# Time-Ordered Product Expansions for Computational Stochastic Systems Biology

Eric Mjolsness

April 8, 2013

**Abstract**

The time-ordered product framework of quantum field theory can also be used to understand salient phenomena in stochastic biochemical networks. It is used here to derive Gillespie's Stochastic Simulation Algorithm (SSA) for chemical reaction networks; consequently, the SSA can be interpreted in terms of Feynman diagrams. It is also used here to derive other, more general simulation and parameter-learning algorithms including simulation algorithms for networks of stochastic reaction-like processes operating on parameterized objects, and also hybrid stochastic reaction/differential equation models in which systems of ordinary differential equations evolve the parameters of objects that can also undergo stochastic reactions. Thus, the time-ordered product expansion (TOPE) can be used systematically to derive simulation and parameter-fitting algorithms for stochastic systems.

## 1 Introduction

The master equation for a continuous-time stochastic dynamical system may be expressed as $dp/dt = W \cdot p$ where $W$ is the time-evolution operator, often an infinite-dimensional matrix. Particular choices for $W$ lead to the special case of the "chemical master equation" for stochastic chemical kinetics, often useful in bioligical applications; we will see this and other applications below. The general master equation has the formal solution $p(t) = \exp(tW) \cdot p(0)$. If $W$ can be decomposed as a sum $W_0 + W_1$, then there is a perturbation theory for $\exp(tW)$ in terms of $\exp(tW_0)$ and its perturbations by $W_1$. The time-ordered product expansion (which we refer to by the acronym TOPE) gives a formula for the solution of a master equation [1-3] which can be expressed as follows [4]:

$$\begin{aligned}\exp\left(tW\right)\cdot p_0 &= \exp\left(t\left(W_0+W_1\right)\right)\cdot p_0\\ &= \sum_{k=0}^{\infty}\left[\int_0^t dt_k\int_0^{t_k}dt_{k-1}\cdots\int_0^{t_2}dt_1\exp((t-t_k)\,W_0)\cdot W_1\right.\\ &\qquad\left.\cdot\exp((t_k-t_{k-1})\,W_0)\cdots\quad\cdot W_1\cdot\exp(t_1W_0)\cdot p_0\right] \qquad (1)\end{aligned}$$

This expression can be derived (as in [4]) by expanding in powers of $W_1$, each expanded to all orders in $W_0$, and using the normalization formula for the Dirichlet distribution to subdivide the time interval $[0,t]$) into $k$ subintervals.

Since $W_0$ and $W_1$ do not generally commute, this expression involves alternation from right to left of $W_0$ and $W_1$ related operations. Using the "time-ordered exponential" of operators [5], this result can be compactly reexpressed as:

$$\exp(t\left(W_0+W_1\right)) \quad = \exp(tW_0)\left(\exp(\int_0^t W_1\left(\tau\right)d\tau)\right)_+ \qquad (2)$$

where

$$W_1\left(\tau\right)\equiv\exp(-\tau W_0)W_1\exp(\tau W_0). \qquad (3)$$

Here $(\exp(\int_0^t G(\tau)d\tau))_+$ is obtained term by term from the Taylor series for the operator exponential, by reordering all monomials containing terms evaluated at different times so that they are indexed by ordered sequences of times $(\tau_k,...,\tau_1)$ that increase right to left (details reviewed in Section 3.5.1 below). In field theory it is standard to prefer Equation 2 over Equation 1 for theoretical calculations, but for algorithmic concreteness this paper will favor the more explicit expression Equation 1 where possible. Indeed, each summand of Equation 1 already looks like a Markov chain in which the matrix or operator product operation "$\cdot$". which sum over states, is supplemented by integration over an extra time variable. This observation will be made precise in Section 3. In general there is a risk that the infinite sum over terms could diverge. However, the master equation must conserve total probability and this constrains $W$ to have zero column-sums and also constrains the spectrum of $W$ to have nonpositive real parts. In this setting some decompositions $W=W_0+W_1$ converge well enough, as we will see by example below.

One particular specialization of TOPE lets us derive Gillespie's Stochastic Simulation Algorithm (SSA): take $W_0=-D=$ the diagonal part of $W$, and $W_1=\hat{W}=$ the off-diagonal part of $W$. Then for chemical reaction networks TOPE generates Feynman-like diagrams. An example is illustrated below for the simple reaction network with just two reactions, the forwards and backwards parts of the generic trivalent reaction $A+B\rightleftharpoons C$, to which others can be reduced.

Figure 1. A time history of the reaction $A+B\rightleftharpoons C$. Time flows left to right. Open circles represent reaction events, with probability factor $\times W_1$. In between

Figure 1 goes about here.

reaction events are unimolecular particle propagators $\exp((t_k - t_{k-1})W_0)$, labelled by arrows and particle names (repeated for clarity). This is a non-spatial version of the Lee model in quantum field theory (cf. for example [6]).

The TOPE (Equation 1 or Equation 2) can be applied recursively, since it reduces one operator exponential $\exp(tW)$ to another one $\exp(tW_0)$. This fact will be exploited in Section 3 below. But eventually one must get to an operator exponential that is tractable by other means. One way to do this is to let $W_0 = D =$ the diagonal part of $W$, as in the SSA algorithm derivation below.

## 2 Methods

### 2.1 Creation/annihilation operator notation

We will use operator notation for molecule (or other reactant) creation and annihilation state changes [1-4]. Here we just review the notation as used in [4]. The elementary operators $a$ and $\hat{a}$ act (respectively) to destroy and create identical particles of a given type. In the particle-number basis their elements have the entirely off-diagonal expressions

$$a_{ij} = j\delta_{ij-1} \quad \text{and} \quad \hat{a}_{ij} = \delta_{ij+1} \quad \text{forall} \quad i,j \in \quad \{0,1,2,...\}\,. \tag{4}$$

Here $\delta_{ij}$ is the Kronecker delta function. The creation and annihilation operators satisfy the Heisenberg algebra $[a,\hat{a}] = I$ but are different from those of quantum mechanics because they are not conjugates or transposes of one another. (This is the reason we do not denote the creation operator$a^\dagger$, as it is in quantum mechanics, or $a^*$.) Instead of being conjugate to $\hat{a}$, the annihilator $a$ encodes the chemical law of mass action since its nonzero entries are equal to the number of particles available to react or decay. The diagonal "number operator" is $N \equiv \hat{a}a$.

The creation and annihilation operators may be represented in terms of their action on probability generating functions $g(z) = \sum_{n=0}^{\infty} p_n z^n$, where $p_n$ is the probability that there exist $n$ particles of a given type. In this case:

$$a = \partial_z \cdots \qquad \text{and} \qquad \hat{a} = z \times \cdots \tag{5}$$

In the presence of different types of particles (eg. molecules or other objects) the creation/annihilation operator notation is generalized, eg to $a_\alpha$ and $\hat{a}_\beta$ for molecule types $A_\alpha$, in which all operators for unequal types commute:

$$[a_\alpha, \hat{a}_\beta] \equiv a_\alpha \hat{a}_\beta - \hat{a}_\beta a_\alpha = \delta_{\alpha\beta}$$

Operating on an empty "vacuum" state $|0\rangle$ with no objects, the monomials in the creation operators $\hat{a}_\beta$ span a Fock space. Molecule or object types indexed by $\alpha$ may even be taken to include arbitrary discrete-valued molecular attributes (or attributes of other objects) such as phosphorylation state or integer-valued parameters. Continuous-valued parameters such as position (in quantum field theory it would more naturally be the conserved momentum, unlike the typical viscous-medium dynamics in biology) may be encoded into a real-valued vector argument $x$which requires a Dirac delta function instead of a Kronecker delta function, so for example:

$$[a_\alpha(x), \hat{a}_\beta(y)] = \delta_{\alpha\beta}\delta(x-y) \tag{6}$$

A non-molecular example of such parameterized objects would be: cells of a given real-valued volume and/or lengthscale as in Section 3.5.5 below.

However for some attributes such as real-valued object positions one may wish to limit the state space to between zero or $n_{\max,\alpha}$ molecules (or other objects) at each unique real value. The resulting commutator is still diagonal as described in [4]. The particular case $n_{\max,\alpha} = 1$ is *not* a stochastic version of fermions because particles with different types or values of the attributes still commute rather than anticommuting.

The basic rule for translating chemical reactions into creation/annihilation operator notiation is: first, annihilate all objects on the incoming or left hand side of a reaction; then create all the objects on the outgoing or right hand side of a reaction. Thus, the off-diagonal part of the operator for a reaction

$$\begin{aligned}\left\{A_{\alpha(p)}(x_p)|1 \leqslant p \leqslant p_{\max}\right\}_* \longrightarrow \left\{A_{\beta(q)}(y_q)|1 \leqslant q \leqslant q_{\max}\right\}_* \\ \textbf{with} \quad \text{reaction rate} \quad \rho_r([x_p]_1^{p_{\max}}, [y_q]_1^{q_{\max}})\end{aligned}$$

that converts an incoming multiset $\{\cdots\}_*$ of numerically parameterized reactants $\left\{A_{\alpha(p)}(x_p)|p \in \text{lhs}(r)\right\}_*$ each with parameter vector $x_p$ (reactants can appear multiple times in a multiset) into an outgoing multiset $\left\{A_{\beta(q)}(y_q)|q \in \text{rhs}(r)\right\}_*$ each with parameter vector $y_q$, is:

$$\hat{O}_r = \rho_r([x_p],[y_q]) \left[\prod_{q\in\text{rhs}(r)} \hat{a}_{\beta(q)}(y_q)\right] \left[\prod_{p\in\text{lhs}(r)} a_{\alpha(p)}(x_q)\right] \;. \tag{7}$$

There is one such operator for every possible set of values for the numerical parameters. Since time-evolution operators for different processes just add, a generic operator for all parameter values must sum and/or integrate the operator of Equation 18 over all the parameters, in the Cartesian product of measure spaces in which they take values:

$$\hat{O}_r = \int\int d\{x\}\, d\{y\}\ \rho_r([x_p],[y_q]) \left[\prod_{q\in\text{rhs}(r)} \hat{a}_{\beta(q)}(y_q)\right] \left[\prod_{p\in\text{lhs}(r)} a_{\alpha(p)}(x_q)\right] \tag{8}$$

The generalization is conceptually straightforward because we have simply used a function $\rho_r([x_a],[y_b])$ to express the possibly infinite number of different reaction rates that pertain to objects that differ only in their attributes. Because of the algebra of noncommuting basic creation and annihilation operators, reaction operators $\hat{O}_r$ and $\hat{O}_{r'}$ for reactions $r \neq r'$ that produce and consume a shared reactant $A_\alpha(x)$ (or $A_\alpha$ for reactants with type $\alpha$ but no other parameters) generally also have nonzero commutators.

Equation 18 or Equation 8 add probability to the new state of the system, but do not take it away from the old state of the system before a reaction. That job requires a negative diagonal matrix as shown in Equation 10 below. In the case of Equation 18, the corresponding diagonal operator is $D_r = \rho_r \ [\prod_{a \in \text{lhs}(r)} N_{\alpha(p)}(x_p)]$. Examples are provided in [4] and below.

## 2.2 Solvable example: An exact solution for SSA behavior

For a few very simple examples, we can not only solve analytically for the behavior of the biochemical system, but we can even add in the behavior of the SSA simulation algorithm and solve for that exactly as well. For example consider the minimal bidirectional reaction $A \longleftrightarrow \varnothing$. This case is analytically solvable, including the complete statistics of its SSA algorithm simulation. It has forward synthesis and backwards decay reactions. The operator expression is therefore:

$$W = k_s(\alpha\hat{a} - I) + k_d(\alpha a - N) \tag{9}$$

Here $\alpha = 1$ is the generating function variable for the total number of reactions, corresponding to off-diagonal matrix elements of $W$. Power series in $\alpha$ will decompose total probability according to this number.

Translating the master equation for Equation 9 into a PDE in the two variables $t$ and $z$ using representation Equation 5, and solving analytically, this model has the exact solution

$$\begin{aligned} g_m(z,t|\alpha) &= \left(\alpha + (z-\alpha)\,e^{-k_d t}\right)^m \exp[-\frac{k_s}{k_d}\left((1-\alpha)\,k_d t + \left(z\alpha - \alpha^2\right)\left(e^{-k_d t} - 1\right)\right)] \\ &= \left(\alpha + (z-\alpha)\,e^{-k_d t}\right)^m \exp[\frac{k_s}{k_d}\left(z\alpha - 1\right)\left(1 - e^{-k_d t}\right)\Big)\Big] \exp[\frac{k_s}{k_d}\left(\alpha^2 - 1\right)\left(k_d t + e^{-k_d t} - 1\right)\Big)\Big] \\ &= \text{Binomialinitialconditionwithdecay} * \text{Poissononforwardreactions} \\ &\qquad * \text{Poissononforward/backwardreactionpairs.} \end{aligned}$$

As usual $z$ is the generating function variable whose exponent is the total number $n_A$ of $A$ molecules or particles, $m$ is the initial number of molecules, and $t$ is continuous time. The $*$ operation is a convolution of probability distributions. A product of generating functions with the same variable is a convolution of distributions [7]. Note the interpretation in terms of Binomials and Poissons with time-varying parameters. The third factor represents a linearly increasing number of canceling forward/backward reaction pairs as a function of time - a kind of random walk.

The full derivation below will generalize this solvable example, again separating the diagonal from the off-diagonal terms in $W$.

## 2.3 Notation for SSA rederivation from TOPE

One specialization of TOPE lets us derive SSA for biochemical reaction networks, as follows. First decompose $W$ into nonegative off-diagonal and nonpositive diagonal parts, as must be possible by the conservation and nonnegativity of probability. For example conservation of probability implies $\forall p \;\; 0 = d(\mathbf{1} \cdot p)/dt = (\mathbf{1} \cdot W) \cdot p \Rightarrow \mathbf{1} \cdot W = 0$. Then

$$\begin{aligned} W = \hat{W} - D, \qquad \text{where} \qquad D \triangleq \operatorname{diag}(\mathbf{1} \cdot \hat{W}), \quad \text{i.e.} \\ \hat{W}_{IJ} \triangleq (1 - \delta_{IJ})\, W_{IJ} \quad \text{and} \quad D_{IJ} \triangleq \delta_{IJ} \sum_K \hat{W}_{KJ} \triangleq \delta_{IJ} D_I \end{aligned} \tag{10}$$

where $I$ and $J$ index the possible states of the system. To prevent negative probabilities from evolving under the master equation, all entries of $\hat{W}$ and therefore $D$ must be nonnegative. In this circumstance the TOPE becomes:

$$\begin{aligned} \exp\left(t\left(\hat{W} - D\right)\right) = \sum_{k=0}^{\infty} & \left[ \int_0^t \cdots \int_0^t \left(\Pi_{q=0}^k d\tau_q\right) \delta\left(t - \sum_{q=0}^k \tau_q\right) \right. \\ & \left. \times \exp(-\tau_k D) \hat{W} \;\; \cdots \exp(-\tau_1 D) \hat{W} \exp(-\tau_0 D) \right] \\ = \sum_{k=0}^{\infty} \int_0^t \cdots \int_0^t & \left(\Pi_{q=0}^k d\tau_q\right) \delta\left(t - \sum_{q=0}^k \tau_q\right) \exp(-\tau_k D) \left[ \prod_{q=k-1\downarrow}^{0} \hat{W} \quad \exp(-\tau_q D) \right] \end{aligned}$$

Since the summands over $k$ and integrands over $[\tau_q]_0^k$are mutually exclusive, exhaustive and nonnegative, we define the conditional probability distribution on $k$ and $[\tau_q]_0^k$ by these summand/integrands (where $[\tau_q]_0^k \triangleq [\tau_0, ...\tau_k]$ denotes an ordered contiguous sequence of time intervals):

$$\Pr(I, [\tau_q]_0^k, k | J, t) \triangleq \left\{ \exp(-\tau_k D) \left[ \prod_{q=k-1\downarrow}^{0} \hat{W} \quad \exp(-\tau_q D) \right] \delta\left(t - \sum_{q=0}^k \tau_q\right) \right\}_{I,J} \tag{11}$$

(where a product over zero terms such as $\prod_{q=-1\downarrow}^{0}$ is interpreted as the identity matrix, and products over negative numbers of terms such as $\prod_{q=-2\downarrow}^{0}$ should not occur). For $D_{II} \neq 0$,

$$\Pr(I, [\tau_q]_0^k, k | J, t) = \left\{ \exp(-\tau_k D) \left[ \prod_{q=k-1\downarrow}^{0} \left(\hat{W} \quad D^{-1}\right) (D \exp(-\tau_q D)) \right] \delta\left(t - \sum_{q=0}^k \tau_q\right) \right\}_{I,J}$$

Either way,

$$\Pr(I|J,t) = \sum_{k=0}^{\infty} \int_0^t \cdots \int_0^t \left(\Pi_{q=0}^{k} d\tau_q\right) \Pr(I, [\tau_q]_0^k, k|J,t) = \left[\exp(t\left(\hat{W} - D\right))\right]_{I,J}.$$

The bracket notation $[X_q]_{\min}^{\max} \overset{\triangle}{=} [\tau_{\min}, ...\tau_{\max}]$ for an ordered set of components indexed by $q$ will also be used for state variables $[I_q]_{\min}^{\max}$. The notation "$\overset{\triangle}{=}$" means "equal by definition" . In what follows, the notation "$\Theta$(Pred)" where Pred is a predicate is the Heaviside step function or indicator function taking the value 1 if the predicate is true and 0 if it is false.

### 2.4 Semigroup property

Suppose $t = t_1 + t_2$, all nonnegative. Then for any time-evolution equation we must have the semigroup property:

$$\Pr(I|J,t) = \sum_K \Pr(I|K,t_2) \Pr(K|J,t_1).$$

Is there a $k$-event version of this rule, for $k = k_1 + k_2$? In other words, can we add (nonnegative) numbers of reaction events rather than time intervals ? We observe (where again $[\tau_q]_0^k \equiv [\tau_0, ...\tau_k]$ )

$$\Pr(I,k|J,t) = \int_0^t \cdots \int_0^t \left(\Pi_{q=0}^{k} d\tau_q\right) \Pr(I, [\tau_q]_0^k, k|J,t).$$

Then, according to a derivation given in Appendix I, if $k = k_1 + k_2$ and for any $\tau'_{k_1} \in [0, \tau_{k_1}]$, there is a semigroup law:

$$\begin{aligned} \Pr(I, [\tau_q]_0^k, k|J,t) = \sum_K \int_0^t d\tau \Pr(I, \left[\tau'_{k_1}, \tau_{k_1+1}, ...\tau_k\right], k_2|K,\tau) \\ \times \Pr(K, \left[\tau_0, ...\tau_{k_1-1}, \tau_{k_1} - \tau'_{k_1}\right], k_1|J, t-\tau). \quad (12) \end{aligned}$$

In this result there is an arbitrary choice of $\tau'_{k_1}$from the interval $[0, \tau_{k_1}]$. However this form does not yet pertain to conditional probabilities of the form $\Pr(I,t|k,J)$, as needed to obtain a computable Markov process algorithm.

## 3 Results and discussion

Given the foregoing notation, we undertake the derivation of a Markov chain representing the SSA algorithm. We then consider extensions of this result, including parameterized reactants, but focussing mainly on hybrid stochastic event/ordinary differential equation dynamical systems.

## 3.1 Derivation of a Markov chain

### 3.1.1 Bayesian recurrence

In Appendix I we argue that the correct Bayesian strategy for moving from $\Pr(I,k|t,J)$ to $\Pr(I,t|k,J)$, as needed to obtain a simulatable Markov chain, is to consider large stopping times $T \gg t$ which are overwhelmingly likely to have large reaction numbers $n \gg k$; then to marginalize the probability distribution $\Pr([I],[\tau],n|J,T)$ over all event numbers $n > k$ and to conditionalize it over all event numbers $q < k$. By that means in Appendix I we derive the recurrence relation

$$\tilde{\Pr}([I_q]_1^k,[\tau_q]_0^{k-1}|k,J) = \tilde{\Pr}(I_k,\tau_{k-1}|1,I_{k-1})\tilde{\Pr}([I_q]_1^{k-1},[\tau_q]_0^{k-2}|k-1,J) \quad (13)$$

where

$$\tilde{\Pr}([I_q]_1^k,[\tau_q]_0^{k-1}|k,J) \triangleq \lim_{T\to\infty}\sum_{n=k+1}^{\infty}\sum_{\{[I_q]_{k+1}^n\}}\int_0^T\cdots\int_0^T [d\tau_q]_k^n \Pr([I_q]_1^n,[\tau_q]_0^n,n|J,T) \quad (14)$$

and in particular

$$\begin{aligned}\tilde{\Pr}(I_k,\tau_{k-1}|1,I_{k-1}) &= \frac{\tilde{\Pr}([I_q]_1^k,[\tau_q]_0^{k-1}|k,J)}{\tilde{\Pr}([I_q]_1^{k-1},[\tau_q]_0^{k-2}|k-1,J)}\\ &= \hat{W}_{I_k I_{k-1}} \quad \exp(-\tau_{k-1}D_{I_{k-1}})\Theta(\tau_{k-1}\geqslant 0). \quad (15)\end{aligned}$$

Note that $\tilde{\Pr}([I_q]_1^k,[\tau_q]_0^{k-1}|k.J)$ marginalizes over $\tau_k$, the time elapsed since the last event $k$, as well as all later times and events. It is a distribution on histories up to and including the "just-fired" $k$'th reaction event, within a much longer history.

### 3.1.2 Markov chain - Summary

From the forgoing Bayesian recurrence equation, and the definition

$$\tilde{\Pr}(I,t_k|k,J) \triangleq \sum_{\{[I_q]_1^{k-1}\}}\int_0^\infty\cdots\int_0^\infty [d\tau_q]_0^{k-1}\tilde{\Pr}([I_q]_1^k,[\tau_q]_0^{k-1}|k,J)\delta\left(t_k-\sum_{q=0}^{k-1}\tau_q\right),$$

Appendix I shows $\tilde{\Pr}(I,t_k|J,k)$ is a probability density function and proves the following Markov-like property:

$$\tilde{\Pr}(I,t|k,J) = \sum_K\int_0^t d\tau\tilde{\Pr}(I,\tau|1,K)\tilde{\Pr}(K,t-\tau|k-1,J).$$

We may reexpress this result as

$$\tilde{\Pr}(I,t|k,J) \approx \sum_K\int_0^t d\tau\mathcal{W}(I,t|K,t-\tau)\tilde{\Pr}(K,\tau|k-1,J)$$

where we define the Markov chain kernal

$$\mathcal{W}(I,t|K,t-\tau) \overset{\triangle}{=} \tilde{\Pr}(I,\tau|1,K) = \hat{W}_{IK} \quad \exp(-\tau D_K)\Theta(\tau \geqslant 0) \tag{16}$$

In vector/operator notation, for $k \geqslant 2$,

$$\tilde{\Pr}(.,.|k,J) \equiv \mathcal{W} \circ \tilde{\Pr}(.,.|k-1,J)$$

and finally the algorithmic Markov chain expression for SSA including now an initial distribution $\tilde{\Pr}(J|k=0)$ over $J$ at time $t=0$, we have for all $k \geqslant 0$:

$$\tilde{\Pr}(.|k) = \mathcal{W}^k \circ \tilde{\Pr}(.|k=0) \tag{17}$$

which expresses the iteration of a Markov chain of the SSA algorithm. Of course the factor $[D_K \exp(-\tau D_K)\Theta(\tau \geqslant 0)]$ in Equation 16 is just the SSA exponential distribution of non-negative waiting times $\tau$ between reaction events, and $\hat{W}_{IK}/D_K$ is just the branching probability for immediately thereafter chosing a reaction that leads to state $I$.

The foregoing derivation was outlined in far less detail in [8]. A similar equation for SSA was reached by very different methods in [9], Theorem 10.1. To our knowledge this is the first complete derivation of SSA from field theory methods such as TOPE.

This Markov chain expression has also been used as the starting point for the derivation of *exact* accelerated stochastic simulation algorithms [10,11] that execute many reactions per step (i.e. they "leap" forward) and thus go much faster than SSA, while also sampling from the exact probability distribution given by the just-fired probabilities above. These derivations proceed by algebraic rearrangement of terms to express computationally efficient versions of rejection sampling. The algorithm of [11] has been parallelized, which is often difficult for discrete-event simulations.

## 3.2 Algorithm: SSA

The SSA algorithm represented by the Markov chain in Equation 16 and Equation 17 above may be written out in pseudocode as follows:

```
repeat {
    compute propensities  $k^{(r)}$
    compute $k^{(\mathrm{total})} = \sum_r k^{(r)}$
    draw waiting time $\Delta t$ from  $k^{(\mathrm{total})} \exp(-\Delta t k^{(\mathrm{total})})$
        $t := t + \Delta t$;    // advance the clock by $\Delta t$
    draw reaction $r$ from distribution $k^{(r)}/k^{(\mathrm{total})}$ and execute reaction $r$
} until $t \geqslant t_{\max}$
```

## 3.3 Extension: Parameterized rule and graph grammar SSA-like algorithm

For biological modeling, including spatial and mechanical modeling of biological systems, it is important to generalize from pure particles to particles with both discrete and continuous attributes. The complication is that reaction or process rates can then depend on the attributes both of the incoming and outgoing objects. A non-molecular example of such parameterized objects would be cells of a given size, whose propensity to divide may actually depend on their real-valued size parameter (as in Section 3.5.5). More generally, this capability enables agent-based modeling and simulation since it allows interacting objects to have dynamic internal state and even (as explained in Section 3.3.2 below) dynamic relationships.

The time-evolution operator of Equation 18 requires that each attribute or parameter vector consist of constants or variables, each variable appearing just once, and any relationships between variables (such as $x_{2,1} = y_{1,2}$, $x_{2,1} = x_{1,2}$, and/or $y_{2,1} = y_{1,2}$) enforced by the reaction rate $\rho_r([x_p],[y_q])$. Alternatively we can allow repeated appearances of symbolic variables $X_c$ upon which the attributes may depend, through the identity function or otherwise. This is a useful improvement in reaction notation which however may require special-purpose symbolic variable-binding algorithms to support efficiently. Generalizing from Equation 18 and Equation 8, as in [4], we include all possible instantiations of parameters $x_p[X]$ and $y_q[X]$, allowing for repeated occurences of some or all of the variables $X_c$ in $[X]$, with the integrated off-diagonal process representation operator

$$\hat{O}_r = \int \cdots \int d\mu_c(X_c)\ \rho_r([x_p[X]]\,,[y_q[X]]) \\ \times \left[ \prod_{q \in \mathrm{rhs}(r)} \hat{a}_{\beta(q)}(y_q[X]) \right] \left[ \prod_{p \in \mathrm{lhs}(r)} a_{\alpha(p)}(x_p[X]) \right] . \quad (18)$$

As before, $\alpha(p)$ and $\beta(q)$ represent the type of a parameterized object i.e. an object with attributes. Now the symbolic variables $X_c$ each have a type $c$ which has an integration measure $\mu_c$. Again (as in Equation 18 or 8), summation over all discrete-valued parameters and integration over all continuous-valued parameters generalizes the operator to handle all possible sets of parameter values.

### 3.3.1 Algorithm: SSA with parametrized reactant objects

The resulting variant of the SSA algorithm for parameterized reactions can be expressed in pseudocode as follows (outlined briefly in [8]):

```
forall reactions $r$ factor $\rho^{(r)}(x_{\mathrm{in}}, x_{\mathrm{out}}) = k^{(r)}(x_{\mathrm{in}}) p^{(r)}(x_{\mathrm{out}}|x_{\mathrm{in}})$;
repeat {
    compute SSA propensities as $k^{(r)}(x_{\mathrm{in}})$;
    compute $k^{(\mathrm{total})} = \sum_r k^{(r)}(x_{\mathrm{in}})$;
    draw waiting time $\Delta t$ from $k^{(\mathrm{total})} \exp(-\Delta t k^{(\mathrm{total})})$ ;
        $t := t + \Delta t$;    // advance the clock by $\Delta t$
    draw reaction $r$ from distribution $k^{(r)}(x_{\mathrm{in}})/k^{(\mathrm{total})}$;
    draw $x_{\text{out}}$ from $p^{(r)}(x_{\text{out}}|x_{\text{in}})$ and execute reaction $r$;
} until $t \geqslant t_{\max}$
```

### 3.3.2 Structural matching

The functions $\rho(x, y)$ appearing in Equation 18 may impose constraints including equality of variables; equivalently we may allow some variables to appear multiple times in object parameter lists. Either way there follows a mechanism to encode structural relations - graphs and labelled graphs - in the input and output variable lists. Object attributes can include Object ID codes which other objects can also include in their parameter lists. (Of course, the numeric values of Object IDs can be *globally* permuted without changing the structural relationships among extant objects.) In this way, the integrated version of the parameterized reaction operator above encodes structural pattern matching, including variable-binding in logical formulae, among the preconditions that can be enforced before such a generalized reaction or "rule" can fire.

From this point of view, syntactic *variable-binding has the semantics of multiple integration* [4]. In this way we can entrain pattern-matching systems such as the computer algebra system *Mathematica*, or logic-based programming languages, to the job of simulating complex process rules. As in rule-based expert systems, when multiple rules might fire the Rete algorithm [12] can be used to speed up the computations required to maintain knowledge of their relative rates.

The resulting systems have the power to model and simulate dynamic labelled graphs including growing multicellular tissues with dynamical cell-neighbor relationships [4] and molecular complexes with dynamical binding structure [13-15]. Thus, the TOPE operator algebra approach also explains why and how structural (graph-) matching computations arise naturally in biochemical and multicellular biological simulation.

## 3.4 Hybrid SSA/ODE setup

As will be shown in Section 3.4.1 below, the operator formulation for a system of ordinary differential equations is [4]:

$$\hat{O}_{\text{drift}} = -\int\int d\{x\}\, d\{y\}\, \hat{a}(\{y\}) a(\{x\}) \left[ \sum_i \nabla_{y_i} \left( v_i(\{y\}) \prod_k \delta(y_k - x_k) \right) \right] \tag{19}$$

Here and in the calculations that follow, the Dirac delta function can be considered as a Gaussian with very small variance, which participates in a limiting process by which, at the end of each calculation, the limit of zero variance is taken.

In [4] this operator expression is generalized from ordinary differential equations to stochastic differential equations, for example those pertaining to the diffusion of particles, as equivalently represented by the Fokker-Planck equation.

### 3.4.1 Computation of matrix elements

From the commutator

$$[a(y), \hat{a}(x)] = \delta(y-x)\left(I + Q(N(x)|n_{\max})N(x)\right),$$

we may calculate matrix elements of $\hat{O}_{\text{drift}}$ in Equation 18 such as:

$$\begin{aligned}
\left\langle w \middle| \hat{O}_{\text{drift}} \middle| z \right\rangle &= -\left\langle \{w\} \middle| \int d\{x\} \int d\{y\} \left( \sum_i \nabla_{y_i} v_i(\{y\}) \delta(\{y\} - \{x\}) \right) \right. \\
&\qquad \left. \times\, \hat{a}(\{y\}) a(\{x\}) \hat{a}(\{z\}) \middle| 0 \right\rangle \\
&= -\left\langle \{w\} \middle| \int d\{x\} \int d\{y\} \left( \sum_i \nabla_{y_i} v_i(\{y\}) \delta(\{y\} - \{x\}) \right) \right. \\
&\qquad \left. \times\, \hat{a}(\{y\}) \delta(\{x\} - \{z\}) [I + Q(N(x)) N(\{x\})] \middle| 0 \right\rangle \\
&= -\int d\{x\} \int d\{y\} \left( \sum_i \nabla_{y_i} v_i(\{y\}) \delta(\{y\} - \{x\}) \right) \delta(\{x\} - \{z\}) \left\langle \{w\} \,|\, \{y\} \right\rangle \\
&= -\int d\{y\} \left( \sum_i \nabla_{y_i} v_i(\{y\}) \delta(\{y\} - \{z\}) \right) \delta(\{w\} - \{y\}) \\
&= +\int d\{y\}\, \delta(\{y\} - \{z\}) \left( \sum_i v_i(\{y\}) \nabla_{y_i} \delta(\{w\} - \{y\}) \right) \\
&\quad - \int_\partial d\{y\} \left( \sum_i v_i(\{y\}) \delta(\{y\} - \{z\}) \delta(\{y\} - \{w\}) \right) \\
&= \sum_i v_i(\{z\}) \nabla_{z_i} \delta(\{w\} - \{z\}) + \text{boundaryterm} \quad (\to 0 \quad \text{here})
\end{aligned}$$

The easiest treatment for the boundary terms is to add the assumptions that boundaries are at infinity in the space of parameters $x, y \text{and} z$, and that initial conditions place zero probability there, and that finite velocities $v(x)$ ensure the probability remains zero at infinity at finite times. In that case boundary terms can be neglected. Alternatively, we can define $O_{\text{drift}} = \hat{O}_{\text{drift}} - \text{diag}(1 \cdot \hat{O}_{\text{drift}})$ which in this case subtracts off the boundary term. Then

$$\begin{aligned}
O_{\text{drift}} = -\int\int d\{x\}\, d\{y\} \left( \hat{a}(\{y\}) a(\{x\}) - \hat{a}(\{x\}) a(\{x\}) \right) \\
\times \left[ \sum_i \nabla_{y_i} \left( v_i(\{y\}) \prod_k \delta(y_k - x_k) \right) \right] \quad (20)
\end{aligned}$$

If we define $x(t)$ as a time-varying version of $z$, satisfying

$$\frac{\partial x_i}{\partial t} = v_i(\{x_k\}),$$

then

$$\sum_i v_i(\{z\})\nabla_{x_i}\delta(\{w\}-\{x(t)\}) = \sum_i \left(\frac{\partial x_i}{\partial t}\right)\left(\frac{\partial}{\partial x_i}\right)\delta(\{w\}-\{x(t)\}) = \left(\frac{d}{dt}\right)\delta(\{w\}-\{x(t)\})$$

Next we calculate $\langle w|\exp(\tau O_{\rm drift})|z\rangle$. To this end, Taylor's theorem may be written

$$\text{Shift}_\tau \circ f(t) = f(t+\tau) \simeq \sum_{n=0}^{\infty}\frac{\tau^n}{n!}\left(\frac{d}{dt}\right)^n f(t) \simeq e^{(\tau D_t)}f(t)$$

if $\tau$ is a constant. For small $\tau$ we have

$$\begin{aligned}\langle w|\exp(\tau O_{\rm drift})|x\rangle &= \langle w|x\rangle + \tau\,\langle w|O_{\rm drift}|x\rangle + O(\tau^2)\\ &= \left(1+\tau(\frac{d}{dt})\right)\delta(\{w\}-\{x(t)\}) + O(\tau^2)\\ &= \text{Shift}_\tau\delta(\{w\}-\{x(t)\}) \equiv \delta(\{w\}-\{x(t+\tau)\}) + O(\tau^2)\end{aligned}$$

For larger $\tau$ we have

$$\langle w|\exp(\tau O_{\rm drift})|x\rangle = \lim_{n\to\infty}\left\langle w\left|\prod_{i=1}^{n}\exp(\frac{\tau}{n}O_{\rm drift})\right|x\right\rangle$$

$$= \int\cdots\int dx_{n-1}\cdots dx_1[\text{Shift}_{\tau/n}\delta(\{w\}-\{x_{n-1}\})]\cdots[\text{Shift}_{\tau/n}\delta(\{x_1\}-\{x\})]$$

$$\begin{aligned}&= \lim_{n\to\infty}\left(\prod_{i=1}^{n}\text{Shift}_{\tau/n}\right)\delta(\{w\}-\{x(t)\})\\ &= \text{Shift}_\tau\delta(\{w\}-\{x(t)\}) \equiv \delta(\{w\}-\{x(t+\tau)\})\\ &= \delta(\{w\}-\left(z(t=0)+\int_0^t v_i(z(t))dt\right))\end{aligned}$$

Thus (where "IC" means initial condition)

$$\begin{aligned}\langle w|\exp(tO_{\{\rm DE\}})|z\rangle &= \exp(t\sum_i v_i(\{z\})\nabla_{z_i})\delta(\{w\}-\{z\})\\ &= \delta(\{w\}-\left(\{z(0)=z\}+\int_0^t v_i(z(t'))dt\right))\\ &= \delta(\{w\}-\left(\text{Solutionof}\frac{\partial x_i}{\partial t}=v_i(\{x_k\})\text{withIC}\ \ z(0)=z\right))\end{aligned} \tag{21}$$

QED.

As far as we know this detailed derivation has not appeared previously, though our previous work [4] outlined a simplified version. As a corrollary, using Equation 21 we may multiply by $f(w)$ and integrate over $w$ to calculate

$$\begin{aligned}\exp\left(tv\left(\{z\}\right)\cdot\nabla_z\right)\delta\left(w-z\right) &= \delta(w-\left(z(0)+\int_0^t v(z(t'))dt'\right))\\ \Rightarrow\quad \exp\left(tv\left(\{z\}\right)\cdot\nabla_z\right)f\left(z\right) &= f(z(0)+\int_0^t v(z(t'))dt'). \quad (22)\end{aligned}$$

## 3.5 Hybrid SSA/ODE: Operator algebra derivation

We now derive a new SSA-like simulation algorithm for hybrid combinations of discrete events and ODE dynamics, using operator algebra. The main idea is to replace the exponential distribution factor $\exp(-tD)$with a time integral [15]:

$$\exp(-tD)\longrightarrow\exp(-\int_0^t D(t')dt'), \quad (23)$$

and to add an extra ODE to the system of ODEs in order to keep track of the integral. We will now use the more compact formulation of TOPE in Equation 2 to derive this method.

### 3.5.1 Heisenberg picture

Let the operators, rather than the states, evolve in time according to $W_0$ according to Equation 3. This is traditionally called the "Heisenberg picture" in distinction to the "Schroedinger picture" in quantum mechanics. Recall Equation 2 and Equation 3, where $(\cdots)_+$ is the time-ordering super-operator :

$$\left(O(t_i)O\left(t_j\right)\right)_+=\begin{cases}O(t_i)O\left(t_j\right) & \text{if}\quad t_i\geqslant t_j\\ O(t_j)O\left(t_i\right) & \text{if}\quad t_i\leqslant t_j\end{cases}$$

(and likewise for higher order products). Note that if $O(t_i)$ and $O(t_j)$ commute for all pairs of times $t_i$ and $t_j$, then $\left(O(t_i)O(t_j)\right)_+=O(t_i)O(t_j)$, and the time-ordering operator $(\cdots)_+$ can be dropped. Often the notation $\mathcal{T}(O(t_i)O(t_j))$ is used in place of $\left(O(t_i)O(t_j)\right)_+$ to denote the super-operator that time-orders operator products.

### 3.5.2 Application to ODE + decay clock

The hybrid system consisting of chemical reactions (possibly parameterized) together with ordinary differential equations has the combined operator $W=(\hat{O}_{\text{react}}-D_{\text{react}})+O_{\text{DE}}$, which we can regroup as

$$W=(O_{\text{DE}}-D_{\text{react}})+\hat{O}_{\text{react}}$$

and then apply TOPE to $O_{\text{DE}}-D_{\text{react}}$ first with $W_{00}=O_{\text{DE}}$ and $W_{01}(t_k)=-D_{\text{react}}(t_k)$, and then again to $(O_{\text{DE}}-D_{\text{react}})+\hat{O}_{\text{react}}$ with $W_0=W_{00}+W_{01}=O_{\text{DE}}-D_{\text{react}}$ and $W_1=\hat{O}_{\text{react}}$.

In the first application of TOPE to $O_{\mathrm{DE}} - D_{\mathrm{react}}$ with $W_{00} = O_{\mathrm{DE}}$, the opererators $W_{01}(t_k) = -D_{\mathrm{react}}(t_k)$ defined at different times are all diagonal in the same (particle number basis and therefore commute:

$$[D_{\mathrm{react}}(t_i), D_{\mathrm{react}}(t_j)] = 0.$$

In this circumstance, we can *simply drop* the time-ordering super-operator $(\cdots)_+$ in Equation 2 and write

$$\exp(t\,(O_{\mathrm{DE}} - D_{\mathrm{react}})) = \exp(tO_{\mathrm{DE}})\exp(-\int_0^t dt' D_{\mathrm{react}}(t')) \tag{24}$$

where, as in Equation 3, $D_{\mathrm{react}}(t') = \exp(-t'O_{\mathrm{DE}})D_{\mathrm{react}}\exp(t'O_{\mathrm{DE}})$. In our case, Equation 24 specializes to :

$$\langle w|\exp(t\left(O_{\{\mathrm{DE}\}} - D_{\mathrm{react}}\right))|z\rangle = \exp(t\sum_i v_i(\{z\})\nabla_{z_i})\exp(-\int_0^t dt' D_{\mathrm{react}}(t'))\delta(\{w\}-\{z\})$$
$$= \exp\left(-\int_0^t dt' D_{\mathrm{react}}\left(z(0) + \int_0^{t'} v(\{z\})dt''\right)\right)\delta(w-\left(z(0) + \int_0^t v(z(t'))dt'\right)) \tag{25}$$

This result looks very similar to Equation 22 applied to

$$f(z) = \exp\left(-\int_0^t dt' D_{\mathrm{react}}(t'))\delta(\{w\} - \{z\})\right),$$

and we now aim to understand and exploit this similarity.

### 3.5.3 Equivalent ODE

Consider the dynamics expressed in Equation 25. Can we obtain the first factor from ODE's alone? Yes, if we introduce a new state variable $\tau$ involved in every ODE-related rule. Set $\tau(0) = 0$ as the new variable's initial condition, and augment the ODE operators as follows

$$\begin{aligned} Z &= (z, \tau) \\ V(z) &= (v(\{z\}), -D(z)) \\ \nabla_Z &= (\nabla_z, \partial_\tau) \\ \tilde{O}_{\{\mathrm{DE}\}} &= V(Z)\nabla_Z = v(\{z\}) \cdot \nabla_z + D(z)\partial_\tau \end{aligned} \tag{26}$$

In other words, we have added a differential equation for $\tau$ to the ODE system

$$\frac{\partial x_i}{\partial t} = v_i(\{x_k\}) \quad \text{and} \quad \frac{d\tau}{dt} = D(z). \tag{27}$$

This equation is solvable in terms of a "warped time" coordinate

$$\tau(t) = \int_0^t D_{\text{react}}(z(t'))dt'. \tag{28}$$

(Cf. Equation 23.) There are degenerate cases $D_{\text{react}} = 0$ only if there are terminal states in the reaction network.

To see that this is the correct procedure, calculate from Equation 21:

$$\begin{aligned}
&\left\langle \begin{pmatrix} w \\ \tau_{\max} \end{pmatrix} \middle| \exp(t\tilde{O}_{\{\text{DE}\}})\exp(-\tau_{\max}) \middle| \begin{pmatrix} z(0) \\ \tau(0)=0 \end{pmatrix} \right\rangle \\
&= \exp\left( t \left( \sum_i v_i(\{z\})\nabla_{z_i} + D\,(z)\,\partial_\tau \right) \right) \delta(\{w\}-\{z\})\delta(\tau_{\max}-\tau)\exp(-\tau_{\max}) \\
&= \delta\left( \{w\} - \left( z(0) + \int_0^t v_i(z(t'))dt \right) \right) \delta\left( \tau_{\max} - \int_0^t D_{\text{react}}(z(t'))dt' \right) \exp(-\tau_{\max}) \\
&= \delta(\{w\} - (\text{SolutionofEquation27}, \quad \text{withIC} \quad z(0), \tau(0)=0)) \times \exp(-\tau_{\max})
\end{aligned} \tag{29}$$

This expression agrees with Equation 25, as required. But how do we insure the IC on $\tau$ ? That can be done as follows:

$$\begin{aligned}
&\left\langle \begin{pmatrix} w \\ \tau_{\max} \end{pmatrix} \middle| \exp(t\tilde{O}_{\{\text{DE}\}})\exp(-\tau_{\max}) \middle| \begin{pmatrix} z \\ 0 \end{pmatrix} \right\rangle \\
&= \left\langle \begin{pmatrix} w \\ \tau_{\max} \end{pmatrix} \middle| \exp(t\tilde{O}_{\{\text{DE}\}})\exp(-\tau_{\max}) \middle| \begin{pmatrix} z \\ 0 \end{pmatrix} \right\rangle \times \left( 1 = \int d\tau' dz' \left\langle \begin{pmatrix} z' \\ \tau' \end{pmatrix} \middle| \begin{pmatrix} z \\ \tau \end{pmatrix} \right\rangle \right) \\
&= \left\langle \begin{pmatrix} w \\ \tau_{\max} \end{pmatrix} \middle| \exp(t\tilde{O}_{\{\text{DE}\}})\exp(-\tau_{\max}) \left( \int d\tau' dz' \middle| \begin{pmatrix} z' \\ 0 \end{pmatrix} \right\rangle \left\langle \begin{pmatrix} z' \\ \tau' \end{pmatrix} \middle| \right) | \begin{pmatrix} z \\ \tau \end{pmatrix} \right\rangle \\
&= \left\langle \begin{pmatrix} w \\ \tau_{\max} \end{pmatrix} \middle| \exp(t\tilde{O}_{\{\text{DE}\}})\exp(-\tau_{\max}) P_{\tau:=0} \middle| \begin{pmatrix} z \\ \tau \end{pmatrix} \right\rangle
\end{aligned} \tag{30}$$

$$P_{\tau:=0} \equiv \int d\tau' dz' \left| \begin{pmatrix} z' \\ 0 \end{pmatrix} \right\rangle \left\langle \begin{pmatrix} z' \\ \tau' \end{pmatrix} \right|$$

is a projection operator (i.e. one that satisfies $P \cdot P = P$) that resets the variable $\tau$ to zero after each use. In summary,

$$\begin{aligned}
&\left\langle \begin{pmatrix} w \\ \tau_{\max} \end{pmatrix} \middle| \exp(t\tilde{O}_{\{\text{DE}\}})\exp(-\tau_{\max}) P_{\tau:=0} \middle| \begin{pmatrix} z \\ \tau \end{pmatrix} \right\rangle \\
&= \delta(\{w\} - (\text{SolutionofEquation27}, \quad \text{withIC} \quad z(0), \tau(0)=0)) \times \exp(-\tau_{\max})
\end{aligned} \tag{31}$$

Clearly this result is equivalent to Equation 25 and is in the correct form for a Markov chain that can represent a computation. Of course, the matrix element calculated is only relevant if $\tau_{\max}$ as drawn from the exponential is constrained to be equal to the final value of $\tau$ in the final state $\langle \binom{w}{\tau_{\max}} |$ as solved by the ODE system $\tilde{O}_{\{\mathrm{DE}\}}$. We can implement this constraint with a factor of $\delta(\tau - \tau_{\max})$ in the Markov chain over states and times. Thus a step in the Markov chain in between reactions can be written as:

$$\begin{aligned} \mathcal{W}_{\mathrm{betweenreactions}} &= \delta(\tau - \tau_{\max}) \exp(t \quad \tilde{O}_{\{\mathrm{DE}\}}) P_{\tau:=0} \exp(-\tau_{\max}) \Theta(\tau_{\max} \geqslant 0) \\ \mathcal{W} &= \hat{O}_{\mathrm{react}} \cdot \mathcal{W}_{\mathrm{betweenreactions}} \end{aligned}$$

As in the SSA derivation, the reaction step is given by factors of $\hat{O}_{\mathrm{react}}$ which need to be normalized by $D_{\mathrm{react}}$. Using $\delta(t - t_{\max}(\tau_{\max}))dt = \delta(\tau - \tau_{\max})d\tau$ and $d\tau/dt = D_{\mathrm{react}}(t)$, we find

$$\begin{aligned} M_{01} &= \exp(-\tau_{\max}) \Theta(\tau_{\max} \geqslant 0) \\ M_{00} &= \delta(t - t_{\max}(\tau_{\max})) \exp(t \quad \tilde{O}_{\{\mathrm{DE}\}}) \cdot P_{\tau:=0} \\ M_1 &= \hat{O}_{\mathrm{react}} / D_{\mathrm{react}} \\ \mathcal{W} &= M_1 \cdot M_{00} \cdot M_{01} \end{aligned} \tag{32}$$

where $\mathcal{W}$ represents the Markov chain corresponding to the simulation algorithm.

In implementations so far [4,15] we have used instead the equivalent differential operator

$$\tilde{O}_{\{\mathrm{DE}\}} = V(Z) \nabla_Z = v(\{z\}) \cdot \nabla_z - D(z) p \partial_p$$

with $p = \exp(-\tau)$, initialized at $p_0 = 1$, and a uniform distribution on $p_{\mathrm{final}} \in [0, 1]$. This variant of the ODE was reported independently in [16], though the derivation there did not proceed by general field theory techniques.

### 3.5.4 Algorithm: Hybrid SSA/ODE solver

By Equation 32 above, a Markov chain algorithm for simulating the hybrid system can be represented in the following SSA-like pseudocode:

```
factor $\rho^{(r)}(x_{\mathrm{in}}, x_{\mathrm{out}}) = k^{(r)}(x_{\mathrm{in}}) p^{(r)}(x_{\mathrm{out}} | x_{\mathrm{in}})$;
repeat {
    initialize SSA propensities as $k^{(r)}(x_{\mathrm{in}})$;
    initialize $k^{(\mathrm{total})} := \sum_r k^{(r)}(x_{\mathrm{in}})$;
    initialize $\tau := 0$ ;
    draw effective waiting time $\tau_{\max}$ from $\exp(-\tau_{\max})$
    solve ODE system, including an extra ODE updating $\tau$:
        $\frac{d\tau}{dt} = k^{(\text{total})}(t)$
        until $\tau = \tau_{\max}$
    draw reaction $r$ from distribution $k^{(r)}(x_{\text{in}})/k^{(\text{total})}$;
    draw $x_{\text{out}}$ from $p^{(r)}(x_{\text{out}}|x_{\text{in}})$ and execute reaction $r$;
} until $t \geqslant t_{\max}$
```

### 3.5.5 Application: Cell division

As a simplified model of stochastic cell division, we may consider constant growth of a linear dimension $l$ of each cell, $dl/dt = v$, coupled with a stochastic cell division rule whose propensity depends on the ratio of $l$ to a threshold length $l_0$ for likely division:

$$\text{cell}(l) \to \text{cell}(l/2), \text{cell}(l/2) \qquad \textbf{with} \qquad \rho_{\text{division}}\sigma(\beta\,(l/l_0 - 1))$$

with a sigmoidal function such as $\sigma(x) = 1/(1 + \exp(-x))$. In this model the parameter $\beta$ varies the sharpness of the threshold, and $\rho_{\text{division}}$ is the maximal propensity for division. Experimental evidence for stochastic dependence of division events on cell size in plant cells is reviewed in [17].

The differential equation for length can also be put in the form of a reaction rule that includes an ODE:

$$\text{cell}(l) \to \text{cell}(l + dl) \qquad \textbf{solving} \qquad dl/dt = v(l)$$

as described in [15]. Clearly this model could be augmented with other parameters such as growth signals with their own dynamics. This was done in models of biological stem cell niches in mouse olfactory epithelium and plant root growth, using the foregoing cell division rules. These systems were studied and simulated using the hybrid SSA/ODE algorithm above, in [15,18].

### 3.5.6 Application: Time-varying propensity for complete polymerization

Consider the n-step polymerization reaction

$$\{A \to X_1 \ \textbf{with} \ k_1, X_1 \to X_2 \ \textbf{with} \ k_2, ..., X_{n-1} \to B \ \textbf{with} \ k_n\}$$
$$\tau_i = \tau/n \quad \text{and} \quad k_i = nk.$$

There is an $n^{(\max)}$ . Then

$$\hat{W} = \lambda \hat{a}_{n+1}$$
$$W = \lambda((\hat{a}_{n+1} + c_{n+1}) - I_{n+1}) = \lambda(\hat{b}_{n+1} - I_{n+1})$$

where $c_{n+1}$ is all zeros except for a "1" entry in the lower right corner. Since $\hat{b}$ and $I$ are matrices that commute, $\exp(tW) = \exp(t\lambda\hat{b}_{n+1})\exp(-t\lambda I_{n+1})$ and we easily compute

$$P(t|\tau, n) = [\exp(tW)]_{1,n+1} = \frac{\lambda^n t^{n-1} e^{-\lambda t}}{(n-1)!}$$

This is the distribution on polymer completion times. It is an Erlang distribution (a Gamma distribution with integral values of $n$). If $\tau$ is held fixed and $n$ tends towards infinity, this distribution approaches a delta function $\delta(t-\tau)$, which can lead to differential-delay equation models for reaction networks involving polymerization processes such as transcription [19]. This probability distribution for termination times also corresponds to the time-varying propensity function

$$\rho(t|\tau, n) = P(t|\tau, n) / \left[1 - \int_0^t P(t|\tau, n) dt\right] = \frac{\lambda^n t^{n-1} e^{-\lambda t}}{\Gamma(n, t\lambda)}, \tag{33}$$

which increases monotonically in time.

As in Equation 32, the resulting time-varying propensity still fits within the framework of a Markov chain $\mathcal{W}(I, t'|J, t)$ that advances the time variable by an increment that is a random variable. The method of the previous section can be used to implement an SSA-like algorithm, with differential equations that govern propensities replaced by algebraic equations (Equation 33) or, if differential equations are also present, by differential-algebraic equations.

Figure 2 goes about here.

Figure 2. Erlang-derived time-dependent propensities for completion of a multistage process $\tau = 1, n \in \{1, ..., 10\}$. Horizontal axis: time, $t$. Vertical axis: propensity, $\rho(t|\tau, n)$. Plots for varying $n$ are superimposed. For larger $n$ there is a "maturation" phenomenon whereby completion at small times is very unlikely, and when a process is "overdue" for completion then its propensity becomes very high. By comparison, propensities for very small $n$ increase rapidly at first and are then relatively flat.

#### 3.5.7 Extended Application: Tissue-level model of *Arabidopsis* root growth

A full tissue-level model of a hybrid SSA/ODE system has been presented in [18], which details a mathematical model of auxin growth hormone patterning along the developing root of the plant *Arabidopsis thaliana,* including the pattern formation system in the root apical meristem (RAM). The model was first formulated using a fixed 1D geometry of cells along the central "stele" of the root, including both passive diffusion of auxin originating in the above ground part of the plant, and more importantly autoregulated active transport of auxin.

This much of the model is formulated using ordinary differential equations and spatial discretization at the scale of one cell.

However the real root involves cell growth, division and possibly biomechanics in an essential way, so the model was reimplemented in the "Plenum" implementation of the "Dynamical Grammars" modeling language. Dynamical Grammars support parameterized rules such as those of Section 3.5 above at multiple scales (eg cellular and/or molecular scales), and the Plenum implementation [15] uses the foregoing hybrid SSA/ODE algorithm as an essential part of its simulation engine. It also uses a data structure of pattern-matched objects (somewhat akin to that of the Rete algorithm [12]) for efficient handling of the variable-binding involved when there are many rules in a grammar, some of which include repeated variable names. The resulting root growth and patterning model includes rules for cell growth, cell division, mechanical forces between neighboring cells in 1D, cell death at the tip of the root, auxin influx from the shoot, production of a hypothetical second morphogen "$Y$" possibly playing a role similar to cytokinin, autoregulated active transport of auxin between neighboring cells, passive transport of auxin and $Y$ between neighboring cells, degradation of auxin and $Y$, and dilution of auxin and $Y$ due to cell growth. There are a total of 13 grammar rules that specify the foregoing mechanisms, with one or two rules per listed mechanism. As in the previous cell division example, each rule is either of "solving" keyword ODE type or of "with" keyword discrete event type. We now present the first four rules of this model.

In the root model there is just one type of parameterized object, a cell. Each cell carries its own internal state information in the form of the values of an ordered list of parameters, each of which is constrained to be of some type (often integers or real values) associated with a measure that can be summed or integrated over. In the plant root model, the parameter types of a cell object are as follows:

$$\text{cell}[\text{currID} : \mathbb{N}, \text{mode} : \mathbb{N}, l : \mathbb{R}, r : \mathbb{R}, A : \mathbb{R}, Y : \mathbb{R}, \text{prevID} : \mathbb{N}, \text{nextID} : \mathbb{N}]$$

Here currID is the integer-valued (or "integer-typed", currID : $\mathbb{N}$) unique identification number (ID) of the current cell, prevID is the integer-valued ID of the previous cell in the 1D line, nextID is the integer-valued ID of the next cell, mode is an integer-valued label specifying the cell's internal growth state, $l$ is the real-valued ($l : \mathbb{R}$) current cell length, $r$ is its real-valued size or "radius", $A$ is its real-valued concentration of auxin, $Y$ is its real-valued concentration of hypothetical substance $Y$. Here $A$ and $Y$ could alternatively be typed as nonnegative integers in a stochastic molecular simulation, but that was not the modeling choice in this investigation. A slightly simplified version of the rules for cell growth, cell division, biomechanics, and passive diffusion of chemical species between neighboring cells is:

**grammar** root {

/* cell growth: */

$$\text{cell}[\text{curr}, \text{mode}, x, r, A, Y, \text{prev}, \text{next}] \longrightarrow \text{cell}[\text{curr}, \text{mode}, x, r + dr, A, Y, \text{prev}, \text{next}]$$

**solving** $\{dr/dt = 1/\tau_{\text{cycle}}\}$

```
/* cell biomechanics (point masses, dissipation dominated) */

  $C_1 = \mathrm{cell}[\mathrm{curr}, \mathrm{mode}, x, r, A, Y, \mathrm{prev}, \mathrm{next}]$,
  $C_2 = \mathrm{cell}[\mathrm{next}, \mathrm{mode}', x', r', A', Y', \mathrm{curr}, \mathrm{nextnext}]$
  $\longrightarrow C_1 = \mathrm{cell}[\mathrm{curr}, \mathrm{mode}, x + dx, r, A, Y, \mathrm{prev}, \mathrm{next}], C_2$
       solving  $\{dx/dt = -\partial_x V_{\mathrm{spring}}(x - x', r, r')\}$
  /*plus similar rule exchanging next and prev ; $dx/dt$ just adds up over rules */

/* switch from growth mode (mode=1) to division-waiting mode (=2): */

  $\mathrm{cell}[\mathrm{curr}, 1, x, r, A, Y, \mathrm{prev}, \mathrm{next}] \longrightarrow \mathrm{cell}[\mathrm{curr}, 2, x, r, A, Y, \mathrm{prev}, \mathrm{next}]$
       with  $\rho_{\mathrm{stop}} / \left(1 + \exp(-(r - r_{\mathrm{lim}}))/T_{\mathrm{div}}\right)$

/* cell replication, preserving 1D structure: */

    $\mathrm{cell}[\mathrm{curr}, 2, x, r, A, Y, \mathrm{prev}, \mathrm{next}], \mathrm{cell}[\mathrm{prev}, \mathrm{mode}', x', r', A', Y', \mathrm{prevprev}, \mathrm{curr}]$
       $\mathrm{cell}[\mathrm{next}, \mathrm{mode}, x'', r'', A'', Y'', \mathrm{curr}, \mathrm{nextnext}]$
    $\longrightarrow \mathrm{cell}[\mathrm{new}_1, 1, x - r + 2r\alpha + r(1-\alpha), r(1-\alpha), A, Y, \mathrm{prev}, \mathrm{new}_2]$,
       $\mathrm{cell}[\mathrm{new}_2, 1, x - r + r\alpha, r\alpha, A, Y, \mathrm{new}_1, \mathrm{next}]$,
       $\mathrm{cell}[\mathrm{prev}, \mathrm{mode}', x', r', A', Y', \mathrm{prevprev}, \mathrm{new}_1]$,
       $\mathrm{cell}[\mathrm{next}, \mathrm{mode}, x'', r'', A'', Y'', \mathrm{new}_2, \mathrm{nextnext}]$
  with  $\left[\mathrm{base} + \mathrm{ampl}\,(Y/Y_0) \,/\, \left(1 + (Y/Y_0)^5\right]\right. \times \Theta(\frac{1}{2} + \Delta \leqslant \alpha \leqslant \frac{1}{2} + \Delta)$

/* auxin/Y passive transport between two neighboring cells: */

    $C_1 = \mathrm{cell}[\mathrm{curr}, \mathrm{mode}, x, r, A, Y, \mathrm{prev}, \mathrm{next}]$,
    $C_2 = \mathrm{cell}[\mathrm{next}, \mathrm{mode}', x', r', A', Y', \mathrm{curr}, \mathrm{nextnext}]$
    $\longrightarrow C_1 = \mathrm{cell}[\mathrm{curr}, \mathrm{mode}, x, r, A + dA, Y + dY, \mathrm{prev}, \mathrm{next}]$,
    $C_2 = \mathrm{cell}[\mathrm{next}, \mathrm{mode}', x', r', A' + dA', Y' + dY', \mathrm{curr}, \mathrm{nextnext}]$
              solving  $\{dA/dt = D_A(A' - A), dA'/dt = D_A(A - A')$,
                        $dY/dt = D_Y(Y' - Y), dY'/dt = D_Y(Y - Y')\}$

}
```

The actual code for these rules is given in Appendix III. It is written using the Plenum implementation [15] of the Dynamical Grammars framework [4]. Plenum is embedded in the *Mathematica* computer algebra problem-solving environment. Thus, ordinary and partial derivatives as used above are actually a part of the language. The full model file is available as Supplementary Data to this paper. Repeated variables on the left hand side (LHS) must have identical values for the rule to apply. This situation occurs in the cell biomechanics, cell replication and passive transport rules above, where left-right cell neighbor pairs point to one another by sharing ID parameter values like “curr”, “prev” and “next” in the first and last two parameter positions. By contrast, there is no repetition of variables on the LHS of the autonomous cell growth or cell mode-switching rules above, since they have only one object on the LHS of each rule. Algorithmically such repeated variable matching is achieved by symbolic pattern matching or variable-binding; mathematically it is expressed by operator integrals such as Equation 18. The coordinate system used in this example may seem "backwards" since it is a minor convention that roots grow from left to right, but that the quiescent center near the right tip is the origin of coordinates.

The Plenum implementation also performs several symbolic computations including variable-binding for efficient implementation of rules with repeated variables (Equation 18 above and the present extended example), and aggregating the ODE $d\tau/dt = k^{(\mathrm{tota})}(t)$ of Section 3.5.4 by adding up the symbolic expressions from the individual rules.

Selected pattern formation snapshots are shown in Figure 3. The phenomenology of the resulting simulations, and of the actual root observations with which they largely agree, is discussed in [18]. Root apical meristem is an example of a stem cell niche in plants. A somewhat more complex stem cell niche model for mouse olfactory epithelium, in two dimensions using Plenum, is given in [15].

Figures 3a,b,c go about here.

Figure 3. Snapshots of the root growth model, showing cell positions along the horizontal axis (root tip to the right), and concentrations of auxin (solid red curve with one or two peaks) and hypothetical substance $Y$ (dashed blue curve with one or two peaks) with increasing time. Cell state (1=idling in preparation for cell division, vs. 0=growth) is shown in green dotted curve. Parameters are: $\rho_{\mathrm{stop}} = 1, \mathrm{base} = .005, \mathrm{ampl} = 100, Y_0 = 5, r_{\mathrm{lim}} = 1, T_{\mathrm{div}} = .01,$ $\Delta = .2, D_A = 0.08, D_Y = 0.16$. Some parameter sets including this one develop extra auxin peaks to the left of the Quiescent Center (∼rightmost blue peak), which may specify the location for a new lateral root. Full interpretation of this model is given in [18].

## 4 Conclusion and outlook

We have shown that the time-ordered product expansion (TOPE) can be used systematically to derive computational simulation and parameter-fitting algorithms for stochastic systems, connecting two seemingly distant areas of research. In doing so we have developed the means to translate formally between field theory language and the language of computable Markov chains in which randomized algorithms can be expressed and derived. By this means we hope to open the door to the use of TOPE and related methods from quantum and statistical field theory in the computational simulation of stochastic biochemical kinetics, with broad applicability in physically based biology. The particular hybrid stochastic process/ordinary differential equation simulation algorithm derived here is very different from interleaving and operator splitting algorithms which are intrinsically approximate; instead, this algorithm is exact in the same sense that SSA is (that is, it draws from the same distribution of just-fired reactions), except for any errors introduced by the ODE solver and in the solver's detection of the ODE stopping criterion, which is that an auxiliary variable reaches a threshold value. A future prospect for the field theory approach is application to reaction-diffusion systems in which the propagator for particles between reactions is the heat kernal Green's function for the diffusion equation. The result may be an alternative avenue for derivation of novel particle-based, off-grid stochastic numerical solvers for reaction-diffusion problems as treated in [2], which, like the algorithms shown here, are also amenable to generalizations to exact "leaping" acceleration and to hybrid stochastic/differential equation solution algorithms.

## 5 Acknowledgements

Research was supported by NIH grants R01 GM086883 and P50 GM76516 to UC Irvine. I also wish to acknowledge the hospitality, travel support, and research environments provided by the Center for Nonlinear Studies (CNLS) at the Los Alamos National Laboratory, the Sainsbury Laboratory Cambridge University, and the Pauli Center for Theoretical Studies at ETH Zürich and the University of Zürich. This paper is dedicated to my father, Raymond C. Mjolsness, physicist and mathematician, on the occasion of his 80th birthday.

## A References

[1] Doi, J. (1976) Second quantization representation for classical many-particle system. 1976 J. Phys. A: Math. Gen. 9 1465 .

[2] Doi, J. (1976) Stochastic theory of diffusion-controlled reaction 1976 J. Phys. A: Math. Gen. 9 1479 .

[3] Mattis D.C. and Glasser M.L. (1998) The uses of quantum field theory in diffusion-limited reactions. Rev.Mod. Phys. 70, 979–1001

[4] Mjolsness E. and Yosiphon G (2006) Stochastic Process Semantics for Dynamical Grammars. Annals of Mathematics and Artificial Intelligence, 47(3-4)

[5] H. M. Fried (2002), *Green's Functions and Ordered Exponentials*, Cambridge University Press

[6] C. M. Bender, S. F. Brandt, J.-H. Chen, and Q. Wang (2005), "Ghost Busting: PT-Symmetric Interpretation of the Lee Model". Physical Review D 71, 025014

[7] Xueying Zhang, Katrien De Cock, Mónica F. Bugallo, and Petar M. Djurić (2005), A general method for the computation of probabilities in systems of first order chemical reactions. J. Chem. Phys. 122, 104101 (2005).

[8] Yosiphon G. and Mjolsness E. (2010) Towards the Inference of Stochastic Biochemical Network and Parameterized Grammar Models. In Learning and Inference in Computational Systems Biology, (Lawrence N., Girolami M., Rattray M., and Sanguinetti G., Eds.) MIT Press.

[9] Darren J. Wilkinson (2006). *Stochastic Modelling for Systems Biology.* Chapman & Hall/CRC Press, Boca Raton, Florida.

[10] E. Mjolsness, D. Orendorff, P. Chatelain, P. Koumoutsakos (2009), “An Exact Accelerated Stochastic Simulation Algorithm”, Journal of Chemical Physics 130, 144110.

[11] Orendorff, David (2012). “Exact and Hierarchical Reaction Leaping: Asymptotic Improvements to the Stochastic Simulation Algorithm”. PhD thesis, UC Irvine Computer Science Department, June 2012. Thesis available at: http://computableplant.ics.uci.edu/~dorendor/thesis .

[12] Forgy C. (1982) Rete: A fast algorithm for the many pattern/many object pattern match problem. Artificial Intelligence, (19):17–37.

[13] Hlavacek WS, Faeder JR, Blinov ML, Posner RG, Hucka M, Fontana W (2006) Rules for modeling signal-transduction systems. Science’s STKE 2006:re6.

[14] Danos V, Feret J, Fontana W, Harmer R, Krivine J (2007) Rule-based modelling of cellular signaling. Lect Notes Comput Sci 4703:17-41.

[15] Yosiphon, G. (2009), “Stochastic Parameterized Grammars: Formalization, Inference, and Modeling Applications”, PhD Thesis, UC Irvine Computer Science Department, June 2009. Thesis and software :
http://computableplant.ics.uci.edu/theses/guy .

[16] Crudu A, Debussche A, Radulescu O (2009). Hybrid stochastic simplifications for multiscale gene networks. BMC Systems Biology 3:89.

[17] Roeder, A. H. K. (2012), When and where plant cells divide: a perspective from computational modeling. Current Opinion in Plant Biology 2012, 15:1–7 .

[18] V.V. Mironova, Nadya A Omelyanchuk, Guy Yosiphon, Stanislav I Fadeev, Nikolai A Kolchanov, Eric Mjolsness and Vitaly A Likhoshvai (2010). “A plausible mechanism for auxin patterning along the developing root”. BMC Systems Biology 4:98.

[19] Likhoshvai, V. A.; Demidenko, G. V.; Fadeev, S. I. (2006), Modeling of Gene Expression by the Delay Equation. Bioinformatics of Genome Regulation and Structure II (2006): Part 3, 421-431, DOI: 10.1007/0-387-29455-4_40 .

[20] Wang Y, Christley S., Mjolsness E., and Xie X. (2010) Parameter inference for discretely observed stochastic kinetic models using stochastic gradient descent. BMC Systems Biology 4:99.

## B Abbreviations list

| | | |
|---|---|---|
| IC | - | Initial Condition |
| ODE | - | Ordinary Differential Equation |
| MC | - | Markov Chain |
| SSA | - | (Gillespie) Stochastic Simulation Algorithm |
| TOPE | - | Time-Ordered Product Expansion |
| LHS | - | Left Hand Side |
| RHS | - | Right Hand Side |

## C Appendix I: Bayesian inference derivation

### C.1 Semigroup property

Here we provide the omitted details for Section 2.4. For nonnegative times $t = t_1 + t_2$, any time-evolution equation must obey the semigroup property:

$$\Pr(I|J,t) = \sum_K \Pr(I|K,t_2)\Pr(K|J,t_1)$$

i.e.

$$\left[\exp(t\left(\hat{W} - D\right))\right]_{I,J} = \sum_K \left[\exp(t_2\left(\hat{W} - D\right))\right]_{I,K} \left[\exp(t_1\left(\hat{W} - D\right))\right]_{K,J}.$$

Is there a $k$-event version of this rule, for $k = k_1 + k_2$? We observe (where again $[\tau_q]_0^k \equiv [\tau_0, ...\tau_k]$ )

$$\Pr(I,k|J,t) = \int_0^t \cdots \int_0^t \left(\Pi_{q=0}^k d\tau_q\right) \Pr(I,[\tau_q]_0^k,k|J,t)$$

and we calculate for $0 \leqslant k_1 \leqslant k$:

$$\sum_K \int_0^t d\tau \Pr(I,\left[\tau'_{k_1},\tau_{k_1+1},...\tau_k\right],k_2|K,\tau)\Pr(K,[\tau_q]_0^{k_1},k_1|J,t-\tau)$$

$$= \int_0^t d\tau \left\{ \left[\prod_{q=k\downarrow}^{k_1+1} \exp(-\tau_q D)\ \ \hat{W}\right] \exp(-\tau'_{k_1} D)\right.$$

$$\left. \times\delta\left(\tau - \left(\sum_{q=k_1+1}^{k} \tau_q + \tau'_{k_1}\right)\right) \exp(-\tau_{k_1} D) \left[\prod_{q=k_1-1\downarrow}^{0} \hat{W}\quad \exp(-\tau_q D)\right] \delta(t - \tau - \sum_{q=0}^{k_1-1} \tau_q)\right\}_{I,J}$$

$$= \left\{ \left[\prod_{q=k\downarrow}^{k_1+1} \exp(-\tau_q D)\ \ \hat{W}\right] \exp(-(\tau'_{k_1} + \tau_{k_1})D)\right.$$

$$\left. \times \left[\prod_{q=k_1-1\downarrow}^{0} \hat{W}\quad \exp(-\tau_q D)\right] \delta\left(t - \left(\sum_{q=k_1+1}^{k} \tau_q + \tau'_{k_1} + \sum_{q=0}^{k_1-1} \tau_q\right)\right)\right\}_{I,J}$$

$$= \Pr(I,\left[[\tau_q]_0^{k_1-1},\tau'_{k_1}+\tau_{k_1},[\tau_q]_{k_1+1}^k\right],k|J,t)\quad .$$

Thus, if $k = k_1 + k_2$ and for any $\tau'_{k_1} \in [0,\tau_{k_1}]$, there is a semigroup law:

$$\Pr(I,[\tau_q]_0^k,k|J,t) = \sum_K \int_0^t d\tau \Pr(I,\left[\tau'_{k_1},\tau_{k_1+1},...\tau_k\right],k_2|K,\tau)$$
$$\times \Pr(K,\left[\tau_0,...\tau_{k_1-1},\tau_{k_1}-\tau'_{k_1}\right],k_1|J,t-\tau).$$

In this derivation there is an arbitrary choice of $\tau'_{k_1}$from the interval $[0,\tau_{k_1}]$.

## C.2 Bayesian recurrence relation

Here we provide the omitted details for Section 3.1.1 . We seek a version of the semigroup law that pertains to $\Pr(I, t|k, J)$ rather than to $\Pr(I, k|J, t)$. This is achieved by a somewhat involved application of Bayes' rule.

We observe (where again by definition $[\tau_q]_0^k \triangleq [\tau_0, ...\tau_k]$ )

$$\Pr(I, [\tau_q]_0^k, k|J, t) = \left[\exp(-\tau_k D) \prod_{q=k-1\downarrow}^{0} \hat{W} \quad \exp(-\tau_q D)\Theta(\tau_q \geqslant 0)\right]_{I,J} \delta\left(t - \sum_{q=0}^{k} \tau_q\right)$$

so we may define (where $J = I_0$ and $I = I_k$ and $D_{I_q} \triangleq D_{I_q I_q}$)

$$\Pr([I_q]_1^k, [\tau_q]_0^k, k|J, t) \triangleq \exp(-\tau_k D_{I_k}) \left[\prod_{q=0}^{k-1} \hat{W}_{I_{q+1} I_q} \quad \exp(-\tau_q D_{I_q})\Theta(\tau_q \geqslant 0)\right] \delta\left(t - \sum_{q=0}^{k} \tau_q\right)$$

We seek a simple expression for $\Pr(I, t|k, J)$, and claim that with suitable caveats it will be determined recursively by

$$\Pr(I, t|k = 1, J) = \hat{W}_{IJ} \quad \exp(-tD_J),$$

the two factors of which have inverse cancelling normalizations. The obstacle to overcome is that, from the Bayesian point of view, simultaneous knowledge of the simulation end time and final event number can trickle backwards and influence the distribution of likely event firing times at earlier times and event numbers - a completely nonphysical artifact. To avoid this effect we must be careful to ask the right questions for Bayesian inference to answer. To begin with we consider simulation ending times $T$ much longer than event times $t$that we wish to sample. All events after event $k$ at time $t$ are assumed to be of no interest, so we integrate them out. All earlier events are assumed to be known already, so we conditionalize over them. This is the correct Bayesian way to introduce time asymmetry into the global distribution $\Pr([I_q]_1^k, [\tau_q]_0^k, k|...)$ of entire trajectories $([I_q]_1^k, [\tau_q]_0^k)$, above.

The strategy then is to consider large times $T \gg t$ which are overwhelmingly likely to have large reaction numbers $n \gg k$; then to marginalize the probability distribution $\Pr([I], [\tau], n|J, T)$ over all event numbers $n > k$ and to conditionalize it over all event numbers $q < k$.

### C.2.1 Marginalizing

Define

$$\hat{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1}|k, J, T) \triangleq \sum_{n=k+1}^{\infty} \sum_{\{[I_q]_{k+1}^n\}} \int_0^T \cdots \int_0^T [d\tau_q]_k^n \Pr([I_q]_1^n, [\tau_q]_0^n, n|J, T) \tag{34}$$

This is a "just-fired" probability, in which any wait times $\tau$ and events after the $k$th event are integrated out (marginalized).

In the limit $T \to \infty$ only summands with $n \gg k$ will contribute (assuming terminal states have been formally eg. by adding an extra, isolated, slow, reversible reaction). First, is this object really a probability distribution? Clearly every value is nonnegative. They also add up to one in the limit $T \to \infty$:

$$\begin{aligned}
&\sum_{\{[I_q]_1^k\}} \int_0^T [d\tau_q]_0^{k-1} \hat{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1}|k, J, T) \\
&\qquad = \sum_{n=k+1}^{\infty} \sum_{\{[I_q]_1^n\}} \int_0^T [d\tau_q]_0^n \Pr([I_q]_1^n, [\tau_q]_0^n, n|J, T) \\
&\qquad \xrightarrow[T\to\infty]{} \sum_{n=0}^{\infty} \sum_{\{[I_q]_1^n\}} \int_0^T [d\tau_q]_0^n \Pr([I_q]_1^n, [\tau_q]_0^n, n|J, T) \quad = 1 \qquad (35)
\end{aligned}$$

due to the normalization of $\Pr([I_q]_1^n, [\tau_q]_0^n, n|J, T)$. So, $\tilde{\Pr}(\cdots|k, J) = \lim\limits_{T\to\infty} \hat{\Pr}(\cdots|k, J, T)$ is also a probability density function.

Next we compute $\hat{\Pr}(\cdots|J, T)$ using TOPE:

$$\begin{aligned}
\hat{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1}|k, J, T) &= \sum_{n=k+1}^{\infty} \sum_{\{[I_q]_{k+1}^n\}} \int_0^T \cdots \int_0^T [d\tau_q]_k^n \delta\left(T - \sum_{q=0}^{n} \tau_q\right) \\
&\quad \times \exp(-\tau_n D_{I_n}) \left[\prod_{q=0}^{n-1} \hat{W}_{I_{q+1}I_q} \quad \exp(-\tau_q D_{I_q}) \Theta(\tau_q \geqslant 0)\right] \\
= \sum_{n=k+1}^{\infty} \sum_{\{[I_q]_{k+1}^n\}} \int_0^T &\cdots \int_0^T [d\tau_q]_k^n \delta\left(T - \sum_{q=0}^{k-1} \tau_q - \sum_{q=k}^{n} \tau_q\right) \exp(-\tau_n D_{I_n}) \\
&\times \left[\prod_{q=0}^{k-1} \hat{W}_{I_{q+1}I_q} \quad \exp(-\tau_q D_{I_q}) \Theta(\tau_q \geqslant 0)\right] [\prod_{q=k}^{n-1} \hat{W}_{I_{q+1}I_q} \quad \exp(-\tau_q D_{I_q})]
\end{aligned}$$

But now the product $[\prod\limits_{q=0}^{k-1} \cdots]$ is a common factor and can be moved out of all the integrals and sums. Thus

$$\hat{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1}|k, J, T) = \left[\prod_{q=0}^{k-1} \hat{W}_{I_{q+1}I_q} \quad \exp(-\tau_q D_{I_q}) \Theta(\tau_q \geqslant 0)\right] \mathcal{I}\left(I_k, k, J, T - \sum_{q=0}^{k-1} \tau_q\right),$$

(the first factor of which is independent of $T$), where

$$\mathcal{I}(I_k, k, J, T') \triangleq \sum_{n=k+1}^{\infty} \sum_{\{[I_q]_{k+1}^n\}} \int_0^T \cdots \int_0^T [d\tau_q]_k^n \delta\left(T' - \sum_{q=k}^{n} \tau_q\right) \times \exp(-\tau_n D_{I_n}) \left[\prod_{q=k}^{n-1} \hat{W}_{I_{q+1} I_q} \quad \exp(-\tau_q D_{I_q})\right].$$

If we define new dummy variables ${I'}_q \triangleq I_{q+k}$, ${\tau'}_q \triangleq \tau_{q+k}$, and $n' \triangleq n - k$, then ${\tau'}_{n'} = \tau_n$ , ${I'}_{n'} = I_n$, and

$$\begin{aligned}\mathcal{I}(I_k, k, J, T') &= \sum_{n'=1}^{\infty} \sum_{\{[{I'}_q]_1^{n'}\}} \int_0^T \cdots \int_0^T [d{\tau'}_q]_k^{n'} \, \delta\left(T' - \sum_{q=k}^{n'} {\tau'}_q\right) \\ &\times \exp(-{\tau'}_n D_{{I'}_n}) \left[\prod_{q=0}^{n'-1} \hat{W}_{{I'}_{q+1} {I'}_q} \quad \exp(-{\tau'}_q D_{{I'}_q})\right] \\ &= \mathbf{1} \cdot \left(e^{T'(\hat{W}-D)} - e^{-T'D}\right) \cdot \delta({I'}_0 - J) \\ &= 1 - \mathbf{1} \cdot e^{-T'D} \cdot \delta({I'}_0 - I_k),\end{aligned}$$

by adding and subtracting the missing $n' = 0$ summand and using TOPE again. Now we can take limits:

$$\lim_{T\to\infty} \mathcal{I}(I_k, k, J, T') = 1,$$

and

$$\begin{aligned}\tilde{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1} | k, J) &\triangleq \lim_{T\to\infty} \hat{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1} | k, J, T) \\ &= \prod_{q=0}^{k-1} \left(\hat{W}_{I_{q+1} I_q} \quad \exp(-\tau_q D_{I_q}) \Theta(\tau_q \geqslant 0)\right).\end{aligned} \tag{36}$$

As a special case, for $k = 1$ we find $\tilde{\Pr}([I_1], [\tau_0] | 1, J) = \hat{W}_{I_1 I_0} \quad \exp(-\tau_0 D_{I_0}) \Theta(\tau_0 \geqslant 0)$ .

### C.2.2 Conditionalizing

If $2 \leqslant k < n$, Bayes' Rule implies:

$$\hat{\Pr}(I_k, \tau_{k-1} | [I_q]_1^{k-1}, [\tau_q]_0^{k-2}, k, J, T) = \frac{\hat{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1} | k, J, T)}{\hat{\Pr}([I_q]_1^{k-1}, [\tau_q]_0^{k-2} | k, J, T)} \tag{37}$$

The denominator is a new quantity (since the $k$'s don't match up the way they do in the numerator) and it is the integral of the numerator that normalizes the left hand side. It can be evaluated in the limit of large $T$:

$$\begin{aligned}\hat{\Pr}([I_q]_1^{k-1}, [\tau_q]_0^{k-2}|k, J, T) &= \sum_{I_k} \int_0^T d\tau_{k-1} \hat{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1}|k, J, T) \\ &= \sum_{n=k+1}^{\infty} \sum_{\{[I_q]_k^n\}} \int_0^T [d\tau_q]_{k-1}^n \Pr([I_q]_1^n, [\tau_q]_0^n, n|J, T) \\ \xrightarrow[T\to\infty]{} & \lim_{T\to\infty} \sum_{n=k}^{\infty} \sum_{\{[I_q]_k^n\}} \int_0^T [d\tau_q]_{k-1}^n \Pr([I_q]_1^n, [\tau_q]_0^n, n|J, T) \\ &= \tilde{\Pr}([I_q]_1^{k-1}, [\tau_q]_0^{k-2}|k-1, J)\end{aligned}$$

from Equation 34 and Equation 36. As before, the second step is justified by the fact that $n \leqslant k$ has probability that approaches zero as $T \to \infty$. Defining

$$\check{\Pr}(I_k, \tau_{k-1}|[I_q]_1^{k-1}, [\tau_q]_0^{k-2}, k, J) \overset{\triangle}{=} \lim_{T\to\infty} \hat{\Pr}(I_k, \tau_{k-1}|[I_q]_1^{k-1}, [\tau_q]_0^{k-2}, k, J, T)$$

we find (from Equation 37)

$$\begin{aligned}\check{\Pr}(I_k, \tau_{k-1}|[I_q]_1^{k-1}, [\tau_q]_0^{k-2}, k, J) &= \lim_{T\to\infty} \frac{\hat{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1}|k, J, T)}{\hat{\Pr}([I_q]_1^{k-1}, [\tau_q]_0^{k-2}|k, J, T)} \\ &= \frac{\lim\limits_{T\to\infty} \hat{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1}|k, J)}{\lim\limits_{T\to\infty} \hat{\Pr}([I_q]_1^{k-1}, [\tau_q]_0^{k-2}|k, J)}\end{aligned}$$

since for valid nonnegative $\tau$s the ratio of limits exists and is finite, as we will shortly see. Thus

$$\begin{aligned}\check{\Pr}(I_k, \tau_{k-1}|[I_q]_1^{k-1}, [\tau_q]_0^{k-2}, k, J) &= \frac{\tilde{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1}|k, J)}{\tilde{\Pr}([I_q]_1^{k-1}, [\tau_q]_0^{k-2}|k-1, J)} \\ &= \frac{\prod_{q=0}^{k-1} \left(\hat{W}_{I_{q+1}I_q} \quad \exp(-\tau_q D_{I_q})\Theta(\tau_q \geqslant 0)\right)}{\prod_{q=0}^{k-2} \left(\hat{W}_{I_{q+1}I_q} \quad \exp(-\tau_q D_{I_q})\Theta(\tau_q \geqslant 0)\right)} \\ &= \hat{W}_{I_k I_{k-1}} \quad \exp(-\tau_{k-1} D_{I_{k-1}})\Theta(\tau \geqslant 0).\end{aligned}$$

The last line is actually independent (in the functional rather than probabilistic sense of "independent") of the quantitites $[I_q]_1^{k-2}, [\tau_q]_0^{k-2}$, and $k$, so we can drop all these arguments from $\tilde{\Pr}(\cdots)$. Restating,

$$\check{\Pr}(I_k, \tau_{k-1}|I_{k-1}) = \hat{W}_{I_k I_{k-1}} \quad \exp(-\tau_{k-1} D_{I_{k-1}})\Theta(\tau_{k-1} \geqslant 0)$$

or

$$\check{\Pr}(I, \tau|K) = \hat{W}_{IK} \quad \exp(-\tau D_K)\Theta(\tau \geqslant 0).$$

Importantly, this expression is equal to $\tilde{\Pr}(I, \tau|k = 1, K)$ as calculated at the end of the last section. Also the recursive statement of the Bayesian recurrence property Equation 37 becomes:

$$\tilde{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1}|k, J) = \check{\Pr}(I_k, \tau_{k-1}|I_{k-1})\tilde{\Pr}([I_q]_1^{k-1}, [\tau_q]_0^{k-2}|k-1, J).$$

Since $\check{\Pr}(I, \tau|J) = \tilde{\Pr}(I, \tau|k = 1, J)$, we find for all $k \geqslant 2$ the Bayesian recurrence relation in terms of $\tilde{\Pr}(\cdots)$ alone:

$$\tilde{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1}|k, J) = \tilde{\Pr}(I_k, \tau_{k-1}|1, I_{k-1})\tilde{\Pr}([I_q]_1^{k-1}, [\tau_q]_0^{k-2}|k-1, J). \quad (38)$$

## C.3 Markov Chain Derivation

Here we provide the omitted details for Section 3.1.2 . Continuing from the foregoing Bayesian recurrence property (Equation 38), we now sum over all $I_q$ except $I_k = I$ and $I_0 = J$, and integrate over all $\tau_q$ , the following equation:

$$\tilde{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1}|k, J)\delta\left(t_k - \sum_{q=0}^{k-1}\tau_q\right) = \tilde{\Pr}(I_k, \tau_{k-1}|1, I_{k-1}) \times \tilde{\Pr}([I_q]_1^{k-1}, [\tau_q]_0^{k-2}|k-1, J)\delta\left(t_k - \sum_{q=0}^{k-1}\tau_q\right).$$

We define and calculate

$$\tilde{\Pr}(I, t_k|k, J) \triangleq \sum_{\{[I_q]_1^{k-1}\}} \int_0^\infty \cdots \int_0^\infty [d\tau_q]_0^{k-1}\tilde{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1}|k, J)\delta\left(t_k - \sum_{q=0}^{k-1}\tau_q\right)$$

$$= \sum_{\{[I_q]_1^{k-1}\}} \int_0^\infty \cdots \int_0^\infty [d\tau_q]_0^{k-1}\tilde{\Pr}(I_k, \tau_{k-1}|1, I_{k-1})\tilde{\Pr}([I_q]_1^{k-1}, [\tau_q]_0^{k-2}|k-1, J)\delta\left(t_k - \sum_{q=0}^{k-1}\tau_q\right)$$

$$= \sum_{I_{k-1}} \int_0^\infty d\tau_{k-1}\tilde{\Pr}(I_k, \tau_{k-1}|1, I_{k-1})$$

$$\times \sum_{\{[I_q]_1^{k-2}\}} \int_0^\infty \cdots \int_0^\infty [d\tau_q]_0^{k-2}\tilde{\Pr}([I_q]_1^{k-1}, [\tau_q]_0^{k-2}|k-1, J)\delta\left(t_k - \tau_{k-1} - \sum_{q=0}^{k-1}\tau_q\right)$$

$$= \sum_{I_{k-1}} \int_0^\infty d\tau_{k-1}\tilde{\Pr}(I_k, \tau_{k-1}|1, I_{k-1})\tilde{\Pr}(I_{k-1}, t_k - \tau_{k-1}|k-1, J)$$

This $\tilde{\Pr}(I, t_k|J)$ is also probability density:

$$\sum_{I_k} \int_0^\infty dt_k \tilde{\Pr}(I_k, t_k|k, J) = \sum_{\{[I_q]_1^k\}} \int_0^\infty \cdots \int_0^\infty [d\tau_q]_0^{k-1} \tilde{\Pr}([I_q]_1^k, [\tau_q]_0^{k-1}|k, J) = 1$$

as shown in Equation 35 above, using the definition of Equation 36. Summarizing its Markov property:

$$\tilde{\Pr}(I, t|k, J) = \sum_K \int_0^t d\tau \tilde{\Pr}(I, \tau|1, K) \tilde{\Pr}(K, t-\tau|k-1, J). \tag{39}$$

## D Appendix II: Maximum likelihood parameter inference

Application of the TOPE to maximum-likelihood parameter learning in stochastic reaction networks has previously been presented [20]. Here, for completeness of presentation for a different audience, we just show the essential gradient calculation step.

Suppose we have observations of the state of a chemical reaction network at times $\{t_s\}$, and wish to improve the probability $P(\text{Data}|\text{Model})$ of a reaction network model for the flow of probability at intermediate times. We will use the TOPE for each time interval in between observation times $t_s$:

$$\left[e^{(t_{s+1}-t_s)\tilde{W}}\right](x(t_{s+1}), x(t_s)) = \sum_{k=0}^\infty \int_{t_s}^{t_{s+1}} \cdots \int_{t_s}^{t_{s+1}} d[\tau]_0^n \delta\left((t_{s+1}-t_s) - \sum_{p=0}^n \tau_p\right) \times \left[e^{\tau_n \tilde{D}} \hat{W} ... e^{\tau_1 \tilde{D}} \hat{W} e^{\tau_0 \tilde{D}}\right](x(t_{s+1}), x(t_s)) \tag{40}$$

We will need to compute the derivatives of this probability with respect to reaction rates:

$$\rho_r \frac{\partial}{\partial \rho_r} [e^{(t_{s+1}-t_s)W}](x(t_{s+1}), x(t_s)) = \sum_{k=0}^\infty \int_{t_s}^{t_{s+1}} \cdots \int_{t_s}^{t_{s+1}} d[\tau]_0^n \delta\left((t_{s+1}-t_s) - \sum_{p=0}^n \tau_p\right)$$
$$\times \sum_{p=0}^n \left[e^{-\tau_n D} \hat{W} ... e^{-\tau_{p+1} D} (\rho_r \hat{W}_r) e^{-\tau_p D} ... e^{-\tau_1 D} \hat{W} e^{-\tau_0 D}\right](x(t_{s+1}), x(t_s))$$
$$- \sum_{k=0}^\infty \int_{t_s}^{t_{s+1}} \cdots \int_{t_s}^{t_{s+1}} d[\tau]_0^n \delta\left((t_{s+1}-t_s) - \sum_{p=0}^n \tau_p\right)$$
$$\times \sum_{p=0}^n \left[e^{-\tau_n D} \hat{W} ... \hat{W} (\rho_r \tau_p D_r) e^{-\tau_p D} \hat{W} ... e^{-\tau_1 D} \hat{W} e^{-\tau_0 D}\right](x(t_{s+1}), x(t_s))$$

$$\rho_r [\hat{W}_r]_{IJ} = \left(\frac{\rho_r [\hat{W}_r]_{IJ}}{\sum_r \rho_r [\hat{W}_r]_{IJ}}\right) \left(\left[\hat{W}\right]_{IJ}\right) = b_{rIJ} [\hat{W}]_{IJ}$$

where we defined the "branching ratio"

$$b_{rIJ} \equiv \left( \frac{\rho_r [\hat{W}_r]_{IJ}}{\sum_r \rho_r [\hat{W}_r]_{IJ}} \right) = \left\langle \delta_{r,R(I,J)} \right\rangle_{p(I|J)}$$

for reaction $r$ in state $J$, *assuming* each reaction $r$ results in just one output state $I$ per input state $J$. Here $R(I, J)$ is the random variable denoting the actual reaction chosen in transitioning from state $J$ to state $I$. Then

$$\rho_r \frac{\partial}{\partial \rho_r} [e^{(t_{s+1}-t_s)\tilde{W}}] \left(x(t_{s+1}), x(t_s)\right) = \sum_{k=0}^{\infty} \int_{t_s}^{t_{s+1}} \cdots \int_{t_s}^{t_{s+1}} d[\tau]_0^n \delta \left( (t_{s+1} - t_s) - \sum_{p=0}^{n} \tau_p \right)$$
$$\times \sum_{p=0}^{n} \left[ e^{\tau_n D} \hat{W} ... e^{\tau_{p+1} D} (b_r \hat{W} - \hat{W} \rho_r \tau_p D_r) e^{\tau_p D} ... e^{\tau_1 D} \hat{W} e^{\tau_0 D} \right] \left(x(t_{s+1}), x(t_s)\right)$$

or

$$\rho_r \frac{\partial}{\partial \rho_r} [e^{(t_{s+1}-t_s)\tilde{W}}] \left(x(t_{s+1}), x(t_s)\right)$$
$$= \sum_{k=0}^{\infty} \sum_{p=0}^{n} \langle b_r(\text{reaction event } p \text{ out of } n) \rangle_{\hat{W}, x(t_{s+1}), x(t_s)}$$
$$- \rho_r \sum_{k=0}^{\infty} \sum_{p=0}^{n} \langle \tau_p D_r \rangle_{\hat{W}, x(t_{s+1}), x(t_s)}$$

This finally is a quantity that is easy to compute as a running average during a simulation of the network with incorrect values of the parameters, thereby contributing to the calculation of an improved set of parameter values in a stochastic gradient descent algorithm. This is the key update equation in a learning algorithm for reaction rates in stochastic biochemical networks (extensible to other process networks). Algorithmic details can be found in [20], noting particularly Equation 2.4 therein. A related stochastic learning algorithm is proposed in [8].

## E Appendix III: Dynamical grammar for root growth

Given the following function simplified definitions among others:

```
gGrowthModelMult = 1;
growthConst = 1 / gCellCyleTime;
yEffectOnDivisionFunc[y_] := Module[{δ, h1, h2},
```

$$0.005 + 100 * \frac{(\frac{y}{q1})^{p_{v1}}}{1 + (\frac{y}{q2})^{p_{v2}}} \;/.\; \{q1 \rightarrow 5,\; q1 \rightarrow 1,\; p_{v1} \rightarrow 1,\; p_{v2} \rightarrow 5\}$$

```
]
```

```
cellGrowthLocFunc[rad_] := Module[{},
gGrowthModelMult * growthConst
];
springXFunc[curPosx_, curRad_, nbrPosx_, nbrRad_] :=
- ∂curPosx springPotential[curPosx, curRad, nbrPosx, nbrRad]
```

The actual grammar for selected rules is shown here:

```
gRootGrowth := Grammar[rules→
{
(* continuous change in cell c1 radius *)
{c1 Equal cell[cellID1, 1(* growth mode *), loc1, rad1, auxin1, y1, cellIDP, cellIDN]}→ c1,
solving[rad1′EqualcellGrowthLocFunc[rad1]],

(* continuous change in cell c1 location *)
{c1 Equal cell[cellID, cMode, loc, rad, auxin, y, cellIDPrev, cellIDNext],
c2 Equal cell[cellIDNext, cModeN, locN, radN, auxinN, yN, cellID, cellIDNN]}→ {c1, c2},
solving[loc′EqualgGrowthModelMult * springXFunc[loc, rad, locN, radN]],

(* continuous change in cell c1 location *)
{c1 Equal cell[cellID, cMode, loc, rad, auxin, y, cellIDPrev, cellIDNext],
c2 Equal cell[cellIDPrev, cModeP, locP, radP, auxinP, yP, cellIDPP, cellID]}→ {c1, c2},
solving[loc′EqualgGrowthModelMult * springXFunc[loc, rad, locP, radP]],

(* change cell mode from growth to wait, when over a radius threshold *)
cell[cellID, 1, loc, rad, auxin, y, cellIDPrev, cellIDNext] →
cell[cellID, 2, loc, rad, auxin, y, cellIDPrev, cellIDNext],
with[gGrowthModelMult * stopGrowthConst * grammarSigmoid[rad - gLimitCellRad, gDivideTemp]],

(* divide a cell when its in wait mode *)
cell[cellID, 2, loc, rad, auxin, y, cellIDPrev, cellIDNext] → {
cell[cellIDPrev, cModeP, locP, radP, auxinP, yP, cellIDPP, cellID] →
cell[cellIDPrev, cModeP, locP, radP, auxinP, yP, cellIDPP, grammarCreateObjectID[1]],
cell[cellIDNext, cModeN, locN, radN, auxinN, yN, cellID, cellIDNN] →
cell[cellIDNext, cModeN, locN, radN, auxinN, yN, grammarCreateObjectID[2], cellIDNN],
cell[grammarCreateObjectID[1], 1, loc - rad + 2rad * cellpart + rad * (1 - cellpart),
rad * (1 - cellpart), auxin, y, cellIDPrev, grammarCreateObjectID[2]],
cell[grammarCreateObjectID[2], 1, loc - rad + rad * cellpart,
rad * cellpart , auxin, y, grammarCreateObjectID[1], cellIDNext]},
with[gGrowthModelMult * yEffectOnDivisionFunc[y] *
grammarPDF[UniformDistribution[{0.5 - gRangeParam, 0.5 + gRangeParam}], cellpart]],

(* ... more rules ... *)

(* auxin / y passive transport between two neighboring cells *)
{c0 Equal cell[cellID0, cMode0, loc0, rad0, auxin0, y0, cellIDP0, cellID1] ,
c1 Equal cell[cellID1, cMode1, loc1, rad1, auxin1, y1, cellID0, cellIDNext]}→ {c0, c1},
solving[auxin1′Equal pt( auxin0 - auxin1), auxin0′Equal pt( auxin1 - auxin0) ,
y1′Equal pty( y0 - y1), y0′Equal pty( y1 - y0)],

(* ... more rules ... *)

}];
```

Note that for efficiency, the symbolic partial derivative is taken out of the grammar (rule 3, biomechanics) and precomputed. Also, the cell division rule above actually has the form of a compound rule, whose right hand side comprises two further rules. This point was simplified out of the notation in the main text. It is an efficiency measure that allows a rule firing to be a multistep process (similar to the subgrammars or macros of [4]) without slowing down the computational identification of cells likely to divide specified by the **with** clause of the rule. However, its use here relies on the dynamically invariant, domain-specific fact that each cell is the $n$th neighbor (in this case $n$=1 or 2) of at most one other cell.

## F Figure legends

Figure 1. A time history of the reaction $A + B \rightleftharpoons C$. Time flows left to right. Open circles represent reaction events, with probability factor $\times W_1$. In between reaction events are unimolecular particle propagators $\exp((t_k - t_{k-1})W_0)$, labelled by arrows and particle names (repeated for clarity). This is a non-spatial version of the Lee model in quantum field theory (cf. for example [6]).

Figure 2. Erlang-derived time-dependent propensities for completion of a multistage process $\tau = 1, n \in \{1, ..., 10\}$. Horizontal axis: time, $t$. Vertical axis: propensity, $\rho(t|\tau, n)$. Plots for varying $n$ are superimposed. For larger $n$ there is a "maturation" phenomenon whereby completion at small times is very unlikely, and when a process is "overdue" for completion then its propensity becomes very high. By comparison, propensities for very small $n$ increase rapidly at first and are then relatively flat.

Figure 3. Snapshots of the root growth model, showing cell positions along the horizontal axis (root tip to the right), and concentrations of auxin (solid red curve with one or two peaks) and hypothetical substance $Y$ (dashed blue curve with one or two peaks) with increasing time. Cell state (1=idling in preparation for cell division, vs. 0=growth) is shown in green dotted curve. Parameters are: $\rho_{\text{stop}} = 1, \text{base} = .005$, $\text{ampl} = 100, Y_0 = 5$, $r_{\text{lim}} = 1, T_{\text{div}} = .01$, $\Delta = .2, D_A = 0.08, D_Y = 0.16$. Some parameter sets including this one develop extra auxin peaks to the left of the Quiescent Center ($\sim$rightmost blue peak), which may specify the location for a new lateral root. Full interpretation of this model is given in [18].

Figure 1:

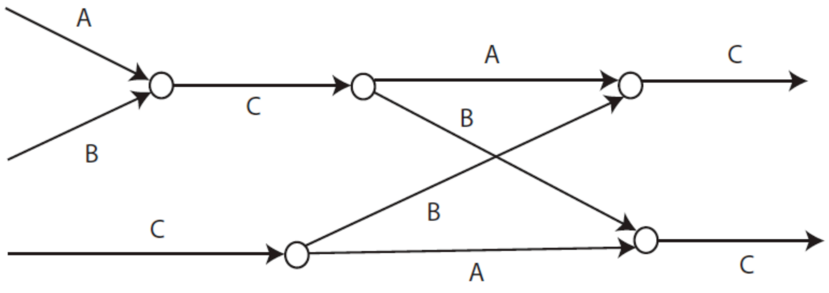


Figure 2:

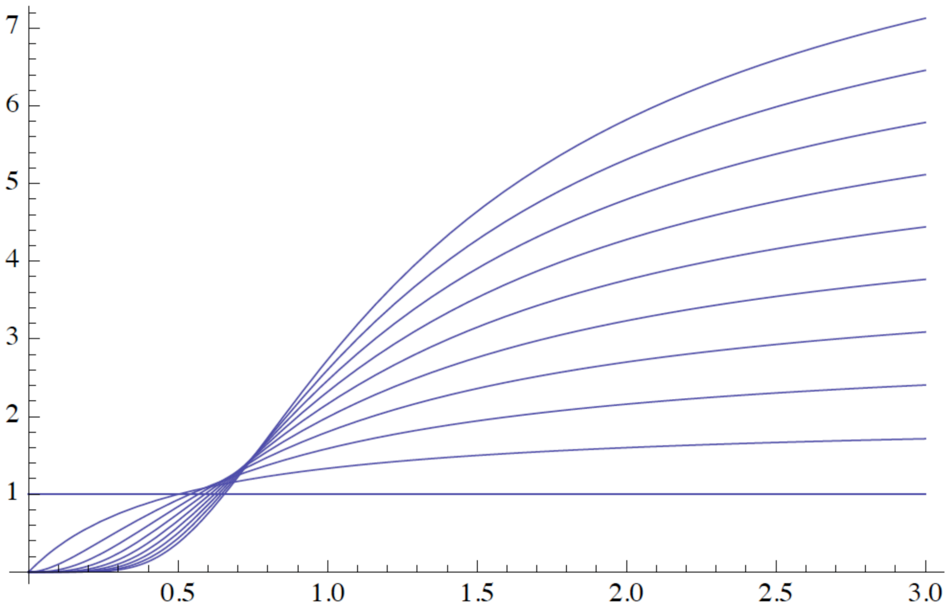
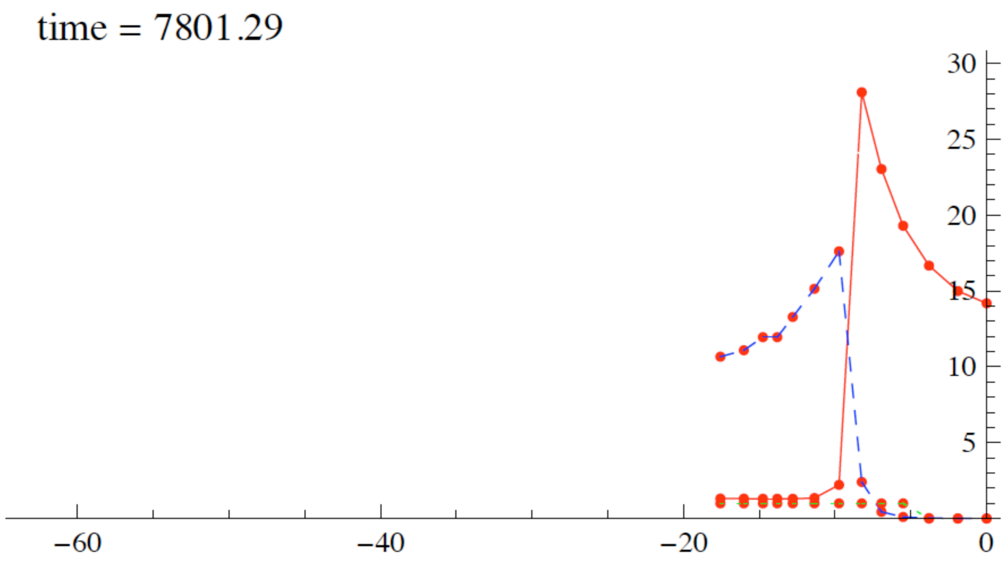
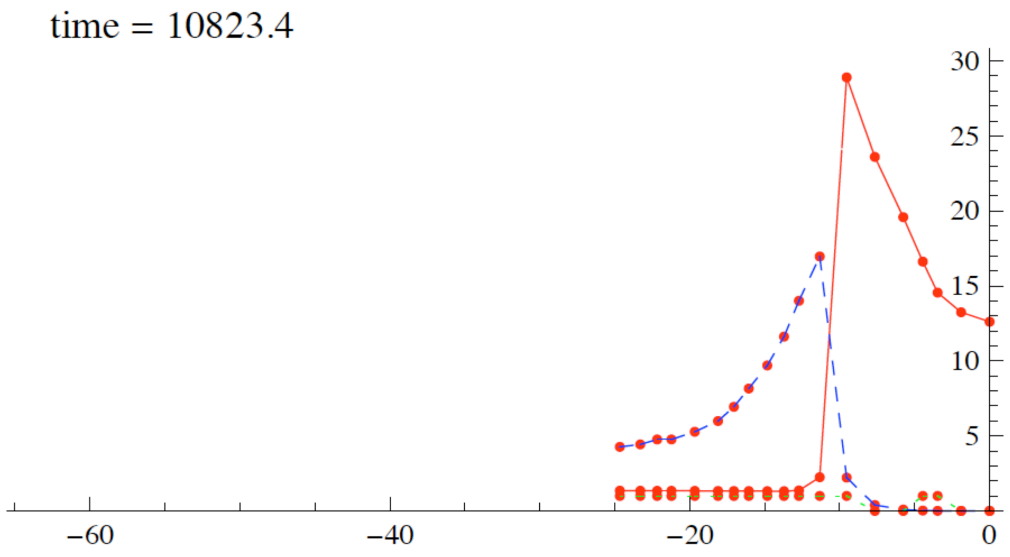


Figure 3, panels a,b, and c

3a:

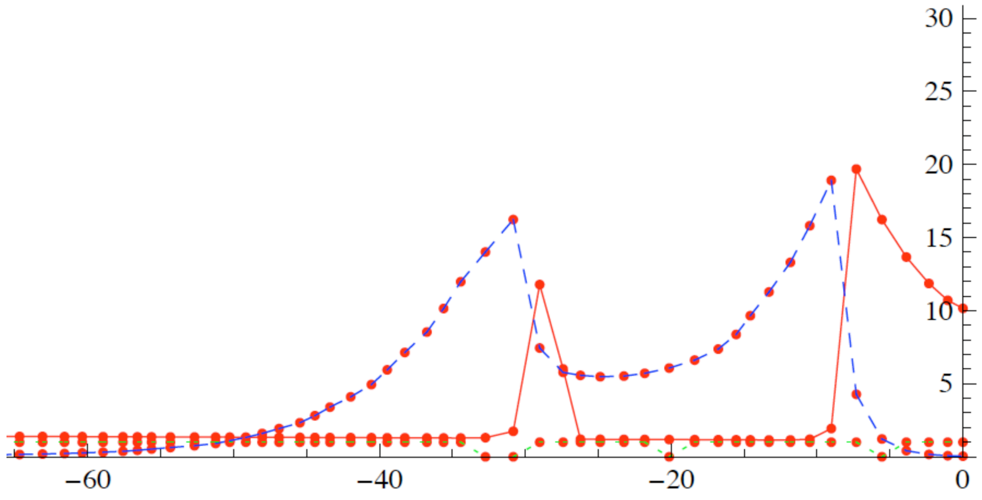


3b:



3c:

time = 19843.3


30
25
20
15
10
5
−60
−40
−20
0