Lecture 10: Stochastic Reaction Equations

Stochastic reaction equations can be used to calculate interactions between molecules, complexes, cells, organisms, or any other species of object that is present in integer numbers and can react to change the number of each species. We represented the number of objects as discrete random variables.

Discrete random variables and probability distributions

If X can certain values, such as integer values, within a finite or infinite interval, than X is a **discrete random variable**. We refer to little x as a specific realization (value) of the random variable X. We define the **probability mass function** $p(x)$ as the probability that $X = x$. For a discrete random variable, $p(x)$ is unitless. Because the sum of all probabilities is always 1, we know the following, which we can often use to scale p(x):

$$\sum_{-\infty}^{\infty} p(x) = 1$$

We can also define the cumulative distribution function $P(a) = prob(X \leq a) = \sum_{-\infty}^{a} p(x)$. While $p(x)$ can have any shape, $P(x)$ always increases monotonically from 0 to 1.

The remaining definitions are identical to those with continuous random variables, but with summations rather than integrals:

The **expectation** $E(X) = \langle X \rangle = \mu_X$ of a random variable is the mean vale:

$$\langle X \rangle = \sum_{-\infty}^{\infty} x p(x)$$

In general, the **expectation of any function** of a random variable is:

$$E\big(f(X)\big) = \langle f(X) \rangle = \sum_{-\infty}^{\infty} f(x) p(x)$$

Expectation is a linear operator: $\langle aX_1 + bX_2 \rangle = a\langle X_1 \rangle + b\langle X_2 \rangle$. However, nonlinear combinations of random variables cannot be calculated the same way: $\langle X^2 \rangle = \sum_{-\infty}^{\infty} x^2 p(x) dx \neq \langle X \rangle^2$.

The variability of a random variable is described by the **variance**,

$$Var(X) = \sigma^2 = \langle (X - \mu_X)^2 \rangle = \langle X^2 \rangle - \mu_X^2$$

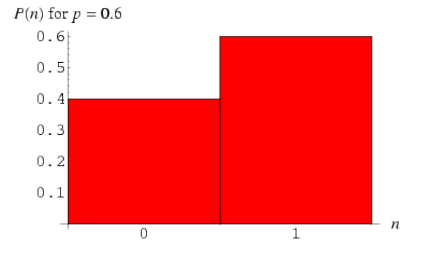
The **standard deviation** is the square root of the variance:

$$\sigma = \sqrt{(\langle X^2 \rangle - \mu_X^2)}$$

There are three common distributions for discrete variables.

The **Bernoulli distribution** takes only the value 0 or 1, with $p(1) = q$, and $p(0) = 1 - q$. This clearly sums to 1. You can use the definition of expectation and variance to show that

$$\mu_X = q, \sigma_X^2 = q(1 - q)$$

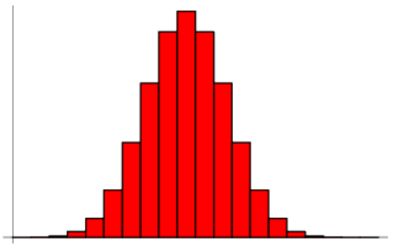


Bernoulli distribution probability density function. Figure from http://mathworld.wolfram.com/

The Bernoulli distribution applies to any questions with a yes or no answer, such as does the dice land on 5, is an ion channel open, does the coin land on heads, is a platelet bound to endothelial cells stationary?

The **Binomial distribution** is the result of N Bernoulli trials. The probability of getting the yes answer to k of the N trials is

$$p(k) = \binom{k}{N} q^k, \qquad where\ \binom{k}{N} = \frac{N!}{k!\,(N-k)!}$$

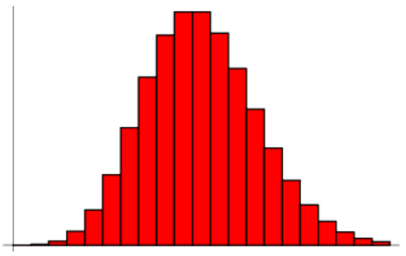


Binomial distribution probability density function. Figure from http://mathworld.wolfram.com/

You can use the definition of expectation and variance to calculate these values:

$$\mu = Nq, \qquad \sigma^2 = Nq(1-q)$$

The **Poisson distribution** is an approximation for the binomial distribution for large N; that is, $\rightarrow \infty$, but $qN \rightarrow \lambda$, because $p \rightarrow 0$. For example, we might ask how many gene deliver vectors bind to a cell. The key to the Poisson distribution is that we assume that reservoir is not depleted significantly, so the events are independent. In contrast, there is a maximum number possible for the binomial distribution.



Poissson distribution probability density function. Figure from http://mathworld.wolfram.com/

$$p(k) = \frac{\lambda^k \exp(-\lambda)}{k!}$$

It can be shown that the mean and variance are both λ:

$$\mu_X = \sigma_x^2 = \lambda$$

Introduction to Stochastic Reaction Methods:

In a reaction model, the dependent random variables are the number or concentration of chemicals or other reactive objects in a compartment. These can be chemicals, macromolecules, virus particles, cells, or even organisms. In stochastic reaction models, the number of objects is sufficiently small that we need to consider the intrinsic variability in the number of reactions that occur, so we need stochastic models.

We cannot use the Wiener process as described in the previous lecture for two reasons. First, we usually need to model the number of objects as integers rather than on a continuum. For example, the single copy of a gene on a bacterial chromosome is either bound to a transcription factor or not at any time. While it may be bound 15% of the time, we cannot have 15% of the one copy bound to transcription factors at one time. Thus, we need to model the objects with discrete random variables. Second, the stochastic changes in different variables are coupled in a way that is determined by the reactions. If a receptor binds to a ligand, we lose one free receptor and one free ligand and make one complex. That is, the number of reactions that occurs is stochastic, but the effect of these reactions on the variables is not. This lecture describes the methods used to model discrete reactions.

Discrete random processes are also called **jump processes** because the number of objects stays the same, then jumps stochastically, rather than steadily changing. When we can assume that all the possible jumps that can occur are all independent of past jumps, and only depend on the current system state, then we refer to this as a **Poisson process**. Stochastic reactions can be modeled as a Poisson process.

In a Poission process, the time to the next jump or event is a continuous random variable with an exponential distribution. If the rate at which an event occurs is a, then the probability that an event will occur at time t is

$$p(t) = a\exp(-at)$$

If instead we ask how many events occur in a given time T, then the answer is a discrete random variable with a Poisson distribution. The expected number of events is $\lambda = aT$, but the probability of k events occurring is

$$p(k) = \frac{\lambda^k \exp(-\lambda)}{k!}$$

These two observations are at the root of the different methods of modeling stochastic reactions.

The **'exact' stochastic algorithm** (often called the **Gillespie Algorithm** because Gillespie's 1977 description is so often cited), jumps forward in time exactly one event at a time by calculating the time to the next event using the exponential distribution. This is called an exact algorithm because the effect of each reaction on the system is determined before another reaction is calculated so there is no chance of making mistakes like deciding two reactions take place when there was only one object left to react. It is best except when it is computationally too expensive.

The **tau-leap algorithm** calculates the *expected* change that would occur in a predetermined time step, then uses the Poisson distribution to calculate the actual number of each reaction that occurs in that time, and implements the effect of those reactions on the chemicals. This allows a jump forward in time, inspiring the name tau-leap. For this algorithm to be sufficiently accurate, the system should not change much during the time step, which is the same requirement as for deterministic simulations. We use the *expected* rather than *actual* change when evaluating whether the predetermined time step meets tolerance, to avoid biasing the simulation, and because systems with few molecules must change by a large fraction when any reaction occurs. The tau-leap is more efficient than the exact for systems with many molecules, since many reactions can occur without changing the system by a large fraction, but is less efficient for systems with few molecules, since even a single reaction changes the system a lot.

The **Chemical Langevin Equation** (CLE algorithm) was developed by Gillespie in 2000 and is similar to the tau-leap algorithm except that it uses the normal distribution instead of a Poisson distribution to determine stochastically the number of events that occur during the time step. For large values of λ, the Poisson distribution approaches the normal distribution with mean λ and variance λ, and the latter provides a more efficient calculation for large λ. The normal distribution is continuous, not stochastic, so must be rounded to maintain discrete values. The CLE on its own has limited utility since it requires existence of a time step where the expected number of all reactions is large without changing the system much.

Formalism for Reaction Based Solving .

All three methods described above need to determine how many of which reactions actually occur in each time interval, and then implement the reactions by changing concentration of substrates. For example, the glucosidase enzyme (E) converts malose (M) to glucose (G) with the following reactions:

$$E + M \rightarrow EM, rate\ k_f$$
$$EM \rightarrow E + M, rate\ k_r$$
$$EM \rightarrow E + G, rate\ k_{cat}$$

For a deterministic simulation, this is converted to the following differential equations, with $E\ + EM = E_{tot}$:

$$\frac{dEM}{dt} = k_f(E_{tot} - EM)M - (k_r + k_{cat})EM$$
$$\frac{dM}{dt} = -k_f(E_{tot} - EM)M + (k_r)EM$$
$$\frac{dG}{dt} = (k_{cat})EM$$

However, for stochastic reaction equations, we will write the system equations in the reaction form instead of the differential equation form. While the three publications describing the algorithms above use different formalisms to represent reactions, we can use the same formalism for all of them, which will make it easier to understand and use them. We will use the formalism described below:

- **R is for Reactions**: we use R for the number of reactions in the system, use the rows of matrices to indicate which reaction, and r as the index into the reaction list.
- **C is for chemicals**: we use C for the number of chemicals in the system, use the columns of matrices to indicate which chemical, and c as the index into the chemical list. Note that the reactants don't need to be chemicals (they might be cells, organisms, molecules, etc, but we call them chemicals for simplicity.
- **S is for Substrates and P is for Products**.: The chemicals that react are called substrates. The results of each reaction are called products. Thus, if an enzyme and its substrate bind to form an enzyme-substrate complex ($E + S \rightarrow ES$), we refer to the enzyme and substrate as substrates and the complex as a product. If the complex dissociates to an enzyme and substrate again ($ES \rightarrow E + S$), we call the complex a substrate and the other two both products. Thus, substrates and products refer to the role in a reaction, not to enzymatic substrates and products. We represent the stoichiometry (number of each type of substrate and product) of a set of reactions with a matrices S and P, in which each row is a reaction and each column indicates the number of each chemical that is a substrate or product of that reaction. (see example below)
- **K is for rate constant**. Each reaction is also characterized as a rate constant. The rate of reaction is the product of the rate constant and the amount of all the reactants. We define the rate constants in a column vector K in which each row indicates the rate of that reaction.
- **The system state is a row vector, in which each column is a different chemical**. We define the initial conditions with a vector, and the trajectory of the simulation will have a column for each time point.

Example: for the set of reactions described above, for this system, there are 4 chemicals and 3 reactions, so we will need a 4X3 matrix for the S and P, and 1X3 (vector) for K, as follows, if we list the chemicals in the order $[EM \quad M \quad G \quad E]$

$$S = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

$$k = \begin{bmatrix} k_f \\ k_r \\ k_{cat} \end{bmatrix}$$

The rate a(r) at which reaction r occurs is then a product of the number of ways we can select the reacting partners, which we refer to as h(r), and the rate constant k(r).
For example, for reaction 1:

$$a(1) = k(1) * h(1) = k_f * M * E$$

More generally, we start with $a(r) = k(r)$, and then check each element of the substrate matrix for that reaction, $S(r, c)$:

- If $S(r, c) = 0$, that is, a chemical c is not a substrate for reaction r, we ignore X(c), the amount of chemical c in the system in its current state.
- If $S(r, c) = 1$, then we multiply $a(r)$ by $X(c)$, so the rate is proportional to the amount of this chemical. We have X(c) ways to select one of chemical c. Note that this equals 0 if there is none of chemical c in the current system state, so the reaction cannot occur.
- If $S(r, c) = 2$, we multiply by $a(r)$ by $\frac{X(c)(X(c)-1)}{2}$. You might have expected $X(c)^2$, but we can't use the same copy of a chemical twice. That is, we take one from the set of X(c), and are left with X(c)-1 for our second selection. We divide by 2 because we double counted since there were two ways to get the same reacting pair.

Note that the stochastic nature of the reactions depends on the actual number of reactions in a compartment, but the rate at which bimolecular reactions occur depends on the concentration of reactants. We thus need to know the volume of our system and modify these rate constants accordingly so the units will now be 1/sec instead of 1/(sec*molar).

Exact Algorithm

Recall that the exact algorithm moves forward one reaction at a time, and then implements this reaction. This can be done in the following steps:

1. Calculate the reaction rates:
   a. Given the system state X, we find the rate of each reaction $a(r)$ as described above.
   b. We add these rates to get the total rate at which the system changes in any way,

$$a_0 = \sum_{r=1}^{R} a(r)$$

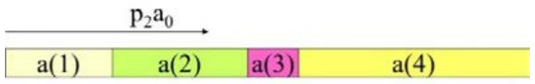
2. Use these rates and random number generators to find out what happens next.

a. Determine the time at which the next event takes place using the exponential distribution $p(\tau) = \exp(-a_0\tau)$, because this is a memoryless Poisson process. In MATLAB, we can implement this as follows: rearrange the probability equation above to get: $\tau = \frac{1}{a_0}\ln\left(\frac{1}{p1}\right)$, where p is a random number uniformly distributed between 0 and 1. (`p1 = rand`).

b. Determine which event it is that occurs. Pick a second random number, p2, also on the uniform distribution from 0 to 1. Then find r such that

$$\sum_{i=1}^{r-1} a(i) < p_2 a_0 < \sum_{i=1}^{r} a(i)$$

This is illustrated below, where p2a0 fell between a(1) and a(1) + a(2), so r = 2.



Note that the probability of reaction r is proportional to a(r)/a0.

3. Update the system state (the number of each chemical) to implement reaction r.
   a. Record the new time: t(end+1) = t(end) + tau;
   b. Update the system state according to the reaction stoichiometry. That is, for each chemical c, $X(c) = X(c) - S(r,c) + P(r,c)$. In matrix format: `X(end+1,:)=X(end,:)-S(r,c)+P(r,c)`

The exact algorithm gets very slow when any one reaction is very fast relative to others, which often happens when there are a lot of some chemicals and not others.

Tau-leap Algorithm

Recall that the tau-leap algorithm jumps forward a set time Δt, and determines how many reactions happen during that time using the Poisson distribution. This can be implemented with the following steps:

1. Find the expected number of each reaction.
   a. Determine the a(r) as in the Gillespie.
   b. Calculate the expected number of each reaction r: $\lambda(r) = a(r)\Delta t$.
   c. If we are dynamically determining the time step, we make sure that the expected change will be just within the relative tolerance allowed. If not, we use a larger or smaller time step and recalculate the expected numbers of reactions. For example, let change = lambda*(-S+P) will be a vector of changes, which we divide by current = X(end,:), which is the vector of the system state. If the maximum of the absolute value of the elements of change/current vector is greater than RelTol, we make the time step smaller.
2. Calculate the actual number of events that occur using the Poisson distribution.
   a. Recall that the Poisson distribution is

$$P(k) = \frac{\lambda(r)^k \exp\left(-\lambda(r)\right)}{k!}$$

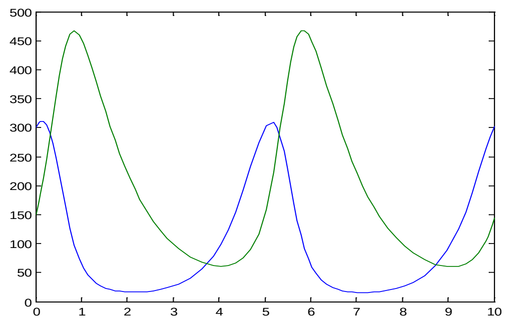
In MATLAB, we can let p(r) be a uniformly distributed random number between 0 and 1, then go in a loop starting at k = 0, calculating P(k) and adding it to the previous sum. When this value is greater than p(r), then the number of reaction r that occur is k. We call this k(r).

   b. Our tau-leap algorithm has the slight modification that we use the normal distribution to determine k(r) if $\lambda(r) > 20$. That is, we let $p(k) = \lambda(r) + \lambda(r) * randn$, where randn is normal distribution with mean and variance 1. Then, we round to nearest integer. Since Poisson approaches normal distribution for large values of λ, this is fairly accurate, and since the loop calculations because expensive for large k's, which occur most of the time for large values of λ, this is more efficient in these cases.

3. Update the system state (the number of each chemical) to implement all the reactions.
   a. Record the new time: t(end+1) = t(end) + tau;
   b. Update the system state according to the reaction stoichiometry. That is, for each reaction r and chemical c, $X(c) = X(c) + k(r) * \left(-S(r,c) + P(r,c)\right)$. In matrix format: `X(end+1,:)=X(end,:) + k*(-S+P)`

Wendy's Stochastic Simulation Software Suite.

- Algorithms are posted on the Thomas lab web site with examples that illustrate the syntax: https://faculty.washington.edu/wendyt/software.html.
- The functions are written like the ODE solvers; two solvers (exact and Tau-leap), that call the equations you write.
- You write the equations using the reaction formalism (S, P, K), instead of the derivative equations as for ODEs.



Example: Lotka-Volterra

Recall the Lotka system:

$$\frac{dH}{dt} = rH - aHP$$

$$\frac{dP}{dt} = bHP - mP$$

Which provides stable oscillations shown here. We can write this as a set of reactions:

$$H \rightarrow 2H, rate\ r$$

$$H + P \rightarrow P + \frac{b}{a} * P, rate\ a\ (requires\ \frac{b}{a}\ to\ be\ an\ integer)$$

$$P \rightarrow \emptyset\ , rate\ m$$

So in our formalism:

$$C = [H \quad P]$$

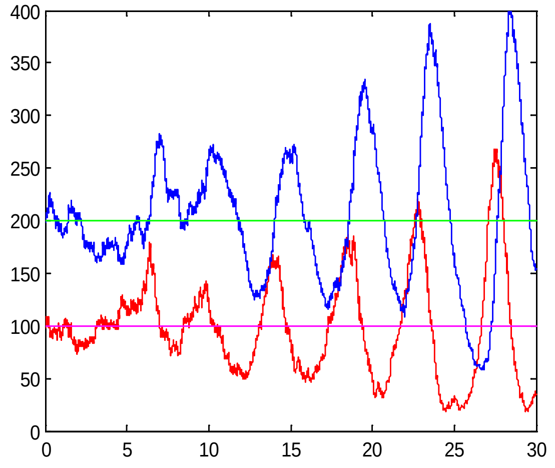
$$S = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{bmatrix}$$

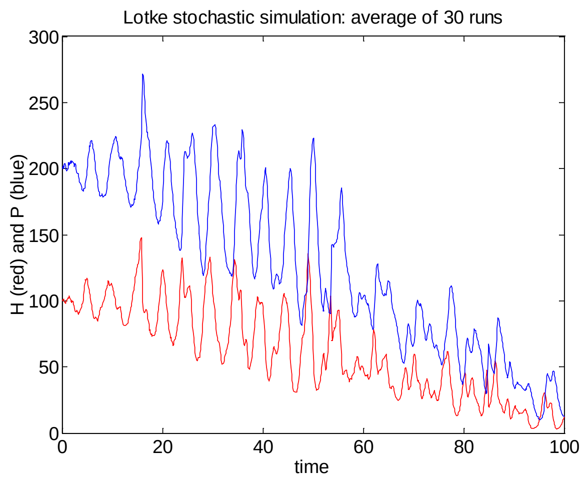
$$P = \begin{bmatrix} 2 & 0 \\ 0 & 1 + b/a \\ 0 & 0 \end{bmatrix}$$

$$K = \begin{bmatrix} r \\ a \\ m \end{bmatrix}$$

Now we simulate the system, starting at equilibrium. Recall from lecture 5 that H = 100, P = 200 was a neutrally stable equilibrium point, so if we started a simulation at this value, it is stationary, and a simulation started at a small perturbation maintained a small oscillation. We expect the stochastic simulation to flucturate, but the actual behavior may surprise you: the oscillations grow over time, as indicated by the single run on the left below, with H is red (stochastic) or pink (deterministic) and P is blue (stochastic) or green (deterministic). When the oscillations get sufficiently big, the Hares may go so low that they go extinct, followed by the Predators. When 30 runs are averaged, as shown in the figure to the right below, the average oscillates, but decreases over time as more and more runs result in extinction. Thus, the average of many simulations looks nothing like the mean. Recall that this is possible for nonlinear systems.

Thus, while (0,0) is a saddle point in the deterministic system, many simulations end at (0,0) in the stochastic system because they jump stochastically onto the (0,P) eigenvector that is the one direction that approaches the saddle point. And, while (100,200) is a neutrally stable equilibrium point, simulations oscillate away from it, because the stochastic 'jumps' more often move the system to a larger rather than a smaller cycle.





Summary of Stochastic Reaction Equations.

1. The **Bernoulli distribution** describes the chance that a single test provides an outcome of yes (or 1), with $p(1) = q, p(0) = 1 - q; \mu = q, \sigma^2 = q(1 - q)$
2. The **binomial distribution** is the sum of N Bernoulli distributions. $p(k) = \binom{k}{N} q^k; \mu = Nq,\ \sigma^2 = Nq(1 - q)$
3. The **Poisson distribution** is a discrete distribution with $p(k) = \frac{\lambda^k \exp(-\lambda)}{k!}; \mu = \sigma^2 = \lambda.$
4. Stochastic chemical reaction equations are necessary when intrinsic noise is significant, or for nonlinear systems, where mean may not approach the deterministic; that is, where stochastic behaves fundamentally different.
5. The **exact' stochastic algorithm** determines the time to the next event using the exponential distribution from the overall rate of change in the system, and determines which event occurs from the uniform distribution and the relative rate of each potential reaction that leads to a system change.
6. The **tau leap algorithm** determines the number of each event that occur in the next time step using the Poission distribution. This algorithm requires that the expected change in the system (the fractional change in each chemical) during that time step is small.
7. The **Chemical Langevan equation (CLE)** is the same as the tau-leap, but uses the normal distribution. This algorithm also requires that the fractional change is small, but in addition requires that the number of each reaction that occurs during the time step is large. The CLE thus requires large numbers of all chemicals.
8. All algorithms calculate the rate at which each reaction occurs, which we called a(r). This is the product of the rate constant and the number of each substrate chemical present in the system, for reactions that only involve one of any chemical. For reactions in which a chemical must react with itself, this is the product of X(X-1)/2, where X is the number of the substrate chemical present in the system.
9. You can't determine whether the time step is too long by testing how much the system changed *after* implementing the stochastic step, since this will bias the system against larger changes. Instead, you need to use the expected change to evaluate the time step.