Introduction

- In our last lecture, you learned how to compose a system's chemical master equation and exploit it to determine something about the state probabilities, including means, variances, and (estimated) distributions of the variables.
- These properties of the probability distribution, however, do not provide us with sample trajectories. If there are multiple behaviors occurring, the probability distribution and its moments will only give us information about them in aggregate.
- Today we will introduce the Gillespie algorithm, one of the most famous stochastic simulation techniques. Before beginning, however, it is necessary to introduce some statistics to motivate how the algorithm works.

Point processes

A point process can be thought of as a "random set of points" in either time or space: the number of points as well as their time/location vary between realizations. Point processes are appropriate descriptors of many biological phenomena with discrete events, including action potentials, births/deaths, and chemical reactions. Because of the variable number of "points" in a realization, their probability distributions are described a bit differently from the usual. We must define a series of functions $P_s(\tau_1, \ldots, \tau_s)$, each of which represent the probability that s events occurred *and* they occurred at the times τ_i. If the τ_i are ordered, then the normalized probability distribution may be written as:

$$P_0 + \int_{-\infty}^{\infty} P_1(\tau_1)\, d\tau_1 + \int_{-\infty}^{\infty} \left[\int_{\tau_1}^{\infty} P_2(\tau_1, \tau_2)\, d\tau_2 \right] d\tau_1 + \ldots = 1$$

This normalization sums over all the possible numbers of events and all of the combinations of τ_i that can occur for a given number of events. Suppose we relax the restriction that the τ_is be ordered: then, for any set $\{\tau_1, \ldots, \tau_s\}$, there are $s!$ permutations. If we require that each function P_s is symmetric in its variables, then we can rewrite the normalization above in a handier form:

$$P_0 + \sum_{s=1}^{\infty} \frac{1}{s!} \int_{\mathbb{R}^s} P_s(\tau_1, \ldots, \tau_s)\, d\tau_1 \ldots d\tau_s = 1 \tag{1}$$

Notice that we have accounted for the expansion of the limits of integration by dividing each term in the sum by the number of possible permutations of the τ_i.

A natural first question is to ask of a point process is how to determine the average number of events that occur within an interval (t_a, t_b).To address this, we introduct the indicator function

$$f(t) = \begin{cases} 1 & : t \in (t_a, t_b) \\ 0 & : \text{otherwise} \end{cases}$$

Then if s events occur at times τ_k, the number that occur within this interval is $N = \sum_{k=1}^{s} f(\tau_k)$. The average number of events within this interval is therefore:

$$\begin{aligned} \langle N \rangle &= \left\langle \sum_{k=1}^{s} f(\tau_k) \right\rangle \\ &= \sum_{s=1}^{\infty} \frac{1}{s!} \int_{\mathbb{R}^s} \left(\sum_{k=1}^{s} f(\tau_k) \right) P_s(\tau_1, \ldots, \tau_s)\, d\tau_1 \cdots d\tau_s \end{aligned}$$

To simplify this expression, we note that we can rewrite it as follows:

$$\begin{aligned}\langle N\rangle &= \sum_{s=1}^{\infty}\frac{1}{s!}\sum_{k=1}^{s}\int_{\mathbb{R}^s} f(\tau_k)P_s(\tau_1,\ldots,\tau_s)\,d\tau_1\cdots d\tau_s \\ &= \sum_{s=1}^{\infty}\frac{1}{s!}\left[\int_{\mathbb{R}^s} f(\tau_1)P_s(\tau_1,\ldots,\tau_s)\,d\tau_1\cdots d\tau_s + \int_{\mathbb{R}^s} f(\tau_2)P_s(\tau_1,\ldots,\tau_s)\,d\tau_1\cdots d\tau_s + \ldots\right]\end{aligned}$$

Since we have required that the functions P_s be symmetric in their variables, each of these integrals is equal:

$$\begin{aligned}\langle N\rangle &= \sum_{s=1}^{\infty}\frac{1}{s!}s\int_{-\infty}^{\infty} f(\tau_1)\left[\int_{\mathbb{R}^{s-1}} P_s(\tau_1,\ldots,\tau_s)\,d\tau_2\cdots d\tau_s\right]d\tau_1 \\ &= \sum_{s=1}^{\infty}\frac{1}{(s-1)!}\int_{t_a}^{t_b}\left[\int_{\mathbb{R}^{s-1}} P_s(\tau_1,\ldots,\tau_s)\,d\tau_2\cdots d\tau_s\right]d\tau_1\end{aligned}$$

In the second line, we have used the definition of the indicator function to change the limits of integration for one variable.

Poisson processes

Consider the special case where events occur independently of one another, e.g., the fact that one event occurred at time $t = \tau_1$ does not influence the probability that another event occurred at time $t = \tau_2$. This idealization is not well-suited to action potentials in neurons owing to their refractory period, but is appropriate for the chemical reactions which we would like to simulate. In this special case, the P_s can be factorized in the form

$$P_s(\tau_1,\ldots,\tau_s) = e^{-\nu}P(\tau_1)\cdots P(\tau_s), \qquad \nu = \int_{-\infty}^{\infty} P(\tau)\,d\tau$$

where the normalization constant follows from equation 1. Plugging this into the formula above allows us to simplify somewhat:

$$\begin{aligned}\langle N\rangle &= e^{-\nu}\sum_{s=1}^{\infty}\frac{1}{(s-1)!}\int_{t_a}^{t_b}\left[\int_{\mathbb{R}^{s-1}} P(\tau_1)\cdots P(\tau_s)\,d\tau_2\cdots d\tau_s\right]d\tau_1 \\ &= e^{-\nu}\sum_{s=1}^{\infty}\frac{1}{(s-1)!}\int_{t_a}^{t_b} P(\tau_1)\left[\int_{\mathbb{R}^{s-1}} P(\tau)\,d\tau\right]^{s-1}d\tau_1 \\ &= e^{-\nu}\int_{t_a}^{t_b} P(\tau_1)\,d\tau_1\left(\sum_{s=1}^{\infty}\frac{1}{(s-1)!}\left[\int_{-\infty}^{\infty} P(\tau)\,d\tau\right]^{s-1}\right) = e^{-\nu}\int_{t_a}^{t_b} P(\tau_1)e^{\nu}\,d\tau_1 \\ &= \int_{t_a}^{t_b} P(\tau)\,d\tau \qquad (2)\end{aligned}$$

It is no harder to show by a similar method that for this type of point process,

$$\sigma_N^2 = \left\langle N^2\right\rangle - \langle N\rangle^2 = \langle N\rangle$$

As you know, equivalence of the mean and variance are properties of the Poisson distribution. To learn more about the probability distribution of N, the number of events occurring in an interval (t_a, t_b), we will find its characteristic function. A characteristic function $G(k)$ for a variable x is defined as the average value of e^{ikx}:

$$G(k) = \left\langle e^{ikx}\right\rangle = \int_{-\infty}^{\infty} e^{ikx}P(x)\,dx$$

For a random variable x that can only take on integer values,

$$P(x = n) = \sum_{n\in\mathbb{Z}} p_n \delta(x - n) \implies G(k) = \sum_{n\in\mathbb{Z}} p_n e^{ikn} \tag{3}$$

Our approach will be to find an expression for the characteristic function of N and compare the powers of e^{ikn} to find an expression for p_n, the probability that $N = n$:

$$\begin{aligned}
\left\langle e^{ikN} \right\rangle &= \sum_{s=0}^{\infty} \frac{1}{s!} \int_{\mathbb{R}^s} e^{ikN} P_s(\tau_1, \ldots, \tau_s)\, d\tau_1 \ldots d\tau_s = \sum_{s=0}^{\infty} \frac{1}{s!} \int_{\mathbb{R}^s} e^{ik\sum_{j=1}^{s} f(\tau_j)} P_s(\tau_1, \ldots, \tau_s)\, d\tau_1 \ldots d\tau_s \\
&= e^{-\nu} \sum_{s=0}^{\infty} \frac{1}{s!} \int_{\mathbb{R}^s} e^{ikf(\tau_1)} P(\tau_1) \cdots e^{ikf(\tau_s)} P(\tau_s)\, d\tau_1 \ldots d\tau_s = e^{-\nu} \sum_{s=0}^{\infty} \frac{1}{s!} \left[\int_{-\infty}^{\infty} e^{ikf(\tau)} P(\tau)\, d\tau \right]^s \\
&= \exp\left[\int_{-\infty}^{\infty} e^{ikf(\tau)} P(\tau)\, d\tau - \int_{-\infty}^{\infty} P(\tau)\, d\tau \right] = \exp\left[\left(e^{ik} - 1\right) \int_{t_a}^{t_b} P(\tau)\, d\tau \right] \\
&= \exp\left[\left(e^{ik} - 1\right) \langle N \rangle \right] = e^{-\langle N \rangle} \sum_{k=0}^{\infty} \frac{\langle N \rangle^k e^{ikN}}{k!}
\end{aligned} \tag{4}$$

Comparing equations 3 and 4, we find that:

$$p(N = n) = \sum_{n=0}^{\infty} \frac{\langle N \rangle^n e^{-\langle N \rangle}}{n!}, \qquad \text{which is the Poisson probability density function with } \lambda = \langle N \rangle$$

It is hopefully now clear why point processes with independent events are called Poisson processes.

Waiting times

We now know enough about the statistics of chemical reactions to begin calculating quantities necessary for our simulations. In particular, we will want to know how much time passes between reactions, i.e., the waiting time. Since nothing of interest is happening during the waiting time, the chemical reaction rate has some constant value k during this period. The average number of events we expect in some period $(0, t)$ over which k is effectively unchanged will be given by equation 2:

$$\langle N \rangle = \int_0^t k\, d\tau = kt$$

If we denote the time at which the first reaction occurs by T, then

$$\begin{aligned}
P(T > t) &= P(\text{zero reactions occur in the time interval } [0,t]) \\
&= \frac{(kt)^0 e^{-kt}}{0!} = e^{-kt}
\end{aligned}$$

This gives us a formula for the cumulative density function of T:

$$P(T < t) = 1 - e^{-kt}$$

To find the probability density function, we take the derivative of the cumulative density function with respect to t:

$$P(T = t) = ke^{-kt}$$

This means that T follows an *exponential distribution*. If we wanted to simulate when the next reaction would occur, we could just draw a random number from an exponential distribution (which

MATLAB and any other self-respecting statistics package will allow us to do). Be wary, however, as some statistics packages – including MATLAB – define exponential distributions like this:

$$P(X = x) = \frac{e^{-t/\lambda}}{\lambda}$$

so double-check whether you need to take the reciprocal when supplying the parameter to the exponential random number generator.

To simulate systems with more than one reaction, we need to know when the next of many possible reactions will happen. (We also want to know which type of reaction it was, but as we will discuss later, we can worry about that problem separately.) In other words, if each reaction i is occuring at a known rate k_i, the we want to know the distribution of times until the next reaction occurs, i.e., the distribution of $T_0 = \min_i T_i$, where each T_i is exponentially-distributed with its own parameter k_i. Since all reactions are assumed to occur independently,

$$P(T_0 > t) = P(T_1 > t) \cdots P(T_m > t) = \prod_i \exp(-k_i t) = \exp\left(-t \sum_i k_i\right)$$

Following the same math we used above to convert this to a probability density function, we find that the time until the first reaction occurs (T_0) is therefore exponentially-distributed with parameter $k = \sum_i k_i$. In other words, once we have calculated the rates for all reactions in the system, we can simulate the time at which the next reaction will occur by drawing a number from an exponential distribution with parameter $\sum_i k_i$.

Now we return to the problem of deciding which reaction will occur next. Reaction i is next if $T_i = T_0 < \min_{j \neq i} T_j$. By ansatz, I'll provide the solution:

$$P(T_i = T_0) = \frac{k_i}{\sum_j k_j}$$

We'll prove this by induction, a method you'll use on problem 2b of problem set 7. The idea is to prove that the statement is true for a "base case," then show that if it is true for a certain number of reactions $n - 1$ then it is true for n reactions as well. By extension the formula will hold for any number of reactions greater than the base case. For our base case, we consider a system with one reaction – then:

$$P(T_i = T_0) = \frac{k_1}{k_1} = 1$$

That was the easy part. Next, we assume that the formula has been found to hold for a system with $n-1$ reactions. Let's define $Y_0 = \min_{i<n} T_i$ to be the minimum time until one of the first $n-1$ reactions occurs We know from the discussion above that Y_0 is exponentially-distributed with effective rate

$k = \sum_{i=1}^{n-1} k_i$. We want to know the probability that the nth reaction occurs first, i.e.:

$$\begin{aligned} P(T_n = T_0) = P(T_n < Y_0) &= \int_0^\infty \int_0^{Y_0} k k_n e^{-k_n T_n} e^{-k Y_0} \, dT_n \, dY_0 \\ &= k k_n \int_0^\infty e^{-k Y_0} \left[-\frac{e^{-k_n T_n}}{k_n} \right]_0^{Y_0} dY_0 \\ &= k \int_0^\infty e^{-(k+k_n) Y_0} \, dY_0 - k_n \int_0^\infty e^{-k Y_0} dY_0 \\ &= 1 - \frac{k}{k + k_n} = \frac{k_n}{k + k_n} = \frac{k_n}{\sum_{i=1}^n k_i} \end{aligned}$$

This completes the induction step. To simulate which reaction will occur next, we normalize the rates to k_i^* so that their total sum is one, then pick a random number uniformly-distributed between 0 and 1. This number will fall into a "bin" ℓ whose endpoints are $(\sum_{i=1}^{\ell-1} k_\ell^*, \sum_{i=1}^{\ell} k_\ell^*)$: we will therefore say that reaction ℓ has occurred. In implementation, the bins can remain figurative, since we can call the MATLAB function `randsample()` with our array of k_is to get a random reaction index.

Gillespie's Stochastic Simulation Algorithm (SSA)

We now have all of the information we need to motivate the Gillespie SSA. In his original paper/presentation (1976), Gillespie proposed two alternative methods, then showed that their assumptions were equivalent (though one algorithm, the "direct method," was more efficient). I will present first the more intuitive "first reaction method".

For both methods, the simulation begins with specified initial numbers of all molecular species, an initial time (wlog, $t = 0$), and a termination condition to prevent the algorithm from going on indefinitely. Most folks store e.g. the times and concentrations in a large array with e.g. one row per n rounds: in this case it is wise to pre-allocate the array in memory and to set the termination condition to when the array will be full. If you write directly to disk you might easily choose a more intricate termination condition.

What happens in each round, however, differs between the two methods. Each round of the first reaction method simulation has four main parts:

1. After checking the termination condition, the rates for this time step are calculated: these typically depend on the current molecule numbers but may also be time-dependent (for example, we can cause production of a molecule to begin at time t_0).
2. For each reaction i, a time T_i until its next reaction is drawn from an exponential distribution with rate k_i.
3. The smallest of the T_is is identified: this T_i is the time elapsed during the step, and the reaction corresponding to this T_i is the one which occurs in this round.
4. The current time t and the number of molecules of each species are updated accordingly. Commonly a stoichiometry matrix is used to facilitate the molecule count updates. The updated values are stored if desired. The remaining T_is and the calculated rates k_i are erased. (At least some of them will not apply in the next round since some molecule numbers will have changed.)

Many people find this algorithm intuitive. But as I hinted above, we can find the time until the first of n reactions occurs without having to generate all of the T_i and calculate their minimum. Furthermore, the time until the next reaction occurs can be selected independently of selecting which reaction occurs next! To see why, consider the probability that the next reaction is i *and* this reaction occurs at time t. The first reaction method asserts that this probability is:

$$\begin{aligned} P(\text{reaction } i \text{ next, at time } t) &\propto P(T_i = t) \prod_{j \neq i} P(T_j > t) \\ &= k_i e^{-k_i t} \prod_{j \neq i} e^{-k_j t} \\ &= k_i \exp\left(-t \sum_i k_i\right) \end{aligned}$$

The direct method, by contrast, assumes that the probabilities of the two events are independent and follow the distributions we spent time calculating earlier in this lecture:

$$\begin{aligned} P(\text{reaction } i \text{ next, at time } t) &\propto P(\text{reaction } i \text{ next}) \times P(\text{next at time } t) \\ &= \left(\frac{k_i}{\sum_i k_i}\right) \times \left[\sum_i k_i\right] \exp\left[-t \sum_i k_i\right] \\ &= k_i \exp\left(-t \sum_i k_i\right) \end{aligned}$$

The results, of course, are equivalent. We will see that the direct method, however, requires fewer random number generations and comparison operations. The four main steps in each round of the direct method are (omitting unnecessary detail given above):

1. After checking the termination condition, the rates for this time step are calculated.
2. A time until the next reaction occurs is randomly generated from an exponential distribution with parameter $\sum_i k_i$.
3. The next reaction to occur is chosen using `randsample()` (or similar) as described above.
4. The current time t and the number of molecules of each species are updated accordingly.

Supplemental: Next Reaction Method

- One of the more time-consuming steps in the Gillespie algorithm is the update of rates with each time step. Typically most of the rates will not change at all, and the few that do change won't change by much, in between two time steps.
- The *next reaction method* is a modification of the first reaction method. However, it attempts to retain the reaction rates and randomly-generated time of the next reaction (τ_i) wherever possible, i.e., whenever the rate of a reaction is not affected by the reaction that occurred last.
- You should be wary of reusing random numbers in general, but for this particular method it is proven to be legitimate (Gibson and Bruck, 2000). The next reaction method uses absolute time rather than relative time.
- To improve efficiency, the suggested implementation uses data structures you may not have encountered previously. These are:

  - A dependency graph, indicating which reactions needs to have their rates/times updated if reaction i occurs because the concentration of one or more reactants has changed (usually represented by a simple adjacency matrix), and
  - An indexed heap (priority queue) allows the smallest τ_i to be easily identified, and makes updating the τ_is rather fast.

- The algorithm initiates by setting $t = 0$, specifying the initial molecule numbers, determining the initial rates, and generating the τ_is.

- In each step:

  1. After checking the termination condition, the reaction with the smallest τ_i is picked off the heap and chosen to be the reaction that occurs next.
  2. $\mathbf{X}$ and t are updated based on the reaction that occurs.
  3. The dependency graph is consulted to determine which rates need to be recalculated and which τ_is need to be regenerated.