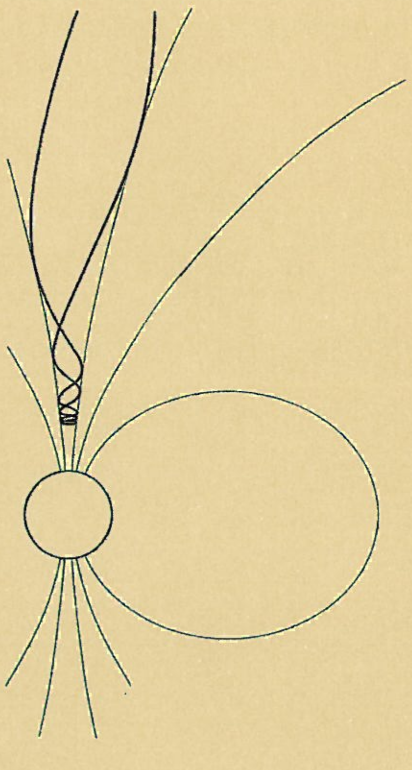
Evaluation of a New Resampling Scheme for δf Monte Carlo Methods

Sadegh Askari



Master of Science Thesis
Stockholm, Sweden 2011

KTH Electrical Engineering

Evaluation of a New Resampling Scheme for δf Monte Carlo Methods

Sadegh Askari

Master Thesis in Electrical Engineering

ED222X Degree Project in Electrical Engineering, Second Level

School of Electrical Engineering

KTH Royal Institute of Technology

SE-10044 STOCKHOLM

Evaluation of a New Resampling Scheme for δf Monte Carlo Methods

S. Askari

Alfven Laboratory, Department of Electrical Engineering,

Royal Institute of Technology, SE-10044 STOCKHOLM, Sweden

Askari@kth.se

Abstract

A new class of methods for reducing the number of particles in delta-f Monte Carlo simulations is presented. The reduction of particles is necessary when there is a continuous growth of the number of particles during the simulation. The method is based on *resampling* the particles distribution in local partitions of the phase-space. The *resampling* is accomplished by replacing particles with fewer numbers of new particles in each partition while ensuring that the moments of distribution are conserved. It's demonstrated that the method well preserves the distribution function.

ABSTRAKT

En ny klass av metoder för att minska antalet partiklar i deltaf- MonteCarlo simuleringar presenteras. Det är nödvändigt att minska antalet partiklar när vi har en kontinuerlig tillväxt av partiklar under en simulering. Metoden baserar på återsampling av fördelningsfunktionen i lokala partitioner i fas-rummet. Återsampling sker genom att man ersätter partiklar med ett färre antal partiklar i varje partition och samtidigt bevarar momenten av fördelningsfunktionen. Metoden visar sig kunna bevara fördelningsfunktionen väl.

Contents

1. Introduction and Background

The Monte-Carlo (MC) method is an established method of solving Fokker-Planck equation in multidimensional systems [1]. The main drawback of MC is the slow convergence, suggesting application of methods such as the δf method [2], which makes the computations faster. In the δf method, only the perturbation is represented by the modeled set of particles instead of the entire solution. Therefore, the error arising from approximating the expected value by a sample average can be reduced. A key feature of the δf method in the Fokker-Planck equation is the modeling of the source term. Particle-weight methods are commonly used to handle the source term [3], but can cause numerical issues. A straightforward method to handle the source term by adding new particles with positive and negative weights has been presented in [4]. However, the growth of the number of particles increases the computational work, and overlapping of particles with positive and negative weight reduces the accuracy. It is indispensable to find a method to remove overlap and reduce the number of particles during the simulation. The issues can be solved by projecting the distribution function on a finite element base, and resampling from the new function, However, this becomes less practical in higher dimensional problems. In the present thesis, a new method for reducing the total number of particles entitled as ball-resampling is developed, which can resolve both problems, simultaneously. Furthermore, the method is independent of dimension making it suitable for multidimensional problems.

The outline of the thesis is as follows. In section 1, the relation between stochastic differential equation and partial differential equation e.g. the Fokker-Planck equation and the method of solving a partial differential equation using it‘s characteristic stochastic differential equation is discussed. The Monte-Carlo method in stochastic differential equations and δf Monte-Carlo method are explained concisely in section 2. In section 3, the resampling algorithm and several subjects related to the algorithm such as kd-tree query, method of reconstruction of a distribution function, and moments of distribution is described. The detail of the algorithm and results of simulation is discussed in section 4, and finally in section 5 the conclusions are presented.

1.1. The Fokker-Planck equation; it's relation with the stochastic processes

To introduce the Fokker-Planck equation and it's relation with stochastic processes we first discuss the stochastic differential equations. The difference between a stochastic differential equation and a deterministic differential equation could be illustrated with a simple model. Consider the case that a small particle of mass m is suspended in a fluid. The friction force, $F_f = -\alpha v$, acts on the particle and the equation of motion in the simplest case is

$$m\dot{v} + \alpha v = 0 \tag{1.1}$$

with a solution determined by the initial condition $v(0)$. This equation is an ordinary differential equation of the form

$$\frac{dX}{dt} = a(X, t) \tag{1.2}$$

that is called a *deterministic equation*, i.e., the solution is completely determined by its initial value, $X(0)$.

In a microscopic picture the friction in the fluid is replaced by collision with molecules. The momentum of particle is transferred to the fluid molecules and the difference between the average velocity of the molecules and the test particle reduces to zero.

The corresponding microscopic form of eq. (1.1) is obtained by adding a random force $F_s = m\Gamma(t)$ to the total force $F = F_f + F_s$ acting on the test particle, and it results in following equation for the particle velocity

$$\dot{v} + \frac{\alpha}{m} v = \Gamma(t) \tag{1.3}$$

The $\Gamma(t)$ term is called Langevin force, and the eq. (1.3) is a stochastic differential equation. The mathematical form of stochastic differential equation corresponding to eq. (1.1) is written as:

$$dX = a(t, X)dt + \sigma(t, X)dW(t) \tag{1.4}$$

where $a(t, X)$ and $\sigma(t, X)$ are called drift and diffusion coefficient, respectively, and $W(t)$ is a Wiener process. The Wiener process or Brownian motion is a continuous-time stochastic process and in one dimension is characterized by the following properties [1]:

- $W(0) = 0$
- If $t = t_n > t_{n-1} > ... > t_0 = 0$ then the increments $\{W(t_0), W(t_1) - W(t_0), ..., W(t_n) - W(t_{n-1})\}$ are independent.
- For all $t \geq s \geq 0$ the increment $W(t) - W(s)$ has the Normal distribution, $N(0, t - s)$

Due to the stochastic term $W(t)$ the eq. (1.4) is not deterministic.

A relation exists between solutions of stochastic differential equations and parabolic partial differential equations e.g. the Fokker-Planck equation, which offers a method of finding the solution of these equations by simulating its corresponding stochastic differential equation. This simulating process could be viewed as a type of Monte-Carlo calculation (see section 2.1). The relation between the Fokker-Planck equation and eq. (1.4) is explained as follows; Suppose eq. (1.4) as the law of motion for a mass of particles with the distribution function $f(0,x)$ at $t=0$. The Fokker-Planck equation

$$\frac{\partial f}{\partial t} = -\frac{\partial}{\partial x}\left(a(x,t)f\right) + \frac{1}{2}\frac{\partial^2}{\partial x^2}\left(\sigma^2(x,t)f\right) \tag{1.5}$$

is the equation for the probability of finding the particles at time t and position x , $f(t,x)$, at each time $t>0$ resulted from the initial probability $f(0,x)$ and the movement of the particles. The present work has been motivated during the research in ion cyclotron radiation heating (ICRH) of fusion plasma where it is essentially to solve a Fokker-Planck equation to determine the distribution function of the heated ion species. In the next section, a brief form of this equation is presented, and the method of solving the Fokker-Planck equation using stochastic process is discussed in section 2.1.

1.2. Application to ion cyclotron heating in fusion plasma

In an ordinary gas, molecules bounce off each other like colliding billiard balls. In a plasma with large Coulomb logarithm, $\ln\Lambda >> 1$, Coulomb collisions mainly cause small deflections of the velocity vector of the particle. Therefore, the velocity of charged particles changes only gradually, which makes it possible to describe Coulomb collisions by a Fokker-Planck equation in velocity space. Similarly, the wave-particle interactions in RF-heating can be described by small random changes of the velocity of the particles that is also described by a Fokker-Planck equation.

Ion cyclotron resonant heating (ICRH) is one of the candidates for additional heating of plasma in fusion devices. The radio-frequency waves accelerate the resonant ions at their cyclotron frequency or low harmonics of the cyclotron resonance. High power ion cyclotron heating results in a non-Maxwellian distribution function with a high-energy tail. A classic work for describing distribution function during ICRH using a Fokker-Planck equation has been presented by Stix [5], and in one dimension has the following form

$$\frac{\partial f}{\partial t} = (C+Q)f \tag{1.6}$$

where C is Coulomb collision operator and Q is the quasi-linear RF operator.

2. δf Monte-Carlo method

2.1 Numerical solution of stochastic differential equations; Monte-Carlo method

Monte Carlo methods are numerical methods where random numbers are used to conduct computational experiments, and are applicable in solving stochastic differential equation (1.4). The first step in solving eq. (1.4) numerically is to find a time-discrete approximation of the equation. The simplest approximation is called Euler-Maruyama approximation that is defined as [6]: Consider the Ito's stochastic differential equation (1.4) with initial condition $X(t_0) = X_0$, for a given discretization $t_0 < t_1 < ... < t_M = T$ of the time interval $[t_0, T]$, an Euler-Maruyama approximation is satisfying the iterative scheme

$$X_{m+1} = X_m + a(t_m, X_m)\Delta t_m + \sigma(t_m, X_m)\Delta W(t_m) \tag{2.1}$$

where $\Delta t_m = (t_{m+1} - t_m)$ and $\Delta W(t_m) = \xi\sqrt{t}$, and $\xi \in n(0,1)$ are normally distributed random numbers. Practically, this equation is solved for M time steps and N number of paths corresponding to N number of random numbers ξ. The error in this method consists of two parts; the statistical error and the time discretization error. The statistical error is due to the limited number of particles N and is estimated as (e.g. see [1])

$$E[g(X(T))] - \sum_{i=1}^{N} \frac{g(\tilde{X}(T)^i)}{N} = C\frac{(Var[g(\tilde{X}(T))])}{N^{1/2}} = O(N^{-1/2}), \tag{2.2}$$

where $\tilde{X}$ is an approximation of X obtained from the numerical discretization. The expected value for a function g using Monte-Carlo method is given by

$$E[g(X(T))] \approx \sum_{i=1}^{N} \frac{g(\tilde{X}(T)^i)}{N} \quad . \tag{2.3}$$

The statistical error or sampling error in Monte-Carlo method is proportional to $1/\sqrt{N}$ where N is the number of realizations.

The time discretization error is the difference between the exact solution and the approximate solution calculated from discretized equation:

$$E[g(X(T))] - E[g(\tilde{X}(T))] \; .$$

The magnitude of this error depends on the time step and the convergence order of the scheme e.g. Euler-Maruyama scheme, that is used. Convergence means that in what sense or how fast does the $\left|\tilde{X}_n - X_n\right| \to 0$ as $\Delta t \to 0$.

2.2 δf method

In the δf method, the total distribution function is decomposed into two parts $f(x,t) = G(x) + \delta f(x,t)$, a background distribution function $G(x)$ and a perturbed component δf that is small relative to $G(x)$, $\delta f / G << 1$. The advantage with the δf method is that only the perturbed part of distribution is calculated rather than the total distribution. Consequently, the statistical error associated with the function $f(x)$ due to the use of finite number of particles is reduced. The number of particles involved in δf simulation is reduced by a factor $\int \delta f dx / \int f dx$. Consider the following differential equation for the distribution function f

$$\frac{\partial f}{\partial t} = L(f)\,, \tag{2.4}$$

where L is a linear operator. Inserting $f(x,t) = G(x) + \delta f(x,t)$ into this equation results in an inhomogeneous equation for δf

$$\frac{\partial \delta f}{\partial t} - L(\delta f) = L(G)\,. \tag{2.5}$$

The left hand side of eq. (2.5) is solvable using the Monte-Carlo method for stochastic differential equations. A problem is how to treat the source term $L(G)$. In conventional δf method [3], each particle is assigned an initial weight such that the particles and their weight provide a Monte-Carlo simulation of δf, and the source term is handled by changing the weight. However, the method can cause some numerical issues [4]. A straightforward method to model the source term is by adding particles directly with the following rate [4]

$$\frac{dN}{dt} = \int_{R^n} \left|L(G)\right| J dx \tag{2.6}$$

where J is the Jacobian. The conservation of the total number of particle necessitates

$$\int L(G) J dx = 0$$

This equation implies that the source term consists of positive $L(G)_+$ and negative parts $L(G)_-$ such that

$$L(G) = L(G)_+ - L(G)_- \ .$$

The linearity of L permits the splitting of eq. (2.5) into two equations corresponding to positive and negative sources

$$\frac{\partial \delta f_+}{\partial t} - L(\delta f_+) = L(G)_+ \tag{2.7a}$$

$$\frac{\partial \delta f_-}{\partial t} - L(\delta f_-) = L(G)_- \ , \tag{2.7b}$$

where δf_+ and δf_- represent the distributions of particles with positive and negative weights satisfying $\delta f = \delta f_+ - \delta f_-$ relation. These equations allow the calculations of the distribution of particles originating from $L(G)_\pm$ sources. Practically, a weight that is a plus (minus) number corresponding to $L(G)_\pm$ is assigned to each particle.

The process of adding particles in modeling of source term results in a continuous increase of the number of particles during the simulation. Note that the eqs. (2.7a) and (2.7b) have the same stationary solutions regardless of their initial conditions. So, we expect solutions with similar shapes approaching each other, which causes an overlap of $\delta f_\pm$ as $\lim_{t\to\infty}\left(\delta f_+ - \delta f_-\right)/\left(\delta f_+ + \delta f_-\right) = 0$. The overlap can deteriorate the accuracy of numerical method when a large fraction of particles cancels locally. A straightforward way of removing the overlap is to find approximations of $\delta f_\pm$ by projecting particles on an interpolating base e.g., finite element methods, and then resampling new particles representing the distribution. However, that would not be a suitable method for multidimensional problems. A new algorithm for reducing the number of particles based on resampling of particles using moments of distributions is suggested in next section.

3. Particle annihilation in δf method

3.1 Ball-resampling algorithm

The aim of the ball resampling is to limit the number of particles while the global distribution is preserved. So, it can solve the problem with the overlap of δf_{+} and δf_{-} during the simulation. Moreover it reduces the computational work. The basic idea is to separate the set of particles to small subsets, and then replacing particles at each subset with fewer number of particles while the local moments of distributions are conserved [7]. The subset of particles could be selected with a ball traversing the set of particles as it's shown schematically in fig. 3.1.

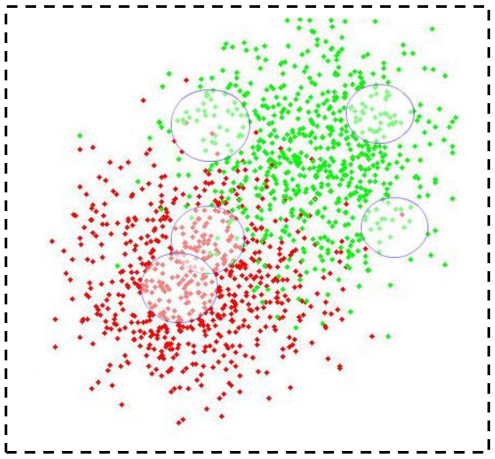


Figure 3.1: Traversing the set of particles with query ball.

Without considering details, the general algorithm is defined as follows. Note that a weight w_i associated to each particle p_i is defined.

Step1: select a set of particles $\{(p^{old}, w^{old})\}$ around an arbitrary particle (p_c, w_c) in the phase space where the overlap is expected to be large.

Step2: sample new set of particles from a normal distribution, $\{p_j^{new}\} \approx n(p_c)$.

Step3: calculate new weights w_c^{new} for p_j^{new} such that the new particles at query ball satisfy some conditions.

Step4: Continue this operation until sufficiently many particles have been resampled.

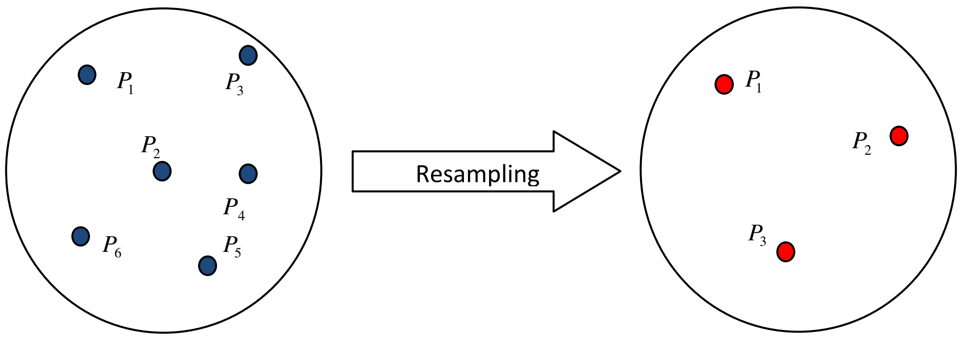


Figure 3.2: The resampling in a typical ball; the initial particles are replaced by fewer number of new particles so that some conditions (eqs. 3.1 a-c) are maintained.

Two different approaches for particle query in an initial set of particles are considered; a query ball with constant radius or a query ball that is containing constant number of particles. In step 2, all selected particles in step 1 $\{(p^{old}, w^{old})\}$ are removed and new particles are re-sampled. The new particles are selected from a normal distribution or just randomly from the old particles. The weight coefficients are then chosen so that the moments of distribution are preserved. For example, consider a one dimensional case:

$$\sum_{k=1}^{m} w_k^{new} = \sum_{i=1}^{n} w_i^{old} \tag{3.1a}$$

$$\frac{1}{m}\sum_{k=1}^{m} w_k^{new} p_k^{new} = \frac{1}{n}\sum_{i=1}^{n} w_i^{old} p_i^{old} \tag{3.1b}$$

$$\frac{1}{m}\sum_{k=1}^{m} w_k^{new} (p_k^{new} - \langle p^{new} \rangle)^2 = \frac{1}{n}\sum_{i=1}^{n} w_i\, (p_k^{old} - \langle p^{old} \rangle)^2 \tag{3.1c}$$

where n and m denote the number of particles in a query ball before and after resampling, respectively. The number of variables is equal to number of new particles in the ball m while only three equations exist. So, the set defines an underdetermined system of equations for the weight w_k^{new}. The adjustment and improvement of this part of algorithm is explained in section 4. Finally, the steps1-3 should be repeated until sufficiently many particles have been resampled, step 4.

In the next section, the kd-tree method that is providing a fast algorithm for finding the nearest neighbor particles around a query particle is explained. Finding nearest neighbors is used in step 1 in the algorithm. The methods of reconstruction of distribution function, and the concept of moments of distribution that would be useful in analyzing the results and understanding step 3 in the algorithm, are also clarified in next sections.

3.2. Nearest neighbors particles around a query particle; kd-tree

Let us consider a database of particles in a 2-dimensional Cartesian space with coordinates (P_x, P_y) where each point represents a particle. We search in the database to find all the particles inside a circle with radius r and centre in a query particle that is called a range search. In Cartesian coordinate system, the distance between two particles is calculated from

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \, .$$

Evidently, a simple binary search through the whole data set is not the most efficient algorithm. The kd-tree, short for k-dimensional tree, is a popular way of querying multidimensional databases. It has the advantage that is easy to implement and also provide a simple algorithm for the closest particle or range query, and foremost it is fast.

In a one dimensional array, splitting the original data to make the kd-tree is straightforward: first, the median value is found with querying into the array. Afterwards, the array splits into two subset of equal size whereas one subset contains the particles smaller than the median value, and the other subset contains the particles larger than the median value. The median value is sorted in the root variable, and stored recursively in the two subtrees. This procedure is repeated.

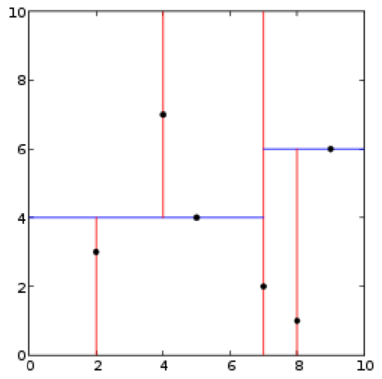


Figure 3.3: Building a kd-tree for a small set of points.

The simple one dimensional algorithm is not applicable in a two-dimensional case, because every particle has an x, and an y value. This problem is solved by splitting on each coordinate separately [8]: first split on the $x-$coordinate and next on the $y-$coordinate, then again on x and so on. This is shown schematically in fig. 3.1.

The set P splits at its median value (in x) with red line into two subsets of roughly equal size, then the right and left subsets, P_r and P_l, are stored in right and left sub-trees. At the second level using the median value in y, the left subset P_l is split into two subsets with horizontal blue line. The particles below or on it are stored in left subtree of left child, and the particles above the line are stored in right subtree. A $2-d$ kd-tree is constructed by continuing this procedure.

The range query algorithm could be written using kd-tree. Generally, the region of a typical node ν, $region(\nu)$, is a rectangle which is bounded by ancestor of ν and is unbounded on one or more sides.

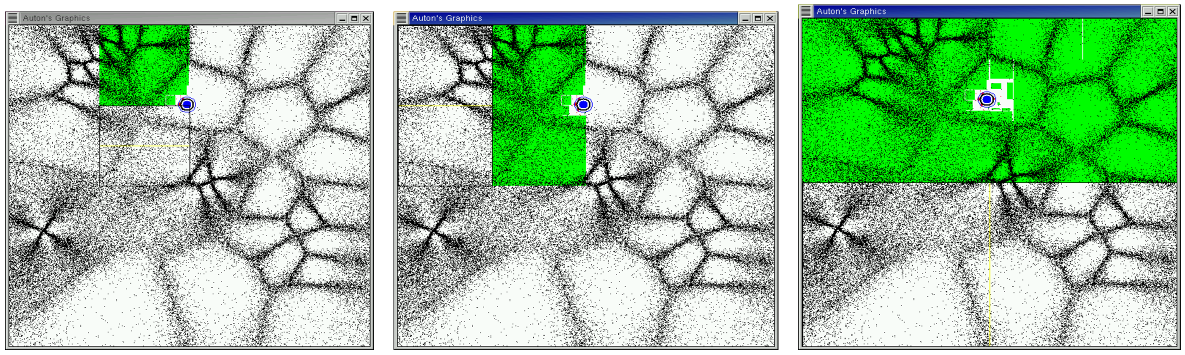


Figure 3.4: animation of range-search on a large data base (dark points in figure); the query region is the small blue sphere. The main advantage is that we do not need searching in green rectangulars.

We need to search subtree ν only if the query region intersects $region(\nu)$. The query algorithm could be described as: the kd-tree is traversed, but notice only nodes whose region is intersected by query region. If a region is completely inside the query region, its entire particle is reported. But if the traversal reaches a leaf, just in case that particle stored at the leaf is inside the query region it should be reported. The green rectangles in fig. 3.4 denote nodes, which are not investigated because they are not intersected by query region.

3.3. Reconstruction of a distribution function; Kernel density estimation

The construction of distribution function from a set of particles would be used in evaluating the accuracy of resampling method (see section 4). A histogram is the simplest method of estimating a density or probability distribution of a set of data. To construct a histogram, the interval covered by the data values is divided into equal sub-intervals and then we only need to depict the frequency of particles occurring in each interval. As it could be seen in fig. 3.5, histogram is not a smooth diagram, and a more advanced method such as the kernel density estimation is used to alleviate this drawback.

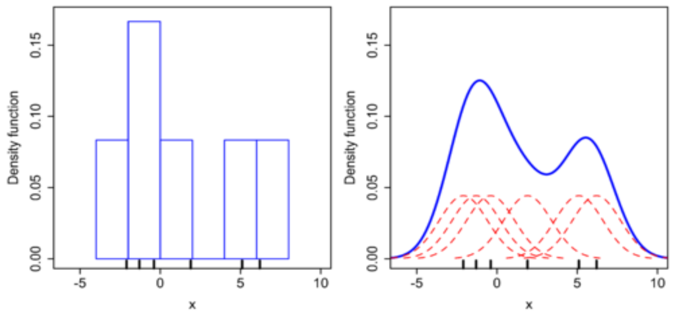


Figure 3.5: comparison of kernel density estimation and histogram as two different methods producing distribution function.

A kernel estimator centers a kernel function K at each data particle [9]. Consider a set of data $(x_1, x_2, ..., x_n)$ drawn from a distribution function f .

Kernel function	$K(x)$
Uniform	$\frac{1}{2} I(\lvert x\rvert \le 1)$
Triangle	$(1-\lvert x\rvert) I(\lvert x\rvert \le 1)$
Gaussian	$\frac{1}{\sqrt{2\pi}} \exp(-\frac{1}{2} x^2)$
Epanechnikov	$\frac{3}{4}(1-x^2) I(\lvert x\rvert \le 1)$
Cosinus	$\frac{\pi}{4}\cos(\frac{\pi}{2} x) I(\lvert x\rvert \le 1)$

Table 3.1: Several types of kernel functions are commonly used in reconstruction of distribution function.

The unknown function f in $d-$dimension is estimated as

$$\hat{f}(\vec{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left\{\frac{1}{h}(\vec{x} - \vec{x}_i)\right\} \tag{3.2}$$

where $h > 0$ is a smoothing parameter that is called window width, and K is a kernel function satisfying the following condition

$$\int_{R^d} K(\vec{x}) d\vec{x} = 1$$

Several kernel functions are usually used as shown in table (3.1). The window width h is more important than the shape of Kernel function in creating a high quality density function. Small value of h makes very spiky estimate while large value leads to over-smoothing. Therefore, a trade-off between these two limits is demanded.

3.4. Moments of distribution; Mean, Variance, Skewness

When a set of values has a tendency to cluster around some particular value, then it may be useful to characterize the set by a few numbers related to its moments. Moment is a suitable quantity for describing the shape of distribution function of the values. The $i-th$ moment μ_i of the continuous function $f(x)$ depending a one-dimensional internal coordinate x (typically, a length scale) is classically defined by

$$\mu_i = \int_0^\infty x^i f(t,x)dx, \quad i = 0,1,2,.. \tag{3.3}$$

which is called an un-centered moment and for a centered moment about x_0

$$\mu_i = \int_0^\infty (x - x_0)^i f(t,x)dx, \quad i = 0,1,2,... \tag{3.4}$$

The function f is usually the probability density function, PDF. According to the definition, the integral of a PDF over a range of possible values gives the probability of the random variable falling within that range. So, if the probability distribution of random variable x admits a PDF then the first moment of a random variable is equivalent to the mean value or expected value of the variable.

For a discrete random variable x, the $i-$moment is defined as

$$\mu_i = \sum_j x_j^i p(x_j) \ , \tag{35}$$

where p is probability mass function, PMF, presenting the probability that a random variable x is exactly equal to some value (x_j). Note the difference between PMF and PDF.

The statistics of a random variable could be estimated if we repeat the experiment, which generates the variable a large number of times. If the experiment is run N times, then each value of x will occur $Np(x)$ times, thus

$$\mu_i = \frac{1}{N}\sum_{j=1}^{N} x_j^i \tag{3.6}$$

Therefore, the first moment would be equal to the mean value, $\langle x \rangle = \frac{1}{N}\sum_j x_j$, and for the second moment

$$\mu_2 = \frac{1}{N}\sum_{j=1}^{N} x_j^2 \ . \tag{3.7}$$

Note that the variance, $Var(X) = \frac{1}{N-1}\sum_i (x_i - \langle x \rangle)^2$, is almost equivalent to the centered second moment of a random variable. The variance in one dimension is a measure of variability or width around the mean value, and in higher dimensions it is a measure of the shape of a cloud of particles as it could be fit by an ellipsoid.
The third moment of distribution describes the skewness that characterizes the degree of asymmetry of a distribution around its mean. While the first and second moments of distributions have the same units as the measured quantities x_j, the skewness is conventionally defined in such a way to make it dimensionless. It is a pure number that characterizes only the shape of the distribution as it's shown in fig. 3.6.

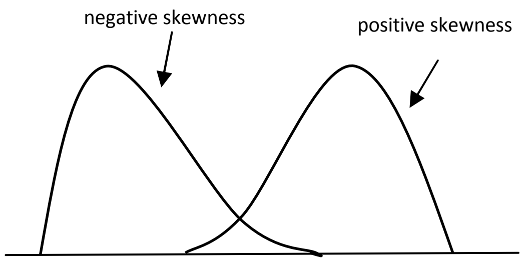


Figure 3.6: The negative and positive skewness for a distribution

The definition of skewness is

$$Skew(x_1,...,x_N) = \frac{1}{N}\sum_{j=1}^{N}\left[\frac{x_j - \langle x\rangle}{\sigma}\right]^3 ,$$

where σ is the standard deviation, $\sigma(x_1,...,x_N) = \sqrt{Var(x_1,...,x_N)}$. A positive value of skewness shows a distribution with an asymmetric tail extending out towards more positive x and a negative value shows a distribution whose tail extends out towards more negative x , see fig. 3.6.

4. Results and discussions

The result of simulation including calculation of the discrepancy between distribution functions before and after resampling is presented. In first section, the error from kernel density, which is used for reconstruction of distribution function from set of particles, is investigated. In next sections, the magnitude of errors from comparison of distribution functions before and after resampling for different parameters is reported.

4.1. Evaluating the accuracy of method

The kernel method is used for reconstruction of a distribution in one dimension. A brief description of the method was presented in previous section. The accuracy of the method could be evaluated by comparing the distribution function of a set of particles f_{old} drawn from an arbitrary function f with the function f. We consider the following function

$$f = e^{-x^2/2} - (e^{-(x-2)^2/2}), \tag{4.1}$$

which is similar to the solution of eqs. (2.7a) and (2.7b) with particle weights $+1$ and -1 for $L(G)_+$ and $L(G)_-$ sources, respectively. The discrepancy between f_{old} and f is calculated using relative second norm error $RN2 = \frac{\|f_{old} - f\|_2}{\|f\|_2}$ that is shown in fig. 4.1 as a function of the number of particles N. The second norm is denoted with $\| \ \|_2$ that is defined as $\|x\|_2 = \left(\sum_i x_i^2\right)^{1/2}$ for variable x. The slope of solid line in fig. 4.1 is equal to $-1/2$ corresponding to Monte-Carlo sampling error, $O\left(1/\sqrt{N}\right)$, and it could be seen that the error decreases as $1/\sqrt{N}$.

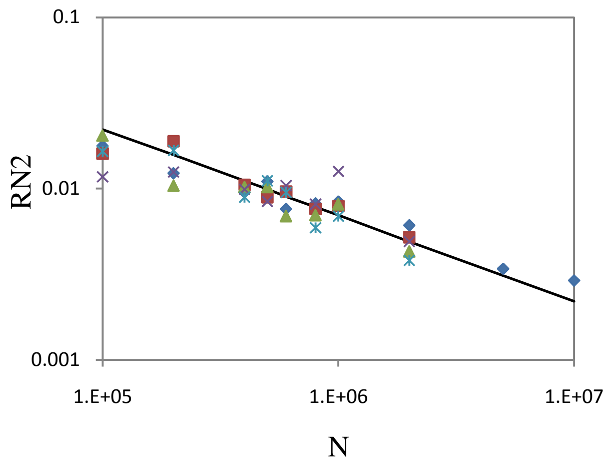


Figure 4.1: The Relative second norm error calculated from discrepancy between the exact function f and reconstructed distribution function using kernel density estimation. The slope of the solid line is $-1/2$ representing the behavior of sampling error.

The accuracy of method is evaluated by comparing the distribution function of resampled particles f_{new} by the initial set of particles f_{old}. In evaluating the accuracy of resampling method, two kinds of error should be noticed; the error from kernel density estimation, and the statistical error due to the finite number of particles. The total values of these errors are at least equal to the value as shown in fig. 4.1. Practically, we calculate the error by comparison of the reconstructed distribution function before, f_{old}, and after , f_{new}, resampling using the relative second and maximum norm errors

$$RN2 = \frac{\|f_{old} - f_{new}\|_2}{\|f_{old}\|_2} \text{ and } RNM = \frac{\|f_{old} - f_{new}\|_{\max}}{\|f_{old}\|_{\max}}, \tag{4.2}$$

where $\| \ \|_{\max}$ shows the maximum error and is defined as $\|x\|_{\max} = \max|x_i|$. Therefore, the reported value of error is an overestimate the error for resampling method because it unavoidably contains errors from the kernel density and the particle sampling.

4.2. Different approaches of resampling

The basic resampling algorithm was discussed in previous section. As it's mentioned, the algorithm is flexible and negotiable especially in resampling method, step 3. The numerical calculations show that the result is sensitive to the form of moments that is used in step 3, and also to the type of query ball, with a fixed-radius or fixed-number of particles. In next sections the result for several different conditions is presented.

4.2.1. First approach

The moments of distribution are useful quantities for evaluating the total distribution function. So, the first idea in reconstructing the same statistics is to keep the same moments through the resampling process, step 3 of the algorithm. In the first approach we use eqs. 3.1 to define the resampling rules with a little change in third line, 3.1c; using an un-centered second moment

$$\frac{1}{m}\sum_{k=1}^{m} w_k^{new}(p_k^{new})^2 = \frac{1}{n}\sum_{i=1}^{n} w_i\,(p_i^{old})^2$$

instead of a centered second moment. In that case, the conservation of total weight, and the two first moments of the distribution after the resampling is expected, if we consider a restriction on the number of sampled new particles in each ball. It is explained as: The total first-moment is conserved if

$$\frac{1}{M}\left[\sum_{k=1}^{m^1} w_k^{new}\, p_k^{new} + \sum_{k=1}^{m^2} w_k^{new}\, p_k^{new} + \dots\right] = \frac{1}{N}\left[\sum_{i=1}^{n^1} w_i^{old}\, p_i^{old} + \sum_{i=1}^{n^2} w_i^{old}\, p_i^{old} + \dots\right],$$

where N and M represent the total number of the initial and the new particles, respectively. The variables with upper indexes, n^b and m^b, denote the numbers of the initial and the new particles in each ball. The first moment (and similarly second moment) is conserved if $M / N = n^b / m^b$.

The simulation confirms that the total first and second moments do not change after resampling. However, the estimated distribution is not acceptable. The magnitude of error calculated from eqs. (4.2) is too large that the function f_{new} could not be considered as an estimation of f_{old}. In other words, considering the first moment and the un-centered second moment, at least for the distribution function such as f, one could not desirably reproduce the distribution function.

4.2.2 Second approach

In the second approach, we considered the following set:

$$\sum_{k=1}^{m} w_k^{new} = \sum_{i=1}^{n} w_i^{old} \tag{4.3a}$$

$$\sum_{k=1}^{m} w_k^{new}\, p_k^{new} = \sum_{i=1}^{n} w_i^{old}\, p_i^{old} \tag{4.3b}$$

$$\frac{1}{m}\sum_{k=1}^{m} w_k^{new}(p_k^{new} - \langle p^{new}\rangle)^2 = \frac{1}{n}\sum_{i=1}^{n} w_i\,(p_i^{old} - \langle p^{old}\rangle)^2 \tag{4.3c}$$

Fig. 4.2 shows three distribution functions from the exact solution f, and the reconstructed distribution before f_{old} and after f_{new} resampling illustrating the matching of the estimated distribution with the exact function.

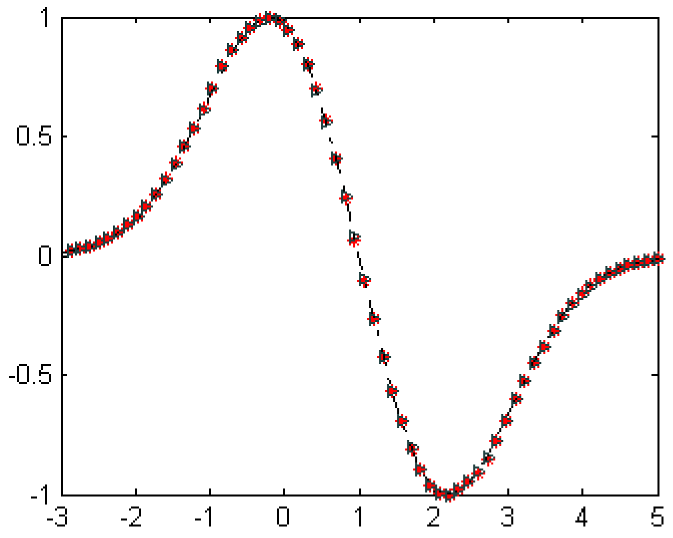


Figure 4.2: Comparison of three distribution functions; function (4.1), and distribution functions before f_{old} and after f_{new} resampling.

In figure 4.3 the results of simulation, RN2 and RNM errors, as a function of ball radius for a total number of particles $N = 5 \cdot 10^5$, and 9 batches are presented. The number of annihilated particles is almost equal to $N = 10^5$, or in other words the ratio of annihilated particles is almost equal to $\beta = 1/5$.

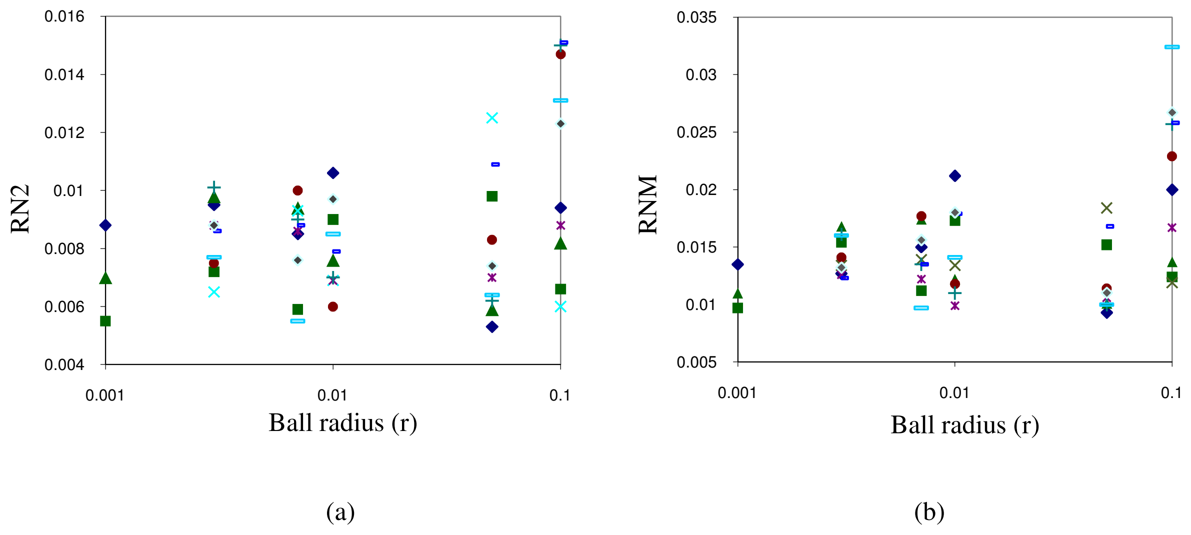


Figure 4.3: the a) Relative second norm error, and b) Maximum second norm error as a function of ball radius for $\beta = 1/5$ and $N = 5 \cdot 10^5$.

The spread of the results for the different batches is due to statistical error. The maximum value of errors decreases toward the smaller radius whereas the RN2 error is less than 0.01 at $r = 0.001$ in this figure. Note that the magnitude of error from the kernel density estimation at

$N=5\cdot10^5$ is almost equal to 0.01, fig. 4.1. The magnitude of the error in $r=0.1$ approaches 0.015 demonstrating an error arising from the resampling algorithm at this radius.

4.2.3. Third approach

In the third approach, the following set is considered

$$\sum_{k=1}^{m} w_k^{new} = \sum_{i=1}^{n} w_i^{old} \tag{4.4-a}$$

$$\sum_{k=1}^{m} w_k^{new} p_k^{new} = \sum_{i=1}^{n} w_i^{old} p_i^{old} \tag{4.4-b}$$

$$\sum_{k=1}^{m} w_k^{new} (p_k^{new})^2 = \sum_{i=1}^{n} w_i (p_i^{old})^2 \tag{4.4-c}$$

The coefficient $1/n$ $(1/m)$ that is defining a localized quantity at each ball is neglected in this new set. The result of the simulation for $N=2\cdot10^5$ and the ratio of annihilated particles $\beta=1/5$ is shown in fig 4.4. The number of particles at each ball, N_b, is fixed rather than the ball radius. A minimum value at $N_b=300$ is distinguishable especially for RN2 error. Note that the magnitude of the error in the range $100\le N_b\le 6000$ is less than the error from kernel density estimation for $N=2\cdot10^5$ that is equal to 0.02, but it grows at large values of N_b so that it reaches to 2-3 times of the minimum value for $N_b=10000$. The raise in error at the left side of the diagram is due to the low number of particles at each cell causing a large statistical error.

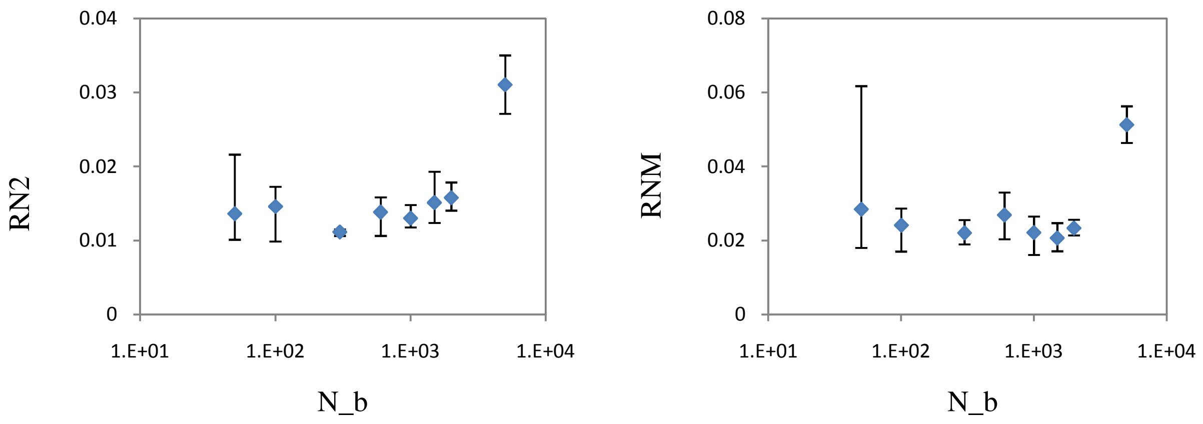


Figure 4.4: the values of RN2 and RNM errors versus the number of particle inside query ball N_b for $N=2\cdot10^5$. The maximum and minimum error for 5 batches is illustrated with a bar around the average value at each point in this figure.

The same result for a larger number of particles $N=5\cdot10^5$ is shown in fig. 4.5. The restriction in processing time does not allow running more batches, but evidently the magnitude of error decreases relative to previous case for $N = 2 \cdot 10^5$. The average value of RN2 error in $N_b = 10^4$ is equal to 0.03, that is larger than the kernel density error 0.01.

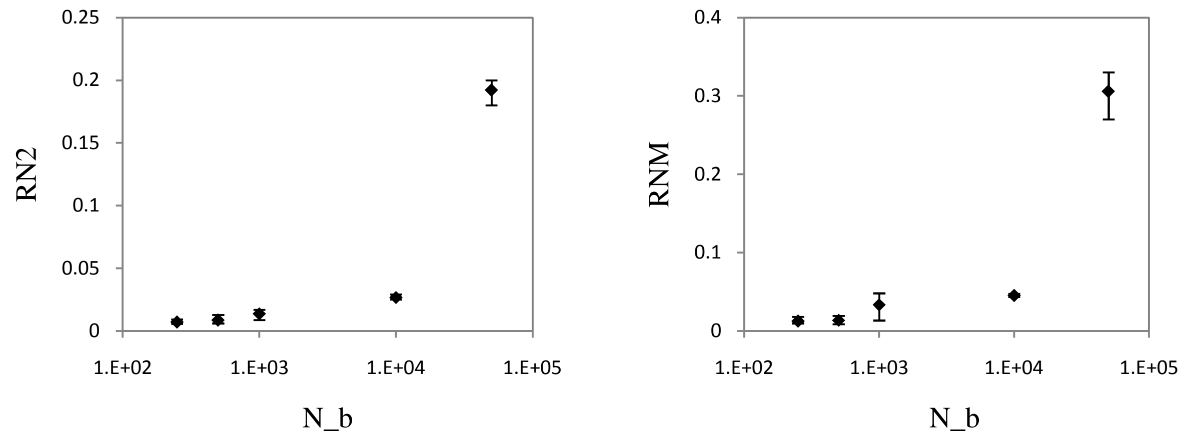
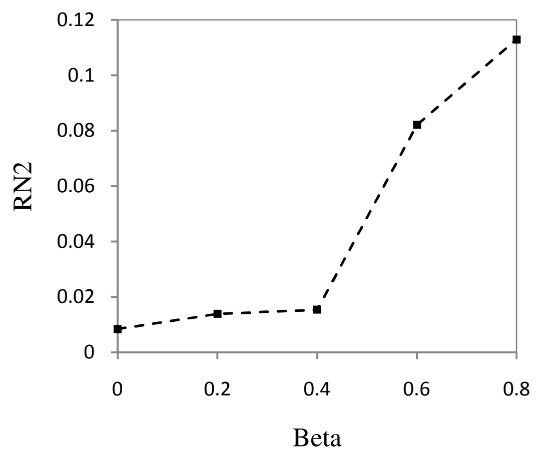


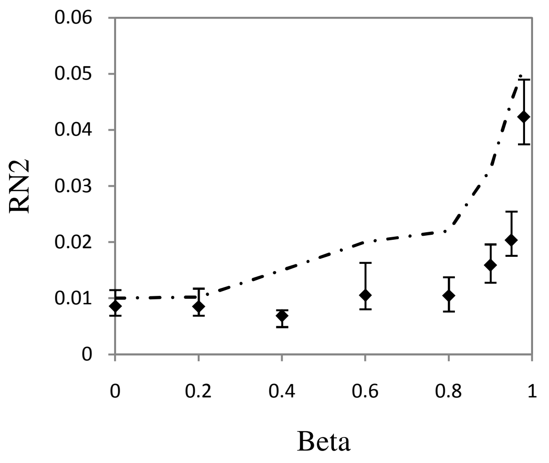
Figure 4.5: the same results as previous figure for $N = 5 \cdot 10^5$.

The magnitude of the error at $N_b = 250$ in fig. 4.5 is less than 0.01 corresponding to the results for $r = 0.001$ in fig. 4.3. The number of balls that is used to cover all the particles is a good criterion in estimating the processing time, and it could be calculated exactly in fixed-number of particle algorithm, $s \approx N / N_b = 2000$. This number in a fixed-radius algorithm could be estimated as: $s \approx \Delta x / r = 7 / 0.001$, where Δx is the range of variation of the particles (fig. 4.2). Therefore the processing time in latter case is almost 3 times larger than the former.

4.3. Sensitivity to the annihilation fraction

The principle purpose of resampling is to reduce the total number of particles while the distribution function is almost maintained identical. The magnitude of error as a function of the ratio of annihilated particles β is plotted in figs. 4.6a and 4.6b for fixed number of particles in a ball and fixed ball radius, respectively. In fig. 4.6a, the number of particles inside a query ball is $N_b = 1000$ and the total number of particles is $N = 5 \cdot 10^5$. The magnitude of error in $\beta = 0$ as expected is equal to the error from kernel density that is $RN2 \approx 0.01$, but it increases at larger values of β mainly due to the deviation at the tail of distribution function. The value of error even at $\beta = 0.4$ is $RN2 \approx 0.02$, which is larger than the error from kernel density proving the dominant error from resampling. The large magnitude of error in the tail is due to the low density of particles in the tail of the distribution, fig. 4.2. Therefore, the method of splitting the set of particle to bunches with the same number of particles does not turn out acceptable results at least for this kind of distribution and at large values of β. We need to query the set of particle with smaller bunches of particle at the tail or leave them without resampling, thus the fixed radius ball method should afford better results.





(a) (b)

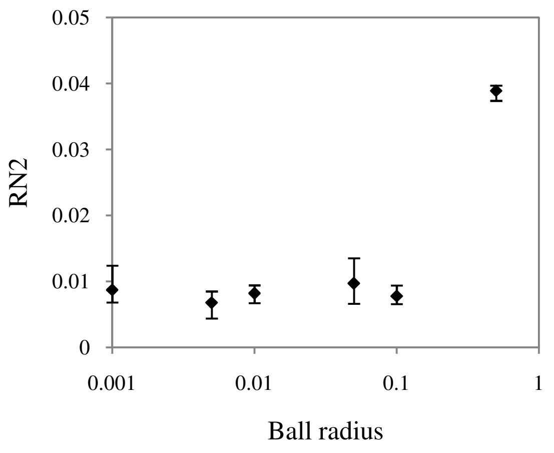
Figure 4.6: The RN2 error as a function of annihilation ratio β for a) fixed number of particle in ball, and b) fixed ball radius algorithms.

Fig. 4.6b shows the RNE error for a wide range of $0 \le \beta \le 0.98$ at a fixed ball radius $r = 0.01$. The number of particles after the resampling decreases resulting in larger sampling errors. The dashed line shows the corresponding kernel density error. The magnitude of the error is less than the error from the kernel density that is scaled as $O(1/\sqrt{N})$. It proves the insensivity of method to the fraction of annihilated particles in the range $0 \le \beta \le 0.98$ for fixed ball radius method.

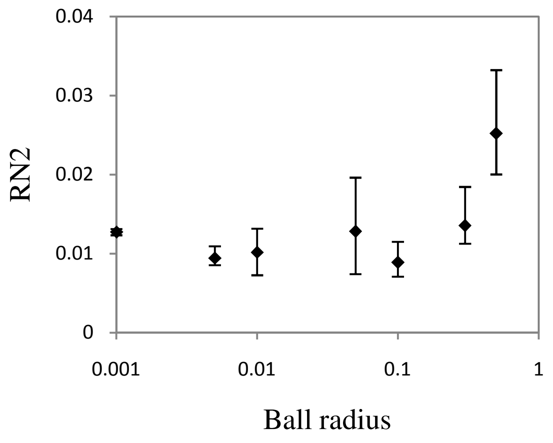
4.4. Sensitivity to the ball radius

The computational work increases at smaller values of ball radius, thus it would be useful to know the dependency of error on ball radius and use larger values of radius as long as the magnitude of error is acceptable. Fig. 4.7 shows the RN2 error as a function of ball radius for two different values of beta $\beta = 1/2$ and $8/10$. The number of particles is equal to $N = 5 \cdot 10^5$ in both figures.

The magnitude of error at $r = 0.5$ is equal to 0.04 in both figures, which is larger than the error from kernel density estimation. However, it decreases to less than the kernel density error for $r < 0.1$.



(a)



(b)

Figure 4.7: The magnitude of RN2 error for two different values of β, a) $1/2$, and b) $8/10$ as a function of ball radius.

Note that the kernel density error for $N = 10^5$ corresponding to $\beta = 8/10$ is equal to 0.02, which can satisfactory explain the larger error in right hand figure relative to the left hand figure. Although, the results do not show any strong dependency of the error with respect to the ball radius, we cannot conclude that the algorithm is insensitive to ball radius. For instance, in fig. 4.7a, a minimum value of error in $r = 0.005$ is distinguishable.

4.5. The accuracy of the method

Figs. 4.6 and 4.7 demonstrate that for $N = 5 \cdot 10^5$ the error from resampling is less than the error from the kernel density, that is scaling as $1/\sqrt{N}$, in a vast range of β and ball radius r. This behavior is recognized in fig. 4.8 that is showing the error in the range $N = 5 \cdot 10^3 - 5 \cdot 10^5$, and for $\beta = 1/2$ and $r = 0.1$.

The spreading of particles in fig 4.8 decreases at larger values of N as it is expected in sampling error. Therefore, the discrepancy between distribution functions before and after resampling is determined with number of sampled particles.

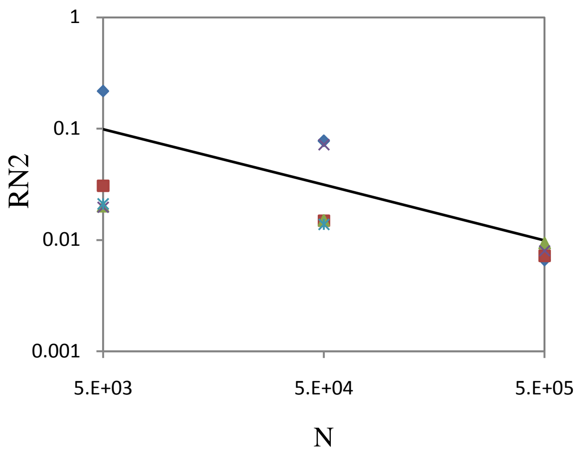


Figure 1: The relative second norm error as a function of particle number for $\beta = 1/2$ and $r = 0.1$. The slope of the solid line is $-1/2$.

The magnitude of relative second norm error at $N = 1 \cdot 10^6$ is of the order $O(10^{-3})$, and interpolation of solid line results in $RN2 = 2 \cdot 10^{-4}$ at $N = 1 \cdot 10^9$.

5. Conclusion

A new method for annihilating a fraction of particles useful in δf Monte-Carlo simulation has been developed. The proposed method relies on conserving the moments of distribution locally, inside query ball, within the collection of particles so that the total distribution function is maintained. The query-ball traverses the collection of particles using a fast algorithm based on a kd-tree method for finding the nearest neighbors, and replaces the new particles with old particles. The accuracy of the method was evaluated by comparing the distribution functions before and after resampling, where the distribution functions are constructed using kernel density estimation method.

The numerical calculation shows the strong dependency of results on the form of moments and the query balls with fixed-radius or fixed number of particles. The results for a fixed-radius query ball, and a global form of moments show the best agreement, i.e., the discrepancy between the distribution function before and after resampling is scaled as the Monte-Carlo error $O\left(1/\sqrt{N}\right)$ and is less than the error form the kernel density estimation for a wide range of annihilation factors and ball radii demonstrating the dominant role of sampling error rather than the error from resampling algorithm. For instance, the magnitude of the relative second norm error at $N = 1 \cdot 10^6$ is of the order $O\left(10^{-3}\right)$. An important conclusion is the insensivity of method to the annihilation ratio in a wide range $0 \le \beta \le 0.98$ permitting annihilation of particles efficiently.

Acknowledgment

I want to thank my supervisor Prof. Torbjörn Hellsten for his inestimable help I have received in writing this thesis. His advices and criticism has been indispensable in preparing the final form of the text. I also must acknowledge Josef Höök for his support and patience to walk me through this work, and his assistance at every step of the thesis. I take this opportunity to thank Mrs. Cecilia Forssman for her sympathetic help in administrative work.

References

[1] J. Goodman et al., “Stochastic and partial differential equations with adapted numerics”, a free manuscript released on May 2, 2006.

[2] R. E. Denton, and M. Kotschenreuther, “δf algorithm”, J. Comput. Physics, Vol. 119, p. 283.

[3] Y. Chen, and R. b. White, “collisional δf method”, Phys. Plasmas, Vol. 4, p. 3591.

[4] L. J. Höök, and T. Hellsten, “An adaptive δf Monte-Carlo method”, IEEE Trans. Plasma Sci., Vol. 38, p. 2190.

[5] T. H. Stix, “Waves in Plasmas”, American Institute of Physics, 1992.

[6] P. l. Kloeden, and E. Platen, “Numerical solution of stochastic differential equations”, Springer, second printing 1995.

[7] J. Höök, “An adaptive δf Monte-Carlo method”, presented in Division of plasma physics, KTH, November-2009.

[8] A. Moore, “An introductory tutorial on kd-tree”, Phd Thesis, University of Cambridge, 1991.

[9] B. W. Silver, “Density estimation for statistics and data analysis”, Chapman & Hall, 1986.