Narrow Escape Problem in Synaptic Molecular Communications

Caglar Koca[*] Meltem Civas[†] Ozgur Akan[*†]
[*] Internet of Everything (IoE) Group
Electrical Engineering Division, Department of Engineering
University of Cambridge, CB3 0FA, Cambridge
United Kingdom
{ck542,oba21}@cam.ac.uk
[†]Koç University Center for neXt-generation Communications (CXC)
Department of Electrical and Electronics Engineering
Koç University, 34450, Istanbul
Turkey
{mcivas16,akan}@ku.edu.tr

ABSTRACT

The narrow escape problem (NEP) is a well-known problem with many applications in cellular biology. It is especially important to understand synaptic molecular communications. Active regions of synapses, also known as apposition zones, are connected to synaptic cleft through narrow slits, from which neurotransmitters can escape to or return from the cleft into the apposition zones. While neurotransmitters leakage into the cleft might be desired for the reuptake process, escaping neurotransmitters might trigger an undesired, i.e., false-positive or action potential in the post-synaptic terminal.

Obtaining analytic solutions to NEPs is very challenging due to its geometry dependency. Slight alterations in either or both shape or the size of the hole and the outer volume may cause drastic changes in the solution. Thus, we need a simulation-based approach to solve NEPs. However, NEP also requires the size of the hole to be much smaller than the dimensions of the volume. Combined with the requirement for Brownian motion, where the step size is much smaller than the dimensions of the volume, simulations can be prohibitively long, even for modern computers. Therefore, in this work, we suggest a simulation algorithm that simultaneously satisfies the NEP and Brownian motion simulation requirements. Our simulation framework can be used to quantify the neurotransmitter leakage within synaptic clefts.

CCS CONCEPTS

• **Computing methodologies** → **Modeling methodologies**.

KEYWORDS

Narrow escape problem, Molecular communication, Synaptic communication, Brownian motion, Brownian motion simulation

NANOCOM '22, October 5–7, 2022, Barcelona, Spain

ACM ISBN 978-1-4503-9867-1/22/10.
https://doi.org/10.1145/3558583.3558856

ACM Reference Format:
Caglar Koca[*] Meltem Civas[†] Ozgur Akan[*†]. 2022. Narrow Escape Problem in Synaptic Molecular Communications. In *The Ninth Annual ACM International Conference on Nanoscale Computing and Communication (NANOCOM '22), October 5–7, 2022, Barcelona, Spain.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3558583.3558856

1 INTRODUCTION

Synaptic communication (SC), which is the transmission of information in nerve endings through information-carrying neurotransmitter molecules, is one of the most ubiquitous branches of molecular communication (MC) [6]. One bottleneck of MC is that molecules in MC remain active for a long time in the channel, unlike electromagnetic (EM) communication, where EM waves passively dissipate at the end of their time slot. Two immediate problems rising from this phenomenon are intersymbol interference (ISI), where remaining molecules interfere with the next transmission, or cross-talk, where molecules leak out to other transmitting sites and interfere with their transmission. In this work, we develop a simulation framework to find a solution to the levels of cross-talk within a synapse due to the neurotransmitter spillage.

There are numerous studies in the literature studying ISI in MC, and SC [12–14]. The cross-talk is also modeled with an information communication theoretic (ICT) perspective [16, 26]. While it is possible to have inter-synaptic cross-talk, existing ICT literature mostly focuses on intra-synaptic cross-talk due to peri-synaptic astrocytes enveloping the entire synapse [18]. The experimental works and the ICT models agree that the impact of cross-talk should be minimum [1].

The existing ICT literature treats the spillover as an extra noise factor. Since only few, if any, neurotransmitters escape into the active zones, signal-to-spillover noise ratio is very high. Concurrently, the synaptic capacity is only marginally reduced due to the spillover. However, these works do not consider the stochastic nature of synaptic communications. SC is equipped with many correction mechanisms to reduce the randomness, including multiple vesicle release cites, multiple synapses between neurons and molecular reuptake [20]. Thus, for a healthy neuron, we expect the impact of the cross-talk to be rare. However, if some neurological condition affects the molecular reuptake, the effects of randomness are amplified. Many neurological conditions arise from problems

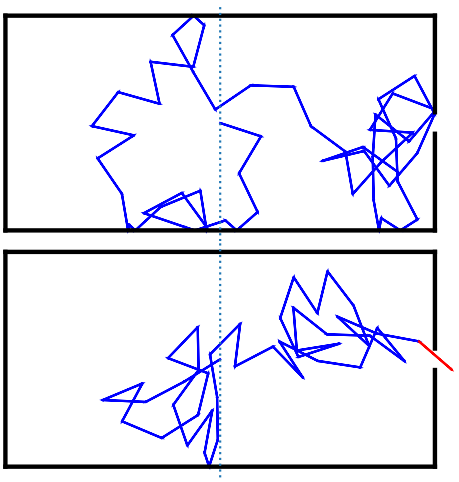

Figure 1: Two depictions of NEP of 50 iterations in 2D. The space step is too large to satisfy the BM simulation requirement. We kept the space step too large for visualisation purposes.

with the neurotransmitter reuptake, showing the importance of studying the rare events in SC [3, 5, 7, 19].

Intra-synaptic cross-talk happens when neurotransmitters released for an active zone escape into another active zone. Since the boundaries of the active zones are *narrow*, the narrow escape problem (NEP) becomes a good candidate for finding the probability of the undesired activation. NEP is a boundary value problem where a particle, whose movement is modelled as Brownian motion (BM), is confined in a volume with a very narrow opening, i.e., a hole, from which it can escape. A NEP demonstration is presented in Fig. 1.

The solutions to NEP are highly dependent on the geometry, making analytic solutions hard, even impossible [8, 22]. Simulation-approach is commonly used in MC when analytic solutions are not readily available [10, 11]. Similarly, simulation-based solutions are developed for NEP as well [9, 21].

NEP dictates that the dimensions of the hole be much smaller than the dimensions of the volume. BM simulations require that the chosen step size be much smaller than all dimensions of the volume, including the hole, making simulations to be prohibitively long. Thus, we have to choose the simulation step size adaptively to prevent the simulations from taking forever. There are numerous application-specific simulation algorithms that adaptively change the step size in BM simulations [4, 17]. However, there is still not a detailed simulation framework directly for the narrow escape. The most recent solutions for the narrow escape choose a constant step size, 1000 times smaller than the radius of a circular hole. While this choice satisfies the simulation requirements, such a simulation would be prohibitively long for practical runs [9].

In this work, we provide an adaptive simulation framework for NEP targeted toward estimating the spillover rate into the active zones. Our framework compartmentalizes the volume and calculates the effective step length. The compartmentalization takes the shape of the hole into account in step length calculations.

The rest of this paper is organized as follows. Sec. 2 offers background on narrow escape problem. Sec. 3 outlines the errors in BM simulations. Sec. 4 presents our simulation algorithm in detail. Sec. 5 includes our discussion and results. Sec. 6 concludes this paper.

2 NARROW ESCAPE PROBLEM

NEP is an exit problem, also called a hitting problem in BM, where a hitting, or *escape* happens only through a *hole*, which is narrow compared to the rest of the simulation domain [25].

NEP is also a type of mean first passage time (MFPT) problem [15]. MFPT is essential for both biological systems and MC.

We can characterize NEP as a boundary value problem with hybrid boundary conditions. Most simulated volume obeys the reflecting boundary conditions, while only the hole satisfies the absorbing boundary conditions. Therefore, it is extremely hard to solve analytically.

NEP is solved for several different geometries [23–25]. The general solution is of the form

$$\tau \sim \frac{|V|}{HD}, \tag{1}$$

where τ, $|V|$, H, and D are the mean escape time, the size of the volume, the size of the hole, and the diffusion coefficient, respectively [25]. For a rectangular $|V| = a \times b$, a hole of length $H/2$ situated in one of the corners of V, τ becomes

$$\tau = \frac{2|V|}{D\pi}\left[\log\frac{a}{H} + \log\frac{2}{\pi} + \frac{\pi b}{6a} + 2\beta^2 + O\left(\frac{H}{a}, \beta^4\right)\right], \tag{2}$$

where $\beta = e^{-\frac{\pi b}{a}}$. (2) is the closest expression to an exact solution even though the geometry is perfectly symmetrical. Thus, using simulations is a powerful approach to solve NEP.

3 BROWNIAN MOTION DISCRETIZATION ERROR

Discretization of BM has been studied for several decades. The primary motivation behind these studies is that many BM problems do not have an analytic solution. As a result, computational approaches are relied upon. However, computational resources are not infinite; thus, we need to choose the largest step size for the acceptable discretization errors.

The maximum discretization error for a BM, $B(t)$, within $0 \leq t' \leq t$ is

$$\varepsilon_{\Delta t}(t) = \max_{0 \leq t' \leq t} \left(B(t') - B_{\Delta t}(t')\right) \approx \sqrt{2\sigma^2 \Delta t \ln\left(\frac{t}{\Delta t}\right)}, \tag{3}$$

where σ^2 is the volatility or variance of $B(t)$ and Δt is the time step of the discretization [2]. Since we are more interested in BM as a measure of molecular movement, we replace σ^2 with $2D$, where D is the diffusion constant. Thus, (4) becomes

$$\varepsilon_{\Delta t}(t) = \max_{0 \leq t' \leq t} \left(B(t') - B_{\Delta t}(t')\right) \approx \sqrt{4D\Delta t \ln\left(\frac{t}{\Delta t}\right)}. \tag{4}$$

For B to move within the next time step *almost surely*, we choose l as

$$l = \sqrt{2D\Delta t}. \tag{5}$$

Substituting (5) into (4) gives us the relation between the space step and the discretization error, i.e.,

$$\varepsilon_{\Delta t}(t) \approx l\sqrt{2}\sqrt{\ln\left(\frac{t}{\Delta t}\right)}. \tag{6}$$

One important implication of (3) is that errors tend to increase relatively slowly with t, i.e.,

$$\frac{\mathrm{d}}{\mathrm{d}t}\varepsilon_{\Delta t}(t) = \frac{4D\Delta t}{2t\sqrt{4D\Delta t \ln\left(\frac{t}{\Delta t}\right)}}. \tag{7}$$

Similarly,

$$\frac{\varepsilon_{\Delta t}(at)}{\varepsilon_{\Delta t}(t)} = \left(\frac{\ln(at/\Delta t)}{\ln(t/\Delta t)}\right)^{1/2}, \tag{8}$$

where $a > 1$. (7) and (8) both implies that as long as the order of t is known and a Δt is chosen accordingly, we do not need to fine tune the Δt value.

The discretization error in simulation of reflecting BM, $B(t)$ for fixed t with n time steps is

$$\mathbb{E}|B(t) - B_{t/n}(t)| \approx -\sqrt{\frac{Dt}{\pi n}}\zeta\left(\frac{1}{2}\right), \tag{9}$$

$$\approx 0.5826\sqrt{\frac{2Dt}{n}} = 0.5826\sqrt{2D\Delta t}, \tag{10}$$

where $\zeta(.)$ is the Riemann Zeta function, D is diffusion coefficient and $\Delta t = \frac{t}{n}$ is the simulation time step [2]. Substituting (5) to (10), the expected error becomes

$$\mathbb{E}|B(t) - B_{t/n}(t)| \approx 0.5826l. \tag{11}$$

(11) implies that the expected discretization error is approximately half of the discretization length. In fact, both (6) and (11) implies that BM errors are proportional to l. While it is not possible to compare them as one is the maximum error, and the other is the expected error; however, we deduce that discretization errors are on the order of l.

4 A SIMULATION FRAMEWORK FOR THE NARROW ESCAPE PROBLEM

As we state in Sec. 1, simulating NEP is particularly challenging due to the length requirements for the step size. In this section, we first present the mathematical description of the problem and the flow of the algorithm. Then, we describe the modules of our algorithm for NEP simulations. Throughout this section, we use a variety of symbols in our calculations. Even though we explain all symbols inline, we also present them in Table 1.

4.1 NEP Simulation Framework

We know that for an accurate BM simulation, the step size, l, should be much smaller than the dimension of the simulated volume, V, i.e.,

$$l \ll \min(V_i) \tag{12}$$

where l is given by (5), and V_i are the lengths of the dimensions of a cuboid V, i.e., $|V| = V_1 \times V_2 \times V_3$, for 3D.

Since D is a macroscopic parameter, we adjust time step, Δt, to satisfy (12). As a result, lower l implies lower Δt.

Assume that there is a narrow hole, $\mathcal{H}$, on ∂V, i.e., the surface enclosing V. By definition of NEP as discussed in Sec. 2, H, the largest dimension of $\mathcal{H}$, needs to satisfy

$$H \ll \min(V_i). \tag{13}$$

When the Brownian particle, $\mathcal{P}$, is in the vicinity of the hole, $\mathcal{H}$ becomes part of the geometry of the volume. Accordingly, we need to satisfy (12) for h, as well as for $\min(V_i)$, turning (12) into,

$$l \ll h \ll \min(V_i), \tag{14}$$

where h is the smallest dimension of $\mathcal{H}$.

As we present in Sec. 3, the discretization errors are proportional to l. As (7) and (8) shows, the errors accumulate slowly during the simulation. However, if $\mathcal{P}$ exits $\mathcal{H}$ due to the error, it ends the simulation prematurely. Thus using a small l is especially important for NEP simulations.

Satisfying (14) requires further adjustment on Δt, which might cause the simulations to run prohibitively long. To mitigate this problem, we first assert that $\mathcal{H}$ is not part of the geometry for the entire V. The effect of $\mathcal{H}$ on the simulation depends entirely on the distance between $\mathcal{P}$ and $\mathcal{H}$, $d(\mathcal{H}, \mathcal{P})$. Accordingly, we can relax the first part of (14), i.e.,

$$l \ll d(\mathcal{H}, \mathcal{P}), \tag{15}$$

as long as $h \ll d(\mathcal{H}, \mathcal{P})$. This allows us to compartmentalize the volume depending on the distance from $\mathcal{H}$.

Algorithm 1 presents our approach to NEP simulations. To walk through the algorithm 1, we elaborate on some key points. Line 11 initializes a flag variable, f, which is used to break the main loop when the particle escapes. Line 14 chooses a space step from L, depending on the output of the $dist(.)$ module, which is discussed in

Table 1: Definition of symbols used in Sec. 4.

Variable	Symbol
Volume	V
Dimensions of V	V_i
Hole	$\mathcal{H}$
Particle	$\mathcal{P}$
Largest dimension of $\mathcal{H}$	H
Smallest dimension of $\mathcal{H}$	h
Location of $\mathcal{P}$ at iteration j	$\mathcal{P}(j) = [x(j), y(j), z(j)]^T$
Intersection of trajectory of $\mathcal{P}$ with the plane of $\mathcal{H}$	$P_I = [P_{Ix}, P_{Iy}, P_{Iz}]^T$
Centre of $\mathcal{H}$	$H_c = [H_{cx}, H_{cy}, H_{cz}]^T$
Expected simulation time	τ
Number of iterations	n
i^{th} compartment of V	V^i
Simulation space step in V^i	l_i
Space step set	L
Expected time spent in V^i	τ_i
Closest distance between $\mathcal{P}$ and $\mathcal{H}$	$d(\mathcal{P}, \mathcal{H})$

Algorithm 1 Brownian Motion for NEP in dD

1: **Inputs:**
2: T: maximum simulation time,
3: L: set of space steps,
4: D: diffusion constant,
5: V: simulation volume,
6: $\mathcal{H}$: hole, or absorbing boundary,
7: $\vec{R}$: initial point
8: **Initialise:**
9: $\mathcal{P}(0) = \vec{R}$
10: $t = 0$
11: $f = 1$
12: $i = 1$
13: **while** $[(t < T)$ **and** $f]$ **do**
14: $d = dist(\mathcal{P}(i-1), \mathcal{H})$
15: $l = L[d]$
16: $\Delta t = l^2/2D$
17: $\vec{a} = [\text{Uniform}([0,1)) - 0.5]^d$
18: $\vec{a} \leftarrow l \times \dfrac{\vec{a}}{\|\vec{a}\|_2}$
19: $\mathcal{P}(i) = \mathcal{P}(i-1) + \vec{a}$
20: $t \leftarrow t + \Delta t$
21: **if** *out_volume*
22: **if** $escape(\mathcal{P}(i-1), \mathcal{P}(i))$
23: $f \leftarrow 0$
24: **return** t
25: **else**
26: $\mathcal{P}(i)) \leftarrow R\{\mathcal{P}(i)\}$
27: $i \leftarrow i + 1$
28: **return** 0

Sec. 4.4. Line 15-17 creates a new vector with length l to illustrate the motion of $\mathcal{P}$ in the next time step. Line 21 calls the $out\_volume(.)$ function, which returns 1 if $\mathcal{P}$ is out of V. If it is out of V, $escape(.)$ module we present in Sec. 4.2 determines whether the particle escaped through $\mathcal{H}$, and we break the main loop by changing f to 0 in line 23. Otherwise, $\mathcal{P}$ is reflected back to V with $R\{\}$. If the particle is still in V when t exceeds the predetermined simulation time, Algorithm 1 returns 0.

Algorithm 1 has two challenges. Firstly, constantly measuring $dist(\mathcal{H}, \mathcal{P})$ increases the computational burden, even if it does not increase the computational complexity. Secondly, calculating $dist(\mathcal{H}, \mathcal{P})$ is not necessarily simple. A successful estimation of $dist(\mathcal{H}, \mathcal{P})$ is the central point in our algorithm.

4.2 Escape Module

The escape module calculates if the particle leaves the volume in the last iteration. Thus, it uses the current and last position of $\mathcal{P}$. Shortly, it uses linear interpolation to check if the line $\mathcal{P}$ traversed in the last time step coincides with $\mathcal{H}$. Note that we only need to check if $\mathcal{P}$ leaves through $\mathcal{H}$ if $\mathcal{P}$ is outside V. Checking whether $\mathcal{P}$ is in V is easier as it only requires $2d$ comparisons. To find whether $\mathcal{P}$ is in V, we need a simple comparison routine which returns 1 only if any of the coordinates of $\mathcal{P}(j)$ is outside V, i.e.,

$$in\_volume(\mathcal{P}(j)) = \sum_{i=1}^{d} \neg\, (0 \le P_i(j) \le V_i)\,, \tag{16}$$

where $\neg$ stands for the Boolean NOT operation.

As with any BM simulation, we assume that $\mathcal{P}$ moves in a straight line between these points. Thus, $escape(.)$ checks if the straight line intersects $\mathcal{H}$, i.e.,

$$escape(\mathcal{P}(j-1), \mathcal{P}(j)) = \begin{cases} 1, & \text{if } [\mathcal{P}(j-1), \mathcal{P}(j)] \cap \mathcal{H} \neq \emptyset \\ 0, & o.w \end{cases}. \tag{17}$$

(17) has two stages. In the first stage, we determine the point, P_I, the intersection of the trajectory of $\mathcal{P}$ and the surface that $\mathcal{H}$ lies. In the second stage, we check if P_I is on $\mathcal{H}$.

In 2D, assuming $\mathcal{H}$ lies on $x = V_1$ line, where V_1 is the dimension of V in the direction of $\mathcal{H}$, one way to find P_I is

$$P_{Iy} = \frac{y(j) - y(j-1)}{x(j) - x(j-1)} \Big(V_1 - x(j-1) \Big) + y(j-1), \tag{18}$$

where $\mathcal{P}(j) = [x(j), y(j)]^T$ and $P_I = [V_1, P_{Iy}]$. Similarly, in 3D, we also need P_{Iz}. We find P_{Iz} by replacing $y(j)$ and $y(j-1)$ of (18) with $z(j)$ and $z(j-1)$, assuming $\mathcal{H}$ is located at $x = V_1$ plane.

Determining whether $P_I \in \mathcal{H}$ depends on the exact geometry of $\mathcal{H}$. In 2D, for $\mathcal{H}$ centered at $H_c = [V_1, H_{cy}]$ and of length H, $P_I \in \mathcal{H}$ if

$$|P_{Iy} - H_{cy}| < \frac{H}{2}. \tag{19}$$

In 3D, for $\mathcal{H}$ centered at $H_c = [V_1, H_{cy}, H_{cz}]$ and of radius $H/2$ for circular hole, $P_I \in \mathcal{H}$ if

$$(P_{Iy} - H_{cy})^2 + (P_{Iz} - H_{cz})^2 \le \left(\frac{H}{2}\right)^2. \tag{20}$$

Similarly, for an elliptical hole of semimajor axis $H/2$ and semiminor axis $h/2$, $P_I \in \mathcal{H}$ if

$$\left(\frac{P_{Iy} - H_{cy}}{H/2}\right)^2 + \left(\frac{P_{Iz} - H_{cz}}{h/2}\right)^2 \le 1, \tag{21}$$

and for a rectangular hole of size $H \times h$, $P_I \in \mathcal{H}$ if

$$\left(|P_{Iy} - H_{cy}| < \frac{H}{2}\right) \,\&\, \left(|P_{Iz} - H_{cz}| < \frac{h}{2}\right). \tag{22}$$

For (21) and (22), we assume that $\mathcal{H}$ is aligned with the coordinate axes, and the longer dimension falls on the y axis. If $\mathcal{H}$ is not aligned, we can rotate the coordinate system to ensure that $\mathcal{H}$ is aligned with the coordinate axes.

Note that $escape(.)$ routine calls (18) and one of (19, 20, 21, 22). Therefore, calling $escape(.)$ at each iteration increases the computational burden, even though it does not increase the computational complexity. Thus, we do not call $escape(.)$ at each iteration, but only when there is a possibility that $\mathcal{P}$ may escape. We detail our approach in Sec. 4.3.

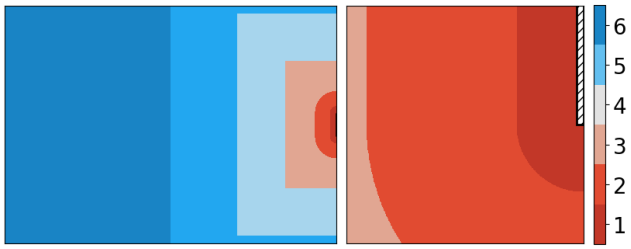

Figure 2: Possible compartmentalization of a 2D volume into six compartments, with a zoomed view in the vicinity of the hole. We depict the hole located at the boundary as the protruding shaded white rectangle for visualization purposes.

4.3 Compartmentalization

As described in Sec. 4.1, we choose a space step, l depending on the distance between $\mathcal{P}$ and $\mathcal{H}$. Rather than finding the exact distance, we *compartmentalize* V, and find which compartment $\mathcal{P}$ lies at a given time. Then, depending on the compartment, we assign an appropriate l value for the next iteration. A possible compartmentalization of a 2D volume is presented in Fig. 2.

We use two principles in compartmentalization:

- Transition between compartments is possible only for adjacent compartments, and escape is only possible from the innermost compartment.
- Compartmentalization is *smooth*, i.e., changes in l should not be abrupt.

The first principle ensures that $\mathcal{P}$ might escape when we model its movement with an l that satisfies (14). The second principle is to increase the computational performance.

To this end, we divide V into $V = V^1 + V^2 + \cdots + V^k$ compartments with space steps $L = [l_1, l_2, \ldots, l_k]$, with

$$l_1 < l_2 < \cdots < l_k \tag{23}$$

and l_1 satisfying (14).

Due to the first principle, for $i < k$, the width of V^i should be at least l_{i+1}. For a 2D V, assuming V^i are concentric rectangles, we find V^i using

$$V^i = (H + 2\alpha_{i+1}) \times \alpha_{i+1} - \sum_{j=1}^{i} V^{j-1} \tag{24}$$

$$= (H + 2\alpha_{i+1}) \times \alpha_{i+1} - (H + 2\alpha_i) \times \alpha_1 \tag{25}$$

$$= Hl_{i+1} + 4\alpha_i l_{i+1} + 2\alpha_i^2 \tag{26}$$

where $V^0 = 0$ and

$$\alpha_i = \begin{cases} \sum_{j=2}^{i} l_j & \text{if } i > 1, \\ 0 & \text{if } i = 1. \end{cases} \tag{27}$$

We also depict l_i and α_i in Fig. 3.

We know that (24) is valid as long as $H + 2\alpha_i$ is smaller than the length of V in that direction. As we see from Fig. 2, for the 5^{th} compartment, this assumption is not valid. For these compartments, V^i is approximately

$$V^i \approx V \frac{l_{i+1}}{\alpha_{i+1}}. \tag{28}$$

The outermost compartment, 6^{th} in Fig. 2, is the remaining volume:

$$V^k = V - \sum_{i=1}^{k-1} V_i. \tag{29}$$

For a 3D V, V^i are concentric cuboids rather than rectangles. Thus, we find (24) for 3D as

$$V^i = [(H + 2\alpha_i) \times (h + 2\alpha_i) \times \alpha_i] - \sum_{j=1}^{i} V^{j-1}. \tag{30}$$

Having too many compartments might introduce too many overheads and slow the simulation. Since l_i in each V^i must satisfy (23), following (24), $|V^i|$s become too large compared to l_i, reducing the computational performance.

Similarly, having too few of them is detrimental to the simulation efforts. We illustrate this using a two-element set, $L = [l_1, l_2]$, where l_1 satisfies (14). Due to the first principle we outline in this section, the compartment which we use l_1 as the space step has dimensions $(H + 2l_2) \times l_2$ thus number of steps within this compartment is proportional to $\left(\frac{l_2}{l_1}\right)^2$. Consequently, the simulation time is decoupled from the size of V. However, we either choose a small l_2 for V^2, or l_1 is designated to a large compartment, which deteriorates the simulation performance.

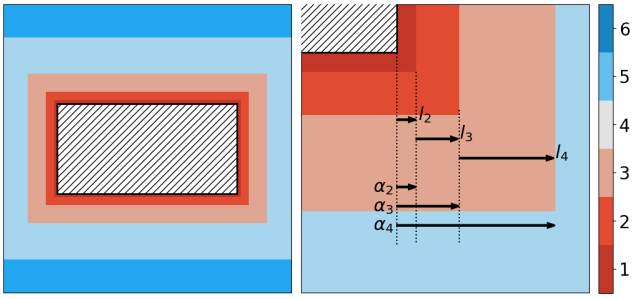

Figure 3: Possible compartmentalization of a 3D volume and illustration of l_i and α_i. Here, we only show the plane on which the hole is located. The hole is depicted as a shaded white rectangle.

4.4 Dist Module

In the previous section, we introduce the compartmentalization process. In this section, we describe the $dist(.)$ module, which estimates the distance between $\mathcal{P}$ and $\mathcal{H}$ and assigns $\mathcal{P}$ into a compartment.

$dist(.)$ routine, if very intricately designed, becomes *computationally heavy*. Since it is called at each iteration, we prefer $dist(.)$ to estimate rather than determine the exact distance. This is especially

important if $\mathcal{H}$ is not circular or rectangular. For example, solving the distance from a point to an ellipse requires solving a 4^{th} degree polynomial. To avoid making $dist(.)$ heavier than it needs to be, we avoid obtaining exact distances.

To reduce the computational burden, we call $dist(.)$ module *partially*. Once we know $\mathcal{P}$ is in V^i, due to the first principle we state in Sec. 4.3, $\mathcal{P}$ is in one of V^i, V^{i-1}, V^{i+1}. Hence, we can check if $\mathcal{P}$ is in V^i and V^{i+1}. We do not use V^{i-1}, per our discussion in Sec. 4.3, i.e., the closer $\mathcal{P}$ i to $\mathcal{H}$, the heavier $dist(.)$ needs to be. Regardless, we know that $\mathcal{P}$ is in V^{i-1} if it is not in V^i or V^{i+1}.

We separate the $dist(.)$ module into three subroutines: *fast_check*, *regular_check* and *slow_check*. *fast_check* is called when $\mathcal{P}$ is far from $\mathcal{H}$. When $\mathcal{P}$ approaches $\mathcal{H}$, we call a heavier and more precise *regular_check*. Finally, we call *slow_check* when $\mathcal{P}$ is in the vicinity of $\mathcal{H}$. We describe these subroutines in order.

4.4.1 fast_check subroutine. In dD space, $\mathcal{H}$ is $d-1$D, so we can choose our coordinate systems such that there is one dimension, orthogonal to $\mathcal{H}$. Choosing the orthogonal dimension as x, as we do in Sec. 4.2 for $dist(\mathcal{H},\mathcal{P}) \gg H$,

$$dist(\mathcal{H},\mathcal{P}) \approx V_1 - x(i). \tag{31}$$

(31) is the fastest way to estimate the position of $\mathcal{P}$; therefore, we name this subroutine as *fast_check*. However, since it does not use any information on other dimensions, it is only applicable to the outermost compartments.

4.4.2 regular_check subroutine. As $\mathcal{P}$ approaches $\mathcal{H}$, more intricate details of $\mathcal{H}$ need to be taken into account. To this end, we set rectangles for a 2D V as illustrated in Fig. 2 and cuboids for a 3D V. We describe this process in detail in Sec. 4.3. $\mathcal{H}$ is necessarily a line in 2D, and we do not need to make any adjustments. However, in 3D, to obtain cuboid compartments, we extend non-rectangular $\mathcal{H}$ to a $H \times h$ rectangle, i.e., the smallest rectangle, that $\mathcal{H}$ can fit into. Then, for $\mathcal{P}$ to be in V^i, the following should be satisfied.

$$V_1 - \alpha_{i+1} > x > V_1 - \alpha_i, \tag{32a}$$
$$H_{cy} + H/2 + \alpha_i > y > H_{cy} - H/2 - \alpha_i, \tag{32b}$$
$$H_{cz} + h/2 + \alpha_i > z > H_{cz} - h/2 - \alpha_i, \tag{32c}$$

where (32c) is used only for 3D.

Note that this subroutine involves six comparisons. After compartments are finalized, the comparison limits do not change during the simulation. Therefore, it is still very robust.

4.4.3 slow_check subroutine. We design the *slow_check* routine to make small adjustments for innermost compartments. The corners of the innermost compartments are larger than necessary. Thus, we can round the corners using quarter circles in 2D or one-eighth spheres in 3D. We can make these adjustments only if the projection of $\mathcal{P}$ on the line (or plane in 3D) that $\mathcal{H}$ is located is outside $\mathcal{H}$. Since we are looking for projection of $\mathcal{P}$, x coordinate does not come into play.

slow_check subroutine is responsible for the round corners of V^1 and V^2 in Fig. 2. To do so, *slow_check* anticipates which side of $\mathcal{H}$ $\mathcal{P}$ lies in and calculates $d^2(\mathcal{H},\mathcal{P})$ and compares it with α_{i+1}^2. Note that we use $d^2(.)$ rather than $d(.)$ to avoid using the computationally heavy $sqrt(.)$ function.

slow_check consumes higher resources than other subroutines, and depending on the choice of L, it might not be necessary. Therefore, the depth to which *slow_check* is used, if it is used at all, should be application-specific.

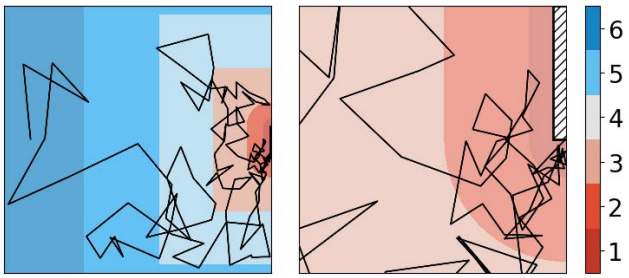

Figure 4: 2D demonstration of NEP using the compartmentalization in Sec. 4.3.

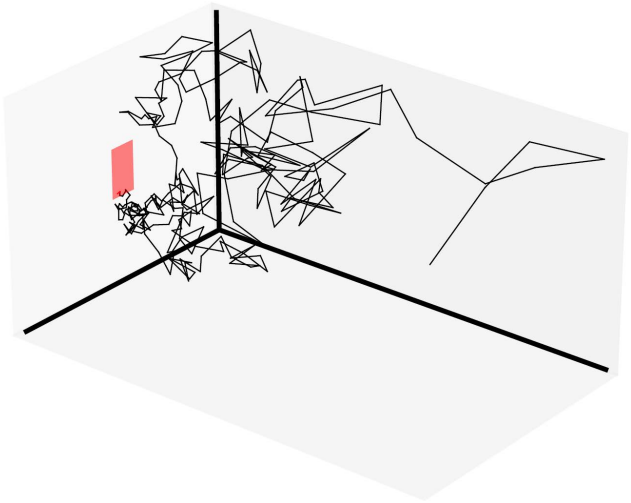

Figure 5: 3D demonstration of NEP using the compartmentalization in Sec. 4.3 extended to 3D.

5 SIMULATION RESULTS

In the previous section, we provide the details of our simulation framework. In this section, we demonstrate our framework and compare its results with the literature values presented in Sec. 2.

Fig. 4 and 5 are our demonstrations in 2D and 3D respectively. In both figures we see that as $\mathcal{P}$ approaches $\mathcal{H}$, we use a smaller space step for increased accuracy.

We illustrate the performance of our simulation framework in 2D with $V = q \times q$, $H = q/500$, $D = 0.2$ and $L = \{(\frac{q}{500})^i | 1 \leq i \leq 6\}$ in Fig. 6. The start point is chosen as a random point with $x, y \in [q/4, 3q/4]$ and $\mathcal{H}$ is located at the corner. We obtain the theoretical result is using (2).

Clear from Fig. 6 that our framework offers similar results to the theoretical values. The discrepancy between simulation results and theoretical results is higher in smaller volumes. We believe

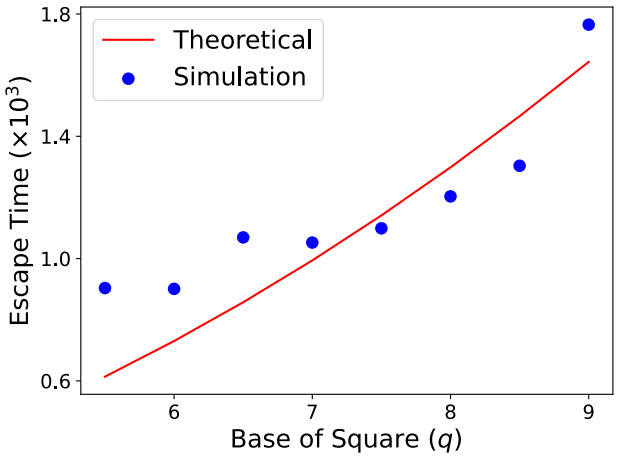

Figure 6: Simulation vs. theoretical results for NEP for a 2D rectangular volume.

this is due to the corners constituting a larger area compared to the total area for smaller q. Corners *trap* $\mathcal{P}$ and thus increasing the simulated escape time. To improve our algorithm, we need to reduce the step size around the other features of the area. Accordingly, we need to use distance to edges and corners of the area with the proximity to the $\mathcal{H}$ to determine the space step. We believe such an improvement will reduce the probability of entrapment around the corners, increasing the correlation between the theoretical and simulation results.

6 CONCLUSION

In this work, we present a novel simulation framework for NEP specifically designed for rectangular and cuboid volumes. Particle motion at respective scales follows BM. However, BM simulations introduce additional challenges to the solution of NEP as the simulation step size is required to be much smaller than the hole size, resulting in enormous computation time. The proposed adaptive simulation framework, which dynamically selects simulation step sizes, is much faster than existing approaches based on constant step size. As a future improvement, we plan to apply smaller space steps around the corners to increase the accuracy as the corners appear to increase the escape time. We believe our work improves the understanding of SC dynamics by quantifying the escaping neurotransmitters into the apposition zones.

ACKNOWLEDGMENTS

This work was supported in part by the AXA Research Fund (AXA Chair for Internet of Everything at Koç University) and Huawei Graduate Research Scholarship.

REFERENCES

[1] Nina Arnth-Jensen, Denis Jabaudon, and Massimo Scanziani. 2002. Cooperation between independent hippocampal synapses is controlled by glutamate uptake. *Nature Neuroscience* 5, 4 (2002), 325–331. https://doi.org/10.1038/nn825

[2] Soren Asmussen, Peter Glynn, and Jim Pitman. 1995. Discretization error in simulation of one-dimensional reflecting Brownian motion. *The Annals of Applied Probability* 5, 4 (1995), 875–896.

[3] Richard J. Bridges and C. Sean Esslinger. 2005. The excitatory amino acid transporters: Pharmacological insights on substrate and inhibitor specificity of the EAAT subtypes. *Pharmacology & Therapeutics* 107, 3 (2005), 271–285. https://doi.org/10.1016/j.pharmthera.2005.01.002

[4] Poria Hasanpor Divshali, Matti Laukkanen, R Bhandia, AA VanderMeer, E Widl, C Steinbrink, A Kulmala, and K Mäki. 2019. Smart grid co-simulation by developing an FMI-compliant interface for PSCAD. In *25th International Conference on Electricity Distribution (CIRED 2019)*. 849.

[5] Xiao-xia Dong, Yan Wang, and Zheng-hong Qin. 2009. Molecular mechanisms of excitotoxicity and their relevance to pathogenesis of neurodegenerative diseases. *Acta Pharmacologica Sinica* 30, 4 (2009), 379–387.

[6] Alireza Ghasempour. 2015. Using a genetic-based algorithm to solve the scheduling optimization problem for long-range molecular communications in nanonetworks. In *2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE, 1825–1829.

[7] Christof Grewer and Thomas Rauen. 2005. Electrogenic glutamate transporters in the CNS: molecular mechanism, pre-steady-state kinetics, and their impact on synaptic signaling. *The Journal of membrane biology* 203, 1 (2005), 1–20.

[8] David Holcman and Zeev Schuss. 2014. The narrow escape problem. *siam REVIEW* 56, 2 (2014), 213–257.

[9] Aoife Hughes, Richard J Morris, and Melissa Tomkins. 2021. PyEscape: a narrow escape problem simulator package for Python. *Journal of Open Source Software* 5, 47 (2021), 2072.

[10] Caglar Koca, Meltem Civas, and Ozgur B Akan. 2021. Evolutionary Game Theoretic Resource Allocation Simulation for Molecular Communications. *ITU Journal on Future and Evolving Technologies* 2, 3 (2021).

[11] Caglar Koca, Meltem Civas, Selin Merve Sahin, Onder Ergonul, and Ozgur B Akan. 2021. Molecular Communication Theoretical Modeling and Analysis of SARS-CoV2 Transmission in Human Respiratory System. *IEEE Transactions on Molecular, Biological and Multi-Scale Communications* 7, 3 (2021), 153–164.

[12] Murat Kuscu and Ozgur B Akan. 2016. On the physical design of molecular communication receiver based on nanoscale biosensors. *IEEE Sensors Journal* 16, 8 (2016), 2228–2243.

[13] Murat Kuscu and Ozgur B Akan. 2018. Maximum likelihood detection with ligand receptors for diffusion-based molecular communications in Internet of bio-nano things. *IEEE transactions on nanobioscience* 17, 1 (2018), 44–54.

[14] Murat Kuscu and Ozgur B Akan. 2019. Channel sensing in molecular communications with single type of ligand receptors. *IEEE Transactions on Communications* 67, 10 (2019), 6868–6884.

[15] AE Lindsay, T Kolokolnikov, and JC Tzou. 2015. Narrow escape problem with a mixed trap and the effect of orientation. *Physical Review E* 91, 3 (2015), 032111.

[16] Sebastian Lotter, Arman Ahmadzadeh, and Robert Schober. 2020. Synaptic channel modeling for DMC: Neurotransmitter uptake and spillover in the tripartite synapse. *IEEE Transactions on Communications* 69, 3 (2020), 1462–1479.

[17] Silong Lu, Fred J Molz, and Hui Hai Liu. 2003. An efficient, three-dimensional, anisotropic, fractional Brownian motion and truncated fractional Levy motion simulation algorithm based on successive random additions. *Computers & geosciences* 29, 1 (2003), 15–25.

[18] Maiken Nedergaard and Alexei Verkhratsky. 2012. Artifact versus reality—how astrocytes contribute to synaptic events. *Glia* 60, 7 (2012), 1013–1023.

[19] Sinead M O'Donovan, Courtney R Sullivan, and Robert E McCullumsmith. 2017. The role of glutamate transporters in the pathophysiology of neuropsychiatric disorders. *npj Schizophrenia* 3, 1 (2017), 1–14.

[20] Hamideh Ramezani, Caglar Koca, and Ozgur B Akan. 2017. Rate region analysis of multi-terminal neuronal nanoscale molecular communication channel. In *2017 IEEE 17th International Conference on Nanotechnology (IEEE-NANO)*. IEEE, 59–64.

[21] Zeev Schuss. 2012. The narrow escape problem—a short review of recent results. *Journal of Scientific Computing* 53, 1 (2012), 194–210.

[22] Zeev Schuss, Amit Singer, and David Holcman. 2007. The narrow escape problem for diffusion in cellular microdomains. *Proceedings of the National Academy of Sciences* 104, 41 (2007), 16098–16103.

[23] Amit Singer, Zeev Schuss, and David Holcman. 2006. Narrow escape, Part II: The circular disk. *Journal of statistical physics* 122, 3 (2006), 465–489.

[24] A Singer, Z Schuss, and David Holcman. 2006. Narrow escape, Part III: Non-smooth domains and Riemann surfaces. *Journal of statistical physics* 122, 3 (2006), 491–509.

[25] Amit Singer, Zeev Schuss, David Holcman, and Robert S Eisenberg. 2006. Narrow escape, part I. *Journal of Statistical Physics* 122, 3 (2006), 437–463.

[26] Mladen Veletić and Ilangko Balasingham. 2018. Capacity estimation in MIMO synaptic channels. In *Proceedings of the 5th ACM International Conference on Nanoscale Computing and Communication*. 1–6.