Improving the Efficiency of a Multiscale Method for Rarefied Flows

D.A. Kessler, E.S. Oran and C.R. Kaplan

Laboratory for Computational Physics and Fluid Dynamics, Naval Research Laboratory

Abstract. The coupled multiscale, multiphysics method, CM^3, is a novel method for simulating transition-regime gas flows. The basic idea of the method is to incorporate physics from the small-scale molecular motions into the continuum framework of the Navier-Stokes equations. CM^3 uses a Monte Carlo procedure to solve the Boltzmann equation at various instants in time and calculates the viscous stress tensor, $\boldsymbol{\tau}$, and heat flux vector, $\mathbf{q}$, from the molecular velocities. The computed $\boldsymbol{\tau}$ and $\mathbf{q}$ are then substituted into the Navier-Stokes equations and these equations are advanced forward in time using a finite-volume method. A difficulty in implementing multiscale methods of this type lies in initializing the velocity distribution functions used in the Monte Carlo solver at the beginning of each integration cycle. In this paper, we describe a method of particle velocity reinitialization that is significantly more efficient than using a standard velocity distribution function.

Keywords: multiscale methods, direct simulation Monte Carlo, transition-regime flows, Boltzmann
PACS: 47.11.St, 47.45.-n, 47.61.Fg

INTRODUCTION

Gas flows are generally classified by the level of rarefaction of the gas, which can be described by the Knudsen number, $Kn = \lambda/L$, where λ is the mean free path of the molecules and L is a characteristic length scale of the flow. When Kn is small, the flow is in the continuum regime, and the Navier-Stokes equations are commonly used. When Kn is large, the flow is said to be rarefied, and the constitutive laws that describe momentum and energy transport in the Navier-Stokes equations break down. The Boltzmann equation is commonly used instead to treat these high-Kn flows [1]. This equation can be solved efficiently when Kn is large, but it becomes more difficult and time-consuming to solve when Kn is small. Thus, accurate and efficient algorithms exist for computing large- and small-Kn flows, but for a range of Kn ($\mathcal{O}(10^{-2}) < Kn < \mathcal{O}(10)$), commonly referred to as the transition regime, the continuum equations are not sufficiently accurate and Boltzmann's equation is too time-consuming to use.

A variety of multiscale methods [2] that couple the Boltzmann and continuum approaches have been used to calculate transition-regime flows. One divides the spatial domain into continuum and non-continuum regions. In these spatial hybrid methods [3, 4, 5, 6, 7, 8], the Navier-Stokes equations are used to compute the flow in the continuum regions, and the Botzmann equation is solved using the direct simulation Monte Carlo (DSMC) method in the non-continuum regions. Coupling between the regions is accomplished by matching the continuum and molecular-level fluxes at the interfaces. Other domain decomposition techniques couple the two regions directly in the governing equations [9, 10]. A second approach is to use information about the molecular-level system to form a new set of macroscopic evolution equations. The gas-kinetic hydrodynamic method [11, 12], the coarse-grained acceleration method for the Boltzmann equation [13], and the micro-macro upscaling method [14] are all based on this principle.

A new multiscale method, the coupled multiscale multiphysics method (CM^3) [15], solves the continuum-level conservation laws using molecular-level fluxes calculated directly from a Monte Carlo simulation in the place of constitutive models for the energy and momentum fluxes. If these fluxes are updated frequently enough, the CM^3 significantly extends the range of applicability of the continuum-level conservation laws. It has been used successfully to calculate several one-dimensional transition-regime test flows with Kn up to unity [15]. The challenge of this method is that it becomes less computationally efficient as Kn increases. In this paper, we discuss why this is so and test methods for improving the efficiency of the CM^3, a brief overview of which is given in the following section.

PHYSICAL MODEL

We can compute moments of Boltzmann's equation by multiplying it by m_w, $m_w\mathbf{c}$, and $(1/2)m_w c^2$ and integrating over all velocity space to obtain the following set of evolution equations [1]:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}_0) = 0, \tag{1}$$

$$\frac{\partial}{\partial t}(\rho \mathbf{u}_0) + \nabla \cdot (\rho \mathbf{u}_0 \mathbf{u}_0) + \nabla \cdot (\rho \overline{\mathbf{c}'\mathbf{c}'}) = 0 \tag{2}$$

$$\frac{\partial}{\partial t}(\rho e) + \nabla \cdot (\mathbf{u}_0 \rho e) + \nabla \cdot (\rho \overline{\mathbf{c}'\mathbf{c}'} \cdot \mathbf{u}_0) + \nabla \cdot (\tfrac{1}{2}\rho \overline{\mathbf{c}' c'^2}) = 0. \tag{3}$$

In deriving this expression, the molecular velocity has been decomposed into the sum of a mean velocity $\mathbf{u}_0$ and fluctuating or thermal velocity $\mathbf{c}'$, and the continuum-level density, linear momenta, and energy are defined as $\rho = \overline{nm_w}$, $\rho \mathbf{u}_0 = \overline{nm_w \mathbf{c}}$, and $\rho e = \overline{nm_w c^2}/2$, respectively.

If we define the viscous stress tensor, heat flux vector, and scalar pressure as

$$\boldsymbol{\tau} = -(\rho \overline{\mathbf{c}'\mathbf{c}'} - P\mathbf{I}), \tag{4}$$

$$\mathbf{q} = \frac{1}{2}\rho \overline{c'^2 \mathbf{c}'}, \tag{5}$$

$$P = \frac{1}{3}\rho \overline{c'^2}, \tag{6}$$

respectively, Eqs. (1–3) are identical to the continuum conservation equations for mass, momenta, and energy [16].

The fundamental idea of the CM^3 is to compute directly the properties of the small-scale molecular motions and use this information to close Eqs. (1–3) instead of using constitutive laws for $\boldsymbol{\tau}$ and $\mathbf{q}$. Starting from a particular continuum-level flow field, we generate a compatible distribution of molecular velocities and then solve the Boltzmann equation for a short interval of time to calculate $\boldsymbol{\tau}$ and $\mathbf{q}$ using Eqs. (4–5). The computed $\boldsymbol{\tau}$ and $\mathbf{q}$ are then substituted into Eqs. (1–3), and these equations are advanced forward in time using a finite volume method. The values of $\boldsymbol{\tau}$ and $\mathbf{q}$ are periodically updated using Boltzmann's equation to ensure proper time evolution of the continuum-level flow field. A brief description of the CM^3 is given below. A more complete description can be found in [15].

COMPUTATIONAL ALGORITHM

Figure 1 shows a schematic diagram of the CM^3 algorithm. A continuum-level flow field is initialized on a regular, cartesian grid that will serve as the initial condition for the simulation. This initial field is passed into the CM^3 integration cycle, which is composed of four stages: reconstruction, Monte Carlo, compression, and continuum. The CM^3 integration cycle is repeated continuously to update the transport terms, $\boldsymbol{\tau}$ and $\mathbf{q}$, and ensure that the unsteady evolution of the flow is calculated correctly.

In the first stage of the cycle, reconstruction, a collection of particles is generated in each grid cell with the property that the average density, velocity, and temperature are equal to that of the continuum flowfield at that particular point in space. The particles are randomly placed throughout the cell with velocities chosen from a prescribed velocity distribution function. The choice of this distribution function is critical to the efficiency of the CM^3, and this will be discussed in detail in the Discussion section.

In the second stage, Monte Carlo, a modified version of the DSMC procedure is used to move and collide the particles in order to obtain a sequence of progressively more accurate estimates of the true velocity distribution $f_1, \ldots f_N$. During each iteration we take M time steps so that the Boltzmann equation is advanced forward in time $\delta t_{DSMC} = M\Delta t_{DSMC}$, where Δt_{DSMC} is the timestep used for the modified DSMC computations, taken to be less than the mean collision time of the gas molecules. We want to calculate the *instantaneous* values of $\boldsymbol{\tau}$ and $\mathbf{q}$ for a given flow field, so we choose δt_{DSMC} to be as small as possible so that the average velocity and temperature of each successive new distribution f_{i+1} at the end of an iteration are close to those of the original distribution f_i. To ensure that the average velocities and temperatures do not drift from the continuum-level values for each cell over N iterations, a velocity shifting and scaling procedure is used (see [15] for details). The process ensures $\mathbf{u}^* = \mathbf{u}_0$ and $T^* = T_0$, but

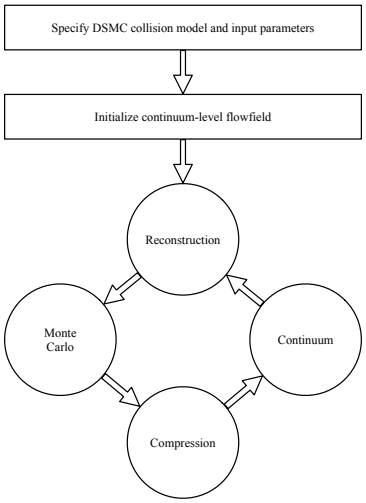

FIGURE 1. Schematic of the CM^3 algorithm.

allows the distribution to change so that new $\boldsymbol{\tau}^*$ and $\mathbf{q}^*$ evolve to their correct values. The Monte Carlo simulation is continued using this procedure until $\boldsymbol{\tau}^*$ and $\mathbf{q}^*$ no longer change.

The Monte Carlo procedure used to obtain f_N is a stochastic process, so statistical fluctuations are inherent in the solutions. These statistical fluctuations can introduce unacceptable levels of uncertainty into the computations of the moments of f_N that will result in discontinuous continuum-level fields of $\boldsymbol{\tau}$ and $\mathbf{q}$. In the third stage of the cycle, compression, we attempt to reduce this statistical noise. Generally speaking, any viable method of noise reduction can be used. For the one-dimensional flows considered here, we employ an ensemble-averaging approach in which a number of independent ensembles are simulated. We then calculate $\boldsymbol{\tau}$ and $\mathbf{q}$ in each cell using Eqs. (4–5). Cubic B-splines [17] are used to construct continuous fields of these quantities over the entire domain from the discontinuous averaged DSMC data. This procedure further reduces the statistical noise in the transport terms, and the resulting fields are smooth enough to be differentiated and used to solve the continuum-level equations.

In the final stage, continuum, a finite-volume method is used to advance Eqs. (1–3) for a period of time δt_{FV}, assuming $\boldsymbol{\tau}$ and $\mathbf{q}$ are constant. Each timestep, Δt_{FV}, is composed of two substeps. In the first, the divergence of $\boldsymbol{\tau}$, $\mathbf{q}$, and $\boldsymbol{\tau}\cdot\mathbf{u}_0$ are calculated using standard second order central finite-difference stencils and then used to partially advance the solutions forward in time. In the second, the convective terms of Eqs. (1–3) are calculated using the flux-corrected transport algorithm [18, 19], and the conserved variables are updated and sent to the beginning of a new CM^3 integration cycle.

DISCUSSION

Algorithmic Efficiency

Let N_1 be the total number of DSMC time steps taken during the Monte Carlo stage of one cycle ($N_1 = M * N_{MC}$), where N_{MC} is the number of Monte Carlo iterations performed each cycle, and let N_2 be the number of equivalent DSMC timesteps taken during the continuum stage of each cycle ($N_2 = \delta t_{FV}/\Delta t_{DSMC}$). Recall that the intent of the DSMC stage of the CM^3 cycle is to calculate correct values for $\boldsymbol{\tau}$ and $\mathbf{q}$ and not to advance the solution forward in time. Hence, we assume that time advancement of the bulk flow is done entirely in the continuum stage. The total number of CM^3 cycles that are performed is $Y = N_{tot}/N_2$, where N_{tot} is the total number of timesteps taken in the DSMC simulation. Finally, define E_{CM^3} as the number of ensembles performed for averaging purposes during the Monte Carlo stage of the CM^3 and E_{DSMC} as the number of ensembles over which averages are collected in the DSMC simulations.

If we assume that the wall clock time needed to advance Eqs. (1–3) forward in time Δt_{DSMC} in the continuum stage is much smaller than that required to do the same in the Monte Carlo stage, we can write the estimated ratio of CM^3

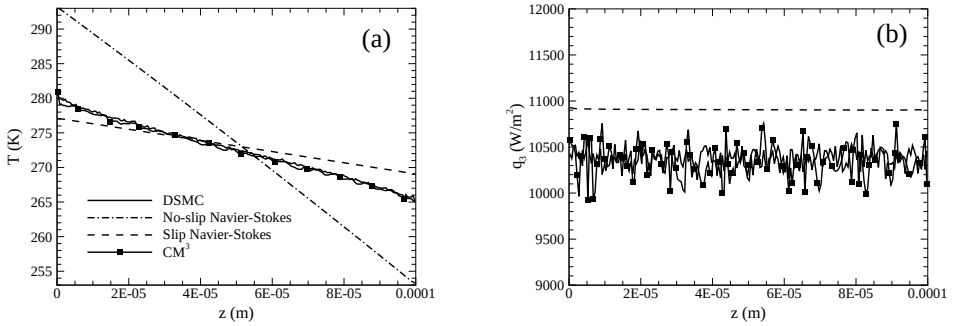

FIGURE 2. Steady state (a) temperature and (b) heat flux profiles for the Fourier flow with $Kn = 1$ computed using standard DSMC (solid), Navier-Stokes equations with (dash-dotted) and without (dashed) slip boundary conditions, and CM^3 with Maxwellian initialization (solid with symbols).

wall clock time to independent DSMC wall clock time for the same number of timesteps, N, as

$$E^* = \frac{Y(E_{\mathrm{CM}^3} N_1 T_{DSMC})}{E_{\mathrm{DSMC}}(N T_{DSMC})} = \frac{(N/N_2)(E_{\mathrm{CM}^3} N_1)}{E_{\mathrm{DSMC}} N} = \frac{E_{\mathrm{CM}^3} N_1}{E_{\mathrm{DSMC}} N_2}. \tag{7}$$

In [15], E^* varied from $\mathscr{O}(0.1)$ to $\mathscr{O}(1)$ for a series of one-dimensional Rayleigh flows with Kn ranging from 0.02 to 1. A series of Fourier flows with Kn in the same range were also computed for which E^* was found to vary from $\mathscr{O}(1)$ to $\mathscr{O}(10)$. The decrease in efficiency for both sets of simulations for larger Kn occurred because during each cycle f_N was quite different from the initial guess of a Maxwellian velocity distribution, and a greater number of iterations in the Monte Carlo stage of the algorithm were needed to reach this distribution. For the Fourier flows, more iterations were needed to ensure that the velocity distribution was sufficiently converged to compute the third-order moments for the heat flux.

From Eq. (7), we find that the efficiency of the CM^3 can be improved by (1) decreasing the number of independent ensembles used to reduce the statistical noise in $\boldsymbol{\tau}$ and $\mathbf{q}$, (2) increasing the length of time spent integrating Eqs. (1–3) each cycle, or (3) decreasing the number of iterations needed to converge to correct velocity distribution function. The use of the FCT time advancement procedure in the CM^3 naturally filters some of the noise from the molecular velocities used to compute $\boldsymbol{\tau}$ and $\mathbf{q}$ [20], and accordingly fewer ensembles are needed in the CM^3 to achieve statistical fluctuations in the final solutions with amplitudes similar to those calculated using DSMC. The test flows computed in [15] using the CM^3 required only 10% of the number of ensembles used in the DSMC simulations to reach similar noise levels. This reduction in computational cost is a consequence of the choice of integration method and is likely problem dependent. Improving upon this by changing the finite volume integrator is unlikely and will not be pursued here. Likewise, any method used to reduce the statistical noise in the Monte Carlo stage of the CM^3 could be used in the same manner in a standard DSMC simulation. Hence, equal gains in wall clock time would be achieved for each simulation. Hence, item (1) is not a viable option for achieving gains in computational efficiency. The second possibility of increasing efficiency (decreasing E^*) is to increase the length of time spent integrating Eqs. (1–3). However, to achieve maximum accuracy in the transient continuum-level solution, we want δt_{FV} (or, equivalently, N_2) to be as small as possible. Upper bounds on the size of δt_{FV} are set by the physics of the flow and not by the details of the solution procedure. Hence, using option (2) is not likely to produce meaningful gains in efficiency without causing a corresponding decrease in the accuracy of the solution. Option (3), decreasing the number of iterations needed to achieve a converged velocity distribution function during the Monte Carlo stage of the CM^3, is the most viable option for decreasing E^*, and the remainder of this paper will be focused on this issue.

Convergence of Velocity Distribution Functions

We examine the convergence of the molecular velocity distribution using one of the same tests flows used in [15], the Fourier flow with $Kn = 1$. Consider two parallel plates, one located at $z = 0$ and the other at $z = 1.0 \times 10^{-4}$ m, that are infinite in extent in the x- and y-directions. The temperature of the top plate is 253 K and that of the bottom

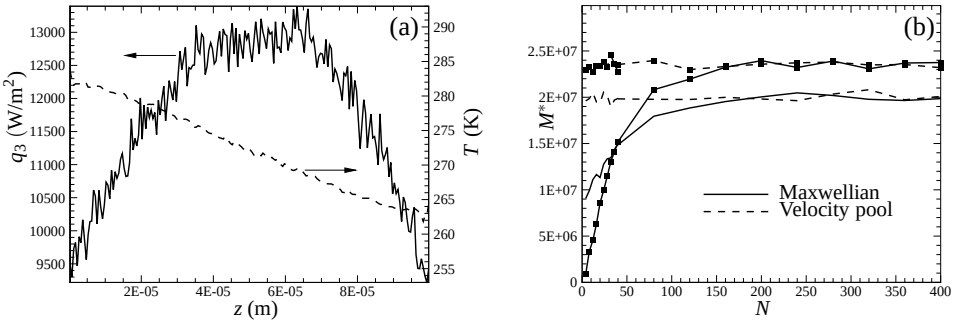

FIGURE 3. (a) Initial temperature (dashed line) and heat flux (solid line) profiles used as the initial condition for a single CM^3 cycle and (b) the resulting convergence behavior of a third-order moment of f as a function of the number of Monte Carlo iterations initialized from a Maxwellian distribution (solid lines) and from a velocity pool generated during the previous cycle (dashed lines) at two different locations in the flow: near the wall (no symbols) and near the center of the gap (symbols).

plate is 293 K. The average temperature and pressure of gas are 273 K and 1.542×10^{-3} atm, respectively, so that $\lambda = 1.0 \times 10^{-4}$ m and $Kn = 1$. We wish to compute the one-dimensional continuum-level temperature profile between the plates. The steady-state temperature and heat flux solutions calculated using DSMC, CM^3, and the Navier-Stokes equations with slip boundary conditions are shown in Fig. 2. The DSMC solution differs noticeably from the Navier-Stokes solution, which suggests that the constitutive laws for $\boldsymbol{\tau}$ and $\mathbf{q}$ do not correctly describe the molecular-level momentum and energy fluxes at this level of rarefaction. Since, the formulation of $\boldsymbol{\tau}$ and $\mathbf{q}$ are consistent with a Chapman-Enskog velocity distribution, we can assume the correct velocity distribution differs significantly from this function and that this flow can be characterized as being in the transition regime.

In the CM^3 calculation, each Monte Carlo cycle was initiated from a Maxwellian velocity distribution funcion, and the resulting collection of molecular velocities was matured for 400 iterations (equivalent to 2000 total Monte Carlo time steps each cycle). Due to the rapid time evolution of the temperature field, $\boldsymbol{\tau}$ and $\mathbf{q}$ were updated frequently, every $10\Delta t_{DSMC}$. As mentioned above, the ratio of the number of ensembles used for averaging molecular properties in the CM^3 to that in a DSMC simulation is $E_{\mathrm{CM}^3}/E_{DSMC} = 1/10$, and, hence, $E^* = 20$ for this particular case.

Now we consider a new method of initializing the particle velocities that does not require the use of an analytical distribution function. Instead, we select the initial velocities for each cycle from a pool of velocities gathered at the end of the previous cycle. The velocity distribution function resulting from this initialization process differs only slightly from the true velocity distribution since the new flow conditions in each cell have also changed only slightly during the continuum portion of the cycle. The velocity pool collection procedure does not add to the overall computation time of the Monte Carlo portion of the cycle since these velocities have already been computed. A total of $N_P^* E_{\mathrm{CM}^3}$ total velocity samples are collected, where N_P is the average number of particles in each cell over the E_{CM^3} ensembles.

To study the convergence of f to f_N, we track the evolution of velocity distributions initialized using a Maxwellian function and using the velocity pool approach during the Monte Carlo stage of a single CM^3 cycle. A particular solution in the unsteady evolution of the $Kn = 1$ Fourier flow, shown in Fig. 3(a), is used as an initial condition. For Maxwellian initialization, we use the standard approach (see, for instance, [1]), and for the velocity pool approach we initialize the particle velocities in the manner described above using the collection of velocities gathered over the E_{CM^3} ensembles at the end of the previous cycle as the sample pool. Figure 3(b) shows the convergence behavior of a third moment of f, $M^* = \int_{-\infty}^{\infty} c'^2 w'_3 f \, \mathrm{d}\mathbf{c}$, which is used to compute the z-component of $\mathbf{q}$, at two different locations in the flow: near the bottom wall and near the center of the gap, as a function of the number of Monte Carlo iterations. Here, $c' = \sqrt{u'^2 + v'^2 + w'^2}$ is the magnitude of the thermal velocity vector with x-, y-, and z-components, u', v', and w'.

For a Maxwellian velocity distribution, M^* is identically zero, and we see that it takes a finite number of Monte Carlo iterations for M^* to approach a converged value. The convergence time depends somewhat on the local flow conditions, and varies from about 250 to 300 iterations for the examples shown in Fig. 3(b). When using the velocity pool method, for which the velocity distribution function of the stored velocities is already close to the true distribution, the convergence time is significantly faster. In fact, it is difficult to ascertain exactly how rapidly the velocity distribution converges since $M_N^* - M_0^*$ is likely smaller than the statistical noise in the computations. Regardless, M^* appears to be converged at least within 40 iterations. If the flow shown in Fig. 2 were to be recomputed with CM^3 using the velocity

pool method with $N = 40$ to initialize the particle velocities at the beginning of each cycle, E^* would equal 2, a vast improvement over the original value.

CONCLUSIONS

We have introduced a new initialization procedure for the Monte Carlo stage of a CM^3 cycle that significantly reduces the number of iterations to achieve converged values for $\boldsymbol{\tau}$ and $\mathbf{q}$. At the beginning of each cycle, we initialize particle velocities by randomly choosing them from a large ensemble of particle velocities collected at the end of the previous cycle. The distribution of velocities in this ensemble is then already very close to that of the new ensemble we wish to calculate. This simple procedure does not add to the computation time for each cycle, since these velocities are already being computed. Rather, the decrease in the number of iterations needed for $\boldsymbol{\tau}$ and $\mathbf{q}$ to reach converged values allows for a significant increase in computational efficiency. For the test flow considered in this paper, the total computation time for the CM^3 method was on the same order as standard DSMC simulations; however, for some of the other test flows considered in [15], CM^3 would be considerably more efficient than standard DSMC.

There are other potential modifications to CM^3 that could also help to increase its efficiency. For instance, adaptively changing δt_{FV} as a function of time based on the local rate of change of the macroscopic variables could decrease the number of updates to $\boldsymbol{\tau}$ and $\mathbf{q}$ when the flow is not rapidly varying. A similar adaptive procedure could be used to check for convergence of $\boldsymbol{\tau}$ and $\mathbf{q}$ to minimize the number of iterations that are performed each cycle. Another potential benefit in using CM^3 is that it can be easily integrated into a spatial hybrid method where the Navier-Stokes equations are solved in one portion of the domain and Boltzmann's equation in another. Generally, DSMC has been used to solve the Boltzmann equation in these types of simulations, but CM^3 has an inherent advantage in that the continuum variables are readily available and accessible in this method. Using these variables directly would avoid the use of complicated coupling methods between DSMC and Navier-Stokes regions that are currently being used.

ACKNOWLEDGMENTS

This work was supported by the Naval Research Laboratory through the Office of Naval Research. Computational resources were provided by the Department of Defense High Performance Computing Modernization Program.

REFERENCES

1. G. Bird, *Molecular Gas Dynamics and the Direct Simulation of Gas Flows*, Oxford Science Publications, Oxford, U.K., 1994.
2. W. E, and B. Engquist, *Commun. Math. Sci.* **1**, 87–132 (2003).
3. T. E. Schwartzentruber, and I. D. Boyd, *J. Comp. Phys.* **215**, 402–416 (2006).
4. Y.-Y. Lian, J.-S. Wu, G. Cheng, and R. Koomullil, Development of a parallel hybrid method for the DSMC and Navier-Stokes solver, Paper 2005-0435, AIAA (2005).
5. R. Roveda, D. B. Goldstein, and P. L. Varghese, *J. Spacecraft and Rockets* **35**, 258–265 (2000).
6. D. C. Wadsworth, and D. A. Erwin, Two-dimensional hybrid continuum/particle approach for rarefied flows, Paper 92-2975, AIAA (1992).
7. A. L. Garcia, W. Y. C. J. B. Bell, and B. J. Alder, *J. Comp. Phys.* **154**, 134–155 (1999).
8. H. Wijesinghe, R. Hornung, A. Garcia, and N. Hadjiconstantinou, *ASME J. Fluids Eng.* **126**, 768–777 (2004).
9. P. Degond, and S. Jin, *SIAM J. Numer. Anal.* **42**, 2671–2687 (2005).
10. P. Degond, S. Jin, and L. Mieussens, *J. Comp. Phys.* **209**, 665–694 (2005).
11. K. H. Prendergast, and K. Xu, *J. Comp. Phys.* **109**, 53–66 (1993).
12. K. Xu, and K. Prendergast, *J. Comp. Phys.* **114**, 9–17 (1994).
13. H. A. Al-Mohssen, N. G. Hadjiconstantinou, and I. G. Kevrekidis, *ASME J. Fluid Eng.* **129**, 908–912 (2007).
14. P. Degond, J.-G. Liu, and L. Mieussens, *Multiscale Model. Simul.* **5**, 940–979 (2006).
15. D. Kessler, E. Oran, and C. Kaplan, *J. Fluid Mech.* (to appear 2010).
16. R. Panton, *Incompressible Flow*, Wiley-Interscience, New York, 1996.
17. P. Dierckx, *Curve and surface fitting with splines, Monographs on numerical analysis*, Oxford University Press, New York, 1993.
18. J. Boris, A. Landsberg, E. Oran, and J. Gardner, LCPFCT-a flux-corrected transport algorithm for solving generalized continuity equations, Memorandum Report 6410-93-7192, U.S. Naval Research Laboratory, Washington, DC (1993).
19. E. S. Oran, and J. P. Boris, *Numerical Simulation of Reactive flows*, Elsevier, New York, 2001.
20. C. Kaplan, and E. Oran, *AIAA Journal* **40**, 82–90 (2002).