SPARTA
Stochastic Particle Real Time Analyzer Validation and Verification Test Suite

Michael A. Gallis, Timothy P. Koehler, Steven J. Plimpton

Sandia National Laboratories

Stochastic Particle Real Time Analyzer (SPARTA) Validation and Verification Suite

Michael A. Gallis, Timothy P. Koehler
Fluid Science and Engineering Department
P.O. Box 5800-0840
Albuquerque, New Mexico 87185-MS0840

Steven J. Plimpton
Multi Scale Science Department
Sandia National Laboratories
P.O. Box 5800-1316
Albuquerque, New Mexico 87185-MS1316

ABSTRACT

This report presents the test cases used to verify, validate and demonstrate the features and capabilities of the first release of the 3D Direct Simulation Monte Carlo (DSMC) code SPARTA (Stochastic Particle Real Time Analyzer). The test cases included in this report exercise the most critical capabilities of the code like the accurate representation of physical phenomena (molecular advection and collisions, energy conservation, etc.) and implementation of numerical methods (grid adaptation, load balancing, etc.). Several test cases of simple flow examples are shown to demonstrate that the code can reproduce phenomena predicted by analytical solutions and theory. A number of additional test cases are presented to illustrate the ability of SPARTA to model flow around complicated shapes. In these cases, the results are compared to other well-established codes or theoretical predictions. This compilation of test cases is not exhaustive, and it is anticipated that more cases will be added in the future.

ACKNOWLEDGMENTS

CONTENTS

FIGURES

TABLES

1. INTRODUCTION

The Direct Simulation Monte Carlo (DSMC) method was introduced in the early 1960s [1] to compute re-entry flow fields between the free-molecular and continuum regimes, for which results could not be obtained from more traditional approaches based on solving partial differential equations (PDEs). Since then, the fundamental physics-based nature of the DSMC algorithm in conjunction with ever-increasing computational power has allowed DSMC to attack complicated non-equilibrium problems outside its original regime of applicability, even in the near-continuum regime.

Over nearly a half-century of development, the capabilities of the DSMC method have been dramatically enhanced with the addition of improved physical and chemical models. It is a testament to the fundamental stochastic nature of the method that all these enhancements provided continual improvement to the method without any major change to the basic algorithm or significantly impacting the computational efficiency of the method. It is worth noting that DSMC can provide simulations of flow fields even in cases where the Boltzmann equation cannot be rigorously formulated, such as for chemically reacting polyatomic species.

Wagner [2] provides a rigorous proof that DSMC simulations actually provide solutions to the Boltzmann equation for monatomic molecules in the limit of vanishing discretization and statistical error. Further evidence indicates that highly refined DSMC simulations provide results that agree with exact solutions to the Boltzmann equation, such as the near-equilibrium infinite-approximation Chapman-Enskog and the non-equilibrium Moment Hierarchy methods [3,4] The validity of the DSMC method for more complicated flow fields for which analytic solutions cannot be obtained has been firmly established through comparisons with experimental data and molecular-dynamics simulations.

Even today, more than 50 years since originally introduced, DSMC remains unchallenged for non-continuum rarefied-gas-dynamics calculations. The advantages of DSMC compared to other methods that predict flows in the non-equilibrium regime come at a cost: DSMC is computationally intense, like most Monte Carlo methods. Therefore, its successful application to real problems may depend heavily on the parallel performance of the particular implementation. Monte Carlo methods usually have good parallel performance. This is because the workload depends mainly on the simulators within a cell: there is relatively less need to communicate information between cells. DSMC codes have demonstrated near-linear scaling up to several thousand processors. With the advent of massively parallel computers, problems considered impossible only a few years ago, such as transient three-dimensional simulations, are currently within reach.

Since next-generation machines will require hybrid programming approaches, MPI in addition to some kind of node-based parallelism, the DSMC kernels (particle moves, collisions, and chemistry) need to be reworked in a manner that will allow for fine-grain parallelism within a node. Since the computational architecture is subject to frequent and dramatic changes, it would be ideal to decouple the physical aspects of the simulation from the computational ones. One possible avenue would be to develop architecture-agnostic programming models for in-node parallelism, such as the Sandia National Laboratories Kokkos project [5]. Its goal is to provide

data abstractions and parallel looping constructs that enable a single set of source code kernels in an application code to run reasonably efficiently across a diverse spectrum of node hardware (multi-core, many-core, GPU, etc.).

SPARTA is a new open source 3D DSMC code [6] developed at Sandia National Laboratories with these ideas in mind. SPARTA employs a multi-level (practically up to 16 levels on 64 bit architectures) Cartesian-grid with embedded geometries for the description of physical space. Load balancing is based on the recursive coordinate bisection method. To enable efficient communications, a buffer zone is assigned to each processor that includedes near-by cells. The goal of this is to allow, in most cases, completion of molecular moves using a maximum of one communication with a modest number of processors. However, the motion of a molecule is never truncated, and multiple passes are possible. Currently, the DSMC algorithm as outlined in Bird's 1994 monograph is implemented. Later versions will incorporate more advanced physical models.

In this report, the ability of SPARTA to deliver improved computational efficiency while maintaining the unprecedented accuracy of the DSMC method, in dealing with gas flows is examined. The benchmark cases used for this purpose varying from ostensibly simple steady flow in a closed box, to the continuum, transient Richtmyer-Meshkov instability [7,8]. These test cases have been selected because they are amenable to theoretical as well as numerical investigation.

2. TEST CASES

2.1. Flow in a closed box

2.1.1. Motivation

Despite its simplicity, this class of test cases is one of the most useful test cases for DSMC codes since it allows the development of an equilibrium steady state flow, where one can have complete confidence in having achieved correct results.
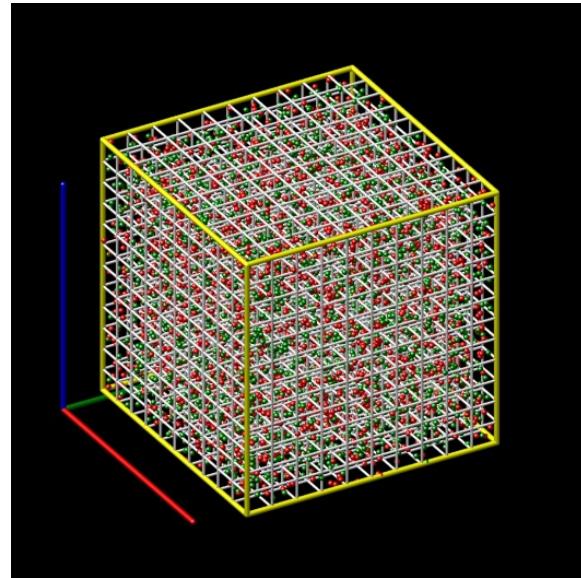
Figure 1. Flow inside a closed box

Molecular collisions in a gas inside a closed box (1, 2 or 3-D) with diffuse or specular walls, will lead the gas, regardless of its initial state, to complete thermodynamic equilibrium. Conservation of mass, momentum and energy, exactly conserved in every DSMC collision, allow for a number of code functions to be verified.

2.1.2. Description

The main test case is a uniform hard sphere gas in a 3-D box. The collision frequency of this gas is exactly known (Eq. 4.63 of Reference 1). In a fully diffuse isothermal box the temperature of the gas should converge to that of the walls. In a specular box, exact conservation of energy should maintain the temperature at exactly the level to which it was set by the initial condition. Variations of this test case include: Maxwell-molecule gas (all collision attempts are accepted for collision), and VSS and VHS gases foe which the relevant functions of the code can be tested.

2.1.3. Parts of the code being checked

Molecular advection, Molecular tracking in a 3-D domain, Surface reflections, Particle collision frequency, Conservation of mass, momentum and energy, Polyatomic gas collisions, Initial conditions, diffuse, specular wall boundary condition, load balancing.

2.1.3 Results

The following is a printout of the log file.

```
SPARTA (19 Aug 2014)
Created orthogonal box = (0 0 0) to (0.0001 0.0001 0.0001)
Created 1000 child grid cells
  parent cells = 1
  CPU time = 0.000895977 secs
  create/ghost percent = 98.3502 1.64981
Balance grid migrated 0 cells
  CPU time = 0.00139093 secs
  reassign/sort/migrate/ghost percent = 5.31368 0.154268 1.14844 93.3836
Created 10000 particles
  CPU time = 0.00199509 secs
Memory usage per proc in Mbytes:
  particles (ave,min,max) = 1.02997 1.02997 1.02997
  grid      (ave,min,max) = 1.83113 1.83113 1.83113
  surf      (ave,min,max) = 0 0 0
  total     (ave,min,max) = 2.8611 2.8611 2.8611
Step CPU Np Natt Ncollave temp
       0           0   10000        0              0   274.40561
     100  0.048908949   10000      968        700.8   274.40561
     200  0.097182035   10000      985       702.64   274.40561
     300   0.14856601   10000     1005    704.95333   274.40561
     400   0.19848394   10000      998     705.7675   274.40561
     500   0.24520898   10000     1022       706.47   274.40561
     600   0.29146504   10000     1025    706.61833   274.40561
     700   0.33719492   10000     1042    706.68571   274.40561
     800   0.38323998   10000     1029    706.80125   274.40561
     900     0.430583   10000     1052    706.89778   274.40561
    1000   0.47937894   10000     1022      706.741   274.40561
Loop time of 0.479393 on 1 procs for 1000 steps with 10000 particles

Particle moves    = 10000000 (10M)
Cells touched     = 13606155 (13.6M)
Particle comms    = 0 (0K)
Boundary collides = 401016 (0.401M)
Boundary exits    = 0 (0K)
SurfColl checks   = 0 (0K)
SurfColl occurs   = 0 (0K)
Collide attempts  = 1010885 (1.01M)
Collide occurs    = 706741 (0.707M)
Reactions         = 0 (0K)
Particles stuck   = 0

Particle-moves/CPUsec/proc: 2.08597e+07
Particle-moves/step: 10000
Cell-touches/particle/step: 1.36062
Particle comm iterations/step: 1
Particle fraction communicated: 0
Particle fraction colliding with boundary: 0.0401016
Particle fraction exiting boundary: 0
Surface-checks/particle/step: 0
Surface-collisions/particle/step: 0
Collision-attempts/particle/step: 0.101088
Collisions/particle/step: 0.0706741
Reactions/particle/step: 0

Move  time (%) = 0.240076 (50.0791)
Coll  time (%) = 0.201777 (42.09)
Sort  time (%) = 0.0367324 (7.66228)
```

```
Comm  time (%) = 9.87053e-05 (0.0205896)
Outpt time (%) = 0.000492334 (0.1027)
Other time (%) = 0.000216961 (0.0452574)

Particles: 10000 ave 10000 max 10000 min
Histogram: 1 0 0 0 0 0 0 0 0 0
Cells:     1000 ave 1000 max 1000 min
Histogram: 1 0 0 0 0 0 0 0 0 0
GhostCell: 0 ave 0 max 0 min
Histogram: 1 0 0 0 0 0 0 0 0 0
EmptyCell: 0 ave 0 max 0 min
Histogram: 1 0 0 0 0 0 0 0 0 0
```

2.1.4. Comments

There are many objectives of this simulation. Observing the number of simulators and the temperature to be exactly constant is a clear indication that the three conservation principles are satisfied during the simulation. Also, the collision frequency can be calculated and compared to that given by equilibrium kinetic theory. For the particular problem simulated herein (argon at 2 torr pressure at 273.15 K) the theoretical and simulated collision frequencies were (7.13949×10^{29} s^{-1} and 7.13103×10^{29} s^{-1})) giving a theory to simulation ratio of 1.00014, an indication that the collision frequency is correctly simulated.

If boundary conditions are changed to fully accommodating (diffuse), the simulation maintains a constant number of simulators, but the temperature is only on average constant.

2.2. Inflow in a closed box

2.2.1. Motivation

Starting with an empty box (see Figure 1) with one side open, at which a free stream boundary condition is applied, the number of incoming molecules as a function of time is tracked.

2.2.2. Description

The hypersonic inflow boundary condition implemented in SPARTA allows a particular number of molecules to enter the computational domain at every time step. Tracking this number after a number of moves allows a direct comparison with the numerical results Eq. 4.24 of Reference 1.

2.2.3. Parts of the code being checked:

Inflow boundary condition.

2.2.4. Results

The following is a printout of the SPARTA log file.

```
SPARTA (19 Aug 2014)
Created orthogonal box = (0 0 0) to (0.001 0.001 0.001)
Created 1000000 child grid cells
  parent cells = 1
  CPU time = 1.65443 secs
  create/ghost percent = 20.8415 79.1585
Balance grid migrated 0 cells
  CPU time = 0.633072 secs
  reassign/sort/migrate/ghost percent = 4.42998 3.05717 21.9823 70.5305
Balance grid migrated 0 cells
  CPU time = 0.821505 secs
  reassign/sort/migrate/ghost percent = 8.85449 0.884218 7.524 82.7373
Memory usage per proc in Mbytes:
  particles (ave,min,max) = 0 0 0
  grid      (ave,min,max) = 175.552 175.552 175.552
  surf      (ave,min,max) = 0 0 0
  total     (ave,min,max) = 175.552 175.552 175.552
Step CPU T/CPU Np
      0            0              0
     10    2.0094299 4.9765134e-08
Loop time of 2.00948 on 1 procs for 10 steps with 951305 particles

Particle moves    = 5232207 (5.23M)
Cells touched     = 8206731 (8.21M)
Particle comms    = 0 (0K)
Boundary collides = 20366 (20.4K)
Boundary exits    = 0 (0K)
SurfColl checks   = 0 (0K)
SurfColl occurs   = 0 (0K)
Collide attempts  = 0 (0K)
Collide occurs    = 0 (0K)
Reactions         = 0 (0K)
Particles stuck   = 0
```

```
Particle-moves/CPUsec/proc: 2.60376e+06
Particle-moves/step: 523221
Cell-touches/particle/step: 1.5685
Particle comm iterations/step: 1
Particle fraction communicated: 0
Particle fraction colliding with boundary: 0.00389243
Particle fraction exiting boundary: 0
Surface-checks/particle/step: 0
Surface-collisions/particle/step: 0
Collision-attempts/particle/step: 0
Collisions/particle/step: 0
Reactions/particle/step: 0

Move  time (%) = 0.355214 (17.6769)
Coll  time (%) = 0 (0)
Sort  time (%) = 0 (0)
Comm  time (%) = 1.81198e-05 (0.000901717)
Outpt time (%) = 0.00825596 (0.410851)
Other time (%) = 1.64599 (81.9113)

Particles: 951305 ave 951305 max 951305 min
Histogram: 1 0 0 0 0 0 0 0 0 0
Cells:    1e+06 ave 1e+06 max 1e+06 min
Histogram: 1 0 0 0 0 0 0 0 0 0
GhostCell: 0 ave 0 max 0 min
Histogram: 1 0 0 0 0 0 0 0 0 0
EmptyCell: 0 ave 0 max 0 min
Histogram: 1 0 0 0 0 0 0 0 0 0
```

2.2.5. Comments

The goal of this simulation is to compare the number of simulators that enter the domain over a period of 10 time steps and compare it to that theoretically predicted in order to verify that the inlet boundary condition introduces the correct number of simulators to the box. In the simulation above, 951305 simulators were introduced in 10 time steps which is in very good agreement (0.14% error) with the theoretically predicted (Eq. 4.24 of Reference 1) for the conditions of test case 1 (zero velocity, 2 torr pressure, 273.15 K, argon gas.).

2.3 Internal energy relaxation

2.3.1. Motivation

The ability of the internal energy exchange routines to achieve complete thermal equilibration under conditions of thermodynamic equilibrium is examined.

2.3.2. Description

Large number of molecules of a diatomic gas are allowed to perform energy exchange through successive molecular collisions (specular walls). Initially the molecules are assumed to have all their energy in the translational mode. Eventually, all three energy modes, translational, rotational and vibrational, reach thermal equilibrium.

2.3.3. Parts of the code being checked

Collisions, non-reactive energy exchange.

2.3.4. Results

Figure 2 presents the relaxation of the three available degrees of freedom towards equilibration. Initially all the energy is stored in the translational mode. After successive molecular collisions, all of which are allowed to exchange energy, thermal equilibration is achieved.
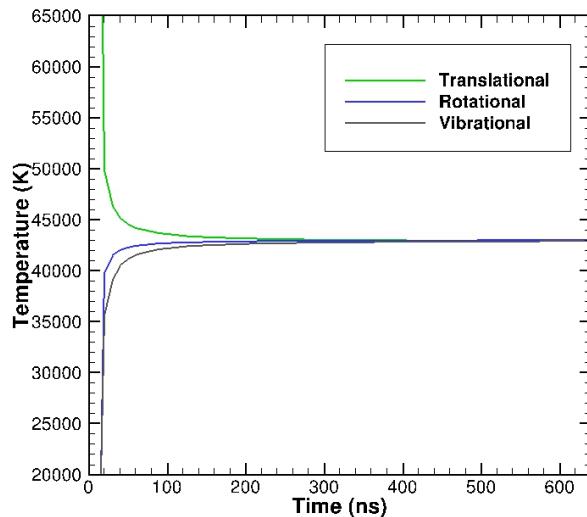
Figure 2. Relaxation of internal degrees of freedom

2.3.5. Comments

This test case demonstrates the ability of the code to reproduce thermodynamic equilibrium. The rotational and vibrational relaxation numbers were assumed to be unity. Replacing them with realistic ones the results can be compared to measured relaxation rates.

2.4 Chemical reactions in a closed box

2.4.1. Motivation

The ability of the code to reproduce a reaction rate as specified in the chemistry input deck can be tested by simulating the dissociation of a gas within an isothermal closed box.

2.4.2. Description

A large number of simulators are allowed to move and collide in a closed isothermal box. The walls may be specular or diffuse. Diffuse walls will allow internal energy reassignment. In this particular example molecular nitrogen dissociation reactions are simulated. However, The reaction is set not to change the species type or reduce the energy content of the flow.

2.4.3. Parts of the code being checked

Chemical reactions, energy exchange, post collision velocity scattering, boundary conditions.

2.4.4. Results

The following is a printout of the SPARTA log file.

```
SPARTA (19 Aug 2014)
Created orthogonal box = (-0.5 -0.5 -0.5) to (0.5 0.5 0.5)
Created 1 child grid cells
  parent cells = 1
  CPU time = 0.000795841 secs
  create/ghost percent = 98.2624 1.73757
Created 1000000 particles
  CPU time = 0.221867 secs
Memory usage per proc in Mbytes:
  particles (ave,min,max) = 102.997 102.997 102.997
  grid      (ave,min,max) = 1.83113 1.83113 1.83113
  surf      (ave,min,max) = 0 0 0
  total     (ave,min,max) = 104.828 104.828 104.828
Step CPU Np Natt Ncoll temp Nattave
      0           0 1000000       0        0    99956.821           0
     10   1.4615109 1000000  401022    77222    99860.643      3911.6
     20    2.902142 1000000  401022    77612    99811.567     3922.25
     30    4.330446 1000000  401022    77550    99629.689        3924
     40   5.7504139 1000000  401023    77286    99550.496    3924.825
     50    7.179774 1000000  401022    77407    99531.592     3926.84
     60     8.60816 1000000  401022    77385    99474.452   3927.9167
     70   10.030703 1000000  401022    77600    99506.309   3929.6143
     80   11.454959 1000000  401023    77380        99448     3927.95
     90   12.896433 1000000  401022    77064    99416.421   3927.3444
    100   14.342392 1000000  401022    77280    99392.408     3925.49
Loop time of 14.3424 on 1 procs for 100 steps with 1000000 particles

Particle moves    = 100000000 (100M)
Cells touched     = 100000000 (100M)
Particle comms    = 0 (0K)
Boundary collides = 1292185 (1.29M)
Boundary exits    = 0 (0K)
SurfColl checks   = 0 (0K)
```

```
SurfColl occurs   = 0 (0K)
Collide attempts  = 40102217 (40.1M)
Collide occurs    = 7725885 (7.73M)
Reactions         = 392549 (0.393M)
Particles stuck   = 0

Particle-moves/CPUsec/proc: 6.97233e+06
Particle-moves/step: 1e+06
Cell-touches/particle/step: 1
Particle comm iterations/step: 1
Particle fraction communicated: 0
Particle fraction colliding with boundary: 0.0129219
Particle fraction exiting boundary: 0
Surface-checks/particle/step: 0
Surface-collisions/particle/step: 0
Collision-attempts/particle/step: 0.401022
Collisions/particle/step: 0.0772589
Reactions/particle/step: 0.00392549

Move  time (%) = 1.62788 (11.3501)
Coll  time (%) = 11.8648 (82.7251)
Sort  time (%) = 0.788682 (5.49895)
Comm  time (%) = 0.000124454 (0.000867737)
Outpt time (%) = 0.0608909 (0.424551)
Other time (%) = 6.36578e-05 (0.000443843)

Particles: 1e+06 ave 1e+06 max 1e+06 min
Histogram: 1 0 0 0 0 0 0 0 0 0
Cells:     1 ave 1 max 1 min
Histogram: 1 0 0 0 0 0 0 0 0 0
GhostCell: 0 ave 0 max 0 min
Histogram: 1 0 0 0 0 0 0 0 0 0
EmptyCell: 0 ave 0 max 0 min
Histogram: 1 0 0 0 0 0 0 0 0 0
```

2.4.5 Comments

The goal of this simulation is to compare the simulated reaction rate to that of the Arrhenius rate k that was used as part of the input deck. Repeating this simulation for a number of temperatures, the reaction rate k(T) as a function of temperature is obtained. As shown in Table 1, the Arrhenius rate and the simulated reaction rate are in good agreement for the range of temperatures studied.

Table 1. Comparison of theoretical and simulated reaction rates for nitrogen dissociation.

Temperature (K)	Arrhenius rate (ln(k(T))	Simulated Rate (ln(k(T))
15000	-17.89	-18.07
20000	-17.27	-17.36
25000	-16.93	-16.98
30000	-16.73	-16.75
35000	-16.61	-16.60
40000	-16.53	-16.50

2.5. Flow in a closed cylinder

2.5.1. Motivation

For many applications, assumptions about axial symmetry can be made to reduce the complexity of the simulation. Similar to two-dimensional flow assumptions in DSMC, the domain is represented on two axes; however, out-of-plane motion cannot be ignored, and azimuthal velocities must be considered. Therefore, the axisymmetric boundary condition requires additional geometric transformations that must be performed at every time step. To verify that the implementation of this boundary condition is correct, axially symmetric flow in a closed cylinder was studied to show that molecule number densities and temperatures are conserved as the flow approaches thermodynamic equilibrium.

2.5.2. Description

Consider a cylinder with a height of 1 mm and a radius, r, of 1 mm. The container walls are specularly reflective and located at x = 0 mm, x = 1 mm, and r = 1 mm as diagrammed in Figure 3. The azimuthal, or out-of-plane, boundary condition is periodic. The container is initially filled by a monatomic gas at a nominal temperature and pressure. Any initial state will eventually reach thermodynamic equilibrium. In this particular case, room-temperature argon molecules populated the cylinder with no initial velocity and hard-sphere collisions are assumed. As a result, the flow is governed purely by diffusion maintaining the thermal equilibrium of the initial state.
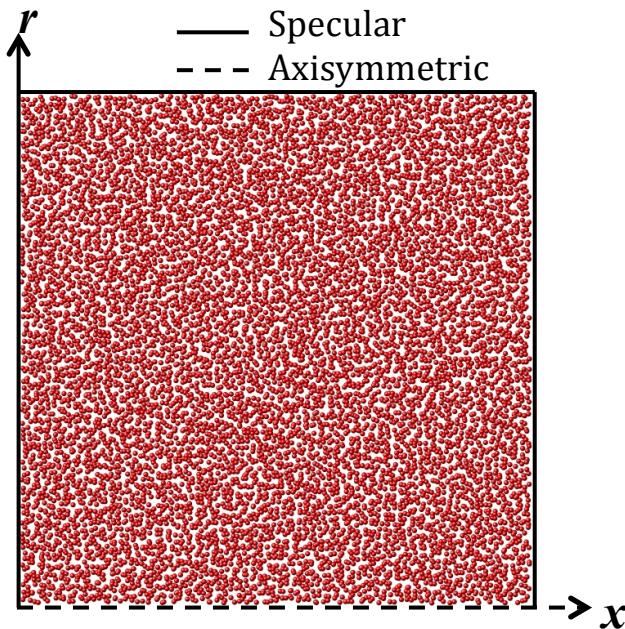
Figure 3. Initial configuration and boundary conditions for flow in a closed cylinder

2.5.3. Parts of the code being checked

Axially symmetric boundary condition, axisymmetric move (molecule cloning and deleting.)

2.5.4. Results

To verify the implementation of the axisymmetric boundary condition, a simulation is run per the above description such that the gas within the cylinder reaches equilibrium. As the flow equilibrates, the number density and temperature of the molecules can be tracked to determine deviations from the initial conditions. Any bias or significant difference would indicate issues with the implementation of the algorithm.
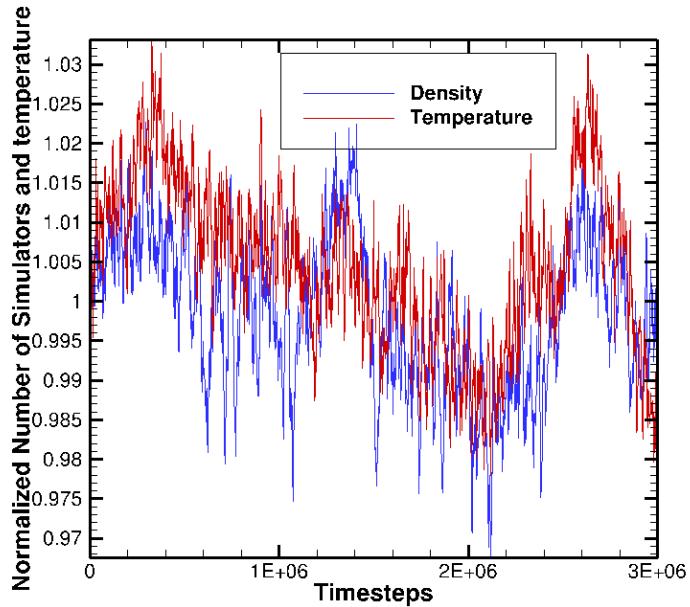
Figure 4. Normalized number density and temperature for flow in a closed cylinder

As seen in Figure 4, the density and temperature of the molecules normalized by the initial state are approximately equally distributed with no clear bias or trend that would indicate loss of molecules or errors in collisional calculations. Random walks that could appear in long duration axially symmetric calculations also appear to be avoided.

2.6 Sonine polynomials

2.6.1. Motivation

All previous test cases involved gases at equilibrium. Although this is an important limit for the code to reach, it is hardly ever the desired state. This test case examines the ability of the code reproduce a known, non-equilibrium, steady state of a gas.

2.6.2. Description

A gas under the influence of an external shearing force or thermal gradient is expected to reach a non-equilibrium steady state. For a modest gradient (near-equilibrium non-equilibrium), the state of the gas can be adequately described by the Chapman-Enskog Theory [9]. The non-equilibrium state of the gas can be best described by the velocity distribution function. First-order CE theory generates a closed-form expression for the velocity distribution function in terms of macroscopic hydrodynamic fields and their gradients: $f = f^{(0)}(1 + \Phi^{(1)} + \Psi^{(1)})$ where $f^{(0)} = n\exp[-\tilde{c}^2]/\left(\pi^{3/2}c_m^3\right)$ and $\Phi^{(1)} = -\left(8/5\right)\tilde{A}[\tilde{c}]\tilde{\mathbf{c}} \cdot \tilde{\mathbf{q}}$ $\Psi^{(1)} = -2\tilde{B}[\tilde{c}](\tilde{\mathbf{c}} \circ \tilde{\mathbf{c}} : \tilde{\tau})$.
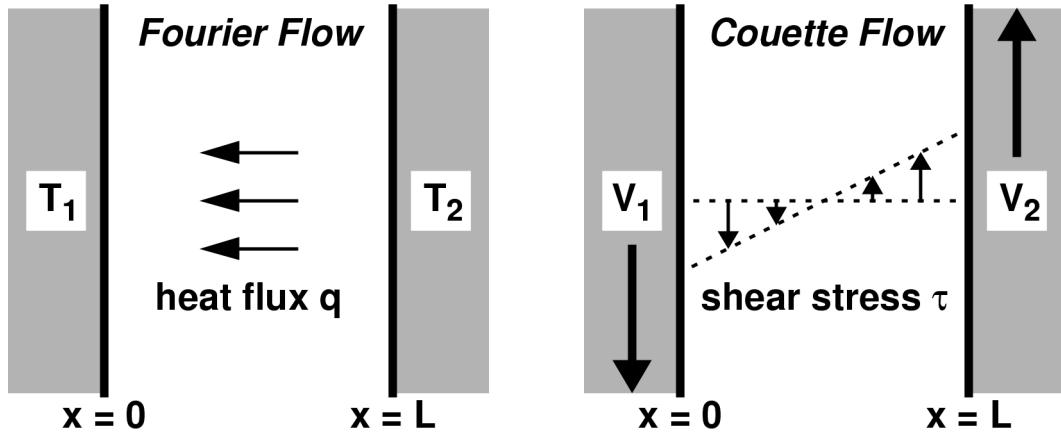
Figure 5. Schematic of Fourier and Couette flow

Here, $f^{(0)}$ is the equilibrium (Maxwellian) distribution, $\Phi^{(1)}$ and $\Psi^{(1)}$ are the first-order non-equilibrium perturbations from this distribution, $c_m = \sqrt{2k_B T/m}$ is the most probable molecular thermal speed for the equilibrium distribution, m is the molecular mass, n is the number density, T is the temperature, k_B is the Boltzmann constant, $\mathbf{c} = \mathbf{u} - \mathbf{U}$ is the thermal velocity of a molecule, $\mathbf{u} = (u,v,w)$ is the velocity of a molecule, $\mathbf{U} = (U,V,W) = \langle\mathbf{u}\rangle$ is the average value of $\mathbf{u}$, $\tilde{\mathbf{c}} = \mathbf{c}/c_m$ is the normalized molecular thermal velocity, $\tilde{\mathbf{c}} \circ \tilde{\mathbf{c}} = \tilde{\mathbf{c}}\tilde{\mathbf{c}} - (\tilde{c}^2/3)\mathbf{I}$ is a traceless dyadic, $\tilde{\mathbf{q}} = \mathbf{q}/(mnc_m^3)$ and $\tilde{\tau} = \tau/(mnc_m^2)$ are the non-dimensional heat-flux vector and shear-stress tensor, and $\tilde{A}$ and $\tilde{B}$ are expansions in the Sonine polynomials $S_j^{(k)}$:

$$\tilde{A}[\tilde{c}] = \sum_{k=1}^{\infty}(a_k/a_1)S_{3/2}^{(k)}[\tilde{c}^2], \ \tilde{B}[\tilde{c}] = \sum_{k=1}^{\infty}(b_k/b_1)S_{5/2}^{(k-1)}[\tilde{c}^2] \text{ where } S_j^{(k)}[\xi] = \sum_{i=0}^{k}\frac{(j+k)!(-\xi)^i}{(j+i)!i!(k-i)!}.$$

2.6.3. Parts of the code being checked

Molecular advection, molecular tracking in a 3-D domain, surface reflections, particle collision frequency, conservation of mass, momentum and energy, polyatomic gas collisions, onitial conditions, diffuse, specular wall boundary condition, load balancing

2.6.4. Results

Figure 6 and Figure 7 present the temperature and velocity profiles as calculated with SPARTA and DSMC1 (a modified code based on G.A Bird's DSMC1 [1]), the code which was used in Reference 3. Figure 8 and Figure 9 present the Sonine polynomial coefficients as calculated by SPARTA in comparison with the theoretically calculated Sonine coefficients [3]. In both cases (DSMC1 and SPARTA), a hard-sphere molecular model was assumed.
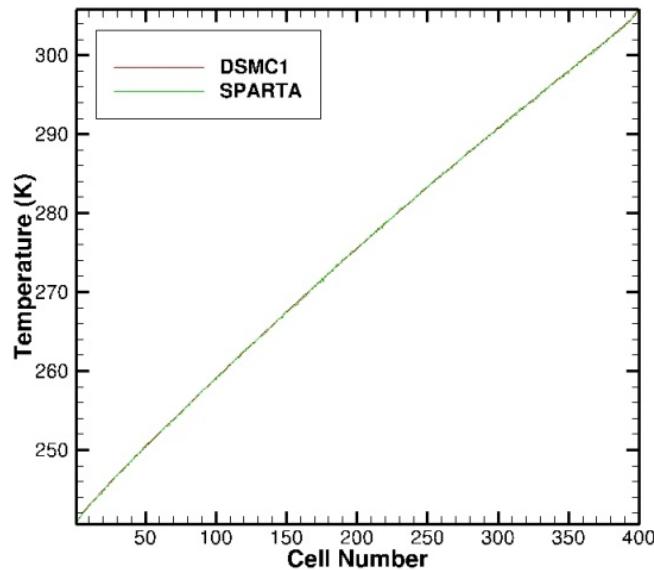
Figure 6. Temperature profile
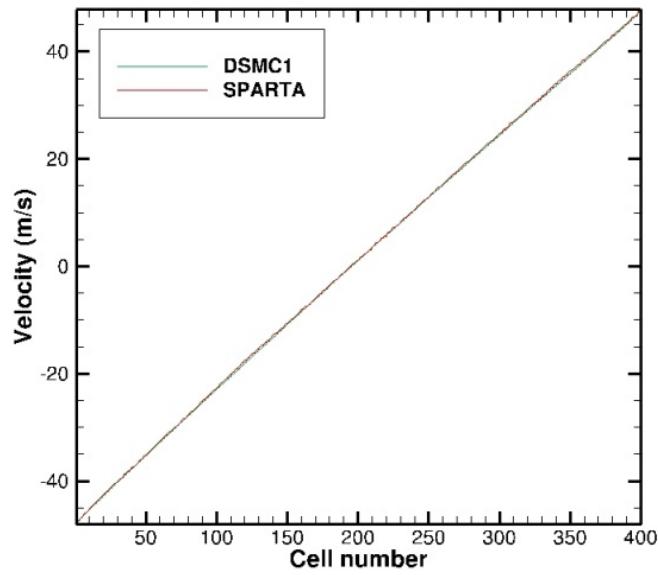
Figure 7. Velocity profiles
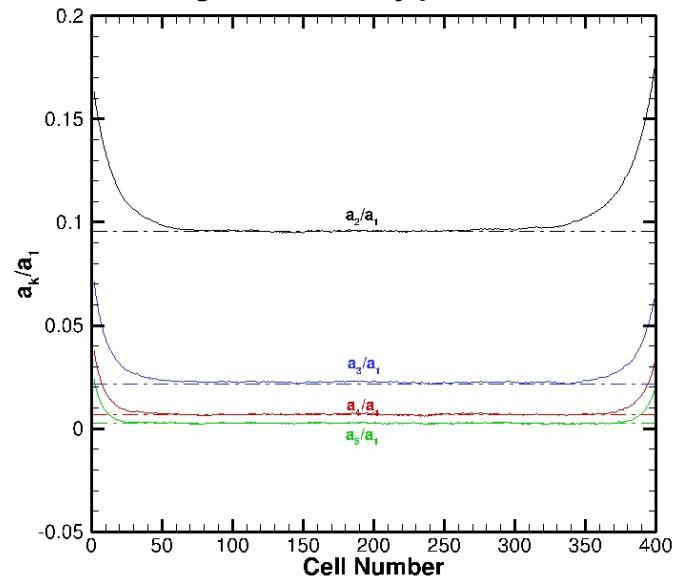
Figure 8.   Sonine Coefficients
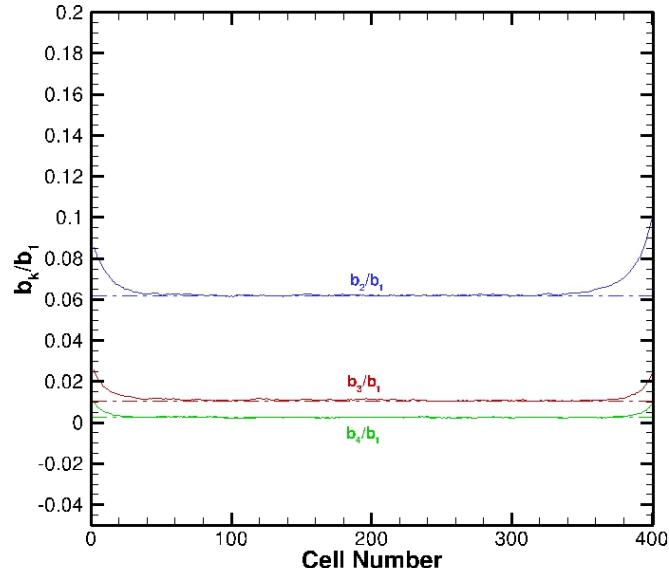
Figure 9. b_k / b_1 **Sonine Coefficients.**

2.6.5. Comments

The calculation of the Sonine polynomials is probably the most comprehensive test case for monatomic gases. Since the Sonine polynomials represent the velocity distribution function outside the Knudsen layers, their accurate calculation indicates that the simulation reproduces the correct velocity distribution function.

2.7. Volume around a sphere

2.7.1. Motivation

SPARTA uses Cartesian cells to discretize physical space. Cells near surfaces are "cut" by the geometry. For the accurate calculation of the collision frequency,- an accurate calculation of the cut-cell volume is needed.

2.7.2. Description

The "flowfield" volume, ie. the volume between the bounding box and an enclosed sphere is calculated. Two different-resolution spheres are used. SPARTA calculates the volume of the bounding box outside the enclosed sphere. This volume is also calculated analytically.

2.7.3. Parts of the code being checked

Gas-phase and surface grid description.

2.7.4. Results

Using a 1,200 (Figure 11) and a 15,000 (Figure 11) surface cell element description of a 1 m-radius sphere, the flow field volume of a 4×4×4 bounding box is calculated. The results are presented in Table 2. Volume calculations for two different-resolution spheres. As the surface resolution increases, the calculated volume approaches the theoretical value. Evidently the theoretical volume is retrieved for a number of cells tending to infinity.
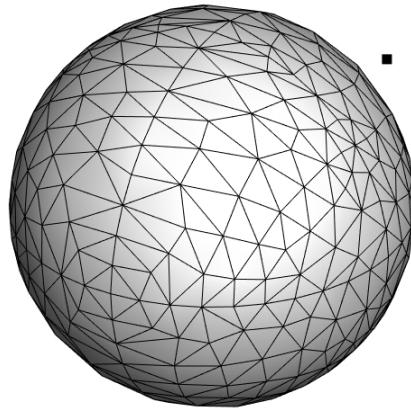
Figure 10. 1,200 surface element sphere description.
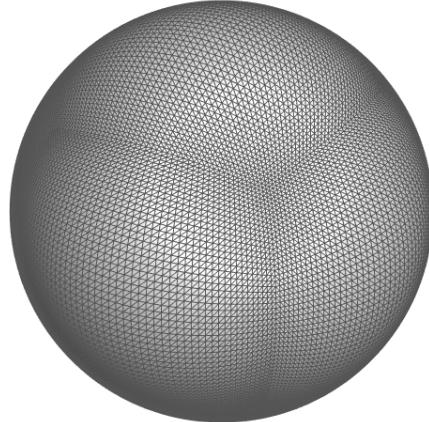
Figure 11. 15,000 surface element sphere description.

.

Table 2. Volume calculations for two different-resolution spheres.

Method/resolution	Flow-field volume (m^3)
1200 elements	59.86171
15000 elements	59.81132
Theory	**59.81121**

The volume appears to be converging to the right volume from higher values. This is because the simulated sphere is contained in the unit radius theoretical sphere. Therefore the volume of the sphere is always less than the theoretical limit, and the flow-filed volume greater than the theoretical limit. It should be noted that the volume calculation is insensitive to the gas-phase grid resolution. The previous volume calculations were repeated with 10^3 ($10\times10\times10$) and a 100^3 ($100\times100\times100$) grids, yielding identical results.

2.7.5. Comments

The goal of this simulation is to examine the ability of the code to calculate the cut-cell volumes accurately. Since these cells are usually near body cells, the accurate calculation of the volume is critical for the simulation of the correct collision frequency, and therefore transport properties to the surface.

2.8. MIR space station

2.8.1. Motivation

SPARTA uses a grid generation function to handle complex shapes and geometries. As an example and demonstration of this feature, a grid has been created from a representative geometry of the MIR space station and used in a stylized re-entry simulation.

2.8.2. Description

The simulation of flow around the MIR space station is an example of the complexity the SPARTA grid generation function can accomdate. The MIR geometry comprises 12 concave bodies with thicknesses varying from 1 cell to multiple cells. Figure 12 shows the surface grid geometry for the MIR space station. The surface description was provided by NASA JSC (G.J. LeBeau).
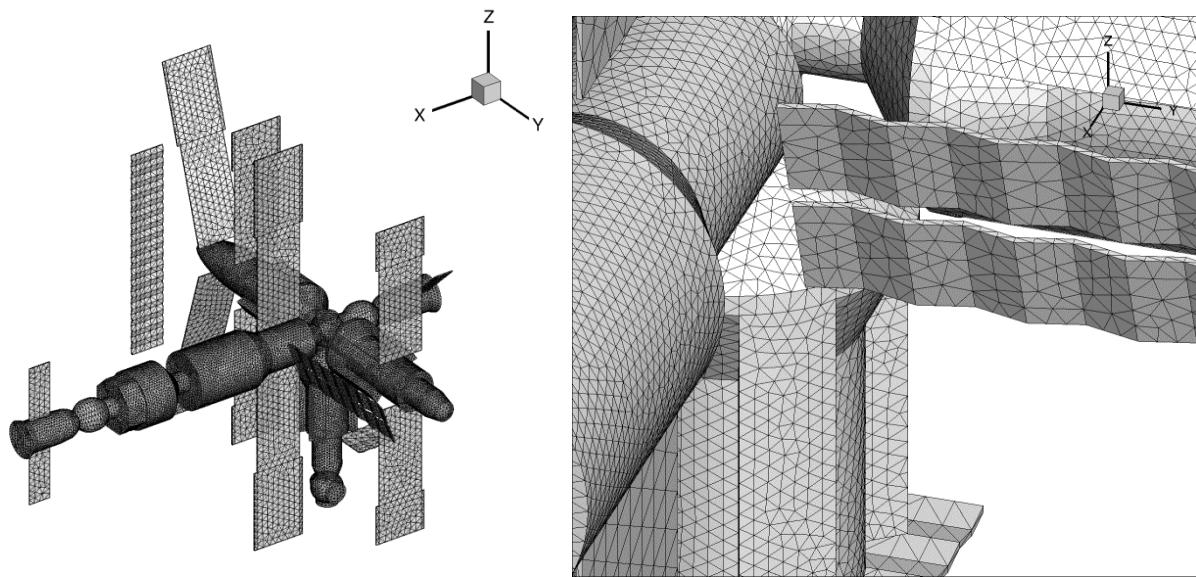
Figure 12. Diagrams of the MIR grid.

In SPARTA, a surface is a collection of surface elements that represent the surface of one or more physical objects which will be embedded in the global simulation box. The surface elements are triangles in 3D or line segments in 2D. Within the input deck, the user can specify a number of options to control the way in which the surface elements are treated. Firstly, particle collisions with the surface must be designated through a surface collision model, which can optionally handle surface chemistry. Additionally, geometric scaling, rotation and placement of the surface elements within the global box can be specified. For simplicity, the MIR space station is centered within the global grid with diffuse particle reflections from the surfaces prescribed. The global box is given open boundary conditions with inflow of air at 7500 m/s at an angle of 30° below the horizontal plane.

2.8.3. Parts of the code being checked

Gas-phase and surface grid description, molecular collisions, surface reflections, chemical reactions, energy exchange. Complex grid generation with cut cells. Diffuse surface reflections. Reading surface geometry. Computing surface properties.

2.8.4. Results

With surface geometries, SPARTA can calculate and output properties on the surface elements. Figure 13 not only shows slices across the y- and z-planes of the global box with temperature contours normalized by the maximum temperature of the flow but also shows contours of absorbed heat flux on the surface elements of MIR normalized by the maximum value.
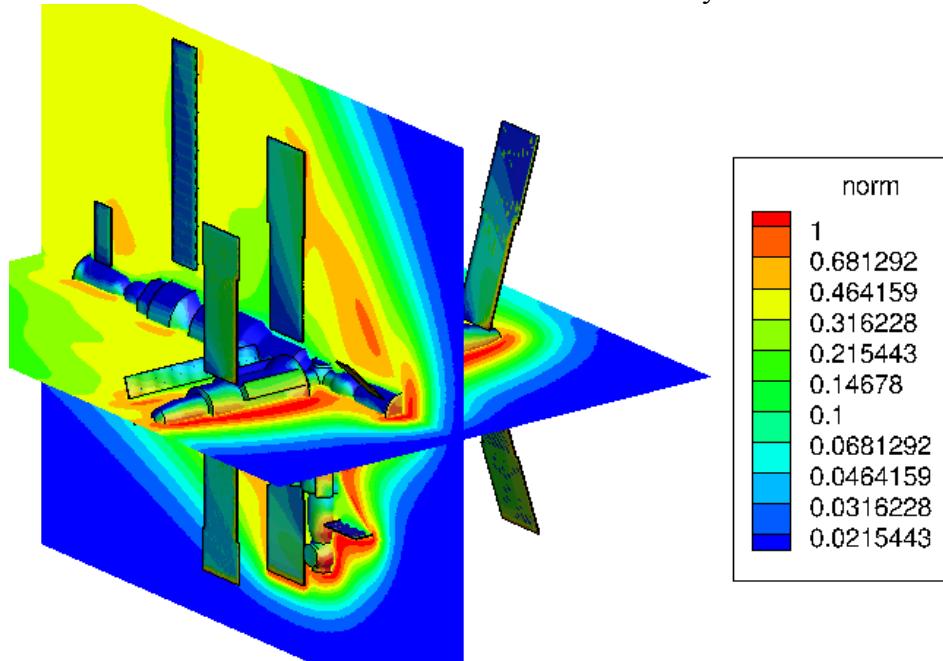
Figure 13. Normalized contours of temperature and surface heat flux on MIR.

2.9 Grid adaptation

2.9.1. Motivation

Spatially resolving three-dimensional calculations can be particularly expensive. Thus, a grid adaptation may be required to resolve the grid where needed. SPARTA offers the ability to create multiple level grids, up to 16 in 64 bit architectures.

2.9.2 Description

In SPARTA, the computational grid can be adapted to match a particular solution. Currently this operation is "static". In the following examples, part of the flow-field domain was refined to a second level, where each cell was partitioned in 8 level-2 cells.

2.9.3. Parts of the code being checked

Grid generation, grid adaptation, cut-cell volume calculation.

2.9.4. Results

Figure 14 shows a variation of the computational grid used in the previous simulation, where the center part of the grid has been adapted to a second level by 2×2×2.
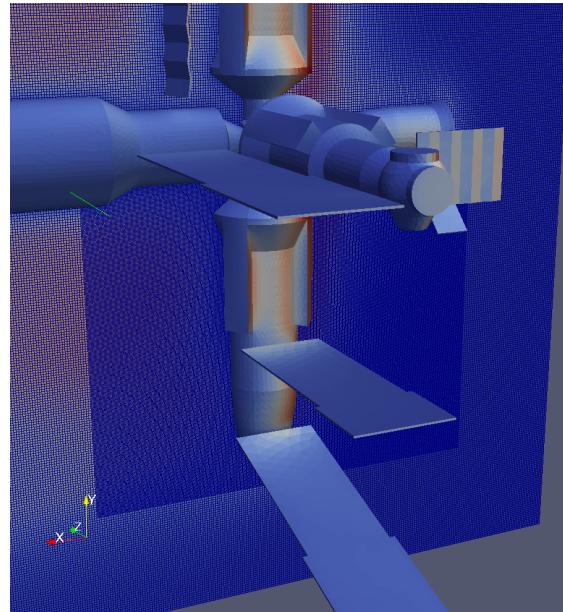
Figure 14. Grid adaptation around MIR space station

Likewise, Figure 15 presents the refinement of a small area around the nose tip of a space-shuttle-like geometry. In both cases, the background properties are seamlessly reproduced.
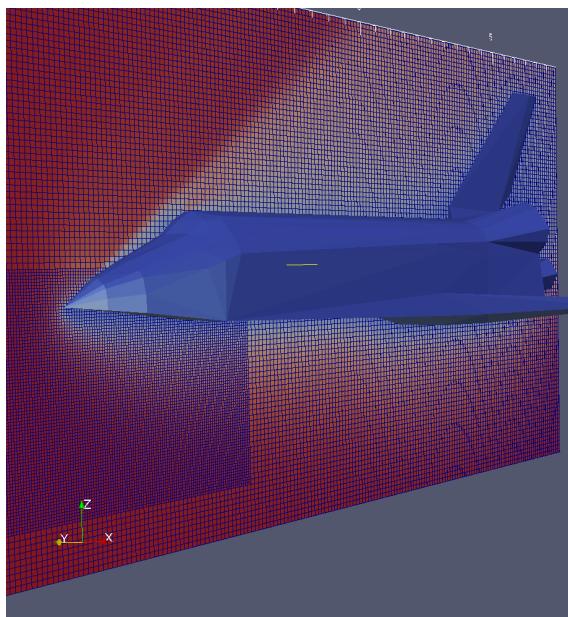
Figure 15. Grid adaptation around a space shuttle-like geometry

2.10. Flow around a sphere

2.10.1. Motivation

The classical fluid dynamics problem of flow around a sphere is simulated to demonstrate the functionality of SPARTA options for simplifying computations of symmetric flows. The implementation of symmetric boundary conditions can be checked by comparing results from simulations of the full, half, and quarter flow fields. Additional checks to the validity of the SPARTA results can be performed by running the same test case in NASA's established DSMC Analysis Code (DAC) [10] and comparing the results of the two codes for quantifiable parameters like maximum number density, temperature, velocity and drag force on the sphere.

2.10.2. Description

The nominal case is defined by a 1-m-diameter sphere centered in a 10-m cube composed of 100×100×100 cells. The boundary conditions of all sides are open, while argon gas at a temperature of 300 K and velocity of 2500 m/s flows into the grid along the x axis. Variations on this nominal case make use of options available to the user to simplify the simulation based on symmetry planes. Results from simulations of a half sphere and a quarter sphere are shown here for reference. In this example, the input deck for the full sphere uses open boundaries on all six sides and also allows inflow of particles through these open boundaries. To adjust to the symmetric geometry, the input deck must be changed to clip the sphere at the planes of symmetry. Also, the box and grid must be redefined to the symmetric geometry. The boundary conditions at the planes must be updated to specularly reflect particles and closed to the injection of new particles. The table below shows the difference in inputs for the full, half, and quarter geometries.

Table 3. Input deck modifications to reduce the flow around a sphere by symmetry

Command	Full Sphere Arguments	Half Sphere Arguments	Quarter Sphere Arguments
boundary	o o o	o ro o	o ro ro
create_box	-5.0 5.0 -5.0 5.0 -5.0 5.0	-5.0 5.0 0.0 5.0 -5.0 5.0	-5.0 5.0 0.0 5.0 0.0 5.0
create_grid	100 100 100	100 50 100	100 50 50
fix in inflow air	All	xlo xhi yhi zlo zhi	xlo xhi yhi zhi
read_surf	data.big.tri 1	data.big.tri 1 clip	data.big.tri 1 clip

Figure 16 shows slices of temperature contours for the full, half, and quarter sphere cases. The slices are shown along the y- and z- planes at a distance of 0.1 m from each axis. It is worthy to note here that the results from the simplified symmetric runs do not fully represent the flow around the sphere as the data output is cell-centered and therefore cannot be interpolated to the bounds of the region, which includes the sphere centerline.
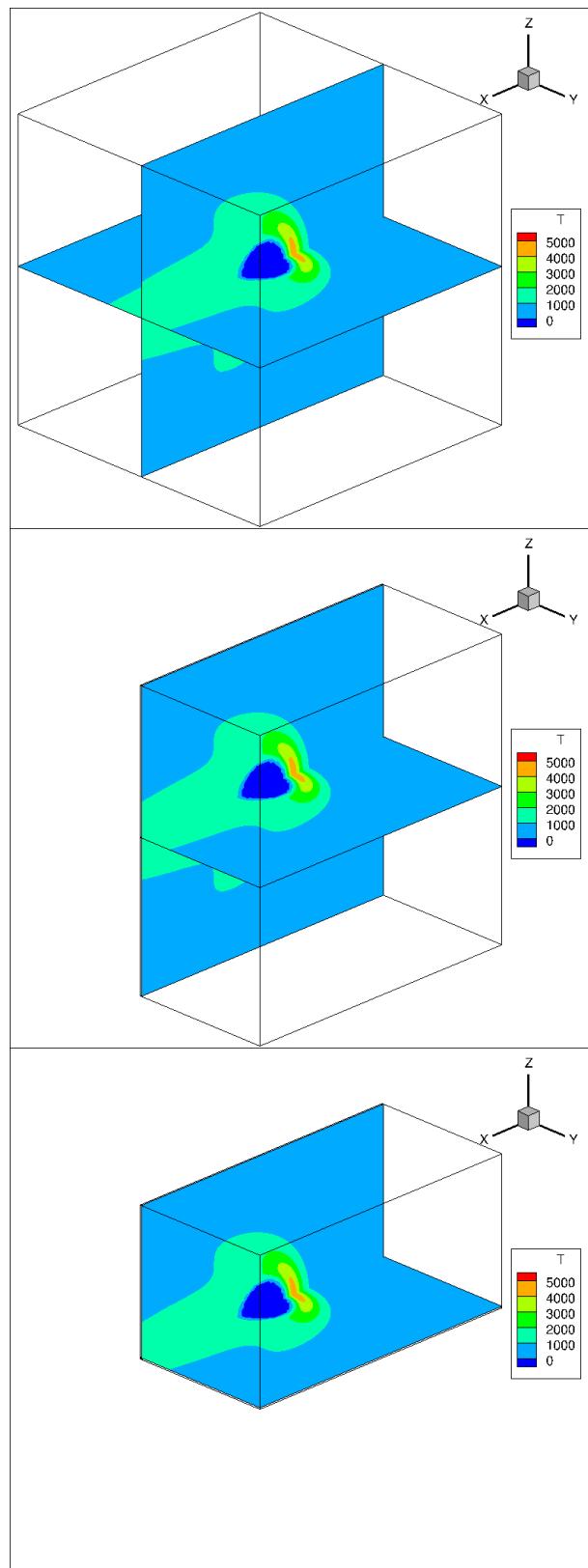
Figure 16. Temperature contours for flow around a sphere using geometric symmetry to simplify computations.

These results illustrate that the simulations produce identical results when utilizing the specular boundary condition at the planes of symmetry.

2.10.3. Parts of the code being checked

Surface clipping. Diffuse reflection boundary condition.

2.10.4. Results

To verify the SPARTA simulations of flow around a sphere, the results for the full sphere case shown above were compared to a simulation of identical input parameters executed using DAC. Contour comparisons of temperature, velocity, and number density are shown in Figure 17 while the maximum values for these flow properties are given in Table 4. The contours are generally in good agreement and capture the phenomenology of the flow including the wake structure. Slight variances in the resultant flow properties may be attributed to differences in the duration of the simulations.

Table 4. Maximal values of flow properties for flow around a sphere

| Property	DAC	SPARTA
Temperature (K)	4677	4506
x Velocity (m/s)	2501	2495
Number Density ($1/m^3$)	1.024×10^{-5}	1.282×10^{-5}
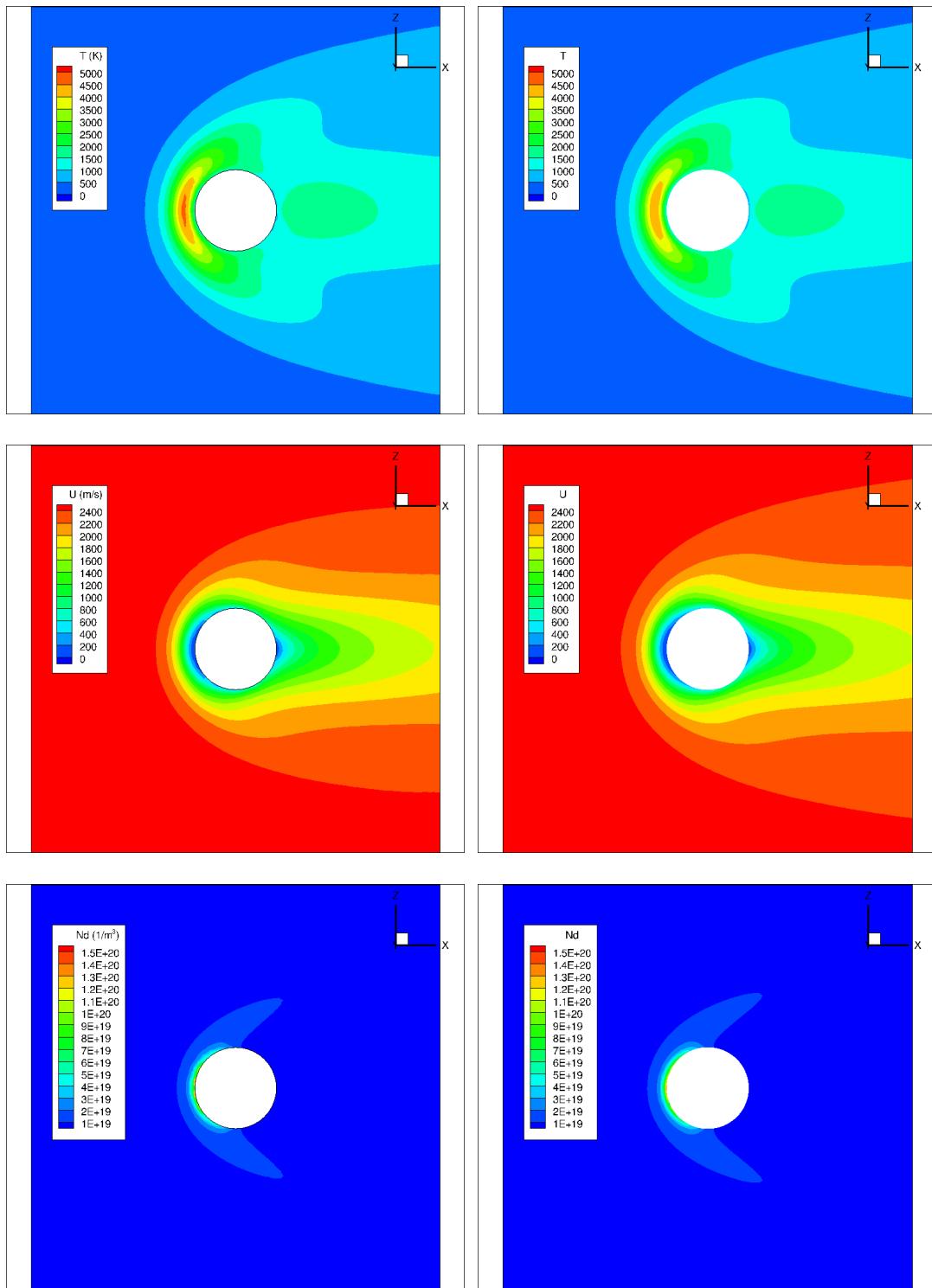
Figure 17. Contours for flow around a sphere on the y-plane for (a) temperature, (b) velocity in x direction, and (c) number density, as calculated by DAC (first column) and SPARTA (second column).

2.11. Richtmyer-Meshkov instability

2.11.1 Motivation

SPARTA has an unprecedented capability to utilize the largest supercomputers available. As a challenging scenario with broad scientific applications, the Richtmyer-Meshkov instability (RMI) was selected to attempt to extend SPARTA to flow regimes untested by traditional DSMC codes. RMI is a hydrodynamic instability that results from the misalignment of density and pressure gradients across a fluid interface perturbation due to an impulsive acceleration, such as a shock wave. RMI is commonly observed in inertial confinement fusion and supernova events.

2.11.2 Description

Following the refraction of the incident shock wave at the interface between two gases, a distorted shock is transmitted into the second gas. Then, and depending on the properties of the two gases, a distorted shock or rarefaction wave is reflected back in to the first gas. As a result of this process, the interface is impulsively accelerated to a constant velocity and travels in the same direction as the transmitted shock. Vorticity is deposited baroclinically along the interface wherever the density gradient (due to the initial interface) and pressure gradient (due to the shock) are misaligned. This results in time dependent growth of the initial perturbation. Initially the growth is linear. When the amplitude of the perturbation becomes comparable to its characteristic wavelength the growth becomes non-linear. The non-linear growth regime is followed by a regime influenced by the Kelvin-Helmoltz instability, which causes roll-up structures resembling mushrooms in the heavy fluid.

2.11.3. Results

In the simulations shown in Figure 18, a region 1mm wide and 4mm high with 1mm depth, is filled with two monatomic gases (helium and argon) at atmospheric pressure and temperature. The gases are initially separated such that the lighter gas (helium) rests atop the heavier gas (argon) with an interface, sinusoidally perturbed with a wavelength of 0.5 mm and a crest to trough amplitude of 0.2 mm, as shown in Figure 18a. In this case, a Mach = 1.2 shock in helium enters the computational domain from the top. Figure 18 show late-time stages of development of the instability in which Kelvin-Helmoltz instability occurs and progresses eventually to turbulence. This qualitative description of RMI is in good agreement with published experimental data as shown in Morgan et al. (2012).
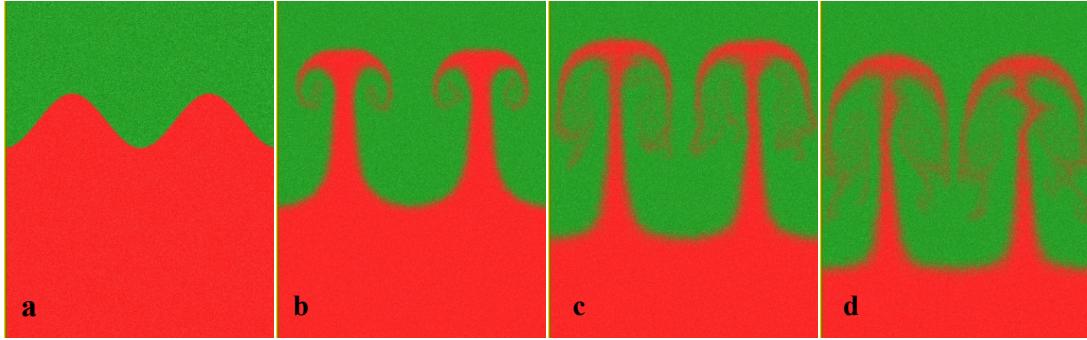
Figure 18. Development of the Richtmyer-Meshkov instability.

The amplitude of the instability is often used as a more quantifiable measure of RMI growth, and comparisons to theory and empirical correlations are common in the literature. In Figure 19, the non-dimensional amplitude growth is shown as a function of time as calculated by DSMC and the impulsive linear model (Richtmyer, 1960). We note that, while the two models are in good agreement for small times, as the perturbation amplitude increases and becomes comparable to unity, non-linear phenomena appear.
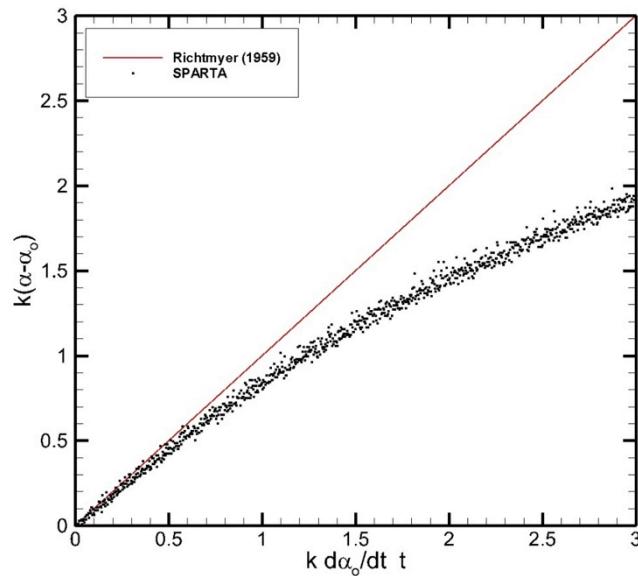

Figure 19. Time-dependent growth of the Richtmyer-Meshkov instability.

2.12. Parallel efficiency

2.12.1. Motivation

The advantages of DSMC compared to other methods that predict flows in the non-equilibrium regime come at a cost: DSMC is computationally intense like most Monte Carlo methods. Therefore, its successful application to real problems may depend heavily on the parallel performance of the particular implementation. Monte Carlo methods usually have good parallel performance. This is because the workload depends mainly on the simulators within a cell: there is relatively less need to communicate information between cells. DSMC codes have demonstrated near-linear scaling up to several thousand processors. With the advent of massively parallel computers, problems considered impossible only a few years ago, such as transient three-dimensional simulations, are currently within reach. Here, the test case for flow in a closed box is revisited to determine the parallel efficiency of SPARTA.

2.12.2. Description

Since next-generation machines will require hybrid programming approaches (MPI + node-based parallelism), the DSMC kernels (particle moves, per-cell collisions and chemistry) were reworked within SPARTA in a manner that allows for fine-grain parallelism within a node. To estimate the parallel efficiency of SPARTA, the flow inside a closed, isothermal box was simulated. All physical and numerical parameters are fixed except for the cells per processor and the size of the domain, which is adjusted to keep the number of particles per cell constant (10). Maintaining the same number of particles per cell increases the communication load as the number of processors is increased. Thus, this constitutes a worst-case scenario for a scaling assessment. The number of cells was ranging from 10^4 to 10^{11}.

2.12.3. Parts of the code being checked

Node and core-level communications.

2.12.4. Results

Figure 20 to Figure 22 present the performance of SPARTA on Sequoia, a 17-Pflop, 1.57-million core platform at Lawrence Livermore National Laboratory. Here, the performance is defined as the number of particle moves (time steps × number of particles) per CPU second and node. In both plots, the x-axis is node count and the y-axis is per-node performance. Single curves show strong scaling. Weak scaling is demonstrated following performance parallel to the x-axis from curve to curve. The left-side speed-up (in each curve) is a cache effect as the particle/processor count gets smaller; the right-side slow-down is when there are too few particles/processor.

Through hyper-threading, up to 4 tasks per core were utilized, while each node employed 16 cores. A total of 98,304 nodes were used in this scaling study. The left plot presents the performance for 1 task/core and the right one for 4 tasks/core. The maximum number of particles

was 1 trillion (in 100 billion cells). The maximum number of particles simulated was 3 trillion in 1 trillion cells. Meaningful simulations were performed up to a maximum of 2.6 million tasks
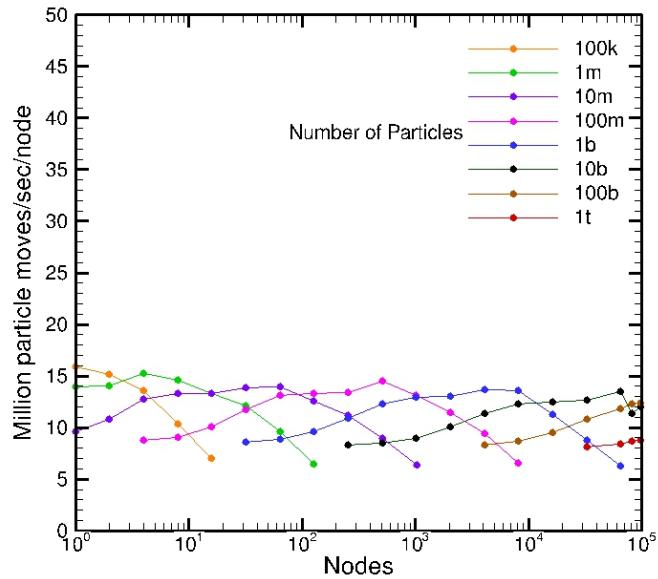

**Figure 20. Weak and strong scaling of SPARTA on Sequoia using 1 MPI task/node.**
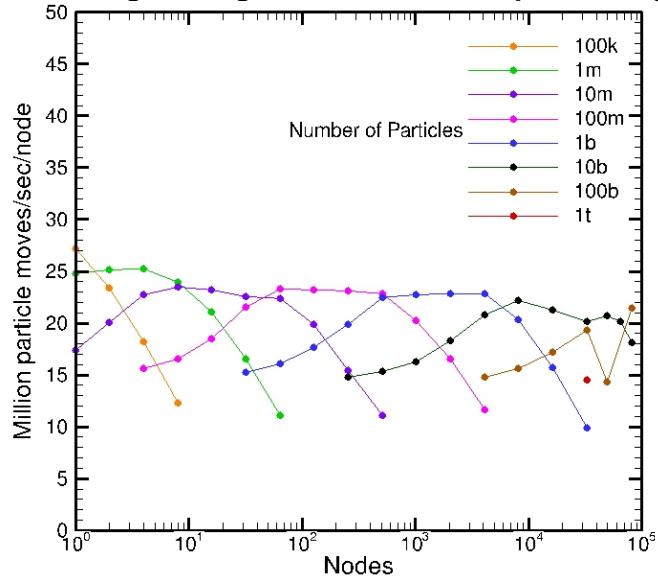
**Figure 21. Weak and strong scaling of SPARTA on Sequoia using 2 MPI tasks/node.**
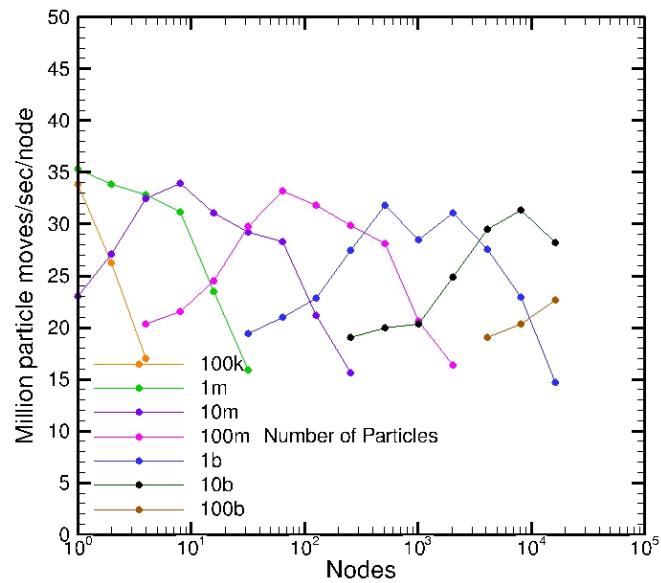
Figure 22. Weak and strong scaling of SPARTA on Sequoia using 4 MPI tasks/node.

3. CONCLUSIONS

Sandia's new DSMC code SPARTA has been demonstrated to accurately reproduce known solutions for a number of cases, ranging from steady state equilibrium flow fields to continuum transient flow fields. Simulations include flows around complicated three-dimensional vehicles and have been validated against theoretical solutions and other established simulations when possible.

SPARTA has been demonstrated to be able to take advantage of new exascale technologies demonstrating a thousand-fold improvement in computational speed. Such an improvement will bring many problems that are currently impossible to simulate within the reach of DSMC.

4. REFERENCES

1. Bird, G.A., Molecular Gas Dynamics and the Direct Simulation of Gas Flows. 1994. Clarendon Press, Oxford.
2. Wagner, W., "A convergence proof for Bird's direct simulation Monte Carlo method for the Boltzmann equation," J. Stat. Phys. 66, 1011, 1992.
3. Gallis, M. A., Torczynski, J. R., Rader, D. J., "Molecular Gas Dynamics Observations of Chapman-Enskog Behavior and Departures Therefrom in Nonequilibrium Gases", Physical Review E, 69, 042201, 2004.
4. Gallis, M. A., Torczynski, J. R., Rader, D. J., Tij, M., and Santos, A., "Normal Solutions of the Boltzmann Equation for Highly Nonequilibrium Fourier and Couette Flow," *Physics of Fluids*, Vol. 18, No. 1, 2006, paper 017104, pp. 1-15.
5. Edwards, H. Carter, Trott, Christian R., Sunderland, D. "Kokkos: Enabling manycore performance portability through polymorphic memory access patterns", Journal of Parallel and Distributed Computing, ISSN 0743-7315, http://dx.doi.org/10.1016/j.jpdc.2014.07.003, 2014.
6. Plimpton, S. J., Gallis, M. A., SPARTA Direct Simulation Monte Carlo (DSMC) Simulator, Sandia National Laboratories, USA, see http://sparta.sandia.gov.
7. Richtmyer, R. D., Comm. Pure Appl. Math.,13 pp.297-319, 1960.
8. Morgan, R. V., Aure, R., Stockero, J.D., Greenough, J.A., Cabot, W., Likhachev, O.A., Jacobs, J.W., "On the late-time growth of the two-dimensional Richtmyer-Meshkov instability in shock tube experiments," J. Fluid Mech., 712, 2012.
9. Chapman, S., and Cowling, T. G., *The Mathematical Theory of Non-Uniform Gases*, third edition, Cambridge University Press, Cambridge, UK, 1970.
10. LeBeau, G. J., "A Parallel Implementation of the Direct Simulation Monte Carlo Method", *Computer Methods in Applied Mechanics and Engineerin*g, Vol. 174, 1999, pp. 319-337.

DISTRIBUTION

1	MS1320	S. S. Collis	01440
1	MS1321	V. Tikare	01444
1	MS1316	S. J. Plimpton	01444
1	MS0836	B. Hassan	01510
1	MS0840	M. A. Gallis	01513
1	MS0840	T. P. Koehler	01513
1	MS0840	D. J. Rader	01513
1	MS0840	J. R. Torczynski	01513
1	MS0825	R. M. Wagnild	01515
1	MS0899	Technical Library	09536 (electronic copy)