Chemical Reaction Modelling

Continuous Time Markov Chains, Stochastic Reaction Networks(SRNs)
and Simulations of SRNs

Introduction

Continuous-time Markov chains (CTMCs) are stochastic processes that follow the Markov property in which the system makes transitions in continuous time. Unlike discrete-time Markov chains where transitions can only take place at discrete time steps, CTMCs can transition through states at any point in time.

Stochastic Reaction Networks (SRNs) are a class of CTMCs that are used to model evolution of chemical systems where molecules of different chemical species can undergo a finite set of reactions [7]. SRNs are particularly useful when dealing with systems with small number of molecules as it helps in incorporating the inherent randomness of phase space of molecules in the evolution of the system.

In this chapter, we define CTMCs, SRNs, and discuss algorithms such as *Stochastic Simulation Algorithm* (SSA), *Next Reaction Method* (NRM), *Modified Next Reaction Method* (MNRM) and *tau-leap method*, that can be used to simulate SRN trajectories. SSA, NRM and MNRM are considered to be exact as the paths simulated by these methods follow correct statistical distributions. Tau-leap methods are used to simulate SRN paths using discrete time steps, in order to address some of the computational and performance related shortcomings of exact methods.

1 Continuous Time Markov Chains

Let $\mathbf{X} = \{X(t), t \in [0, \infty)\}$ be a continuous-time stochastic process with a countable state space $\mathbf{S}$. $\mathbf{X}$ is called a continuous-time Markov chain [8] if for any set of time indices $0 \leq t_1 < t_2 < \ldots < t_{s-1} < t_s < t$ and corresponding states $c_1, c_2, \ldots, c_{s-1}, c_s, c_t \in \mathbf{S}$ such that $P(X(t_1) = c_1, X(t_2) = c_2, \ldots, X(t_s) = c_s) > 0$, the following is true:

$$P(X(t) = c_t | X(t_1) = c_1, \ldots, X(t_s) = c_s) = P(X(t) = c_t | X(t_s) = c_s). \quad (1)$$

The property (1) is also called the Markov Property. We see from (1) that the probability of X being in a state c_t at time t is only conditioned on the state $X(t_s)$ at time index t_s, and does not depend on the states of the process in time indices

$t_1, t_2, \cdots, t_{s-1}$.

2 Stochastic Reaction Networks

An SRN $\mathbf{X}$ is a type of CTMC that describes evolution of a homogeneous chemical mixture of molecules in continuous time.

Given an underlying probability space $(\Omega, \mathcal{F}, P)$, $X : [0,T] \times \Omega \rightarrow \mathbb{Z}^N_{\geq 0}$ describes the time-evolution of a system of molecules belonging to N different species $(S_1, S_2, \ldots, S_N)$ in continuous time $t \in [0,T]$. Elements in $X(t) = (x_1(t), x_1(t), \ldots, x_N(t))$ indicate the abundances of the N species of molecules at time t. The molecules can undergo a finite set of reactions $\mathbf{R} = (\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_M)$, where a reaction $\mathcal{R}_| \in \mathbf{R}$ consists of a pair (ν_j, a_j). Each element in $\nu_j = (\nu_{j,1}, \nu_{j,2}, \ldots, \nu_{j,N}), 1 \leq j \leq, M$ corresponds to the change in the number of molecules of that particular species when the j^th reaction fires in the mixture of molecules. The vectors ν_j is also known as *stoichiometric vectors*. The *propensity* $a_j : \mathbb{Z}^N_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ of a reaction $\mathcal{R}_j$ depends on the state $X(t)$ at time t and describes the reaction rate of $\mathcal{R}_j$ as follows:

$$P(X(t + \Delta t) = x + \nu_j | X(t) = x) = a_j(x)\Delta t + o(\Delta t). \tag{2}$$

A larger value of $a_j(X(t))$ at time t indicates faster rate of reaction $\mathcal{R}_j$ at time t. Probability of firing reaction $\mathcal{R}_j$ in interval $(t, t + \Delta t]$ increases as $a_j(X(t))$ increases.

The probability that no reaction fires in the time interval $(t + \Delta t]$ in the SRN $\mathbf{X}$ is given by:

$$P(X(t + \Delta t) = x | X(t) = x) = 1 - \sum_{j=1}^{M} a_j(x)\Delta t + o(\Delta t). \tag{3}$$

Defining a new quantity $a_0(x) := \sum_{j=1}^{M} a_j(x)$, (3) can be rewritten as:

$$P(X(t + \Delta t) = x | X(t) = x) = 1 - a_0(x)\Delta t + o(\Delta t). \tag{4}$$

Given $X(t) = x$ at time t, it can be derived from (4) that the time to fire next reaction follows an exponential distribution with parameter $a_0(x)$.

The stochastic mass-action kinetics principle can be represented by the diagram

given below for some reaction $\mathcal{R}_| \in \mathbf{R}$:

$$\alpha_{j,1}S_1 + \ldots + \alpha_{j,n}S_n \rightarrow \beta_{j,1}S_1 + \ldots + \beta_{j,n}S_n. \tag{5}$$

As implied by (5), when a reaction $\mathcal{R}_j$ is fired, $\alpha_{j,1}, \cdots, \alpha_{j,N}$ molecules of the corresponding species $S_1, \cdots, S_N$ are consumed and $\beta_{j,1}, \cdots, \beta_{j,N}$ molecules are produced. The stoichiometric vector $\nu_j := (\beta_{j,1} - \alpha_{j,1}, \ldots, \beta_{j,n} - \alpha_{j,n})$ can contain both positive and negative integers, and hence $\nu_j \in \mathbb{Z}$.

Given a positive reaction constant c_j and state $X(t) = x = (x_1, \ldots, x_N)$ at time t, the propensity function $a_j(x)$ of reaction $\mathcal{R}_j$ as derived from mass-action kinetics is given by:

$$a_j(x) = c_j \prod_{i=1}^{N} \frac{x_i!}{(x_i - \alpha_{j,i})!} \mathbf{1}_{\alpha_{j,i} \leq x_i} \tag{6}$$

where $\alpha_{j,i}$ is the number of S_i molecules consumed when reaction R_j is fired and $\mathbf{1}_{\alpha_{j,i} \leq x_i}$ is an indicator function that can be described as below:

$$\mathbf{1}_{\mathbf{A}} = \begin{cases} 0, & \text{if } \mathbf{A} \textit{ is false}, \\ 1, & \text{if } \mathbf{A} \textit{ is true}. \end{cases}$$

In the remaining literature, (6) will be used to calculate propensities of reactions given the state $X(t)$ at time t.

Random Time Change Representation

Let x_0 be the initial state of an SRN $\mathbf{X}$ with reaction channels $(\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_M)$. The random time-change representation of $\mathbf{X}$ as shown by T.G. Kurtz [6, 2, 5] is given as:

$$X(t) = x_0 + \sum_{j=1}^{m} \nu_j Y_j \left(\int_0^t a_i(X(s))ds \right), \tag{7}$$

where $Y_j : \mathbb{R}_{\geq 0} \times \Omega \rightarrow \mathbb{Z}_{\geq 0}$ are independent unit-rate Poisson processes. The representation (7) models the path of the process $\mathbf{X}$ in continuous time as a combination of paths of M independent unit-rate Poisson processes. Each reaction is represented by a unit-rate Poisson process, and for $1 \leq j \leq M$, the value $\int_0^t a_j(X(s))ds$ is called the internal time of Y_j.

Later in the literature, we will see how *Next Reaction Method*(NRM) and *Modified Next Reaction Method*(MNRM) make use of (7) to simulate exact paths of SRNs by sampling next-reation times and next-reaction internal times respectively.

Example - Gene transcription and translation

For illustration, we look at an example of an SRN that models gene transcription and translation. The SRN, say $\mathbf{X}$, consists of three species (M, P, D) and five reactions as shown below:

- $\Phi \to M$, Transcription of a gene into mRNA.
- $M \to M + P$, Translation of mRNA into proteins.
- $P + P \to D$, Production of stable *Dimers* from proteins.
- $M \to \Phi, P \to \Phi$, Degradation of mRNA and proteins respectively.

From the above reactions, we can construct ν_j for each reaction as follows:

$$\nu_1 = \begin{bmatrix}1\\0\\0\end{bmatrix}, \quad \nu_2 = \begin{bmatrix}0\\1\\0\end{bmatrix}, \quad \nu_3 = \begin{bmatrix}0\\-2\\1\end{bmatrix}, \quad \nu_4 = \begin{bmatrix}-1\\0\\0\end{bmatrix}, \quad \nu_5 = \begin{bmatrix}0\\-1\\0\end{bmatrix}.$$

The propensity functions of the five reactions as functions of the state $X(t) = (m(t), p(t), d(t))^T$ at some time t are given by:

$$a_1(X(t)) = 25, a_2(X(t)) = 10^3 m(t), a_3(X(t)) = 10^{-3} p(t)(p(t) - 1),$$
$$a_4(X(t)) = 0.1m(t) \text{ and } a_5(X(t)) = p(t).$$

If the initial state $X(0) = x_0$, the process can be modeled with mass action kinetics as:

$$\begin{aligned} X(t) = x_0 &+ Y_1\left(\int_0^t 25ds\right)\begin{bmatrix}1\\0\\0\end{bmatrix} + Y_2\left(\int_0^t 10^3 m(s)ds\right)\begin{bmatrix}0\\1\\0\end{bmatrix} \\ &+ Y_3\left(\int_0^t 10^{-3}p(s)(p(s)-1)ds\right)\begin{bmatrix}0\\-2\\1\end{bmatrix} + Y_4\left(\int_0^t 0.1m(s)ds\right)\begin{bmatrix}-1\\0\\0\end{bmatrix} \\ &+ Y_5\left(\int_0^t p(s)ds\right)\begin{bmatrix}0\\-1\\0\end{bmatrix} \end{aligned}$$

Where Y_js are unit-rate Poisson processes with internal times given as $\int_0^t a_j(X(s))ds$.

3 Simulation of SRNs

Analytical and numerical solutions to differential equations involving SRNs are usually infeasible. Some challenges met while finding analytical solutions for an SRN, lets say $\mathbf{X}$, include: i. non-linearity of propensity functions a_js w.r.t. states of $\mathbf{X}$, ii. large volume of state space of $\mathbf{X}$, to name a few. For this reason, simulating paths of SRNs using Monte Carlo method can help us circumvent these challenges and find approximate values of several quantities of interest including but not limited to time taken by process $\mathbf{X}$ to reach a specified subset of its state space and expected value of an observable g at some time T, i.e. $E[g(X(T))]$.

3.1 Exact Algorithms

Exact algorithms are methods that can be used to simulate paths of an SRN $\mathbf{X}$ that obey the probabilities described by (2) and (3). These paths have correct statistical distributions and hence can be used to estimate values such as expected value of an observable g at time T, i.e. $E[g(X(t))]$. We will discuss three exact methods: 1. *Stochastic Simulation Algorithm* (SSA), 2. *Next Reaction Method* (NRM) and 3. *Modified Next Reaction Method* (MNRM) [3, 1].

3.2 Stochastic Simulation Algorithm(SSA)

In [4], SSA was popularised by Gillepsie to simulate paths of SRNs that could be used to describe evolution of chemical reactions. The algorithms involves sampling two random numbers at each step: i. First random number is used to sample the time to next reaction, lets say h, and ii. Second random number is used to sample which reaction takes place after time h. The algorithm is based on the an observation from (2) that, given $X(t) = x$ at time t, the probability of $\mathcal{R}_j$ being the only reaction that takes place in time interval $(t, t + h)$ is given by $a_j(x) \times exp(-a_0(x)h)$ which can be split and rewritten as $(a_j(x)/a_0(x)) \times a_0(x)exp(-a_0(x)h)$.

As observed from (4), given $X(t) = x$ at time t, time to next reaction has an exponential distribution with parameter $a_0(x)$. Hence, the probability of firing reaction $\mathcal{R}_j$ in time interval $(t, t+h)$ can be split into two independent probabilities: probability of selecting reaction $\mathcal{R}_j$ from set of all the reactions, and probability density that follows an exponential distribution with parameter $a_0(x)$.

SSA involves the following sequence of steps to produce an exact simulation of a path of an SRN $\mathbf{X}$:

Algorithm: Stochastic Simulation Method
{Initialise state, time}
Initialise $\mathbf{x} = x_0$, $\mathbf{t} = 0$
{Declare array to store propensities}
Declare array **a**
{Declare constant time limit, array of stoichiometric vectors}
Declare constant T, array ν
while $\mathbf{t} < \mathbf{T}$ **do**
  for each reaction-channel **i do**
    Calculate propensity **a**[**i**] using x
  end for
  Set $\mathbf{a_0} = \sum_j \mathbf{a}[\mathbf{j}]$
  Generate a random number $\mathbf{r_1}$ from $unif(0, 1)$
  {Calculate time to next reaction using $\mathbf{r_1}$}
  Set $\mathbf{h} = (1/\mathbf{a_0})ln(1/\mathbf{r_1})$
  Generate a random number $\mathbf{r_2}$ from $unif(0, 1)$
  {Decide which reaction to fire based on probabilities proportional to **a**[**j**]}
  Find m such that $\sum_{k=1}^{m-1} \mathbf{a}[\mathbf{k}] < \mathbf{a_0 r_2} \leq \sum_{k=1}^{m} \mathbf{a}[\mathbf{k}]$
  Set $\mathbf{x} = \mathbf{x} + \nu[\mathbf{m}], \mathbf{t} = \mathbf{t} + \mathbf{h}$
end while

It can be observed from the algorithm that for each reaction that is fired in the simulation, two random numbers are being generated, first to sample the time to next reaction and second to sample the reaction that should be fired based on their propensities. We will see in the following sections that NRM and MNRM require generation of only one random number each time a reaction is fired.

3.3 Next Reaction Method(NRM)

The representation (7) of SRN paths as shown by Kurtz is given as, assuming SRN **X** is initially at state x_0:

$$X(t) = x_0 + \sum_{j=1}^{m} \nu_j Y_j \left(\int_0^t a_i(X(s))ds \right).$$

The representation (7) separates the randomness of the system from the state of the system, as the randomness is only contained in the Y_js which are unit-rate independent Poisson processes that count the number of times the corresponding reactions $\mathcal{R}_j$ fired until time t. The intensities of the Y_js are given by $a_j(X(t))$. Hence, each Y_j comes with its own time-frame, apart from the actual time frame of the chemical system. We can define $T_j(t) = \int_0^t a_j(X(s))ds$, to represent the

internal times of the corresponding Y_j. Using internal times $T_j(t)$ in simulating SRN paths requires solving two crucial challenges: i. how to sample firing times of each Y_j in its own time frame and ii. how to translate these firing times to the actual time of the chemical system. We will see that NRM and MNRM try to use solve these challenges in two different ways but both essentially use the same property of separation of randomness from the state of the system as suggested in (7).

For an SRN with M reactions, NRM first calculates the firing times of all the reactions, say $\delta t_j, j \leq M$ at time $t = 0$ by sampling from exponential distributions with internal times of Y_js as parameters, based on the assumption the a_j remain unchanged until j^th reaction is fired. The reaction with the least firing time is fired first. After firing a reaction, the change in state of the system leads to change in propensities of the reaction-channels. This change is propensities requires the firing times to be updated, but the internal times of Y_j until their next firing remain the same. In the subsequent steps, the reaction with the nearest firing time is fired, and the firing times of the rest of the reactions are updated accordingly. For the reaction that got fired, new reaction time is sampled based on the updated propensity. The simulation is carried out until it is explicitly stopped or time limit of the simulation is reached.

Lets say, at some time t, we know $X(t)$, internal times $T_j = T_j(t)$ and propensities $a_j = a_j(X(t))$. Assuming a_j remains constant over the next firing time of Y_j, we can also derive δt_j for each reaction. Now, the next reaction after time t that has to be fired should be the one with the minimum δt_j. Let us denote this value, i.e. $min_j \delta t_j$ as Δ, and let μ be the value of j with minimum δt_j. The system can then be updated to $\bar{t} = t + \Delta$ and all the propensities can be updated to $\bar{a}_j = a_j(X(\bar{t}))$. For $j = \mu$, next firing time should be sampled using the updated propensity $a_\mu(X(\bar{t}))$. As the propensity functions have changed for all the reactions, the next firing time would not be same as old firing times for $j \neq \mu$. But it is to be noted that the internal times of the reactions until their next firing remain the same. Internal times that have passed until $\bar{t}$ are given by $T_j(\bar{t}) = T_j(t) + a_j\Delta$. Hence the amount of internal time that should pass before firing Y_j is given by

$$(T_j(t) + a_j \delta t_j) - (T_j(t) + a_j \Delta) = a_j(\delta t_j - \Delta).$$

Also, we note that the difference in internal times between t and $\bar{t}$ for Y_j are given by $\bar{a}_j \delta \bar{t}_j$. Hence, equating the differences between internal times for reactions, we get

$$\delta \bar{t}_j = \frac{a_j}{\bar{a}_j}(\delta t_j - \Delta).$$

NRM generates M random numbers during initialisation to sample the firing times

of all the reactions. After the first step, it requires only one random variable per firing to sample the next firing time of the reaction that fired in the current step. NRM algorithm involves following sequence of steps:

Algorithm: Next Reaction Method
{Initialise state, time}
Initialise $\mathbf{x} \leftarrow x_0$, $\mathbf{t} \leftarrow 0$
{Declare array to store propensities, firing times}
Declare array **a**, array τ
{Declare constant array of stoichiometric vectors, time limit}
Declare constant T, array ν
for each reaction-channel **i do**
  Calculate propensity **a**[**i**] using x
  Generate random number **r** from $unif(0,1)$
  Set $\tau[\mathbf{i}] = (\mathbf{1}/\mathbf{a}[\mathbf{k}])\mathbf{ln}(\mathbf{1}/\mathbf{r})$
end for
while $\mathbf{t} < \mathbf{T}$ **do**
  Set $\mathbf{t_{next}} = min_j \tau[\mathbf{j}]$, $\mu = argmin_j \tau[\mathbf{j}]$
  Set $\mathbf{x} = \mathbf{x} + \nu[\mu], \mathbf{t} = \mathbf{t_{next}}$
  for each reaction-channel **i do**
    Calculate propensity $\overline{\mathbf{a}}[\mathbf{i}]$ using x
    if $\mathbf{i} \neq \mu$ **then**
      Set $\tau[\mathbf{i}] = \mathbf{t} + (\mathbf{a}[\mathbf{i}]/\overline{\mathbf{a}}[\mathbf{i}])(\tau[\mathbf{i}] - \mathbf{t})$
    end if
  end for
  Generate a random number **r** from $unif(0,1)$
  Set $\tau[\mu] = \mathbf{t} + (\mathbf{1}/\overline{\mathbf{a}}[\mu])\mathbf{ln}(\mathbf{1}/\mathbf{r})$
  for each reaction-channel **i do**
    Set $\mathbf{a}[\mathbf{i}] = \overline{\mathbf{a}}[\mathbf{i}]$
  end for
end while

Unlike SSA where two random variables were required in each step, NRM requires only one random number per firing of a reaction.

3.4 Modified Next Reaction Method(MNRM)

MNRM is similar to NRM in the way that it uses Kurtz' representation of evolution of X to simulate paths of $\mathbf{X}$, but unlike NRM, it does not require time conversions and works explicitly with internal times to calculate next firing times of reactions. At some time t, let P_j denote the first firing internal time of Y_j that is strictly larger than T_j: $P_j = min_s\{s > T_j : Y_j(s) > Y_j(T_j)\}$, where T_js are the internal

times of Y_js until t. Hence the next firing time of the reaction can be given as:

$$\delta t_j = (P_j - T_j)/a_j.$$

The internal time until the next firing of a reaction Y_j should remain constant irrespective of the evolution of the system until Y_j is fired, only the actual firing time changes based on the changes in propensities. MNRM uses this fact and lets us sample internal times, here P_j, directly instead of sampling actual next times of reactions. MNRM algorithm is as follows:

Algorithm: Modified Next Reaction Method
{Initialise state, time, internal times arrays P and T}
Initialise $\mathbf{x} = x_0$, $\mathbf{t} = 0$, array $\mathbf{P} = 0$, array $\mathbf{T} = 0$
{Declare array to store propensities, and delta-time to store simulated next firing time}
Declare array $\mathbf{a}$, array $\delta\mathbf{t}$
{Declare constant array of stoichiometric vectors, time limit}
Declare constant T, array ν
for each reaction-channel **i do**
  Calculate propensity **a**[**i**] using **x**
  Generate random number **r** from $unif(0, 1)$
  Set $\mathbf{P}[\mathbf{i}] = (\mathbf{1}/\mathbf{a}[\mathbf{i}])\mathbf{ln}(\mathbf{1}/\mathbf{r})$
end for
while $\mathbf{t} < \mathbf{T}$ **do**
  for each reaction-channel **i do**
    Set $\delta\mathbf{t}[\mathbf{i}] = (\mathbf{P}[\mathbf{i}] - \mathbf{T}[\mathbf{i}])/\mathbf{a}[\mathbf{i}]$
  end for
  Set $\Delta = min_i \delta\mathbf{t}[\mathbf{i}]$, $\mu = argmin_i \delta\mathbf{t}[\mathbf{i}]$
  Set $\mathbf{t} = \mathbf{t} + \mathbf{\Delta}, \mathbf{x} = \mathbf{x} + \nu[\mu]$
  for each reaction-channel **i do**
    Set $\mathbf{T}[\mathbf{i}] = \mathbf{T}[\mathbf{i}] + \mathbf{a}[\mathbf{i}]\mathbf{\Delta}$
  end for
  Generate a random number **r** from $uniform(0, 1)$
  Set $\mathbf{P}[\mu] = \mathbf{P}[\mu] + \mathbf{ln}(\mathbf{1}/\mathbf{r})$
  for each reaction-channel **i do**
    Calculate **a**[**i**] using **x**
  end for
end while

MNRM calculates M random numbers during initialisation to sample the internal times of first firings of all the Y_js, and subsequently uses only one random number per iteration to re-sample the internal time of the fired reaction. MNRM and NRM

differ in the way they use random numbers. MNRM utilizes random numbers to sample the internal times until the first firing, eliminating the need for time-conversions as in NRM.

4 Approximate Algorithms - Tau Leap Algorithm

Exact algorithms, although being useful in simulating paths of SRNs with correct distributions, may not be feasible for many reasons. High propensities can lead to lower and lower reaction times, and hence increase computation load of the algorithms. Approximate algorithms mitigate these challenges by simulating paths of SRNs over discrete time steps. One such algorithm is the Tau-leap algorithm [7].

In this algorithm, a small time interval τ is fixed and a path of an SRN $\mathbf{X}$ is simulated by calculating $X(t)$ at $t = 0, \tau, 2\tau, \ldots, N\tau$ where $N\tau$ is the time limit of the simulation. After k time-steps, when $t = k\tau$, the algorithm progresses by sampling the number of each kind of reaction that would take place in the interval $(k\tau, (k+1)\tau]$ and update the state of the system accordingly. Let the number of reactions that take place in the interval $(k\tau, (k+1)\tau]$ for M reaction channels be $(\mu_{k+1,1}, \mu_{k+1,2}, \ldots, \mu_{k+1,M})$, then the system is updated as follows:

$$X((k+1)\tau) = X(k\tau) + \sum_{j=1}^{M} \mu_{k+1,j}\nu_j.$$

If the initial state of the SRN is $X(0) = x_0$, and if the time interval $(0, t]$ is partitioned into N time intervals, $(0, s_1], (s_1, s_2], \ldots, (s_{N-1}, s_N]$, the integral $\int_0^t a_j(X(s))ds$ in Kurtz's representation (7) can be approximated as a summation over discrete time intervals so that we get

$$\int_0^t a_j(X(s))ds \approx \sum_{q=0}^{N-1} a_j(X(s_q))(s_{q+1} - s_q).$$

Let us represent the path simulated by approximate methods by $\overline{X}(t)$ to differentiate it from exact paths. The approximate path $\overline{X}$ that can be modeled by approximating the integrals in random time change representation (7) is thus given as:

$$\overline{X}(t) = x_0 + \sum_{j=1}^{m} Y_j \left(\sum_{q=0}^{N} a_j(\overline{X}(s_q))(s_{q+1} - s_q) \right) \nu_j. \tag{8}$$

While simulating $\overline{X}(t + \tau)$ given the state $\overline{X}(t)$, the second term in (8) becomes

$Y_j\left(\sum_{q=0}^{N} a_j(\overline{X}(s_q))(s_{q+1}-s_q) + a_j(\overline{X}(t))\tau\right)$. Y_j becomes a Poisson process with internal time as sum of two terms. Hence, Y_j can be split into two terms: $Y_j\left(\sum_{q=0}^{N} a_j(\overline{X}(s_q))(s_{q+1}-s_q)\right)$ and a Poisson random Variable $\mathcal{P}_j(a_j(\overline{X}(t))\tau)$. The first term is already known as it was used to arrive at the state $\overline{X}(t)$. Hence the expression for $\overline{X}(t+\tau)$ becomes:

$$\overline{X}(t+\tau) = \overline{X}(t) + \sum_{j=1}^{m} \mathcal{P}_j(a_j(x)\tau)\nu_j,$$

where $\mathcal{P}_j(\lambda_j), j = 1, 2, \ldots, M$, are independent Poisson processes with intensity λ_j.

Tau-Leap algorithm involves the following sequence of steps:

Algorithm: Tau-Leap Algorithm
{Initialise state, time}
Initialise $\mathbf{x} = x_0$, $\mathbf{t} = 0$
{Declare array to store propensities, random Poisson variables}
Declare array **a**, array **p**
{Declare constant time-step τ_0, time limit, array of stoichiometric vectors}
Declare constant $\tau_\mathbf{0}$, **T**, array ν
while t $<$ **T do**
  for each reaction-channel **i do**
    Calculate propensity **a**[**i**] using **x**
    Generate random number **p**[**i**] from Poisson distribution $\mathcal{P}(\mathbf{a}[\mathbf{i}]\tau)$
  end for
  Set $\mathbf{t} = \mathbf{t} + \tau, \mathbf{x} = \mathbf{x} + \sum_{\mathbf{j=1}}^{\mathbf{M}} \mathbf{p}[\mathbf{j}]\nu[\mathbf{j}]$
end while

References

[1] David F. Anderson. A modified next reaction method for simulating chemical systems with time dependent propensities and delays. *Department of Mathematics, University of Wisconsin-Madison*, 2007. Received 20 June 2007; accepted 26 September 2007; published online 6 December 2007; publisher error corrected 28 January 2008.

[2] S. Ethier and T. Kurtz. *Markov Processes: Characterization and Convergence (Wiley Series in Probability and Statistics)*. Wiley-Interscience, 2nd edition, 2005.

[3] M. A. Gibson and J. Bruck. Efficient exact stochastic simulation of chemical

systems with many species and many channels. *The Journal of Physical Chemistry A*, 104(9):1876–1889, 2000.

[4] D. T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, 22:403–434, 1976.

[5] H. Koeppl, D. Densmore, G. Setti, and M. di Bernardo, editors. *Design Analysis of Biomolecular Circuits*. Springer, 2011.

[6] T. G. Kurtz. *Approximation of Population Processes (CBMS-NSF Regional Conference Series in Applied Mathematics)*. Society for Industrial and Applied Mathematics, 1987.

[7] Alvaro Moraes. *Simulation and Statistical Inference of Stochastic Reaction Networks with Applications to Epidemic Models*. Doctoral dissertation, Name of the University, 2023.

[8] G. George Yin and Qing Zhang. *Continuous-Time Markov Chains and Applications: A Two-Time-Scale Approach*, volume 37 of *Stochastic Modelling and Applied Probability*. Springer, 2006. New chapters added on backward equations and LQG control problems.