Supplementary Methods: The Next Subvolume Method

The algorithm described below generates exact realizations of the Markov process described by the reaction diffusion master equation. An early version of the algorithm was presented at the SPIE conference on fluctuations and noise in biological, biophysical and biomedical systems in Santa Fe 2003 [1].

Short explanation for those who are familiar with the "next reaction method"[2] and the "direct method" [3] The reaction and diffusion rates in a single subvolume are given by the numbers of the different types of molecules that it contains. The time for the next event in each subvolume, *i.e.* a chemical reaction or a diffusion jump out from it, can thus be calculated individually by the *direct method*. The question is in which subvolume an event occurs first. To identify the subvolume where the first event occurs, we use the *next reaction method*. A single event can change the state of only one or two subvolume(s). If the event was a chemical reaction, the next event time has to up-dated only for the subvolume where it occurred. If the event was a diffusion jump out, next event times have to be up-dated for the subvolume from which the jump occurred and for the subvolume to which the molecule jumped.

The algorithm

Initialization

1. Generate a *connectivity matrix* (Fig. M2) that describes the geometry of the system.
2. Distribute the initial numbers of molecules between the subvolumes and store in the *configuration* matrix. This can be done randomly or according to any initial distribution.
3. Calculate the sum of reaction rates r_i for each subvolume i and store in the *rate matrix*. The reaction rates are calculated for the size Δ of the subvolume, as in the reaction-diffusion master equation.
4. Calculate the sum of diffusion rates for each subvolume, $s_i = n_i \sum_{j=1}^{M} d_j X_j^i$, where $d_j = D_j / \ell^2$ is the diffusion rate constant for species j. X_i^j is the number of molecules of species j in subvolume i. M is the number of species. n_i is the number of directions in which the molecules can diffuse. Store s_i in the *rate matrix*.
5. For each subvolume i: (a.) sum r_i+s_i, (b.) generate a random number, *rand*, uniformly distributed between 0 and 1 and (c.) calculate the first event time for each subvolume as $\tau_i = -\ln(rand)/(r_i + s_i)$.
6. Make an initial ordering of the subvolumes according to their next event times. The subvolumes are kept sorted in a binary tree (an *event queue*, see below) so that the subvolume for which the event occurs first is on the top and all branches have increasing event times.

Iterations

7. Assume that λ is the subvolume in which the next event occurs at time $t = \tau_\lambda$ according to the top element of the event queue. Generate a random number *rand* uniformly distributed between 0 and 1, choose a chemical reaction if *rand*< $r_\lambda/(r_\lambda + s_\lambda)$ and otherwise a diffusion jump.
8. Reaction event:
    a. Rescale *rand* from (7.) linearly to [0,1] to determine which reaction occurred as in the direct method.
    b. Update the state of the subvolume λ in the configuration matrix according to the state changes by reaction.
    c. Recalculate the sum $r_\lambda + s_\lambda$ for the subvolume λ, generate a new random number and calculate the time of the next event as $t_\lambda = -\ln(rand)/(r_\lambda + s_\lambda) + t$.
    d. Insert the active subvolume's new event time in the event queue and order the queue (see below).
9. Diffusion event:
    a. Rescale *rand* from (7.) linearly to [0,1] to determine which type of molecule that diffused away. The diffusion intensities are given by the numbers of the different types of molecules weighted by their diffusion rate constants.
    b. The direction of the diffusion event is chosen by randomly selecting a column in the connectivity matrix. This can be done by rescaling the random number used in 9a (again).
    c. Update the states of both subvolume λ and its neighbor, γ, that got an additional molecule.
    d. Recalculate the sums $r_\lambda + s_\lambda$ and $r_\gamma + s_\gamma$, sample the time to the next event in the subvolumes as in 8c.
    e. Insert the subvolumes' new event times in the event queue and order the queue (see below).
10. Return to 7. for the next iteration.

Further Improvements

The algorithm can be modified in a number of ways for particular systems depending on tradeoffs between memory usage and speed. E.g., it is possible to store all reaction rates for each subvolume, such that they do not have to be recalculated in 8a. Alternatively, it is possible to sample the reaction or diffusion event that will occur next simultaneously with the event time in 8c and 9d. The draw-back is that a random number is wasted if a molecule diffuses into the subvolume before its event time.

Another possible improvement is to reuse the event time for the subvolume where the state changed because a molecule diffused into it in step 9d. Gibson and Bruck [2] proved that the old event time can be reused, without sampling a new random number. In the procedure above it is re-sampled every time. Let γ be the subvolume for which the state has changed because a molecule diffused into it, so that its total rate changed from $r_{\gamma,old}+ s_{\gamma,old}$ to $r_{\gamma,new}+ s_{\gamma,new}$. The next event time τ_{new} can be recalculated as $\tau_{new} = \left(r_{\gamma,old} + s_{\gamma,old}\right) / \left(r_{\gamma,new} + s_{\gamma,new}\right)\left(\tau_{old} - t\right) + t$, instead of as $-\ln(rand)/(r_{\gamma,new} + s_{\gamma,new}) + t$, which would require an additional random number. See [2] for more details. This improvement has not been used in the present simulations.

For reaction systems where only a few of the reaction rates in a subvolume are affected by a state change it may be advantageous to implement the Next Reaction Method [2] to sample the reaction or diffusion event in each subvolume. The procedures described above use the Direct Method [3] at the level of subvolumes.

Blue et al. [4] and Wong and Easton [5] have earlier described another binary tree search algorithm for exact Monte Carlo simulation of the master equation [3, 6], which could possibly be useful also for reaction diffusion problems as suggested by Breuer et al. [7].

In a direct application of the Next Reaction Method, the event times are calculated for each event, rather than for each subvolume. Further all events are ordered in the propriety queue and the geometry of the system is implicitly built into the dependency graph, that is used to keep track of how many and which event times that should be recalculated after each event. When the number of subvolumes is large, the Next Subvolume Method (NSM) is more efficient than a direct application of the Next Reaction Method (NRM) to the RDME. The major reason is that the NRM requires an exceptionally large dependency graph to describe which and how many event times that need to be recalculated after each event. The size of this data structure would cause memory problems. Furthermore, in the NSM the number of elements in the priority queue is equal to the number of subvolumes, C, and the queue is reordered twice for each diffusion event. In the NRM the number of elements in the priority queue is approximately C(6N+R), where N is the number of diffusing species and R is the number of different reactions in a subvolume. This larger queue must at average be reordered more than 12 times for each diffusion event.

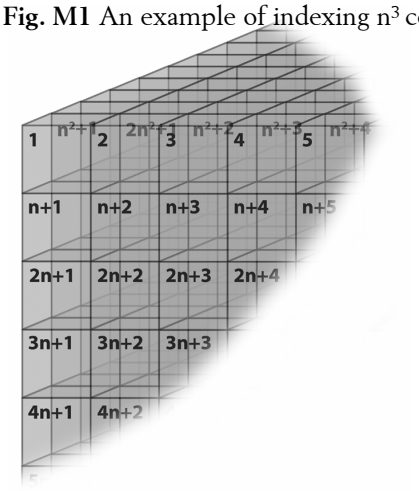
The event queue

The event queue allows identification of the subvolume in which a next reaction will occur without searching through scheduled reaction times for all subvolumes or keeping them all sorted, which would take a time proportional to the number of subvolumes (*N*). The event queue data structure is a binary tree, in which each element contains the index of a subvolume and the time for its next event, provided that no molecule enters by diffusion. The queue is ordered so that an element with an earlier event time is higher up on a branch. When the event time for a subvolume is changed, its position is changed up or down in the tree. When it gets an earlier time it changes place with the cell above until the branch is ordered. When it gets a later time it changes place with the subvolume below with the earliest scheduled time until the branch is ordered. Therefore it will take maximally $\log_2(N)$ swaps per iteration to keep the queue sorted. The branched structure is conveniently stored in a *queue array*, where each row is an element of the queue. The elements above element *k* in the queue are thus placed on a row with index "(k/2) truncated to an integer" and the elements below have the row indices 2k and 2k+1. Each element in the queue is listed with a reference from an array sorted on subvolume number. This array is necessary to identify the element in the queue that corresponds to the neighbor of the active subvolume.

The connectivity matrix and boundary conditions

In order to rapidly find the subvolume number of a neighboring subvolume we generate a look-up table; the connectivity matrix. Each row in the matrix corresponds to one subvolume and the subvolume indices are conveniently chosen as the row numbers. The subvolume indices for each of the six neighbors are stored in different columns. This determines the geometry of the system. Using the connectivity matrix, one obtains the index for the subvolume where a molecule diffuses by randomly choosing a column in the row corresponding to the active subvolume. Periodic or closed boundaries are simply created by assigning the appropriate neighbors. For closed boundaries diffusion can be directed back to the same subvolume by assigning the row index to some element in the row.

Fig. M1 An example of indexing n^3 cells.



i	*n1*	*n2*	*n3*	*n4*	*n5*	*n6*
1	2	1	3	1	5	1
2	2	1	4	2	6	2
3	4	3	3	1	7	3
4	4	3	4	2	8	4
5	6	5	7	5	5	1
6	6	5	8	6	6	2
7	8	7	7	5	7	3
8	8	7	8	6	8	4

Connectivity matrix

#A	*#B*	*#C*
10	2	0
9	1	3
5	0	2
7	1	1
4	0	2
7	1	3
8	2	4
5	0	2

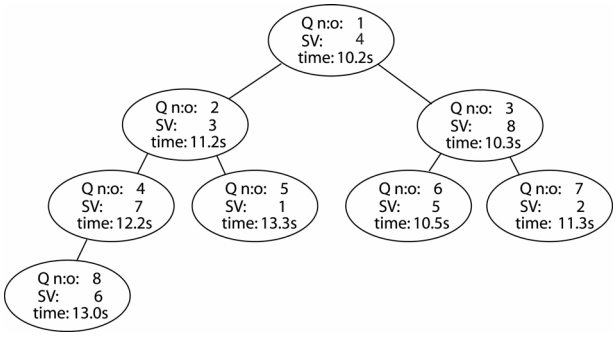
Configuration

r_i [s^{-1}]	s_i [s^{-1}]	r_i+s_i [s^{-1}]
2.2	10	12.2
4.2	11.3	15.5
2.3	5.4	7.3
1.4	6.4	7.8
0.4	4.3	4.7
0.5	10.3	10.8
1.0	13.3	14.3
5.3	5.4	10.7

Rate matrix

Q
5
7
2
1
6
9
4
3

Q-array



Event Queue

Position in Queue (Q)	Subvolume (SV)	τ_i (s)
1	4	10.2
2	3	11.2
3	8	10.3
4	7	12.2
5	1	13.3
6	5	10.5
7	2	11.3
8	6	13.0

Fig. M2. Data structures The structures within solid borders are arrays used in the algorithm. The *connectivity matrix* (N×6) stores the neighboring subvolumes' indices (*n1-n6*) for each subvolume (*rows*). This defines the geometry and boundary conditions for the system. The *configuration matrix* (N×M) stores the present number of molecules of each species in each cell. The *rate matrix* (N×3) stores the sum of reaction rate constants (r) and the sum of diffusion rate constants (d). The Q array keeps a reference to the subvolume's position in the *event queue*. In the *event queue* the subvolumes are ordered such that the one with the first scheduled event time (t) is at the top and each branch is sorted with increasing event times.

Justification

At time t the probability that any event will occur in subvolume m between $t+\tau$ and $t+\tau+\Delta\tau$ and that no event occurs in any subvolume before time τ is

$$\Delta\tau P(m,\tau) = \Delta\tau\sum_i r_i^m \times \prod_{j,n}\exp(-r_j^n\tau) = \Delta\tau\sum_i r_i^m \times \exp\left(-\sum_n\sum_j r_j^n\tau\right). \quad (1)$$

Here, $\Delta\tau r_i^m$ is the probability that event i in subvolume m will occur during the short time $\Delta\tau$. $\Delta\tau\sum_i r_i^m$ is the probability that any event in subvolume m will occur during the short time $\Delta\tau$. $\exp(-r_j^n\tau)$ is the probability that reaction j in subvolume n has not occurs during time τ. $\prod_{j,n}\exp(-r_j^n\tau)$ is the probability that no event has occurred in any subvolume during time τ.

If the total rate of events in a subvolume m $\sum_i r_i^m = a_m$, Eq. (1) reduces to

$$\Delta\tau P(m,\tau) = \Delta\tau a_m \times \exp\left(-\sum_n a_n\tau\right) \quad (2)$$

This expression for the probability that the next event occurs in subvolume m between $t+\tau$ and $t+\tau+\Delta\tau$ is equivalent to the expression sampled with the Next Reaction Method [2]. The Next Reactions Method can therefore be used to determine in which sub volume the next event will occur as well as the time of this event.

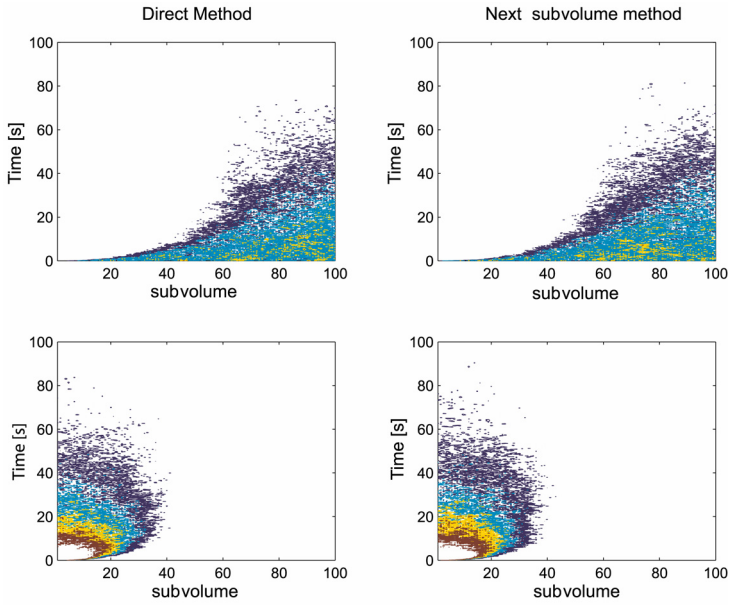
Next, we sample which event actually occurred in proportion to the rates, r_i^m, of the events in subvolume m, i.e. the occurrence of event i in subvolume m between t+τ and t+τ +$\Delta\tau$ is sampled with probability

$$\Delta\tau P(i,m,\tau) = \Delta\tau P(i|m)P(m,\tau) = \Delta\tau\left(\frac{r_i^m}{a_m}\right)a_m \times \exp\left(-\sum_n a_n\tau\right) = \Delta\tau r_i^m \times \exp\left(-\sum_n a_n\tau\right). \quad (3)$$

It can now be seen that Eq. (3) is equivalent to the probability distribution sampled by Gillespie's direct method [3]. This means that the method can be applied also to spatially homogenous reaction networks with a large number of sparsely connected sub-networks, where the sub-networks have high internal connectivity.

Fig M3. Comparison between the Next Subvolume Method and the Direct Method.

To illustrate in a concrete case the equivalence of the algorithm to previous exact methods for simulation of the Markov process described by the master equation, we have simulated the reaction $A+B\xrightarrow{k_1}\varnothing$ with the Direct Method [3] and the Next Subvolume Method [this work]. The volume is 0.4μm x 0.4μm x 40μm and it is divided into 1x1x100 subvolumes. $k_1=10^8$ $M^{-1}s^{-1}$ and $D=5\cdot10^{-8}$ cm^2s^{-1}. Initially, 1000 A molecules are evenly distributed over the subvolumes, whereas 1000 B molecules all are located in one of the outermost subvolumes. As time goes by all molecules are eventually annihilated. The figure shows contour plots for A molecules at the top and B molecules at the bottom. The contours corresponds to 5, 10, 15, 20 molecules.



Supplementary References

1. Elf, J., Doncic, A., and Ehrenberg, M.: Mesoscopic reaction-diffusion in intracellular siganling, *SPIE: Fluctuations and noise in Biological, Biophysical and Biomedical Systems*, 2003, **5110**:114-124.
2. Gibson, M. and Bruck, J.: Efficient exact stochastic simulation of chemical systems with many species and channels, *J Phys. Chem. A*, 2000, **104**:1876-1889.
3. Gillespie, D.: A general method for numerically simulating the stochastic time evolution of coupled chemical reactions, *J. Comp. Phys.*, 1976, **22**:403-434.
4. Blue, J.L., Beichl, I.I., and Sullivan, F.: Faster Monte Carlo simulations, *Physical Review. E. Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, 1995, **51**:R867-R868.
5. Wong, C.K. and Eatson, M.C.: *SIAM J. Comput.*, 1980, **9**:111.
6. Bortz, A., Kalos, M., and Lebowitz, J.: A New Algorithm for Monte Carlo Simulation of Ising Spin Systems, *J. Comp. Phys.*, 1975, **17**:10-8.
7. Breuer, H.P., Huber, W., and Petruccione, F.: Fast Monte Carlo algorithm for nonequilibrium systems, *Physical Review. E. Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, 1996, **53**:4232-4235.