Computers and Fluids 186 (2019) 1–14

Contents lists available at ScienceDirect

Computers and Fluids

journal homepage: www.elsevier.com/locate/compfluid

A novel spatio-temporally adaptive parallel three-dimensional DSMC solver for unsteady rarefied micro/nano gas flows

Mirvat Shamseddine, Issam Lakkis*

American University of Beirut, Lebanon

ARTICLE INFO

Article history:
Received 30 August 2018
Revised 16 January 2019
Accepted 8 March 2019
Available online 8 March 2019

MSC:
00-01
99-00

Keywords:
DSMC
Octree hierarchical grid
Spatio-temporal adaptive scheme
Unsteady
Parallel
Ray tracing
Rarefied gas flow

ABSTRACT

An efficient parallel multi-scale direct simulation Monte Carlo algorithm to simulate three-dimensional rarefied gas flows over complex geometries is presented. The proposed algorithm employs a novel spatio-temporal adaptivity scheme. Based on the gradients of flow macro-properties, the spatio-temporal adaptivity scheme computes the cell size distribution and assigns the appropriate number of time sub-steps for each cell. The temporal adaptivity scheme provides local time step adaptation through different temporal levels employed in different cells. Spatial representation is based on a hierarchical octree Cartesian grid with low memory storage requirement. The hierarchical octree grid endows the method with straightforward and efficient data management suitable for particle ray tracing and dynamic grid refinement and coarsening. Solid objects, represented by triangulated surfaces, are incorporated using a cut-cell algorithm. A new parallelization scheme suitable for simulating strongly unsteady, non-equilibrium flows is proposed. The parallelization scheme, implemented for multi-core Central Processing Units (CPUs), significantly reduces the computational cost of modeling these flows. Performance of the method is assessed by comparing with benchmarked test cases for various rarefied gas flows.

1. Introduction

Micro and nano-technologies are advancing rapidly, and the computational tools to predict the flow dynamics efficiently and accurately at these scales are continuously in demand. Gas flows in Micro and Nano devices are typically rarified and usually fall into the slip and transition flow regimes with a Knudsen number range $0.001 < \mathrm{Kn} < 10$ [1]. The direct simulation Monte Carlo (DSMC) method [2] is the most widely used computational tool for efficiently simulating fluid flows at these scales. The method has been successfully applied to investigate physical phenomena in a wide range of applications. These include shock waves [2], spacecraft aerodynamics [3], squeeze-films and oscillating microstructures [4], microsensors [5], microfluidics [6], and various rarefied flows in micro/nano-systems [7].

The geometry model in DSMC simulations refers to both the computational mesh of the flow domain and the surface representation of solid objects. Two primary approaches for the geometry model in existing state-of-the-art DSMC solvers have been used. These include body-fitted unstructured grids such as in MONACO [8] and dsmcFOAM [9], and Cartesian structured grids such as in DAC [10], SMILE [11], Bird's DSnV [12], MGDS [13], and SPARTA [14]. Table 1 gives an overview of the geometry and general features of commonly used DSMC solvers. These algorithms can run with a single processor or in parallel using the Message Passaging Interface (MPI) library on multiple processors. The spatial domain decomposition parallelization method is often used. This flavor of parallelization suffers from two drawbacks. The first is the challenge of load balancing and distributed storage of the computational domain. This challenge is particularly severe when the solver employs a spatio-temporally adaptive algorithm that dynamically adjusts the computational grid and the time step in response to the evolving flow field structures in unsteady flows. The second challenge is that the criteria for spatio-temporal adaptivity are based on macroscopic properties computed as averages over a statistically meaningful number of realizations (a realization is a DSMC simulation initiated from a unique random number generator seed). This latter observation suggests parallelization over independent realizations. In the proposed framework, each thread or core processes a realization of the simulation of the unsteady flow over the entire computational domain.

Bird [2] points out that dynamic grid adaptation is a main concern when applying the DSMC method to multi-dimensional problems and states that an ideal DSMC grid must fulfill three requirements: high computational efficiency, grid adaptation to

* Corresponding author.
E-mail address: issam.lakkis@aub.edu.lb (I. Lakkis).

https://doi.org/10.1016/j.compfluid.2019.03.007
0045-7930/

Table 1
Overview of geometry and general features of the well-known DSMC solvers and the presented one.

DSMC Algorithm	Grid Structured/Unstructured	Parallel Processor Distribution	Spatial adaptivity criteria grid adaptation	Temporal adaptivity criteria time step	Other
MONACO	Unstructured grid	Over grid	Based on λ	Single fixed time step	Localized data structure Ray tracing through an unstructured grid
dsmcFOAM	Unstructured grid	Over grid	Based on λ	Single fixed time step	openFOAM C++ toolbox
DAC	Two-level Cartesian grid	✗	Based on λ	Variable time step Variable particle weight	Cut-cell method Restricted distribution to US users Parallel processing capabilities
DSnV	Cartesian grid	✗	Based on N_{sim}	Variable time step	✗
SMILE	Cartesian grid	✗	Based on λ	✗	Cut-cell method Parallel processing capabilities
MGDS	Three-level Cartesian grid	Over grid	Based on λ	Variable time step Variable particle weight	Cut-cell method Ray tracing through a Cartesian grid
SPARTA	Hierarchical Cartesian grid	Over grid	Based on λ	Single fixed time step	Highly probable C++ Run one/multiple simulations simultaneously in parallel
Code Presented	Hierarchical Cartesian grid	Over realization	Based on λ & N_{sim}	Variable time step Different Temporal levels	Octree-based Cartesian grid Geometric tools in computer graphics Ray tracing through a structured grid Spatio-temporal adaptivity scheme

arbitrary geometries as well as to local flow conditions. Besides parallel computing implementation, fully automated mesh adaptation of the flow field can effectively save computational cost and provides efficient management of grid resolution. While time step adaptation depends on the local mean collision time, the adaptation of collision grid cells is based on the local mean-free-path, the number of simulated molecules, and other aspects such as the presence of surface meshes. The use of a single fixed time step is computationally inefficient when localized high-density gradients regions exist in the computational domain. In such flows, different time steps are needed in different parts of the domain, which calls for an automated adaptive time stepping scheme. With variable time steps, a single iteration of the DSMC algorithm no longer represents the same amount of physical time in each collision cell [15]. One way to tackle this challenge is proposed by Kannenberg and Boyd [15], where the disparity in the elapsed time is accounted for by weighting all particles by a time scale factor, defined as the ratio of the local time step to the reference time step for the simulation. The use of varying time steps according to this scheme increases computational efficiency by reducing the number of simulated molecules in the computational domain while maintaining relatively uniform molecule distribution and sufficient molecules for obtaining accurate collision statistics per cell. Implementations, such as DAC [10] and MGDS [13], that use variable scaling of particle weighting with spatially dependent time-steps are common. In the DAC algorithm, however, the particle weight and time step size vary independently. This requires cloning or deleting molecules to guarantee a balance of flux when the molecule crosses one collision cell to the next. A different approach for handling variable time steps was presented as a recent improvement to the DSMC algorithm in [16], and has been implemented in a recent work done by Wade et al. [17]. This approach is based on updating a desired local time step (DTS) for each collision cell, which is set to the minimum of user-specified fractions of the relative collision and transit times of the cell. A time parameter is assigned to all molecules and to all collision cells in which each molecule in a particular cell inherits the time step of that cell. The flow time is advanced in steps equal to the smallest value of DTS over all cells in the computational domain. The cell and molecule time parameters are advanced based on the flow time and the DTS values within the cells.

In this paper, we present a novel three-dimensional DSMC algorithm for simulating unsteady gas flows in complex domains. Novel aspects of the proposed algorithm include parallelization over realizations and a new spatio-temporal adaptation scheme. The flow domain is represented by a hybrid mesh consisting of a hierarchical octree-based Cartesian grid [18,19], whereas the surfaces of solid objects are represented by a triangular mesh. A cut-cell method to simulate flows around immersed objects of complex boundaries is implemented [20]. For near-boundary computational cells that are cut by the true physical boundary of the solid object, the method computes the effective volume of cut-cells for accurate prediction of molecular collisions and macroscopic properties in these cells. The cut-cell method also allows for decoupling of the flow field mesh from the solid boundaries surface mesh, making it suitable for simulating near-continuum flows with large density variations. The hierarchical octree-based Cartesian grid representation of the domain allows for efficient data storage and management that is compatible with the spatio-temporal adaptation scheme. When compared to unstructured meshes, such representation significantly improves memory requirement and is, therefore, more suitable for simulating large-scale DSMC problems. The hierarchical octree-based Cartesian grid representation also enables a potentially more general scheme for varying cell volumes over a large range of the molecular length scales. The hybrid mesh representation allows for simple integration of a variety of effective geometric tools used in computer graphics, including fast particle-tracing algorithms. This enables DSMC calculations to be performed with less number of operations, such as in successive grid adaptation, particle movement, and particle sorting. The dynamic spatio-temporal adaptation scheme is based on the local macroscopic flow properties which are computed as statistical averages over a number of realizations. Instead of advancing with most limiting (smallest) time step, the scheme handles spatial dependence of the time step by employing a number of discrete temporal levels. The method implements a smart algorithm that efficiently loops over these levels, in descending order of the time step size, where within each loop, all cells sharing the same time step are handled. An additional feature of the DSMC algorithm proposed in this work is that it is optimized for simulating unsteady flows in parallel over multiple cores. In contrast with distributing the computational domain over the cores (or threads, or CPUs), the independent realizations are distributed over the cores. Due to the lack of communication between the cores when each is handling an independent realization, the parallelization efficiency is almost 100%. Besides, this type of parallelization is optimal when simulating highly unsteady rarefied flows over complex geometries. These flows typically experience considerable variability in the spatial gradients of the macroscopic thermodynamic properties, and as such, spatial adaptation needs to be frequently carried out. With each core being assigned a DSMC realization, local flow properties at different time steps are collected simultaneously in parallel and averaged over the multicores for local mesh refinement. Table 1 presents an overview of the geometry and general features

of the proposed DSMC solver in comparison with existing state-of-the-art DSMC solvers.

This paper is organized as follows: Section 2 summarizes the DSMC methodology. In Section 3, we present the three-dimensional hybrid mesh scheme. Section 4 discusses the effective three-dimensional particle ray-tracing scheme. The spatio-temporal adaptivity scheme is presented in Section 5. Results for several benchmark DSMC simulations are presented in Section 6. Finally, Section 7 concludes the paper.

2. DSMC methodology

Among the particle simulation methods, DSMC, pioneered by G. A. Bird in the 1960s [2], is the most powerful numerical technique for the simulation of complex, non-equilibrium rarefied gas flows. The DSMC method emulates the physics of a real gas and provides a solution to the non-linear Boltzmann equation. It follows a representative set of randomly selected simulated molecules, each representing a large number of physical molecules, as they collide and move in physical space. The molecules' motion, their interactions with boundaries and intermolecular collisions alter with time their spatial coordinates, velocity components, and their internal energies. Molecular motions are modeled on a deterministic basis, while their collisions are treated on a probabilistic basis according to an appropriate collision model. The Simulation of the real gas flow is carried out by statistical sampling of the macroscopic flow properties in grid cells discretizing the physical space of the flow field.

Accuracy of the DSMC method is highly dependent on the discretization of space and time. The collision grid cell size, Δx, must be small compared to the local mean free path, $\lambda \sim |\pi/\nabla\pi|$; the length scale characterizing the spatial variations of the macroscopic properties, π. So, we choose $\Delta x \ll \lambda$, where λ generally varies with space and time. The simulation time step, Δt, over which molecular motions and collisions are uncoupled must be smaller than the local mean collision time, $\Delta t < \tau_c = \lambda/v_{\mathrm{mp}}$, where v_{mp} is the most probable velocity. The number of simulated gas molecules per cubic mean free path, N, must be larger than a minimum (typically 20 molecules) to preserve collision statistics and for the molecules to yield a reasonable approximation of the local velocity distribution function. Unless the gas is highly rarefied and the simulation domain is small, the constraints on Δx, Δt, and N make DSMC computationally expensive. Thus, adaptive techniques are required in hypersonic near-continuum flows spanning a wide range of length and time scales.

The interaction of simulated molecules with physical boundaries is an important boundary condition in DSMC simulations. Molecule-surface interactions include solid wall boundaries (e.g. thermal walls), periodic boundaries, and inflow/outflow boundaries. A given gas-solid surface interaction can be treated as being fully specular, fully diffuse, or a combination of the two. The inflow/outflow boundary conditions can accommodate supersonic and subsonic flows. They are implemented by injecting particles into the computational domain at the external flow conditions. Two methodologies are proposed for their implementation in DSMC: standard and reservoir method. The standard method involves a particle emission surface set at the flow boundary [2]. The well-known Maxwellian Reservoir method employs ghost cells at the boundaries of the DSMC computational domain that act as sinks and sources of the simulation particles [21]. The velocity of the molecules entering the flow field is generated according to the velocity distribution of the external flow.

Simulating molecular collisions is a statistical process which allows DSMC to achieve faster numerical performance than deterministic simulation methods such as molecular dynamics. Roohi et al. [22] reported a comprehensive review of the different collision models developed in the framework of the DSMC method. The popular collision scheme employed in DSMC is the No-Time-Counter (NTC) method introduced by Bird in 1994 and used in conjunction with the sub-cell method [23]. In the NTC collision procedure, the number of possible collision pairs that should be checked for collision within a collision cell of volume V_c over a time step Δt, is defined as:

$$N_{collisionpairs} = \frac{N\bar{N}F_N(\sigma_T v_r)_{\max}\Delta t}{2V_c}, \tag{1}$$

where N is the instantaneous number of simulated molecules in a cell, $\bar{N}$ is the time average number of simulated molecules in a cell, F_N is the number of real molecules represented by each simulated molecule, σ_T is the total collision cross section, and v_r is the relative speed in the collision. In 2007, Bird has proposed a modification to the NTC method where $N(N-1)/V_c$ replaces $N\bar{N}/V_c$ [24]. Thus, if Δt is kept constant and the volume V_c is divided into eight octants, we obtain $N_{collisionpairs} \sim 8(\frac{N}{8})(\frac{N}{8}-1)\,\Delta t/(\frac{V_c}{8}) \sim N(N-8)\Delta t/V_c$, which means that $N_{collisionpairs}$ is conserved as long as $N \gg 1$. Furthermore, if N and V_c are kept constant and Δt is divided by m, then $N_{collisionpairs} \sim N(N-1)m(\Delta t/m)/V_c$ and $N_{collisionpairs}$ is conserved if we carry m collisions.

Several collision models, designed to reproduce the real macroscopic flow behavior, were applied successfully to numerous DSMC simulations ranging from micro/nano flows to hypersonic flows. These models include the inverse power law model, the hard sphere (HS) model, the variable hard sphere model (VHS), and the variable soft sphere (VSS) model [2]. In the VSS model, the mean free path is given as:

$$\lambda_{\mathrm{VSS}} = \frac{4\alpha(7-2\omega)(5-2\omega)}{5(\alpha+1)(\alpha+2)}\left(\frac{\mu}{n}\right)(2\pi m k_B T)^{-1/2}$$
$$\mu = \mu_{ref}\left(\frac{T}{T_{ref}}\right)^{\omega} \tag{2}$$

where m is the molecular mass, $n = \rho/m$ is the number density, μ_{ref} is the viscosity at reference temperature, k_B is the Boltzmann constant, α is the scattering parameter ($\alpha = 1$ for VHS model), and T is the temperature. The viscosity index ω is the power exponent of temperature in the viscosity law given by $\omega = \frac{1}{2}\frac{\eta+3}{\eta+1}$. η is the repulsive power exponent in the inverse power law model, $F = K/r^{\eta}$, where F is the force, K is constant, and r is the distance between molecules. The values of ω and η for the HS model, the VHS model (and also the inverse power law IPL), and for the Maxwellian molecule are respectively 1/2 and ∞, 0.75 and 9, and 1 and 5 [25].

In the HS model, the mean free path is expressed as:

$$\lambda_{\mathrm{HS}} = \frac{1}{\sqrt{2}\pi d^2 n} \tag{3}$$

where d is the molecular diameter.

The mean collision time is related to the mean free path λ and to the most probable velocity $v_{mp} = \sqrt{2k_BT/m}$ by

$$\tau_c = \frac{\lambda}{v_{mp}} \sim \frac{\lambda}{\sqrt{T}} \tag{4}$$

It can be seen from the above equations that the mean free path and the mean collision time are inversely proportional to the number density as:

$$\lambda \sim \frac{T^{\alpha^*}}{n}$$
$$\tau_c \sim \frac{T^{\alpha^*-0.5}}{n} \tag{5}$$

where $\alpha^* \in [0, \omega - 1/2]$; $\alpha^* = 0$ for HS model.

The DSMC simulation starts from a set of prescribed initial conditions and proceeds in small time steps discretizing the physical

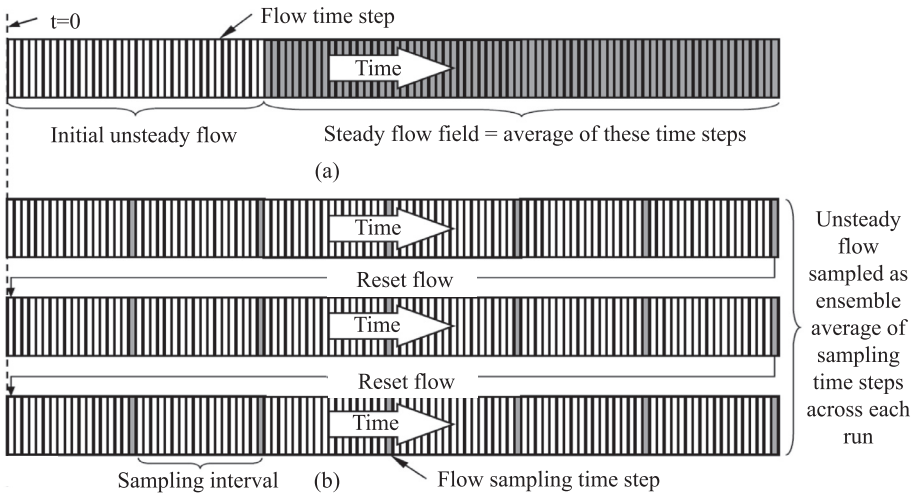

Fig. 1. (a) The schematic of the time-averaging of the flow properties over a long interval of simulation time. (b) The schematic of the ensemble-averaging of the flow properties over three independent DSMC simulations, each initiated from a different random number generator seed [26].

time in the real flow. Fig. 1 illustrates the two sampling methods used in DSMC: steady and unsteady sampling techniques. For predicting steady flows, each independent Monte Carlo simulation proceeds until a steady behavior is established at a sufficiently large time. Time-averaging of the macroscopic flow quantities over a number of time steps is required to reduce statistical fluctuations and obtain smooth results. For simulating unsteady flows, ensemble averaging of many independent Monte Carlo simulations, each originating from a different random number generator seed, is carried out to obtain final results with acceptable statistical accuracy. The flow field is sampled at the appropriate flow sampling time steps, denoted by dark bands in Fig. 1. Unsteady flow sampling to yield the macroscopic flow quantities at a given time requires averaging over all the independent realizations of the transient simulations at that time.

3. Three-dimensional hybrid mesh scheme

In the current study, an octree-based Cartesian grid divides the computational domain into cubic cells, whereas the surface of the 3D solid object is triangulated using the preprocessing open-source software SALOME 7.5.1 [27]. Tree-based methods with the simple Cartesian structure and embedded hierarchy make use of recursive encoding schemes. These schemes render processes such as mesh adaptation, rebuilding, data access, and handling of fluid-solid interaction both simple and efficient.

Each solid object is bounded by a rectangular box. An axis-aligned box-box intersection test [28] is then carried out to identify collision cells neighbors and all cubic cells that overlap with the bounding box surface. In addition, the fast 3D triangle-box overlap testing by Moller [29] is implemented to test overlapping between triangular elements of the solid object surface mesh and cubic cells inside the bounding box. This test enables linking each surface mesh triangular element to the overlapping Cartesian cells. Fig. 2(a) shows a schematic diagram illustrating the hybrid mesh scheme. The proposed three-dimensional hybrid mesh scheme employs a flexible data structure which enables simulation of particles movement and sorting processes with fewer operations, thereby reducing the CPU time.

Special treatment of the cells being crossed by the solid boundary, i.e., the so-called cut-cells, is also applied. Fig. 2(b) shows a zoomed-in view of the box bounding the solid object (sphere) and a cut-cell representation. The Monte Carlo random marker cut-cell method is implemented in this work [20]. It is used to estimate the cut-cell effective volume needed to accurately model collisions and predict the macroscopic properties in the cut-cell. This method is one of the two main cut-cell methodologies for DSMC simulations of rarefied gas flows around moving obstacles. It

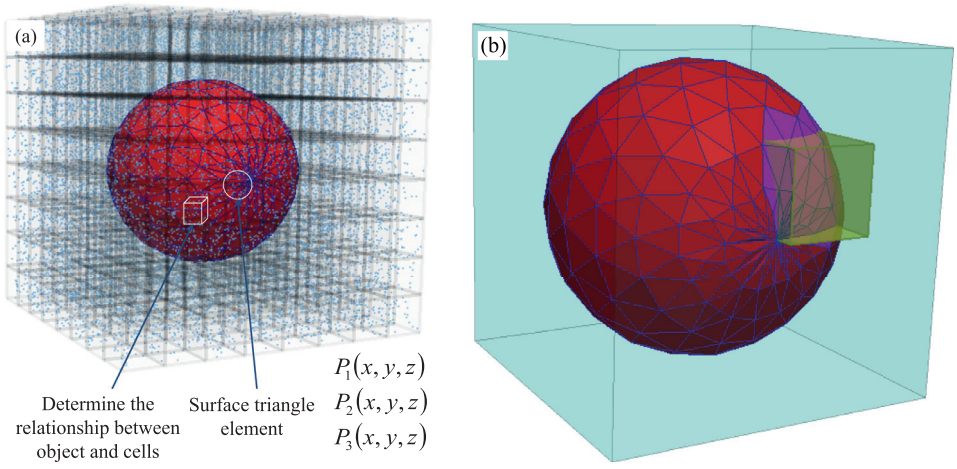

Fig. 2. (a) Schematic representation of a triangulated surface mesh of a sphere embedded in a 3D octree Cartesian grid. (b) Bounding box and a cut-cell representation.

performs two main functions: First, all triangular surface elements are sorted into the appropriate octree Cartesian cells within the geometry data structure. Second, a number of particles, N_p, is randomly generated within each cell with a volume V. Possible intersections between a ray going out from a particle and directed along the unit normal vector of a given surface triangle element are determined. If no intersections occur with all surface triangle elements within the cell, the particle lies inside the flow field. Then, the volume of the cut-cell is simply determined by dividing the fraction of particles determined to lie within the flow field, N_c, by the total number of Monte Carlo particles considered, N_p, as: $V_c = V\left(N_c/N_p\right)$.

4. Three-dimensional particle ray-tracing scheme

Molecular movement constitutes a significant fraction of the computational cost in a DSMC simulation. Simulated molecules move along linear trajectories defined in a vector form as $\boldsymbol{r}_f = \boldsymbol{r}_i + \boldsymbol{v}\Delta t$, where $\boldsymbol{r}_f$ is the final position of the particle, $\boldsymbol{r}_i$ is the initial position vector, and Δt is the simulation time step. To achieve both robustness and efficiency in tracking particle movement within the hierarchical octree-based Cartesian grid, a special particle ray-tracing technique is employed. This technique is used in the vicinity of the solid object surface where the region of the bounding box is treated as follows. During a single time step, a molecule cannot move more than one collision cell size along each dimension (a DSMC time constraint). Ray tracing is performed only for particles that leave their assigned cell and intersect the box bounding the solid object. If an intersection with the bounding box occurs, a cell-by-cell particle tracking procedure is performed to determine whether the particle reaches a boundary surface triangle, stays in or leaves the current cell. If no ray-triangle intersection occurs, the particle's position is updated if the particle stays in the current cell; otherwise, ray-box intersection tests with all possible neighbour collision cells are performed to track the particle from the current cell to its nearest neighbour collision cell. The particle-tracking algorithm is then invoked again to move the particle over the remainder of the time step. At the completion of the molecular movement phase, each particle is automatically stored within its final cell by the sort subroutine. In summary, the hierarchical octree-based data structure allows for efficient intersection testing within the ray tracing algorithm. We point out here that the grid cells are axis-aligned boxes whose edges are all parallel to the basis vector, which enhances the efficiency.

5. Spatio-temporal adaptivity scheme

A unique feature of the proposed algorithm is that it runs transient parallel Monte Carlo simulations simultaneously and independently on multicore CPUs. Most DSMC solvers are parallelized through decomposition of the physical domain into groups of cells that are distributed among the processors. The efficiency of such parallelization scheme may suffer due to the intensive communications between the processors and load imbalance among the processors. The spatial domain decomposition parallelization scheme is convenient for simulating low speed flows where a uniform Cartesian grid is used, and high-gradient flows where grid resolution in both space and time is evoked once before steady state is reached. In contrast, the proposed parallelization scheme is more suitable for simulating highly unsteady rarefied flows. The length and time scales in such flows vary considerably over the domain. Thus, frequent spatio-temporal adaptation for variable resolution of the different flow regions is required. The upper diagram in Fig. 3 shows a schematic representation of the proposed

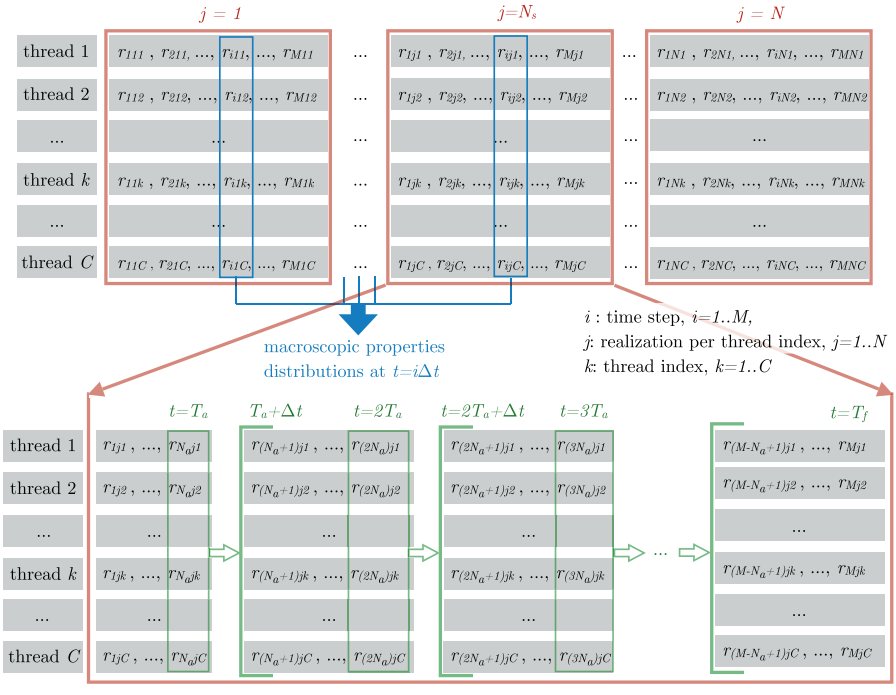

Fig. 3. The schematic of running transient DSMC simulations on different threads. Top Diagram: Each thread runs sequentially N realizations of the transient simulation consisting of M time steps. N_sC realizations r_{ijk}, $j = 1 \ldots N_s$, $k = 1 \ldots C$, are averaged every output time interval, $T_o = N_o\Delta t$, to compute the macroscopic properties distributions at output time step N_o. The decision to include a new set of N_sC transient simulations (increment N by N_s) is based on the relative statistical difference between macroscopic properties of the last NC realizations and the previous $(N - N_s)C$ realizations. Note that N is an integer multiple of N_s. Bottom Diagram: The spatio-temporal adaptivity is carried out for all threads every $N_a = T_a/\Delta t$ time steps. C realizations r_{ijk}, $k = 1 \ldots C$ are averaged (by the sniffer) at $t = T_a, 2T_a, \ldots, T_f$ (corresponding to time steps $i = N_a, 2N_a, \ldots, M$) to estimate the macroscopic properties distributions needed for the spatio-temporal adaptivity criteria. These criteria will set the grid size and the associated temporal levels distribution for the time intervals $[T_a + \Delta t, 2T_a]$, $[2T_a + \Delta t, 3T_a], \ldots$ While the sniffer carries out sampling of the microscopic properties and subsequently updates the distribution of cell sizes and temporal levels, the threads pause. Once the sniffer completes its task, the threads resume.

parallelization scheme using multiple threads. Each thread runs sequentially N realizations (each initiated from a unique random number generator seed) of the transient DSMC simulation consisting of M time steps. The realizations from different threads are averaged (by a SNIFFER algorithm) to compute the macroscopic properties distributions at flow sampling time steps. Due to the lack of communication between the threads when each is handling an independent realization, the parallelization efficiency is almost 100%.

The proposed algorithm employs a novel spatio-temporal adaptivity scheme to dynamically adjust local grid spacing and time steps, and to accurately resolve local flow features. The kinetic spatial scale, defined by the mean free path λ, and the temporal scale, defined by the mean collision time, $\tau_c = \lambda/\sqrt{2kT/m}$, are calculated according to the binary elastic collision model used in the DSMC simulation. Prior to the adaptation process, the DSMC simulation starts with a uniform mesh. As depicted in the bottom diagram of Fig. 3, spatio-temporal adaptivity is carried out for all threads every T_a, where T_a is the spatio-temporal adaptivity time interval. This parallel process is frequently interrupted by the SNIFFER (serial) algorithm to compute the macroscopic properties from the average of the realizations from different threads and subsequently set the grid size and the associated temporal levels distributions. The additional cost of the SNIFFER serial activity is very small and the measured parallelization efficiency of the parallelization scheme including the SNIFFER is more than 95%.

Spatial adaptation of collision cells follows the conventional DSMC constraint on the collision cell size, $\Delta x_c = \alpha_c \lambda$, where α_c is a user-defined collision cell size factor. The DSMC constraint on the minimum number of simulated molecules per collision cell, $N_{\min}$, must be preserved in spatial adaptation process to disallow collision cells with too few molecules. It is concluded from previous DSMC studies [30] that it is necessary that the cell size is constrained to be less than one-third of the local mean-free-path, $\alpha_c \in [1/4,\ 1/2]$, and the number of simulated molecules per cell should exceed 20 ($N_{\min} > 20$) for slip flows and 10 ($N_{\min} > 10$) for transition flows. Spatial adaptation involves two processes: refinement and coarsening. During the refinement process, the average of the local mean free path, λ_{av}, and the minimum number of simulated molecules over all CPU cores is computed for each collision cell. Then, each collision cell is tested for spatial adaptivity by computing the nearest division integer n_d, given by:

$$n_d = \frac{\log\left(\frac{\Delta x_c}{\alpha_c \lambda_{av}}\right)}{\log(2)} \tag{6}$$

The collision cell is refined into $(2^{n_d})^3$ new octants provided that n_d is greater than or equal to one and the minimum number of simulated molecules in the collision cell is greater than $N_{\min} \times (2^{n_d})^3$. In the coarsening process, the average local mean free path over all CPU cores is computed in each parent cell of eight collision octants. A parent cell is coarsened if its size is less than $\alpha_c \lambda_{av}$ or it has a child with few numbers of simulated molecules. The geometry octree data structure is then updated and simulated molecules are re-sorted into the new tree structure.

Temporal adaptation requires computing the desired time step, Δt_d, in every collision cell and updating the DSMC simulation flow time step. Usually, the desired time step is adapted to the minimum time between a specified fraction of the local mean collision time in each collision cell, $\Delta t_1 = \alpha_1 \tau_c$, and a specified fraction of the time needed for a molecule to travel a local collision cell size, $\Delta t_2 = \alpha_2\left(\Delta x_c / v_{\text{mp}}\right)$, as follows:

$$\Delta t_d = \min\left(\alpha_1 \frac{\lambda_c}{v_{\text{mp}}}, \alpha_2 \frac{\Delta x_c}{v_{\text{mp}}}\right) \tag{7}$$

Temporal adaptation algorithm:
If (inflow/outflow boundaries exist) then
- Generate/Move molecules at flow boundaries.
- Assign FLAG to molecules entering the domain. These molecules don't move while iterating over temporal levels.

End if
Do l=0, N_l-1
  Do j=1, 2^l
    - Move Molecules($m, \Delta t_l$): move molecules in cells in temporal level $m \geq l$ with a time step Δt_l.
      - Ray-Trace molecule movement
      - Carry out molecule-surface interactions
      - Update molecular position
    - Sort Molecules
    - Collide Molecules($l, \Delta t_l$): collide molecules in cells in temporal level l.
      - Multiple collisions

  End Do
  - Assign FLAG '*' to molecules initially at level 'l' and moved to level '$m>l$'. These molecules don't move in the next iterations of the loop.
  - Retain molecules initially at level '$m>l$' to their initial position.

End Do
Global Sort (loop over all cells/molecules)

Fig. 4. Temporal adaptation algorithm within the DSMC code. N_l: number of temporal levels in the domain; 2^l: number of time steps in temporal level l; Δt_l: time step in temporal level l.

where, according to [30], $\alpha_1,\ \alpha_2 \in [1/3,\ 1/2]$. The DSMC simulation flow time step is updated according to a user-defined criterion. One possible criterion, used by Bird in 2007 [24], is to choose the flow time step to be equal to the smallest value of the desired time step among all collision cells. In the present study, different temporal levels are considered in the computational domain where the number of temporal levels, N_l, is computed from the minimum, $\Delta t_{d\min}$, and maximum, $\Delta t_{d\max}$, desired time steps as follows:

$$\frac{\Delta t_{d\max}}{\Delta t_{d\min}} = 2^{N_l} \tag{8}$$

Each temporal level l, $l = 0 \ldots N_l - 1$, is characterized by 2^l time steps. Collision cells are assigned to different temporal levels such that $\Delta t_l = \frac{\Delta t_{d\max}}{2^l} \leq \Delta t_{dc}$, where Δt_l is the time step in temporal level l and Δt_{dc} is the desired time step in collision cell c. After grouping collision cells into different temporal levels, only levels populated by 10% or more collision cells are considered in temporal adaptation. The DSMC simulation flow time step is advanced in steps equal to the average of the desired time steps of all collision cells in the first temporal level. The simulation then proceeds to iterate over the different temporal levels, in descending order of the time step size, where within each loop, all cells sharing the same time step are handled. Fig. 4 presents a flowchart of the implemented temporal adaptation algorithm. The use of different temporal levels allows better handling of the spatial dependence of the time step in the flow domain and decoupling of molecular motion and collision in the variable time step. The temporal adaptivity scheme results in spatial dependence of the time step based on the criteria expressed in Eqs. (7) and (8). Employing a discrete set of temporal levels enables effective handling of the movement of molecules between cells of different time steps. At a certain temporal level l, the molecular and diffusion time steps are equal to Δt_l (Move Molecules (m, Δt_l) and Collide Molecules (l, Δt_l) as shown in Fig. 4). Multiple collisions are carried out at the end of each time step Δt_l; the number of collisions is equal to the nearest integer of the ratio of the assigned temporal level time step of a cell to its desired time step (round($\Delta t_l / \Delta t_{dc}$)). Thus, the temporal adaptivity scheme handle molecular collisions in each collision cell while preserving the number of possible collision pairs. A schematic of the temporal adaptation procedure for a two-temporal levels case is presented in Fig. 5. For the sake of clarity,

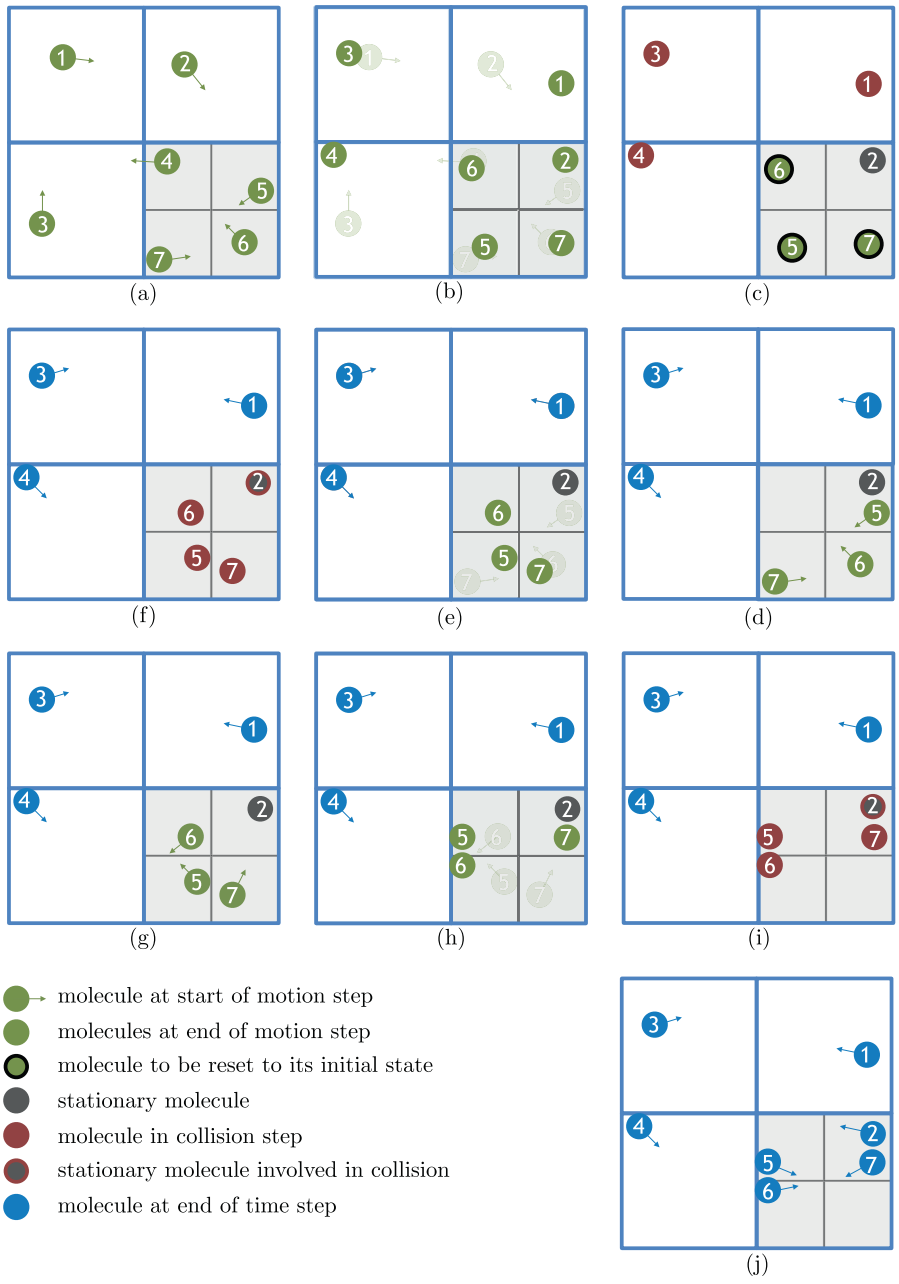

Fig. 5. Schematic describing the temporal adaptation procedure using two temporal levels.

the large cells correspond to the first temporal level with a time step Δt, and the small ones correspond to the second temporal level with a time step $\Delta t/2$. We follow the loop over the temporal levels as depicted in Fig. 4. During the loop over temporal level $l = 0$, simulated molecules within cells of temporal levels $l = 0$ and $l = 1$ are allowed to move (sub-figures (a) and (b)). The molecules are sorted at the end of the move step, and collisions are performed within cells of temporal level $l = 0$ (sub-figure (c)). Before ending the loop over temporal level $l = 0$, simulated molecules that moved to level $l = 1$ (simulated molecule 2) are held stationary during the loop over the temporal level $l = 1$, and simulated molecules that moved from level $l = 1$ to $l = 0$ (simulated molecule 5, 6, and 7) are reset to their initial positions (sub-figures (c) and (d)). The loop over temporal level $l = 1$ considers an inner loop over two time steps of $\Delta t/2$. The same criteria is followed. Sub-figures ((d) and (e)) and sub-figures ((g) and (h)) correspond respectively to molecular movement over the first and second time steps within the inner loop for temporal level $l = 1$. Sub-figure (f) and sub-figure (i) correspond respectively to the sort/collision step over the first and second time steps within same loop for $l = 1$. Sub-figure (j) corresponds to the end of the temporal adaptation procedure.

At the end of the temporal adaptation algorithm, a global sort algorithm re-sorts all molecules into the appropriate collision cells. The simulation is then resumed using the new mesh and the updated temporal levels.

6. Benchmark test cases

Several time-dependent numerical simulations of benchmarked rarefied gas flow in the slip and transition flow regimes are investigated to validate the proposed three-dimensional DSMC code. The simulations were carried out using Fortran in CentOS Linux 7 on 16 core Intel Xeon(R) CPU E5-2650 v2 running at 2.6 GHz.

6.1. Oscillatory shear-driven Couette flow

In the following, we investigate the oscillatory Couette flow. This study showcases using the unsteady DSMC method to simulate time-periodic rarefied gas flows. A flow of argon gas between two infinite parallel plates at a distance H apart is simulated such that the bottom plate is stationary and the top plate oscillates in a simple harmonic motion with a velocity $U = U_0 \sin(\omega t)$ in the lateral direction. The oscillatory Couette flow is characterized by the Knudsen Kn, Mach Ma, and Stokes β dimensionless parameters. The Knudsen number is the ratio of the mean free path λ to the characteristic system length H, $\mathrm{Kn} = \lambda/H$. The Stokes number β represents the ratio of the diffusion to oscillation characteristic time scales,

$$\beta = \sqrt{\frac{\omega H^2}{\nu}} = \left(\frac{H^2/\nu}{1/\omega}\right)^{1/2}, \tag{9}$$

where ν is the kinematic viscosity and ω is the oscillation frequency. The Mach number is the ratio of the flow velocity to the local speed of sound, $Ma = U_0/a$.

Values of the simulation parameters are selected from previous work by Park et al. [31]. The gas medium is initially at rest under standard atmospheric conditions ($P_0 = 101325$Pa and $T_0 = 273$K). The two plates are maintained at the same temperature $T_w = 273$K. The oscillation amplitude of the upper plate is kept constant at $U_0 = 100$ m/s resulting in $Ma = 0.3248$. The characteristic system length and the oscillation frequency are set to $H = 0.625$ μ m, and $\omega = 8.096 \times 10^8\ rad/s$, respectively. This results in $\mathrm{Kn} = 0.1$ and $\beta = 5$. We employ the Hard Sphere (HS) collision model for molecular collisions and the No-Time-Counter (NTC) scheme for collision pair selection. The horizontal plates are assumed to be fully accommodating and periodic boundary conditions are applied on the side walls at the $x-z$ and $y-z$ planes. For this unsteady flow, ensemble averaging over 5000 independent unsteady realizations is performed. Each unsteady realization simulates the flow over a time span long enough for the flow to reach the quasi-stationary behavior. The macroscopic properties are computed every $T/4$, where $T = 2\pi/\omega$ is the period of oscillation.

The analytical solution of this problem is obtained by solving the Navier-Stokes equations subject to the appropriate slip-flow boundary conditions. Indeed, it is that of a one-dimensional boundary-value problem of heat conduction with non-homogeneous boundary conditions of the first kind. This problem is solved with the integral transform (Fourier transform) technique [32] where the velocity is expressed as:

$$u(z,t) = \sum_{n=1}^{\infty} e^{-\alpha\lambda_n^2 t} K(\lambda_n, z)\left[\int_0^t e^{\alpha\lambda_n^2 t'} A(\lambda_n, t')dt'\right] \tag{10}$$

where

$$A(\lambda_n, t') = \frac{K(\lambda_n, 1)}{C_1 \mathrm{Kn}} \tag{11}$$

λ_n are the eigenvalues, C_1 is the modified slip coefficient given by:

$$C_1 = 1.298 + 0.718 \tan^{-1}\left(-1.175\mathrm{Kn}^{0.586}\right) \tag{12}$$

and $K(\lambda_n, z)$ is the Kernel corresponding to non-homogeneous boundary conditions of the first kind.

A semi-analytical/empirical model that is applicable for quasi-steady flows ($\beta \leq 0.25$) in the entire Knudsen regime, and for any Stokes number flow in the slip flow regime ($\mathrm{Kn} \leq 0.1$), is presented in [31]:

$$u(z,t) = \Im\left[\left(U_0 \frac{\sinh(\sqrt{j\beta}Z) + \sqrt{j\beta}C_1\mathrm{Kn}\cosh(\sqrt{j\beta}Z)}{(1+j\beta^2 C_1^2\mathrm{Kn}^2)\sinh(\sqrt{j\beta}) + 2\sqrt{j\beta}C_1\mathrm{Kn}\cosh(\sqrt{j\beta})}\right)e^{j\omega t}\right], \tag{13}$$

where $Z = z/L$ and the symbol $\Im$ denotes the imaginary part of a complex expression. The shear stress at the oscillating plate can be obtained as

$$\tau_{xz}|_{z=L} = \mu \left.\frac{du(z,t)}{dz}\right|_{z=L}. \tag{14}$$

DSMC results of the velocity distribution at different times of an oscillation period within the time-periodic state are shown in Fig. 6. The results are in good agreement with the published data in [31], and with the analytical solutions (Eqs. (13) and (10)). Departure of the analytical solution from that predicted by DSMC is expected due to the fact that it employs a first order slip boundary condition at solid walls. Comparison of the the shear stress obtained from DSMC simulations and the analytical solution is presented in Fig. 7 at different times.

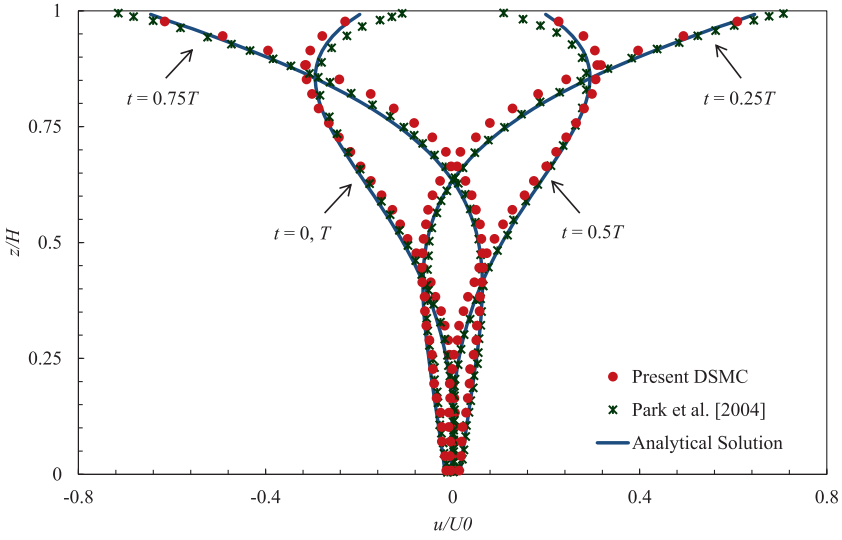

Fig. 6. Normalized velocity profile for the shear-driven oscillatory Couette flow at $\mathrm{Kn} = 0.1$, $\mathrm{Ma} = 0.3248$, and $\beta = 5.0$.

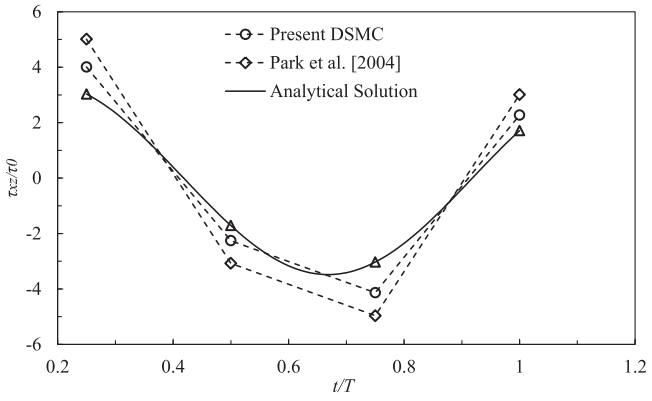

Fig. 7. Normalized wall shear stress for the shear-driven oscillatory Couette flow at different times. $\mathrm{Kn} = 0.1$, $\beta = 5.0$, and $\tau_0 = U_0/H$.

6.2. Impulsively started Couette flow

Here, we investigate the transient behavior of argon gas between two parallel, diffusely reflecting plates, each at temperature $T = 273$K, in response to impulsively started motion of the plates at a speed of $U_0 = 100$ m/s. The separation between the two plates is $H = 0.2976$ μm so that $\mathrm{Kn} = 0.21$. Unsteady sampling of macroscopic properties and the viscous shear stress is performed at time $t = 16.2\epsilon^{-1}$, where ϵ denotes the molecular collision frequency. The viscous shear stress, τ, derived using molecular gas dynamics [2], is defined as the negative of the stress tensor with the static pressure subtracted from the normal components. It is written in tensor notation as:

$$\boldsymbol{\tau} \equiv \tau_{ij} = -\left(\rho\overline{v_i' v_j'} - \delta_{ij} p\right), \tag{15}$$

where ρ is the mass density, δ_{ij} is the Kronecker tensor, and v_i' and v_j' are the components of the molecular velocity relative to the stream velocity. The hydrostatic pressure, defined as the average of the three normal components of the pressure tensor, is

$$p = \frac{1}{3}\rho\left(\overline{v_x'^2} + \overline{v_y'^2} + \overline{v_z'^2}\right) = \frac{1}{3}\rho\overline{v'^2} \tag{16}$$

To examine the accuracy of our DSMC methodology, comparison of the velocity and stress fields, with previous results by Hadjiconstantinou [33], and with the exact solution of the Navier-Stokes equation for second slip flow is shown in Fig. 8. The comparison shows a fairly good agreement.

6.3. Thermal Couette flow

In the following Couette-flow test, Argon gas, initially at rest and at temperature $T = 273$K, is contained in the region between two stationary plates, which are held at temperatures 173K and 373K, respectively. The system is then allowed to reach a steady state. Fig. 9 represents steady-state temperature profiles across the channel at different Knudsen numbers ($\mathrm{Kn} = 0.1, 1.0$). A very good agreement is obtained with previous results by Olson et al. [34].

6.4. Hypersonic flow past a flat-nosed cylinder

Hypersonic flow of argon gas at a temperature of 100K, a number density of $1 \times 10^{21} m^{-3}$, and a velocity of 1000 m/s over a flat-nosed cylinder with a radius of 0.01 m is considered. The corresponding stream Mach number and Knudsen number based on the diameter of the cylinder, are respectively 5.37 and 0.0474. The simulated flow domain is 0.04 m in the axial $x-$direction, and 0.03 m in the $y-$ and $z-$ directions. The total length of the cylinder is 0.02 m and the centre of its flat face is located at $x = 0.02$ m, $y = 0$, and $z = 0$. The free stream flow boundary conditions are set at left ($yz-$plane), backward ($y = 0.03$ m), and top boundary ($z = 0.03$ m) of the computational domain using the standard method and by providing the free stream velocity and free stream temperature. Vacuum condition is imposed at the outlet boundary ($x = 0.04$ m). Periodic boundary condition is imposed at the bottom ($xy-$plane), and the in front boundary ($xz-$plane) of the computational domain. Diffuse wall boundary condition is applied at the cylinder

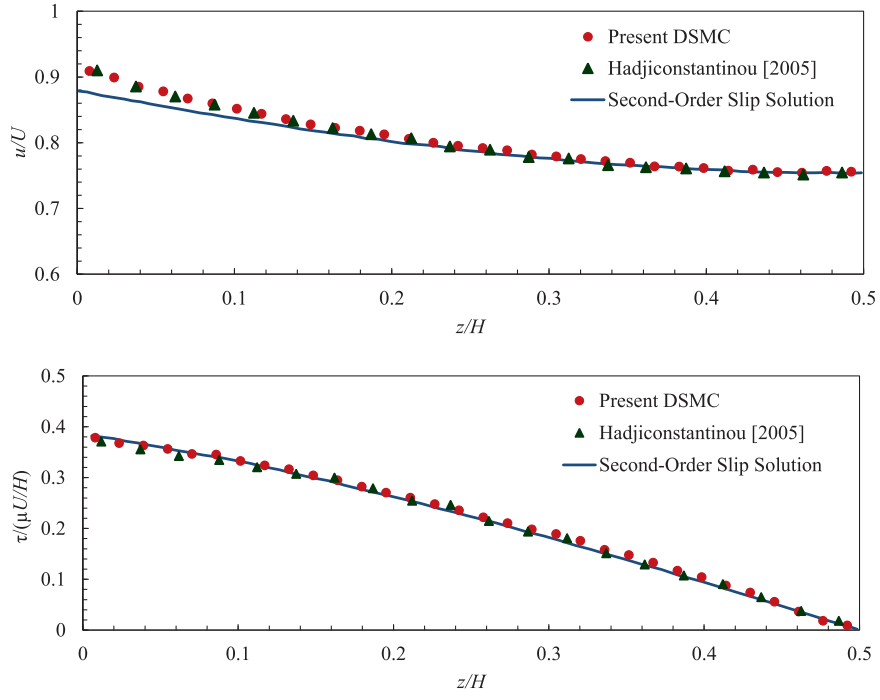

Fig. 8. Velocity (top) and Stress (bottom) fields for the impulsive start Couette flow for $\mathrm{Kn} = 0.21$ at times $16.2\epsilon^{-1}$.

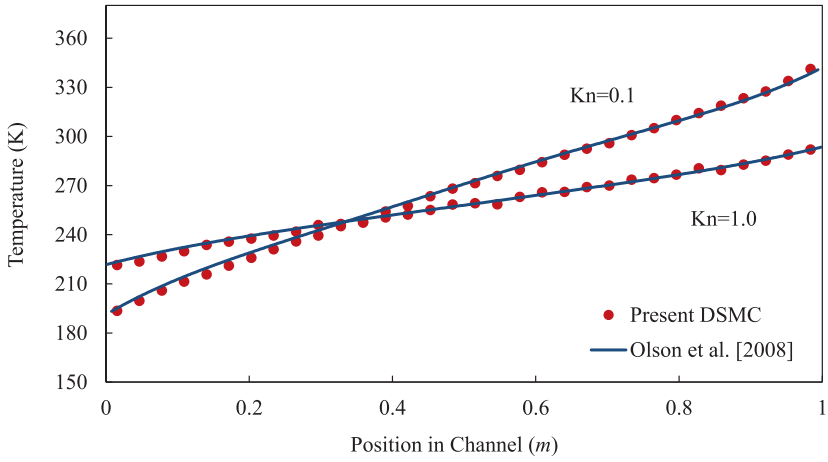

Fig. 9. Temperature profile in a thermal Couette flow problem.

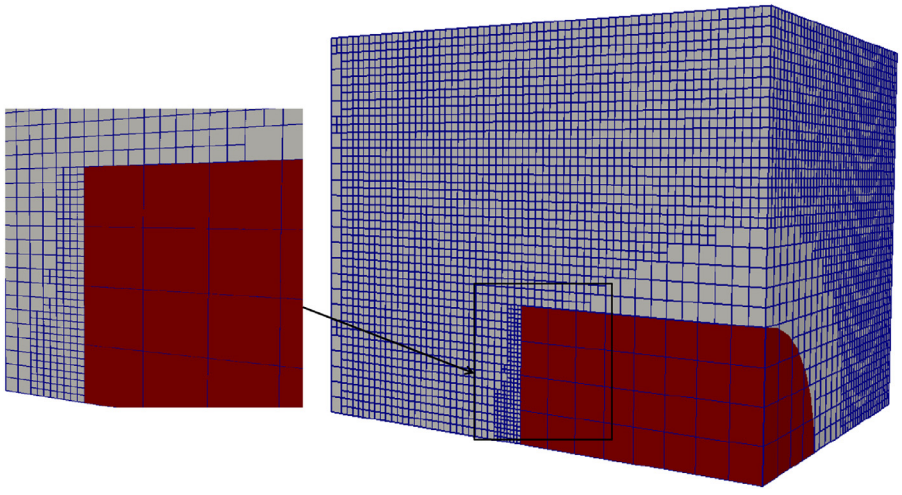

Fig. 10. Flow field refined mesh for a flat-nosed cylinder.

wall by incorporating a half-range Maxwellian distribution determined by the wall temperature and velocity. The temperature at the cylinder wall is set to 300K.

This problem has been studied by Bird [2] as a 2D axisymmetric problem and as a 3D quarter-section model with symmetry boundary conditions by Scanlon et al. [9]. To assess the spatial adaptivity scheme in our DSMC algorithm, this flow is simulated with and without dynamic grid adaptation. For the non-adaptive simulation, the grid spacing was chosen to be $\Delta x \sim \lambda_0/2$, where λ_0 is the mean free path at the initial conditions. For the adaptive simulation, the initial grid size is $\Delta x \sim \lambda_0$ and the grid spacing is computed as $\Delta x \sim \lambda/2$, where at each adaptation time interval, λ for each cell is computed using Eq. (2), where T and n for the cell are averaged over all the CPU cores. Fig. 10 shows the mesh upon spatial adaptation where the size of the grid cells near the flat face of the cylinder is adapted to 1/4 the size of the initial grid. This illustrates that the high-density region near the flat face of the cylinder is resolved with a finer mesh distribution. The transient solution and spatial adaptivity sampling intervals are listed in Table 2, in addition to other parameters. The table also shows that at the end of the simulation, the number of cells in the adaptive simulation increased from 18,432 to 136,893 which is comparable to that used in the non-adaptive case. The computational cost of the adaptive simulation is, however, less by a factor of 4.3. Besides, the percentage of time spent by the sniffer algorithm, when parallelizing over 15 CPU cores, is less than 2% of the total time per realization per core. This reflects the high performance and efficiency of the implemented parallelization method. The temperature and density contours are in good agreement with the results from Bird's DSMC2A code [2], as seen in Fig. 11 and with results using dsmcFoam by Scanlon et al. [9], as seen in Fig. 12.

Table 2
Comparison of elapsed time per DSMC realization for non-adaptive and adaptive hypersonic flow past a flat-nosed cylinder.

	Non-adaptive	*Adaptive*
Time step	1.55 μs	1.55 μs
Sampling time	0.98 ms	0.98 ms
Transient solution sampling interval	0.196 ms	0.196 ms
Spatial adaptivity sampling interval		0.244 ms
Total no. of time steps	633	633
Total Time per Realization per core	102440.0 s	23719.0 s
Percentage of Time Spent by Sniffer	0.18%	1.1%
Initial no. of Cells	147,456	18,432
Final no. of Cells	147,456	136,893

6.5. Hypersonic flow over a cylinder

Mach-10 hypersonic cylinder flow is a well-known 2D-benchmark problem [24,35,36]. Fig. 13 shows the schematic of a Mach-10 (2634.1m/s) flow of argon gas at a temperature of 200K and a number density of 4.274×10^{20} m^{-3} past a circular cylinder

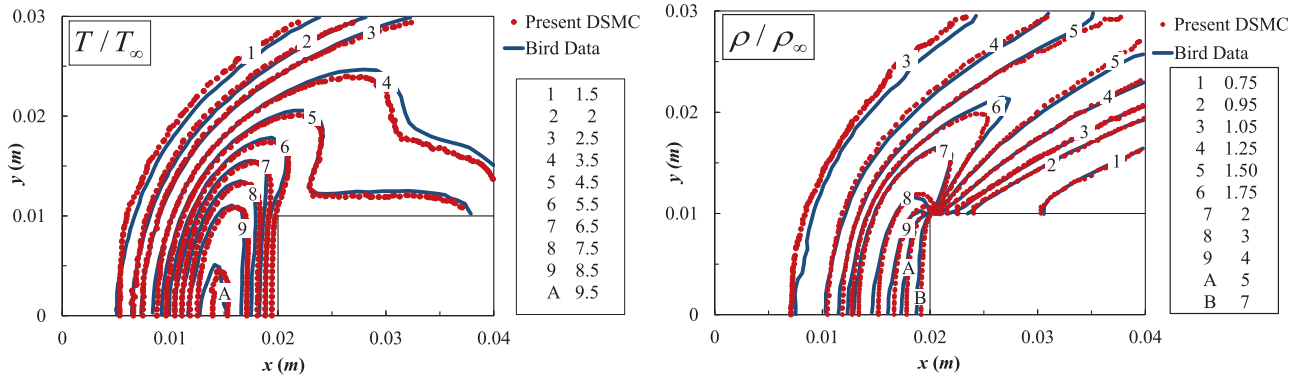

Fig. 11. Temperature and density contours for hypersonic flow past a flat-nosed cylinder. A comparison of the results in this work with those computed in previous work done by Bird [2].

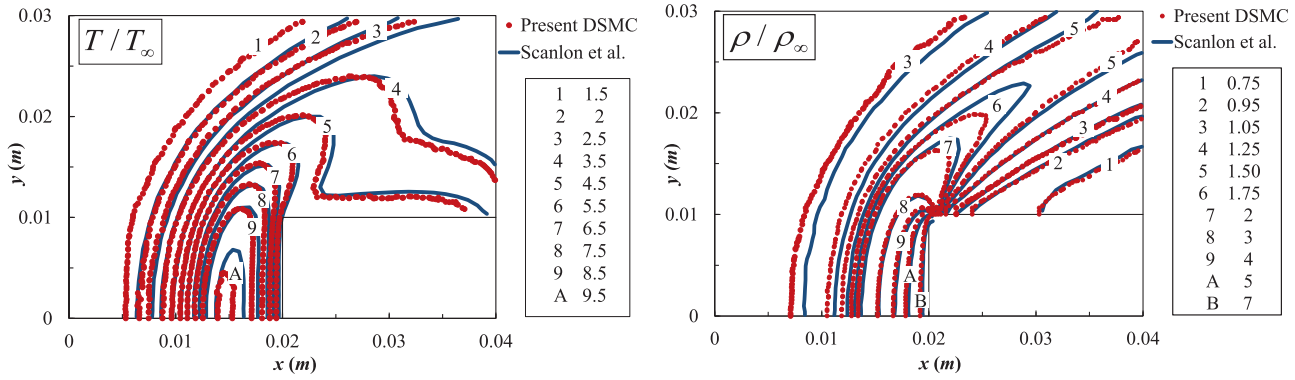

Fig. 12. Temperature and density contours for hypersonic flow past a flat-nosed cylinder. A comparison of the results in this work with those computed in previous work done by Scanlon et al. [9].

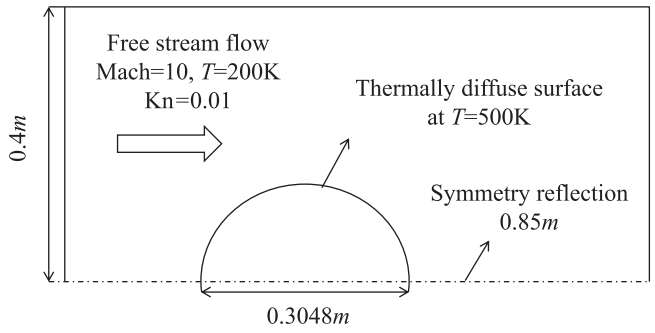

Fig. 13. Sketch of the computational domain of argon hypersonic flow over a cylinder at $\mathrm{Kn}_\infty = 0.01$, $\mathrm{Ma}_\infty = 10$, $T_\infty = 200$ K, $n_\infty = 4.274 \times 10^{20}\, m^{-3}$.

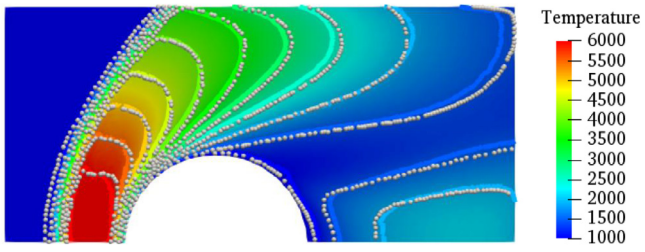

Fig. 14. Contours of temperature of Mach-10 hypersonic flow past a circular cylinder; colored lines are DSMC data; the grey spheres are VMR data [35].

with a fully diffusive wall at a temperature of 500K. The corresponding free-stream Knudsen number is 0.01, based on the free-stream mean free path ($\lambda_\infty = 0.003$ m) and the diameter of the cylinder ($D = 0.3048$ m).

As part of the validation process, the spatio-temporal adaptivity scheme is applied to this problem. This is motivated by the rapid variations in the local molecular mean free path and the mean collision time. The computational grid is generated initially with a cell size of approximately $(1-2)\ \lambda$, based on free-stream conditions, and using 100 particles per cell. As the flow evolves, the parameters λ and τ_c are periodically computed for each cell. All DSMC requirements are checked, i.e., the cell size is smaller than the local mean free path, the time step is smaller than the local mean collision time, and the number of simulated molecules is around $20-30$ molecules. If these conditions are not satisfied in a given cell, the spatio-temporal adaptation algorithm is called, and the parameters Δx_c and Δt are modified. In doing so, the cell size is adapted to 0.25λ and the desired time step is adapted to $0.333\tau_c$. The total number of cells in the simulation domain changed from 7,680 to 10,480, the time step Δt changed from 3.176 μs to 1.42 μs and 0.71 μs in grid cells belonging to the first and second temporal levels, respectively. Fig. 14 compares contours of temperatures obtained using our DSMC algorithm with those reported previously using the virtual mesh refinement (VMR) module [35]. Figs. 15 and 16 show the density and temperature distribution along a vertical line just before the cylinder ($x = 0.205$ m) and in the wake region ($x = 0.6$ m), respectively. Our results are in good agreement with the benchmarked results.

6.6. Hypersonic flow over a flat plate

In this section, we benchmark our DSMC solver against the DAC and MONACO solvers for the hypersonic flow over a flat plate investigated by Padilla et al. [37]. The flow domain, shown in Fig. 17, is a rectangular region with dimensions 180, 1, and 205 mm in the

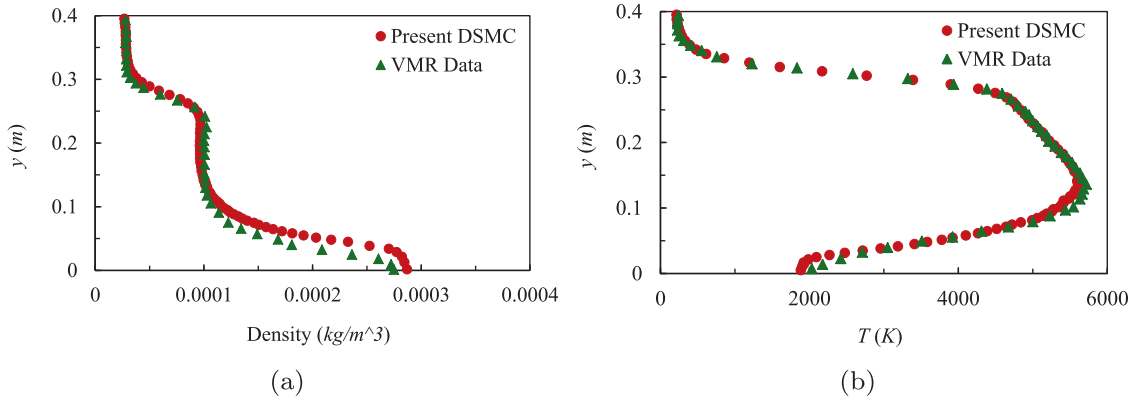

Fig. 15. Density and temperature distribution along a vertical line before the cylinder (x=0.205 m). (a) Density (b) Temperature.

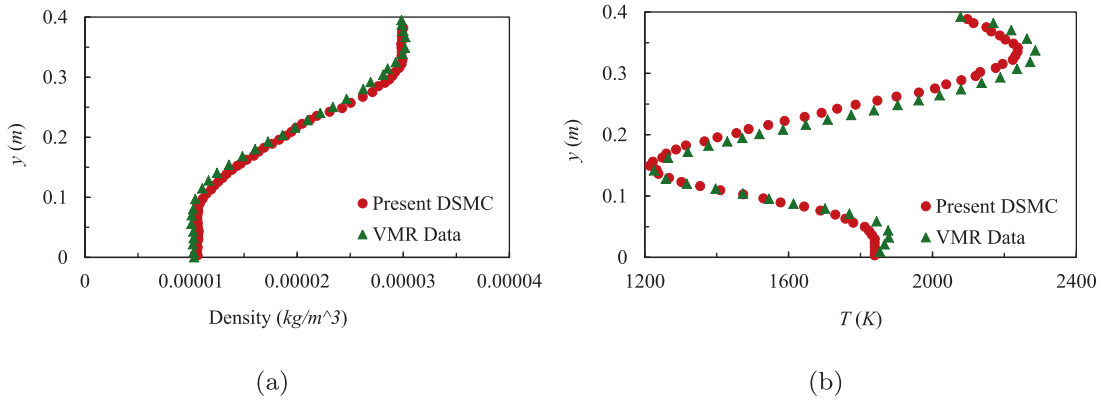

Fig. 16. Density and temperature distribution along a vertical line in the wake region (x=0.6 m). (a) Density (b) Temperature.

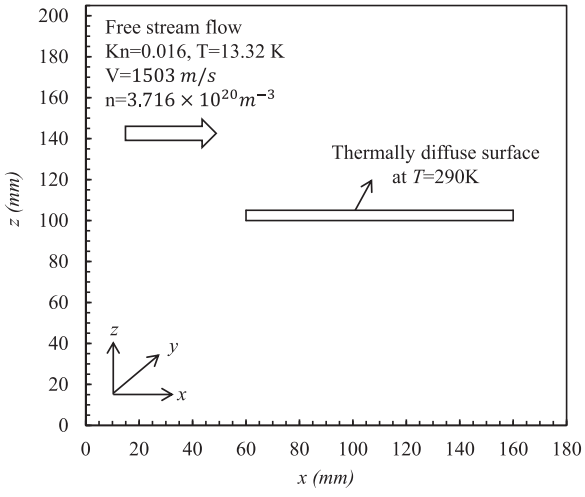

Fig. 17. Schematic of the flow field domain for hypersonic flow over a flat plate.

x, y, and z directions, respectively. The flat plate length and thickness are 100 mm and 5 mm respectively. Nitrogen gas (VHS gas) enters a the flow domain with a velocity of 1503 m/s and a temperature of 13.32K. Inflow boundary conditions are applied to the left, bottom, and top faces of the flow domain. Outflow boundary condition is applied at the right face. Periodic boundary conditions are enforced at the $x-z$ planes on either sides of the domain. The boundary condition at the flat plate, of fixed temperature (290K), is that of diffuse reflection and full thermal accommodation. Other parameters can be found in [37].

We compare results of our DSMC code with those reported in [37] for the case when the molecular rotational energy exchange is not considered. The pressure and shear stress distributions along the upper surface of the flat plate are compared in Fig. 18, and contours of bulk flow velocity magnitude are compared in Fig. 19. Reasonable agreement in the results between the two codes is observed. Comparison of our DSMC algorithm with DAC and MONACO in terms of the memory spread and computational cost of the molecular motion, collisions, and other procedures are presented in Table 3. Our algorithm consumes ~2 μs per realization per simulated molecule, whereas DAC and MONACO consume ~1.4 μs. The relative time cost of the main procedures show that our collision procedure is more expensive than that of DAC and MONACO. This may be attributed to the fact that we employ more simulators per cell resulting in a larger number of collision pairs. In terms of memory, our code consumes 360 bytes of memory per simulator, which is more than twice that of MONACO and five times that of DAC. The higher memory consumption of our algorithm is because it is designed for transient flows with complex geometries. For these problems, employing an oct-tree and the associated data structures and the toolset that comes along are expected to consume more CPU time and memory. For example, checking whether (and where) a ray intersects a rectangle (with sides aligned with the coordinates) is much cheaper and faster than checking whether (and where) a ray intersects a complex surface represented by a triangular mesh. From the memory perspective, the amount of memory it takes to store the geometry of a rectangle is much less than that of a triangulated surface mesh where the length scale of the triangles is of the order of the collision cell size. In addition, the spatio-temporal adaptivity also requires additional memory cost (such as re-sorting the molecules in the new cells), but

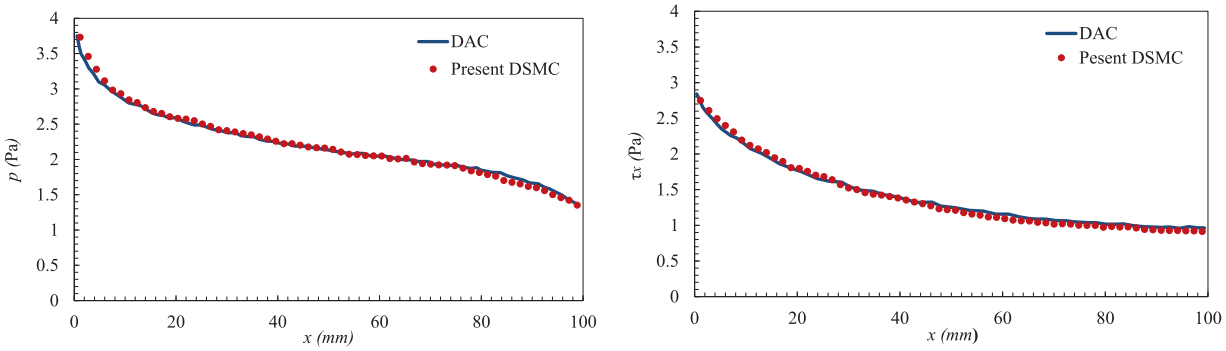

Fig. 18. Comparison of the surface pressure and the shear stress in the x-direction along the upper surface of the flat plate.

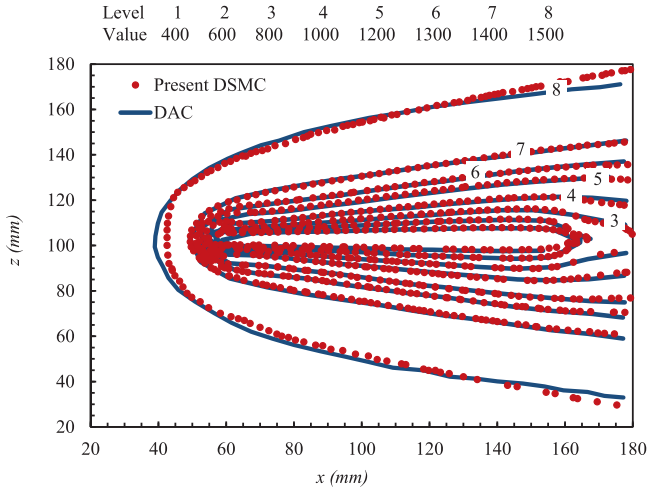

Fig. 19. Schematic of the flow field domain for hypersonic flow over a flat plate.

Table 3
Comparison of DSMC metrics for hypersonic flow over a flat plate.

	DAC	*MONACO*	*Present DSMC*
Sampling technique	Steady sampling	Steady sampling	Unsteady sampling
Number of simulators	3,169,267	3,173,070	6,342,950
Simulation time per realization per simulator	1.378μs	1.376μs	1.96μs
Memory per realization per simulator	70 bytes	163 bytes	370 bytes per core
Relative cost	Move: 48%, Collide: 6%, other: 46%	Move: 36%, Collide: 12%, other: 52%	Move: 43%, Collide: 15%, other: 42%

with the potential to yield significant savings, as demonstrated in previous examples. Whether the spatio-temporal adaptivity algorithm yields significant savings or not depends on the problem. For some problems, such as steady flows with moderate spatial gradients, the spatio-temporal adaptivity is not expected to offer any savings.

7. Conclusion and future work

In this paper, a new parallel spatio-temporally adaptive multiscale DSMC algorithm is proposed and implemented to simulate unsteady rarefied flows in complex geometries. A hierarchical octree-based Cartesian grid is generated and a cut-cell method, where the triangulated solid boundary surface intersects the grid cells, is adopted. The selected geometry model is characterized by its low memory storage requirements compared to the use of non-Cartesian meshes, its ability to implement an efficient ray-tracing particle movement scheme, and its flexibility to incroporate a fully dynamic three-dimensional spatial and temporal adaptive scheme that maintains DSMC constraints consistent with the local variations of flow field properties. The proposed algorithm employs a novel spatio-temporal adaptive scheme based on the macroscopic averages of local flow properties. It also incorporates a new parallelization method based on running parallel unsteady DSMC simulations simultaneously and independently over multiple CPU cores. Results for two- and three-dimensional benchmark test cases demonstrate the accuracy and robustness of the proposed DSMC algorithm. In future work, we expect to extend the algorithm to simulate moving boundary flows and then showcase the efficiency and accuracy of the algorithm in predicting unsteady flows encountered in transient three-dimensional fluid-structure interaction problems and several micro- and nano-electromechanical systems (MEMS/NEMS) applications.

Acknowledgments

This work is supported by the University Research Board of the American University of Beirut (grant # 103371).

References

[1] Gad-el Hak M. The fluid mechanics of microdevices–the freeman scholar lecture. J Fluids Eng 1999;121(1):5–33.
[2] Bird GA. Molecular gas dynamics and the direct simulation of gas flows. no. v. 1. Clarendon Press; 1994. ISBN 9780198561958. 94003873
[3] Oran E, Oh C, Cybyk B. Direct simulation Monte Carlo: recent advances and applications. Annu Rev Fluid Mech 1998;30(1):403–41.
[4] Diab NA, Lakkis IA. Investigation of the squeeze film dynamics underneath a microstructure with large oscillation amplitudes and inertia effects. J Tribol 2016;138(3):031704.
[5] Gerdroodbary MB, Ganji D, Taeibi-Rahni M, Vakilipour S. Effect of Knudsen thermal force on the performance of low-pressure micro gas sensor. Eur Phys J Plus 2017;132(7):315.
[6] Frangi A, et al. Advances in multiphysics simulation and experimental testing of MEMS, 2. Imperial College Press; 2008.
[7] Roohi E. DSMC simulations of nanoscale and microscale gas flow. In: Encyclopedia of microfluidics and nanofluidics. Springer; 2015. p. 681–93.
[8] Dietrich S, Boyd ID. Scalar and parallel optimized implementation of the direct simulation Monte Carlo method. J Comput Phys 1996;126(2):328–42.
[9] Scanlon T, Roohi E, White C, Darbandi M, Reese J. An open source, parallel DSMC code for rarefied gas flows in arbitrary geometries. Comput Fluids 2010;39(10):2078–89.
[10] LeBeau G. A parallel implementation of the direct simulation Monte Carlo method. Comput Methods Appl Mech Eng 1999;174(3–4):319–37.
[11] Ivanov M, Kashkovsky A, Gimelshein S, Markelov G, Alexeenko A, Bondar YA, et al. SMILE system for 2D/3D DSMC computations. In: Proceedings of 25th international symposium on rarefied gas dynamics, St. Petersburg, Russia; 2006. p. 21–8.
[12] Bird G, Capitelli M. The DS2V/3V program suite for DSMC calculations. In: AIP conference proceedings, 762. AIP; 2005. p. 541–6.
[13] Gao D, Zhang C, Schwartzentruber T. A three-level Cartesian geometry-based implementation of the DSMC method. In: 48th AIAA aerospace sciences meeting including the new horizons forum and aerospace exposition; 2010. p. 450.

[14] Klothakis A, Nikolos I. Modeling of rarefied hypersonic flows using the massively parallel DSMC kernel SPARTA. 8th int. congress computational mechanics; 2015.
[15] Kannenberg KC, Boyd ID. Strategies for efficient particle resolution in the direct simulation Monte Carlo method. J Comput Phys 2000;157(2):727–745.
[16] Gallis MA, Torczynski J, Rader D, Bird GA. Convergence behavior of a new DSMC algorithm. J Comput Phys 2009;228(12):4532–48.
[17] Wade ACJ, Baillie D, Blakie P. Direct simulation monte carlo method for cold-atom dynamics: classical boltzmann equation in the quantum collision regime. Phys Rev A 2011;84(2):023612.
[18] Samet H. The design and analysis of spatial data structures. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.; 1990. ISBN 0-201-50255-0.
[19] Arslanbekov R, Kolobov V, Burt J, Josyula E. Direct simulation Monte Carlo with octree Cartesian mesh. In: 43rd AIAA thermophysics conference; 2012. p. 2990.
[20] Zhang C, Schwartzentruber TE. Robust cut-cell algorithms for DSMC implementations employing multi-level Cartesian grids. Comput Fluids 2012;69:122–135.
[21] Tysanner MW, Garcia AL. Non-equilibrium behaviour of equilibrium reservoirs in molecular simulations. Int J Numer Methods Fluids. 2005;48(12):1337–1349.
[22] Roohi E, Stefanov S. Collision partner selection schemes in DSMC: from micro/nano flows to hypersonic flows. Phys Rep 2016;656:1–38.
[23] Bird GA. Direct simulation of gas flows at the molecular level. Commun Appl Numer Methods 1988;4(2):165–72 pmid, doi:10.1002/cnm.1630040205.
[24] Bird GA. Sophisticated DSMC, Notes Prep. a Short Course DSMC07 Meet. St. Fe, USA; 2007.
[25] Shen C. Rarefied gas dynamics: fundamentals, simulations and micro flows. Springer Science & Business Media; 2006.
[26] Cave HM, Tseng KC, Wu JS, Jermy MC, Huang JC, Krumdieck SP. Implementation of unsteady sampling procedures for the parallel direct simulation Monte Carlo method. J Comput. Phys. 2008;227(12):6249–71. doi:10.1016/j.jcp.2008.03.015.
[27] CEA/DEN, EDF R&D and OPEN CASCADE, SALOME: the open source integration platform for numerical simulation. 2015. URL: https://www.salome-platform.org/ [accessed July 2018].
[28] Coutinho MG. Guide to dynamic simulations of rigid bodies and particle systems. Springer Science & Business Media; 2012.
[29] Akenine-Möller T. Fast 3D triangle-box overlap testing. In: ACM siggraph 2005 courses. ACM; 2005. p. 8.
[30] Sun Z-X, Tang Z, He Y-L, Tao W-Q. Proper cell dimension and number of particles per cell for DSMC. Comput Fluids 2011;50(1):1–9.
[31] Park JH, Bahukudumbi P, Beskok A. Rarefaction effects on shear driven oscillatory gas flows: a direct simulation Monte Carlo study in the entire Knudsen regime. Phys Fluids 2004;16(2):317–30.
[32] Issa L, Lakkis I. Reduced-order modeling of low mach number unsteady microchannel flows. J Fluids Eng 2014;136(5):051201.
[33] Hadjiconstantinou NG. Validation of a second-order slip model for dilute gas flows. Microscale Thermophys Eng. 2005;9(2):137–53.
[34] Olson SE, Christlieb AJ. Gridless DSMC. J. Comput. Phys. 2008;227(17):8035–64. doi:10.1016/j.jcp.2008.04.038.
[35] Su C, Tseng K, Wu J, Cave H, Jermy M, Lian Y. Two-level virtual mesh refinement algorithm in a parallelized DSMC code using unstructured grids. Comput Fluids 2011;48(1):113–24.
[36] Goshayeshi B, Roohi E, Stefanov S. DSMC simulation of hypersonic flows using an improved SBT-TAS technique. J. Comput. Phys. 2015;303:28–44.
[37] Padilla JF. Comparison of DAC and MONACO DSMC codes with flat plate simulation; 2010.