Parallel Hybrid Method for Subsonic Flows

Carolyn R. Kaplan, Junhui Liu and Elaine S. Oran

Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory
Washington, DC 20375 U.S.A.

Abstract: We present a 3-D parallel hybrid continuum-particle method to simulate multi-scale gas flows. The continuum method solves the 3-D compressible Navier-Stokes (NS) equations, while the particle method uses direct simulation Monte Carlo (DSMC). The hybrid code improves computational efficiency by using the particle method only in the rarefied regions, and by using the faster NS approach everywhere else. We discuss the parallel hybrid methodology, present results for two test problems and discuss the coupling and load-balancing challenges inherent when using a hybrid method on parallel architectures.

Keywords: direct simulation Monte Carlo, hybrid method

PACS: 40.47.45.-n

INTRODUCTION

A number of gaseous flow problems of current importance span the range from continuum to rarefied flow. These include flows in microdevices and flows in the earth's and planetary atmosphere. Solution methods that are appropriate for continuum flows are not accurate for rarefied flows, while methods that are appropriate for rarefied flows are usually computationally expensive for continuum flows. Therefore, a hybrid code that allows each method to be used in the appropriate region is an efficient way to solve multi-scale problems.

Over the past 15 years, hybrid methods have been developed by coupling continuum and rarefied codes for 1-D and 2-D multi-scale gas flow problems. Previous work includes studies of various coupling techniques, steady and unsteady flows, adaptive interface, and adaptive mesh and algorithm refinement . A summary of published work on coupled hybrid schemes is presented in Ref. 1.

In this paper, we present a scalable hybrid method [2] that has been developed to simulate multi-scale 3-D gas flows on parallel computers. The flow conditions in problems of interest can range from the continuum regime (Kn<0.01), through the slip flow regime (0.01 < Kn < 0.1) to the transitional regime (Kn>0.1). Flows in the continuum regime are simulated using a 3-D parallel NS code, while flows in the slip flow and transitional regimes are simulated using a 3-D parallel DSMC code. Parallelization within each individual code (NS and DSMC) and for the overall hybrid code is accomplished by Message Passing Interface (MPI). In the following sections, we discuss the hybrid code methodology, present results for validation test cases, and discuss the challenges in coupling the NS and DSMC regions and in maintaining acceptable load-balancing in a hybrid parallel code.

PARALLEL HYBRID METHODOLOGY

Table 1 lists the steps that comprise the parallel hybrid code. At the start of the calculation, the entire computational domain is divided into NS-regions and DSMC-regions, based on the local Knudsen number. As shown in Steps 2 and 3, separate MPI communicators are established for messsage passing within each individual code, while a global MPI communicator is established to exchange data between the codes when they are coupled for the hybrid method. As shown in Step 4, each computation proceeds independently, at

1. Divide computational domain into NS and DSMC regions, based on local Kn.
2. Establish MPI communicator within each code:
   NS_WORLD, contains NS_nproc processors
   DSMC_WORLD, contains DSMC_nproc processors
3. Establish MPI communicator between codes
   MPI_COMM_WORLD
4. Initially, conduct independent NS and DSMC calculations in each region
   Call NS_Initial
   Call DSMC_Initial
5. Calculate appropriate time interval for each cycle of hybrid code
   Cycle_time = min (# NS timesteps*Δt_NS, # DSMC timesteps*Δt_DSMC)
6. Begin cycles:
   a) Exchange data at interface (through MPI_COMM_WORLD)
      DSMC=>NS: Calculate macroscopic properties in DSMC cells at interface, and send pressure and velocity to adjacent NS cell, as boundary condition
      NS=>DSMC: Using properties from adjacent NS cell, generate particles in DSMC reservoir cells
   b) Run independent NS and DSMC calculations for appropriate time interval for cycle
      Call NS_cycle
      Call DSMC_cycle
   c) Return to (a) and repeat until all cycles are completed.

Figure 1. Steps for a parallel hybrid code.

first. The main purpose of initially running each calculation independently is to allow enough time for the DSMC calculation to have achieved sufficient sampling so that the DSMC data are not too statistically noisy, and therefore do not create instability when coupled with the NS code.

Flow in the continuum region is simulated by solving the 3-D compressible Navier-Stokes equations, using Flux-Corrected Transport (FCT) [3-5] as the fluid convection algorithm. FCT is a high-order finite-volume algorithm for integrating generalized continuity equations while maintaining conservation, monotonicity, and positivity. This NS code, TINY3D [6], includes all of the viscous fluxes in the 3-D compressible NS equations, which are calculated by explicit finite differencing and then are included as source terms in the momentum conservation equation.

Flow in the slip-flow and transitional regimes are simulated with the DSMC method [7]. The computational grid for the flowfield is 3-D cartesian, and the user can place diffuse or specular solid surfaces along the flowfield grid lines. Boundary conditions for the flowfield grid can be fixed pressure (subsonic boundary conditions of Nance et al. [8]), free stream, vacuum or symmetry. Variable Hard Sphere [6] particles are used, and the real-to-simulated particle count is set such that approximately 50 particles per cell are maintained.

After running each code independently for a user-specified number of timesteps, we calculate the appropriate time interval for each cycle of the hybrid code, as shown in Step 5. That is, if we want to exchange data every time step (which would be required for an unsteady flow), there would be one NS timestep and one DSMC timestep in each cycle, and the appropriate time interval would be the minimum timestep for the NS and DSMC codes.

The cycling then begins in Step 6, and continues until the specified number of cycles is completed. Data are exchanged at the interface, as shown in Figure 2. We use a state-based [9] coupling technique at the NS<=>DSMC interface. That is, at the NS=>DSMC border, we maintain several reservoir cells that are used in the DSMC calculation. At each coupling, we remove all existing particles in the reservoir cells, and generate new ones in the reservoir cells (from a Maxwellian distribution) based on the NS cell centered properties. The purpose of the buffer cells is to improve the statistical representation of the generated particles. That is, when new particles are generated in the reservoir cells, there is a significant amount of statistical scatter, and the buffer cells (where no new particles are generated) help to reduce that noise. The DSMC calculation then proceeds in the DSMC region, and is influenced by the particles in the reservoir and buffer cells, which have the macroscopic properties from the NS calculation. At the DSMC=>NS

interface, we calculate the cell-centered macroscopic properties of the cells at the interface, and these become the boundary condition for the NS calculation. As discussed above, a sufficient amount of sampling must have occurred in these DSMC cells, to ensure that the statistical noise does not introduce numerical instability into the NS code. After the data are exchanged, the DSMC and NS codes run independently again for the specified amount of time in a cycle. The cycling process (exchange data and run independent calculations) is continued until the total number of cycles is completed.

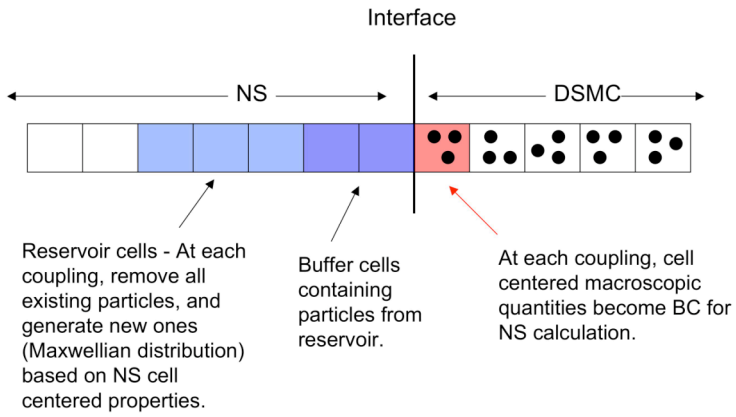

Figure 2. Coupling between the NS and DSMC codes at the interface.

RESULTS

Low Kn Validation Problem

As a validation test, we solve a low-Knudsen-number microchannel flow using the hybrid code, the NS code alone, and DSMC code alone. We choose a channel flow configuration so that we can compare the computations with a known analytical solution. Also, we choose a low-Kn flow, because the NS method is only valid for continuum flows. A schematic of the flow configuration is shown in Fig. 3. For the individual code validation tests, the entire domain was solved with the NS method or with the DSMC method, and for the hybrid validation test, the first half of the domain was solved with NS, while the second half was solved with DSMC, as indicated in Fig. 3.

For the NS test problem, and for the NS portion of the hybrid problem, the wall boundary has a no-slip boundary condition. For the DSMC test problem, and for the DSMC portion of the hybrid problem, the wall boundary is a fully-diffuse wall, with a momentum accommodation coefficient equal to one. For the DSMC calculations, subsonic boundary conditions [8] are used to set the pressure at the inflow and outflow boundaries.

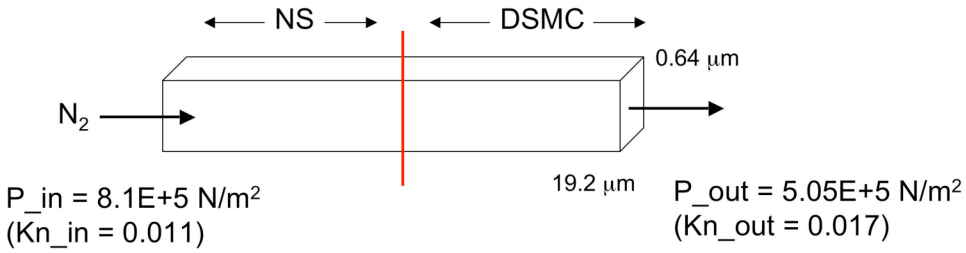

Figure 3. Schematic for validation test problem. The entire computational domain contains 3840 x 4 x 128 cells, where $\Delta x = \Delta y = \Delta z = 0.005$ μm. For the hybrid test problem, the left half of the domain is solved with NS, and the right half is solved with DSMC.

Figure 4 shows results calculated from the hybrid code. The contours of pressure and axial velocity show a smooth transition at the NS <=> DSMC interface. Figure 5 shows line profiles of pressure and axial velocity along the longitudinal centerline for the three codes, and for an analytical-NS solution [10]. The DSMC solution contains statistical scatter, as would be expected for this subsonic flow (outlet Mach number = 0.18), however, the results indicate that the solutions from the individual codes and the hybrid code match that predicted by the analytical NS-solution.

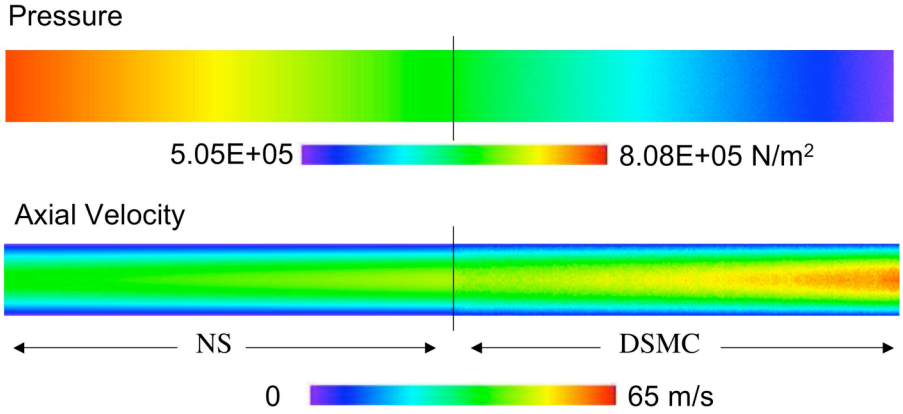

Figure 4. Pressure and axial velocity for the hybrid code.

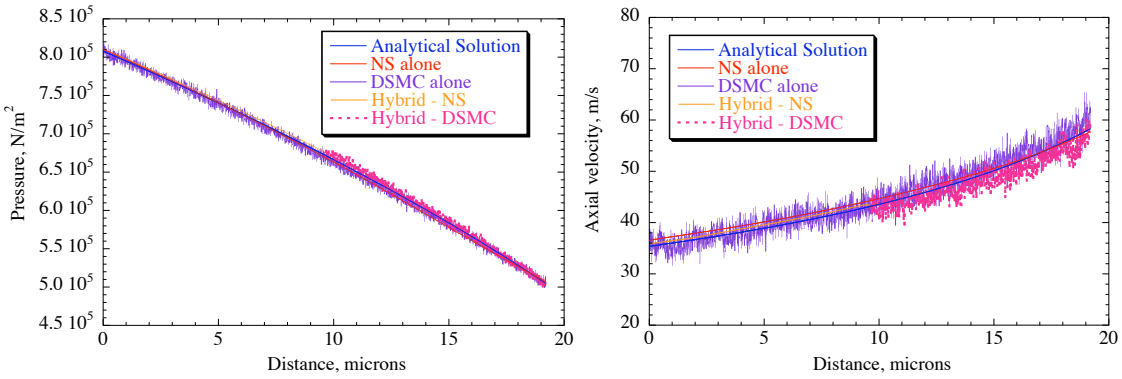

Figure 5. Pressure and axial velocity along the longitudinal centerline for the three codes and the analytical solution.

Microfluidic Filter

The second test problem simulates flow in a microfluidic filter, which contains a rarefied region surrounded by two continuum regions. as shown in Fig. 6.

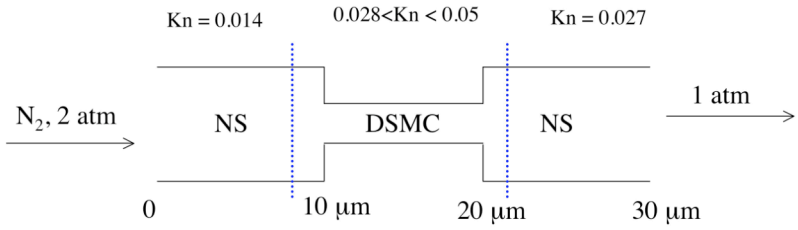

Figure 6. Schematic for microfluidic filter test problem.

Simulations were conducted using DSMC alone and the hybrid code, as shown in Figs. 7 and 8. When using the hybrid code, we used DSMC in the filter region, where the Knudsen number varies from 0.028 to 0.05, and NS at the inlet and outlet, where the Knudsen number is lower. Results from the two methods are qualitatively similar, with the hybrid method obviously having no statistical noise in the NS regions, and requiring 30% less CPU time.

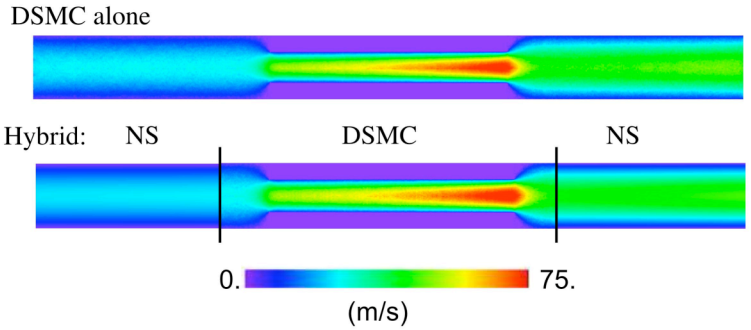

Figure 7. Axial velocity for microfluidic filter test problem.

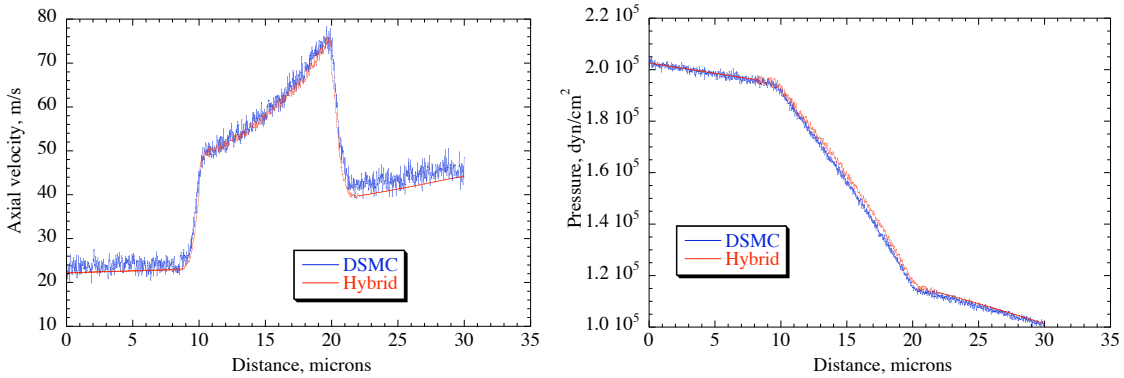

Figure 8. Axial velocity and pressure profiles along centerline, comparing hybrid and DSMC-alone solution.

DISCUSSION

To date, there have been several notable challenges in the development of this hybrid code. One involves the coupling between the NS and DSMC codes, and is applicable to any hybrid code (serial or parallel). Another challenge involves load balancing, and obviously, only applies for parallel hybrid codes.

Coupling and Statistical Noise

In standard DSMC methods, sampling (calculating macroscopic properties in each computational cell from the individual particles in that cell) usually begins after the flowfield reaches steady state. In the hybrid method, since the flowfield solution may change considerably after the NS and DSMC codes have been coupled, it may be necessary to clear the sample buffers and restart sampling after the two codes have already been coupled. Once the sample buffers have been cleared, there will be a considerable amount of statistical noise in the DSMC solution, which can create instability in the NS solution.

As discussed in the hybrid code methodology section, the user specifies how frequently data are exchanged at the NS<=>DSMC interface. For an unsteady flow, it would be necessary to exchange data at each timestep. In other hybrid codes using reservoir/buffer cells at the interface, the data exchange also occurs at each timestep, even for steady flows. In this work, we noted that exchanging data each timestep reduces the parallel efficiency of the hybrid code. That is, it is considerably more efficient to run the NS

and DSMC codes for many timesteps between data exchanges, rather than for just one timestep per data exchange.

At the DSMC=>NS interface, we calculate macroscopic properties in the DSMC cells, and these become boundary conditions for the NS calculation. If there is a lot of statistical noise in the DSMC data, this could easily create an instability in the NS solution. To avoid this problem, we do not begin coupling between NS and DSMC until the statistical noise in the DSMC solution is reduced significantly. If we need to clear the sample buffers in the DSMC solution after the coupling has begun, then we do not calculate new macroscopic properties until the statistical noise has been significantly reduced. We are also considering another way to reduce the statistical noise by incorporating a previously-developed nonlinear filter [11] into the DSMC solution.

Load Balancing

The magnitude of a timestep is determined by the CFL stability limit for a NS code, and by the mean collision time for a DSMC code. For the low Kn validation test, one NS timestep corresponds to approximately $1x10^{-12}$ s, while one DSMC timestep corresponds to approximately $2x10^{-11}$ s. However, because this was a continuum flow, one DSMC timestep took approximately four times as much CPU time as one NS timestep, on the same number of processors. Therefore, if a computational domain contains equal numbers of DSMC and NS cells, one must use four times as many DSMC processors as NS processors to achieve good load-balancing. This is true for Step 4, where independent initial NS and DSMC calculations are running, and in Step 6, where the NS and DSMC calculations are part of the cycling process. Computational efficiency of the hybrid code is destroyed if, for example, the NS processors have finished and are sitting idle while the DSMC processors are still computing.

Load balancing can be improved by appropriately adjusting the number of DSMC and NS processors, based on the number of computational cells for each method, and on the amount of CPU time needed for an NS and DSMC timestep. This hybrid code has been tested on three architectures: SGI Origin 3900, IBM p690, and Linux Networx Evolocity, and excellent scalability can be obtained as long as the user takes into account the factors discussed above.

REFERENCES

1. H.S. Wijesinghe and N.G. Hadjiconstantinou, *International Journal for Multiscale Computational Engineering* **2**, 189-204 2004 (and references within).
2. C.R. Kaplan, J. Liu, and E.S. Oran, "Parallel Hybrid Method for Subsonic Flows: Coupling and Load-Balancing Challenges," AIAA Paper 06-992, 2006.
3. J.P. Boris, and D.L. Book, *Journal of Computational Physics* **11,** 38-69 (1973).
4. J.P. Boris, A.M. Landsberg, E.S. Oran, J.H. Gardner, "LCPFCT - A Flux Corrected Transport Algorithm for Solving Generalized Continuity Equations," NRL Memorandum Report 6410-93-7192, Naval Research Laboratory, 1993.
5. E.S. Oran and J.P. Boris, *Numerical Simulation of Reactive Flow*, Cambridge: Cambridge University Press, 2001, pp. 247-254.
6. J. Liu, E.S. Oran, and C.R. Kaplan, *Journal of Computational Physics*, **208**, 416-434 (2005).
7. G. Bird, *Molecular Gas Dynamics and the Direct Simulation of Gas Flows*, Oxford:Clarendon Press, Oxford, 1994.
8. R.P. Nance, D.B. Hash, and H.A. Hassan, *Journal of Thermophysics and Heat Transfer* **12**, 447-449 (1998).
9. T.E. Schwartzentruber, L.C. Scalabrin, and I.D. Boyd, "Hybrid Particle-Continuum Simulations of Non-Equilibrium Hypersonic Blunt Body Flow Fields," AIAA Paper 06-3602, 2006.
10. E.B. Arkilic, "Measurement of the Mass Flow and Tangential Momentum Accommodation Coefficient in Silicon Micromachined Channels," PhD Thesis, Massachusetts Inst. of Technology, 1997.
11. C.R. Kaplan and E.S. Oran, *AIAA Journal* **40**, 82-90 (2002).