Reed College

Senior Thesis

Split Reactive Brownian Dynamics

Chiao-Yu Yang

supervised by

Prof. Aleksandar Donev and Prof. David Perkinson

April 28, 2016

Dedicated to Huai-Yuan Zhang

1. Motivation

In this thesis, we aim at studying the evolution of a fluid system of reactive particles that undergo Brownian motion. That is, given a fluid system in homogeneous medium, particles undergo Brownian motion with constant diffusion rate. Moreover, the particles in our system are candidates for different types of reactions, where reactions take place with certain probability per unit time whenever all the reaction candidates are within a separation less than or equal to the reactive radius. Traditionally, differential equations are frequently used for the simulation of fluids. Notable examples include the Navier-Stokes equation, the Euler equation, the Boltzmann equation and so on. However, there are some limitations in their application. For example, the Navier-Stokes equation assumes continuity of the fluid and that the fluid in a whole moves with zero relativistic velocities. Such assumptions do not hold in all situations. In particular, at the molecular level, a reactive system's macroscopic behavior can be affected by factors such as thermally induced fluctuations. For example, combustion in a low-temperature system can lead the system to a state where the fluctuation effect is much stronger and causes the macroscopic description given by the differential equation to deviate from reality [17]. Another example would be a chemical system with catalitic reactions. In such a system, if the reactants exist in a small region, then the spatial fluctuation of that region grows as the reaction happens. This needs to be accounted for in computation.

In our case, the formulation based on deterministic differential equations is inaccurate since it assumes well-mixedness of the whole system, which is not necessarily true. Hence, for an accurate result, we need to use more accurate descriptions. In the following sections, we will investigate existent methodology and then discuss our new method.

2. Background

In this section, we lay down the theoretical backgrounds that constitute the basis of our method of efficiently performing a reactive system's simulation.

2.1. **Particle Methods.** Traditional macroscopic methods are inadequate for the description of certain situations, and we need to use mesoscopic or microscopic methods to describe the system's behavior. A much more accurate simulation can be obtained if one uses a particle-based model instead of PDEs. In the following, we will first review two particle-based methods.

Molecular dynamics (MD), developed in the early 60s, has gained popularity in research areas such as physics, material science, biology, and chemistry. It views the system as a collection of interacting particles and reduces the original problem to an N-body problem. Usually the velocities and displacement of particles after events such as collision are calculated by solving Newton's equations of motion. However, for the study of a reactive system's dynamics, MD is very limited due to its high computational cost. Actually, a deterministic algorithm is too computationally expensive to be of practical use in many cases. We need to turn to stochastic algorithms to achieve higher efficiency.

An alternative method used for rarefied gas modeling is the direct simulation Monte Carlo (DSMC) method. DSMC also views the system as a collection of interacting particles. It divides the system into small cells and only particles in the same cell are allowed to interact. Instead of calculating the interaction in a deterministic way, DSMC uses kinetic theory to predict certain parameters such as the collision rate, and then processes collisions stochastically. Unlike MD, DSMC is not correct at the molecular scale but accurate in a scale smaller than mean free path. Hence, even at the microscopic level, DSMC is accurate.

2.2. **Time and Event Driven Algorithms.** In a particle model, the evolution can be driven by time or events. We illustrate both of them in a model where particles are assumed to be hard spheres and travel in straight lines between collisions. The only event here is the collision of particles. In a time-driven simulation, the system evolves as time progresses. Given the discrete nature of computer time, we cannot use a continuously changing model. Hence, we need to discretize time into small intervals dt. Starting from the initial configuration, we let each particle move by a time dt and then we check if any two of them are overlapping. If not, then we go on to move the particles in the next timestep. If there are some particles overlapping with each other, then we go back to the moment of collision and calculate the after-collision velocities for each particle and then their positions at the end of that interval. For the algorithm to be accurate, we need to use a sufficiently small dt. Otherwise, we can miss collisions since particles are only observed at the end of each timestep and a collision that happens in the middle of a timestep can go unnoticed. However, using a small timestep would significantly increase the computational cost of the algorithm, and this makes time-driven simulation a less useful technique.

On the other hand, we have the event-driven simulation. In the same hard-sphere collision example, instead of evolving the system by time, we evolve it by events. Given an initial configuration, we build a queue consisting of events and times of occurences. After building the event queue, we choose the event with highest priority, that is, the first one that happens, and we process this event. After that, we need to make changes to the system. For example, the colliding particles will have new trajectories and thus future events in the queue that involve these particles should be invalidated while some new events should be added. Then, we repeat the procedure above until the desired stopping time is achieved. Compared to time-driven simulation, event-driven simulation is more efficient and accurate in many cases.

2.3. **Reaction-Diffusion Models.** There are several models used for stochastic reaction-diffusion models. Here, we introduced the Smoluchowski model and Doi model. We will start with the Smoluchowski model, which was proposed by Smoluchowski in 1917. We illustrate the idea by the bimolecular coagulation reaction $A + B \to \emptyset$. Here, the reaction generates no product, which does not mean some particles disappear during reaction. Instead, there can be various interpretations. For example, the resulting product can be particles that do not interfere with any particle of interest in our system and we just ignore them in the simulation. The reaction is assumed to take place instantaneously once two molecules are in a separation r, which we call the reactive radius. As an example, we start with much more B molecules than A molecules and add A molecules to replace those annihilated during the reaction. Choose an A molecule as the central particle, around which is a group of particles undergoing Brownian motion. The central A particle acts like a sink: once a B molecule diffuses toward it and is exactly r away from it, the B molecule would annihilate. With these conditions, we can set up a differential equation, with the dependent variable being the number density of B and the independent variable being the distance away from the central A molecule. However, there are two serious limitations in this model: (1) the static reaction area is different from physical situation where the reaction region should be dynamic; (2) the particles surrounding the central particle do not interact with each other. These two limitations make the model unphysical.

After its first publication, the Smoluchowski model has been modified and improved. In an attempt to construct a more accurate model, M. Doi proposed the Doi model in [6]. In this model, binary reactions take place stochastically when two reagents are separated by less than or equal to their reactive radius. The probability of the reaction taking place is fixed per unit of time.

There are deterministic algorithms for the above models but such approaches lack accuracy, and a stochastic description is necessary, see [9]. We now further develop the basic background of stochastic simulation, and then illustrating Gillespie's stochastic simulation algorithms (SSA) and the reaction-diffusion master equation (RDME) model.

We start with the single annihilation reaction, i.e.,

$$A \to \emptyset. \tag{1}$$

Let $A(t)$ denote the number of A molecules at time t. With the basic assumption that a reaction takes place with a fixed reaction rate k per unit time, we derive that at time t the probability that a reaction would happen in the next timestep dt

$$f(t)dt = A(t)kdt. \tag{2}$$

Then, let $g(A(t), s)$ be the probability that there is no reaction from t to s. Then, by time invariance of the system, we have

$$g(A(t), s + dt) = g(A(t), s)(1 - f(t + s)dt). \tag{3}$$

Since no reaction happens until s, we have

$$f(t + s)dt = A(t)kdt. \tag{4}$$

Hence, by rearranging the equation and passing dt to 0, we get

$$\frac{\partial g(A(t), s)}{\partial t} = -A(t)kg(A(t), s) \tag{5}$$

which is a simple ODE for g. The solution for this ODE with initial condition $g(A(t), 0) = 1$ we can get the solution to be

$$g(A(t), s) = e^{-A(t)ks}. \tag{6}$$

Again, by time invariance of the system, we can derive the probability that the first reaction since time t happens at $t + s$ is

$$h(s) = g(A(t), s)A(t)kds. \tag{7}$$

This is a probability density function, and the reader can easily verify that its integral over $[0, \infty)$ is 1. Hence, if we assume that reactions happen uniformly with respect to the probability distribution function we just calculated, we can determine the next reaction time τ. In practice, we need to generate a uniform psuedo-random number $r \in (0, 1)$ and then use r to calculate τ as follows. Suppose τ is the cumulative distribution function $F(r)$. Then we need to have the following condition

$$\int_{F(r_1)}^{F(r_2)} h(s)ds = r_2 - r_1. \tag{8}$$

This condition implies that $hds = dF^{-1}$, hence

$$F^{-1}(s) = e^{-A(t)s}. \tag{9}$$

Hence, we can derive F and therefore for any uniform random number $r \in (0,1)$ we can calculate

$$\tau = F(r) = \frac{-1}{A(t)k} \ln(r). \tag{10}$$

In this way we derive the stochastic first reaction time for the single annihilation system. However, we need modifications of the method to deal with a more complicated system consisting of multiple reactions, such as one consisting of both annihilation and birth reactions. To begin with, we define the *propensity function* for a reaction. Note that throughout we assume the number of reagents of a reaction to be less than or equal to two, since any reactions involving more than two reagents can be treated as several reactions, each of which contains at most two reagents. Then, for reactions with no reagents and reaction rate k_0, the propensity function at time t is defined to be $\alpha(t) = k_0$; for reactions with one reagent and reaction rate k_1, we define $\alpha(t) = A(t)k_1$, where $A(t)$ is the number of reagents; for reactions with two reagents and reaction rate k_2, we define $\alpha(t) = A_1(t)A_2(t)k_2$, where $A_1(t), A_2(t)$ are the number of two reagents. Essentially, the propensity function describes the likelihood of a certain reaction taking place at time t.

An important stochastic simulation algorithm (SSA) that we will use is the one developed by Gillespie in [10]. In a system of n reactions with certain initial conditions, Gillespie's SSA works as follows:

(1) Generate a uniform random number $r \in (0,1)$.

(2) Calculate the values of propensity functions for each reaction to be $\alpha_1, \cdots, \alpha_n$ and then define

$$\alpha_0 = \sum_{i=1}^{n} \alpha_i. \tag{11}$$

(3) Determine the next reaction time to be

$$\tau = \frac{-1}{\alpha_0} \ln(r). \tag{12}$$

(4) Generate a new uniform random number $r_1 \in (0,1)$.

(5) Determine which of the n reactions will take place next to be reaction i such that

$$\sum_{j=0}^{i-1} \frac{\alpha_j}{\alpha_0} \leq r_1 \leq \sum_{j=0}^{i} \frac{\alpha_j}{\alpha_0}. \tag{13}$$

(6) Change the number of molecules for each species involved in the reaction we chose accordingly. Then, go back to step one.

Now we are ready to introduce the concept of the *chemical master equation* (CME). For a general system of n reactions and m species starting at t_0 with $\boldsymbol{A_0} = (A_1(t_0), \cdots, A_m(t_0))$, suppose the probability for configuration $\boldsymbol{A}$ at time t is $P(\boldsymbol{A}, t)$, then the probability of $P(\boldsymbol{A}, t + dt)$ is given by

$$P(\boldsymbol{A}, t + dt) = P(\boldsymbol{A}, t)(1 - \alpha_0 dt) + \sum_{i=1}^{n} P(\boldsymbol{A} + \boldsymbol{v}_i, t)\alpha_i dt \tag{14}$$

where $\boldsymbol{v}_i$ is the vector of change for ith reaction. Underlying the previous equation is the assumption that there is at most one reaction within dt, which is justified as the differential of an exact differential is 0. Hence, by rearranging the equation and passing dt to 0, we get

$$\frac{\partial P(\boldsymbol{A},t)}{\partial t} = -\alpha_0 P(\boldsymbol{A},t) + \sum_{i=1}^{n} \alpha_i P(\boldsymbol{A} + v_i, t). \tag{15}$$

This coupled system of ODEs is the CME. In the general case, the dimension of the system can be infinite. Traditional numerical schemes are inappropriate for solving such equations. Even though in an extremely simple case like the single annihilation reaction we just encountered, the solution to the stochastic equations we just constructed can be equal to that of deterministic ODEs, in general the solutions are different. See [9] for details.

So far we have considered system evolution from a global perspective and an assumption underlying our discussion is well-mixedness of the system. However, real situations are more complicated. According to Einstein's theory, particles undergo Brownian motion since frequent collision between particles make the trajectories random walks instead straight lines. For particles of a certain species, we can determine its diffusion constant by the formula

$$D = \frac{RT}{6\pi\eta r N} \tag{16}$$

where R is the gas constant, T is the absolute temperature, η is the viscosity of the fluid, r is the radius of the particle, and N is the Avogadro's constant. With this diffusion constant, we can compute the position of a particle to be

$$\boldsymbol{x}(t + \Delta t) = \boldsymbol{x}(t) + \sqrt{2D\Delta t}\ \boldsymbol{r}_n \tag{17}$$

where $\boldsymbol{r}_n$ is an n-dimensional uniform number with each component in the range $(0, 1)$. We also need to consider the existence of different boundary conditions. We suppose that the system is located in $[0, L_1] \times \cdots \times [0, L_d]$. There are three boundary conditions that we consider:

(1) Periodic: if in some dimension a particle diffuses into a position $x_i < 0$ or $x_i > L_i$, then we add a multiple, possibly negative, of L_i to x_i to make it in the range $[0, L_i]$. Essentially when a particle passes a boundary, it goes into the other side.

(2) Reflective: if in some dimension a particle diffuses into a position $x_i < 0$ or $x_i > L_i$, then we change its position to $-x_i$ or $2L_i - x_i$ respectively. In this case, the particle hits the boundary and is reflected back by the boundary.

(3) Reservoir: if in some dimension a particle diffuses into a position $x_i < 0$ or $x_i > L_i$, then the particle goes into the reservoir and is flagged as an irrelevant particle. Also, in the reservoir, if some particle diffuses into the system, it would be marked as a new particle in the system.

Now we proceed to illustrate the reaction-diffusion master equation (RDME) model. In the RDME model, the whole system is divided into a collection of cells of specific, the choice of which will be illustrated later. Each particle is viewed as a point undergoing Brownian motions. Reactions can take place only between particles in the same cell. A few versions of RDME are available. Some of them suffer from the problem that as cell size goes to zero, the first reaction time would approach infinity. See [14] for a detailed discussion. In particular, RDME no longer assumes the system to be globally well-mixed but to be locally well-mixed, i.e., partition positions within the same cell are assumed to be uniform.

Note that after including diffusion and allowing the system not to be globally well-mixed, we need to modify the chemical master to justify these changes. We assume the system to be located in $[0, L_1] \times \cdots \times [0, L_d]$ and partition it into cells of size $\boldsymbol{h} = h_1 \times \cdots \times h_d$. Also, we suppose there are n species and m reactions. It is straightforward to give each cell in the system an index j, where $j \in \{1, \cdots, \prod_{i=1}^{n} L_i/h_i\}$, or a multiindex $\boldsymbol{j} = (j_1, \cdots, j_n)$, where $j_i \in \{1, \cdots, L_i/h_i\}$. For simplicity we will use a single index and also in our implementation of IRDME in the third section, due to computational consideration, we use single index.

We can define a state vector $\boldsymbol{J} = (\boldsymbol{J}^1, \cdots, \boldsymbol{J}^n)$, where $\boldsymbol{J}^i$ is the vector of ordered single index of the cells of each particle with species i. Similar to the CME, if we denote the probability of having state vector $\boldsymbol{J}$ at time t with cell size vector $\boldsymbol{h}$ to be $P_{\boldsymbol{h}}(\boldsymbol{J}, t)$, then we can write the RDME as,

$$\frac{\partial P_{\boldsymbol{h}}(\boldsymbol{J}, t)}{\partial t} = (L + R) P_{\boldsymbol{h}}(\boldsymbol{J}, t) \tag{18}$$

where L is the diffusion operator, and R is the reaction operator. Now we derive each operator explicitly. Firstly, for the diffusion operator, we consider the system to be affected only by diffusion effect. A special form of the Fokker-Planck equation shows that for particles undergoing Brownian motion with diffusion constant D, if $f(\boldsymbol{x}, t)$ is probability density function of position $\boldsymbol{x}$ and time t, then we have

$$\frac{\partial f(\boldsymbol{x}, t)}{\partial t} = D \Delta f(\boldsymbol{x}, t) \tag{19}$$

where Δ is the Laplace operator. With this, we can just apply the discrete Laplace operator obtained by finite difference to derive the case where cell size is $\boldsymbol{h}$,

$$\frac{\partial f_{\boldsymbol{h}}(\boldsymbol{x}, t)}{\partial t} = \frac{1}{h^2} \sum_{i=1}^{d} f(\boldsymbol{x} + e_i) + f(\boldsymbol{x} - e_i) - 2f(\boldsymbol{x}). \tag{20}$$

Repeating this for each species would lead to the desired diffusion operator.

A description of converting the differential equation system we just described into particle model can be found in [13]. Even though the RDME has advantage over older models, it suffers from the problem of cell size. In particular, as shown in [16], in order to approximate the Doi or Smoluchowski model, the RDME model needs to have a cell size that is neither too large nor too small. Also, if the cell size approaches zero, then the time it takes for the next reaction to happen approaches infinity. Even with an appropriate choice of cell size, the idea of RDME would still introduce problems. When we implement the algorithm using a particle method, the grid structure of RDME would generate artifacts in the result. In a real situation, the particle distribution should be unrelated to the grid. However, once we implement RDME, there would be difference in particle distribution between the center and near the boundary of a cell.

Now, we are going to briefly discuss the direct simulation Monte Carlo (DSMC) algorithm. Traditional particle methods, notably molecular dynamics, simulate a system of particles in an N-body setting. Trajectories of particles are calculated by solving a system of equations of motion. The extremely high computational cost makes their applications limited to very small system size. DSMC is a stochastic alternative to MD in dilute gas. In the initial configuration, each particle has some position and velocity. The system is divided into n regular cells. At the beginning of each timestep, we update each particle's position to $\boldsymbol{r} + \boldsymbol{v}t$, where $\boldsymbol{r}$ is the particle's position, $\boldsymbol{v}$ is the velocity, and t is the length of the timestep. Then, we implement same-cell stochastic collisions. To do so, we first use kinetic theory to determine the number of collisions to process. Then, we randomly select pairs of particles in the same cell and randomly determine if they are going to collide or not. In particular, the trajectories of these particles need not overlap with each other since this is a stochastic rather than real collision. Also, not every pair we choose would collide. We first compute some maximum relative speed and then generate a uniform random number $r \in (0, 1)$. Then, if $|\boldsymbol{v}_r| < |\boldsymbol{v}_{\max}|r$ then we propose the collision and reset the velocity as if a collision happens to these two particles. Otherwise, we choose another pair of particles, which need not be different from the pair we just chose, and repeat the process above. We repeat doing so until processing the predetermined number of collisions. A detailed version of DSMC can be found in [1].

In the past decades, DSMC has achieved success in rarefied gas simulation. However, there are some intrinsic problems. Notably, the model is not Galilean invariant and not microscopically isotropic. The grid artifact introduces a difference in distribution of particles in the center of the cell and particles near the boundary of the cell. Even though random shifting before and after each timestep can mollify the problem, it introduces new problems regarding the treatment of boundary condition. In addition, DSMC is not isotropic and does not conserve angular momentum. Therefore, it can lead to very inaccurate results, for example, in rotating flows.

In [7], a scheme based on DSMC that is isotropic was proposed and named I-DSMC, where “I” means isotropic. Instead of allowing only same-cell collisions, in I-DSMC, two particles can still be allowed to collide if they are in neighboring cells, as long as the separation between them is not larger than the collision diameter.

In order to perform I-DSMC, in each timestep we create a random ordering of all the cells. Then we visit each cell in order, at which point we create a list of particles in the center cell and also a list of particles in the neighboring, and also the center, cells. Then, we process stochastic collisions with some rule, which will not be discussed here since it is not related to the topic of our interest. The reader may find the detailed description in [7].

3. S-BD-RME and S-RBD

There are three problems with the RDME approach,

(1) cell-size dependence;

(2) expensive computation cost, especially in well-mixed regime;

(3) unphysical diffusion process.

In [15], a convergent RDME is considered to fix the first issue above, where the reaction operator is modified so that reactions between different cells are taken into account. In [2], multinomial diffusion is proposed as an improvement for the second problem. For the third issue, particle tracking can be used, which generates a Brownian dynamics.

In this section we describe two particle algorithms, the Split Brownian Dynamics with Reaction Master Equation (S-BD-RME) and Split Reactive Brownian Dynamics (S-RBD). Here, S-BD-RME is based on one of the algorithm proposed in [5], with the difference that S-BD-RME uses strang splitting in diffusion process and a random shifting to reduce the error. Our novel method is the S-RBD algorithm, which is grid free.

3.1. **Model Description.** We want to investigate the dynamics of particles in a rectangular box in d-dimensions. Throughout this section, we will assume the hard-sphere model, that is, we view particles as hard spheres of fixed radii. Also, we assume the particle to diffuse in a homogeneous medium, where the medium has no impact on reaction rates. We tried to simulate this system as time evolves, where the changes come from diffusion and reaction of the particles.

For a system, we need to specify a set of parameters. We need two constants N_s and N_r, which represent the number of particles' species and the types of reactions respectively. For each species, we are also given its diffusion constant, radius, and number density. We allow reactions with 0, 1, or 2 reagents only. In reality, reactions involving three or more reagents do not take place immediately. Instead, they consist of sequences of reactions with at most 2 reagents. Hence, our assumption is physical. Also, we need to build a reaction network from input data. We classify reactions in the following way.

(1) Birth: reactions of the form $\emptyset \to A$ and $\emptyset \to A + B$, i.e., reactions with 0 reactants and 1 or 2 products, where the products are uniformly generated in the box.

(2) Annihilation: reactions of the form $A \to \emptyset$ and $A + B \to \emptyset$, i.e., reactions with 1 or 2 reactants and 0 products.

(3) Replacement: reactions of the form $A \to B$ or $A \to B + C$. If there is only one product, then the product takes the position of the reagent; otherwise, one of the product takes the position of the reagent and the other one is born uniformly in the sphere centered at the reagent's position with radius the sum of radius of two product particles.

(4) Catalytic birth: reactions of the form $A \to A + B$, i.e., reactions with a reagent A and the same product A, along with another product of any species. The catalyst particle will remain in its original position and the other product will be generated uniformly within reactive radius.

(5) Merge: reactions of the form $A + A \to B$ and $A + B \to C$. The product will be born randomly in one of the two reagents' positions.

(6) Catalytic annihilation (coagulation): reactions of the form $A + B \to A$ or $A + A \to A$, i.e., reactions with reagent A, another reagent of any species, and a single product of species A. If the two reagents have the same species then the product will be born randomly in one of the reagent's position; otherwise, the product will be born in the position of reagent with the same species.

(7) Catalysis: reactions of the form $A + B \to A + C$, i.e., reactions with catalyst A, along with a different reactant and a different product. The catalyst particle remains in its original position during reaction and the other product will take the position of the other reagents.

(8) Transformation: reactions of the form $A + A \to A + B$, $A + A \to B + C$, $A + B \to A + A$, and $A + B \to C + D$, i.e., reactions of two reagents and two products but which are not catalyst reactions. One product will take the position of one of the two reagents' positions and the other product will take the position of the other one.

The reactive system is confined to a rectangular box in d-dimensions. For reasons that will be clear soon, we divide the box into rectangular cells of equal sizes. Given the system size and a set of parameters that specify how many cells of both types we want in each dimension, we can calculate the sizes of the cells.

3.2. **Boundary Condition.** Among the different types of boundary conditions, we study the periodic boundary condition only. There are reasons to exclude the use of some other boundary conditions. For example, because we assume that particles undergo Brownian motions, a rigid wall that reflects particles when they hit the wall is not viable since the particles do not have a straight-line trajectory or a constant velocity. Also, reservoir boundary condition will only be used in the S-BD-RME simulation, because the S-RBD simulation requires a propensity function that depends on neighboring cells and the treatment of the boundary cells becomes complicated.

For a periodic boundary condition (PBC), whenever a particle exits the rightmost boundary of the box, it automatically enters the leftmost boundary, with its new position being its original position modulo the domain length, and similarly for the case where it exits the leftmost boundary.

3.3. **Setup.** With all the cells initialized, we can now a function to fill the cells with particles, where we use the given number density as the mean to generate a Poisson distributed number of particles for each cell and fill the cell with a corresponding number of particles with uniformly generated positions. The motivation for using a Poisson distributed density is that the medium is assumed to be homogeneous and that cells are supposed to be independent of each other.

Now, we will discuss S-BD-RME and S-RBD.

3.4. **S-BD-RME.** Suppose we are interested in the system's evolution from $t = 0$ to $t = T$. We need to first divide T into N timesteps of equal length, where $dt = T/N$ such that a particle on average diffuses a small fraction of the cell size. For each timestep, we shift all particles by a random amount, and then let each particle undergo Brownian motion for $dt/2$ with the mass diffusivity D_i, that is, $r_i = r_i + N_d\sqrt{2D_i dt}$, where N_d is the standard normal vector with dimension d. Then, process stochastic reactions for a timestep dt independently for each cell.

Now, we explain what the algorithm does in each timestep. To reduce the error in the algorithm, we use a random shifting for the grid. For actual implementation, we translate all particles by a small fraction (less than one half in absolute value) of the cell size in each dimension. As discussed earlier, this algorithm suffers from issues such as non Galilean-invariance. The motivation for using such a translation is to ameliorate the effect introduced by the grid structure. Later at the end of this timestep we will do a backward translation so that this translation will reduce the artifact without introducing new artifacts.

After the random translation, we start simulating the evolution of the system. In order to combine both diffusion and reaction together simulltaneously, it requires an extremely high computational cost. Hence, we need to divide the reaction-diffusion process into two separate parts, reaction and diffusion. In [5], Lie splitting was used. In other words, the algorithm processes a diffusion of time dt, followed by a reaction process of time dt. Here, as an improvement, we use strang splitting, that is, we process a diffusion by $dt/2$, followed by a reaction process of dt, and eventually another diffusion by $dt/2$.

The diffusion part is simple. We just generate random numbers from the standard normal distribution and use them, along with the mass diffusivity and diffusion time, to determine the displacement.

The formula for displacement in each dimension of a particle is,

$$s = \sqrt{2Dt}Z \tag{21}$$

where D is the mass diffusivity of the particle, t is the length of time, and Z is the standard normal random variable. Note that after making this change, the particle may move outside the box and we need to make corresponding changes with respect to the boundary conditions.

For the reaction, we iterate through each cell and call the reactor module to process reactions. Recall that in S-BD-RME, we only allow reactions where all the reagents are in the same cell. The reactor module works as described below.

(1) Set $i = 1$.

(2) Build a recorder matrix of size N_s by $C_{\max}$, where $C_{\max}$ is the maximum allowed number of particles in a cell, estimated separately at the beginning. Then, let the recorder matrix store the particle indices and species of the i-th cell.

(3) Set the clock to 0.

(4) Calculate the propensity functions and generate the probabilities of each reaction. Also, update the time correspondingly.

(5) Generate a uniform random real in $[0, 1]$ and decide the corresponding reaction to process. Then, process that reaction with randomly chosen candidates from the recorder matrix.

(6) Update the recorder matrix.

(7) If the current time is still smaller than dt, go to step 3. Otherwise, increase i by 1.

(8) If i is larger than the number of reaction cells in the box, then end the routine. Otherwise, go to step 2.

In particular, recall that the pronpensity function for reactions is given by,

- 0 reagents: $\alpha_0 = k_0 V_c$, where k_0 is the reaction rate and V_c is the volume of the cell;
- 1 reagent: $\alpha_1 = k_1 A_1$, where k_1 is the reaction rate and A_1 is the number of reagent particles;
- 2 reagents: $\alpha_2 = k_2 A_1 A_2 / V_c$, where k_2 is the reaction rate and A_1, A_2 are the numbers of candidate reagent particles.

For S-BD-RME, the numbers A_1, A_2 refer to the numbers of particles of corresponding species in the cell of interest. Here we need V_c as a normalizing constant so that the cell volume does not change the reaction count.

The detailed implementation of the algorithm above will not be discussed since S-BD-RME is not the central topic of this thesis and also because its implementation is relatively simple.

After calling the reaction module, we can diffuse the particle again by $dt/2$ and then do a backward translation to finish the timestep.

3.5. **S-RBD.** For S-RBD, we do not need the random shifting since S-RBD is essentially grid-free. However, to make the system grid-free, we need to allow binary reactions where two reagents are in different cells. Hence, we need to use a rather different reaction module. In S-BD-RME, we can process reaction for each cell independently. However, to make our method grid-free, there will be interference between neighboring cells. Therefore, we need to have a proper ordering for the cells when processing reactions. An inspiration comes from the next subvolume method (NSM) proposed in [8]. The NSM first calculates the propensity functions for each cell, and then generates the corrsponding event time for each cell, where the event time means the time of occurence for the first event in a cell. With these times, we can then build a priority queue and process reactions by reading data and then making appropriate changes to the queue. Essentially, the idea of S-RBD is a combination of NSM, I-DSMC, and CRDME.

Again, we use strang splitting for higher accuracy and the outline for each timestep is the following.

(1) Let each particle undergo Brownian motion for $dt/2$ with the mass diffusivity D is exactly the diffusion constant given in the input file.

(2) Call the S-RBD reaction module to process stochastic reactions for dt.

(3) Let each particle undergo Brownian motion for $dt/2$.

Note that we abandon the random shifting since the algorithm is grid-free and there is no need for a random shifting. However, because of the possibility of inter-cell reactions, we cannot call the reaction module cell by cell. To see why this is true, consider the following case. Suppose we still process reactions cell by cell and we visit cell i first. While visiting cell i, the reaction module may process an inter-cell reaction in cell i and its neighboring cell j. However, in the global system, one inter-cell reaction that involves particles in cell j and another cell k may happen first, and thus consuming some particles in cell j and k. In this way, the reaction in cell i and j may never happen, if the reagent in cell j is the last particle in the cell. Even in case there are other reagents left in cell j, it still suffers from the problem of a wrong propensity function since this mistakenly decreases the propensity function of cell k.

Essentially, what we want to do is to create a list of all overlapping particles that can react. To do so, we use the idea of NSM and combine it with Gillespie's SSA. Hence, we need a different way to calculate the propensity function since now we are taking inter-cell reactions into account. For reactions with 0 or 1 reagents, the propensity functions remain the same for the obvious reason. Before defining the propensity function, we need to define a few concepts.

In a system of dimension d, we are interested in systems only when $d \leq 3$. However, when $d < 3$, we do not just view the system as in lower dimension. Instead, we view the system as if the dimensions from $d + 1$ to 3 do not affect system evolution at all. That is, particles' diffusion in the extra dimension is not important and separation in the extra dimensions does not affect reactions. That being said, the underlying system is still 3-dimensional and the extra dimensions have impact on parameters such as the volume of the system. Then, we define the "thickness" h for the system to be the product of the lengths in the extra dimensions. When $d = 3$, h is a trivial 1.

Also, we define the effective area S of a binary reaction to be the volume of the d-ball with radius equal to the sum of radii of the two participating reagent particles. Now, for binary reactions with reagents of species sp_1 and sp_2, we modify the propensity function for this reaction in cell i as follows,

(1) calculate a normalizing factor to be $1/(hS)$;

(2) if $sp_1 = sp_2$, go to step 3; otherwise, go to step 5;

(3) calculate S-RBD reaction rate to be $k' = k/(hS)$;

(4) calculate S-RBD propensity function for this binary reaction to be $\alpha = k' A_{sp_1} A'_{sp_1}/V_c$, where A_{sp_1} is the number of particles of species sp_1 in cell i and A'_{sp_1} is the number of particles of species sp_1 in cell i and its neighboring cells, that is cells that share at least one boundary with cell i;

(5) calculate S-RBD reaction rate to be $k' = k/(2hS)$;

(6) divide this reaction into two reactions, where the first one has sp_1 as its first reagent and the second one has sp_2 as its first reagent;

(7) calculate S-RBD propensity for these two sub-reactions to be $\alpha_1 = k' A_{sp_1} A'_{sp_2}/V_c$ and $\alpha_2 = k' A_{sp_2} A'_{sp_1}/V_c$.

We make some remarks of the calculation of the propensity function above. To include inter-cell reactions, we need to use a propensity function that accounts for particles in neighboring cells. However, if we simply define the propensity function of the reaction to be $\alpha = kA'_{sp_1} A'_{sp_2}$, then we are overestimating the propensity since in this way it is possible that neither participating particles are in cell i. Hence, to ensure that at least one of the reagent particle is in cell i, we need to have a primary particle that is in cell i and the other particle can be in either cell i or its neighboring cells. However, for a reaction where $sp_1 \neq sp_2$, the number of possible pairs for reactions depends on the species of the primary particle. Therefore, in such a case, we need to divide the reaction into two reactions, where they have different primary particles and thus different propensity functions $kA_{sp_1} A'_{sp_2}/2$ and $kA_{sp_2} A'_{sp_1}/2$, where the factor $1/2$ is used since each of the sub-reaction should only have half of the original reaction's reaction rate.

With this modification, the propensity function seems to allow inter-cell reactions. However, a careful reader can notice that this propensity function is still not ideal since in case of large cell size it allows for particles with big separation to react. To resolve this issue, we require furthermore that even if a pair of particles are chosen to be reaction candidates, they still need to be within a reactive separation for the reaction to take place. In case where the cell size is larger than the particle's typical radius, where almost all realistic cases fall into this category, the number of binary reactions declines if we use the same propensity function proposed in last paragraph. Therefore, to compensate for this difference and make the reaction rate in S-RBD consistent with that in S-BD-RME, we need to find the normalizing constant.

Now, we have the propensity functions for each cell. However, as mentioned earlier, we cannot process reactions cell by cell and must perform the reactions with respect to the global clock. To achieve this goal, we create an event queue and store the list of time for occurence of reactions in all cells in ascendending order. In this way, we can always visit the cell where the first reaction takes place, take appropriate actions, and update the corresponding cell and its negihboring cells' status, propensity functions, and positions in event queue.

In the next section, we will present a detailed description of the implementation and also provide some psuedocode.

3.6. **Implementation of S-RBD.** In this section, we will demonstrate the S-RBD process. Note that there are other alternatives in implementation, which we do not use for memory or time reasons but may be useful in certain circumstances. We will start with a discussion of the major data structures and setups, then shift to initialization of simulation, and then illustrate the diffusion and reaction processes in each timestep.

3.6.1. *Setup.* Particles are central building blocks of our algorithm. In implementation, we define a particle to be a data structure that carries position and species information. For the position, regardless of the dimension d, we always assume that a particle has 3-dimensional position. In cases where $d < 3$, a particle's position in the extra dimensions will always be default to 0. The reason we keep a 3-dimensional position is for consistency and simplicity in code.

For the reaction part, we need a way to find particles in any given cell. The way we achieve this goal in S-BD-RME is to build a local recorder matrix. However, for S-RBD, a local recorder matrix is insufficient because of inter-cell reactions. The memory cost for using a global recorder matrix can be huge. Instead, we chose to use linked list, which is less efficient in time but much more efficient in memory. For our purpose, a singly linked list is sufficient. Alternatively, one can also use a doubly linked list, which speeds up some part of searching but increases the cost of maintaining the data. Specifically, we use a linked list called "nextParticle" and another array called "firstParticle". We used firstParticle to find the first particle of a certain species in a given cell and then for particle p_i, $\text{nextParticle}(p_i) = p_j$ if p_j is the next particle in the same cell with the same species and $\text{nextParticle}(p_i) = 0$ if p_i is the last particle in that cell with the same species.

For our event queue, we used a min-heap to store the list of next occurence times and index for each cell. We order the heap by the time for next occurence in a ascendening way.

3.6.2. *Algorithm Outline.* Here we give an outline for the simulation.

(1) Initialize the box and fill cells with particles. Set clock to 0.
(2) Call the diffusion module for $dt/2$.
(3) Call the reaction module for dt.
(4) Call the diffusion module for $dt/2$.
(5) Clean data that are no longer useful.
(6) Increment the time by dt. If the new time is larger than desired simulation time, stop the simulation; otherwise, go to step 2.

In the following, we will give an outline for initialization, diffusion, and reaction processes.

Simulation Part I: Initialization

---

(1) Initialize simulation parameters such as the calculating size of a reaction cell or a sample cell, the mass diffusivities for all species, and classifying reaction types.

(2) Create the box and all necessary data structure.

(3) Use the number densities for all species as the mean of Poisson distribution to generate Poisson random numbers and fill the sample cells with particles accordingly.

Simulation Part II: Diffusion

---

(1) For each particle, generate a standard normal random vector of dimension d. The displacement for the particle is this vector multiplied by $\sqrt{2Dt}$ termwise.

(2) Handle boundary conditions if necessary.

Simulation Part III: Reaction

---

(1) Set the clock to 0.

(2) Reset th which is used as an event queue. Then, go through the list of cells and calculate their S-RBD propensity functions. Generate corresponding stochastic occurence times for all cells.

(3) Check if the first event in the queue has a time exceeding dt. If so, stop the module.

(4) Choose the first event in the queue and go to the corresponding cell. Calculate its propensity functions and decide the reaction type for the next reaction.

(5) If this reaction is not binary, process the reaction immediately. Otherwise, first choose two random particles of proper species, with the first one from our central cell and the second one from a neighboring (including the central one) cell. If these two particles are within the reaction radius, process the reaction. Otherwise, this reaction is invalid and we do not change the particles.

(6) If a reaction takes place in some reaction cells the previous step, erase the first event from the queue and update the occurence times for all reaction cells whose propensity functions change. Also, update the clock time. Otherwise, update the queue only for the first event's corresponding cell. Go to step 3.

4. Numerical Test

In this section we show the result of numerical tests.

4.1. **Selection Test.** This test checks if certain candidates are chosen uniformly to be reaction candidates. Specifically, we test (1) for birth reaction, if each cell is chosen uniformly; (2) for annihilation reaction, if each particle is chosen uniformly, (3) for binary reactions, if each overlapping pair is chosen uniformly.

In this test, we used an input file that contains four reactions:

- $\emptyset \to A$
- $A \to \emptyset$
- $A + B \to \emptyset$
- $A + A \to \emptyset$

We included all four reactions to make sure they are not affecting each other. There is no diffusion. For test, we used two criteria: (1) visual check of the plot to see if it looks uniform; (2) Pearson's Chi-square test to see if the data is close to uniform distribution.

The following figures are the bar plots of the four reactions in order as listed above. The horizontal axis is the index of each candidate, which may be a cell, a particle, or an overlapping pair. The vertical axis shows the number of observation. The horizontal red line is the mean. The two green lines are mean plus/minus two square root of mean. Hence, around 95% of the bars should lie inside the two green lines.

As for the result of the statistical test, we used a significance level of 0.05. The corresponding Chi square critical values are 77.9305, 340.3278, 62.8296, 47.3999, all higher than the Chi square statistic obtained from data. The visual check and the Chi square test thus suggest that the selection is very likely to be uniform.

4.2. **Position Test.** This test checks if position is generated uniformly within the reactive radius. The reactions are $C \to A + B$ and $\emptyset \to A + B$. In this test, we simulated both 2D and 3D input, processing two reactions together (without diffusion, without real reactions). We want to check if (1) for the first reaction, one particle is born in C's position and the other one is uniformly distributed inside the circle of radius $r_A + r_B$ centered at C; (2) for the second reaction, one particle is uniformly distributed in the domain and the other is uniformly distributed inside the circle of radius $r_A + r_B$ centered at the first particle.

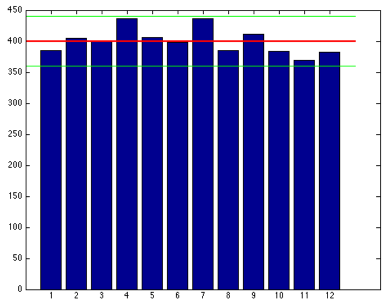
For this test, we used two criterions: (a) visually check if the plotted points fill the circle uniformly; (b) plot the probability density function and compare it with the percentage histogram. For (a), plotting the points inside a 3D ball does not convey enough information since the detail inside the ball cannot be seen clearly. For (b), I first collected the first n sets of data, where n was chosen to be around 40,000 because plotting too many data points is too slow in Matlab and $40,000$ provides sufficient accuracy already. For 2D, when projected to the x-dimension, the theoretical probability density function is given by

$$p(x) = \frac{2\sqrt{r^2 - x^2}}{A} \tag{22}$$

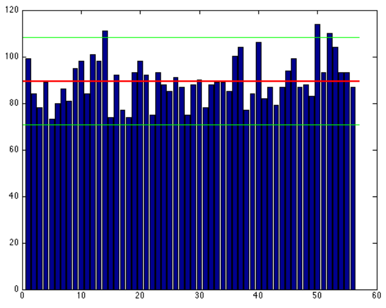
where A is the area of the circle, $r = r_A + r_B$. For 3D, it should be

$$p(x) = \frac{\pi(r^2 - x^2)}{V} \tag{23}$$

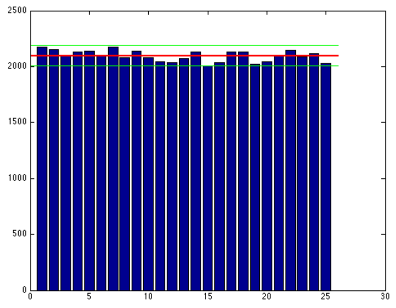
(A) birth



(B) annihilation



(C) pair selection: A+B



(D) pair selection: A+A

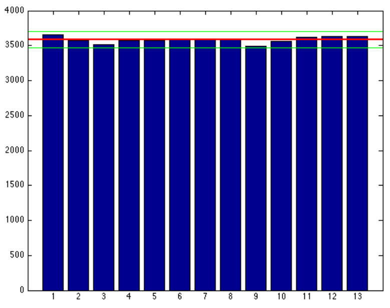


FIGURE 1. Selection Test

where V is the volume.

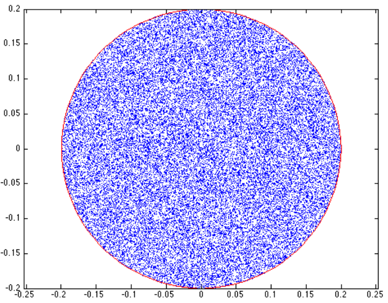
In particular, we manually created a particle C near boundary so we can test if boundary condition works. We checked all plots and they worked well. Three sample plots are also attached in the below, where they correspond to 2D uniform circle visual check, 2D PDF-histogram comparison, and 3D PDF-histogram comparison, in order.

4.3. **Rate Test.** This test checks if reaction rate approximates theoretical prediction. Specifically, we allowed diffusion but no real reactions. This test consists of two reactions:

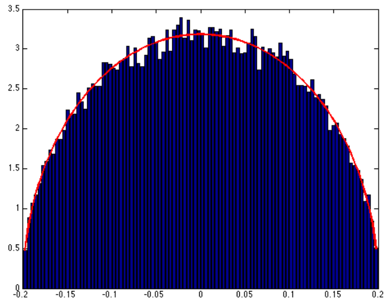
- $A + B \to \emptyset$
- $A + A \to \emptyset$

We start with a certain number of different particles. In each timestep, a particle diffuses a fraction (on average, 0.15-0.3) of cell length (for convenience, I set the cells to be cubic). We record the number of reactions in each step and calculate the relative error by using theoretical prediction. Suppose the input rate for a binary reaction with reactants A and B is r, then the corresponding

(A) 2D Uniform Circle



(B) 2D PDF-Histogram Comparison



(C) 3D PDF-Histogram Comparison

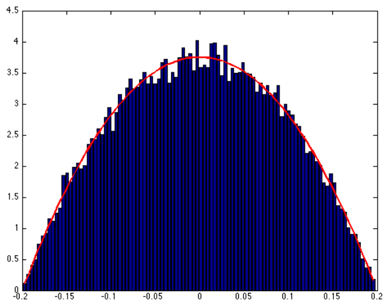


FIGURE 2. Position Test

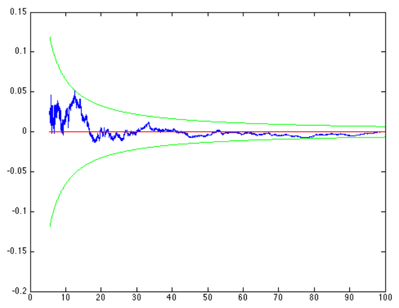
real rates for RDME and IRDME models are:

$$\begin{aligned}\lambda_{\text{RDME}} &= r * c * /V \\ \lambda_{\text{IRDME}} &= r * c * V_{AB}/V\end{aligned} \tag{24}$$

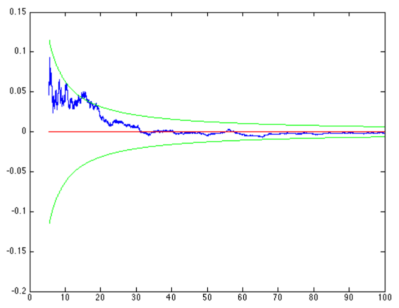
where $c = N_A * N_B$ if $A \neq B$, with N_A, N_B being the number of A, B particles in the domain and $c = N_A * (N_A - 1)$ if $A = B$, V is the volume of the box, and V_{AB} is the volume of the ball with radius equal to the sum of radius of A and B.

In this test, we let the particles diffuse and process fake reactions. The recorded reaction counts are then used to compare with theoretical values. In the following, we attach several plots, where values of horizontal axis correspond to real time and values of vertical axis correspond to relative error, given by $(C_R - C_E)/C_E$, where C_R is the reaction count generated by the code and C_E is the theoretical count calculated separately. In addition, the red line is theoretical mean and the two green lines are plus/minus 2/3 sqrt(mean), respectively. The first two plots use 2 sqrt(mean) and the last two use 3 sqrt(mean).

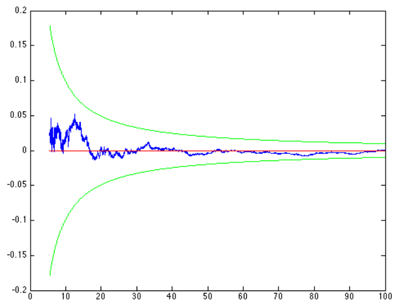
(A) 2D $A + A \to \emptyset$, 95% Confidence



(B) 2D $A + B \to \emptyset$, 95% Confidence



(C) 2D $A + A \to \emptyset$, 99.7% Confidence



(D) 2D $A + B \to \emptyset$, 99.7% Confidence

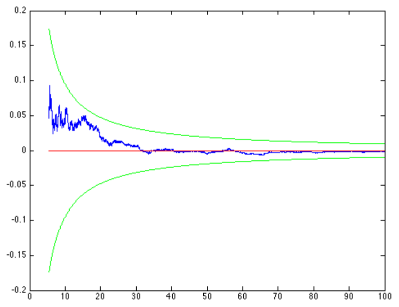


FIGURE 3. Rate Test

4.4. **Test on BPM Model.** Originally developed in [4] and [3], the Baras-Pearson-Mansour model is a variation of the Gray-Scott model (see [11] and [12]). Unlike the Gray-Scott model, the BPM model does not include trimolecular reactions. The BPM model assumes six different species and the following reaction network:

(1) $U + W \Leftrightarrow V + W$

(2) $2V \Leftrightarrow W + S$

(3) $V \Leftrightarrow S$

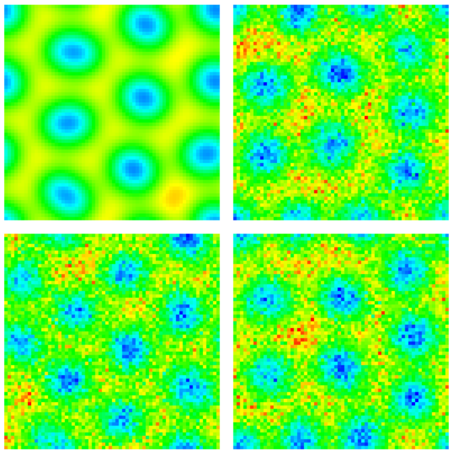
(4) $U \Leftrightarrow U_f$

(5) $V \Leftrightarrow V_f$

The BPM model can exhibit limit cycles and bimodal states when provided with appropriate initial and boundary conditions. In the following, we compare the simulation results of the BPM model with four different methods: deterministic approach, fluctuating hydrodynamics, S-BD-RME, and S-RBD. In particular, for the fluctuating hydrodynamics, we used multinomial diffusion and Gillespie's SSA with a splitting scheme.

Firstly we compare the time until occurence of the Turing pattern.

(A) 2D Turing Pattern, Limit Cycle



(B) 2D Turing Pattern, Bimodal

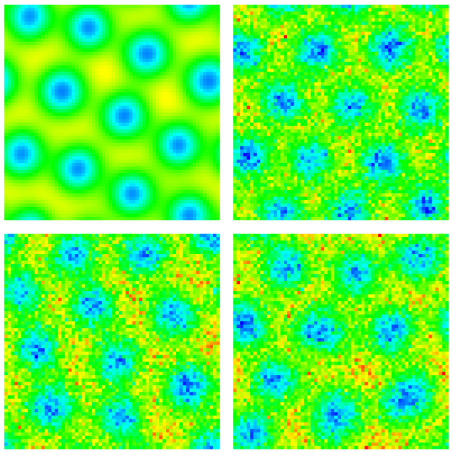
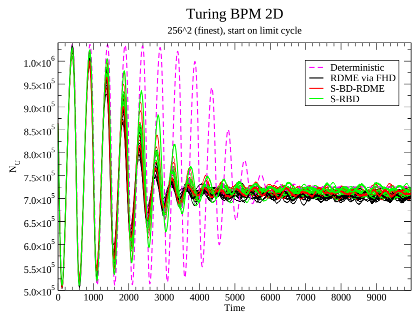


FIGURE 4. Turing Pattern in 2 Dimensions

(A) 2D BPM, Limit Cycle



(B) 2D BPM, Bimodal

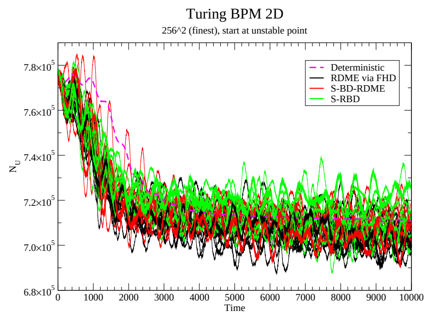


FIGURE 5. Total Number of U Particles

In the two figures above, the top left results from deterministic approach, the top right results from fluctuating hydrodynamics, the bottom left results from S-BD-RME, and the bottom right results from S-RBD. As can be seen from the figure, there is a qualitative difference between the deterministic approach and the other three approaches. However, the quantitative difference is unclear from these figures along.

Starting from uniform configurations for both limit cycle and bimodal states, we generate the following results for the four methods, where the $y-$axis represents the total number of U-particles.

We chose a fitting formula,

$$y = (1 - tanh((x - a_0)/a_2)) * (a_1 * sin(a_3 * x + a_4) + a_5) + a_6 \tag{25}$$

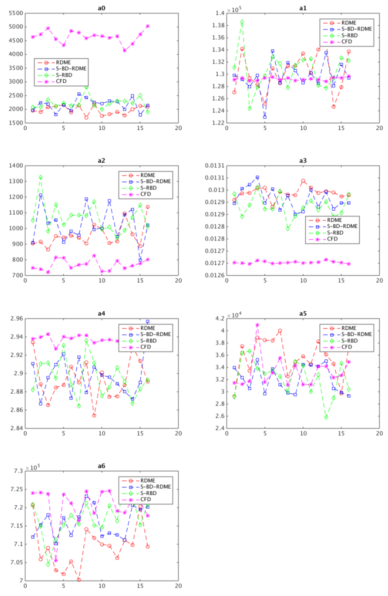


Figure 6. 2D BPM Coeffcients Fitting, Limit Cycle

The fitted values for the limit cycle case is,

For a_2, one observes some statistical differences. We believe that there are quantitative difference among these three stochastic methods and will investigate the underlying reasons with more numerical tests.

5. Future Work

At this point, we have finished a series of implementation and numerical tests. There are some more questions concerning the difference among fluctuating hydrodynamics, S-BD-RME, and S-RBD to be answered. Moreover, we would like to provide a theoretical reason to explain the difference and understand the behaviors of these three methods better. The future work will be summarized on our ongoing research paper.

References

[1] Francis J Alexander and Alejandro L Garcia. The direct simulation monte carlo method. *Computers in Physics*, 11(6):588, 1997.

[2] Ariel Balter and Alexandre M Tartakovsky. Multinomial diffusion equation. *Physical Review E*, 83(6):061143, 2011.

[3] Florence Baras, M Malek Mansour, and JE Pearson. Microscopic simulation of chemical bistability in homogeneous systems. *The Journal of chemical physics*, 105(18):8257–8261, 1996.

[4] Florence Baras, JE Pearson, and M Malek Mansour. Microscopic simulation of chemical oscillations in homogeneous systems. *The Journal of chemical physics*, 93(8):5747–5750, 1990.

[5] TaiJung Choi, Mano Ram Maurya, Daniel M Tartakovsky, and Shankar Subramaniam. Stochastic operator-splitting method for reaction-diffusion systems. *The Journal of chemical physics*, 137(18):184102, 2012.

[6] Masao Doi. Stochastic theory of diffusion-controlled reaction. *Journal of Physics A: Mathematical and General*, 9(9):1479, 1976.

[7] Aleksandar Donev, Berni J Alder, and Alejandro L Garcia. A thermodynamically consistent non-ideal stochastic hard-sphere fluid. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(11):P11008, 2009.

[8] Johan Elf and Måns Ehrenberg. Spontaneous separation of bi-stable biochemical systems into spatial domains of opposite phases. In *Systems Biology, IEE Proceedings*, volume 1, pages 230–236. IET, 2004.

[9] Radek Erban, Jonathan Chapman, and Philip Maini. A practical guide to stochastic simulations of reaction-diffusion processes. *arXiv preprint arXiv:0704.1908*, 2007.

[10] Daniel T Gillespie, Andreas Hellander, and Linda R Petzold. Perspective: Stochastic algorithms for chemical kinetics. *The Journal of chemical physics*, 138(17):170901, 2013.

[11] P Gray and SK Scott. Autocatalytic reactions in the isothermal, continuous stirred tank reactor: isolas and other forms of multistability. *Chemical Engineering Science*, 38(1):29–43, 1983.

[12] P Gray and SK Scott. Sustained oscillations and other exotic patterns of behavior in isothermal reactions. *The Journal of Physical Chemistry*, 89(1):22–32, 1985.

[13] Samuel A Isaacson. Relationship between the reaction–diffusion master equation and particle tracking models. *Journal of Physics A: Mathematical and Theoretical*, 41(6):065003, 2008.

[14] Samuel A Isaacson. The reaction-diffusion master equation as an asymptotic approximation of diffusion to a small target. *SIAM Journal on Applied Mathematics*, 70(1):77–111, 2009.

[15] Samuel A Isaacson. A convergent reaction-diffusion master equation. *The Journal of chemical physics*, 139(5):054101, 2013.

[16] Samuel A Isaacson and David Isaacson. Reaction-diffusion master equation, diffusion-limited reactions, and singular potentials. *Physical Review E*, 80(6):066106, 2009.

[17] A Lemarchand and B Nowakowski. Fluctuation-induced and nonequilibrium-induced bifurcations in a thermochemical system. *Molecular Simulation*, 30(11-12):773–780, 2004.