JOURNAL OF COMPUTATIONAL PHYSICS **139,** 410–443 (1998)
ARTICLE NO. CP975878

Two-Dimensional PDF/SPH Simulations of Compressible Turbulent Flows

Walter C. Welton

Department of Mechanical and Aerospace Engineering, Cornell University, Ithaca, New York 14853
E-mail: welton@mae.cornell.edu

Received June 24, 1997; revised October 20, 1997

This paper describes recent progress made in the development and implementation of a two-dimensional Monte Carlo/PDF/SPH particle method. The approach, which is applicable to compressible turbulent flows, incorporates elements of smoothed particle hydrodynamics to extract mean quantities from the particles, including the mean pressure gradient. A new and efficient algorithm based on Fourier-series expansions of the SPH kernel is used to compute these means; for a simulation with N particles the computational work scales purely as $\mathcal{O}(N)$. A thorough study of numerical errors introduced by the finite series expansion is also performed, and results are presented which show that these errors scale as expected and can be made negligibly small using modest computing requirements. The particle method has been used to simulate a variety of 2D flows, including a stationary turbulent plane wake, flows under the influence of simple body forces, and an unsteady flow featuring compression/expansion waves and a pair of decaying vortices. The plane wake calculation includes comparisons with self-similar experimental data and good agreement is obtained for the mean velocity and Reynolds stress profiles. © 1998 Academic Press

Key Words: PDF methods; Monte Carlo methods; smoothed particle hydrodynamics; turbulent flow; compressible flow.

1. INTRODUCTION

The probability density function (PDF) approach has become a useful computational tool for predicting the properties of turbulent reacting flows [1]. For this class of flows the PDF method has unique advantages over traditional moment-closure methods. In particular, modeling of turbulent transport is unnecessary, and finite-rate nonlinear chemistry can be treated both exactly and naturally [2, 3].

The majority of work using PDF methods has focused on incompressible flows, although the method is known to be applicable to compressible flows as well [4, 5] ("compressible" implying that variations in pressure induce corresponding variations in density). One possible reason for this is the lack of a general and robust treatment of pressure in the PDF framework. While numerous approaches have been successfully used [6–9], they are often limited in scope or introduce additional complexities into the problem.

Progress has recently been made in addressing this issue with the development and testing of a new PDF formulation applicable to quasi-1D compressible turbulent flows [10]. In this approach the treatment of pressure is quite general, and is based on kernel estimation techniques derived from smoothed particle hydrodynamics (SPH). The current work extends this new approach to 2D and 3D flows.

The decision to use SPH within the PDF formulation is motivated by three reasons. The first is compatibility of the two methods. The favored approach for implementing PDF methods is via Monte Carlo particle-mesh simulation. Monte Carlo simulation is used primarily because it is an efficient solution method for problems involving large dimensionality. In a Monte Carlo/PDF flow simulation, the fluid is represented using a large sample of stochastic particles which mimic the physical behavior of fluid particles, and properties for these particles are advanced in time according to specified evolution equations. Simulations using SPH are performed using the same approach [11] (although without a mesh), and thus the two methods are similar in their implementation.

A second reason for using SPH is that it allows the pressure to be obtained directly from the particles—apart from an equation of state, no additional assumptions are needed. In many applications of PDF methods, the approach used to obtain the pressure has been to couple the Monte Carlo/PDF code to a finite-volume solver which returns the mean pressure field [6, 12–14]. Using SPH this external dependence is removed to give a completely self-contained Monte Carlo/PDF code.

A final reason for using SPH is that it is a grid-free approach. All required quantities, including gradients, are calculated without reference to a fixed spatial grid. Complicated flow patterns are therefore handled easily and naturally by the method. The 2D results presented in this paper are the first ever obtained using a grid-free Monte Carlo/PDF method.

The main challenge to using SPH within the Monte Carlo/PDF framework has been to reduce its computational cost. SPH is a locally N-body method, meaning that each of the N particles in a given simulation interacts directly with the subset of particles in its locally defined neighborhood. In all known applications of SPH, the size of this neighborhood (termed the *smoothing length*) is purposely chosen so that the number of neighbors for each particle remains relatively constant, typically on the order of a few dozen or less. Given this constraint, it is then easy to code an algorithm whose computational work scales like $\mathcal{O}(N)$.

In the Monte Carlo/PDF framework, however, this constraint is unacceptable due to the stochastic nature of the problem. Since statistical error in the SPH kernel estimates depends directly on the number of particles $\overline{N}$ which contribute to the estimates, convergence of the method requires that $\overline{N}$ approach infinity *independent of the smoothing length*. The trivial implementation of SPH in this situation leads to an $\mathcal{O}(N\overline{N})$ algorithm, which for large $\overline{N}$ (hence, large N) is clearly inefficient. The challenge therefore is to seek and develop an algorithm which implements SPH in this context, but whose computational work scales purely as $\mathcal{O}(N)$ (independent of $\overline{N}$). The previous work [10] described such an algorithm for 1D problems, but it was not readily extendible to higher dimensions. A few $\mathcal{O}(N \log N)$ tree-based algorithms that work in higher dimensions exist [15, 16], but these typically have additional overhead associated with them for maintaining and updating the tree data structure. The new algorithm described in this paper overcomes all these limitations in that it scales purely as $\mathcal{O}(N)$, has relatively low overhead, and is easily implemented in any number of dimensions.

In summary so far, this paper makes three significant contributions to this field of research. First, it extends the coupled PDF/SPH approach with its general treatment of mean pressure to 2D flows. Second, it presents the first 2D results ever obtained using a grid-free Monte Carlo/PDF method. And third, it presents a new algorithm which allows multidimensional SPH kernel estimates to be accurately obtained in $\mathcal{O}(N)$ time independent of the smoothing length.

The paper begins with a brief section presenting relevant background information on PDF methods and SPH. The 2D particle evolution equations are also given. In Sections 3 through 5, the paper moves on to present the new 2D algorithm, to analyze numerical errors that it introduces, and to compare its computational cost relative to other methods. Following a review of the numerical implementation of the Monte Carlo/PDF method, Section 7 presents a number of sample 2D results generated using the new method. The paper concludes after a discussion in Section 8 of some variations and extensions to the algorithm.

2. COMBINED PDF/SPH METHOD

A unique feature of the particle method described in this paper is its combination of SPH with a PDF-based Monte Carlo method. Until recently the two techniques have not been used together, quite possibly solely because of computational limitations. The purpose of this section is to provide a brief description of each technique and show how the two methods complement each other.

2.1. PDF Method

In simulations of compressible turbulent flows without combustion, the relevant pdf is the Lagrangian *mass density function* (mdf) of velocity and position, denoted by $\mathsf{F}(\mathbf{V}, \mathbf{x}; t)$ [2]. Two important properties of F are

$$\int \mathsf{F}(\mathbf{V}, \mathbf{x}; t)\, d\mathbf{V} = \langle \rho(\mathbf{x}, t) \rangle \tag{1}$$

and

$$\int Q(\mathbf{V}, \mathbf{x}, t) \mathsf{F}(\mathbf{V}, \mathbf{x}; t)\, d\mathbf{V} = \langle \rho Q \rangle = \langle \rho \rangle \tilde{Q}, \tag{2}$$

where $\langle \rho \rangle$ is the mean fluid density, Q is a random variable (itself a function of velocity, position, and time), and $\tilde{Q}$ is by definition the Favre average of Q. Both integrals are over the velocity sample space.

Using the Monte Carlo approach, F is represented by an ensemble of N stochastic particles which are continuously distributed in the domain and which model the behavior of fluid particles. Each particle has properties of velocity $\mathbf{U}^{*(n)}$ and position $\mathbf{x}^{*(n)}$ (asterisks are used to indicate these properties are modeled), and from them the discrete Lagrangian mdf is defined to be

$$\mathsf{F}_N(\mathbf{V}, \mathbf{x}; t) = \Delta m \sum_{n=1}^{N} \delta\big(\mathbf{V} - \mathbf{U}^{*(n)}\big)\, \delta\big(\mathbf{x} - \mathbf{x}^{*(n)}\big), \tag{3}$$

where Δm is the mass of each particle ($\Delta m = \mathcal{M}/N$ for simulation of a fluid of constant mass $\mathcal{M}$). The relationship between F_N and F is [2]

$$\langle \mathsf{F}_N(\mathbf{V}, \mathbf{x}; t) \rangle = \mathsf{F}(\mathbf{V}, \mathbf{x}; t), \quad \text{all } N \geq 1. \tag{4}$$

Some insight into this relationship can be gained by considering the density. Temporarily replacing F with F_N in the integral of Eq. (1) gives, after simplifying,

$$\rho_N(\mathbf{x}, t) = \Delta m \sum_{n=1}^{N} \delta\left(\mathbf{x} - \mathbf{x}^{*(n)}\right), \tag{5}$$

usually referred to as the *fine-grained* density. Equation (4) subsequently implies

$$\langle \rho(\mathbf{x}, t) \rangle = \langle \rho_N(\mathbf{x}, t) \rangle. \tag{6}$$

In a numerical implementation, the right side of this expression can be estimated through ensemble averages.

In the 2D implementation the position $\mathbf{x}^*$ of a particle consists of two coordinates $\{x_1^*, x_2^*\}$ and the velocity $\mathbf{U}^*$ has three components $\{U_1^*, U_2^*, u_3^*\}$ (where a lower case velocity denotes a fluctuation with respect to the Favre average). These properties evolve according to the modeled system of stochastic equations:

$$dx_i^* = U_i^* \, dt, \qquad i = 1, 2 \tag{7}$$

$$dU_i^* = -\frac{1}{\langle \rho \rangle} \frac{\partial \langle P \rangle}{\partial x_i} \, dt - \beta\omega(U_i^* - \tilde{U}_i) \, dt + (C_0 \omega \tilde{k})^{1/2} \, dW_i(t), \quad i = 1, 2 \tag{8}$$

$$du_3^* = -\beta\omega u_3^* \, dt + (C_0 \omega \tilde{k})^{1/2} \, dW_3(t). \tag{9}$$

The velocity evolution equations correspond to the simplified Langevin model, a stochastic model for inhomogeneous, incompressible turbulence [3]. Appearing in the equations are various coefficients (determined from the particles) and parameters. Coefficients include the mean pressure $\langle P \rangle$, the Eulerian Favre-averaged velocities $\tilde{U}_i$, and the turbulent kinetic energy $\tilde{k}$. These are evaluated at the particle position (x_1^*, x_2^*). Depending on the type of problem being solved, the turbulent frequency ω (defined as $\langle \epsilon \rangle / \tilde{k}$, where $\langle \epsilon \rangle$ is the mean dissipation) may also be a coefficient. Parameters specified by the user include a universal constant C_0 (the value $C_0 = 2.1$ is used here), and a drift constant β (defined to be $\frac{1}{2} + \frac{3}{4} C_0$). The final input is $\mathbf{W}(t)$, an isotropic vector Wiener process that is simulated for each particle at each time step. Details on the simplified Langevin model can be found in Refs. [3, 17, 18].

Taken together the modeled particle evolution equations imply an equivalent modeled transport equation for the Lagrangian mdf F^* [2], the derivation of which yields

$$\frac{\partial \mathsf{F}^*}{\partial t} + \sum_{j=1}^{2} \frac{\partial}{\partial x_j}[V_j \mathsf{F}^*] = \sum_{j=1}^{2} \frac{\partial}{\partial V_j} \left[\left\{ \frac{1}{\langle \rho \rangle} \frac{\partial \langle P \rangle}{\partial x_j} + \beta\omega\left(V_j - \tilde{U}_j\right) \right\} \mathsf{F}^* \right] + \frac{\partial}{\partial V_3}[\beta\omega V_3 \mathsf{F}^*] + \frac{C_0 \omega \tilde{k}}{2} \frac{\partial^2 \mathsf{F}^*}{\partial V_j \partial V_j}. \tag{10}$$

This equation may subsequently be used to obtain all evolution equations for moments of velocity (such as Reynolds stresses) corresponding to the original system of particle evolution equations. Note that this equation is the two-dimensional extension of the one given in [10], but without the area-effect source terms.

2.2. SPH Method

Solution of the particle evolution equations (7)–(9) requires evaluation of various coefficients, including $\langle\rho\rangle$, $\nabla\langle P\rangle$, $\tilde{U}_i$, and $\tilde{k}$, and smoothed particle hydrodynamics offers a means to accomplish this. Using the grid-free kernel estimation techniques of SPH, these coefficients are evaluated directly from the particles.

For any field $Q(\mathbf{r})$ (random or deterministic), a "smoothed" kernel estimate to $\langle Q\rangle$ at the point $\mathbf{x}$ is given by the integral relation

$$\langle Q(\mathbf{x})\rangle_h = \int_{-\infty}^{\infty} Q(\mathbf{r})K(\mathbf{x}-\mathbf{r}, h)\,d\mathbf{r}, \tag{11}$$

where $K(\mathbf{r}, h)$ is a user-specified interpolating kernel, and the *smoothing length h* is a measure of the kernel width. The subscript h is added to the angle brackets to distinguish the quantity from the true expectation and to indicate that it is obtained by using a smoothing length of magnitude h.

Assuming the field Q is represented by a distribution of N particles with positions $\{\mathbf{x}^{*(n)}\}$ and properties $\{Q^{(n)}\}$, the integral in Eq. (11) can be estimated using the discrete form [11]

$$\langle Q(\mathbf{x})\rangle_{N,h} = \Delta m \sum_{n=1}^{N} \frac{Q^{(n)}}{\langle\rho\rangle^{(n)}} K\left(\mathbf{x}-\mathbf{x}^{*(n)}, h\right), \tag{12}$$

with Δm the mass of each particle. Note that smaller values of h give more spatially accurate estimates to $\langle Q(\mathbf{x})\rangle$, but result in fewer particles giving significant contributions, and hence, more statistical error and bias. (As a side note, for the case of mean density, Eq. (12) gives

$$\langle\rho(\mathbf{x})\rangle_{N,h} = \Delta m \sum_{n=1}^{N} K\left(\mathbf{x}-\mathbf{x}^{*(n)}, h\right), \tag{13}$$

which is precisely the expression obtained if $Q(\mathbf{r})$ is replaced by the fine-grained density $\rho_N(\mathbf{r})$ in Eq. (11).

The smoothing kernel $K(\mathbf{r}, h)$ must satisfy the two properties

$$\int_{-\infty}^{\infty} K(\mathbf{r}, h)d\mathbf{r} = 1 \tag{14}$$

and

$$\lim_{h\to 0} K(\mathbf{r}, h) = \delta(\mathbf{r}), \tag{15}$$

so that $\lim_{h\to 0}\langle Q(\mathbf{x})\rangle_h = \langle Q(\mathbf{x})\rangle$ [19]. For the purpose of this study, the 2D kernel is also assumed to be a tensor-product kernel; it can be written in the form

$$K(\mathbf{x}, \mathbf{h}) = \hat{K}(x_1, h_1)\,\hat{K}(x_2, h_2), \tag{16}$$

with h_1 and h_2 the smoothing lengths in each coordinate direction. Furthermore, the 1D kernel $\hat{K}$ has compact support in that $\hat{K}(r, h) = 0$ for $|r| > h$. This latter property is required for the 2D algorithm. For future reference, the functional form of $\hat{K}$ used here is [20]

$$\hat{K}(r, h) = \begin{cases} \frac{5}{4h}\,(1 + 3|r|/h)(1 - |r|/h)^3 & \text{for } |r| \le h \\ 0 & \text{for } |r| > h. \end{cases} \tag{17}$$

Note that $\hat{K}$ is symmetric about $r = 0$, and piecewise-quartic with a continuous second derivative.

3. $\mathcal{O}(N)$ FOURIER-SERIES ALGORITHM

This section describes the new $\mathcal{O}(N)$ algorithm for computing multidimensional SPH kernel estimates. The availability of such an algorithm is crucial to the success of the combined PDF/SPH approach. As stated previously, the average number of particles $\overline{N}$ contributing to each kernel estimate must be made large in order to reduce the statistical error introduced by the Monte Carlo method. In the 2D simulations performed here, values for $\overline{N}$ on the order of a thousand are necessary to keep statistical fluctuations in mean flow quantities reasonably small. Clearly then, for a simulation with N particles, where N is large, an $\mathcal{O}(N\overline{N})$ direct sum implementation for computing kernel estimates is prohibitive; the only feasible way to compute these quantities is via an $\mathcal{O}(N)$ algorithm.

The new algorithm is based on series expansions of the kernel. For the purpose of describing the algorithm, consider the example of calculating mean densities via kernel estimates at all particle locations in a two-dimensional region R,

$$\langle \rho(\mathbf{x}^{*(i)}) \rangle_{N,\mathbf{h}} = \sum_{n=1}^{N} m^{(n)} \hat{K}\left(x_1^{*(i)} - x_1^{*(n)}, h_1\right) \hat{K}\left(x_2^{*(i)} - x_2^{*(n)}, h_2\right) \quad \{i : \mathbf{x}^{*(i)} \in R\}, \tag{18}$$

where $\mathbf{h}$ is written for $\{h_1, h_2\}$, and the tensor-product property of the 2D kernel, Eq. (16), has been assumed. For generality each particle is allowed to have a unique mass $m^{(n)}$. The region R should be thought of as a subregion of the computational domain D; it is hereafter assumed to be rectangular with dimensions w_1 and w_2, centered at $\mathbf{x}^R$, and aligned with the coordinate axes so that

$$R = \left[x_1^R - \frac{w_1}{2}, x_1^R + \frac{w_1}{2}\right] \times \left[x_2^R - \frac{w_2}{2}, x_2^R + \frac{w_2}{2}\right] \tag{19}$$

as in Fig. 1.

Because the kernels $\hat{K}$ are compact, the mean density estimate at each position $\mathbf{x}^{*(i)}$ consists of contributions from a local subset of the particles. More specifically, contributions to any kernel estimate in R will come only from particles in region S of Fig. 1,

$$S = \left[x_1^R - \frac{w_1}{2} - h_1, x_1^R + \frac{w_1}{2} + h_1\right] \times \left[x_2^R - \frac{w_2}{2} - h_2, x_2^R + \frac{w_2}{2} + h_2\right], \tag{20}$$

so that the summation in Eq. (18) can be changed to give

$$\langle \rho(\mathbf{x}^{*(i)}) \rangle_{N,\mathbf{h}} = \sum_{\mathbf{x}^{*(n)} \in S} m^{(n)} \hat{K}\left(x_1^{*(i)} - x_1^{*(n)}, h_1\right) \hat{K}\left(x_2^{*(i)} - x_2^{*(n)}, h_2\right) \quad \{i : \mathbf{x}^{*(i)} \in R\}. \tag{21}$$

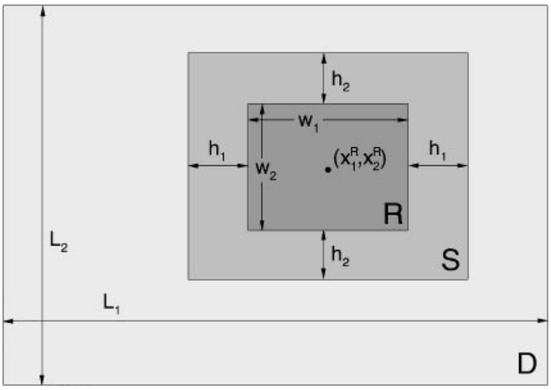


FIG. 1. Computational domain *D*, including a region of interest *R* and the corresponding region of support *S* for any kernel estimate in *R*.

With the summation restricted to particles in S, the range of arguments passed to the 1D kernels is now clearly bounded. For example, the maximum value of $|x_1^{*(i)} - x_1^{*(n)}|$ is $h_1 + w_1$ which occurs if $x_1^{*(i)}$ is on the left or right boundary of R and $x_1^{*(n)}$ is on the opposite boundary of S. Given this bound, the kernel may be replaced by a series expansion valid in the interval $r \in [-h_1 - w_1, h_1 + w_1]$, for example,

$$\hat{K}(r, h_1) = \sum_{p=0}^{\infty} \hat{a}_p \phi_p(r), \tag{22}$$

where $\phi(r) = \{\phi_p(r)\}_{p=0}^{\infty}$ is any set of basis functions that is orthogonal over the interval $[-h_1 - w_1, h_1 + w_1]$. Existence of the $\mathcal{O}(N)$ algorithm subsequently depends on the choice of $\phi(r)$. In particular, $\phi(r)$ must have the separation property

$$\sum_n \phi_p(r - r_n) = F\left(\phi(r), \sum_n \phi(r_n)\right), \quad \text{each } p, \tag{23}$$

as will become clear. Examples of function sets satisfying this property are the trigonometric basis functions, and polynomial sets such as Legendre or Chebyshev polynomials.

For the remainder of the paper $\phi(r)$ is taken to be the set of trigonometric basis functions. With this choice, Eq. (22) gives the Fourier-series expansion for $\hat{K}^P(r, h_1, \ell_1)$, the periodic extension of $\hat{K}(r, h_1)$ with period $\ell_1 = 2(h_1 + w_1)$. Because the kernels are symmetric about $r = 0$, the expansion consists only of cosine terms. Furthermore, the periodicity of $\hat{K}^P$ allows ℓ_1 to be reduced to $2h_1 + w_1$ without consequence, as sketched in Fig. 2. Using the same process, $\hat{K}(r, h_2)$ is replaced with $\hat{K}^P(r, h_2, \ell_2)$, where $\ell_2 = 2h_2 + w_2$. With these substitutions, the resulting Fourier-series expansion of Eq. (21) is

$$\begin{aligned}\left\langle \rho\left(\mathbf{x}^{*(i)}\right)\right\rangle_{N,\mathbf{h}} = \sum_{\mathbf{x}^{*(n)} \in S} m^{(n)} &\left[\sum_{p=0}^{\infty} \hat{a}_{1_p} \cos\left\{\frac{2\pi p\left(x_1^{*(i)} - x_1^{*(n)}\right)}{\ell_1}\right\}\right] \\ &\times \left[\sum_{q=0}^{\infty} \hat{a}_{2_q} \cos\left\{\frac{2\pi q\left(x_2^{*(i)} - x_2^{*(n)}\right)}{\ell_2}\right\}\right],\end{aligned} \tag{24}$$

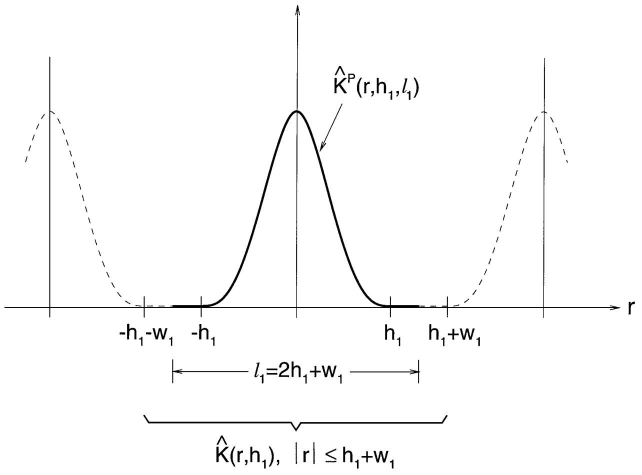


FIG. 2. Periodic extension for the 1D kernel $\hat{K}(r, h_1)$, with period $\ell_1 = w_1 + 2h_1$.

where the Fourier coefficients $\{\hat{a}_{1_p}, \hat{a}_{2_q}\}$ are known functions of the periods $\{\ell_1, \ell_2\}$ and the smoothing lengths $\{h_1, h_2\}$.

This equation can be manipulated. Exchanging the order of summation, expanding the cosine terms (applying the separation property) and considering a finite Fourier expansion, yields

$$\begin{aligned}\left\langle \rho\left(\mathbf{x}^{*(i)}\right)\right\rangle_{N,\mathbf{h},\mathbf{M}} = \sum_{p=0}^{M_1}\sum_{q=0}^{M_2} \hat{a}_{1_p}\hat{a}_{2_q}\Bigg[c_p^i c_q^i \left(\sum_{\mathbf{x}^{*(n)}\in S} m^{(n)} c_p^n c_q^n\right) + c_p^i s_q^i \left(\sum_{\mathbf{x}^{*(n)}\in S} m^{(n)} c_p^n s_q^n\right) \\ + s_p^i c_q^i \left(\sum_{\mathbf{x}^{*(n)}\in S} m^{(n)} s_p^n c_q^n\right) + s_p^i s_q^i \left(\sum_{\mathbf{x}^{*(n)}\in S} m^{(n)} s_p^n s_q^n\right)\Bigg], \end{aligned} \tag{25}$$

in which the following shorthand notation has been used:

$$\begin{aligned} c_p^i &= \cos\left(\frac{2\pi p x_1^{*(i)}}{\ell_1}\right), \quad c_q^i = \cos\left(\frac{2\pi q x_2^{*(i)}}{\ell_2}\right), \\ s_p^i &= \sin\left(\frac{2\pi p x_1^{*(i)}}{\ell_1}\right), \quad s_q^i = \sin\left(\frac{2\pi q x_2^{*(i)}}{\ell_2}\right). \end{aligned} \tag{26}$$

In addition, $\langle \rho(\mathbf{x}^{*(i)})\rangle_{N,\mathbf{h}}$ has been changed to $\langle \rho(\mathbf{x}^{*(i)})\rangle_{N,\mathbf{h},\mathbf{M}}$ to indicate a finite expansion up through modes $\mathbf{M} = \{M_1, M_2\}$. Because the kernels $\hat{K}^P$ are smooth, it is clear that

$$\lim_{\substack{M_1\to\infty \\ M_2\to\infty}} \langle \rho(\mathbf{x})\rangle_{N,\mathbf{h},\mathbf{M}} = \langle \rho(\mathbf{x})\rangle_{N,\mathbf{h}}, \quad \text{all } \mathbf{x} \in R. \tag{27}$$

The quantities in parentheses in Eq. (25) can be calculated independently for each mode pair (p, q), so that the work required to compute $\langle \rho(\mathbf{x}^{*(i)})\rangle_{N,\mathbf{h},\mathbf{M}}$ for all $i : \mathbf{x}^{*(i)} \in R$ scales as $\mathcal{O}(\hat{N} M_1 M_2)$, where $\hat{N}$ is the number of particles in S. By repeatedly applying Eq. (25) to different subregions of D, all mean densities may be obtained in $\mathcal{O}(N M_1 M_2)$ work.

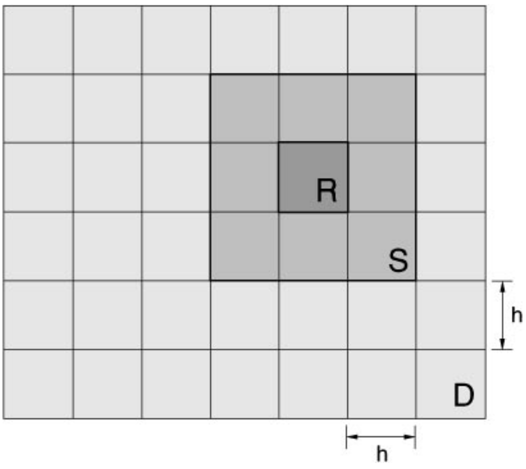


FIG. 3. Domain decomposition for the sample case with $h_1 = h_2 = h$ and $\ell_1 = \ell_2 = 3h$.

A concrete example which clarifies this is now presented. The example consists of a rectangular domain D which is subdivided into a lattice of square cells of side length h, where $h_1 = h_2 = h$. Each subregion R is chosen to be one cell so that $w_1 = w_2 = h$ and $\ell_1 = \ell_2 = 3h$, and each corresponding support region S therefore consists of the 3×3 group of cells centered about R, as shown in Fig. 3. The N particles are assumed to be binned according to this set of cells. For each mode pair (p, q), the following steps are performed:

1. Calculate

$$\sum_{\mathbf{x}^{*(n)} \in R} m^{(n)} c_p^n c_q^n, \quad \sum_{\mathbf{x}^{*(n)} \in R} m^{(n)} c_p^n s_q^n, \quad \sum_{\mathbf{x}^{*(n)} \in R} m^{(n)} s_p^n c_q^n, \quad \sum_{\mathbf{x}^{*(n)} \in R} m^{(n)} s_p^n s_q^n, \tag{28}$$

for each cell R.

2. Using the quantities from step 1, calculate

$$\sum_{\mathbf{x}^{*(n)} \in S} m^{(n)} c_p^n c_q^n, \quad \sum_{\mathbf{x}^{*(n)} \in S} m^{(n)} c_p^n s_q^n, \quad \sum_{\mathbf{x}^{*(n)} \in S} m^{(n)} s_p^n c_q^n, \quad \sum_{\mathbf{x}^{*(n)} \in S} m^{(n)} s_p^n s_q^n, \tag{29}$$

corresponding to each region R. This simply involves sums over cells.

3. Use Eq. (25) to add this mode's contribution to each particle of each region R.

Steps 1 and 3 both require $\mathcal{O}(N)$ work, whereas the work for step 2, $\mathcal{O}(L/h)^2$, is negligible since the number of cells is always much less than the number of particles. Thus, the total work for all mode pairs up through (M_1, M_2) is $\mathcal{O}(NM_1M_2)$. Assuming M_1 and M_2 need only be finite and can be chosen independently of the other numerical parameters (an assumption which will be validated in the next section), this gives a purely $\mathcal{O}(N)$ algorithm.

The periods ℓ_1 and ℓ_2 can be made arbitrarily close to $2h$ by making w_1 and w_2 approach zero. This simply corresponds to making the computational lattice finer. If the cell size in the previous example is reduced to $h/2$ units per side, then each cell R will receive contributions from particles in the 5×5 subset of cells centered about R, and the period can be reduced to $5h/2$. The primary benefit of having a smaller period is faster convergence of $\langle \rho(\mathbf{x}) \rangle_{N,\mathbf{h},\mathbf{M}}$ to $\langle \rho(\mathbf{x}) \rangle_{N,\mathbf{h}}$ with respect to the number of modes.

For practical applications, testing has shown that choosing the number of modes in each direction to be on the order of four gives sufficient resolution. Calculation of gradients requires somewhat higher resolution—on the order of eight modes. As will be shown in Section 5, these requirements are quite modest and allow for efficient calculation of the 2D kernel estimates.

4. ERROR ANALYSIS

As of this point the proper choice of periods ℓ_j and number of modes M_j has not yet been addressed. In this section, an error analysis is performed which yields specific requirements for these parameters given the smoothing lengths h_j.

Two types of error will be studied: relative error and spatial error. The former is useful for estimating how close the approximated kernel estimates $\langle \rho(\mathbf{x}^{*(i)})\rangle_{N,\mathbf{h},\mathbf{M}}$ come to the original desired estimates $\langle \rho(\mathbf{x}^{*(i)})\rangle_{N,\mathbf{h}}$, whereas the latter is studied to determine qualitative effects of the algorithm on convergence of the method.

4.1. Relative Error

The finite Fourier representation of $\hat{K}$ in Eq. (25) corresponds to a modified 1D kernel $\hat{K}'(r, h, \ell, M)$ given by

$$\hat{K}'(r, h, \ell, M) = \begin{cases} \sum_{p=0}^{M} \hat{a}_p(h, \ell) \cos\left(\frac{2\pi pr}{\ell}\right) & \text{for } |r| \leq \ell/2 \\ 0 & \text{for } |r| > \ell/2. \end{cases} \tag{30}$$

A relative error E_R can then be defined as

$$E_R \equiv \int_{-\ell/2}^{\ell/2} |\hat{K}'(r, h, \ell, M) - \hat{K}(r, h)|\, dr, \tag{31}$$

where E_R depends on h, ℓ, and M. This error is simply a measure of how well the modified kernel represents the original kernel, and hence, how close $\langle \rho(\mathbf{x})\rangle_{N,\mathbf{h},\mathbf{M}}$ will be to $\langle \rho(\mathbf{x})\rangle_{N,\mathbf{h}}$. The dependence of E_R on the parameters h, ℓ, and M takes the form

$$E_R\left(M, \frac{\ell}{h}\right) \sim C\left(\frac{Mh}{\ell}\right)^{-q}, \tag{32}$$

where the exponent $q \geq 1$ is determined by the smoothness of $\hat{K}$. In particular, if $\hat{K}$ has j continuous derivatives over $[-\ell/2, \ell/2]$, then q is expected to be $j + 2$. Figure 4 plots E_R versus Mh/ℓ for the piecewise quartic kernel used in this study, Eq. (17). Since the kernel has a continuous second derivative everywhere, q should be 4. This agrees with the plot which clearly shows a $(Mh/\ell)^{-4}$ scaling for large Mh/ℓ. Note that calculation of gradients would reduce this scaling to $(Mh/\ell)^{-3}$, and hence, more modes are required to achieve the same level of resolution.

Equation (32) yields a number of observations:

1. The nondimensional group Mh/ℓ must be large for E_R to be small.
2. To maintain a given tolerance E_R, M must scale as ℓ/h.

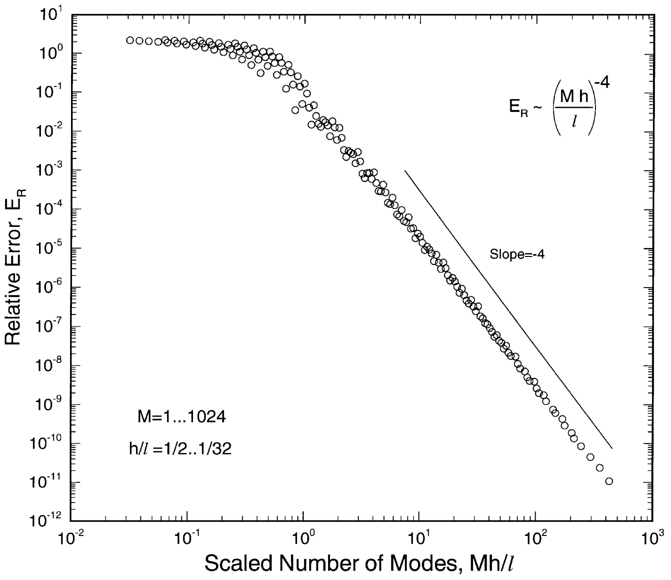


FIG. 4. Computed relative error for the piecewise quadratic kernel, Eq. (17).

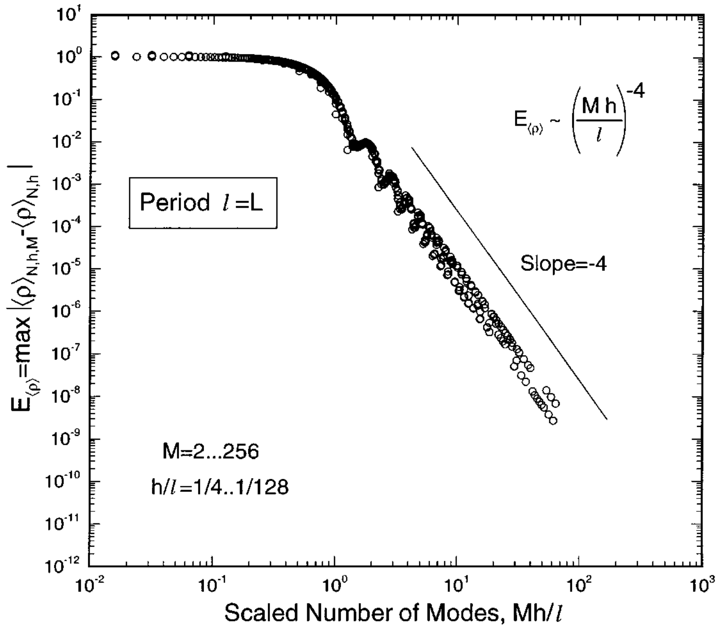
3. With M constant, E_R scales as $(\ell/h)^q$.
4. If ℓ scales with h, then E_R becomes independent of ℓ/h.

The first of these relates to the ability of $\hat{K}'$ to resolve $\hat{K}$. Since the length scale for $\hat{K}$ relative to the period is h/ℓ, modes up through at least $\mathcal{O}(\ell/h)$ are needed for $\hat{K}$ to be accurately resolved. Thus, E_R decreases as the total number of modes becomes large relative to ℓ/h. The second and third observations are intuitive—making ℓ/h smaller lowers minimum mode resolution requirements and therefore decreases the relative error for a given M. At best ℓ may scale as $\mathcal{O}(h)$ (provided $\ell \geq 2h$), in which case the fourth observation applies.

It can be expected that relative errors in kernel estimates scale as E_R. In 1D, for example,

$$E_{\langle Q\rangle} = \max_x |\langle Q(x)\rangle_{N,h,M} - \langle Q(x)\rangle_{N,h}| \sim \alpha E_R\left(M, \frac{\ell}{h}\right), \tag{33}$$

where the coefficient α depends on the quantity $\langle Q\rangle$. This behavior is confirmed in Fig. 5 which shows mean density relative errors for a 1D static test for the two cases $\ell = L$ and $\ell = 2.5h$. To perform the test, 32,768 particles were deterministically positioned according to a predefined density field and the mean density kernel estimates based on $\hat{K}(r, h)$ and $\hat{K}'(r, h, \ell, M)$ were then compared to determine $E_{\langle\rho\rangle}$ as in Eq. (33). The number of modes M was varied between 2 and 256, and the normalized smoothing length h/L was varied between $\frac{1}{4}$ and $\frac{1}{128}$. Each plot shows the behavior of $E_{\langle\rho\rangle}$ versus the relevant scaling parameter: Mh/ℓ for the case $\ell = L$; M for the case $\ell = 2.5h$. In both plots the scaling behavior is as expected. Note also the qualitative similarity between the case $\ell = L$ and Fig. 4, which supports Eq. (33). (The stair-stepping effect visible in the top plot arises due to coupling between the predefined density field chosen for this test and the periodic kernel; it is not an artifact of the Fourier-series algorithm.)



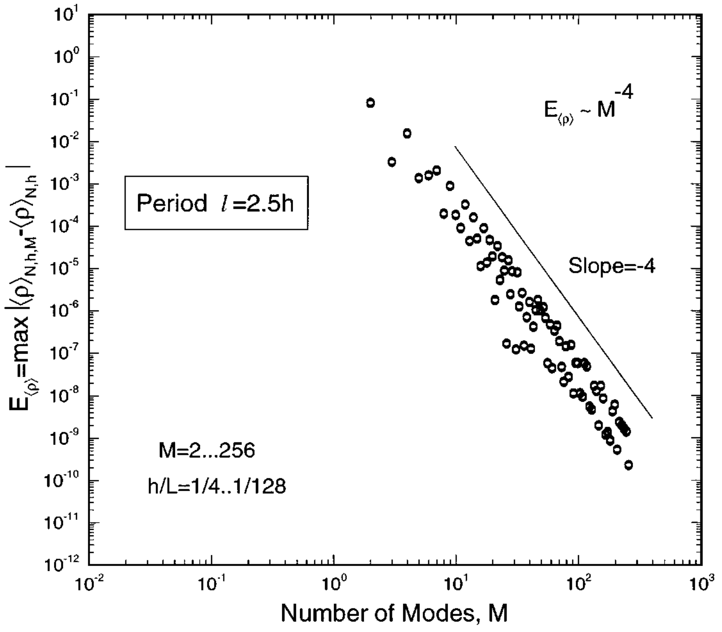


FIG. 5. Mean density relative error, $E_{\langle\rho\rangle}$, for a 1D static test case using $N = 32{,}768$ particles.

4.2. Spatial Error

The dependence of spatial error on the parameters h, ℓ, and M is now discussed. For any quantity Q this error is defined as

$$H_Q = \langle Q\rangle_h - \langle Q\rangle_{h=0}, \tag{34}$$

where $\langle Q\rangle_h$ is given by Eq. (11). Convergence of the spatial error is guaranteed if the kernel satisfies properties (14) and (15). For example, the spatial error for symmetric kernels such as $\hat{K}$ scales as $\mathcal{O}(\bar{h}^2)$, where $\bar{h}$ is the normalized smoothing length h/L. Two important questions to therefore ask are

1. Does the modified kernel $\hat{K}'$ satisfy these properties?
2. If so, what is the scaling of H_Q with $\bar{h}$?

It is straightforward to show that $\hat{K}'$ satisfies the normalization property, Eq. (14), for *all* choices of h, ℓ, and M, since only the mode $p = 0$ contributes to the integral:

$$\int_{-\infty}^{\infty} \hat{K}'(r, h, \ell, M)\, dr = \int_{-\ell/2}^{\ell/2} \hat{a}_0(h, \ell)\, dr = \ell \hat{a}_0 = \ell \left[\frac{1}{\ell} \int_{-\ell/2}^{\ell/2} \hat{K}(r, h)\, dr \right] = 1. \quad (35)$$

The second property requires that $\hat{K}'$ approach a delta function as h approaches zero. In contrast to the previous property, this is *not* guaranteed for arbitrary choices of ℓ and M. An obvious example is $M = 0$ and ℓ is a constant independent of h. This function does not even depend upon h, and hence, cannot satisfy property (15). It can be satisfied, however, by requiring that either ℓ scale as h, or M scale as L/h. The first possibility corresponds to a simple delta sequence for $\hat{K}'$ as h approaches zero, and has the advantage of being valid for any M. The second possibility, although valid in theory, can be rejected from a practical standpoint since it results in an inefficient algorithm as h becomes small. The requirement $\ell = \mathcal{O}(h)$ is thus established which allows $\hat{K}'$ to satisfy the necessary properties of kernels while also taking into account practical considerations.

Given that $\ell = \mathcal{O}(h)$, the kernel $\hat{K}'$ is valid for all choices of M, and spatial error is guaranteed to converge as $\bar{h}$ approaches zero. As it appears in Eq. (30), $\hat{K}'$ is symmetric and thus one would expect H_Q to scale as $\mathcal{O}(\bar{h}^2)$. In the numerical implementation of the $\mathcal{O}(N)$ algorithm, however, the actual kernel used is asymmetric due to phase shifting. Figure 6 illustrates this for the case $\ell = 3h$ and $M = 2$. The solid curves show the modified kernel that would be used to compute kernel estimates at both the center and extreme left of the one-dimensional region R. The kernel used at the center of R is symmetric, but the

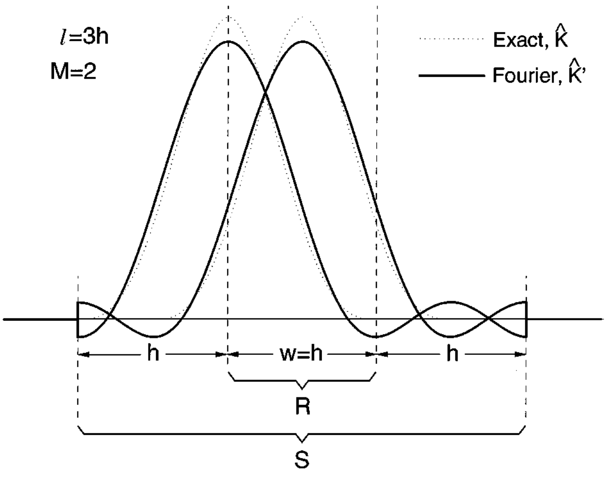


FIG. 6. Kernel asymmetry for the case $\ell = 3h$, $M = 2$.

off-center one (which is phase-shifted by $h/2$) is clearly not symmetric. Consequently, if M is finite then the scaling of H_Q will change to $\mathcal{O}(\bar{h})$ in the limit as $\bar{h}$ approaches zero.

This behavior shows up well in Fig. 7, which presents scalings of H_Q versus $\bar{h}$ as determined from both 1D and 2D numerical tests. These tests involve deterministically positioning a large number of particles according to a known density field and then measuring errors in the mean density kernel estimates relative to the analytic values. The actual number of particles used varies for each value of $\bar{h}$ and is chosen so that the bias is small (less than 1%) relative to the measured spatial error. The top two plots refer to the 1D test, and the bottom two are for the 2D test. On the left is the scaling that results if the original kernel $\hat{K}$ is used (corresponding to direct summation); in both 1D and 2D it is $\mathcal{O}(\bar{h}^2)$ as expected. On the right are results using the new algorithm with $\ell = 3h$ and M ranging from 2 to 16. For large $M (M \geq 8)$ the scaling remains close to $\mathcal{O}(\bar{h}^2)$ over the range of $\bar{h}$ considered. For small M, however, the qualitative change to $\mathcal{O}(\bar{h})$ scaling is evident. For $M = 2$, the change occurs at relatively large values of $\bar{h}$ (comparable to those which would be used in actual simulations). For $M = 4$, the change occurs at values of $\bar{h}$ more than one order of magnitude smaller (for the 2D case the scaling is just beginning to "peel off" at the smallest value of $\bar{h}$).

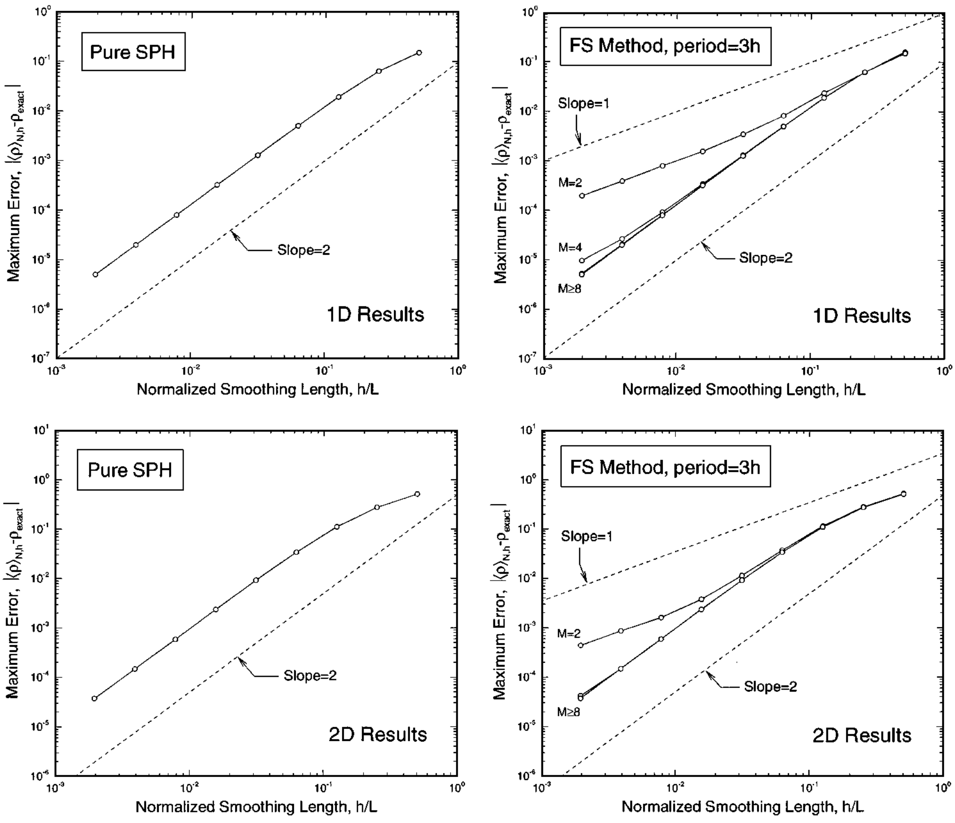


FIG. 7. Spatial error scalings in 1D and 2D using (a) pure SPH and (b) Fourier-series algorithm with $\ell = 3h$. Dotted lines show reference slopes.

Judging from the behavior of these results, the $\mathcal{O}(\overline{h}^2)$ scaling can be recovered for all $\overline{h}$ by allowing M to increase appropriately as $\overline{h}$ decreases, with the precise form of the scaling dependent on the kernel $\hat{K}$. To determine this scaling, the spatial error is written in the form

$$H_Q = g(E_{\langle Q\rangle})\overline{h} + \mathcal{O}(\overline{h}^2), \tag{36}$$

with $g(0) = 0$. For small $E_{\langle Q\rangle}$, g will scale as $E_{\langle Q\rangle}$ to some positive power, and numerical experiments have shown this power to be close to 1. Taking it to be identically one leads to a required scaling $M \sim \overline{h}^{-1/4}$. Note that this scaling need only be implemented in the limit of small $\overline{h}$. For a large working range of $\overline{h}$, the desired $\mathcal{O}(\overline{h}^2)$ scaling will still be observed given a reasonable value of M (say $M = 4$). In addition, the qualitative change in scaling is further delayed by using smaller periods; the results in Fig. 7 are for $\ell = 3h$, whereas in typical simulations ℓ is about $2.25h$.

4.3. Summary

Two different types of error have been studied in this section. Relative error, a measure of how well kernel estimates obtained using the Fourier-series algorithm compare to the original SPH estimates, was shown to scale as the nondimensional group Mh/ℓ to the -4 power for the kernel used in this study. Spatial error, the deterministic error caused by using a finite smoothing length, was shown to converge for the new algorithm for all M, provided $\ell = \mathcal{O}(h)$. With M constant, the scaling is initially $\mathcal{O}(\overline{h}^2)$ and then changes to $\mathcal{O}(\overline{h})$ as $\overline{h}$ approaches zero. This change in scaling, however, has minimal impact, since for reasonable values of M it occurs beyond the typical working range of $\overline{h}$.

5. COMPUTATIONAL COST COMPARISON

This section compares the computational cost of the new Fourier-series algorithm to other algorithms used with SPH. The dependence of computational work on the number of particles is presented for both one- and two-dimensional tests. Although the focus of the paper is on 2D, the 1D results provide a meaningful baseline for comparison.

5.1. Computational Cost in 1D

The Fourier-series algorithm is easily implemented in one dimension, and offers much improved performance compared to a direct summation algorithm. Figure 8 shows the scaling of computational work in 1D for various methods. The test used to determine the scaling simply involved calculating mean densities as per Eqs. (13) and (25) for a fixed number N of deterministically positioned particles. For cases with N small, the test was performed multiple times to increase the timing resolution. The vertical scale is the CPU time required per particle, measured in seconds. This is a useful performance indicator since any method will scale at least as $\mathcal{O}(N)$. This quantity is plotted versus $\overline{N}$, the average number of particles contributing to each kernel estimate. (In D dimensions $\overline{N} = N(h/L)^D$.) In Monte Carlo/PDF simulations of fluid flows, $\overline{N}$ is a key parameter controlling the level of statistical error. For the direct summation algorithm, it is also the key parameter determining the expense of the method.

As expected, the scaling of the CPU time per particle with $\overline{N}$ is linear for the direct summation algorithm, whereas the Fourier-series algorithm is independent of $\overline{N}$. CPU time

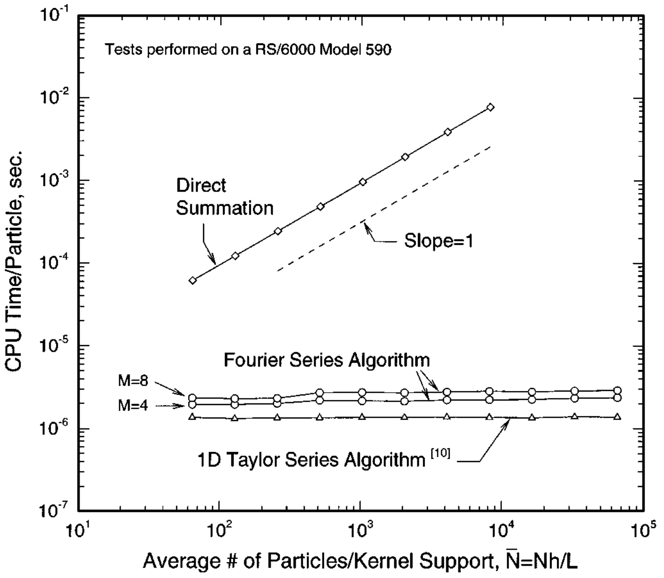


FIG. 8. Comparison of computational cost for various 1D algorithms: (a) direct summation; (b) Taylor-series; (c) Fourier-series with $M = 4$ and $M = 8$.

per particle does increase as M is made larger, as Fig. 8 shows. The intersection of the Fourier-series and direct summation curves, although not explicitly shown on the plot, is roughly at $\overline{N} \approx 1$. even with $M = 8$. If a small amount of relative error is acceptable, then the Fourier-series algorithm is clearly a superior choice over direct summation for SPH kernel estimation.

Also shown on the plot is the scaling for the Taylor-series algorithm described in [10]. In 1D it is approximately 2–3 times more efficient than the Fourier-series algorithm, and it also has zero relative error. It is therefore the method of choice for 1D problems. In higher dimensions, however, its scaling is not as good ($\mathcal{O}(N\overline{N}^{1/2})$ in 2D) and its implementation is also difficult.

5.2. Computational Cost in 2D

Figure 9 shows computational cost scalings for various two-dimensional algorithms. The test used here is the same: measure the CPU time needed to calculate all 2D mean density kernel estimates for a set of N deterministically positioned particles. As in 1D, the work per particle increases linearly with $\overline{N}$ for the direct summation algorithm and remains constant for the Fourier-series algorithm. Two sets of curves are presented for the Fourier-series algorithm, and they are labeled “Fourier–Fourier” and “Fourier–Taylor.” The former corresponds to the algorithm described in Section 3 (in which both 1D kernels are expanded in Fourier-series), whereas the latter is a modification to the algorithm that is described in Section 8 (only one kernel is expanded). In this section focus is still on the “Fourier–Fourier” implementation.

Compared to 1D, computational cost for the Fourier-series algorithm is more by a factor of about five for the two cases shown, a consequence of the scaling changing from $\mathcal{O}(NM)$ to $\mathcal{O}(NM^2)$. Relative to direct summation, however, the algorithm is still superior. The

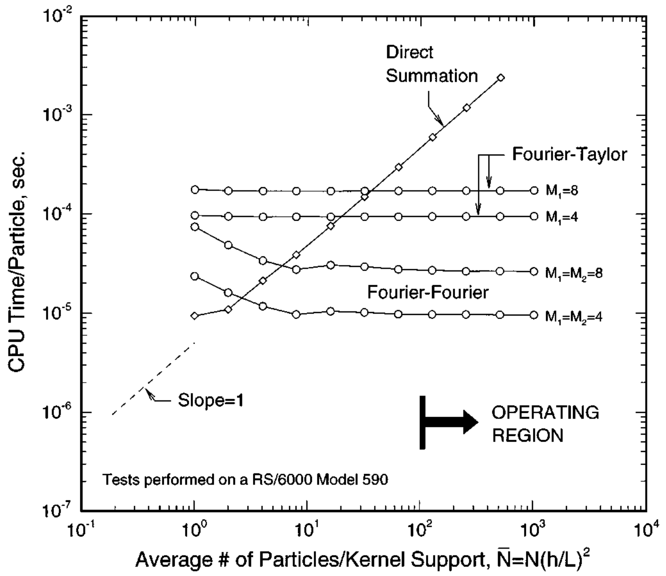


FIG. 9. Computational cost comparison of selected 2D algorithms: (a) direct summation; (b) Fourier–Fourier with $M = 4$ and $M = 8$; (c) Fourier–Taylor with $M = 4$ and $M = 8$.

intersection of their respective curves is at about $\bar{N} \approx 5$, whereas the typical operating region for $\bar{N}$ in Monte Carlo/PDF simulations begins at $\bar{N} \approx 100$ [21]. The results presented in Section 7 have $\bar{N}$ on the order of 1000, corresponding to a computational savings of two orders of magnitude over direct summation.

6. NUMERICAL IMPLEMENTATION

Numerical implementation of the combined PDF/SPH method is straightforward. At the start of a simulation particles are distributed in the domain and their properties are set according to a specified initial condition. This initial condition can be either deterministic or random. In the latter case, requirements are imposed on statistical moments of quantities; for example, the initial mean velocity field may be specified everywhere in the domain. The initial particle properties are then random variables having these moments. With the initial condition in place, the particle simulation is then advanced in time. At each time step this involves computing mean coefficients from the current particle distribution via SPH kernel estimates, enforcing boundary conditions, and integrating the stochastic particle evolution equations forward in time.

6.1. Evaluation of Coefficients

Coefficients for each of the N particles are calculated using the Fourier-series algorithm and include $\langle \rho \rangle$, $\nabla \langle P \rangle$, $\tilde{U}_i$, and $\tilde{k}$. (The special case of ω being a coefficient is described in Section 7.) The mean densities are obtained using Eq. (13), and mean velocities and turbulent kinetic energies are obtained using appropriate forms of Eq. (12). Calculation of the mean pressure gradient is performed using a modified approach which derives from the equation of state. For the isentropic flows considered in this paper, the mean pressure and density are related through

$$\left(\frac{\langle P\rangle}{P_0}\right) = \left(\frac{\langle\rho\rangle}{\rho_0}\right)^{\gamma}, \tag{37}$$

where γ is the ratio of specific heats, and P_0 and ρ_0 are reference values. From this equation of state, the pressure gradient is obtained using

$$\nabla\langle P\rangle = \gamma\frac{\langle P\rangle}{\langle\rho\rangle}\nabla\langle\rho\rangle. \tag{38}$$

This form is used for two reasons. First, it has the advantage of having the density inside the gradient operator [11] and is thus more accurate. Second, it is computationally efficient within the context of the Fourier-series algorithm: $\nabla\langle\rho\rangle$ can be obtained simultaneously along with $\langle\rho\rangle$ by simply replacing the Fourier coefficients $\hat{a}_p$ for the 1D kernel $\hat{K}$ with those for its derivative, $\hat{b}_p = (2\pi/\ell)p\hat{a}_p$.

6.2. Boundary Conditions

Two types of boundary conditions are presently implemented in the 2D particle method: periodic and inflow/outflow. Unlike grid-based methods which apply boundary conditions on grid points, the particle method enforces these conditions directly using particles. This is consistent with the fully Lagrangian framework of the method.

Periodic boundary conditions are easy to implement and have been used extensively during development and testing of the method. A particle which exits the domain along one boundary simply reenters from the opposite side with the same properties. The kernel is also extended periodically.

Inflow/outflow boundary conditions are more difficult to implement but make possible the simulation of more realistic flow problems. The implementation used here is the 2D extension of the one presented in [10]. Since the method is applicable to compressible flows, a characteristic-based approach is used [22, 23]. The approach involves determining values for incoming and outgoing characteristics at the inlet and outlet boundaries. Values for characteristics exiting the domain are obtained from the interior particles using straight kernel estimates, while values for those entering are determined from the applied physical boundary conditions (such as a mean velocity profile at the inlet or an exit pressure at the outlet). Actual values for the applied boundary conditions are then computed from these two sets of characteristics using standard methods.

Rectangular buffer zones upstream of the inlet and downstream of the outlet are used to enforce the boundary conditions. Within these zones, particles are distributed based upon the calculated boundary values for density, velocity, and other variables (which depend on the type of flow). The two main advantages of this approach are that kernel estimates for all interior particles can be obtained using the same method (no special treatment must be made for particles near the boundaries), and particles will flow naturally into and out of the domain. Full details of the inflow/outflow implementation are available in Ref. [10].

6.3. Predictor/Corrector Scheme

The stochastic system of evolution equations is integrated forward in time using a weak second-order accurate two-step scheme. At the beginning of a time step, all coefficients are computed from the current particle distribution, and the evolution equations are integrated forward in time to yield predicted values for each particle property. Predicted values for the coefficients are then obtained from the new particle properties and averaged with the initial coefficients. Finally, the evolution equations are integrated forward in time as before, but this time using the averaged coefficients to obtain corrected values for the particle properties. Using this scheme with a fixed time-step Δt, the temporal error in mean quantities will scale as $\mathcal{O}(\Delta t^2)$ [10, 24].

7. SAMPLE 2D RESULTS

The combined PDF/SPH method has been used to obtain results for a variety of 2D flow problems, and this section presents a sampling of these results. Examples include a stationary turbulent plane wake, hydrostatic flows under the influence of body forces, and a fully unsteady flow featuring compression/expansion waves and a pair of decaying vortices. These examples serve to demonstrate the robustness of the method, as well as its feasibility in terms of required computational resources.

7.1. 2D Plane Wake

The particle code has been used to simulate the spatial evolution of the two-dimensional turbulent plane wake behind a circular cylinder. This is a problem which has been extensively studied in the laboratory and for which much experimental data exist. Comparisons of the particle code results are made to some of this data. The problem also provides a meaningful test of the particle code's ability to simulate a flow of practical importance.

7.1.1. *Background*

Figure 10 shows the physical layout of the problem. A cylinder is placed in a uniform flow of speed U_E so as to produce a developing turbulent wake downstream. The velocity defect U_D at the center of the wake gradually decreases with downstream distance, and the width of the wake, shown as δ, grows as the turbulent fluid mixes and spreads into the surrounding freestream flow. The precise definitions of U_D and δ are

$$U_D(x_1) \equiv U_E - \tilde{U}_1(x_1, 0), \tag{39}$$

$$\tilde{U}_1(x_1, \delta) = U_E - \frac{U_D(x_1)}{2}. \tag{40}$$

Both theoretical and experimental observations have shown that the 2D wake exhibits asymptotic self-similarity in the limit as $x_1 \to \infty$. In this limit, transverse profiles of mean velocity and Reynolds stresses are fully characterized by the parameters U_E, U_D, and δ shown in Fig. 10. Given values for these parameters at a station x_1 sufficiently far downstream from the cylinder, the mean streamwise velocity scales as

$$\tilde{U}_1(x_1, x_2) = U_E - U_D(x_1)\, f(\eta), \tag{41}$$

while the Reynolds stresses scale as

$$\widetilde{u_i u_j}(x_1, x_2) = U_D^2(x_1) g_{ij}(\eta). \tag{42}$$

Here, the similarity variable η is defined x_2/δ.

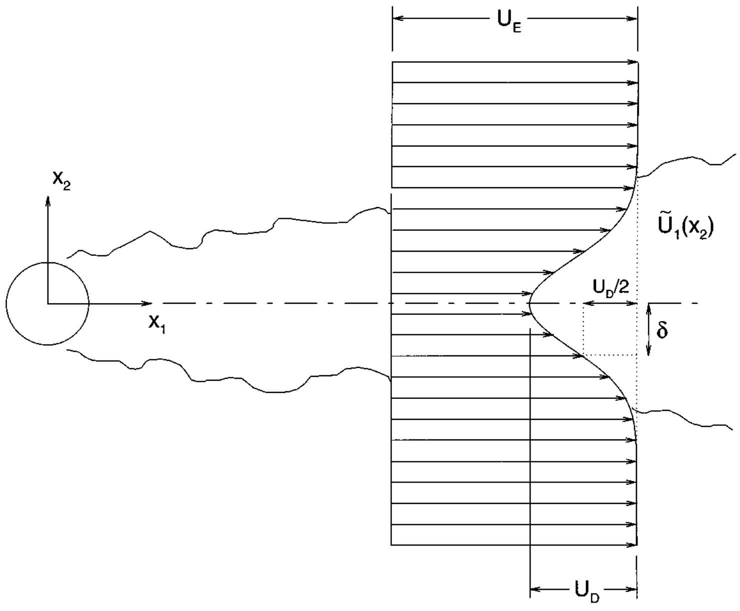


FIG. 10. Physical setup of 2D plane wake problem, showing the parameters U_E, U_D, and δ.

Wygnanski *et al.* [25] found that the function $f(\eta)$ was universal; different wake generators gave self-similar mean velocity distributions which all yielded the same functional form for f. In particular, an excellent fit to their experimental data was provided by the form

$$f(\eta) = \exp(-0.637\eta^2 - 0.056\eta^4), \tag{43}$$

which is plotted in Fig. 11. The Reynolds stresses were found to not be universal. For a given wake generator, the evolution of $\widetilde{u_i u_j}$ did become self-similar, but different generators led to different forms for g_{ij}. Some insight into this behavior is gained through the mean streamwise momentum equation. Assuming $\partial\langle P\rangle/\partial x_1$ is zero, it leads to the following relationship between f and g_{12},

$$g_{12}(\eta) = -S\eta f, \tag{44}$$

where

$$S = \frac{1}{2}\frac{d}{dx_1}\left(\frac{U_E\delta}{U_D}\right) \tag{45}$$

is the *spreading rate*. The value of S depends on the type of wake generator used; experiments for a circular cylinder give $S = 0.082$ [25]. The curve for g_{12} based on this value of S and the function f defined in Eq. (43) is also plotted in Fig. 11. The remaining three curves in the figure show the scaled normal stresses. The data for these come from circular cylinder wake experiments performed by Townsend [26].

7.1.2. *Varying ω Implementation*

A requirement for self-similarity is that the turbulence time scale be proportional to the mean flow time scale. The former is given by $1/\omega$ and is typically assumed constant across

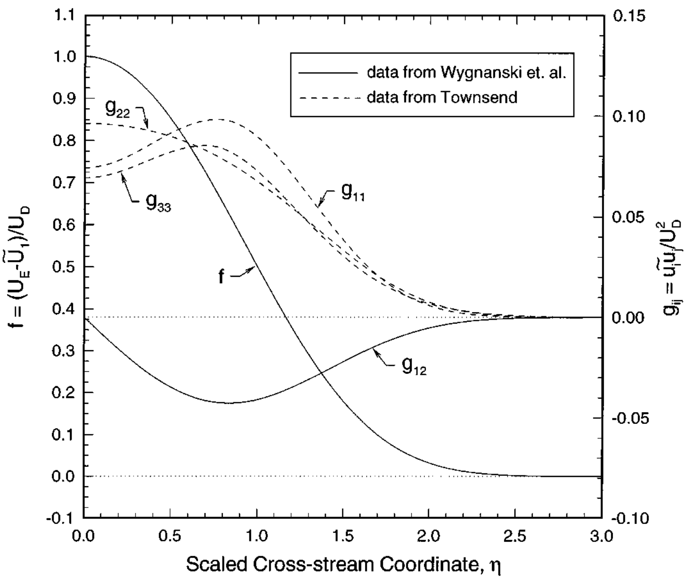


FIG. 11. Scaled experimental profiles for 2D plane wake. Mean velocity and shear stress data are from Wygnanski *et al.* [25], and normal stress data are from Townsend [26].

the flow [17], while the latter is characterized by δ/U_D. The requirement is, therefore,

$$\omega = \omega^*(U_D/\delta), \tag{46}$$

where ω^* is a constant. Using this form ω is a coefficient which varies with downstream position, and it must be computed for each particle.

The procedure to compute ω is straightforward. After multiplying the numerator and denominator of Eq. (46) by U_D and recognizing that $U_D \cdot \delta$ is proportional to the velocity flux defect $\mathcal{F}$, this gives

$$\omega = c_1 U_D^2/\mathcal{F}. \tag{47}$$

Both U_D and $\mathcal{F}$ are computed directly from the particles using 1D kernel estimates. Defining $\hat{U}_1^{(n)} = U_E - U_1^{*(n)}$, these estimates are

$$U_D\left(x_1^{*(i)}\right) = \frac{1}{\left\langle \rho\left(x_1^{*(i)}, 0\right)\right\rangle} \sum_{n=1}^{N} m^{(n)} \hat{U}_1^{(n)} \hat{K}\left(x_1^{*(i)} - x_1^{*(n)}, h_1\right) \hat{K}\left(0 - x_2^{*(n)}, h_2\right), \tag{48}$$

$$\mathcal{F}\left(x_1^{*(i)}\right) = \frac{L_2}{\left\langle \bar{\rho}\left(x_1^{*(i)}\right)\right\rangle} \sum_{n=1}^{N} m^{(n)} \hat{U}_1^{(n)} \hat{K}\left(x_1^{*(i)} - x_1^{*(n)}, h_1\right), \tag{49}$$

where

$$\left\langle \rho\left(x_1^{*(i)}, 0\right)\right\rangle = \sum_{n=1}^{N} m^{(n)} \hat{K}\left(x_1^{*(i)} - x_1^{*(n)}, h_1\right) \hat{K}\left(0 - x_2^{*(n)}, h_2\right), \tag{50}$$

$$\left\langle \bar{\rho}\left(x_1^{*(i)}\right)\right\rangle = \sum_{n=1}^{N} m^{(n)} \hat{K}\left(x_1^{*(i)} - x_1^{*(n)}, h_1\right), \tag{51}$$

and L_2 is the height of the domain. Note that $\mathcal{F}$ will incur some error since the height is finite. This error decreases rapidly though as L_2 is increased (it is negligible for the results shown in this section).

The constant c_1 can be related to ω^* through Eq. (41) in that the ratio c_1/ω^* is just the integral of $f(\eta)$ over all η. Using the form of f given in Eq. (43) this ratio is 2.06, and so the final relationship for ω is

$$\omega = 2.06\,\omega^*\,U_D^2/\mathsf{F}. \tag{52}$$

7.1.3. *Boundary Conditions*

Numerical simulation of the wake is performed in a rectangular domain, with inflow and outflow boundary conditions implemented on the left and right sides, and periodic boundary conditions used on the top and bottom. The inlet boundary condition requires (as an input) profiles for $\tilde{U}_1$, $\widetilde{u_i u_j}$, and $\langle P \rangle$ (or $\langle \rho \rangle$). The choice of these is somewhat arbitrary. Ideally, they should have the same general shape as the experimental profiles, yet still be sufficiently different to show that they can evolve to ones that agree well with experiment. The scaled inlet profiles for $\tilde{U}_1$ and $\widetilde{u_i u_j}$ have been made piecewise linear functions, as Fig. 12 shows. All three normal stresses have the same profile. Peak values for the scaled shear and normal stresses were chosen to be 0.05 and 0.10, respectively.

Given these scaled profiles, only U_E, U_D, and δ need be specified at the inlet. The inlet profile for pressure is determined using the steady-state cross-stream momentum equation:

$$\frac{\langle P \rangle}{\langle \rho \rangle} + \widetilde{u_2^2} = \frac{P_0}{\rho_0}. \tag{53}$$

Since $\widetilde{u_2^2}$ is small relative to the other terms, the inlet profile for $\langle P \rangle$ is nearly uniform. For the outlet boundary condition the only flow quantity which must specified is the pressure,

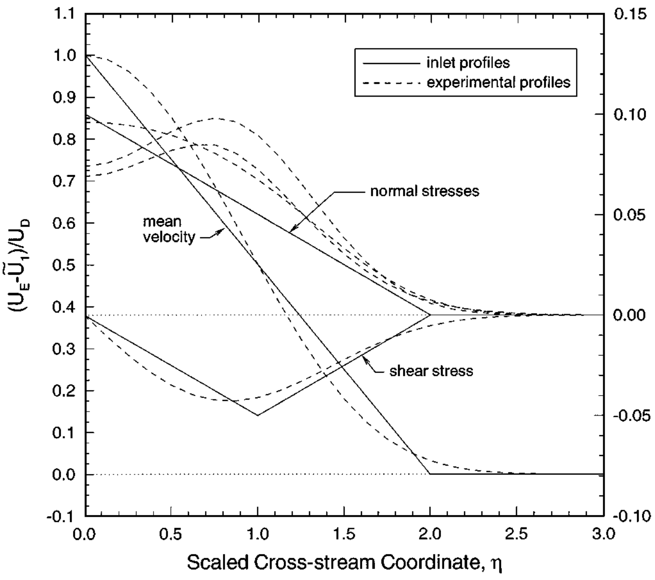


FIG. 12. Specified inlet profiles (scaled) for 2D plane wake problem.

and it is assumed constant and equal to P_0. This is a reasonable assumption since $\widetilde{u_2^2}$ is approximately zero at the outlet.

7.1.4. *Numerical Results*

The simulation for which results will be presented was performed in a domain of length $L_1 = 6.0$ and height $L_2 = 1.5$. A length of 6.0 allows for sufficient evolution of the wake beyond the inlet and also leaves space to study profiles over a reasonable range of downstream stations. The height of 1.5 is large enough so that profiles do not merge across the top and bottom boundaries (a potential problem when using periodic boundary conditions on these boundaries). The smoothing lengths (unnormalized) used in the simulation were $h_1 = \frac{1}{8}$ and $h_2 = \frac{1}{16}$. A larger value can be used for h_1 since variations in flow quantities are significantly smaller in the streamwise direction. The number of particles initially positioned (uniformly) in the domain was 1,180,000. Compared to a particle-mesh implementation, this corresponds to a 48×24 grid of cells, with about 1000 particles per cell. The initial mean and fluctuating velocities for all particles were based on the inlet profiles. The simulation was integrated out to time $T = 40.0$ using a time step of 0.01. This time step allows sufficient resolution of the turbulent time scale, that is $\omega \Delta t \ll 1$ for the range of ω in this problem. To reduce statistical error, time-averaging was performed on all mean quantities of interest from $T = 15.0$ (by which time all the particles initially in the domain had exited) until $T = 40.0$. At the inlet, the free-stream velocity U_E was set to 0.6, the centerline velocity defect U_D to 0.3, and the wake width δ to 0.125. The cross-stream smoothing length of $\frac{1}{16}$ is adequate to resolve the inlet profiles. Finally, the constant ω^* was set to 0.24. This value was selected to make the computed peak shear stress match the experimental value.

Figure 13 compares scaled mean velocity profiles obtained using the particle code to the experimental profile of Wygnanski *et al.* The overall agreement across the wake is

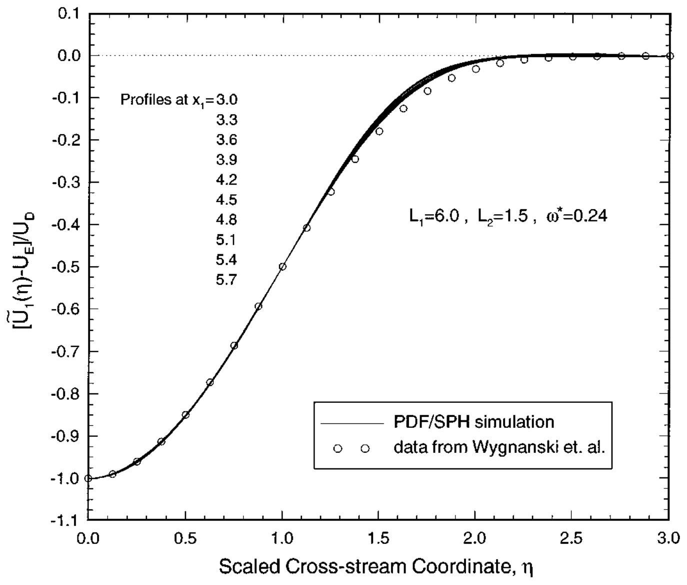


FIG. 13. Scaled profiles of $\tilde{U}_1$ for the 2D plane wake, with $\omega^* = 0.24$. The simulation was performed using 1,180,000 particles and smoothing lengths $h_1 = \frac{1}{8}$ and $h_2 = \frac{1}{16}$. The fields have been time-averaged.

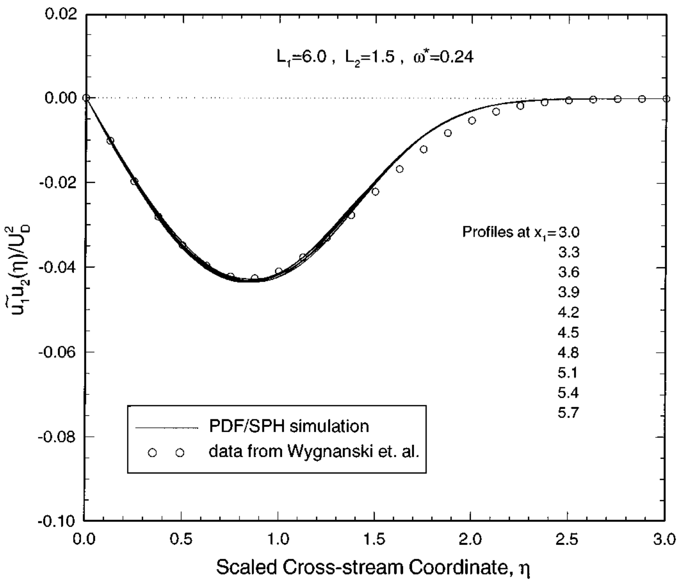


FIG. 14. Scaled profiles of $\widetilde{u_1 u_2}$ for a 2D plane wake simulation with 1,180,000 particles and smoothing lengths $h_1 = \frac{1}{8}$ and $h_2 = \frac{1}{16}$. The fields have been time-averaged.

very good. The only noticeable discrepancy is at the edge of the wake, where the particle code slightly underpredicts the profile width. The 10 profiles plotted in the figure were measured at streamwise positions ranging from $x_1 = 3.0$ to $x_1 = 5.7$, and over this range the self-similarity in $\tilde{U}_1$ is clearly visible.

Scaled shear stress profiles are shown in Fig. 14. Once again they show excellent self-similarity, and the scaled curve agrees well with the experimental data. The shear stress is somewhat underpredicted at the edge to give a slightly narrower turbulent region, but this is expected of the simplified Langevin model [17].

The normal stresses are plotted in Fig. 15. Compared to their initially isotropic state at the inlet, the stresses do indeed evolve to reasonable profiles, and the scaled curves are sufficiently close together to justify self-similarity. Although the agreement with experimental data is not as good here as with the previous quantities, it is consistent with past behavior of the simplified Langevin model. Plane wake computations performed by Haworth and Pope [17] showed that this model, as well as numerous others, put too much energy into the streamwise component, and thus $\widetilde{u_1^2}$ is overpredicted. Figure 15 also shows that the stresses do not decay to zero at the edge of the wake. This is purely a consequence of bias in the particle simulation (and part of the reason why $\overline{N}$ is made large). Understanding sources of bias in PDF computations was recently studied in [27].

Figure 16, the last in the group for this test problem, shows the spatial evolution of various wake quantities including the centerline defect velocity, the wake width, omega, and the edge velocity. In the downstream portion of the domain, both U_D and δ evolve close to their expected rates:

$$U_D(x_1) \sim (x_1 - x_0)^{-0.5}, \tag{54}$$

$$\delta(x_1) \sim (x_1 - x_0)^{+0.5}. \tag{55}$$

Dashed lines in the figure are least-squares curvefits of this form obtained by fitting data

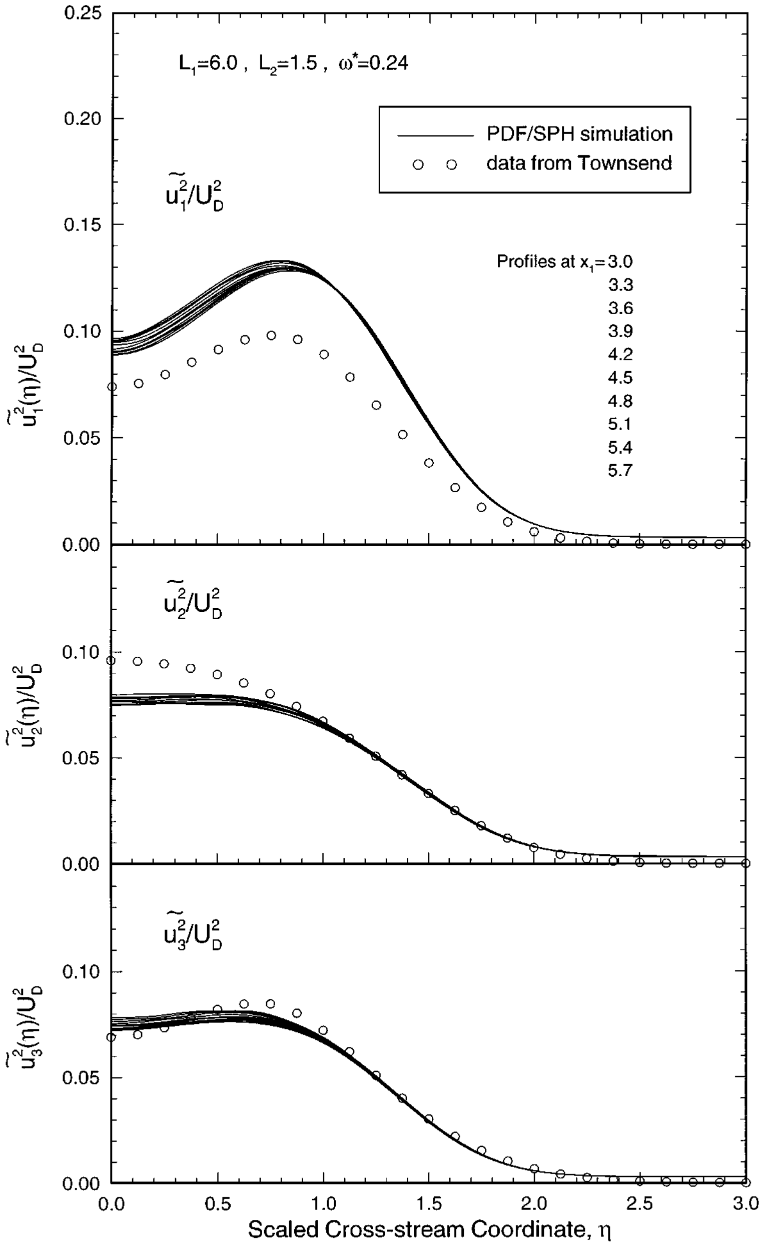


FIG. 15. Scaled profiles of $\widetilde{u_i^2}$ for the 2D plane wake, with $\omega^* = 0.24$. The simulation was performed using 1,180,000 particles and smoothing lengths $h_1 = \frac{1}{8}$ and $h_2 = \frac{1}{16}$. The fields have been time-averaged.

in the range $x_1 \in [3, 6]$. (The same value of x_0 is used in both fits.) Of course, U_D and δ cannot be expected to show the proper evolution in the upstream portion of the domain.

The edge velocity is plotted at the top of Fig. 16. For an incompressible flow this should ideally remain constant. In the current simulation though, the density increases slightly with downstream position which leads to the decay in U_E. The majority of the decay is in the upstream portion, where the density changes most rapidly, although the actual change in ρ

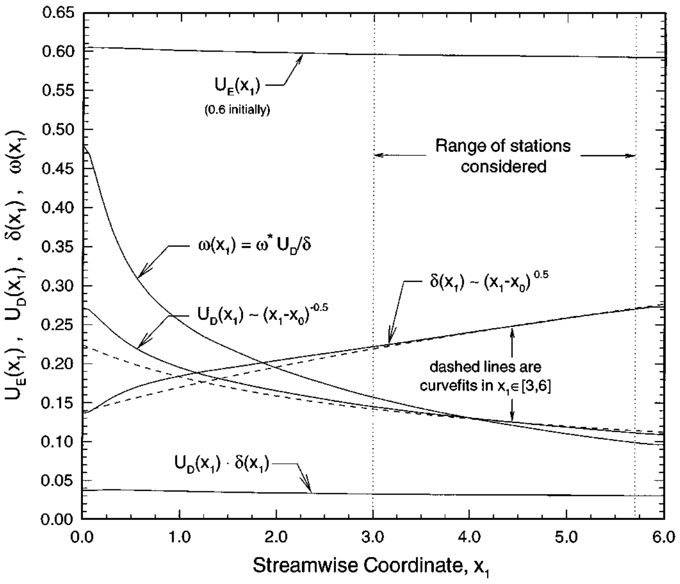


FIG. 16. Spatial evolution of U_E, U_D, δ, and ω for the 2D plane wake, with $\omega^* = 0.24$. The simulation was performed in a domain of length 6.0 and height 1.5 using 1,180,000 particles and smoothing lengths $h_1 = \frac{1}{8}$ and $h_2 = \frac{1}{16}$.

is only about 1%. Over the region used for studying profiles (between the dotted vertical lines), the variation in U_E is less than half of one percent.

7.2. Hydrostatic Flows with Body Forces

The particle method's ability to handle body forces has been studied through two static test problems. In each of these, the effect of the body force is equivalent to a gravitational force. The simulation evolves in time until a steady-state hydrostatic pressure distribution is attained, and the resulting distribution is compared to the predicted theoretical distribution.

The two problems differ in the orientation of the force vector. In the first, a force per unit mass of fixed magnitude g is added in the x_2 direction by modifying the evolution equation for U_2^*:

$$dU_2^* = -\frac{1}{\langle\rho\rangle}\frac{\partial\langle P\rangle}{\partial x_2}dt - \cdots + g\,dt. \tag{56}$$

The direction of the force vector is vertical and always points towards the horizontal line passing through the center of the domain, $x_2 = 0$. This divides the domain into two symmetrical halves, with the hydrostatic pressure distribution being homogeneous in x_1. In the second test case, the force vector retains the same magnitude, but it is set to always point radially towards the center of the domain (the equation for U_1^* is also modified in this case). The resulting pressure distribution is axisymmetric.

In both cases the hydrostatic pressure distributions are nonlinear since the fluid is compressible. At steady state the mean momentum equation in x_2 for the horizontal body force problem is

$$0 = -\frac{1}{\langle\rho\rangle}\frac{d\langle P\rangle}{dx_2} - g. \tag{57}$$

Together with the equation of state, Eq. (37), the solution to this ODE is

$$\langle\rho(x_2)\rangle = \rho^*\left[1 - \frac{(\gamma-1)}{\gamma}\frac{g|x_2|}{P^*/\rho^*}\right]^{1/(\gamma-1)} \quad \text{for } |x_2| \le L/2, \tag{58}$$

with ρ^* and P^* the density and pressure at $x_2 = 0$ (these are found given the total mass of fluid in the domain). The solution for the axisymmetric case is identical, except that x_2 is replaced by the radial distance r.

Figure 17 compares the particle code hydrostatic pressure distributions for the two cases to the theoretical distributions for the choice $g = 5.0$. In both cases the agreement is excellent. To perform each simulation, 128,000 particles were uniformly distributed with zero velocity in the unit square domain. The total mass of the particles was chosen to be 1. Using a normalized smoothing length of $\frac{1}{16}$, the simulation was then evolved in time with a time step of 0.01 until a stationary pressure distribution was reached at time $T = 10.0$. Time averaging was subsequently done on the pressure field until $T = 20.0$. The number of Fourier modes used to approximate each 1D kernel was chosen to be four, and eight modes were used to approximate the kernel derivative. Throughout the simulation, the turbulence model was kept on as a means to speed convergence and keep random fluctuations small, since it explicitly adds dissipation through the drift terms. Apart from this contribution,

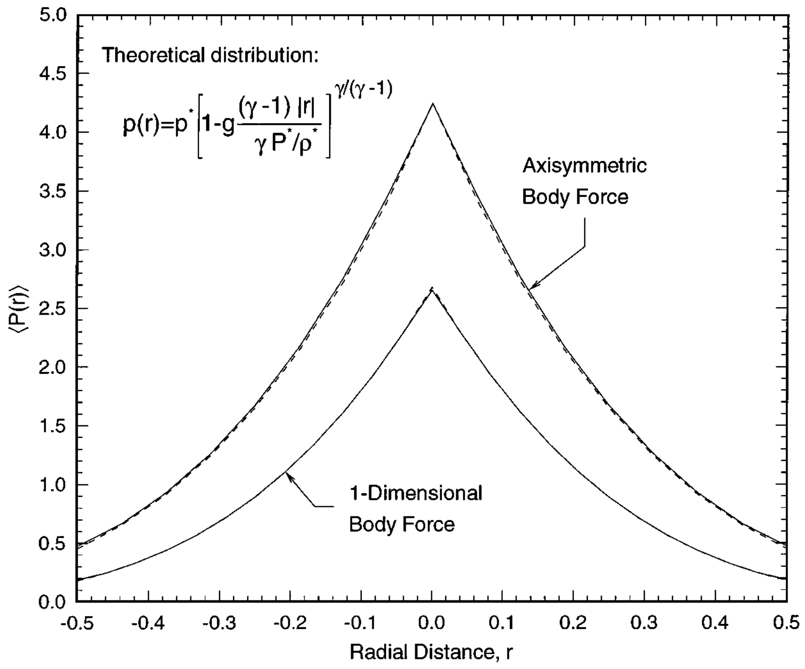


FIG. 17. Hydrostatic pressure distributions for the 1D and axisymmetric body force test cases as predicted from a simulation using 128,000 particles and normalized smoothing lengths $\bar{h}_1 = \bar{h}_2 = \frac{1}{16}$.

the method has very little dissipation and, therefore, would require much longer to reach a stationary state.

7.3. Unsteady Compressible Flow Problem

A third sample flow problem which has been studied highlights some of the effects of pressure on the flow evolution. This flow has an initial condition consisting of a central square region of isotropic turbulent fluid moving with a constant horizontal mean velocity U_H, which is surrounded by ambient fluid. The density is initially constant and equal to 1 everywhere. Although this is a less physical problem than the 2D wake, it is nevertheless interesting: the flow is fully unsteady and inhomogeneous, it contains compression and expansion waves, and it features a pair of decaying eddies. All of these phenomena are exhibited in the simulation results shown in Fig. 18.

For this simulation, the size of the central moving region was set to $\frac{1}{4}$ by $\frac{1}{4}$, and the domain was chosen to be the unit square. The value of U_H was set to 1.0, and the turbulence intensity in the moving fluid was set at 10% the speed of sound. The smoothing length $\overline{h}$ for the simulation was made 1/16, and the number of particles N was chosen to be 512,000. The latter was made quite large so as to keep statistical fluctuations and bias small; the number of particles contributing to each kernel estimate is roughly 8000. Four Fourier modes were used to approximate each 1D kernel, while eight modes were used for the kernel derivative. Using a time-step of 0.005, the solution through $t = 2.5$ was obtained in slightly over 9 h on an IBM RS/6000 Model 590. This is quite reasonable given the magnitude of the simulation.

The top row of plots shows the evolution of pressure, with solid lines indicating compression and dashed lines indicating expansion. At $t = 0.02$, the moving fluid is just beginning to form expansion and compression waves to the left and right of center, respectively. The compression wave shapes itself into a well-defined arc as it propagates through the domain,

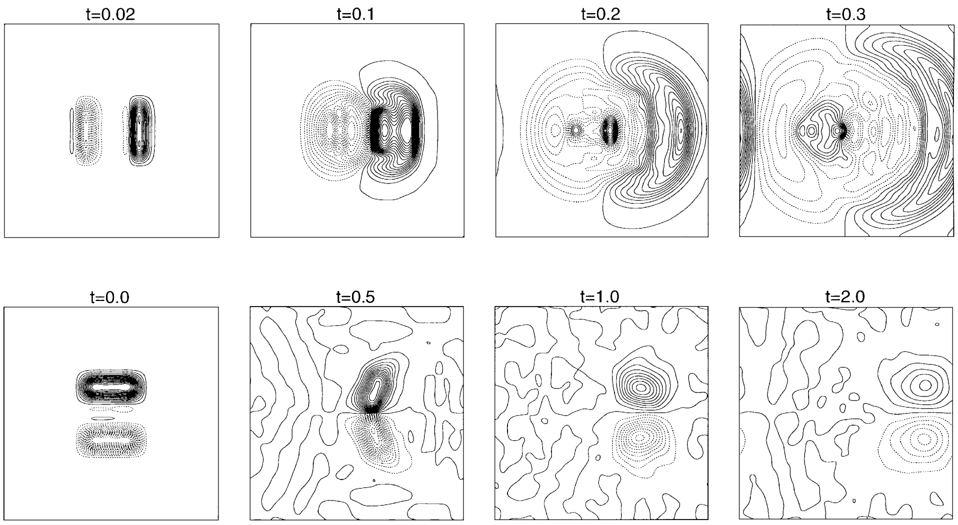


FIG. 18. Evolution of pressure (top) and vorticity (bottom) for the unsteady compressible flow problem. Five hundred twelve thousand particles were used in the simulation, with normalized smoothing lengths equal to $\frac{1}{16}$.

whereas the expansion wave fills most of the left half of the domain. Periodic boundary conditions are used in both coordinate directions, so that at later times these waves come into contact again and interact.

The bottom set of plots show vorticity as computed from the mean velocity field. Solid/dashed lines show positive/negative vorticity, respectively. A pair of eddies forms in the wake of the moving central region as fluid from above and below rushes in to fill the low pressure area. At $t = 0.5$ this behavior is evident. The eddies continue to slowly drift to the right before eventually decaying, as the final bottom two contour plots show.

8. VARIATIONS/EXTENSIONS TO ALGORITHM

8.1. Fourier–Taylor Variation

The algorithm described in Section 3 was formulated by expanding each 1D kernel in a Fourier series. A variation of this approach is now described in which only one kernel is expanded. This variation, referred to by the author as "Fourier–Taylor," was already mentioned in reference to Fig. 9 of Section 5.

As in Section 3, the approach is best explained by using the example of calculating mean densities in a subregion R. Expanding the kernel in x_1, for example, gives

$$\left\langle \rho\left(\mathbf{x}^{*(i)}\right)\right\rangle_{N,\mathbf{h},M} = \sum_{\mathbf{x}^{*(n)}\in S} m^{(n)} \left[\sum_{p=0}^{M} \hat{a}_{1_p} \cos\left\{\frac{2\pi p\left(x_1^{*(i)} - x_1^{*(n)}\right)}{\ell_1}\right\}\right] \hat{K}\left(x_2^{*(i)} - x_2^{*(n)}, h_2\right), \tag{59}$$

which becomes (after manipulation)

$$\begin{aligned}\left\langle \rho\left(\mathbf{x}^{*(i)}\right)\right\rangle_{N,\mathbf{h},M} = \sum_{p=0}^{M} \hat{a}_{1_p} &\left[c_p^i \left(\sum_{\mathbf{x}^{*(n)}\in S} m^{(n)} c_p^n \hat{K}\left(x_2^{*(i)} - x_2^{*(n)}, h_2\right)\right)\right. \\ &\left. + s_p^i \left(\sum_{\mathbf{x}^{*(n)}\in S} m^{(n)} s_p^n \hat{K}\left(x_2^{*(i)} - x_2^{*(n)}, h_2\right)\right)\right].\end{aligned} \tag{60}$$

For each mode p, the quantities inside the large parentheses involve computing sets of cosine- and sine-weighted kernel estimates for all particles i with $\mathbf{x}^{*(i)} \in R$. Computationally this is equivalent to calculating a set of 1D kernel estimates and, thus, the $\mathcal{O}(N)$ Taylor-series algorithm described in [10] can be used to obtain them. Repeatedly using the Taylor-series algorithm for each of the M modes results in $\mathcal{O}(NM)$ computational work overall.

In this variation the definitions of the regions R and S are slightly modified (the rectangular domain D is assumed to be the same). With the Fourier expansion done in x_1, the region R is a vertical strip extending the full height of the domain and having width w_1. Likewise, S is a similar vertical strip, but of width $2h_1 + w_1$. Both are centered horizontally about $x_1 = x_1^R$.

The Fourier–Taylor variation has some unique advantages over the Fourier–Fourier method, one being that the work per particle in 2D scales only as $\mathcal{O}(M)$ as opposed to $\mathcal{O}(M^2)$. This is evident in Fig. 9, which shows that the increase in work for the Fourier–Fourier algorithm as a result of doubling the number of modes is about twice the increase for the Fourier–Taylor algorithm. For values of M used in actual 2D simulations though, the Fourier–Fourier algorithm is currently more cost-effective due to less overhead.

A more important advantage of the Fourier–Taylor variation involves computation of gradients. In SPH gradients are calculated by direct differentiation of the kernel, for example,

$$\frac{\partial}{\partial x_2}\left\langle \rho\left(\mathbf{x}^{*(i)}\right)\right\rangle_{N,\mathbf{h}} = \sum_{\mathbf{x}^{*(n)}\in S} m^{(n)}\hat{K}\left(x_1^{*(i)} - x_1^{*(n)}, h_1\right)\frac{\partial}{\partial x_2}\hat{K}\left(x_2^{*(i)} - x_2^{*(n)}, h_2\right). \tag{61}$$

In the Fourier–Fourier implementation, convergence of relative error with respect to M will therefore be slowest for gradients. With the Fourier–Taylor algorithm this reduction in scaling is ideally eliminated if the Fourier expansion is done on the kernel *not* being differentiated. This behavior is confirmed in Fig. 19 which plots results of a numerical test studying convergence of 2D relative error for the quantity $\langle \partial\rho/\partial x_2\rangle$ using both Fourier–Fourier and Fourier–Taylor implementations. To make the difference in convergence rates more distinguishable, the error in both curves is scaled by M^3. The curve for the Fourier–Fourier implementation is therefore nearly horizontal as expected, while the curve for the Fourier–Taylor implementation continues to decrease with M, the measured slope being about -0.8.

The numerical test, similar to the one performed for Fig. 5, was done using a smoothing length $h = \frac{1}{2}$, and 262,144 particles deterministically positioned in the unit square domain according to a predefined density field. Mean density gradient estimates based on $\partial\hat{K}'(x, h, \ell, M)/\partial x$ were calculated using both the Fourier–Fourier and Fourier–Taylor algorithms, and these were compared to the values obtained using $\partial\hat{K}(x, h)/\partial x$ to obtain the relative errors plotted in the figure. The number of modes M was varied between 4 and 64, while the period was kept fixed at $\ell = 3h$.

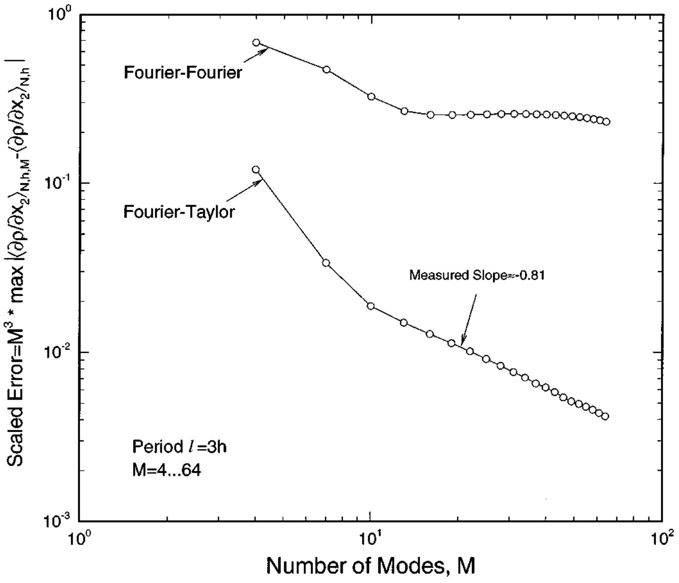


FIG. 19. Relative error in $\langle \partial\rho/\partial x_2\rangle$ for a 2D static test case. Scalings shown for both Fourier–Fourier and Fourier–Taylor implementations.

Figure 19 also shows that for values of M used in simulations ($M \leq 10$, typically), the relative error for the gradient is nearly an order of magnitude less using the Fourier–Taylor implementation as compared to the Fourier–Fourier implementation. In problems where relative error must be kept small, the Fourier–Taylor approach is thus ideal.

8.2. Extension to 3D

Extension of the algorithm to 3D is straightforward, with two possible choices. Still assuming a tensor-product kernel, each of the three 1D kernels can be expanded in a Fourier-series to give an $\mathcal{O}(NM^3)$ method. As in 2D, M need only be finite, provided that the periods in each direction scale with the smoothing lengths. The other choice is to expand two of the three kernels and make use of the $\mathcal{O}(N)$ Taylor-series algorithm. The overall work then scales as $\mathcal{O}(NM^2)$. At present these approaches have been not yet implemented, but no difficulties are foreseen.

8.3. Extension to Isotropic Kernels

As presented in Section 3, the $\mathcal{O}(N)$ algorithm is applicable to tensor-product kernels. For the combined PDF/SPH calculations presented in this paper, this requirement is completely acceptable. In other applications such as astrophysical simulations, such a requirement may not be satisfactory. In fact, virtually all SPH simulations use kernels which are isotropic—they are solely a function of the Euclidean distance between particles. This section explains an approach which extends the $\mathcal{O}(N)$ algorithm to include isotropic kernels.

The key to this approach is determining a proper set of basis functions which is compatible with the separation property, Eq. (23). An isotropic kernel of the form $K(r)$ can be expanded in powers of r using, for example, polynomial sets such as Legendre or Chebyshev polynomials. But the definition of the Euclidean distance r is such that only even powers of r are amenable to separation. This becomes clear through a simple example. Given that the distance in 2D between particles i and n is $r_{i,n} = [(x_i - x_n)^2 + (y_i - y_n)^2]^{1/2}$, consider calculating the pair of quantities

$$S_i^{(1)} = \sum_{n=1}^{N} r_{i,n}, \quad S_i^{(2)} = \sum_{n=1}^{N} r_{i,n}^2 \tag{62}$$

for $i = 1 \cdots N$. The former must be computed using an $\mathcal{O}(N^2)$ direct summation approach, whereas the latter can be expanded and separated into

$$S_i^{(2)} = N\left(x_i^2 + y_i^2\right) - 2x_i \sum_{n=1}^{N} x_n - 2y_i \sum_{n=1}^{N} y_n + \sum_{n=1}^{N} x_n y_n \tag{63}$$

and can, therefore, be obtained in $\mathcal{O}(N)$ time. This is true for all even powers of r. Consequently, if the isotropic kernel is expanded in a finite polynomial series in *even* powers of r,

$$K(r, h) = \sum_{p=0}^{M} a_p r^{2p}, \tag{64}$$

then computation of kernel estimates can be done in $\mathcal{O}(N)$ time. Such expansions readily exist for symmetric kernels such as Eq. (17).

Implementation of this approach is identical to the Fourier-series approach. The kernel expansion is chosen to be valid over a finite interval of length $\ell = \mathcal{O}(h)$, and kernel estimates are sequentially computed in rectangular (or possibly even hexagonal) subregions R surrounded by corresponding support regions S. Relative error between the approximated kernel estimates and the exact kernel estimates will decrease as the number of terms M in the expansion increases, and the rate of convergence will depend on the smoothness of the kernel. The computational work increases quickly, however, as M is made bigger, since the number of terms in the polynomial expansion of r^{2p} grows exponentially with p.

8.4. Parallel Implementation

The Fourier-series algorithm is well suited to implementation on scalable parallel computing architectures using the approach of mesh partitioning [28], also known as domain decomposition. In this approach the flow domain is divided into multiple subdomains, each of which is assigned to a single processor or node. Each node subsequently performs work on only its subset of computational cells and the particles in those cells. In this way both the particle and grid work is divided among the nodes.

To illustrate this in the current context, consider again the example given at the end of Section 3 which lists the sequence of computational steps that are performed to obtain the mean densities in $\mathcal{O}(N)$ time. Assume that the domain has been subdivided into local groups of one or more computational cells and that each group is assigned to one node. The first step, Eq. (28), involves computing sums within computational cells. Since this only involves data local to each cell, each node can perform this work without having to communicate with other nodes. The second step, Eq. (29), is less intensive computationally than the first, but it does require communication between nodes. However, the amount of communicated data is small since it only consists of cell quantities. The third and final step again involves only local data and can, therefore, be performed independently by each node. Thus, computation of mean fields using this parallel implementation of the algorithm is expected to be quite efficient overall.

Additional communication would be required between time steps as the particles cross node boundaries, and this is common to all parallel implementations based on mesh partitioning. Although this communication is expected to dominate other communication, it can be significantly reduced by aligning the node boundaries with the mean-flow streamlines.

9. CONCLUSIONS

This paper has presented a PDF-based Monte Carlo particle method for simulating two-dimensional compressible turbulent flows. While most PDF methods are implemented using a particle-mesh approach, the particle method described here is unique in that it incorporates the kernel estimation techniques of SPH to obtain a grid-free implementation. The main benefit of using SPH is that it allows a general and robust treatment of the mean pressure within the PDF framework; using kernel estimates, the pressure (and all other required quantities) is obtained directly from the particles.

This combination of SPH with the PDF method has also introduced computational challenges. These have been successfully addressed, however, with the development of an efficient algorithm for computing two-dimensional SPH kernel estimates. The algorithm described in this paper is based on Fourier-series expansions of the SPH kernel, and for a simulation with N particles its computational work scales purely as $\mathcal{O}(N)$. Consequently, simulations for a wide variety of flows can be performed in reasonable time. The algorithm is also shown to be easily extendible to three dimensions and applicable to isotropic kernels. Furthermore, its implementation on scalable parallel computing architectures is shown to be straightforward and is expected to be efficient.

Two types of numerical errors introduced by the finite series expansion have been presented and studied. Relative error, a measure of how well kernel estimates obtained using the Fourier-series algorithm compare to the original SPH estimates, was shown to scale as $(Mh/\ell)^{-4}$ for the kernel used in this study. Spatial error, the deterministic error caused by using a finite smoothing length, was shown to converge for the new algorithm for all M, provided $\ell = \mathcal{O}(h)$. In both cases, the errors scale as expected and can be kept negligibly small using modest computing requirements.

The particle method has been used to simulate a wide variety of 2D flows, including a stationary self-similar plane wake, hydrostatic flows under the influence of body forces, and an unsteady flow featuring compression/expansion waves and a pair of decaying vortices. The range in scope of these problems demonstrates the robustness of the method. Furthermore, the plane wake problem shows that the method can be used to accurately simulate a flow of practical importance. In the wake simulation, the turbulent frequency varies with downstream position and is computed from the particles. Beginning from an arbitrary inlet condition, the mean velocity and Reynolds stresses attain self-similar profiles which compare favorably with experimental data.

ACKNOWLEDGMENTS

Financial support for this work was partially provided by NASA Lewis Research Center through Grant NGT-50842 as part of the NASA Graduate Student Researchers Program. Computing resources were provided by the Advanced Computational Concepts Laboratory (ACCL) at NASA Lewis Research Center and by the Cornell Theory Center. The latter receives major funding from the National Science Foundation (NSF) and New York State.

REFERENCES

1. S. B. Pope, Computations of turbulent combustion: Progress and challenges, in *23rd International Symposium on Combustion* (The Combustion Institute, Pittsburgh, 1990), p. 591.
2. S. B. Pope, PDF methods for turbulent reactive flows, *Prog. Energy Combust. Sci.* **11**, 119 (1985).
3. S. B. Pope, Lagrangian PDF methods for turbulent flows, *Annu. Rev. Fluid Mech.* **26**, 23 (1994).
4. B. J. Delarue and S. B. Pope, Application of PDF methods to compressible turbulent flows, submitted to *Phys. Fluids.*
5. B. J. Delarue and S. B. Pope, Stochastic model for compressible turbulence, *Bull. Am. Phys. Soc.* **39**, 1860 (1994).
6. A. T. Hsu, Y.-L. P. Tsai, and M. S. Raju, Probability density function approach for compressible turbulent reacting flows, *AIAA J.* **7**, 1407 (1994).
7. M. S. Anand, S. B. Pope, and H. C. Mongia, PDF calculations for swirling flows, AIAA Paper 93-0106.
8. M. S. Anand, S. B. Pope, and H. C. Mongia, Pressure algorithm for elliptic flow calculations with the PDF method, in *CFD Symposium on Aeropropulsion* (NASA Lewis Research Center, Cleveland, 1990).

9. S. B. Pope, *Position, Velocity and Pressure Correction Algorithm for Particle Method Solution of the PDF Transport Equations*, FDA Report 95-06, Cornell University, 1995.
10. W. C. Welton and S. B. Pope, PDF model calculations of compressible turbulent flows using smoothed particle hydrodynamics, *J. Comput. Phys.* **134**, 150 (1997).
11. J. J. Monaghan, Smoothed particle hydrodynamics, *Annu. Rev. Astron. Astrophys.* **30**, 543 (1992).
12. D. C. Haworth and S. H. El Tahry, Probability density function approach for multidimensional turbulent flow calculations with application to in-cylinder flows in reciprocating engines, *AIAA J.* **29**, 208 (1991).
13. S. M. Correa and S. B. Pope, Comparison of a Monte Carlo PDF/finite-volume mean flow model with bluff-body Raman data, in *24th International Symposium on Combustion* (The Combustion Institute, Pittsburgh, 1992), p. 279.
14. G. C. Chang, *A Monte Carlo PDF/All-Speed Finite-Volume Study of Turbulent Flames*, Ph.D. thesis, Cornell University, 1995.
15. L. Barnes and P. Hut, A hierarchial $O(N \log N)$ force-calculation algorithm, *Nature* **324**, 446 (1986).
16. L. Hernquist and N. Katz, TREESPH: A unification of SPH with the hierarchial tree method, *Astrophys. J. Suppl. Ser.* **70**, 419 (1989).
17. D. C. Haworth and S. B. Pope, A pdf modeling study of self-similar turbulent free shear flows, *Phys. Fluids* **30**, 1026 (1987).
18. M. S. Anand and S. B. Pope, Diffusion behind a line source in grid turbulence, in *Turbulent Shear Flows*, Vol. 4 (Springer-Verlag, Berlin, 1985), p. 46.
19. J. J. Monaghan, Why particle methods work, *SIAM J. Sci. Stat. Comput.* **3**, 422 (1982).
20. M. Schüssler and D. Schmitt, Comments on smoothed particle hydrodynamics, *Astron. Astrophys.* **97**, 373 (1981).
21. S. B. Pope, Particle method for turbulent flows: Integration of stochastic model equations, *J. Comput. Phys.* **117**, 332 (1995).
22. C. Hirsch, *Numerical Computation of Internal and External Flows*, Vol. 2 (Wiley, New York, 1990).
23. A. Jameson, Steady-state solution of the Euler equations for transonic flow, in *Transonic, Shock, and Multi-dimensional Flows: Advances in Scientific Computing* (Academic Press, New York, 1982), p. 37.
24. G. N. Mil'shtein, A method of second-order accuracy integration of stochastic differential equations, *Theory Probab. Appl.* **23**, 396 (1978).
25. I. Wygnanski, F. Champagne, and B. Marasli, On the large-scale structures in two-dimensional, small-deficit, turbulent wakes, *J. Fluid Mech.* **168**, 31 (1986).
26. A. A. Townsend, The fully developed turbulent wake of a circular cylinder, *Aust. J. Sci. Res. Ser. A* **2**, 451 (1949).
27. J. Xu and S. B. Pope, *Sources of Bias in Particle-Mesh Methods for PDF Models for Turbulent Flows*, FDA Report 97-01, Cornell University, 1997.
28. S. B. Pope, PDF/Monte Carlo methods for turbulent combustion and their implementation on parallel computers, in *Turbulence and Molecular Processes in Combustion*, edited by T. Takeno (Elsevier, Amsterdam, 1993).