Gridless DSMC

Spencer E. Olson

FOCUS Center, Physics Department, University of Michigan
450 Church Street, Ann Arbor, Michigan 48109-1120 [1]

Andrew J. Christlieb

Mathematics Department, Michigan State University
D304 Wells Hall, East Lansing, Michigan 48824-1027

Abstract

This work concerns the development of a gridless method for modeling the inter-particle collisions of a gas. Conventional fixed-grid algorithms are susceptible to grid-mismatch to the physical system, resulting in erroneous solutions. On the contrary, a gridless algorithm can be used to simulate various physical systems without the need to perform grid-mesh optimization. An octree algorithm provides the gridless character to a direct simulation Monte Carlo (DSMC) code by automatically sorting nearest-neighbor gas particles into local clusters. Automatic clustering allows abstraction of the DSMC algorithm from the physical system of the problem in question. This abstraction provides flexibility for domains with complex geometries as well as a decreased code development time for a given physical problem. To evaluate the practicality of this code, the time required to perform the gridless overhead from the octree sort is investigated. This investigation shows that the gridless method can indeed be practical and compete with other DSMC codes. To validate gridless DSMC, results of several benchmark simulations are compared to results from a fixed-grid code. The benchmark simulations include several Couette flows of differing Knudsen number, low-velocity flow past a thin plate, and two hypersonic flows past embedded objects at a Mach number of 10. The results of this comparison to traditional DSMC are favorable. This work is intended to become part of a larger gridless simulation tool for collisional plasmas. Corresponding work includes a gridless field solver using an octree for the evaluation of long range electrostatic forces. We plan to merge the two methods creating a gridless framework for simulating collisional-plasmas.

Key words: Direct simulation Monte Carlo, DSMC, octree, gridless, meshless
PACS: 02.50.Ey, 02.70.Tt

1 Introduction

Simulations of gas systems have been useful for understanding many practical problems: space shuttle damage, aerodynamics of craft, and so on. This paper describes a gridless procedure for handling collisions in a direct simulation Monte Carlo (DSMC) calculation. This new method enables shortened development time for studying a specific physical system. It also shows promise for increased accuracy for some very rarefied systems as well as systems with high-gradient flows without a heavy cost increase in computational resources (time, processor power, memory, etc.) required. Following a brief context for simulating gas dynamics, the method is described in detail by springboarding from a brief review of the more traditional and established grid-based DSMC.

Under the assumption that inter-particle interactions in a gas are dominated by binary collisions, the evolution of the gas can be accurately modeled by the Boltzmann equation, given by

$$\frac{\partial f}{\partial t} + [f, H] = \mathcal{Q}(f, f) \tag{1}$$

with initial condition

$$f(\vec{\mathbf{x}}, \vec{\mathbf{p}}, 0) = f_0(\vec{\mathbf{x}}, \vec{\mathbf{p}}) \ . \tag{2}$$

$f(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$ in Eq. (1) describes a phase-space distribution for the gas over all space and momentum ($\vec{\mathbf{x}} \in D \subset \mathbb{R}^3,\ \vec{\mathbf{p}} \in \mathbb{R}^3$) and H is Hamiltonian of the system. The operator $\mathcal{Q}(\cdot, \cdot)$ is a convolution over all phase space that accounts for departure from the Liouville theorem due to collisions between particles in the system [1–3].

In general, gas systems are characterized by the ratio of the mean free path λ_{MFP} to the characteristic length L of the system. λ_{MFP} is given by

$$\lambda_{\mathrm{MFP}} = \frac{1}{\sqrt{2}\,\sigma_{\mathrm{T}}\,n} \tag{3}$$

where σ_{T} is the total elastic scattering cross-section and n is the local number density. The local characteristic length L of a system is the length scale over which a macroscopic property S of the gas, e.g. temperature or density, undergoes significant change and can be expressed as

$$L = S\left(\frac{\partial S}{\partial x}\right)^{-1} \ . \tag{4}$$

Email address: Spencer.E.Olson@umich.edu (Spencer E. Olson).
URL: http://www.umich.edu/~olsonse/ (Spencer E. Olson).
[1] Now at Naval Research Laboratory, Optical Sciences Division, 4555 Overlook Ave. SW, Washington D.C. 20375

The ratio $(\lambda_{\mathrm{MFP}}/L)$, called the local Knudsen number and denoted by Kn, is a measure of the collisionality of a gas. The relative importance of the collision integral in Eq. (1) is directly tied to the value of Kn. As Kn tends to $\ll 1$, action in the gas is dominated by collision events. For $Kn \ll 1$, the solutions of Eq. (1) can be approximated as

$$f_{\mathrm{M}} = \frac{n\left(\vec{\mathbf{x}},t\right)}{N}\left(\frac{\pi}{2mk_{\mathrm{B}}T}\right)^{3/2} exp\left\{-\frac{\left|\vec{\mathbf{p}}-\vec{\mathbf{p}}_0\left(\vec{\mathbf{x}},t\right)\right|^2}{2mk_{\mathrm{B}}T\left(\vec{\mathbf{x}},t\right)}\right\}, \tag{5}$$

where f_{M} is termed a local Maxwellian. In this limit, one can employ the Chapman–Enskog method to derive transport equations for number density $n\left(\vec{\mathbf{x}},t\right)$, stream velocity $\vec{\mathbf{p}}_0\left(\vec{\mathbf{x}},t\right)/m$, and temperature $T\left(\vec{\mathbf{x}},t\right)$ [1]. Thus, Kn serves as a guide for determining the most efficient approach for solving Eq. (1) for a gas system.

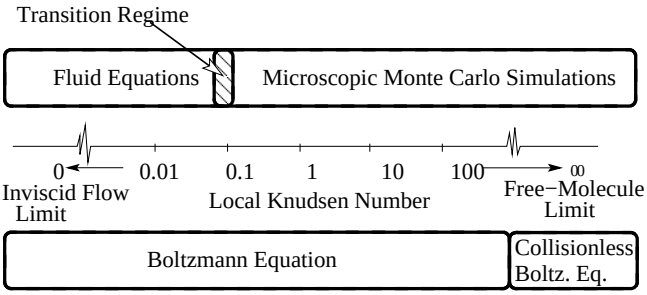

Fig. 1. Validity regions for various gas simulation approaches. For very low Knudsen number Kn, fluid equations, such as the Navier-Stokes equations are typically used. As $Kn \to 0$, the solutions of the fluid equations approach those of incompressible, inviscid solutions of the Euler equations. For $Kn \gtrsim 0.1$, statistical simulations are often used to model microscopic interactions. As $Kn \to \infty$, deterministic free–molecular motion is the limit.

Fig. 1 shows the range of validity for different models of particle dynamics as a function of Kn. As indicated in Fig. 1, there exists a cutoff region beyond which the fluid model does not extend. A wide range of important physical systems fall outside the scope of fluid models in this higher range of Kn. Examples of high Kn systems ($Kn > 0.1$) include (but are not limited to) comet tails [4], spacecraft reentry [5], spacecraft plume interactions [6], micro-/nanoscale gas flows [7–9], evaporative cooling for the formation of Bose–Einstein Condensate [10, 11], and fundamental processes in plasma etching systems [12–14]. Often these systems are far too complex to construct analytic solutions to the governing equation (Eq. 1). Hence, theoretical studies of these systems are mostly limited to numerical simulation and computation. Unfortunately, for most high Kn problems of interest, directly solving Eq. (1) is not computationally tractable. This is because of the excessive computational resource requirement needed to describe and calculate the phase-space distribution function $f\left(\vec{\mathbf{x}},\vec{\mathbf{p}},t\right)$. For example, a six-dimensional $f\left(\vec{\mathbf{x}},\vec{\mathbf{p}},t\right)$ over a spatial grid of $100\times100\times100$ and a minimal momentum grid of $30\times30\times30$ requires $\gtrsim 100$ GB of memory just for basic storage.

A popular approach for reducing the computational requirement is to use a statistical method to evaluate at least part of Eq. (1). The most common statistical method, pioneered by G. A. Bird [3], is direct simulation Monte Carlo (DSMC). As implied by the name, DSMC involves a simulation of the macroscopic gas dynamics by directly simulating the microscale processes of individual particles in the gas. Since the method is not a direct discretization of the Boltzmann equation, questions of accuracy and convergence were initially addressed by comparison with existing theory and experiments. Muntz [15] provides a nice review of computational validation of DSMC. Wagner [16] further validated the method by analytically establishing that DSMC converges to the Boltzmann equation in the limit as $N \to \infty$. More recently, Hadjiconstantinou et al. [17] investigated statistical error associated with results drawn from DSMC and Gallis et al. [18] compared DSMC results with the Chapman–Enskog formalization. A variety of review articles have been written on DSMC and its impact on various fields of study [15, 19–22].

While the individual motion of each gas particle is followed deterministically, the microscale binary collisions in the gas are simulated in a statistical manner. To increase computational efficiency, the number of collisions that must be tested and executed is limited by excluding the least likely of collision pairs (the collision integral in Eq. (1) calls for collision interactions between all points in phase space that conserve energy and momentum). Most-probable collision pairs are found by binning all particles in space to find groups of nearest neighbors. Based on local flow properties, random pairs of particles from within a bin are selected to undergo the collision procedure,

$$C_{\Delta t}\left(\vec{\mathbf{x}}_{\mathrm{j}}, \vec{\mathbf{p}}_{\mathrm{j}}\right) \otimes \left(\vec{\mathbf{x}}_{\mathrm{k}}, \vec{\mathbf{p}}_{\mathrm{k}}\right) = \left(\vec{\mathbf{x}}_{\mathrm{j}}, \vec{\mathbf{p}}_{\mathrm{j}}^{*}\right) \otimes \left(\vec{\mathbf{x}}_{\mathrm{k}}, \vec{\mathbf{p}}_{\mathrm{k}}^{*}\right) \ , \tag{6}$$

where $C_{\Delta t}$ is the collision operator for a time step of Δt and $\vec{\mathbf{p}}^{*}$ denotes a new momentum following the exchange due to the collision. The statistical collision operator $C_{\Delta t}$ conserves momentum and energy and can be elegantly described as a Markovian procedure based on a null collision operator (for details see [23]). Typically, particle bins consist of a fixed underlying mesh which represents a unique non-overlapping decomposition of the domain D,

$$D = \bigcup_{k=1}^{M} \gamma_{\mathrm{k}} \ , \tag{7}$$

where γ_{k} is the k^{th} mesh cell. Although this mesh must cover all spatial regions of interest, its memory requirement is much less than that of the multidimensional phase-space distribution function $f\left(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t\right)$.

Despite the memory requirement being much reduced, a mesh that covers the entire domain D does not allow for optimal use of computational resources. This can be the case, for example, when portions of the mesh remain empty due to uneven distribution of particles throughout the mesh. In addition, a fixed

grid of bins must be customized to the particular physical problem in question. Grid-mismatch to the physical system can be cause for error in simulation results. Conversely, a gridless method of determining local groupings of nearest neighbors makes it possible to dynamically match the distribution of particles, independent of the geometry of the boundary conditions or forces within the system.

In this paper, we present a gridless statistical approach to the simulation of rarefied gas dynamics. Using standard DSMC as a starting point, the gridless method is described and compared to mesh based methods. Following a description of the methodology, we discuss convergence of the gridless method and thereafter present results of benchmark simulations. The results obtained from gridless DSMC are compared with those previously validated and obtained from standard fixed-grid DSMC. A method of including boundary conditions is described and demonstrated.

2 Numerical approach

Each particle in a gas system can be characterized by its position and momentum at time t

$$\{(\vec{\mathbf{x}}_1(t), \vec{\mathbf{p}}_1(t)) \ , \ \ldots \ , \ (\vec{\mathbf{x}}_\mathrm{N}(t), \vec{\mathbf{p}}_\mathrm{N}(t))\} \ , \tag{8}$$

where $(\vec{\mathbf{x}}_\mathrm{i}(0), \vec{\mathbf{p}}_\mathrm{i}(0))$ are chosen by appropriately sampling Eq. (2). The N-particle distribution function at time t may be expressed as

$$f_\delta\left(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t\right) = \frac{1}{N} \sum_{i=1}^{N} \delta(\vec{\mathbf{x}} - \vec{\mathbf{x}}_\mathrm{i}(t)) \, \delta(\vec{\mathbf{p}} - \vec{\mathbf{p}}_\mathrm{i}(t)) \ . \tag{9}$$

In a direct simulation of microscopic gas dynamics, molecular motion and collisional processes must be calculated. This involves following the position of each particle in the system and exchanging energy and momentum between particles during collisions.

The basic tenet of DSMC allows separation of the collisional time scale from the time scale of the free molecular motion. Such a separation decouples simulations of the different physical processes, allowing these to be executed in an almost arbitrary order. It is thus possible to iterate a step-by-step algorithm, wherein each step represents the simulation of a different physical process. A simple implementation of DSMC could include a few steps as shown in Fig. 2. First, collisions are performed between neighbors according to a physical collision probability. Second, the particles are allowed free-molecular movement

according to the system Hamiltonian H. Last, boundary conditions are applied to particles as necessary.

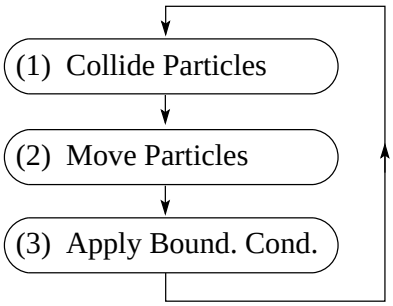

Fig. 2. Simple implementation of DSMC.

Repeated action of the split operator leads to a particular instantiation of the discrete phase-space distribution function $f_\delta(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$ in Eq. (9). To reduce statistical noise, many instances of $f_\delta(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$ are averaged together such that $f(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$ is given by

$$f(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t) \equiv \langle f_\delta(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t) \rangle = \frac{1}{J} \sum_{j=1}^{J} \int \int f_{\delta \mathrm{j}}(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t) \rho(\vec{\mathbf{x}}) \phi(\vec{\mathbf{p}}) dx^3 dp^3 \qquad (10)$$

where J is the number of independent instantiations of $f_\delta(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$ in the average and $\rho(\vec{\mathbf{x}})$ and $\phi(\vec{\mathbf{p}})$ are suitable test functions. In grid based DSMC, the underlying mesh may be thought of as the test function in constructing the averages. For simulations of steady state behavior, the split operator is iterated until transients have substantially decayed. Thereafter, J instances of $f_\delta(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$ (separated by several iterations of the algorithm so as to be statistically disconnected), are averaged together to compute $\lim_{t \to \infty} f(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$. For time dependent problems, the goal is to compute $f(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t_\mathrm{i})$ where t_i is the i^{th} time step in the simulation. In this case, each instance of $f_{\delta \mathrm{j}}(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t_\mathrm{i})$ in Eq. (10) is taken from a separate simulation seeded with a new random sampling of Eq. (2). Typically, the averaging in Eq. (10) is done by first projecting Eq. (9) onto a fixed underlying mesh and then averaging the values in this mesh. In practice, one seeks macroscopic flow properties such as density, stream velocity, and temperature. These properties are derived from moments of the computed $f(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$.

It must be noted that the de-coupling of the free molecular motion from collisional processes is only valid for simulations where the probability of any particle colliding with another in any given time step is much smaller than unity [24]. This is ensured by requiring that the basic time step of the simulation is no more than 10 % of the collision time.

Particle species for which long range interactions can be neglected only have a significant probability for collisions with near neighbors. Thus, for implementing a collision model for such particles, it is valid (and an efficient use of computational resources) to limit collision tests to pairs of particles that are separated by less than or on the order of the collision length. In other words, it is unnecessary to calculate collision probabilities for pairs of particles which are separated by a distance much greater than λ_{MFP}.

A typical approach for implementing efficient collision pair selection involves inserting a preparatory step before step 1 in Fig. 2. During this preparatory step, all particles are sorted into bins that represent a particular portion of the simulated space. Collision pairs are then created by selecting partners from within the same spatial bin. This allows an efficient selection of probable collision pairs without the need to overextend the computational resources.

Traditionally, a fixed grid of computer memory maps to a grid of simulation cells and provides natural binning for collecting and organizing the groups of nearest-neighbor particles. While this fixed-grid approach enjoys much success, development of fixed grids often requires much finesse for creating the correct layout of grid cells. For example, a system with steep gradients (due to strong external forces or sharp changes in flow properties) such as at the front of a shock, a uniform grid is problematic: if the grid cells are too large, such that the shock front is traversed by one or two cells, dramatic error in the collision statistics at the shock front will result. On the other hand, if cells are too small, particles will be non-physically thermally isolated from each other. In addition, a dense grid in a rarefied gas region will result in wasted computational resources. Ideally, creating a physically correct simulation with a fixed grid involves an algorithm with the following steps.

(1) A fixed grid is created to best match the estimated solution.
(2) The simulation is executed until steady state is reached.
(3) The simulation results are compared with the geometry of the grid.
(4) If the stream lines of the flow field differ substantially from the geometry of the grid, the algorithm returns to step 1 to repeat the process.

By iterating this algorithm, a correct grid would be found that produces the correct solutions to the physical system. Adaptive mesh refinement is one method of automatically carrying out these steps [25, 26].

Rather than attempting to create physically correct simulations via the grid-refinement procedure, we have implemented a gridless technique that avoids the issue of grid-mismatch. Without the need to match a grid to an underlying physical system, the gridless approach further allows the abstraction of the DSMC code from the description of the physical environment. In other words, the user is allowed to focus on the details of the physical system (boundary conditions, forces, etc.) rather than the DSMC layer of the simulation.

The main disadvantage of a gridless system stems from the computational time required to sort particles into nearest-neighbor lists. It is of utmost necessity to avoid algorithms where the required computational time increases quadratically (or greater) with particle number, such as bubble sort which requires up to $O(N^2)$ time. Various non-gridded sorting algorithms have been available for some time, but only since the development and work with the generalized binary sort/search algorithm (expanded to k dimensions) has a tractable solution been possible.

The general idea of the binary sort/search algorithm relies on the divide and conquer principle. Consider a number secretly picked at random between two known fixed points on the number line. Consider further that we are given the task of discovering the number and can only ask "is it greater or less than" questions. The binary search algorithm dictates that our first query use the center point between the two fixed points. If the answer is "less than," we submit a second query for the value in the center of the section between the first query and the end point on the left (assuming that the number line is less on the left). We iterate the queries always using the center points of a smaller and smaller section of the number line. This search executes in $O(\log_2[N])$ time.

Instead of searching a previously sorted set of data, the task in DSMC is to sort a set of particles into nearest neighbor containers. To perform this operation, we can continue to use the divide and conquer principle as in the following example. Consider a one-dimensional set of random numbers for which we are given the task to sort numerically. Using the divide and conquer principle, we first estimate the midpoint of the distribution of the numbers. Using this estimated midpoint we compare all numbers to this selected value and put all "less-than" numbers in a container on the left and all "greater-than" numbers in a container on the right. This midpoint value, used to subdivide the group, is called the pivot point. For each of the newly created groups of numbers, we define a new pivot point and subdivide each container further by the same process. By iterating this algorithm, the set of initially random numbers is quickly ordered in $O(N \log[N])$ time. This is known specifically

as the quicksort algorithm.

By retaining the values of the pivot points and the relation of those pivots to the sorted data, a hierarchical tree structure is generated that is optimal for the binary search described above. This tree is called a binary search tree. The size, or depth, of the tree is characterized by the number of iterations required to build the tree, or rather, the number of edges from leaf node to root in a graph view of the tree.

The quicksort and binary search algorithms can easily be extended to k dimensions by iterating the one dimensional algorithms independently over each dimension for each level of the tree. Doing so, each new level of tree results in creating 2^{k} new children for each parent node, or $\left(2^{\mathrm{k}}\right)^{l}$ new total nodes for new tree depth of l. The two and three dimensional specializations of this hierarchical tree method are known as the quadtree and octree, respectively. In a quadtree, each parent node may have as many as four child nodes and in an octree, the parent may have as many as eight child nodes. Because of the simple application of the binary search algorithm to the tree, the quadtree and octree lend themselves well for performing searches for a set of objects in two or three dimensions. This is the approach taken by many modern computer graphics systems for determining which items, from a set of objects, to render [27]. Such a hierarchical organization is also commonly used for simulated collision detection [28], especially in modern computer gaming systems.

Because our particle simulations are at most three-dimensional, we will henceforth refer to the algorithms as the octree algorithm. By implementing the octree algorithm, we have developed a code that automatically adjusts to the changes of the particle distribution in a simulation requiring $O\left(N \log\left[N\right]\right)$ time.

2.2.1 Octree for DSMC

For DSMC, it is necessary to add a series of specializations to the standard octree building process. First, the choice of the pivot location can influence the statistical error and ability of the code to adapt to the physical system. Second, it is necessary to attach rules which disqualify children to avoid poor aspect ratio as well as empty (and not useful) children. Third, it can be advantageous to allow the bounded volume of a node to adapt more closely to the minimal bounding volume of the particles within its boundaries. Finally, the stopping condition, or the condition that halts further subdivision in a tree can have a dramatic effect on the validity of the collision selection process. The remainder of Sec. 2.2 discusses these specializations in detail.

2.2.2 *Pivot location*

In a typical octree implementation, the clusters at each level of the tree are uniform cubes obtained by bisecting the previous generation of clusters in each coordinate direction. In other words, the geometric center of a parent node is used as the pivot point for the quicksort. Although this choice of pivot point requires near zero computational time to calculate, DSMC can benefit by using an alternate scheme. In a gas simulation, particles are randomly distributed according to the Boltzmann equation (Eq. 1). The distribution may exhibit large concentrations of particles in various regions of space as well as large empty spaces in other regions. It is advantageous to select a pivot point, such as the center of mass, that more equally divides the particles into children nodes. By doing so, the sort finishes in a shorter time and results in a shallower and more balanced tree. This facilitates a shorter walk through the tree for later use. Furthermore, by ensuring that leaf nodes have roughly the same particle number, statistical noise that is common in DSMC due to large oscillations in the cell occupancy rate should be minimized, though not eliminated. In addition, a more uniform tree depth tends to keep the computational load spread evenly throughout the leaves of the tree. This is important for vectorization or parallelization schemes which will be demonstrated in a follow-up paper.

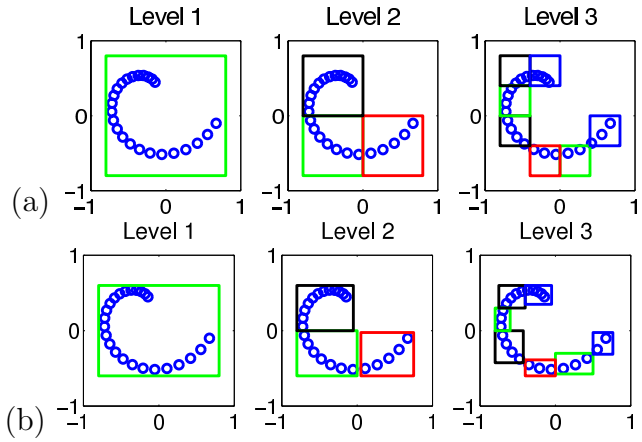

Fig. 3. Examples of a quadtree structure divided using a geometric pivot point; (a) standard scheme, (b) scheme that adapts nodal volumes to the local minimum bounding volume.

2.2.3 *Disqualification rules*

While building the octree, it becomes necessary to disqualify new child nodes that may produce bad/incorrect collision statistics. The first disqualification rule is expressed by a minimum number of particles allowed for a node: any child that contains fewer particles than the minimum is absorbed into its sibling created by the same division process. This provides some element of

adaptability of the octree to the local particle distribution. Fig. 3a shows a spiral set of particles that are divided by geometrically centered pivot points into a three level tree that follows the first disqualification rule. Fig. 3b shows a further adaptation by shrinking the local leaf-node volumes to the minimal containing volume. This adaptivity plays a key role in the gridless simulation of charged particle transport, where long range interactions are approximated by a moment expansion about the geometric center of tree nodes [29]. The choice of the minimum number of particles is discussed below in Sec. 2.2.5.

A second disqualification rule pertains to the aspect ratio of particle clusters as described by the nodes of the octree. For DSMC, it is important to avoid collision pairs that are so far apart that they have a minimal probability of colliding. This non-locality problem is solved by avoiding large aspect ratios defined as $\mathrm{AR_i} = \max\{\Delta_\mathrm{i}/\Delta_\mathrm{j},\ \Delta_\mathrm{i}/\Delta_\mathrm{k}\}$, $i, j, k \in \{x, y, z\}$, and $i \neq j \neq k$, where Δ_i is the size of the cluster in the i^{th} direction. To prevent poor aspect ratio $\mathrm{AR_i}$, a level of the tree is not divided in the i^{th} direction if $\mathrm{AR_i}$ exceeds 3/2. Fig. 4 shows a set of three types of decomposition using this rule of thumb to guard against nodes that fail to meet this aspect ratio criterion. In each case, the contrived distribution of particles is sorted into a three-level quadtree. Fig. 4a shows a geometric division and Figs. 4b-c show a division about the center of mass.

2.2.4 Boundary shrinking

The differences between Figs. 4b-c pertain to the bounded volume of each node. For Fig. 4b, each new child of the tree has its volume shrunken to minimally bound all its particles as was done for the geometric division shown in Fig. 3b. Although this approach does adapt very well to an arbitrary distribution of particles, it can cause dramatic error in the collision rate for DSMC. This is because empty space will be incorrectly ignored and the local density in each node will be calculated to be higher than the physical value. However, level 2 of Figs. 4a and c point out that with no adaptive shrinking, it is possible to have a cluster where large regions of the cluster have no particles, implying that the local Knudsen number will be artificially low.

We therefore arrive at Fig. 4c, which shows the approach used in our gridless DSMC implementation. In this approach, clusters are shrunken if there is a large discrepancy between the defined volume of the cluster and the actual volume of the contained particles. The amount of free space on each side of the node is considered independently. Before each test, the rectangular volume that minimally bounds the particles is measured, with $(\Delta_\mathrm{particles})_\mathrm{i}$ being the length of this rectangle in the i^{th} direction. If the length of empty space on a given side is greater or equal to $(\Delta_\mathrm{particles})_\mathrm{i}$, then the edge of the node on that side is redefined to be the edge of the particle distribution. It is found that

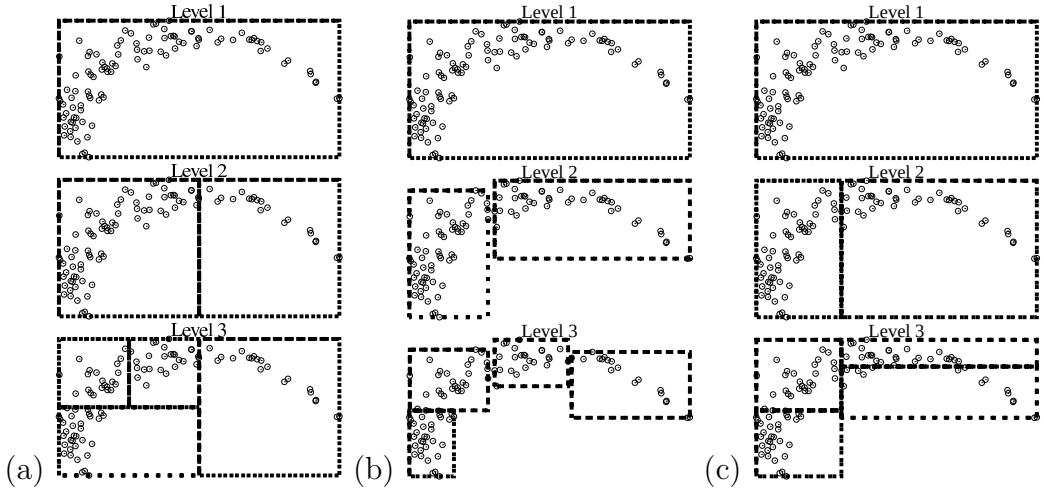

Fig. 4. Examples of tree structure for different models tree node division. (a) Division by the geometric center of the tree node. (b) Division by the center of mass with the volume shrunk to barely contain all particles. (c) Division by the center of mass with the volume shrunk only if a major portion of the volume is empty.

this method can sometimes shrink the leaf nodes to be too small. To avoid shrinking too much, it should be possible to apply the criterion to a parent node for deciding a shrinking question of a child node.

2.2.5 DSMC validity and octree stopping conditions

The goal of DSMC is to obtain correct collision statistics so that a rarefied gas can be correctly modeled. Using the octree to sort particles into nearest neighbor clusters, collision pairs are formed by randomly selecting partners from the same cluster. It is therefore critical to create final clusters of particles from which only probable collisions can occur. For instance, if a cluster is too large, a randomly selected pair may be separated by too great a distance to be considered for a probable collision event. We attempt to ensure the validity of the collision selection process by engineering the size of the leaf node clusters such that only probable collision pairs can be formed. Practically speaking, this translates into choosing a spatially dependent number of particles needed for an octree node. We now describe validity condition and how it translates into the minimum number of particles per tree node.

In the following, let Δ represent the length scale of the cluster of particles from which collision pairs are selected. The validity test for DSMC can be expressed as the ratio of Δ to the mean free path λ_{MFP}. Alexander et al. [30] demonstrated how the error in DSMC results depends on the ratio (Δ / λ_{MFP}). Alexander et al. showed that for (Δ / λ_{MFP}) ≈ 1, error in the hard sphere viscosity and thermal conductivity can be as high as 7.5% and 4.5%, respectively. We therefore seek to keep all collision pairs separated by a fraction of λ_{MFP}, depending on the error that we can tolerate.

For simplicity, we assume that particles in a particular collision cluster are uniformly distributed. If we select the maximum allowed cluster scale size to be $(1/2)\lambda_{\mathrm{MFP}}$, the average separation between two collision partners will be $(1/6)\lambda_{\mathrm{MFP}}$ with a root mean square separation of $\sim 0.204\lambda_{\mathrm{MFP}}$. This is easily demonstrated by measuring the average and root mean square values of the difference between two random numbers that are both less than 0.5. Using this choice, we express the validity condition as

$$\frac{\Delta}{\lambda_{\mathrm{MFP}}} \lesssim \frac{1}{2} \, . \tag{11}$$

Using Eq. (11), we can derive a target number of particles per collision cluster. Assuming a three dimensional simulation, the approximate number of simulation particles in a cluster at position $\vec{\mathbf{x}}$, denoted $\zeta(\vec{\mathbf{x}})$, is given by

$$\zeta(\vec{\mathbf{x}}) = \frac{1}{F_{\mathrm{N}}} \Delta^3 n(\vec{\mathbf{x}}) \tag{12}$$

where $n(\vec{\mathbf{x}})$ is the local number density of the gas and F_{N} is the number of physical particles each simulated particle represents. Using Eqs. (3), (11) and (12), we write

$$\zeta(\vec{\mathbf{x}}) \lesssim \frac{\sqrt{2}}{32 F_{\mathrm{N}} \, {\sigma_{\mathrm{T}}}^3 \, n^2(\vec{\mathbf{x}})} \, . \tag{13}$$

In Eq. (13), $\zeta(\vec{\mathbf{x}})$ represents the minimum number of particles that a cluster can have with a given local number density $n(\vec{\mathbf{x}})$ and produce valid collision selections. For simulations in k dimensions, where $(3-k)$ dimensions are ignored because of a symmetry, Eq. (13) is rewritten as

$$\zeta(\vec{\mathbf{x}}) \lesssim \frac{1}{F_{\mathrm{N}}} \left(\frac{\sqrt{2}}{4\sigma_{\mathrm{T}}} \right)^{\mathrm{k}} \frac{1}{n_{(\mathrm{k})}^{\mathrm{k}-1}(\vec{\mathbf{x}})} \tag{14}$$

where $n_{(\mathrm{k})}$ has inverse units of length to the k^{th} power and is defined by

$$n_{(\mathrm{k})} = n \, (1\mathrm{u})^{3-k}$$

where u is the basic length unit of the simulation (meters for S.I. units).

To apply Eq. (14) to building the octree, we define the octree stopping condition given as

$$N < 2\zeta(\vec{\mathbf{x}}) \, . \tag{15}$$

In other words, subdivision continues, until Eq. (15) becomes true. This has the effect of creating a nodal occupancy rate of $(1.5\,\zeta(\vec{\mathbf{x}}))$. In our implementation, we allow the user to select a maximum and minimum value for this stopping

condition, such that the stopping condition becomes

$$N < 2 \begin{cases} \zeta_{\min} \ , & \zeta(\vec{\mathbf{x}}) \leq \zeta_{\min} \\ \zeta(\vec{\mathbf{x}}) \ , & \zeta_{\min} < \zeta(\vec{\mathbf{x}}) < \zeta_{\max} \\ \zeta_{\max} \ , & \zeta(\vec{\mathbf{x}}) \geq \zeta_{\max} \end{cases} . \tag{16}$$

In our simulations, we set $\zeta_{\min} = 4$ so as to prevent unphysical thermal isolation. The use of $\zeta_{\min}$ should be used as a key warning signal for a simulation that does not contain enough particles to resolve the physical flow. An example of this is demonstrated in Sec. 3.5.2. We also typically set $\zeta_{\max} = 100$, but more investigation needs to be made to determine the effect of high values of $\zeta_{\max}$.

2.2.6 Complex octree example

The advantage to a tree structure is that it gives grid-free localized refinement of space for a given instant in time, providing a local set particles that are good candidates for collision partners. The tree structure adds additional flexibility to traditional DSMC and removes issues associated with grid choice by automatically adapting the refinement so as to give optimal local resolution. In Fig. 5, the finest level of the hierarchical tree for hypersonic flow past a square cylinder is shown. The visualization of the clusters clearly shows how the adaptivity of the tree is advantageous for flows that develop steep gradients in flow properties. The particular flow related to Fig. 5 is discussed in Sec. 3.5.1.

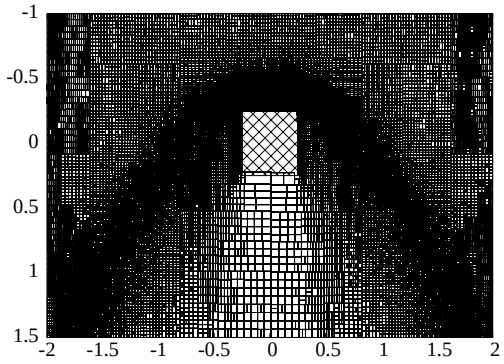

Fig. 5. Visualization of the finest level of the octree used for computing hypersonic flow past a square cylinder in gridless DSMC.

2.3 Revised DSMC algorithm

As described in the introduction for Sec. 2, the DSMC algorithm allows a nearly arbitrary order of the sort, collide, move, and boundary-condition steps. This allows us to optimize the order such that computational time is minimized and implementation is less complicated. An outline of our implementation of Gridless DSMC is shown in Fig. 6. Each time step begins with the current population of particles sorted into an octree that satisfies Eq. (16). After the sort is finished, macroscopic gas quantities (needed to obtain correct collision statistics) are assigned to each of the leaves of the newly created octree. (The following section will detail our method of performing this assignment in a gridless environment.) This is followed by the collision selection and execution methods detailed in Sec. 2.3.2. Free particle movement then takes place, followed by the application of boundary conditions.

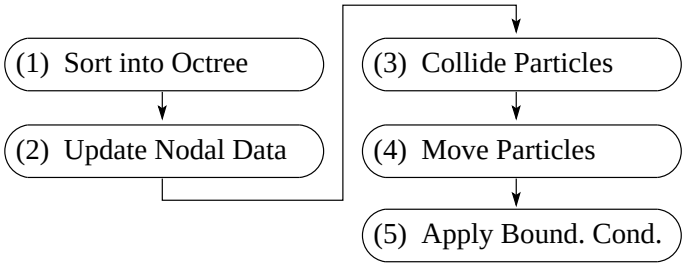

Fig. 6. Flow chart of our implementation of a gridless DSMC algorithm.

2.3.1 Gridless maintenance of macroscopic quantities

In traditional DSMC, local macroscopic gas properties are needed to maintain correct collision rates. Many of these macroscopic quantities tracked per grid cell are further time-averaged to reduce statistical noise. In a gridless approach, there is no underlying structure to track and maintain such time-averaged macroscopic quantities. We have therefore developed an adaptive method that tracks these quantities at the lowest levels of the octree. Near the beginning of each time step (just after sorting), instantaneous quantities, such as $n_{\mathrm{j}}(t_{\mathrm{i}})$ and $\langle \vec{\mathbf{p}} \rangle_{\mathrm{j}}(t_{\mathrm{i}})$, are measured per each j^{th} octree node. To account for time averaging, a weighted average is taken between the instantaneous data and an interpolation of the data from the previous time step, as indicated in Fig. 7. After performing this update, the old tree is then discarded.

To perform the interpolation, any unstructured interpolation scheme will work, such as Shepherd's algorithm [31]. Unfortunately, performing unstructured interpolation is known to be very computationally intensive. We solve this dilemma by accepting a method with an interpolant that does not pass exactly through the data. Hernquist and Katz [32] developed such a method for use with octrees in a smoothed particle hydrodynamics calculation.

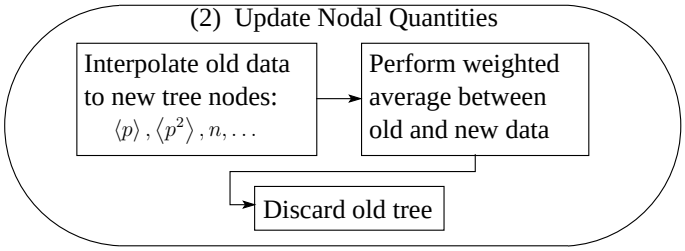

Fig. 7. Steps to update local gas properties using a gridless technique.

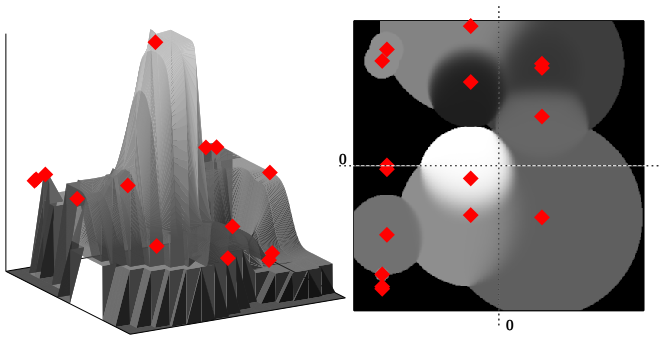

Fig. 8. Spherical spline interpolation of a coarsely sampled $\sin(x)/x$ function. The diamonds indicate data points. Left: Comparing the interpolated spline to the original set of data shows a reasonable interpolation. For visibility, a coarse sampling of the interpolant was used. Right: A two-dimensional view of this interpolated spline highlights the radii of influence assigned to each of the data points. The radius of influence is dynamically assigned such that a given number of nodes fit inside the spherical volume with that radius. For demonstration purposes, the number of nodes within the volume of influence in this figure was limited to one. Thus, some data points with very small separation have a nearly invisible radius of influence.

This method is based on a spherical spline that computes a weighted average of the data in the octree to create approximate values at points not located at the pivot points of the octree.

The kernel for the weighted average is given by

$$w\left(r, r_{\mathrm{I}}\right) = \frac{1}{\pi {r_{\mathrm{I}}}^3} \cdot \begin{cases} 1 - \frac{3}{2}\left(\frac{r}{r_{\mathrm{I}}}\right)^2 + \frac{3}{4}\left(\frac{r}{r_{\mathrm{I}}}\right)^3, & 0 \leq \left(\frac{r}{r_{\mathrm{I}}}\right) \leq 1 \\ \frac{1}{4}\left(2 - \left(\frac{r}{r_{\mathrm{I}}}\right)\right)^3, & 1 \leq \left(\frac{r}{r_{\mathrm{I}}}\right) \leq 2 \\ 0, & 2 \leq \left(\frac{r}{r_{\mathrm{I}}}\right) \end{cases} \tag{17}$$

where r is the distance from the data to the point of evaluation and r_{I} is known as the radius of influence for the respective data point. Evaluation of the kernel in Eq. (17) allows us to minimize the number of nodes used to compute an approximate interpolated value. Using the binary search algorithm described in Sec. 2.2, we find and use all nodes with a non-zero contribution to the

weighted average. This speeds up the operation considerably as compared to other unstructured interpolation routines.

The influence a particular node in the tree has on evaluations of Eq. (17) in its nearby vicinity depends on r_{I}, which is given by

$$r_{\mathrm{I}} = \left(\frac{3V_{\mathrm{min}}}{4\pi}\right)^{1/3} \tag{18}$$

where V_{min} is the smallest volume that includes a specified number of nodes, N_{I}. r_{I} is determined under the assumption of local isotropy in the local distribution of octree nodes.

A demonstration of this technique is shown in Fig. 8, where the interpolation of coarsely sampled $\sin(x)/x$ is performed. As shown in Fig. 8, if N_{I} is chosen too small, the spline may too quickly tend to zero between tree nodes, where non-null values might be expected. Conversely, by choosing N_{I} too high, extrema in the data to be interpolated may be artificially dampened. It is thus necessary to ensure that N_{I} is neither too small nor too large. Depending on the application, we have found reasonable values to be in the range $1 \leq N_{\mathrm{I}} \leq 10$. For the simulations in this paper, we found $N_{\mathrm{I}} = 4$ to be a robust choice.

In terms of using this method in DSMC for tracking local gas properties, the effect is to both average in time as well as in space. Though a more thorough study of this is needed, we believe this may further reduce the typical DSMC noise. A more thorough study could include the comparison with simulations where algorithms similar to Shepherd's interpolation are used.

2.3.2 Collide particles

Within a tree-node, standard DSMC techniques are used for selecting collision pairs with an alteration only in the number of collision pairs to test. G. A. Bird's standard "no-time-counter" method [3] prescribes the number of collision pairs to test N_{sel} as

$$N_{\mathrm{sel}} = \frac{1}{2}\frac{F_{\mathrm{N}} N \langle N \rangle \Delta t \left(\sigma_{\mathrm{T}} v_{\mathrm{rel}}\right)_{\mathrm{max}}}{V} \tag{19}$$

where F_{N} is the number of physical particles each simulated particle represents, N is the number of simulated particles in the cell, $\langle N \rangle$ is this number averaged over time, Δt is the time step of the simulation, V is the volume of the cell, and $\left(\sigma_{\mathrm{T}} v_{\mathrm{rel}}\right)_{\mathrm{max}}$ is the maximum value in the particular cell of the product of the total scattering cross-section σ_{T} and relative particle velocity v_{rel}. The normalized probability of any given collision pair actually exchanging

momentum is then given by

$$P_{\text{coll}} = \frac{(\sigma_{\text{T}} v_{\text{rel}})}{(\sigma_{\text{T}} v_{\text{rel}})_{\text{max}}}. \tag{20}$$

In our gridless algorithm, we replace some of the factors in Eqs. (19) and (20) with their space-time smoothed equivalents, denoted as $< \cdots >_{x,t}$, from the gridless tracking scheme described in Sec. 2.3.1. Doing so, Eqs. (19) and (20) become

$$N_{\text{sel}} = \frac{1}{2} F_{\text{N}} N \langle n \rangle_{x,t} \Delta t \langle (\sigma_{\text{T}} v_{\text{rel}})_{\text{max}} \rangle_{x,t}, \tag{21}$$

and

$$P_{\text{coll}} = \frac{(\sigma_{\text{T}} v_{\text{rel}})}{\langle (\sigma_{\text{T}} v_{\text{rel}})_{\text{max}} \rangle_{x,t}} \tag{22}$$

where $\langle n \rangle_{x,t}$ and $\langle (\sigma_{\text{T}} v_{\text{rel}})_{\text{max}} \rangle_{x,t}$ are the space-time averages of number density n and $(\sigma_{\text{T}} v_{\text{rel}})_{\text{max}}$ respectively.

2.3.3 Move particles

Transport is handled via a method of characteristics for a time step Δt,

$$\vec{\mathbf{x}}_{\text{i}}(t + \Delta t) = \vec{\mathbf{x}}_{\text{i}}(t) + \frac{1}{m_{\text{i}}} \int_t^{t+\Delta t} \vec{\mathbf{p}}_{\text{i}}(\tau) d\tau, \tag{23}$$

$$\vec{\mathbf{p}}_{\text{i}}(t + \Delta t) = \vec{\mathbf{p}}_{\text{i}}(t) + \int_t^{t+\Delta t} \vec{\mathbf{F}}_{\text{i}}(\vec{\mathbf{x}}_{\text{i}}(\tau), \tau) d\tau \ . \tag{24}$$

The particular integrator that one uses to move the particles under free molecular motion depends on the application. For instance, in a non-harmonic atomic trap it is often best to use a high accuracy method, such as 5th–4th order adaptive Runge–Kutta. For the case when there are trivial external potentials, a simple leap-frog or 2nd order Runge–Kutta routine is sufficient.

2.3.4 Apply boundary conditions

After moving the particles and before continuing with the next time step, it is necessary to apply boundary conditions. To do this, we introduce the concept of boundary objects. A boundary object can be something as simple as a reflective surface that has a bounded volume (for generality, the bounding volume of a boundary object is allowed to be as large or small as possible). While moving the particles, we maintain and update the maximum distance any one particle in a particular octree leaf has traveled, $\Delta x_{\text{max}} = (||\vec{\mathbf{v}}||\Delta t)_{\text{max}}$. We expand the bounding box by Δx_{max} on each side and test for resulting

overlap with the bounded volume of the octree leaf, as depicted in Fig. 9. The associated boundary condition for the overlapping boundary objects is then tested and applied to all relevant particles within the leaf. This method is briefly discussed in Sec. 3 with an example application.

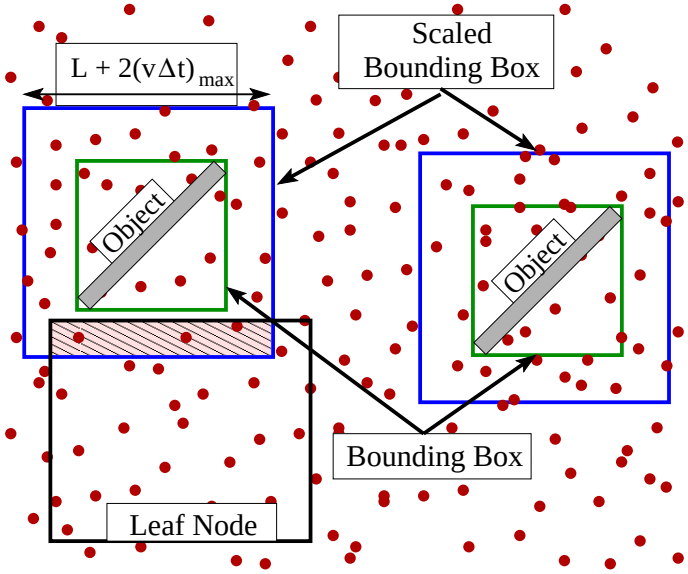

Fig. 9. Schematic of boundary object collision test for applying boundary conditions. The side lengths of bounding boxes of all boundaries are first increased by the maximum distance traveled $(||\vec{\mathbf{v}}||\Delta t)_{\max}$ by any one particle in the given octree node. Boundary conditions are only applied from scaled boundaries that overlap with the octree node. The shaded region indicates the overlap between one boundary and the octree node.

For each new geometrical type of boundary condition, a new boundary object must be coded. To streamline the application of complex physical boundaries, an approach more similar in nature to graphics display code might be helpful: large complex surfaces are often reduced into simple constituents.

2.4 Operator splitting of Eq. (1)

As pointed out earlier, DSMC typically involves operator splitting into two distinct phases: transport and collisions. The transport step is purely deterministic and, for some external forces, can be solved analytically. The collisional step is handled via a Monte Carlo sampling procedure, where two nearby test particles are selected at random and allowed to collide. For the following, let the transport operator for a time step of Δt be denoted by $S_{\Delta t}$ and the collision operator be denoted by $C_{\Delta t}$. The original approach proposed by Bird is an Euler splitting, whereby, in l time steps, the system will evolve as

$$f\left(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t + l\Delta t\right) = \left(S_{\Delta t} C_{\Delta t}\right)^l f\left(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t\right) . \tag{25}$$

Wagner [16] showed that this approach converges to a solution of the Boltzmann equation as $N \to \infty$. Rjasanow and Wagner [23] further investigated the effect of the type of operator splitting used: Euler ($S_{\Delta t}C_{\Delta t}$), Strang ($C_{\Delta t/2}S_{\Delta t}C_{\Delta t/2}$), and non-split methods. Rjasanow and Wagner found that non-split methods converge faster and that the convergence for Strang splitting is almost as fast as non-split schema. The results in this paper were computed using Euler splitting. Future work will investigate the Strang type splitting in context of the gridless DSMC algorithm.

3 Results

To verify the validity of this numerical approach, we demonstrate some standard DSMC simulation tests. We begin this section with a demonstration of the convergence properties of gridless DSMC. Next we discuss the speed of the algorithm and give timing results for the major bottlenecks. To show the accuracy of the algorithm, we further present the results from several different simulations. The first set of results pertain to low density gradient flows including a series of standard benchmark simulations of Couette flow and a simple test of drag from low-velocity ($Ma \approx 0.13$) flow past a flat plate. The second set of results are from two simulations of hypersonic flow ($Ma \approx 10$) where density gradients are large and the gridless method stands most to prove its utility.

For both sets of simulations, boundary conditions consist of specularly and diffusely reflecting surfaces. For diffusely reflecting surfaces which are held at a given temperature, incident particles are re-emitted in an effusive manner with a velocity given by a random sampling of the thermal distribution of the surface. In some cases, the walls are also given a shear velocity component (parallel to the surface) which causes the incident particles to be re-emitted in a reference frame moving at the velocity of the wall. For specular surfaces, incident particles are reflected such that energy is conserved and the momentum component transverse to the surface is reversed.

3.1 Convergence

The major advantage to using DSMC versus a direct solution of the Boltzmann equation (Eq. 1) is to lessen the amount of computational resources needed. As previously discussed, a single iteration of the DSMC algorithm is clearly less computational demanding than a single iteration of a finite difference method that might be used to solve Eq. (1) directly. It thus becomes important to determine the convergence properties of the seemingly less de-

manding algorithm. (In other words, if DSMC requires near infinite time to converge, we have gained nothing.) In this section, we discuss convergence of our DSMC implementation. Convergence in terms of the central limit theorem (CLT) will be described first. Convergence tests are shown for a Couette flow similar to Sec. 3.3.2. To demonstrate the CLT in our DSMC implementation, we also show the results of a simulation for a uniform three-dimensional gas in a periodic box.

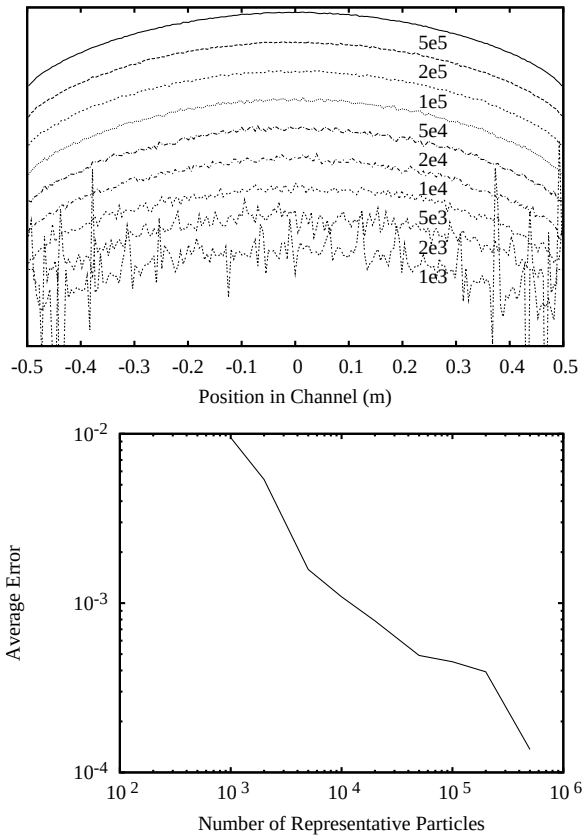

Fig. 10. Top: Waterfall plot of the temperature showing convergence of the velocity driven Couette flow as a function of N (see Sec. 3.3.2). Bottom: Shows the rate of convergence as a function of N.

At least two forms of a central limit theorem (CLT) apply to the DSMC algorithm. The first is the standard CLT which states that the mean of a sample of random variables of a particular distribution becomes more normally distributed as the number of random samples increases. In accordance with the Berry–Esséen theorem, convergence goes as

$$\epsilon_Z \propto 1/\sqrt{Z}\,, \tag{26}$$

where Z is the number of random samples and ϵ_Z is the error. This convergence law pertains to the value of J (number of samples taken) for computing the final form of $f(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$ in Eq. (10). In addition, it is evident in Eq. (9) that such a CLT applies to the value of N (number of representative particles in

the system). To verify the standard CLT for N, we first examine a refinement study with respect to N for the Couette flow described in Sec. 3.3.2. The size of the representative particles, F_{N}, is varied such that the physical number density remains constant throughout the refinement study. The top panel of Fig. 10 shows a waterfall plot of the temperature profile in the Couette flow as a function of N. Each curve of the waterfall plot represents the average of $J = 5400$ samples and is displaced vertically by 5 K from the neighboring curves. As N increases, the measured momentum distribution becomes more thermal and the associated effective temperature profiles become more smooth across the channel. The bottom panel of Fig. 10 depicts, as a function of N, the average error of a particular run to the apparent limiting solution (taken from smoothed results of the simulation with $N = 5 \times 10^5$). This plot shows that the convergence goes as $\approx 1/N^{0.6}$. It should be noted that for the simulations presented in this paper, J was chosen large enough such that $\epsilon_N \gg \epsilon_J$.

We assert that an additional, non-standard form of the CLT applies to the thermalization of a discrete gas and is related to the number of collisions within the gas. For simplicity, consider first a homogeneous gas with a particular momentum distribution $f(\vec{\mathbf{p}})$. For such a gas, Wild [33] showed that it is possible to solve Eq. (1) via an iterative process. Each iteration takes into account more and more of the collision history of the gas and the resulting form of $f(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$ is called the Wild expansion. For Maxwellian molecules, the Wild Expansion takes on a simplified form, the first three iterations of which are given by

$$\begin{aligned} f_1(t) &= e^{-t} f_0 \\ f_2(t) &= f_1(t) + e^{-t}(1 - e^{-t})\mathcal{Q}(f_0, f_0) \\ f_3(t) &= f_2(t) + e^{-t}(1 - e^{-t})^2 \\ &\quad \cdot \left[\mathcal{Q}(f_0, \mathcal{Q}(f_0, f_0)) + \mathcal{Q}(\mathcal{Q}(f_0, f_0), f_0)\right]/2 \end{aligned} \tag{27}$$

where it has been established that $\lim_{n\to\infty} f_{\mathrm{n}}(t) = f(t)$ [33]. The iteration number n of the expansion is related to the number of collisions the system has undergone by time t, as evidenced by the depth of the nested collision operator $\mathcal{Q}(\cdot, \cdot)$. For example, terms in Eq. (27) such as $\mathcal{Q}(f_0, \mathcal{Q}(f_0, f_0))$ describe particles that undergo collisions with particles that have already collided once. It is worth noting that the Wild expansion has been used as a time accelerant in standard DSMC by tracking the probability of multiple binary collisions per particle [34, 35].

Carlen et al. [36] determined a convergence rate for the expansion in Eq. (27) and showed that a CLT applies, such that the convergence of $f_{\mathrm{n}}(t)$ to the true solution $f(t)$ is exponential in n. As a corollary to the CLT for truncation of the Wild expansion, Carlen et al. further showed that a CLT applies to $f(t)$ (the true solution of Eq. (1)) such that $f(t)$ converges to a Maxwellian exponentially in time. In other words, an initially non-thermal distribution

function $f(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$ will thermalize in exponential time. In this context, time is typically measured in units of the collision time.

We expect that the discrete system $\langle f_\delta(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)\rangle$ will exhibit similar behavior. It is easy to see that a given $f_{\delta \mathrm{j}}(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$ of the discrete particle system maps onto parts of $f_{\mathrm{n}}(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$ and one might expect that $\langle f_\delta(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)\rangle$ provides a fairly complete coverage of $f_{\mathrm{n}}(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$. Hence we expect exponentially fast convergence of $\langle f_\delta(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)\rangle$ to a Maxwellian where the final error is subject to Eq. (26) for N. As a demonstration of this, we consider a homogeneous three-dimensional gas placed in a cubic volume (1 m^3) with periodic boundary conditions in all three dimensions. The simulated domain is filled with Ar such that $Kn = 0.1$, $\lambda_{\mathrm{MFP}} = 0.1$ m, and the time step is constrained to be $\lesssim 0.1\tau_{\mathrm{coll}}$, where τ_{coll} is the mean time between collisions. The initial spatial distribution is uniform and the initial momentum distribution $f(\vec{\mathbf{p}})$ is given by

$$f(\vec{\mathbf{p}}) = \prod_{\mathrm{i}} \mathfrak{f}(p_{\mathrm{i}}) \,, \qquad i \in x, y, z \tag{28}$$

where

$$\mathfrak{f}(p_{\mathrm{i}}) \propto \begin{cases} 0 & p_{\mathrm{i}} \leq p_{\mathrm{min}} \\ 2\sqrt{\beta/\pi} & p_{\mathrm{min}} < p_{\mathrm{i}} \leq 0 \\ \exp\left[-\beta(p_{\mathrm{i}} - p_0)^2\right] & 0 < p_{\mathrm{i}} \end{cases}, \tag{29}$$

$\beta = (2Mk_{\mathrm{B}}T)^{-1}$, and (p_0/M) is the drift velocity.

As the non-thermal gas begins to evolve, collisions between particles cause momentum exchange and $f(\vec{\mathbf{p}})$ is altered through the process, as shown in the left panel of Fig. 11. Each subsequent evolution of $f(\vec{\mathbf{p}})$ through a time span of length τ_{coll} might be represented by a subsequent iteration of the Wild expansion. As in Eq. (27) as n increases, a larger collision history accrues and one would expect that $f(\vec{\mathbf{p}})$ effectively becomes the result of a nested convolution of the initial condition in Eq. (29). Thus, following a CLT, $f(\vec{\mathbf{p}})$ becomes more normally distributed until a thermal system is achieved as depicted by the bottom curve in the left panel of Fig. 11.

The right panel of Fig. 11 shows the deviation of $f(\vec{\mathbf{p}})$ from a Gaussian as a function of the average number of collisions per particle, denoted by N_{coll}. In accordance with the expected convergence rate, the deviation does indeed decrease at an exponential rate until the statistical fluctuation ϵ_N becomes dominant. Further, the right panel of Fig. 11 shows that ϵ_N goes as Eq. (26) as dictated by the standard CLT. As shown, an increase of N by four times leads to a decrease in the final error by a factor of 2.

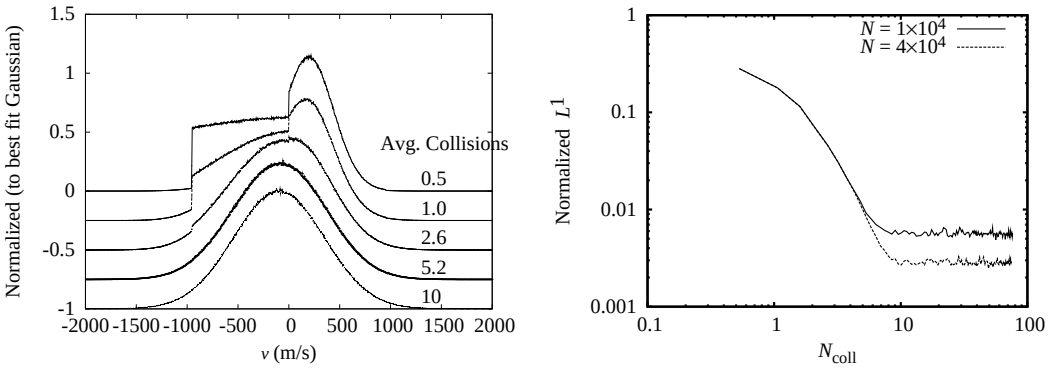

Fig. 11. Left: Waterfall plot showing convergence of an initially non-thermal gas to a Gaussian velocity distribution. Each curve represents a time-slice where, on average, N_{coll} collisions per particle have accrued. Neighboring curves are vertically offset by 0.25 for clarity. Right: Shows the normalized L^1 deviation from a Gaussian as a function of the number of collisions.

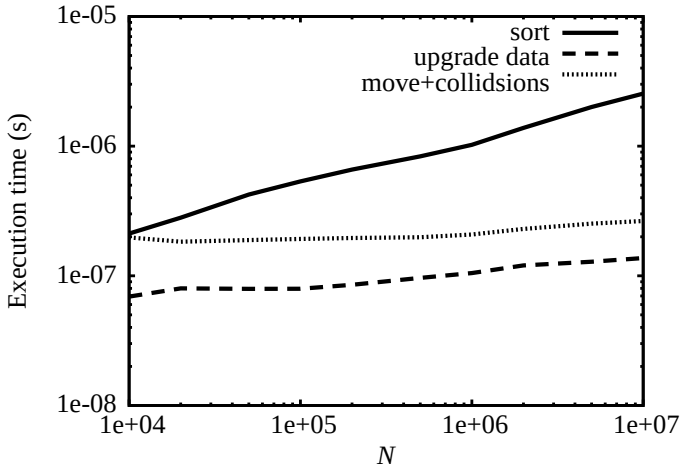

Fig. 12. Computational time required per particle per DSMC component as a function of N.

3.2 Computational time issues

For large DSMC simulations, the computational time required for an iteration of the algorithm is of utmost importance. Because the physical system is simulated by paying attention to the microscale interactions, memory as well as CPU resources are taxed very heavily. Time step iterations on the order of 1 μs per simulation particle are considered to run at average speed, while fast codes, all of which are optimized grid-based systems, have been known to need as little as ~ 0.5 μs (on comparable hardware) per particle [2].

While we do not anticipate creating a gridless methodology that is faster than

[2] Private communication with I. D. Boyd, 2005

gridded DSMC with the fastest integer based (hash) sorting algorithms, we seek to show that any performance degradation relative to the typical speeds is not too prohibitive. We therefore analyze the major components in our implementation to help validate this method as being reasonably useful. The timing results shown in this section were produced on an AMD Athlon 64 X2 4800+ processor (2.4 GHz). Fig. 12 shows the time required to compute each component of the algorithm for a simple three-dimensional periodic box. Timings for the move and collision portions of the algorithm are combined because, although they still stand to undergo significant optimization in our implementation, the basic differences between these and those of a gridded DSMC implementation are very small. Fig. 12 clearly shows that the primary bottleneck in gridless DSMC, as it stands, is the octree sorting. We are aware of several aspects in our code that can undergo significant optimization. In spite of this, we do not expect the general trend exhibited Fig. 12, nor the conclusion that the octree sort is responsible for the major time sink, to change.

As the most time-intensive portion of our code occurs within the octree sorting algorithm, we further discuss the execution time required to sort. Theoretically, because we use a quick sort, this sorting time should go as $O(N \log(N))$. As described in Sec. 2.2, the sorting algorithm can be augmented to be optimal for a particular calculation. For DSMC, we augment the algorithm to divide by the center of mass. Because of the additional time required to calculate the center of mass, it is expected that this augmentation causes an additional computational load. To analyze the severity of this additional load, we compare three different types of octree sort. The first method we name the Center method. This is the most standard division routine for the octree, where a node divides into children based on the geometrical center of the node. Because of this type of division, each of the child nodes are of the same shape and size. This represents the least expected computational load as is verified in Fig. 13, which shows the time-to-sort per particle as a function of increasing total particles.

The second method we name the CM method, or center-of-mass method. In this case, an octree node is divided into children with each of the axes of the division passing through the center of mass of the node. As shown in Fig. 13, the CM method is slower than the Center method. This is because the center of mass of each node must be computed before subdividing the node.

The third method we name the CM-Shrink method. Similar to the CM method, subdivisions are performed about the center of mass of the parent node. The difference is that the volume of each of the newly created children are shrunken to only bound the particles within this box. Though using the CM-Shrink method for DSMC would result in high collision rates (because the density would be over estimated), this type of subdivision will be useful for integration with plasma codes. This is the slowest of the three methods as shown

in Fig. 13, since, in addition to measuring the center of mass, the bounding volume must also be measured, although this represents only a slight increase in computational load per particle over the CM method.

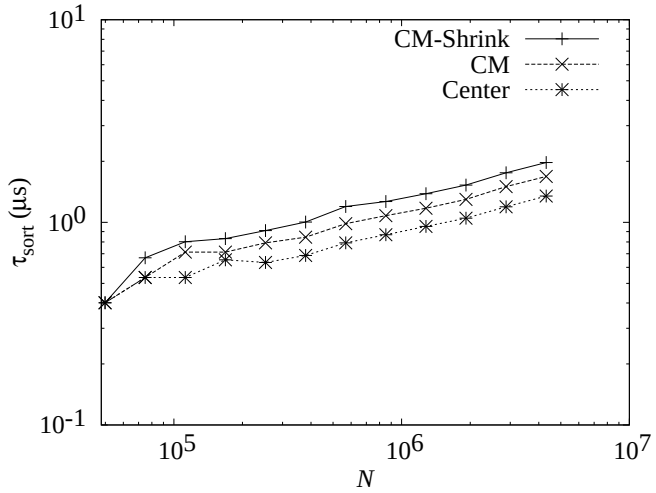

Fig. 13. Time to sort per particle (τ_{sort}) as a function of number of particles in the system N. The three volume defining methods shown only differ by a component linear in N. For DSMC calculations, where local density should not be over-estimated if we are to obtain correct collision rates, the most appropriate volume defining techniques are either the CM, or Center division methods. The CM–Shrink routine would be useful for plasma simulations.

It can be seen from Fig. 13 that though the performance of the three methods varies, the difference between them is not too great. In addition, the execution time of this sort is on the order of 1 μs, or less than an order of magnitude slower than optimized fixed-grid DSMC. We conclude therefore, that this gridless method is a viable DSMC method with respect to sort time.

3.3 Couette flow

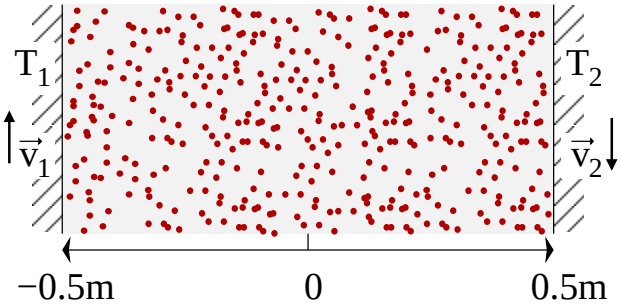

Fig. 14. Setup for Couette flow simulations. The channel between the two plates is initially filled with a uniform distribution of simulated Ar gas at 273 K.

A classic test to help validate a DSMC code is that of a Couette flow. A Couette system is described by two parallel, diffusely (*i.e.* non-specular) reflecting

plates, each at some specified temperature and transverse velocity, as depicted in Fig. 14. The dynamics of one-, two-, or three-dimensional gas between the plates are allowed to approach steady state. For the two following Couette-flow tests, the simulations were run as a function of Kn.

3.3.1 Thermal diffusion

In the first Couette-flow test, a uniformly distributed one-dimensional sample of Ar gas, initially at rest and at 273 K, is placed between two stationary plates held at 173 K and 373 K respectively. The system is then allowed to evolve until steady state is reached. For each of these simulations, the number of simulation particles was kept constant at $N = 10^4$. Many samples were averaged together to reduce the statistical noise in the presented results. For the fixed grid data, 1×10^5 samples were used [37], whereas for the gridless data, 2×10^4 to 7×10^4 samples were used, depending on the value of Kn.

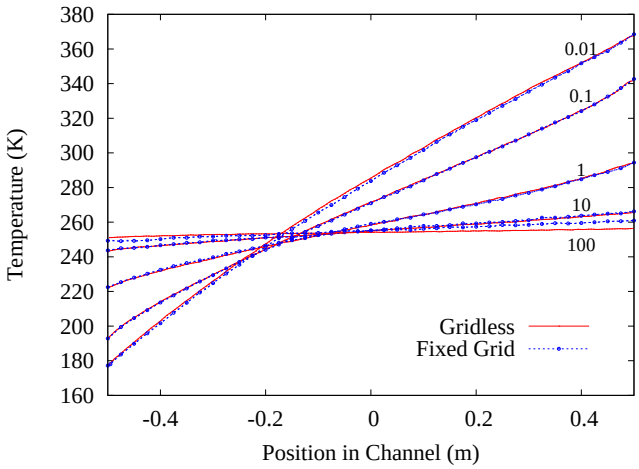

Fig. 15. Temperature profile across a channel in a thermal Couette flow simulation. The left and right walls are held at 173 K and 373 K respectively. The particles are initially uniformly distributed between the two plates with zero average velocity. The results for Knudsen numbers of 0.01, 0.1, 1, 10, and 100 are compared to similar results from a fixed grid approach. (Fixed grid data as published in [37] used with permission.) Fixed grid data courtesy of Dr. Quanhua Sun [37].

Profiles of the steady-state temperature across the channel for various orders of Kn are shown in Fig. 15. We find very good agreement with the results from the fixed grid simulation, which are also shown in Fig. 15, for $Kn <= 10$. For the very rarefied case with $Kn = 100$, we find that the gridless technique appears to produce results closer to the free molecular flow regime than does the fixed grid system. As discussed below, we believe this may be an indication that the gridless method actually performs better than the fixed grid system for very rarefied gases.

3.3.2 Velocity diffusion

In the second Couette-flow test, the two parallel plates indicated in Fig. 14 are held at a temperature of 273 K, but given anti-parallel velocities (transverse to the channel where the gas resides) of $\vec{\mathbf{v}}_1 = -150$ m/s $\hat{\mathbf{y}}$ and $\vec{\mathbf{v}}_2 = 150$ m/s $\hat{\mathbf{y}}$. Initial conditions as well as averaging procedures for this simulation coincided with those of the first Couette-flow test.

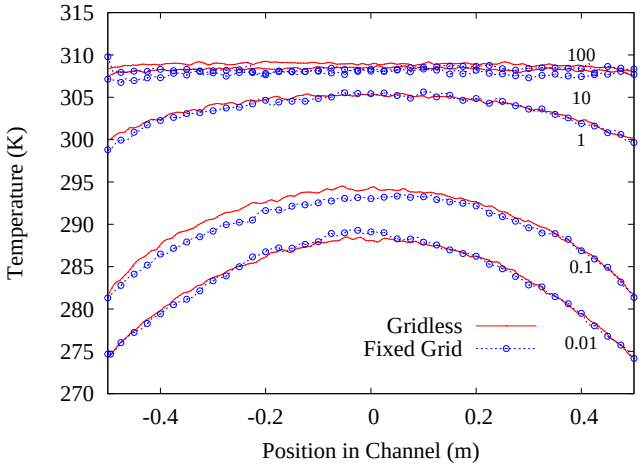

Fig. 16. Temperature profile of a Couette flow simulation. The particles are initially uniformly distributed between the two plates, the transverse (to the channel) velocity of each plate is $\pm 150 \frac{\mathrm{m}}{\mathrm{s}}$ respectively. Fixed grid data courtesy of Dr. Quanhua Sun [37].

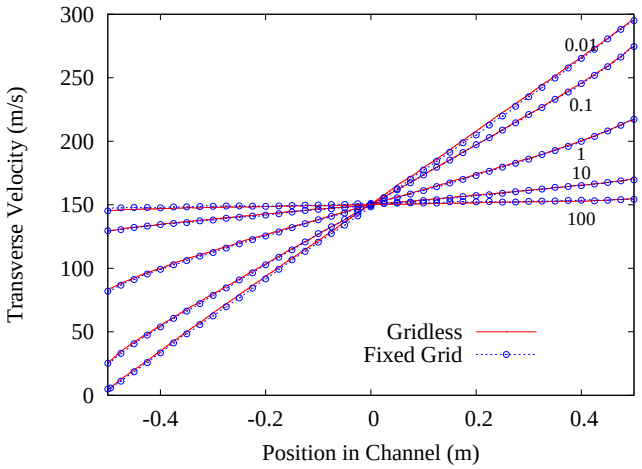

Fig. 17. Velocity profile of a Couette flow simulation. The particles are initially uniformly distributed between the two plates, the transverse (to the channel) velocity of each plate is $\pm 150 \frac{\mathrm{m}}{\mathrm{s}}$ respectively. Fixed grid data courtesy of Dr. Quanhua Sun [37].

Temperature profiles of this simulation are compared to similar fixed-grid results in Fig. 16 and velocity profiles are likewise compared in Fig. 17. From each of these two figures, we can see a very good agreement with the fixed

grid data for the higher density cases. Similar to the thermal Couette flow, there is a slight disagreement in the low to very rarefied cases ($Kn = 10$ and $Kn = 100$). Looking at a closeup of the temperature profiles in these low density cases, shown in Fig. 18, there appears to be slightly more curvature in the gridless results than in the fixed-grid results. This appears to be the case even for a much lower number of averages in the gridless case ($\sim 5 \times 10^4$ for $Kn = 10$).

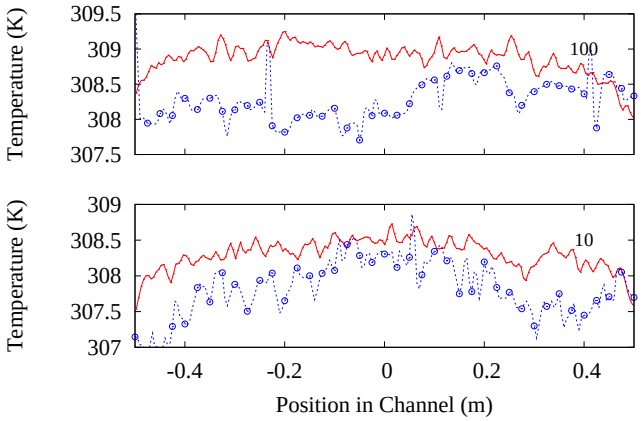

Fig. 18. Temperature profile of a Couette flow simulation for significantly high values of Kn. Fixed grid data courtesy of Dr. Quanhua Sun [37].

The differences between the sets of results of these two Couette-flow tests cannot be easily attributed to systematic error in the gridless approach. This is because for one case, the simulation appears to be slightly more collisional (the case of velocity diffusion where curvature in the temperature profiles is seen), while for the other case the gridless simulation appears to be slightly less collisional (the case of thermal diffusion where the results more closely resemble a free molecular flow). In spite of these initial findings, a refinement study is needed to determine the relative performance of the fixed-grid to a gridless system for the very rarefied case. More alignment in the timing of sampling for averages between the two methods will need to take place.

3.4 Low-velocity flow past a thin plate

We also demonstrate simulations of low-velocity ($Ma \approx 0.13$) flow past a thin, flat plate. As depicted in Fig. 19, a thin, diffusely reflecting plate is placed within a stream of Ar gas at $T = 273$ K and at an angle θ with respect to the stream velocity $\vec{\mathbf{v}}_0 = 40$ m/s $\hat{\mathbf{x}}$. In this paper we present the results for $\theta = 0$ and compute drag coefficient C_D of the plate. We compare C_D to values computed by various grid-based calculations, including a set of comparative calculations from Ref. [8] and a DSMC code implemented by G. A. Bird called

DS2V [38] which has the option of adapting its mesh after each run for subsequent runs. Ref. [8] presents a comparison of C_{D} as calculated by three different means: traditional DSMC, DSMC with information preservation (IP-DSMC), and a non-statistical model known as the transition probability matrix model (TPM).

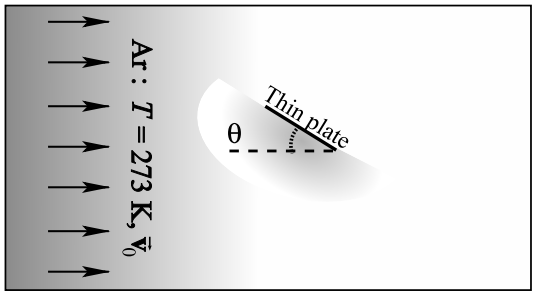

Fig. 19. A 2D flow past a thin plate using gridless DSMC.

A thin plate of length L is placed centered at the lower boundary of a simulation domain of size $6L \times 6L$. The boundary wherein the plate is embedded is treated as a specular reflecting surface. This mirrors the physics of the simulation into a virtual domain below the plate and allows us to properly simulate the infinitesimally thin plate completely surrounded by Ar gas. The boundary opposite of the plate is set to be diffusely reflecting with a velocity equal to $\vec{\mathbf{v}}_0$ such that it emulates a completely thermalized and uniform section of the Ar stream above the simulation domain. With $\vec{\mathbf{v}}_0$ parallel to the top and bottom boundaries, atoms are injected and removed from the left and right boundaries at a rate that sustains the thermal stream of Ar gas. We should note that because of technical details, the flat plate in the DS2V calculation is placed in the center of the computational domain with specularly reflecting walls parallel to the stream velocity.

For this simulation $L = 0.5$ m, $Kn = 0.1$, and $\langle N \rangle_{\mathrm{t}} = 5 \times 10^5$ where each particle represented $F_{\mathrm{N}} = 4.66 \times 10^{14}$ Ar atoms. With the domain initially filled, the simulation was allowed to run to steady state, after which 1×10^5 samples of $f_\delta(\vec{\mathbf{x}}, \vec{\mathbf{p}}, t)$, each separated by $\approx 6.8\Delta t$, were averaged together as in Eq. (10). Fig. 20 shows contour plots of the number density and velocity components of the flow past the thin plate.

To compare the results of this simulation to those produced by various grid-based calculations, we compute the drag coefficient of the thin plate. The drag coefficient C_{D} is given by

$$C_{\mathrm{D}} = \frac{2\vec{\mathbf{F}} \cdot \vec{\mathbf{v}}_0}{AMn\left|\vec{\mathbf{v}}_0\right|^3} \tag{30}$$

where $\vec{\mathbf{F}}$ is the force on the plate due to collisions with the gas, A is the area of the plate, and M is the molecular mass of the gas. For the system with results shown in Fig. 20, $A = 0.5$ m^2, $\vec{\mathbf{F}} \cdot \vec{\mathbf{v}}_0 / v_0 = 1.44$ mN, M is the atomic mass of

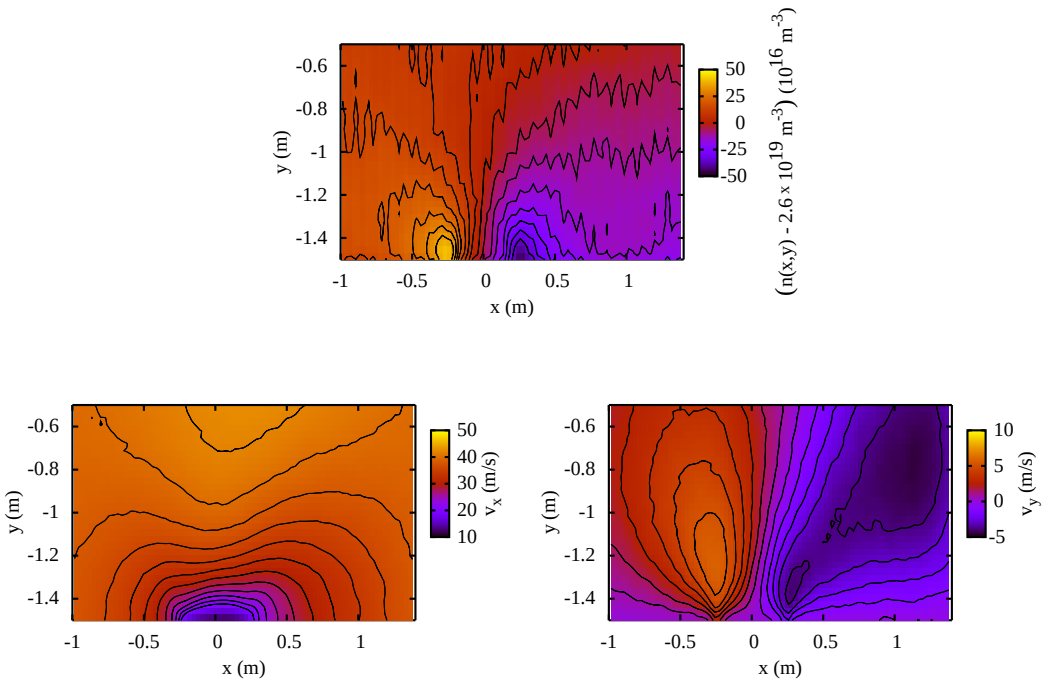

Fig. 20. Results for flow past a flat plate with zero angle of attack. The undisturbed stream of Ar gas has a flow velocity of $|\vec{\mathbf{v}}_0| = v_\mathrm{x} = 40$ m/s and density such that $Kn = 0.1$. Top: number density in region near plate; contours represent changes by 5×10^{16} m^{-3}. Lower-left: x component of the stream velocity; contours are separated by 2 m/s. Lower-right: y component of stream velocity; contours are separated by 0.9 m/s.

Ar (39.948 amu), $n = 2.6 \times 10^{19}$ m^{-3}, and $\vec{\mathbf{v}}_0 = 40$ m/s $\hat{\mathbf{x}}$. This corresponds to $C_\mathrm{D} = 2.1$. Tab. 1 shows C_D for $0.05 \leq Kn \leq 10$ for gridless DSMC and all the previously mentioned grid-based calculations.

As shown in Tab. 1, the gridless C_D compares relatively well with the values from the grid-based calculations for most of the range of Kn. However, as Kn increases, a difference arises between the gridless value and those from the other DSMC calculations, including DSMC of Ref. [8], DS2V DSMC, and IP-DSMC. We performed a variety of tests to discover where the discrepancy between these calculations may lie. These tests included a refinement study in total number of simulated particles, changing the boundary conditions such that both walls parallel to the flow were specularly reflecting, and increasing the boundaries of the computational domain up to two-fold for both the gridless and DS2V cases.

The first two tests were primarily conducted using the gridless code. For the first test, particle number refinement, we noticed very little change in the computed value of C_D over a range of 1×10^5 to 1×10^6 particles with a spread in values of less than 1%. For the second test, we changed the boundary

Kn	C_{D}				
	Gridless DSMC	TPM	DSMC	DS2V DSMC	IP-DSMC
0.05	1.5	1.6	1.5	1.5	1.5
0.1	2.2			2.3	
0.2	3.1	3.1	3.0	3.0	3.0
0.8	3.8	3.7	4.3	4.3	3.9
1.2	3.9	4.1	4.8	4.4	4.7
10	4.3	4.4		4.6	

Table 1
Drag coefficient (C_{D}) for flow past a flat plate at $\theta = 0$ incidence angle as computed by gridless DSMC, two versions of traditional DSMC (DSMC and DS2V-DSMC), DSMC with information preservation (IP-DSMC), and a non-statistical model known as the transition probability matrix model (TPM) [8].

condition in the gridless simulation for the wall opposite the thin plate to specularly reflecting, as was done in the DS2V case. Similar to the particle number refinement, we saw very little change in the computed value of C_{D}.

For the third test, we increased the size of the domains in both the DS2V and gridless codes. With the gridless code, we noticed no clear changes in C_{D} even for a domain as large as $6L \times 12L$, with the largest distance in the transverse direction. With DS2V, we did notice a general decrease in C_{D} as the domain size increased. For $Kn = 10$, the value decreased from $C_{\mathrm{D}} = 4.8$ with a domain size of $6L \times 6L$ to $C_{\mathrm{D}} = 4.6$ with a domain size of $6L \times 12L$, again with the largest distance in the transverse direction.

During this series of tests, we made two other key observations concerning the differing computed values of C_{D}. First, we noticed a difference between the drag computed by DS2V before and after adaptive mesh refinement. In each of the rows for the DS2V results, at least one iteration of adaptive mesh refinement occurred. We noticed a decrease in C_{D} after refinement for two values of Kn: $C_{\mathrm{D}} = 2.5 \rightarrow 2.3$ for $Kn = 0.1$, $C_{\mathrm{D}} = 3.4 \rightarrow 3.0$ for $Kn = 0.2$, but little increase in C_{D} for $Kn = 10$. The pre-adapted values of C_{D} for $Kn = 0.8$ and 1.2 were not recorded.

We also noticed that there was a difference between how the averaged value of C_{D} changed as more and more samples were accumulated. For gridless DSMC, the average value did not change significantly after the first 100 samples, although the statistical noise of the value became much reduced. For DS2V on the other hand, we noticed that the value tended to increase as more and more samples were taken. For example, for $Kn = 10$ at sample number 400,

C_{D}, as computed by DS2V, was 4.4 and steadily increased over the next 500 samples after which it remained at 4.6. We conclude that there are likely many factors that affect the computed value of C_{D} and that these additive effects become more prominent for large values of Kn.

To further demonstrate the utility of gridless DSMC, we turn to hypersonic flows. As objects travel through a medium at hypersonic speeds, a shockwave follows the disturbance wherein density, temperature, and velocity gradients are large. To simulate such a flow accurately, a DSMC code must take into account the density variations when selecting pairs of particles for collisions. As discussed in Sec. 2.2.5, the selection process must ensure that collision partners are not separated by a distance larger than a fraction of the mean free path λ_{MFP}. For flows with large density gradients, this translates into large variations in the scale size Δ of the clusters from which collision pairs are taken. This represents the most intensive mesh refinement needed for grid-based DSMC and the case which stands to benefit most from a gridless routine such as presented here.

We present two simulations of flow of Ar gas around rectangular and non-rectangular objects. For both simulations, the undisturbed stream velocity is set to $\sim$ 3000 m/s. With an input temperature of 273 K, this corresponds to a Mach number of $Ma = 10$. Simulation results using the gridless DSMC are compared to those of a standard gridded DSMC as implemented by G. A. Bird's DS2V program [38]. The DS2V code has also been subject to validation tests of hypersonic flow and comparison with other codes [39].

3.5.1 Flow past a square cylinder

Our first hypersonic flow consists of a 0.5 m wide square cylinder embedded in the flow of Ar moving at a speed of $\sim$ 3043 m/s with a stream number density equal to 2.6×10^{19} m^3. This corresponds to a Knudsen number of $Kn = 0.1$. The computational domain is set to be 8 m wide (in the direction transverse to the flow velocity) and 3 m long with the embedded square cylinder at the center of the domain.

The simulations begin with an empty computational domain which fills until steady state is reached. After steady state is reached, a series of samples are taken and averaged together to produce the final results. The results in this section represent an average of 4200 samples for gridless DSMC which used 2.3×10^6 particles (on average) and 4239 samples for the DS2V case which used 1.4×10^6 particles (on average). The DS2V simulation underwent one iteration of adaptive mesh refinement. To compare the results of the two simulations, we examine contour profiles of the number density, Mach number, and temperature. For clarity, only a subset of the computational domain is shown such that the bow shock developed by the embedded object is most visible.

The results of these two simulations show little difference in the density and Mach number profiles as shown in Figs. 21 and 22. Some differences between the quality of the plots (*i.e.* apparent noisiness of the data) may be most likely attributed to differences in the sampling routines. Also qualitative, the gridless DSMC method appears to demonstrate exceptional symmetry in the results. The most significant differences between the results, albeit also relatively minor, can be seen in the profiles of the Ar temperature (see Fig. 23) behind the object in the flow direction. It is possible that these minor differences are also the result of the different sampling routines in the test cases.

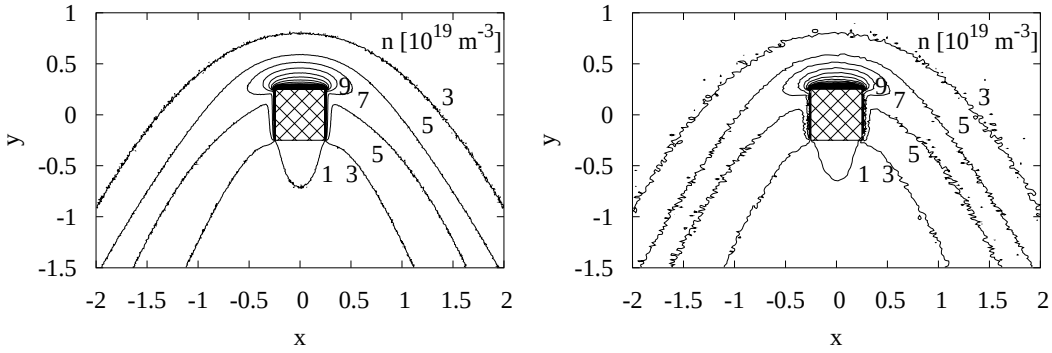

Fig. 21. Number density for hypersonic flow past a square cylinder. Left: gridless DSMC. Right: DS2V.

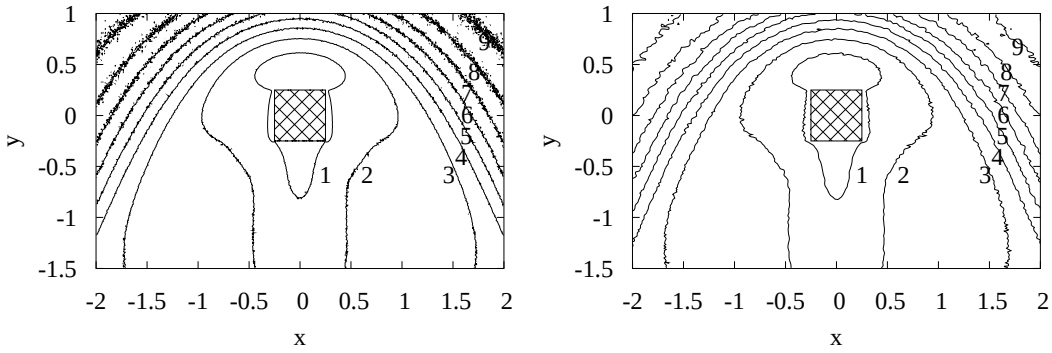

Fig. 22. Mach number for hypersonic flow past a square cylinder. Left: gridless DSMC. Right: DS2V.

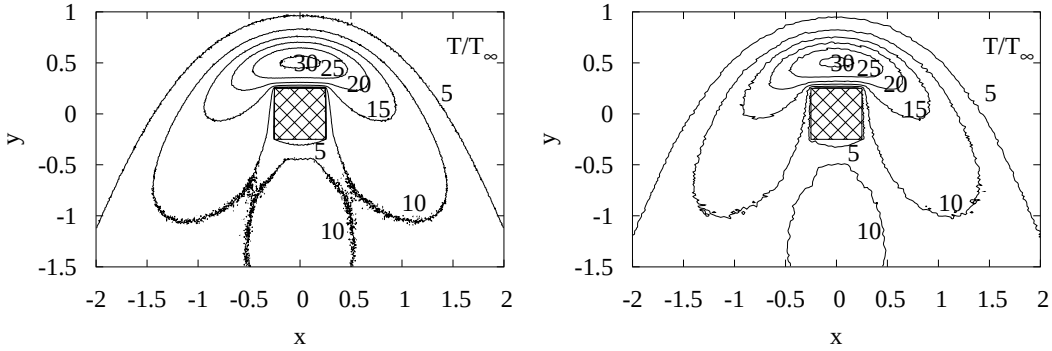

Fig. 23. Temperature for hypersonic flow past a square cylinder. Left: gridless DSMC. Right: DS2V.

As discussed in Sec. 2.2.5, the collision selection system must ensure that collision partners are separated by only a fraction of the mean free path λ_{MFP}. In gridless DSMC, we do this by engineering the size of the octree leaf-node, or dynamic collision cell, according to λ_{MFP}. We attempt to achieve a flat profile in the leaf-node scale size Δ to λ_{MFP} ratio with a maximum value given by $(\Delta/\lambda_{\mathrm{MFP}}) \lesssim 0.5$. In the left panel of Fig. 24, we show the mean value of this ratio for the hypersonic flow past a square cylinder. As can be seen in the figure, the profile is relatively flat throughout the simulation domain, even across the bow shock in the flow. This demonstrates two key aspects of the gridless technique described in this paper. First, the leaf nodes can be made uniformly small enough to ensure that only probable collisions occur, and second, the leaf nodes can be made to be no smaller than necessary. This second key point is important to ensure that the simulation does not force an unphysical thermal isolation.

The right panel of Fig. 24 shows the validity condition reported by the DS2V program to the user. For flatness comparison, the data in the right panel were normalized to twice their average value. DS2V retains information per grid cell about the ratio of the average separation of collision partners to λ_{MFP} and warns the user as this value approaches unity or greater. Although Fig. 24 shows that this validity condition is routinely met in the results of the DS2V version of this simulation, Fig. 24 also shows the lack of uniformity of the condition. Barring further mesh refinement, it is possible that the cell sizes become smaller than necessary and begin thermally isolating particles in the stream.

It should be noted that the difference in smoothness of the two profiles in Fig. 24 is expected. The DS2V data in the right panel result from averaging over many collisions through simulation time whereas the gridless DSMC validity metric in the left panel represents an instantaneous state of the generated octree at a given time in the simulation.

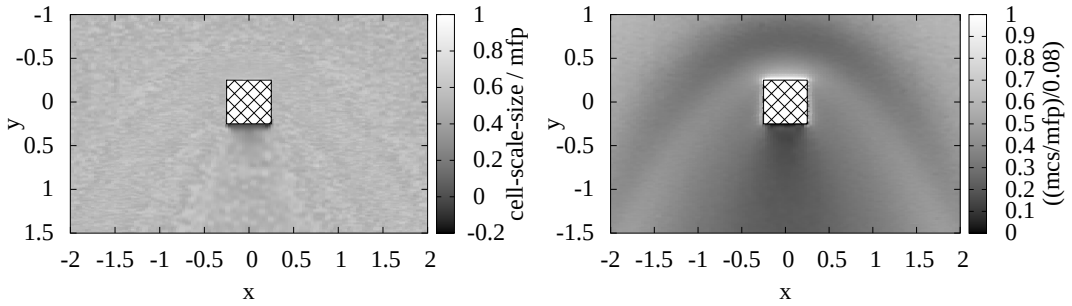

Fig. 24. Validity metric for DSMC. Left (gridless): ratio of the scale size of the leaf nodes of the tree to the mean free path $(\Delta/\lambda_{\mathrm{MFP}})$. Right (DS2V): ratio of mean collision separation to λ_{MFP} renormalized for visualization to twice the average value (2×0.04).

3.5.2 Flow past a double-flare

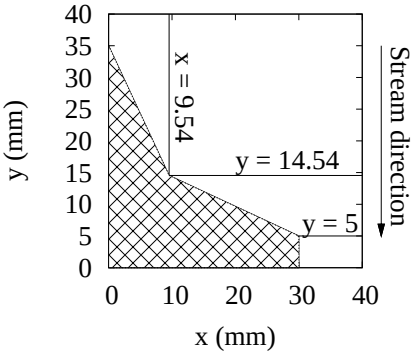

Fig. 25. Dimensions of the double flare.

Our second hypersonic flow resembles a set of simulations and experiments of flow past a biconic cylinder as described in Refs. [39–43]. A representative half cross-section of the biconic body is shown in Fig. 25. The biconic cylinder, placed in a hypersonic flow condition, creates sharp bow shock that exhibits very large density, velocity, and temperature gradients. The work represented in Refs. [39–43] compares various axially symmetric simulations of the flow to a set of experimental data.

Because our code is not currently set up to work in an axi-symmetric geometry, we instead simulate a hypersonic flow in two dimensions around an object of similar cross-section. Although we cannot compare with experimental results as is done in Refs. [39–43], we compare the results from gridless DSMC to those of the DS2V program as done in the previous section. We note that DS2V was used in Ref. [39] to simulate flow past the double cone and compared well with both experimental data and previous simulation results.

The geometry of the double flare object in the simulations presented in this section is as shown in Fig. 25. The undisturbed stream velocity and number density are 3000 m/s and 2.6×10^{21} m^{-3} respectively. The computational domain is set to be 20 cm along the direction of the flow velocity and 25 cm in the transverse direction. The object is embedded into the center of a specularly reflecting wall parallel to the stream velocity.

Similar to the simulations of flow past the square cylinder, the simulations begin with an empty computational domain which fills until steady state is reached. The simulations are allowed to reach steady state, after which a series of samples are taken to form the final result. For the gridless DSMC case, the results represent 2×10^4 samples averaged together with 2.6×10^6 particles in the simulation. The DS2V simulation underwent one iteration of adaptive mesh refinement and its results are an average of 21779 samples with 1.6×10^6 particles.

To compare the results of the two simulations, we again examine contour profiles of the number density, Mach number, and temperature. From Figs. 26, 27 and 28 we see that the comparison is generally favorable, although differences do appear in portions of the flow, especially behind the object.

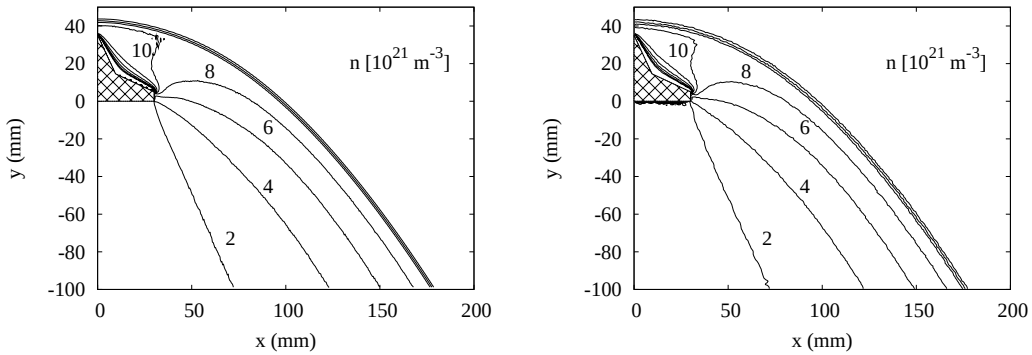

Fig. 26. Number density for hypersonic flow past double flared object. Left: gridless DSMC. Right: DS2V.

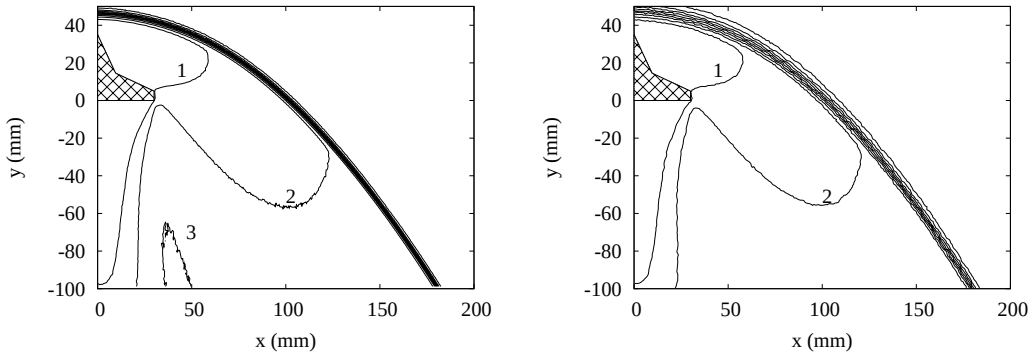

Fig. 27. Mach number for hypersonic flow past a double flared object Left: gridless DSMC. Right: DS2V.

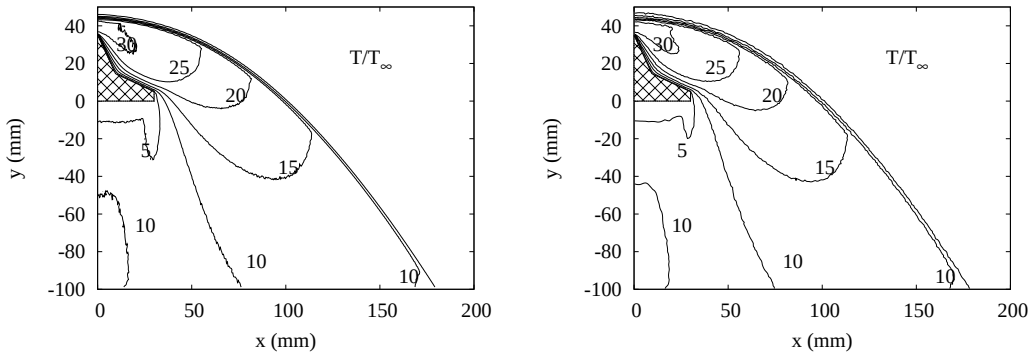

Fig. 28. Temperature for hypersonic flow past a double flared object Left: gridless DSMC. Right: DS2V.

The differences in this case are a slightly more pronounced than for the flow past a square cylinder as seen in Figs. 26, 27, and 28. In Fig. 26, for example, upper tip of the contour for 10×10^{21} m^{-3} is slightly pulled further back in

the gridless case. For Mach number and temperature, the most significant differences occur behind the object in the flow direction as seen in Figs. 27 and 28. Although sampling may play a small role for this comparison, to correctly place the blame for the mismatch, we must examine the validity factors for each of the two simulations. In Fig. 29, we again show the validity metrics for each of the simulations, with the gridless version in the left panel and the DS2V version in the right panel. Similar to the previous section, for visualization, the DS2V quantity (mean collision separation to λ_{MFP} ratio) is normalized by twice the undisturbed stream value.

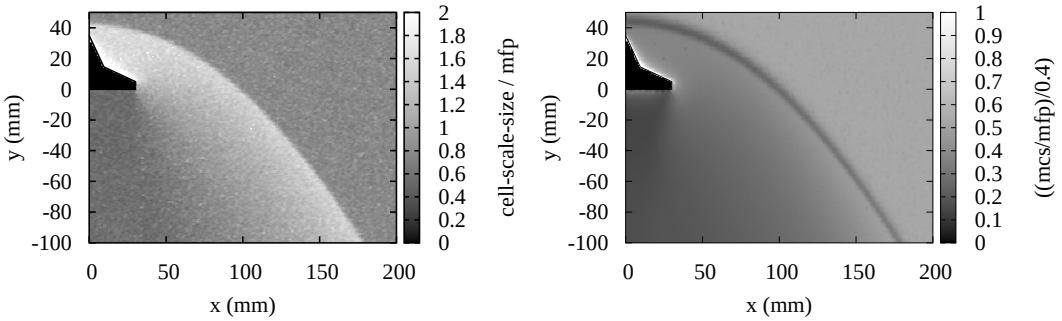

Fig. 29. Validity metric for DSMC. Left (gridless): ratio of the scale size of the leaf nodes of the tree to the mean free path ($\Delta/\lambda_{\mathrm{MFP}}$). Right (DS2V): ratio of mean collision separation to λ_{MFP} renormalized for visualization to twice the average value (2×0.2).

One of the first things to note from Fig. 29, is that neither of the two simulations is completely valid according to the given tests. In the gridless case, we see that the simulation was nearly unable to make any leaf node small enough such that $(\Delta/\lambda_{\mathrm{MFP}}) \lesssim 0.5$. This means that more particles were needed to allow correct size adjustment for the leaf nodes. In the DS2V case, we see that there appear to be regions of good calculation, as well as regions of bad computational value. The bad regions are primarily very close the object, while the best regions are at, and around the shock wave front. Similarly, we conclude that more particles were needed in the simulation. We note that the DS2V program did issue a warning concerning the situation: "There is a region in the flow where the criterion for a good DSMC calculation is not satisfied, more molecules are desirable."

Although the validity tests for both simulations indicate that more particles are needed to create a valid calculation, the differences in the panels of Fig. 29 bring to light significant differences between the two approaches. First, although the gridless case had larger set of working particles (2.6×10^6 compared to 1.6×10^6 for DS2V), it appears that the only region of good computational value is just behind the double flared object. Second, although the DS2V code indicates good computational worth from the regions at the front of the shock wave, the gridless code shows this as among the worst regions in its simulation. In other words, the shock wave front increases the density such

that a cluster cannot be created with ≥ 4 particles such that $(\Delta/\lambda_{\mathrm{MFP}}) \lesssim 0.5$. This reinforces the discussion in the previous section that suggests that the cell sizes in the gridded calculation can possibly become too small.

To help correct this simulation, we can apply Eq. (14) to help determine the new number of particles needed to obtain correct collision selections. Being conservative for the sake of computational resources, let us set a target to obtain correct flow collisions up to a density of 2×10^{22} m^{-3}. Using Eq. (14), we find that we need $F_{\mathrm{N}} \approx 5 \times 10^{12}$ instead of 6.47×10^{13} as was used for the gridless simulation above. This means that we need to add a factor of ~ 13 more particles to the system.

Figs. 30, 31, and 32 show data from similar simulations that correctly resolve the flow using an order of magnitude more particles. In this case, the number of particles in the DS2V and gridless cases was the same. The data compare very well with a very few minor differences which are again most pronounced in the region behind the object in the flow direction. To verify the validity of this pair of simulations, we examine the validity metrics for the two methods in Fig. 33. Similar to Fig. 24 of the square cylinder simulation, Fig. 33 indeed gives evidence that there were enough particles to trust the collision statistics in the simulations. Also similar to the square cylinder example, we see in Fig. 33 that the instantaneous gridless metric in the left panel is very flat even through the steep density gradients of the shock front. The metric for the DS2V code on the other hand, shows a variation especially at the shock front. This implies that grid cells may have, in fact, been made too small, or at least are in danger of being made too small.

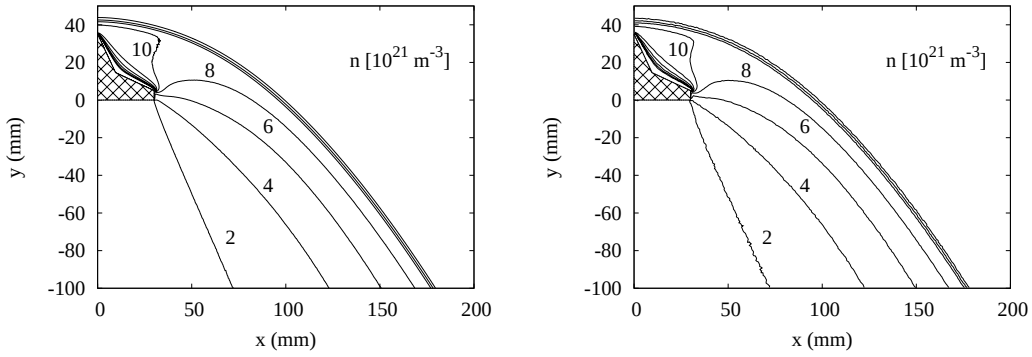

Fig. 30. Number density for hypersonic flow past double flared object. Left: gridless DSMC. Right: DS2V.

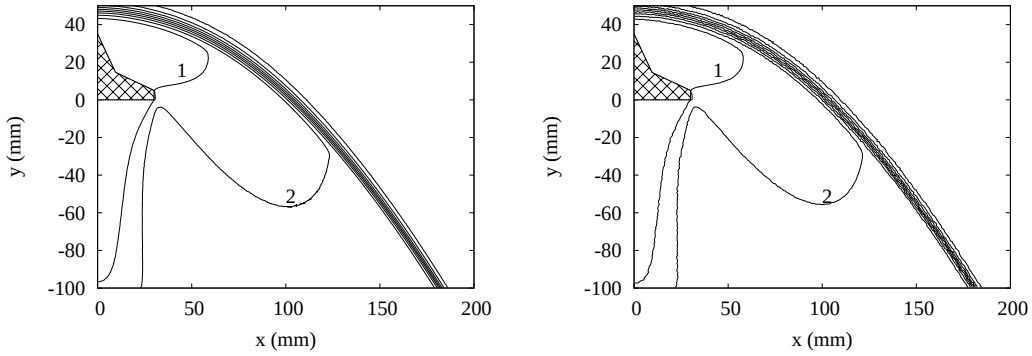

Fig. 31. Mach number for hypersonic flow past a double flared object Left: gridless DSMC. Right: DS2V.

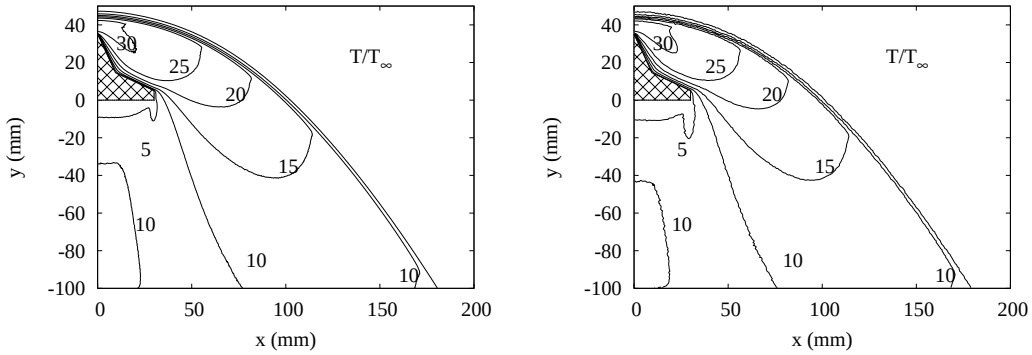

Fig. 32. Temperature for hypersonic flow past a double flared object Left: gridless DSMC. Right: DS2V.

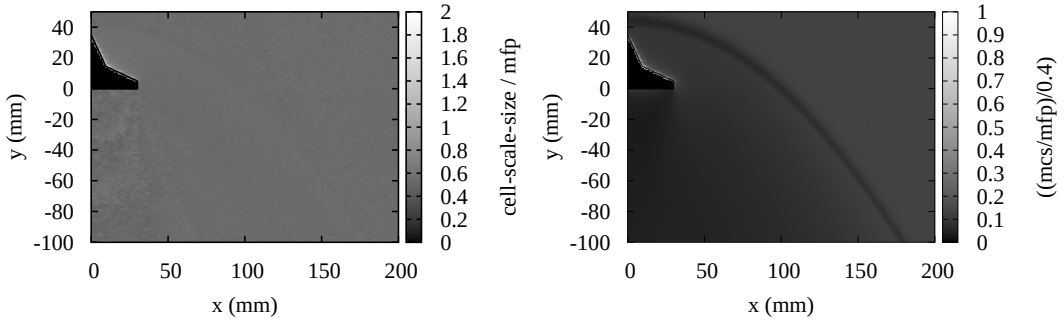

Fig. 33. Validity metric for DSMC. Left (gridless): ratio of the scale size of the leaf nodes of the tree to the mean free path ($\Delta/\lambda_{\mathrm{MFP}}$). Right (DS2V): ratio of mean collision separation to λ_{MFP} renormalized for visualization to twice the average value of the unresolved case shown in Fig. 29 (2×0.2).

4 Conclusion

We have developed a gridless method for DSMC. The method makes use of a hierarchical tree structure for determining collision partners in DSMC and a spherical spline interpolation for tracking local time averages of gas properties at the tree nodes. Initial tests indicate that in 1 and 2 dimensions, the method is at least as accurate as the traditional grid based approach. Further, it appears, that for a given accuracy, the gridless approach requires fewer samples in the averaging process than does the grid based method. This may be a result of the clustering method used or it could be attributed to the strong smoothing associated with spline interpolation and it requires further exploration. In addition, the novel treatment of boundaries provides a highly flexible tool, capable of handling flow past complex objects without needing to tune volumetric meshes.

We discuss the practicality of gridless DSMC with respect to computational resources (hardware and time). While the lack of a fixed mesh allows a minimal memory footprint, a demonstration of the execution time of the major components of the code shows that the primary bottleneck is in building the hierarchical octree. From this demonstration, we see that the CPU time per particle in the current implementation is less than an order of magnitude higher than the fastest grid based methods. Thus, we conclude that, based on timing alone, gridless DSMC can be competitive with traditional fixed grid approaches. We expect that with optimization, the speed of our gridless DSMC algorithm may be further increased.

We see several key advantages to the gridless approach that justify the moderate sacrifice in computational time. First, simulated particles are clustered based on local density. Thus, the nodal occupancy fluctuations should be minimized and improved collision statistics will result. Second, the abstraction of the DSMC algorithm from the physical system establishes an ease of use (*i.e.* simple reusability of code). Because of this ease of use, simulation development can focus on the physics of the problem at hand. Third, the abstracted DSMC layer can be wrapped inside an additional layer that forms the basis of a parallel computation scheme. This outer layer would be primarily for managing particle exchange and load balancing issues between different blocks of the underlying DSMC layer.

Gridless DSMC is currently being employed to simulate collision processes within ultra-cold (~ 10 μK down to nK regime) gas systems. This effort aims to model and evaluate the forced evaporative cooling process in novel atomic traps such as long ($\gtrsim 2$ m) magnetic atom guides [44] and dark, all-optical atom traps [45] to achieve sub-microkelvin temperatures. Preliminary results using gridless DSMC for forced evaporative cooling have been promising [46].

In addition to simulating neutral gas dynamics, this work represents a larger effort to develop a gridless methodology (and associated suite of software tools) that includes multi-scale plasma simulations. Initial effort focused on collisionless plasma simulations and the gridless approach of evaluating boundary integrals using a tree code (BIT) [29, 47, 48]. Using the methods from this work, a consistent strategy for including collisions in kinetic plasma simulations will follow. As such, the merger of BIT and gridless DSMC will provide a self contained tool for collisional plasma simulations.

5 Acknowledgments

S. E. Olson thanks support from the Army Research Office and Office of Naval Research (Project number 42791-PH) as well as the Michigan Center for Theoretical Physics. A. J. Christlieb would like to thank the Air Force Office of Scientific Research (grant numbers FA9550-07-0144 and FA9550-07-01-0092), AFRL-PRSA at Edward's Air Force Base, and AFRL-DEHE at Kirtland Air Force Base for support of this work. The authors would also like to thank Dr. Quanhua Sun for supplying results for Couette flow calculations using standard DSMC.

References

[1] S. Chapman and T.G. Cowling. *The Mathematical Theory of Non-Uniform Gases*. Cambridge University Press, Cambridge, England, 1952.

[2] W.G. Vincenti and C.H. Kruger. *Introduction to Physical Gas Dynamics*. John Wiley & Sons, New York, 1965.

[3] G. A. Bird. *Molecular Gas Dynamics and the Direct Simulation of Gas Flows*. Oxford University Press, New York, 1994.

[4] M. Combi. Time-Dependent Gas Kinetics in Tenuous Planetary Atmospheres: The Cometary Coma. *Int. J. Sol. Sys. Stud.*, 123:207–226, 1996.

[5] M. S.Ivanov, G. N. Markelov, S. F. Gimelshein, and S. G.Antonov. DSMC studies of high-altitude aerodynamics of reentry capsule. In *Proc. 20th International Symposium on Rarified Gas Dynamics, Beijing, China*, pages 453–458., 1997.

[6] L. Brieda, R. Kafafy, J. Pierru, and J. Wang. Development of the DRACO Code for Modeling Electric Propulsion Plume Interactions. In *40th Joint Propulsion Conference, Fort Laudardale, Florida, USA*, 2004.

[7] A. J. Christlieb and W. N. Hitchon. Three-dimensional solutions of the boltzmann equation: Heat transport at long mean free paths. *Phys. Rev. E*, 65(5):056708–+, May 2002. doi: 10.1103/PhysRevE.65.056708 .

[8] A.J. Christlieb, W.N.G. Hitchon, I.D. Boyd, and Q.H. Sun. Kinetic description of flow past a micro-plate. *J. Comp. Phys.*, 195:508–527, 2004.

[9] A. J. Christlieb, J. Rossmanith, and Smereka P. The broadwell model in a thin channel. *Comm. Math. Sci.*, 2:443–476, 2004.

[10] Huang Wu and Christopher J. Foot. Direct simulation of evaporative cooling. *J. Phys. B*, 29:321, 1996.

[11] E. Mandonnet, A. Minguzzi, R. Dum, I. Carusotto, Y. Castin, and J. Dalibard. Evaporative cooling of an atomic beam. *Euro. Phys. J. D*, 10(1): 9–18, March 2000.

[12] G. J. Parker, W. N. G. Hitchon, and J. E. Lawler. Self-consistent kinetic model of an entire DC discharge. *Phys. Lett. A*, 174:308–311, 1993.

[13] A.J. Christlieb, W.N.G. Hitchon, and E.R. Keiter. A computational investigation of the effects of varying discharge geometry for an inductively coupled plasma. *IEEE Transactions Plas. Sci.*, 28:2214–2231, 2000.

[14] J.P. Verboncoeur. Particle simulation of plasmas: review and advances. *Plas. Phys. Controlled Fusion*, 47:A231–A260, 2005.

[15] E.P. Muntz. Rarefied gas dynamics. *Ann. Rev. Fluid Mech.*, 21:387–417, 1989.

[16] W. Wagner. A convergence proof for Bird's direct simulation Monte Carlo method for the Boltzmann equation. *J. Stat. Phys.*, 66:1011–1044, 1992.

[17] N. G. Hadjiconstantinou, A. L. Garcia, M. Z. Bazant, and G. He. Statistical error in particle simulations of hydrodynamic phenomena. *J. Comp. Phys.*, 187:274–297, May 2003.

[18] M. A. Gallis, J. R. Torczynski, and D. J. Rader. Molecular gas dynamics observations of Chapman–Enskog behavior and departures there-

from in nonequilibrium gases. *Phys. Rev. E*, 69(4):042201–+, April 2004. doi: 10.1103/PhysRevE.69.042201 .

[19] G.A. Bird. Monte Carlo simulation of gas flows. *Ann. Rev. Fluid Mech.*, 10:11–31, 1978.

[20] G.A. Bird. Recent advances and current challenges for DSMC. *Comp. Math. Appl.*, 35:1–14, 1998.

[21] E.S. Oran andC.K. Oh and B.Z. Cybyk. Direct simulation Monte Carlo: Recent advances and applications. *Ann. Rev. Fluid Mech.*, 30:403–441, 1998.

[22] M. S. Ivanov and S. F. Gimelshein. Computational hypersonic rarefied flows. *Ann. Rev. Fluid Mech.*, 30:469–505, 1998.

[23] S. Rjasanow and W. Wagner. Time Splitting Error in DSMC Schemes for the Spatially Homogeneous Inelastic Boltzmann. *SIAM J. on Numer. Anal.*, 45:54–67, 2007.

[24] N. G. Hadjiconstantinou. Analysis of discretization in the direct simulation Monte Carlo. *Phys. Fluids*, 12:2634–2638, October 2000. doi: 10.1063/1.1289393 .

[25] A. L. Garcia, J. B. Bell, W. Y. Crutchfield, and B. J. Alder. Adaptive mesh and algorithm refinement using direct simulation Monte Carlo. *J. Comp. Phys.*, 154:134–155, sep 1999.

[26] J.-S. Wu, K.-C. Tseng, and F.-Y. Wu. Parallel three-dimensional DSMC method using mesh refinement and variable time-step scheme. *Comp. Phys. Comm.*, 162:166–187, oct 2004. doi: 10.1016/j.cpc.2004.07.004 .

[27] Philippe Lacroute and Marc Levoy. Fast volume rendering using a shear-warp factorization of the viewing transformation. In *SIGGRAPH '94: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, pages 451–458, New York, NY, USA, 1994. ACM Press. ISBN 0-89791-667-0. doi: 10.1145/192161.192283 .

[28] Matthew Moore and Jane Wilhelms. Collision detection and response for computer animation. In *SIGGRAPH '88: Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*, pages 289–298, New York, NY, USA, 1988. ACM Press. ISBN 0-89791-275-6. doi: 10.1145/54852.378528 .

[29] A. J. Christlieb, R. Krasny, J. P. Verboncoeur, J. Emhoff, and I. D. Boyd. Grid-free plasma simulation techniques. *IEEE Transactions Plas. Sci.*, 34:149–165, 2006.

[30] F. J. Alexander, A. L. Garcia, and B. J. Alder. Cell size dependence of transport coefficients in stochastic particle algorithms. *Phys. Fluids*, 10: 1540–1542, jun 1998.

[31] Robert J. Renka. Algorithm 661: Qshep3d: quadratic shepard method for trivariate interpolation of scattered data. *ACM Trans. Math. Softw.*, 14(2):151–152, 1988. ISSN 0098-3500. doi: 10.1145/45054.214374 .

[32] L. Hernquist and N. Katz. TREESPH – a unification of SPH with the hierarchical tree method. *Astro. J. Supp. Series*, 70:419–446, June 1989.

[33] E. Wild. On Boltzmann's equation in the kinetic theory of gases. *Proc.*

Cambridge Philos. Soc., 47:602–609, 1951.

[34] L. Pareschi and R. E. Caflisch. An Implicit Monte Carlo Method for Rarefied Gas Dynamics. *Journal of Computational Physics*, 154:90–116, September 1999.

[35] Lorenzo Pareschi and Giovanni Russo. Time relaxed Monte Carlo methods for the Boltzmann equation. *SIAM J. Sci. Comput.*, 23(4):1253–1273, 2001. ISSN 1064-8275. doi: 10.1137/S1064827500375916 .

[36] E. A. Carlen, M. C. Carvalho, and E. Gabetta. Central Limit Theorem for Maxwellian Molecules and Truncation of the Wild Expansion. *Comm. on Pure and App. Math.*, LIII:370–397, 2000.

[37] Quanhua Sun. *Information preservation methods for modeling micro-scale gas flows*. PhD thesis, University of Michigan, 2003.

[38] G. A. Bird. The DS2V/3V Program Suite for DSMC Calculations. In M. Capitelli, editor, *Rarefied Gas Dynamics: 24th International Symposium on Rarefied Gas Dynamics*, volume 762 of *American Institute of Physics Conference Series*, pages 541–546, May 2005. doi: 10.1063/1.1941592 .

[39] J. N. Moss, G. A. Bird, and G. N. Markelov. DSMC Simulations of Hypersonic Flows and Comparison With Experiments. In M. Capitelli, editor, *Rarefied Gas Dynamics: 24th International Symposium on Rarefied Gas Dynamics*, volume 762 of *American Institute of Physics Conference Series*, pages 547–552, May 2005. doi: 10.1063/1.1941593 .

[40] James N. Moss. Hypersonic flows about a 25° sharp cone. *NASA/TM-2001-211253*, December 2001.

[41] James Moss, Gerald J. LeBeau, and Christopher E. Glass. Hypersonic shock interactions about a 20°/65° sharp double cone. *NASA/TM-2002-211778*, August 2002.

[42] Graham V. Candler, Ioannis Nompelis, Marie-Claude Druguet, Michael S. Holden, Timothy P. Wadhams, Iain D. Boyd, and Wen-Lan Wang. CFD Validation for Hypersonic Flight: Hypersonic Double-Cone Flow Simulations. In *AIAA Aerospace Sciences Meeting & Exhibit, 40th*, volume 762 of *American Institute of Physics Conference Series*, January 2002.

[43] B. Chanetz, T. Pot, R. Benay, and J. Moss. New Test Cases in Low Density Hypersonic Flow. In A. D. Ketsdever and E. P. Muntz, editors, *Rarefied Gas Dynamics*, volume 663 of *American Institute of Physics Conference Series*, pages 449–456, May 2003. doi: 10.1063/1.1581581 .

[44] S. E. Olson, R. R. Mhaskar, and G. Raithel. Continuous propagation and energy filtering of a cold atomic beam in a long high-gradient magnetic atom guide. *Phys. Rev. A*, 73(3):033622, March 2006. doi: 10.1103/PhysRevA.73.033622 .

[45] S. E. Olson, M. L. Terraciano, M. Bashkansky, Z. Dutton, and F. K. Fatemi. Cold atom confinement in an all-optical dark ring trap. *Phys. Rev. A*, 76(6):061404, 2007. doi: 10.1103/PhysRevA.76.061404 .

[46] Spencer E. Olson. *Long, High-Gradient Magnetic Atom Guide and Progress Towards an Atom Laser*. PhD thesis, University of Michigan,

2006.

[47] A.J. Christlieb, R. Krasny, and J.P. Verboncoeur. Efficient particle simulation of a virtual cathode using a grid-free treecode Poisson solver. *IEEE Transactions Plas. Sci.*, 32:384–389, 2004.

[48] A. J. Christlieb, R. Krasny, and J. P. Verboncoeur. A treecode algorithm for simulating electron dynamics in a Penning-Malmberg trap. *Comp. Phys. Comm.*, 164(1-3):306–310, December 2004. doi: 10.1016/j.cpc.2004.06.076 .