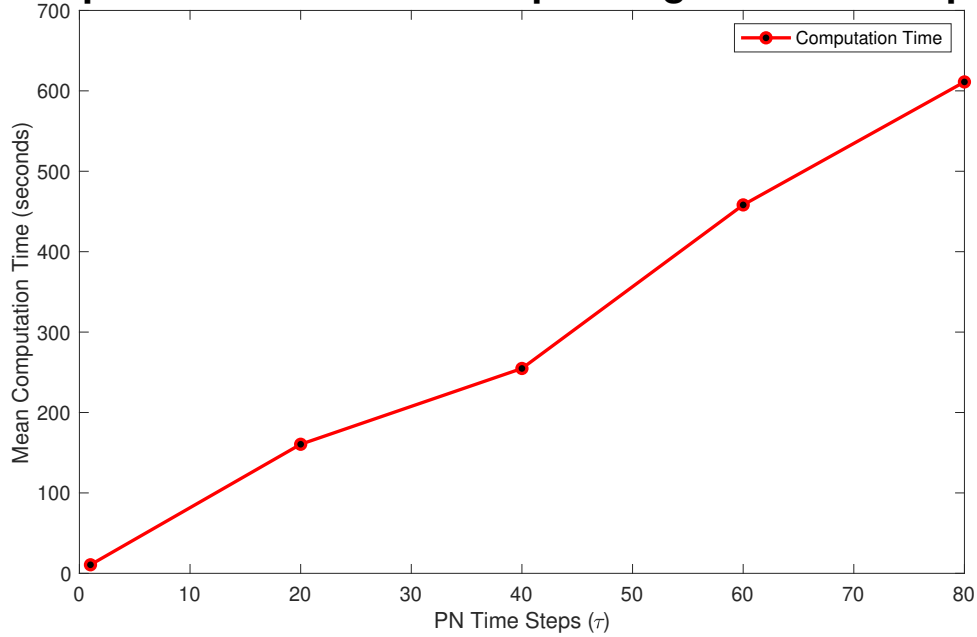


Figure 12: Mean computation time for one PN model run for various times steps as seen in Figures 6,7,8, 9, and 10.

For Figure 12 the values were found while being run in MATLAB 2023b, using GPenSIM v.10 software, on a computer with 2.3 GHz 8-Core processor, with 64 GB of memory. However, these models were also run on the Arizona State University SOL supercomputer, where even when allocated 4 cores and 32 GB of memory, only had a maximum memory used value of 5.6 GB. In other words, this runs as efficiently on a laptop system in the current formulation. We hope that with these reasonable computation times and minimal memory usage, others will also be able to build off of our work in order to improve PN models further.

5 Discussion and Conclusions

The results of Figures 6-11, particularly that of Figure 11 show a clear improvement of RRMSE for each population (place or compartment) of the basic SIR model. Although the RRMSE looks relatively high for Figure 6, for the range of biologically realistic parameters, the RRMSE is still below 10%. By the time we are up to a level 20 PN time steps per unit time in Figure 7 we are already at a level of less

than 1% RRMSE for all biologically realistic parameters and less than 5% RRMSE for the full parameter range. When we increase the PN time steps per unit time 40, 60 and 80, in Figures 8, 9, and 10, respectively the region of less than 0.1% RRMSE grows. If a model is used in practical application, the most critical population is the infected. When looking at this population, the RRMSE remains less than 1% across the entire parameter range. Deaths and hospitalizations are typically viewed as the most important with extended SIR models, but that will be addressed when models with compartments/places are tested in future work. Regardless, the PN structure, rounding method, and time steps per unit time combine to allow the less than 1% RRMSE performance from PN model. The methodology of this paper can and will be repeated for many of the most common extensions of the SIR model in future work. The yellow regions that remain within Figures 8-10 result from rounding arc weights. While this work addresses some issues Petri nets have when being used to fit and forecast diseases, many more issues will need to be investigated further in future work. Further investigation on the best rounding method, assuming discrete arc weight values are used, is among the most important.

Additional hyper-parameters to scale the parameters of the PN to fit the ODE better will also be introduced in future work. These larger models and additional parameters will likely increase the computation time to run the models. These factors include choosing an appropriate τ level to reach a tolerable RRMSE for the chosen modeling and forecasting objective.

In the basic SIR model structure, a Petri net's firing dynamics do not benefit RRMSE or follow standard biological dynamics. However, that is not to say that having an arc enabled to fire can not provide any benefit, as there are particular applications, such as batch immunization protocols or event-based infection events, where firing dynamics will likely be a helpful feature.

All of our code used in this initial paper can be found at <https://github.com/trevorreckell/Number>. From here, there are sections in the main simulation file of `simple_pn_SIRv1_preprint.m` to reproduce each figure and alter the code for individual purposes. We hope people can use this repository to help with their work. When associated papers are published, more files with more advanced models and further research will be added.

Funding

Trevor Reckell and Stefano Chiaradonna are partially supported by NIH grant DMS-1615879, Petar Jevtić, Beckett Sterner, NIH grant 5R01GM131405-02.

Conflict of Interest

The authors declare no conflict of interest.

References

- [1] J. H. Moore, “The ubiquitous nature of epistasis in determining susceptibility to common human diseases,” *Human heredity*, vol. 56, no. 1-3, pp. 73–82, 2003.
- [2] D. J. Wilkinson, *Stochastic modelling for systems biology*. Chapman and Hall/CRC, 2018.
- [3] S. Connolly, D. Gilbert, and M. Heiner, “From epidemic to pandemic modelling,” *Frontiers in Systems Biology*, vol. 2, p. 861562, 2022.
- [4] I. Koch, W. Reisig, and F. Schreiber, *Modeling in systems biology: the Petri net approach*, vol. 16. Springer science & business media, 2010.
- [5] L. Peng, P. Xie, Z. Tang, and F. Liu, “Modeling and analyzing transmission of infectious diseases using generalized stochastic petri nets,” *Applied Sciences*, vol. 11, no. 18, p. 8400, 2021.
- [6] M. Chen and R. Hofstaedt, “Quantitative petri net model of gene regulated metabolic networks in the cell,” *In Silico Biology*, vol. 3, no. 3, pp. 347–365, 2003.
- [7] S. Libkind, A. Baas, M. Halter, E. Patterson, and J. P. Fairbanks, “An algebraic framework for structured epidemic modelling,” *Philosophical Transactions of the Royal Society A*, vol. 380, no. 2233, p. 20210309, 2022.
- [8] V. B. Kumbhar and M. S. Chavan, “A review of petri net tools and recommendations,” in *International Conference on Applications of Machine Intelligence and Data Analytics (ICAMIDA 2022)*, pp. 710–721, Atlantis Press, 2023.
- [9] W. J. Thong and M. Ameen, “A survey of petri net tools,” in *Advanced Computer and Communication Engineering Technology: Proceedings of the 1st International Conference on Communication and Computer Engineering*, pp. 537–551, Springer, 2015.
- [10] C. G. Cassandras and S. Lafortune, *Introduction to discrete event systems*. Springer, 2008.
- [11] R. Davidrajuh, “Optimizing simulations with gpensim,” *Modeling Discrete-Event Systems with GPenSIM: An Introduction*, pp. 59–79, 2018.

- [12] P. Auger, P. Magal, and S. Ruan, *Structured population models in biology and epidemiology*, vol. 1936. Springer, 2008.
- [13] J. Lu, K. Deng, X. Zhang, G. Liu, and Y. Guan, “Neural-ode for pharmacokinetics modeling and its advantage to alternative machine learning models in predicting new dosing regimens,” *Iscience*, vol. 24, no. 7, 2021.
- [14] T. Reckell, K. Nguyen, T. Phan, S. Crook, E. J. Kostelich, and Y. Kuang, “Modeling the synergistic properties of drugs in hormonal treatment for prostate cancer,” *Journal of theoretical biology*, vol. 514, p. 110570, 2021.
- [15] B. D. Aguda, Y. Kim, M. G. Piper-Hunter, A. Friedman, and C. B. Marsh, “MicroRNA regulation of a cancer network: consequences of the feedback loops involving mir-17-92, e2f, and myc,” *Proceedings of the National Academy of Sciences*, vol. 105, no. 50, pp. 19678–19683, 2008.
- [16] W. O. Kermack and A. G. McKendrick, “A contribution to the mathematical theory of epidemics,” *Proceedings of the royal society of london. Series A, Containing papers of a mathematical and physical character*, vol. 115, no. 772, pp. 700–721, 1927.
- [17] C. Segovia, “Petri nets in epidemiology,” *arXiv preprint arXiv:2206.03269*, 2022.
- [18] J. C. Baez and J. Biamonte, “Quantum techniques for stochastic mechanics,” *arXiv preprint arXiv:1209.3632*, 2012.
- [19] W. Yang, D. Zhang, L. Peng, C. Zhuge, and L. Hong, “Rational evaluation of various epidemic models based on the covid-19 data of china,” *Epidemics*, vol. 37, p. 100501, 2021.
- [20] R. Davidrajuh, *Modeling Discrete-Event Systems with GPenSIM*. Cham: Springer International Publishing, 2018.
- [21] R. Davidrajuh, *Colored Petri Nets for Modeling of Discrete Systems: A Practical Approach With GPenSIM*. Springer Nature, 2023.
- [22] R. Davidrajuh, *Petri Nets for Modeling of Large Discrete Systems*. Springer, 2021.
- [23] R. Melberg and R. Davidrajuh, “Dynamic arc weight in petri nets,” in *Proceedings of the IASTED International Conference on Applied Simulation and Modelling (ASM 2009)*, vol. 682, pp. 83–89, 2009.

- [24] F. Liu and M. Heiner, “Colored petri nets to model and simulate biological systems,” *Recent advances in Petri Nets and concurrency*, vol. 827, pp. 71–85, 2012.
- [25] P. L. Delamater, E. J. Street, T. F. Leslie, Y. T. Yang, and K. H. Jacobsen, “Complexity of the basic reproduction number (r_0),” *Emerging infectious diseases*, vol. 25, no. 1, p. 1, 2019.
- [26] H. W. Berhe and O. D. Makinde, “Computational modelling and optimal control of measles epidemic in human population,” *Biosystems*, vol. 190, p. 104102, 2020.
- [27] L. J. Allen, “Stochastic population and epidemic models,” *Mathematical biosciences lecture series, stochastics in biological systems*, vol. 1, pp. 120–128, 2015.

Vector Borne Model Petri Net fi

Infected humans RRMSE=0.12983

