LUNDS
UNIVERSITET

Tau-Leaping

Implementations outside of chemistry

Author: Lotta Kornö
Supervisor: Farrukh Javed
Date: March 2024
Department: Statistical institution
Course: STAH11

1 Abstract

Tau-leaping is an algorithm for model simulations most often used in kinetic chemistry. It was created to make simulations more efficient at the cost of some accuracy. However, its uses outside of chemistry are limited but could help make some model simulations more efficient. This study aimed to implement Tau-leaping and analyze it from a statistical perspective. The Lotka-Volterra prey-predator and SIR models were used to implement Tau-leaping and compared to the exact SSA version. The result was greatly improved run times at the cost of some accuracy. Since these models' purpose is to understand the evolution and trends in a population, the loss of accuracy wasn't detrimental. The conclusion then became that Tau-leaping is preferred depending on the purpose of the simulation.

Contents

2 Introduction

In the field of chemistry, there is a need to simulate and understand the transformation and evolution of a reacting system over a period of time. Before Daniel T. Gillespie formulated the tau leaping method in 2001 the industry standard was to use the stochastic simulation algorithm (Gillespie, 2001). In this report, the stochastic simulation algorithm will be referred to as SSA, but the algorithm has many names including the direct- and exact methods. SSA makes it possible to simulate every reaction occurring in a system while still taking the randomness of the system into account. Since SSA simulates every reaction in the system, the method is extremely slow. This led Daniel T. Gillespie to develop the Tau-leaping algorithm in 2001 (Gillespie, 2001). Since then Tau-leaping has mostly been used in chemistry and nearby fields. This report aims to look at Tau-leaping from a statistical point of view.

2.1 Background

Chemical kinetics aims to describe the rates of reactions and the changes in the system caused by the reactions. Traditionally these analyses have been done with an assumption that the system is deterministic. However, this is not accurate for three reasons. Firstly, it is hard to keep a chemical system in constant isolation. Secondly, it is impossible to know the exact time of a reaction. Lastly, even if all initial states of the system were known, aka deterministic, the evolution of the system would not be deterministic (Gillespie, 2007). This led to the introduction of stochastic simulation methods in kinetic chemistry. One of the most used methods for this is SSA.

However, SSA is a slow method, which is why the Tau-leaping algorithm was created. The algorithm is based on the idea that, in most cases, it is not necessary to know exactly when and where a reaction happened but rather the number of reactions of each type. The idea then became that leaps of small time steps could be taken and the number of each reaction type could be simulated, for each leap. Resulting in a significantly faster method, at the cost of some accuracy (Gillespie, 2001). The details of how the algorithm works are complex and will be further explained in section 3.

The question then becomes how big the leaps should be for maximum efficiency and minimal accuracy loss. This led to the formation of the following leap condition, which describes the boundaries the leap size must adhere to.

The time leap ($= \tau$) has to be so significantly small
that the changes in all sub-populations during the

time leap $[t, t + \tau)$ are so small that the no rate function changes significantly (Gillespie, 2001).

Over time many different variations of choosing the leap size have been suggested. The first and simplest suggestion was given by Gillespie in 2001, called the explicit Tau-leaping algorithm. A constant value for the time leap is decided upon based on the leap condition. Algorithms used for computing the leap size have also been suggested, the most common ones being the optimized- and implicit Tau-leaping algorithm. Finally, it is also possible to mix methods, the most common is adaptive tau leaping. It is a combination of SSA, explicit- and implicit Tau-leaping. Their main similarity is that they all have to meet the leap condition(Gillespie, 2007). This report will include SSA, explicit- and optimized Tau-leaping.

A vital part of the leap condition is the rate function $a_j(x)dt$, which has to be constant during the entire leap $[t, t + \tau)$. This ensures that the derivative of the rate functions $a_j(x)dt$ is the probability that any reaction type R_j will occur during the time leap, no matter what any other reaction type is doing during the time. In section 3, the Tau-leaping algorithm will be described in detail as well as the SSA (Gillespie, 2001).

2.1.1 Uses outside of chemistry

As stated above Tau-leaping is mainly used in the field of kinetic chemistry. However, that doesn't mean that it isn't possible to use it for other purposes. Due to how the algorithm is formulated, it is best suited for simulations of a population, containing two sub-populations that interact. These interactions or as they are also called transitions correspond to the reactions happening in a chemical system. The sub-populations correspond to the two types of atoms/molecules reacting in chemistry.

One recent use case of the algorithm outside of chemistry is the simulation of the spread of vaccine resistance strains of the virus SARS-Cov-2 in society. The model aims to help study the likelihood of this happening and the consequences it would bring. The authors suggested Tau-leaping and SSA as alternatives for simulating the model (A. Rella et. al, 2021). Another example is ecology, where Tau-leaping can be used to simulate the relationship between prey and predators in nature. In this report, a SIR model to model the spread of disease and the Lotka-Volterra model to simulate the relationship between animals are used.

2.2 Simulations

The basis of simulations is using models to mimic a process or system, aiming to describe how a model evolves over time. In reality, simulations are used to support ideas and decisions. Making them capable of helping increase performance, training, optimizing processes and much more. A few advantages of using good simulations could be lower costs, analysis of long-term changes, model non-standard distributions and many more. However, simulations have limitations for example insufficient models, high cost and long run time. For companies using simulations as a tool, the run time must be low since this is an important way of reducing costs (TWI, n.d.).

2.3 Research question

In this study, Tau-leaping was implemented in R-studio and used to simulate statistical models. The study intends to analyze use cases of Tau-leaping outside of chemistry, with a statistical perspective. In order to evaluate the method's accuracy as well as its efficiency. To be able to greatly improve the run-time.

3 Method

As stated above three different methods will be included in this report: SSA, Explicit- and Optimized Tau leaping. The result of SSA will be used as the baseline for the other methods' results to be compared to. This is due to it not making any leaps bigger than for one transition to have the time to occur. This is also why SSA was chosen to be included in the study, without it the analyses of the explicit- and optimized Tau-leaping methods. The explicit method is the simplest version of Tau-leaping and was chosen to be one of the methods in this study to highlight the problems it can create. The optimized Tau-leaping method is one of the most commonly used and solves some of the issues with explicit Tau-leaping. Therefore it was also chosen as one of the methods in this study. These three methods are different but still similar enough that they can be easily compared and they show a wide range of what the Tau-leaping algorithm is capable of. This section will describe all three methods through a statistical lens.

3.1 Stochastic simulation algorithm

Imagine a population of size N, divided into different sub-populations $\{S_1, ..., S_N\}$. The sub-populations then interact in M different ways, referred to as transition types $\{R_1, ..., R_M\}$. The time is denoted as t and the sizes of the sub-population are a function of the time $X_i(t)$. This is because it is the size of the sub-populations that evolves as transitions in the population occur. The size of the sub-populations at time t can then be written as $X_i(t) = (X_1(t), ..., X_N(t))$. The algorithm

aims to simulate how the sub-population sizes $X_i(t) = (X_1(t), ..., X_N(t))$ change over a period of time t, given that the initial sub-population sizes $X(t_0) = x_0$ are known. Two things are essential for the simulation, the state change vector v_{ij} and the rate function a_j. The state change vector v_{ij} describes the changes in the sub-population sizes$\{S_1, ..., S_N\}$. caused by caused by each transition type $\{R_1, ..., R_M\}$. Despite its name, the state change vector is usually written as a matrix, usually consisting of ones and zeros. The reason is that the calculations and programming become simpler (Gillespie, 2007). An example of how this can look is shown below, if the population consists of three different sub-populations and can interact in three different transition types.

Transition type:	1	2	3
Sub-population 1	1	-1	0
Sub-population 2	0	1	-1
Sub-population 3	-1	0	1

The matrix can be interpreted as if transition type 1 occurs sub-population 1 grows with one, sub-population 2 doesn't change and sub-population 3 declines with one.

The rate function a_j describes the rate at which the different transition types R_j occur. If the derivative of the rate function is taken with respect to t, $a_j(x)dt$, it gives the probability that one transition type R_j happens in the given time leap $[t, t + dt)$, provided that $X(t) = x$.

$$a_j(x)dt = P(\textit{ One transition type } R_j \textit{ occur somewhere in the population during the time leap } [t, t + \tau)|\ X(t) = x)$$

The rate function is different depending on the model that is being simulated, but usually, some parameters are provided and put into equations where they interact with the current sub-population sizes. Hence it should be updated every after every loop in the simulation (Gillespie, 2007).

Using the rate function and state change vector, a density function is made. It is used to calculate the probability of the next transition type. The density function is denoted as $p(\tau, j|x, t)$, where τ is the time from t to the next transition (Gillespie, 2007).It looks as follows and a_o is defined as (2).

$$p(\tau, j|x, t) = a_j(x)e^{(-a_0(x)\tau)} \qquad \tau \geq 0,\ \ j = (1, ..., M) \tag{1}$$

$$a_o(x) = \sum_{j=1}^{M} a_j(x) \tag{2}$$

The derivative of the density function $p(\tau, j|x, t)$, concerning τ, $p(\tau, j|x, t)d\tau$ then becomes the probability that the next transition type is R_j and that it happens in the time interval $[t+\tau, t+\tau+d\tau]$. The derivative is now a joint probability density function based on two random variables, τ and

j, given that $X(t) = x$. j is the index of the next transition type. Equation 1 shows that τ is a random variable following an exponential distribution. That distribution has a mean and standard deviation of $-a_o(x) = \frac{1}{a_0(x)}$, hence $\tau \sim Exp(\frac{1}{a_o(x)}$ (Gillespie, 2001).

The goal now is to generate values from the random variables pair τ and ȷ. The most common way of doing this is through the so-called direct method. The first step is to rewrite the density function $p(\tau, j|x, t)$ in its conditioned form. This means that the density function for the pair (τ, j) is separated, providing one function for each random variable (Gillespie, 2001).

$$p(\tau, j|x, t) = p_1(\tau|x, t)p_2(j|\tau, x, t) \tag{3}$$

With p_1 and p_2 the τ and ȷ can now be generated (Gillespie, 2001).

$$p_1(\tau|x, t) = a_0(x)e^{(-a_0(x)\tau)} \qquad \tau \geq 0 \tag{4}$$

$$p_2(j|\tau, x, t) = \frac{a_j(x)}{a_0(x)} \qquad j = (1, ..., M) \tag{5}$$

The conclusion then becomes that $\tau \sim Exp(a_o(x))$ and $j \sim U(1, M)$ with the point probability $\frac{a_j(x)}{a_0(x)}$. This means that τ and ȷ can be sampled with the help of two random samples $r_1 \sim U(0, 1)$ and $r_2 \sim U(0, 1)$(Gillespie, 2001).

$$\tau = \frac{1}{a_0(x)} ln(\frac{1}{r_1}), \;\; r_1 \sim U(0, 1) \tag{6}$$

$$j = min(\sum_{j=1}^{j} a_j(x) > r_2 a_0(x)), \;\; r_2 \sim U(0, 1) \tag{7}$$

All the components of SSA are now known, and the algorithm looks as follows:

1. Define the initial states x_0 of the sub-populations and determine the start time t_0
2. Set $x = x_0$ and $t = t_0$ then compute the rate function $a_j(x)$ for each transition type, then the sum of all of them $a_0(x) = \sum_{j=1}^{M} a_j(x)$
3. Use equation 6 to generate the time until the next reaction τ
4. Generate the next transition type´s index j using equation 7
5. Update the current state of all sub-populations x. By using j to determine the current transition type and then applying it to the state change vector to find out how that impacts the state $x = x + v_j$

6. Update the current time with τ $t = t + \tau$
7. Append the updated time t and state x. Then return to point 2 until the decided end time (Gillespie, 2007).

This was done with a while loop that ran until the given final time was reached.

3.2 Explicit Tau leaping

The big different between SSA and Tau-leaping is the number of transitions occurring during one leap. Since Tau-leaping takes a bigger leap the variable K_j is created and indicates the number of times one transition type R_j happens during one leap. The probability that the number of transitions k_j of one transition type R_j happens during the leap $[t, t + \tau)$ for each type $j = 1, ..., k$, given that $X(t) = x$, can be denoted as $Q(k_1, ..., K_m|\tau; x, t)$.This means that Q is a joint probability density function of the M random variables, denoted as $K_j(\tau; x, t)$. This leads to K_j being a probability function that describes the number of times each transition type R_j occurs during the leap, given that $X(t) = x$. This is where the leap condition mentioned in the introduction matters, because given that the condition is followed the probability function Q can be estimated. Provided that the leap condition is meat the derivative of the rate function $a_j(x)dt$ gives the probability that the transition type R_j occurs within the time interval $[t, t + \tau)$. This leads to the random variable K_j following a Poisson distribution (Gillespie, 2001):

$$K_j(\tau; x, t) = Poisson(a_j(x), \tau) \qquad j = (1, ..., M) \tag{8}$$

The M random variables produced $K_j(\tau; x, t), ..., K_M(\tau; x, t)$ and since they are independent the joint density function Q can be estimated as the product of the individual K_j random variables density functions, denoted as follows (Gillespie, 2001):

$$Q(k_1, ..., k_M|\tau; x, t) = \prod_{j=1}^{M} P_{pois}(k_j; a_j(x), \tau) \tag{9}$$

It is now possible to take samples from the Poisson random variable. Making it possible to simulate the number of transitions from each transition type for every time leap, given that the leap condition is met. The closer the leap condition is met and the smaller the leap the more accurate the results will be. All the elements of explicit Tau-leaping are now known and the summarized algorithm looks as follows (Gillespie, 2001):

1. Set initial states $x_0 = x$ and $t_0 = t$.
2. Compute all rate functions $a_j(x)$ as well as their sum $a_0(x) = \sum_{j=1}^{M} a_j(x)$
3. Choose a valid τ, meaning a τ that makes the changes in the subset so small that the rate doesn't change significantly.
4. Use the Poisson random variable $K_j(\tau; x, t) = Poisson(a_j(x), \tau)$ to simulate the number of times k_j for each transition type $j = (1, ..., M)$.
5. If v_j is a vector of the change in each subset for all transition types. Then the change in each subset given the number of times each transition occurs can be calculated as $\sum_{j=1}^{M} k_j v_j$
6. Update all of the sub-populations with the calculated change, $x = x + \sum_{j=1}^{M} k_j v_j$
7. Update the time with τ, $t = t + \tau$ 8. Append updated t and x, then restart from 2.

3.3 Optimized Tau leaping

One problem that can arise when using Tau-leaping is negative sub-populations. Since negative populations are impossible in reality, it is important to ensure that they don't occur. However, it can be determined that if the leap condition is followed this would never happen. This is because a change from positive to negative in a sub-population size would always bring a significant change in the rate function a_j. If this happens the solution would be to make τ smaller, which makes the problem of using a constant τ obvious. One solution to this would be to check for negative subsets and if they occur try again with a smaller τ. This solution would quickly become inefficient, which is the opposite of the goal of using Tau-leaping. Further, it could quickly lead to accuracy problems. In that case, it might have been better to use SSA from the start (Cao et.al, 2005).

The idea behind the optimized Tau-leaping method is to use the fact that only small sub-populations can become negative in one leap. Since the random variable K_j used to compute the number of times each transition type happens is Poisson distributed, the probability that a subset becomes negative given that the leap condition is met can be calculated. The transition types able to produce negative sub-populations in one leap are now called critical. The common result is that a sub-population is at risk of becoming negative if it has between 2 and 20 individuals. The sub-population size for when a transition type could become critical is n_c and is chosen before the simulation. The goal now is to be able to check if a transition type is at risk of becoming critical during the next leap. This is done by calculating the upper limit of the number of possible transitions to occur before the sub-population becomes negative L_j. Where L_j is the upper limit for transition type j. The formula for L_j looks as follows:

$$L_j = \min_{i\in[1,N];(v_{ij}<0)}[\frac{x_i}{|v_{ij}|}] \tag{10}$$

If L_j is equal to or smaller than n_c, then R_j is a critical transition during the next time leap. It is important to note that the critical transitions can change after each leap since the sub-population sizes change. Therefore for every loop in the while loop, the critical transitions have to be checked. To summarize the point of the optimized Tau-leaping algorithm is to make sure no subset population becomes negative, by making sure that only one critical transition can occur during the time leap. In other words, the size of τ has to be so small that only one critical transition happens (Cao et.al, 2005).

To make Tau-leaping as efficient as possible the leap size τ should be as large as possible, while still being compatible with the leap condition (Cao et.al, 2006). The method for finding this value looks as follows:

$$tau = \min_{j\in[1,M]}\{\frac{\epsilon a_0(x)}{|\mu_j(x)|}, \frac{(\epsilon a_0(x))^2}{\sigma_j^2(x)}\} \tag{11}$$

$$\mu_j(x) = \sum_{j'=1}^{M} f_{jj'}(x)a_{j'}(x), \quad j = (1, ..., M) \tag{12}$$

$$\sigma_j^2(x) = \sum_{j'=1}^{M} f_{jj'}^2(x)a_{j'}(x), \quad j = (1, ..., M) \tag{13}$$

$$f_{jj'}(x) = \sum_{i=1}^{N} \frac{\partial a_j(x)}{\partial x_i} v_{ij}, \quad j, j' = (1, ..., M) \tag{14}$$

In this case, ϵ is an error control parameter and most often set to 0.03. It can then be shown that the mean of the estimated change in the rate function $a_j(x)$ during the time leap τ is $\mu_j(x)\tau$, with the standard deviation $\sqrt{\sigma_j^2(x)\tau}$. It follows that both the mean and standard deviation are bound by $\epsilon a_0(x)$ for all j. By setting j' to only run over the noncritical transitions, τ' can be chosen for those transitions. Then if $\tau' < \frac{1}{a_0(x)}$ SSA is run instead of tau leaping, meaning only one transition per leap is simulated (Cao et.al, 2006). Now all components of the optimized tau leaping method are known, the execution looks as follows:

1. Set the initial states to $x_0 = x$ and $t_0 = t$
2. Compute all rate functions $a_j(x)$ as well as the sum of them $a_0(x) =$

$\sum_{j=1} Ma_j(x)$
3. Using equation 10 calculate L_j for every transition type aka. the number of times the transition can occur before the subset becomes negative.
4. Identify critical transition types, meaning if $L_j \leq n_c$ and $a_j(x) > 0$.
5. Calculate the largest possible time step τ' that still follows the leap condition using equations $11 - 14$.
6. Check if $\tau' \leq \frac{1}{a_0(x)}$. If true stop the simulation and run the SSA several times.
7. Compute the second alternative for τ'' by sampling one integer from $exp(\frac{1}{a_0(x)})$.
8. If $\tau' < \tau''$ set $\tau = \tau'$. Set any critical transitions to 0 $k_j = 0$, meaning no critical transitions will happen. Compute the number of times other transition types occur k_j during the time leap from $Poisson(a_j(x)\tau)$
9. If $\tau'' \leq \tau'$ set $\tau = \tau''$. Use the random variable with the point probability $\frac{a_j(x)}{a_0(x)}$ to generate a random number j_c. j_c id is used to decide the critical reaction, then set $k_j c = 1$ and $k_j = 0$ for all other critical transitions. Meaning only one critical reaction occurs and only once. Simulate the number of times the noncritical transitions happen during τ using the random variable $Poisson(a_j(x)\tau)$.
10. Update the subsets $x = x + \sum_{j=1}^{M} k_j v_j$
11. Update the current time $t = t + \tau$
12. Check for any negative subsets, if true then start over with $\tau' = \frac{\tau'}{2}$
13. Append the updated x and t, then start over from step 2.

3.4 Implementation

Two models describing changes in different demographics were chosen to analyze the differences between the Tau-leaping methods in a field outside of chemistry. The first is the Lotka-Volterra model which describes the changes and dynamics in a population of animals, where one species is classified as prey and the other as predator. The second model is the SIR model, which describes the spread of diseases. The models will be simulated with the different Tau-leaping methods on two different computers for comparison of the run times. The simulations will be done in R-studio, using custom functions. The reason these two models were chosen is that they are very well-researched and readily used in their respective fields. These two facts make these models good for analysis because the result will be simpler to interpret and analyze.

3.4.1 Lotka-Volterra model

The Lotka-Volterra model also known as the predator-prey model was first introduced by Vito Volterra in 1926. The model aims to describe the changes in a population of animals, where one species is defined as prey and the other as a predator (Morrow et.al, 1987). The model is based on the following four assumptions:

- Limited to two species, one prey, and one predator
- The prey species has no limit of food
- The predator species can ONLY use the prey as its food source
- The rate at which the changes in the populations happen is directly proportional to the size of it

The model is based on the two following differential equations:

$$\frac{d}{dt}x = \alpha x - \beta xy \tag{15}$$

$$\frac{d}{dt}y = \delta xy - \gamma y \tag{16}$$

X defines the number of prey, Y the number of predators in the population and t the time. α is the fastest possible growth rate of the prey. β defines how the predators affect the growth rate of prey. δ defines the change in the predator growth rate based on the size of the population of prey. Finally, γ defines the death rate of the predators (Morrow et.al, 1987).

The parameters α, β, γ and δ can then be used to compute the rate function for each transition type. The three different transition types of the model and their effect on the populations are described in Table 1.

Table 1: Models transition types and their effect on the population sizes, through the matrix state change vector v_ij.

1. The population size of prey grows
2. The population size of predators grows
3. The population size of predators decreases

Transition type:	1	2	3
Predator	1	-1	0
Prey	0	1	-1

3.4.2 SIR model

The SIR (=Suspected, Infected, Recovered) model is used to model the spread of diseases, usually to learn more about how different epidemics spread. The SIR model is based on the following model of transmission rates (Rubin, M. D. et.al, 2021).

$$r(t) = \beta I(t) S(t)$$

Where β is a proportionality factor calculated for a case-by-case basis for each disease. For the SIR model to work it has to be assumed that all infected individuals make a full recovery. Then if $R(t)$ is the recovered individuals, $S(t) + I(t) + R(t)$ represents the whole population and is constant(Rubin, M. D. et.al, 2021). The differential equations for the SIR model then look like:

$$\frac{d}{dt} S(t) = -r(t) + \gamma R(t) \tag{17}$$

$$\frac{d}{dt} I(t) = r(t) - \beta I(t) \tag{18}$$

$$\frac{d}{dt} R(t) = \beta I(t) - \gamma R(t) \tag{19}$$

The immunity loss rate is regulated by γ and the recovery rate is regulated by β. These parameters can then be used in tau leaping to compute the rate functions. The SIR model has two different transition types displayed in Table 2 below (Rubin, M. D. et.al, 2021).

Table 2: Models transition types and their effect on the population sizes, through the matrix state change vector $v_i j$.

1. Someone gets infected
2. Someone recovers

Transition type:	1	2
Suspected	-1	0
Infected	1	-1
Recovered	0	1

4 Result

4.1 Lotka-Volterra model

The three simulations of the Lotka-Volterra model will be done using the initial variable values seen in Table 3, as well as the state change vector matrix seen in Table 2. Note that the unit of time is unspecified since that is usually the case for the Lotka-Volterra model.

Table 3: Initial value used for all tau leaping simulations of the Lotka-Volterra model.

Variables		
Start time	0	
Final time	3	
N		2000
	Predators	1000
	Prey	1000
Parameters		
	α	10
	β	0.01
	δ	10

4.1.1 SSA

The first simulation of the Lotka-Volterra model was done with SSA, hence no leaps were made. The number of transitions done was 289500 during the entire simulation time. The result of this simulation will be the foundation on which the results of the tau leaping methods will be judged.

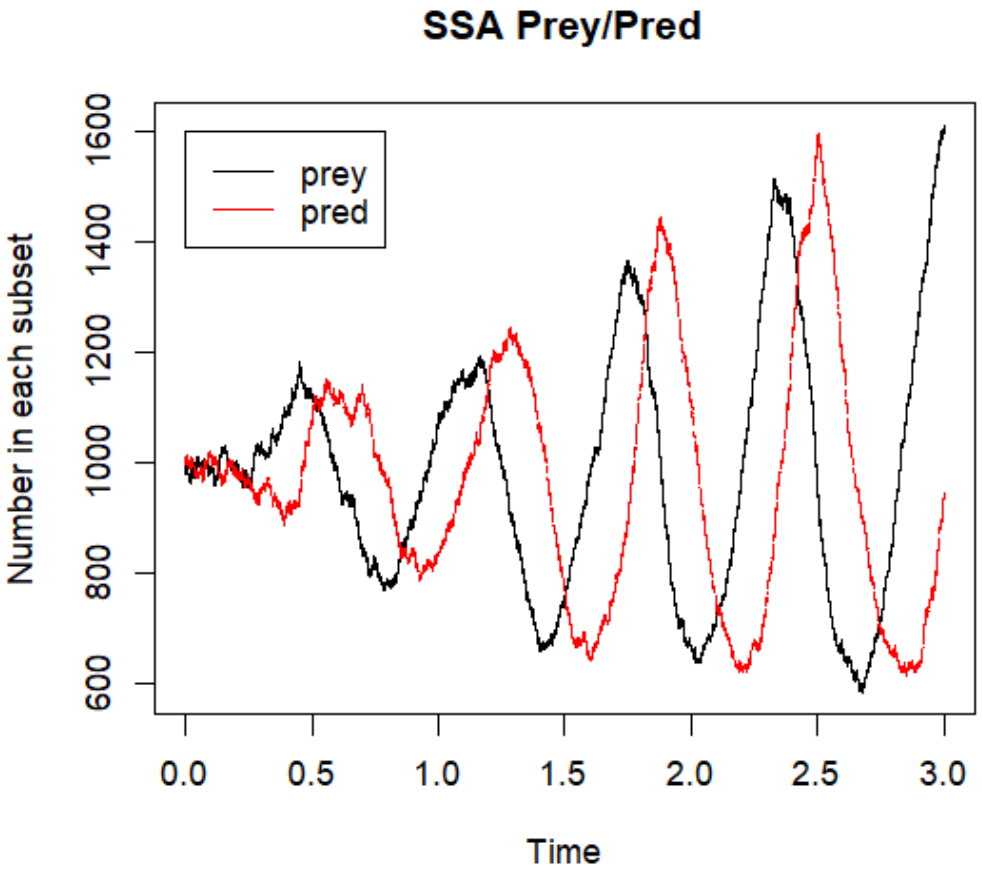

Figure 1: The simulated changes in a population of prey and predators using SSA.

Figure 1 shows how the sub-populations of prey and predators change and interact during the simulation time. The prey reaches its peak slightly before the predators and then starts going down again. It is also clear that the population size becomes bigger for every local max point.

4.1.2 Explicit tau leaping

As stated above the explicit tau leaping method uses a constant value for τ, in this case, $\tau = 0.002$. Because it is small enough to comply with the leap condition outlined in the introduction, while still being large enough to improve the efficiency.

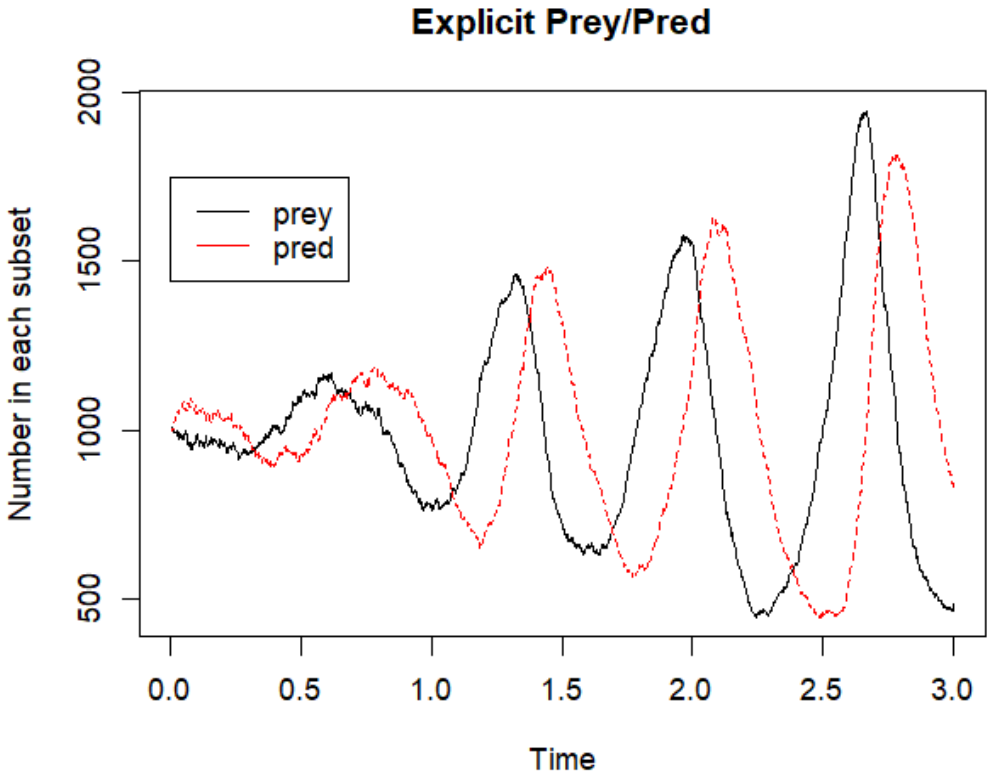

Figure 2: The simulated changes in a population of prey and predators using explicit tau leaping, with $\tau = 0.002$.

At first glance figure 2 looks fairly similar to figure 1, however, some precision has been lost which is expected. It also looks like figure 2 has larger sub-population sizes, than figure 1, the further into the simulation it gets. It might also be the case that the peeks of the population sizes appear a bit later in figure 2 than in figure 1. However, these differences will be further analyzed in the upcoming section 4.1.4.

4.1.3 Optimized tau leaping

As explained in section 3.3 τ gets chosen for each loop during the entire simulation. The mean of τ during the simulation was 0.00217, notably similar to the τ used for the explicit tau leaping.

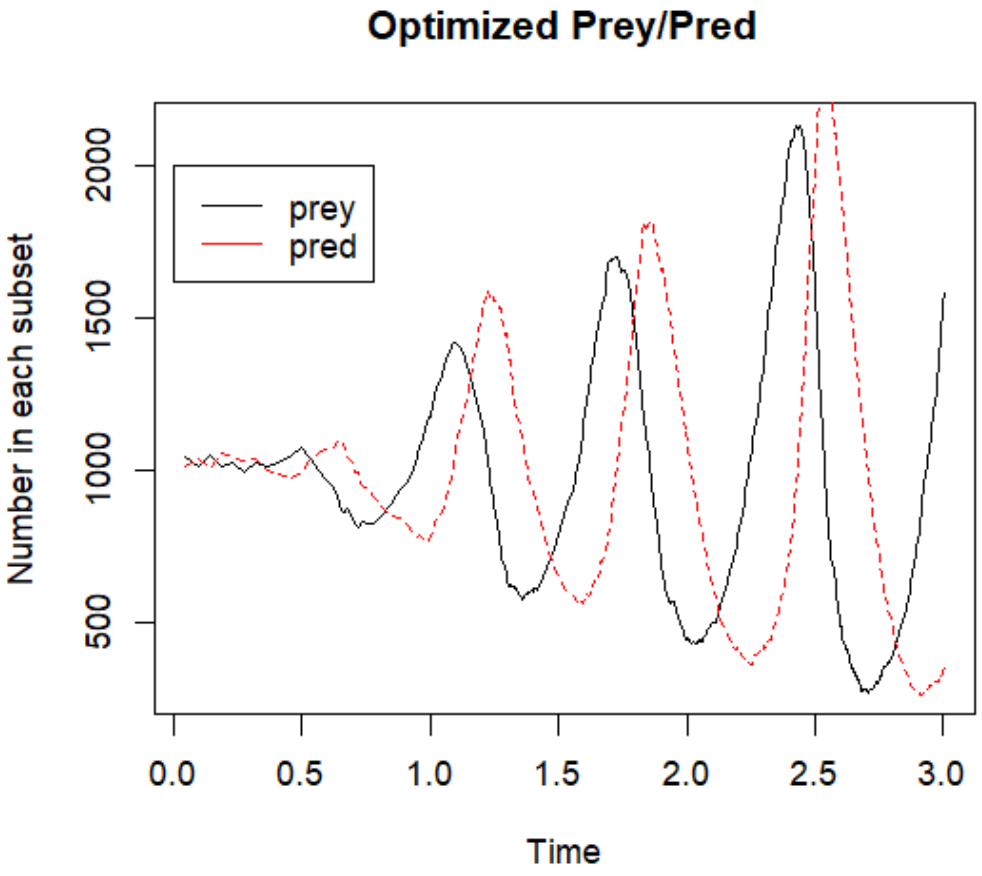

Figure 3: The simulated changes in a population of prey and predators using optimized tau leaping.

Figure 3 has roughly the same shape as both figure 1 and -2. However, figure 3 seems to have even bigger local max points than figure 2, which were already bigger than figure 1. Worth noting is that figure 3 seems to not peek further along the x-axis unlike figure 2.

4.1.4 Comparison

The simulations will be compared to analyze if the time gained from τ is worth the loss of precision in this case. Table 4 has some basic statistics from all simulations, followed by graphs comparing the results visually.

Table 4: Mean and median values from all simulations, as well as run time from two different computers.

		SSA	Explicit	Optimized
Run time 1 (s)		1541.10	1.96	0.98
Run time 2 (s)		1412.31	1.53	0.89
Num. of leaps		261486	4503	1383
Mean	Prey	1020.3	989.2	843.9
	Predator	1001.6	1012.705	1080.9
Median	Prey	1010	969	706
	Predator	986	1005	1023
Standard deviation	Prey	229.1	322.8	465.1
	Predator	226.6	313.5	533.2

For easier interpretation, 1541.10 seconds are roughly 26 minutes and 1412.31 seconds are roughly 24 minutes. The explicit simulation was around 800 times faster than SSA and the optimized simulation was around 1600 times faster than SSA. The SSA simulation made 261486 leaps, the explicit simulation made 4503 leaps and the optimized simulation made 1383 leaps. Which explains the significant time differences. The mean of SSA and explicit for both prey and predators are fairly similar while the mean of optimized is difference more. The optimized simulation also has a significantly larger standard deviation.

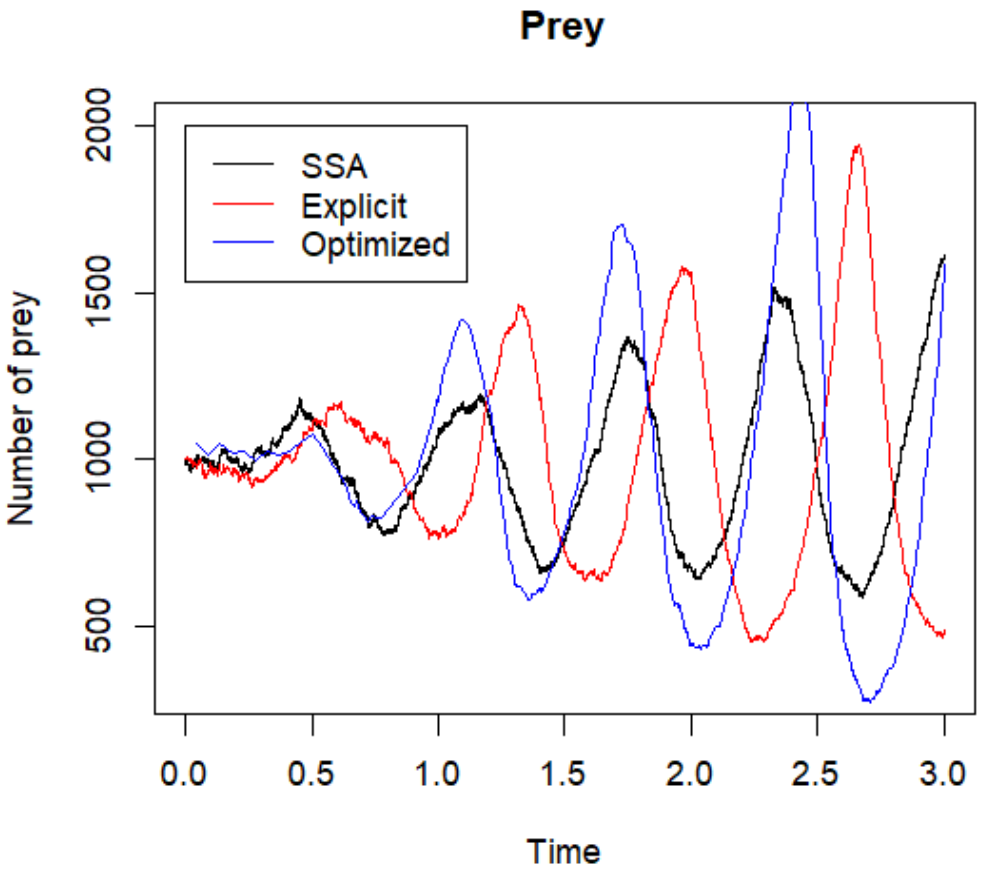

Figure 4: Comparison of the changes in prey simulated by the three tau leaping methods.

Figure 4 shows the difference between the simulation methods for the sub-population prey. The graph supports that explicit- and optimized tau leaping gave larger sub-population size peaks. It also shows that explicit tau leaping peaks slightly later in the simulation than both SSA and optimized tau leaping. This is also shown by the explicit tau leaping ending at a local min point compared to the other two.

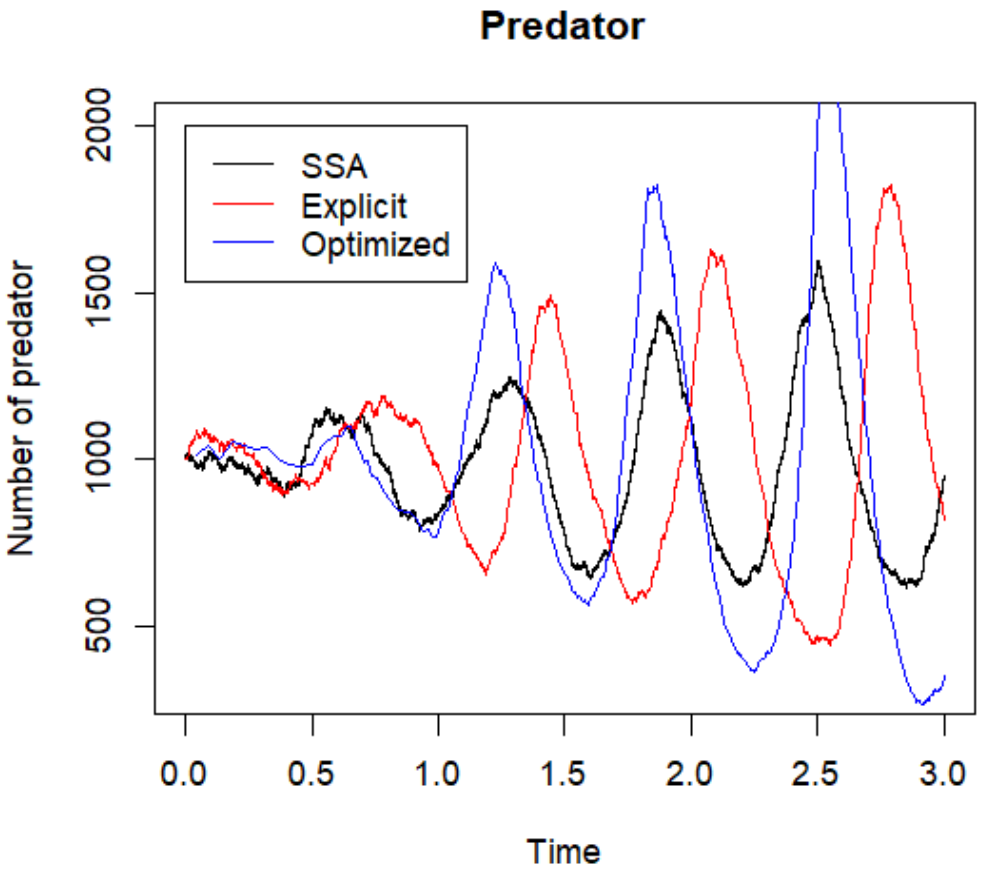

Figure 5: Comparison of the changes in predator simulated by the three tau leaping methods.

Figure 5, which shows the changes in predators, supports the conclusion that explicit- and optimized Tau-leaping produced larger peaks. As well as explicit Tau-leaping peaks occurring slightly later than the other twos.

The comparison shows that the optimized method follows the pattern of SSA more closely, while the explicit method follows the SSA more closely on the y-axis. It is worth noting that the optimized tau leaping was almost twice as fast as the explicit tau leaping. In this case, that difference is minimal but if a much larger simulation is run that difference could be significant. Since the purpose of the Lotka-Volterra model is to get an idea of how the sub-populations interact the preferred method in this case would be the optimized method.

4.2 SIR model

Three simulations of a SIR model with the initial variables seen in table 5 were run. Unlike the Lotka-Volterra model, the SIR model has days as a unit of time.

Table 5: Initial values used for all simulation methods of the SIR model.

Variables		
Start time	0	
Final time	150	
N		1010
	Suspected	1000
	Infected	10
Parameters	Recovered	0
	β	0.1
	γ	0.02

4.2.1 SSA

The simulation of the SIR model done with SSA will be the baseline for analyses of the two other methods. The number of transitions done with SSA was 289500.

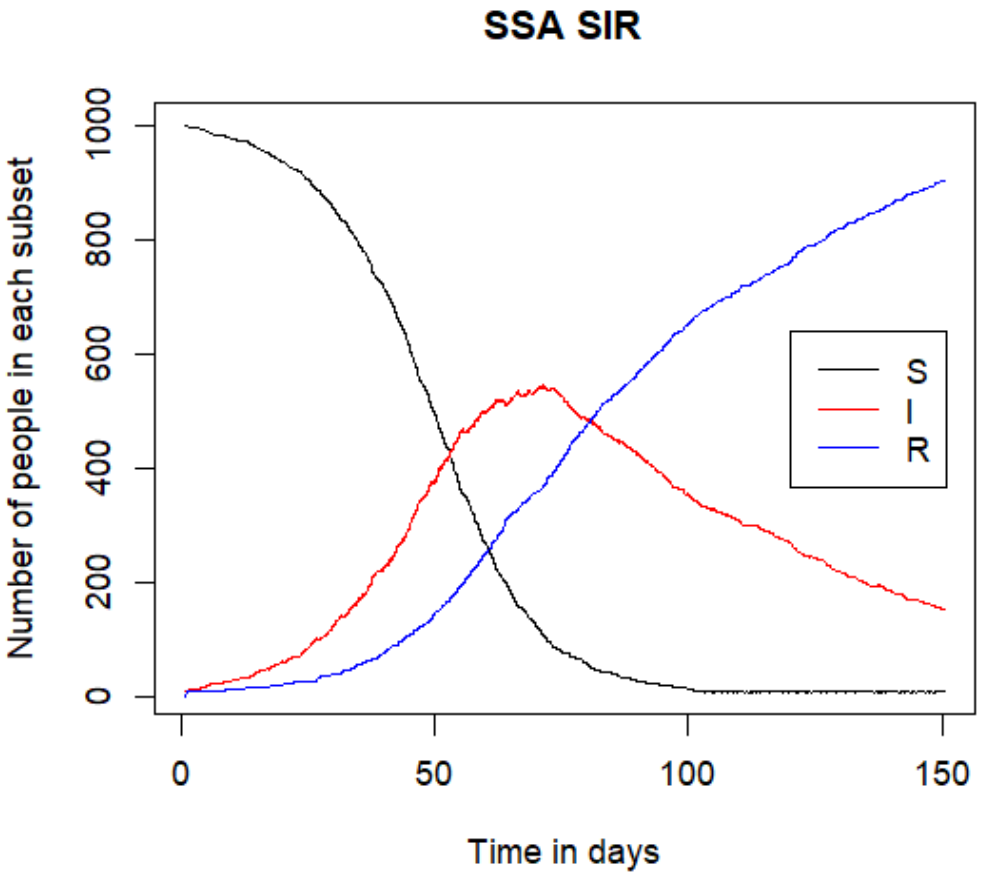

Figure 6: The simulated spread of a disease in a population using SSA.

Note that for all the graphs of the SIR model S stands for suspected, I for infected and R for recovered. Figure 6 shows the exact version of the SIR model simulation, this will later be compared to both the tau leaping methods.

4.2.2 Explicit tau leaping

The value of tau used for the explicit method was $\tau = 2$ since it is small enough to comply with the leap condition, while still being large enough to make it significantly more efficient.

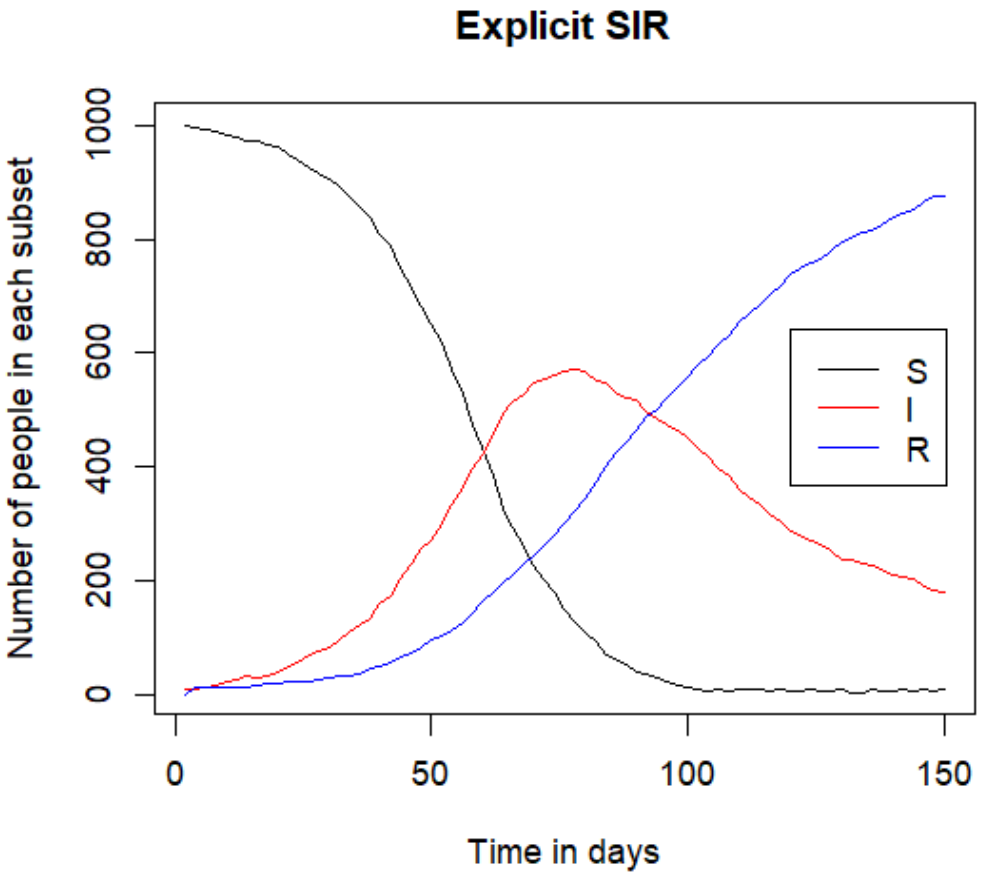

Figure 7: The simulated spread of a disease in a population using explicit tau leaping.

The difference between figure 6 and 7 is minimal and hard to see. The only obvious difference is that in figure 6 the lines show the exact transitions. In the upcoming comparison section, the differences between the figures will be examined.

4.2.3 Optimized tau leaping

As mentioned in section 3.3 the size of tau gets computed every leap for the entire simulation. The average *tau* of the simulation became 0.0135. The reason tau is so much larger than the chosen tau for the explicit method is that when one sub-population is very small the optimized algorithm makes tau significantly small. This results in the average tau being small, even though the value for tau gets as big as 4 when all the sub-populations don't risk becoming negative.

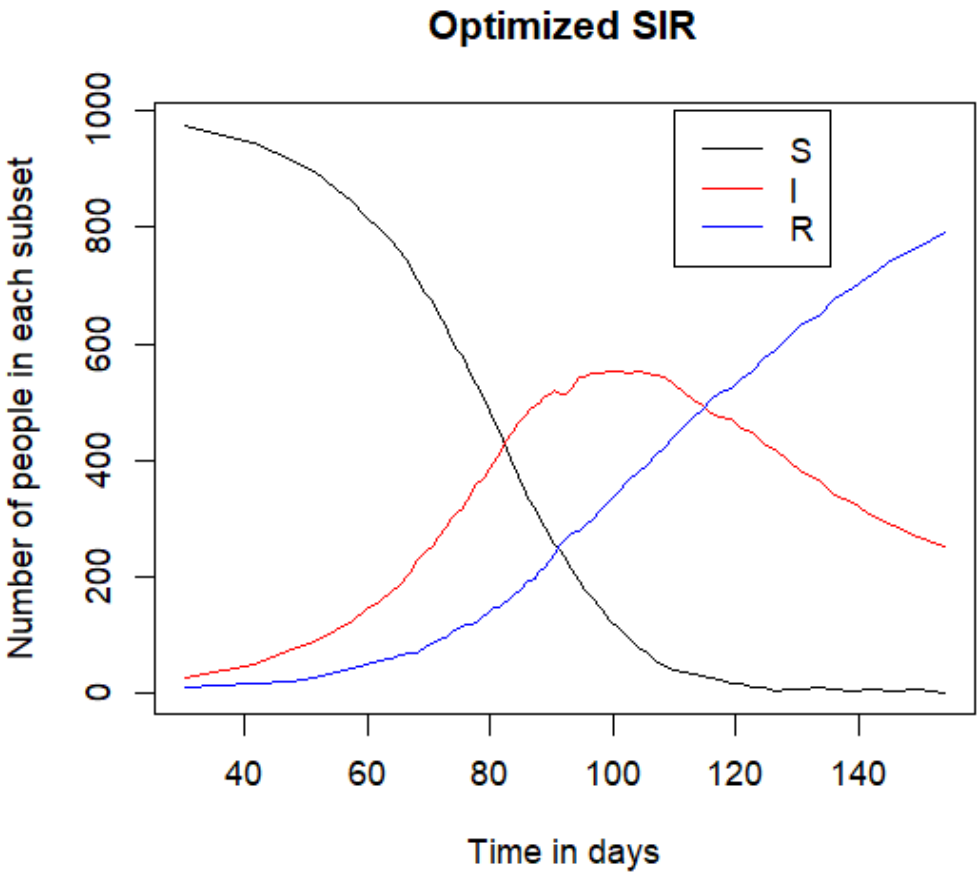

Figure 8: The simulated spread of a disease in a population using optimized tau leaping.

Figure 8 also looks similar to figures 6 and 7. One difference is that the line for the suspected population starts declining later than the SSA and explicit methods. The result is that the line for the recovered and infected populations starts increasing later.

4.2.4 Comparison

All simulation results will be compared, starting with some basic statistics in table 6.

Table 6: The simulated spread of a disease in a population using optimized tau leaping.

		SSA	Explicit	Optimized
Run time 1 (s)		2216.73	4.18	4.22
Run time 2 (s)		1831.19	4.10	4.17
Num. of leaps		289500	11250	11100
Mean	Suspected	350.2	371.1	341.1
	Infected	351	287.4	384.2
	Recovered	322.2	368.4	288.0
Median	Suspected	272.5	150	294.0
	Infected	375	270	421.5
	Recovered	248	302	213
Standard deviation	Infected	313.8	397.4	299.8
	Suspected	148.2	178.6	143.5
	Recovered	255.5	311.9	217.7

Table 6 shows the big difference between the number of leaps computed for each simulation, SSA computed around 25 times more leaps than the others. The result is that SSA was roughly 500 times slower than the optimized- and explicit method. The mean of all the simulations and all the sub-populations are fairly similar, while the medians diverge more from each other. Overall the optimized method has the smallest standard deviations while the explicit method has the biggest. Note that the means and medians can vary between the methods due to the difference in the number of leaps.

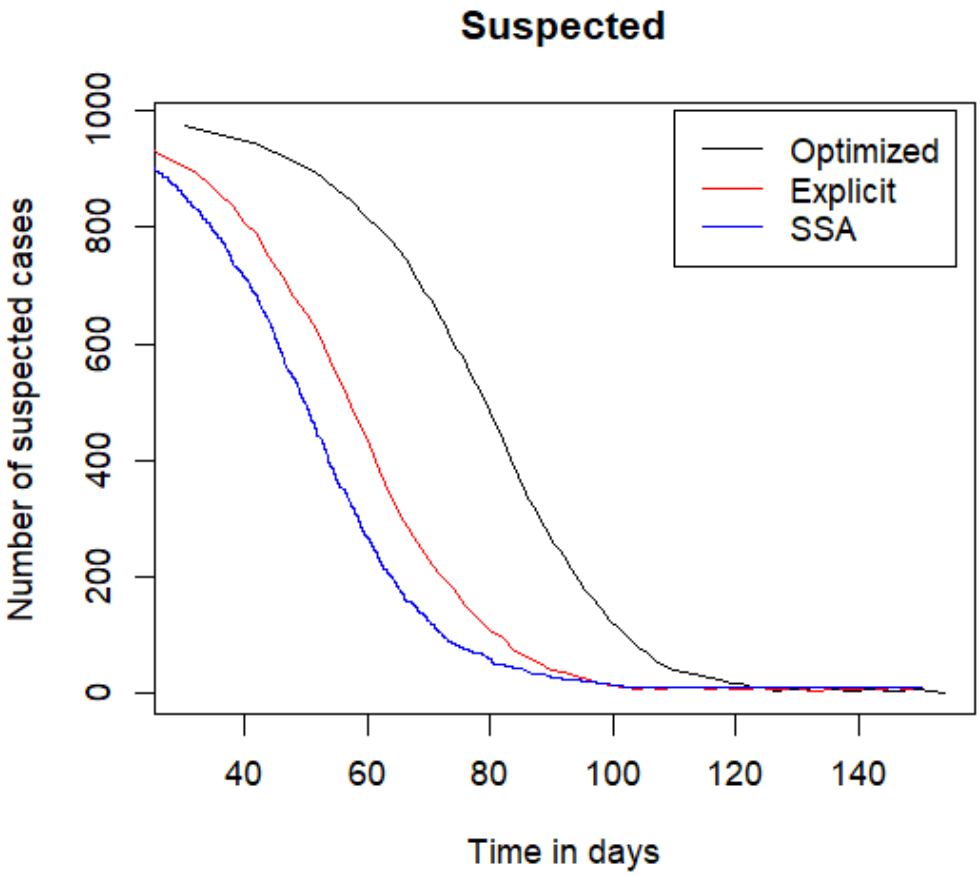

Figure 9: The three different simulations of the changes in the suspected sub-population.

As expected in section 4.2.3 the optimized method had the suspected population start to decrease later than SSA and explicit tau leaping. However, the inclination of the decrease seems fairly similar between all the simulations.

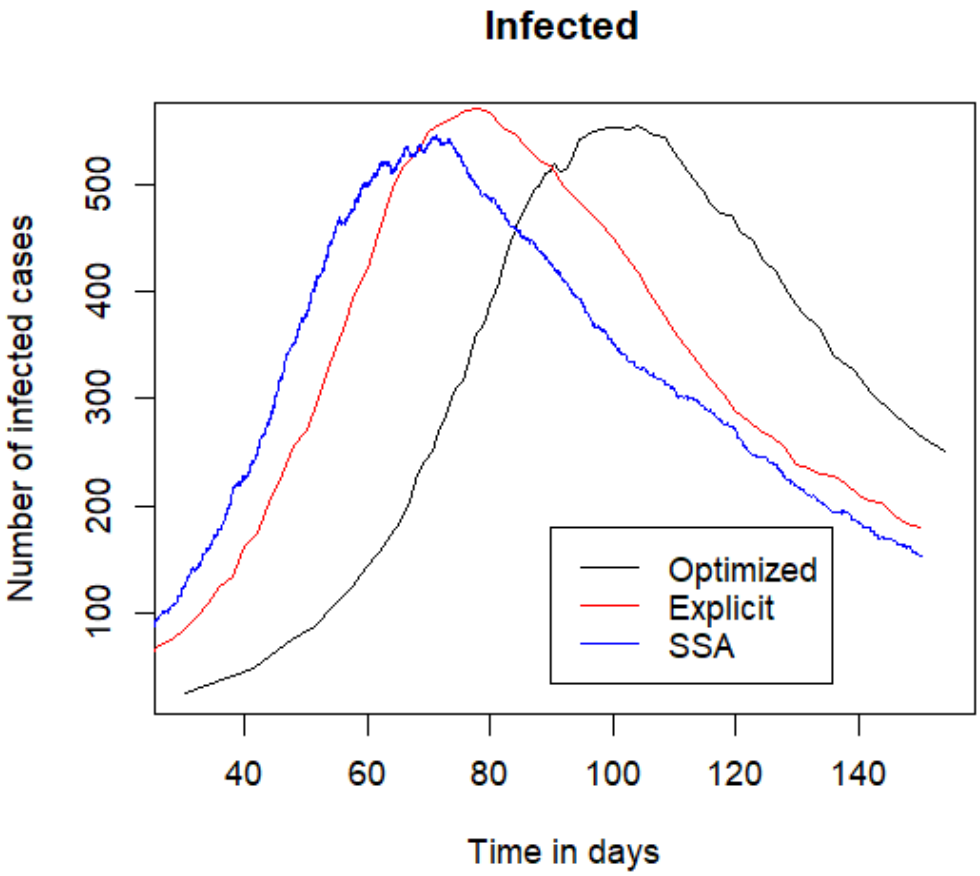

Figure 10: The three different simulations of the changes in the infected sub-population.

Figure 10 shows that the optimized method resulted in the curve occurring later than for the two other simulations.

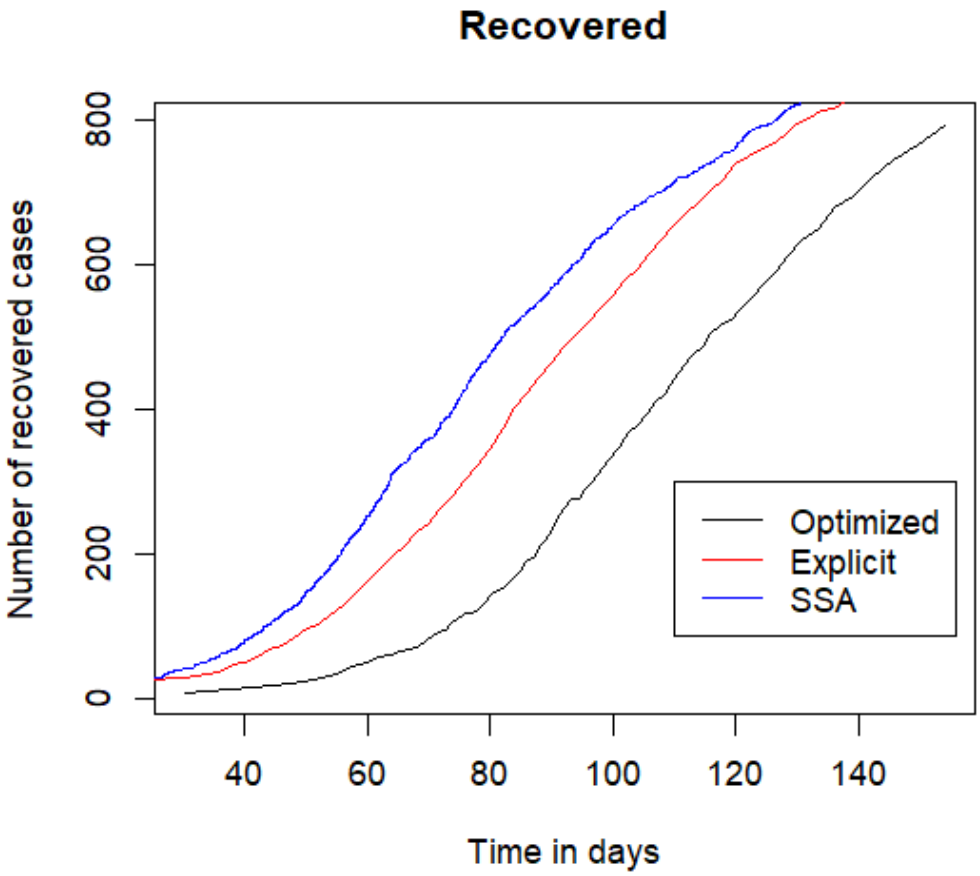

Figure 11: The three different simulations of the changes in the recovered sub-population.

The line representing the optimized method again starts to increase later than the others. However, just like in Figure 9, the inclination of the lines seems roughly the same.

This comparison concludes that the explicit method followed the SSA method the most, but the difference between them was relatively small. The biggest difference between the simulations was their run time, since the differences were very significant the slight loss of exactness could be worth the time saved.

5 Discussion

5.1 Result summarize

Both Tau-leaping methods produced significantly faster simulations of both models but suffered from some loss of accuracy. The problem with explicit Tau-leaping is the need to choose the leap size and it being constant. Since it is constant issues can arise if systems don't evolve at a constant rate. If the system changes quickly in some parts and slowly in others the method won't adapt. Leading to too big leaps being taken during fast changes and too small leaps being taken during slow changes. Resulting in either an inefficient simulation or an inaccurate one. This could be the reason for the explicit method computing a less accurate simulation for the Lotka-Volterra model. While the optimized Tau-leaping method leap size changes dynamically with the system, therefore the optimized method is preferable.

5.2 Run-time vs. Accuracy

As stated in the introduction one of the aims of this study was to analyse the possible improvements of simulation run-times. The result was staggering with improvements between 500 and 1600 times faster simulations. For a company doing multiple simulations, the time gained could be significant leading to cost savings. However, it has to be acknowledged that the results have a significant loss of accuracy. Depending on the goal of the simulation this could be an issue. If the simulations are done to make predictions the loss of accuracy could make them significantly less accurate. Firstly, by making a less accurate point prediction. Secondly, by making either larger- or inaccurate confidence intervals. However, many simulations are done to get an idea of how a process or system evolves over time. In this case, the loss of accuracy is less impactful, since the trends of the system stay mostly intact. Another way of looking at it is that the faster the run time the more simulations can be done. This could lead to a more accurate result. Since the standard deviation becomes smaller the more data is available. The conclusion then becomes that Tau-leaping is preferable depending on the wanted result of the simulation, especially if there are time- or cost limitations.

5.3 Suggested use cases

As stated before Tau-leaping is mostly used in chemistry, but could be used for other purposes especially if efficiency is a requirement. After doing this study we have some suggestions on other topics Tau-leaping could be implemented on. Models suited best for Tau-leaping are those that describe interactions between distinct sub-populations. This is why it would be great suited for any simulations of demographics, for example, a simulation of how information spreads in a population. As well as the evolution of population sizes. Another use case could be the spread and evolution of

mutations in a population. A completely different suggestion would be the simulation of the flow of traffic, strategies in game theory and controls of inventory.

5.4 Further research

If I were to continue studying Tau-leaping I would implement more methods for tau selection. Then analyze which methods are preferable to what kind of data or model. I would also analyze how much time could be saved for significantly larger simulations than presented in this report. If Tau-leaping were implemented for a company the time saved could be directly translated to a cost save. Another interesting research would be to analyze how accurate results could be if running the simulation multiple times. Then compare the result with SSA and determine if the time saved is worth it.

5.5 Conclusion

To conclude this report, Tau-leaping would be preferable over SSA in some cases, especially when prediction accuracy isn't a big issue. The time savings could be significant and lead to cost savings, without losing too much information about the system.

Efternamn, A. A. (År). Titel (Serie serienummer). Utgivare. URL

6 References

Gillespie, T. A. (2001). Approximate accelerated stochastic simulation of chemically reacting systems. (115, 1716–1733) The Journal of Chemical Physics.
m,

Gillespie, T. A. (2007). Stochastic Simulation of Chemical Kinetics (58:35–55). The Annual Review of Physical Chemistry.
http://home.thep.lu.se/ henrik/sysBioLund/2014/GillespieReview2007.pdf

Cao, Y., Gillespie, T. D. & Petzold, R. P. (2005). Avoiding negative populations in explicit Poisson tau-leaping (123, 054104). The Journal of Chemical Physics.
https://pubs.aip.org/aip/jcp/article-abstract/115/4/1716/451187/Approximate-accelerated-stochastic-simulation-of?redirectedFrom=fulltext

Cao, Y., Gillespie, T. D. & Petzold, R. P. (2006). Efficient step size selection for the tau-leaping simulation method (124, 044109). The Journal of Chemical Physics.
https://pubs.aip.org/aip/jcp/article-abstract/124/4/044109/562210/Efficient-step-size-selection-for-the-tau-leaping?redirectedFrom=fulltext

Morrow, J.(1987)The Lotka-Volterra Predator-Prey Model (675). UMAP.
https://staff.ulsu.ru/semoushin/_index/_pilocus/_gist/docs/mycourseware/1-basmod/5-assignments/group_projects/Group-project-assignments/Lotka-Volterra%20Model.pdf

Simon, A. R., Kulikova, A. Y., Dermitzakis, T. E. & Kondrashov, A. F. (2021) Rates of SARS-CoV-2 transmission and vaccination impact the fate of vaccine-resistant strains (15729). Scientific Reports.
https://www.nature.com/articles/s41598-021-95025-3

TWI. (n.d.). What is simulation? What does it mean? (Definition and examples). TWI.
https://www.twi-global.com/technical-knowledge/faqs/faq-what-is-simulation?fbclid= IwAR2L5V d9zxzRJunE6nGZlEPMgyeg9uicdalABo4_S_OoHCxoJnCZfw4XQwI#Limitations

Rubin, M. D., Achari, S., Carlson, S. C., Letts, F. R. R., Pantanowitz, A., Postema, M., Richaards, L. X. & Wigdorowitz1, B. (2021). Facilitating Understanding, Modeling and Simulation of Infectious Disease Epidemics in the Age of COVID-19 (9: 593417). Front Public Health.
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7907159/

Rathinam, M., Petzold, R. L., Cao, Y. & Gillespie, T. D.(2003) Stiffness in stochastic chemically reacting systems: The implicit tau-leaping method (119, 12784–12794). The Journal of Chemical Physics.
https://pubs.aip.org/aip/jcp/article-abstract/119/24/12784/185818/Stiffness-in-stochastic-chemically-reacting?redirectedFrom=fulltext

7 Appendix

Processor computer 1: AMD Ryzen 7 1800X 8-Core Processor 3.60 GHz
Processor computer 2: AMD Ryzen 5 3600 6-Core Processor 3.60 GHz