A comparative study of Lagrangian Methods using axisymmetric and deforming blobs.[*]

Louis F. Rossi[†]

Technical Report No. 2003-09

[*]This work was supported by National Science Foundation grant DMS-9971800.

[†]Copyright 2003 Louis F. Rossi. Department of Mathematical Sciences, University of Delaware. Newark, DE 19716, USA (rossi@math.udel.edu)

A COMPARATIVE STUDY OF LAGRANGIAN METHODS USING AXISYMMETRIC AND DEFORMING BLOBS. [*]

LOUIS F. ROSSI[†]

Key words. Convection-diffusion, particle methods, computational fluid dynamics, deforming blobs

AMS subject classifications. 35Q30, 41A25, 65M12, 65M60, 76D05

Abstract. This paper presents results from a head-to-head comparison of two Lagrangian methods for solutions to the two-dimensional, incompressible convection-diffusion equations. The first Lagrangian method is an axisymmetric core spreading method using Gaussian basis functions. The second method uses deforming elliptical Gaussian basis functions. Previous results show that the first method has second-order spatial accuracy and the second method has fourth-order spatial accuracy. However, the deforming basis functions require more computational effort per element, so this paper examines computational performance as well as overall accuracy. The test problem is the deformation and diffusion of ellipsoidal distribution of scalar with an underlying flow field that has closed circular streamlines. The test suite includes moderate, high and infinite Peclét number problems. The results indicate that the performance tradeoff for the sample flow calculation occur at modest problem sizes, and that the fourth-order method offers distinct advantages as a general approach for challenging problems.

1. Introduction.

Lagrangian schemes for solving convection-diffusion equations are naturally adaptive in the sense that information about the computed field closely follows physical path lines. These schemes are a natural reduction from vortex methods for Navier-Stokes and other nonlinear schemes, and differ only in the sense that the velocity is specified rather than determined from the computed field. Since these methods express the computed field as a linear combination of basis functions, sometimes called blobs, that move with the local flow velocity, one can argue that computational resources are dedicated only where the computed field is substantial and nowhere else. To boost the efficiency of Lagrangian methods, many investigators have explored anisotropic and dynamically deforming computational elements as a way to use fewer elements in complex flow geometries. For instance, Marshall & Grant, Meiburg and Teng have specified anisotropic vortex elements to develop more efficient means of capturing boundary layers and mixing layers [9, 10, 18]. Furthermore, it makes sense that improvements to the accuracy of moving, localized basis functions should come through local corrections in the shape of the basis function. Teng, Leonard, Moeleker, and Rossi have developed methods that permit the computational elements to deform dynamically based on linearized deviations in the flow velocity [8, 11, 15, 19, 20]. All these papers report improved accuracy though not necessarily an order of spatial convergence beyond the anticipated second-order and not necessarily improved computational performance. Since anisotropic and deforming basis functions require more computational resources than simple anisotropic elements, one would hope that the more sophisticated scheme will have substantial improvements in accuracy to justify the additional cost. To the contrary, Moeleker & Leonard report that the increased computational effort over calculations with axisymmetric elements for the convection-diffusion equations does not justify the cost of using dynamically deforming elements [11]. It is important to note that Moeleker & Leonard use a

[*]This work was supported by National Science Foundation grant DMS-9971800.

[†]Copyright 2003 Louis F. Rossi. Department of Mathematical Sciences, University of Delaware. Newark, DE 19716, USA (`rossi@math.udel.edu`)

method that uses velocity data measured at the basis function centroid to control the motion and deformation of the computational elements. Recently, Rossi determined that the general category of method used by Moeleker & Leonard has second-order spatial correctness so that the deforming method and axisymmetric methods have the same order of convergence. However, this same analysis indicates that a curvature correction to the velocity field coupled with the adaptive deformations achieves fourth order spatial accuracy [16]. Tests using deforming methods with and without the correction conclusively show that the theoretical convergence rates are indicative of the method's actual convergence rate, and that the method using curvature corrections outperforms the same method without the corrections. What remains to be studied is whether it is better to use the more expedient axisymmetric elements or the more accurate but computationally intensive deforming basis functions for reasonably challenging problems. This paper makes a head-to-head comparison between Lagrangian methods with axisymmetric blobs and high-order deforming blobs to understand flow-dependent performance tradeoffs between the efficiency and accuracy of the two techniques when solving problems that are challenging on current desktop machines.

This paper focuses on solutions to the two-dimensional, incompressible, convection-diffusion equations,

$$\partial_t \rho + (\vec{u} \cdot \nabla)\rho = \frac{1}{\mathrm{Pe}} \nabla^2 \rho, \tag{1.1a}$$

$$\nabla \cdot \vec{u} = 0, \tag{1.1b}$$

on the unbounded plane where ρ is a passive scalar quantity that moves and diffuses with the flow. The $\vec{u}$ is a known fluid velocity field and Pe is the Peclét number, a dimensionless quantity expressing the ratio of convective to diffusive effects in the flow. Both the method using axisymmetric blobs and the method using deforming blobs are known as *core-spreading methods* because the width of each computational element grows with time to capture the diffusive term. Core-spreading methods are known to be inconsistent except when $\mathrm{Pe} = \infty$ because the growth of the core is governed by the Peclét number (or Reynolds number for vortex methods) and cannot be reduced below a nontrivial value. C. Greengard discusses the matter in some detail in [2]. Regarding practical issues when computing flows, Kida et. al. have since shown that there are important applications where these spatial errors are considerably smaller than other sources of numerical error so meaningful calculations can be performed with these inconsistent techniques [3, 4, 5]. Essentially, the inconsistent formulation is useful for applications where the timescale of interest is much smaller than the growth in core size. If one strives for consistency, there are a number of ways to check the growth of core widths including remeshing onto a regular array elements (see [1] for a survey), as well as adaptive refinement and merging [13, 14], but this is secondary to considerations of the accuracy and performance of the underlying method.

This comparative study has two goals. First, we will determine roughly where the high spatial order method outperforms the low order method. It worth knowing whether or not the method has concrete advantages over lower order but more efficient methods for reasonable applications on current computing platforms. Second, we use this study to highlight quantitative and qualitative differences in the convergent behavior of the two methods. While both methods are inconsistent methods, using deforming elements extends the range of applicability before one must apply corrections to the method. In other words, both algorithms are naturally adaptive in the sense that computational effort is expended only where the field is non-trivial, but the scheme using deforming elements is more adaptive than the method using rigid elements in a concrete sense that will be explored in section §4.

These goals are similar but not identical to Legras & Dritschel's comparison of contour surgery and psuedo-spectral methods [7]. Both Legras & Dritschel's study and this work represent an effort to understand the strengths and weaknesses of a new method by comparing it to an established method. This study has a more intense focus on the computational performance and efficiency of the methods in addition to their convergence toward an exact solution. In Legras & Dritschel's study, they examine a complex flow with the two numerical schemes and compare the results to one another. In this paper, we test two similar Lagrangian methods on a flow problem using three different Peclét numbers by comparing approximate solutions to a highly resolved reference solution. Where the Lagrangian methods are general algorithms, the reference solution is calculated using a spectral scheme that is customized for this particular problem geometry. Legras & Dritschel's study compares quantities such as contour moments and area increments to measure the relative proximity of solutions. This paper will present the root-mean-squared distances (L_2) between the two methods that are being tested and the reference solution. Legras & Dritschel use two levels of refinement, coarse and fine. This paper examines solutions over different flow parameters at six distinct resolutions to establish convergence properties and performance trends.

This paper is organized as follows. This section (§1) describes the background of the problem and provides a review of relevant works by other investigators. In §2, we will present the two methods under examination and discuss the time integration scheme used by both. In §3, we define the fluid flow problems in the test suite and the practicalities of the error measurements used in this paper. The results of these tests are in §4. Finally, §5 contains summary remarks and future directions in this work.

2. The semi-discrete systems and temporal integration.

This study compares two methods, one using spreading Gaussian blobs and the other using deforming elliptical Gaussian blobs. As noted before, both methods are inconsistent when $\mathrm{Pe} < \infty$ but can be corrected with adaptive refinement or remeshing. The spreading term arises when we capture the diffusive term in teh same discrete dynamical system used to capture the convective terms in (1.1). Some popular alternatives for capturing the diffusive term include field redistribution and random walks, but these suffer from a lack of adaptivity in the former, and low efficiency and accuracy in the latter. Both require operator splitting with the associated splitting errors. The advantages of core spreading methods are that they are fully deterministic, naturally adaptive and the particle dynamics capture the underlying partial differential equation without splitting the operator.

In this paper, we define $\widehat{\rho}$ to be the approximation using a linear combination of moving basis functions:

$$\widehat{\rho} = \sum_{i=0}^{N} \gamma_i \phi(\vec{x} - \vec{x}_i, \ldots) \tag{2.1}$$

The shape of the basis function is determined by ϕ, and most often is chosen to be axisymmetric. For instance, if one is trying to solve the Navier-Stokes equations where $\vec{u}$ and ρ are related via a Biot-Savart integral, then radial symmetry makes

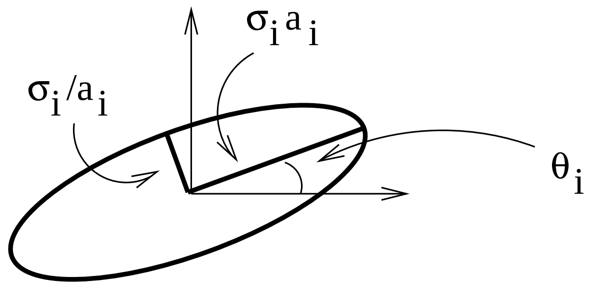

FIG. 2.1. *A schematic diagram of an elliptical Gaussian basis function as defined in (2.3a). The ellipse is representative of a single level set.*

the velocity integration much simpler. The first of the two methods, which we shall refer to as the Axisymmetric Blob Method, is the simplest available core spreading method.

$$\phi(\vec{x}, t; \vec{x}_i, \sigma_i) = \frac{1}{4\pi\sigma_i^2} \exp\left[-\frac{|\vec{x}-\vec{x}_i|^2}{4\sigma_i^2}\right], \tag{2.2a}$$

$$\frac{d\vec{x}_i}{dt} = \widehat{u}_i, \tag{2.2b}$$

$$\frac{d\sigma_i^2}{dt} = \frac{1}{\mathrm{Pe}}. \tag{2.2c}$$

where $\vec{x} = \begin{bmatrix} x \\ y \end{bmatrix}$. This method can use centroid velocity data ($\widehat{u}_i = \vec{u}(\vec{x}_i)$) with no penalty to its spatial accuracy. We shall refer to the second method in this study as the Deforming Blob Method.

$$\phi(\vec{x}; \vec{x}_i, \sigma_i, a_i, \theta_i) = \frac{1}{4\pi\sigma_i^2} \exp\left\{-\frac{|A_{\theta_i,a_i}(\vec{x}-\vec{x}_i)|^2}{4\sigma_i^2}\right\}, \tag{2.3a}$$

$$\frac{d}{dt}\vec{x}_i = \widehat{u}_i, \tag{2.3b}$$

$$\frac{d}{dt}(\sigma_i^2) = \frac{1}{2\mathrm{Pe}}(a_i^2 + a_i^{-2}), \tag{2.3c}$$

$$\frac{d}{dt}(a_i^2) = 2[d_{11}(c_i^2 - s_i^2) + (d_{12}+d_{21})s_i c_i]a_i^2 + \frac{1}{2\sigma^2\mathrm{Pe}}(1 - a_i^4), \tag{2.3d}$$

$$\frac{d}{dt}\theta_i = \frac{d_{21}-d_{12}}{2} + \left[\frac{d_{21}+d_{12}}{2}(s_i^2 - c_i^2) + 2d_{11}s_i c_i\right]\frac{(a_i^{-2}+a_i^2)}{(a_i^{-2}-a_i^2)}. \tag{2.3e}$$

where d_{ij} are the constituent elements of $D\vec{u}$, the matrix of partial derivatives of the velocity field $\vec{u}$, $c_i = \cos(\theta_i)$, $s_i = \sin(\theta_i)$, and

$$A_{\theta,a} = \begin{bmatrix} \cos\theta/a & \sin\theta/a \\ -a\sin\theta & a\cos\theta \end{bmatrix}. \tag{2.4}$$

So, in addition to the usual parameters, these basis functions have an orientation (θ_i) and an aspect ratio (a_i^2) as shown in Figure 2.1. The system (2.3) is larger and more complex than (2.2), and so there is a computational penalty for using the Deforming Blob Method. Also, to have fourth order spatial accuracy [16], the velocity field is adjusted away from centroid velocity data as

$$\widehat{u}_i = \vec{u}(\vec{x}_i) + \sigma_i^2 \left(\vec{u}_{xx}(\vec{x}_i) M_{xx} + 2\vec{u}_{xy}(\vec{x}_i) M_{xy} + \vec{u}_{yy}(\vec{x}_i) M_{yy} \right), \tag{2.5}$$

where

$$M_{xx} = c_i^2 a_i^2 + s_i^2 / a_i^2, \tag{2.6a}$$

$$M_{xy} = c_i s_i (a_i^2 - a_i^{-2}), \tag{2.6b}$$

$$M_{yy} = c_i^2 / a_i^2 + s_i^2 a_i^2. \tag{2.6c}$$

There are no known advantages to curvature corrections in $D\vec{u}$ when using this method, though this was explored somewhat in [16].

When we approximate solutions to (1.1) using one of these schemes, we must choose a temporal integrator. For a problem where the velocity field is known and easy to compute, a Runga-Kutta method might be appropriate. However, there are many nonlinear problems where Lagrangian methods would be desirable, and in these cases, the velocity field might be expensive to compute. For instance, in vortex methods, the velocity field depends upon $\widehat{\rho}$ though a Biot-Savart integral. This is the most expensive part of the computation, even if one is using fast summation. To be general and open to the reality of extended applications, we perform the study with an explicit multistep integrator.

We use a third order Adams-Bashforth multistep integrator for the trajectories in this comparative study. Also, we use a composite scheme because the deformation of the elements can occur on a timescale that is much faster than changes in the flow field. The deformation of each element occurs on half-steps because we would like to remain flexible and achieve second-order accuracy if the velocity were to depend upon ρ (even though it does not in this paper). Thus, the time-stepping algorithm is as follows:

1. Integrate either (2.2b) for the Axisymmetric Blob Method or (2.3b) for the Deforming Blob Method to move particles forward one half a time step using an explicit stepper. Evaluate the velocity field.
2. Integrate (2.2c) for the Axisymmetric Blob Method or (2.3c-2.3e) for the Deforming Blob Method to deform basis functions for a full timestep. *Since (2.2c) has a simple analytic solution, numerical integration is not necessary, and we find that the axisymmetric implementation runs about five times faster because of this feature. Thus, there is a substantial performance advantage to using the lower order method.*[1] Evaluate the velocity field.
3. Integrate (2.2b) or (2.3b) to move particles forward one more half time step. Evaluate the velocity field.
4. To continue, return to step 1.

It can be shown that this simple splitting technique is second-order correct in time though it would be worthwhile to investigate whether or not there is a high order multistep method for the entire system. The algorithm is implemented in a modular program, and the codes are identical except for the evolution equations governing the computational elements. That is, input/output, the looping structures and all other routine tasks are absolutely identical in both methods.

[1]In the case of (2.3c-2.3e), an adaptive fourth order Runge-Kutta algorithm is used with half-stepping for error control. We experimented with an adaptive Cash-Karp error control algorithm as outlined in [12], but found this to be less efficient than a simple half-stepping for this particular application.

3. The test problem. To examine the performance of the two methods, each will approximate the solution of (1.1) with an underlying differentially rotating flow given by the streamfunction

$$\psi = \frac{\pi}{4} r^2 + \delta(r^2 + c_1 r^4 + c_2 r^6), \tag{3.1a}$$

$$\vec{u} = \begin{bmatrix} \frac{\partial \psi}{\partial y} \\ -\frac{\partial \psi}{\partial x} \end{bmatrix} \tag{3.1b}$$

where $\delta = 3/10$, $c_1 = -7/6$ and $c_2 = 1/10$ on an unbounded domain. This is the same flow field used in [16], but these tests will use different initial data. Where we applied axisymmetric initial data in previous work so that an exact solution could be used for convergence studies, in this study we will use a normalized, anisotropic distribution of scalar,

$$\rho(\vec{x}, 0) = \frac{32}{\pi} e^{-32\left(x^2/8 + 8y^2\right)}, \tag{3.2}$$

so that, in the absence of dissipation, fine filaments will be generated after a small number of rotations as seen in Figure 3.1. We select this sample problem because one uses Lagrangian methods when the quantity of interest, ρ, occupies a fraction of the computational domain at any given time during the simulation. While axisymmetric structures are useful for a convergence study, it is unlikely anyone would choose a Lagrangian method if the computed field was expected the fill the domain of interest. Instead, a performance measurement should involve a distribution that is more typical for the method being tested.

With this flow structure, the turnover time near the origin is about 3 but the turnover time near the edge of the structure is roughly $\frac{9}{2}$. The computation is performed until $T = 9$ so that the inside of the initial distribution is twisted a full revolution relative to the edges as seen in Figure 3.1. To test the two Lagrangian methods, the calculation is performed with $\mathrm{Pe} = 10^4$, $\mathrm{Pe} = 10^5$ and $\mathrm{Pe} = \infty$. The initial data for both methods is identical. Since the initial data has infinite support, we initially arrange computational elements over the domain

$$8(x^2/8 + 8y^2) \leq 1, \tag{3.3}$$

beyond which ρ has decayed to less than 2% of its initial maximum value. The strength of each element is determined using an exact deregularization procedure, as discussed in [17], so that the growth of spatial errors in solving (1.1) are much greater than the growth of initial errors. Isotropic particles are arranged on the domain (3.3) in a regular square lattice of width $\frac{\sigma_0}{2}$ where σ_0 is the initial core width for the trial simulation. Both methods were tested in trials with initial core widths of 8×10^{-4}, 4×10^{-4}, 2×10^{-4}, 10^{-4}, 5×10^{-5} and 2.5×10^{-5}.

A high resolution spectral calculation will serve as a reference ("exact") solution for the convergence study. The spectral code is implemented in Matlab using scripts from Trefethan's monograph [21]. The spectral code takes advantage of the fact that we know the domain of interest apriori, so we can compute spatial derivatives spectrally on the unit disk (see chapter 11 of the monograph) using Chebyshev interpolation in the radial direction and trigonometric interpolation in the azimuthal direction. The spectral code used in this paper modifies Trefethan's Matlab program to take advantage of sparsity in the 2D differentiation matrix. Time stepping is performed using third order Adams-Bashforth with timesteps of either 10^{-3}, 6×10^{-5}

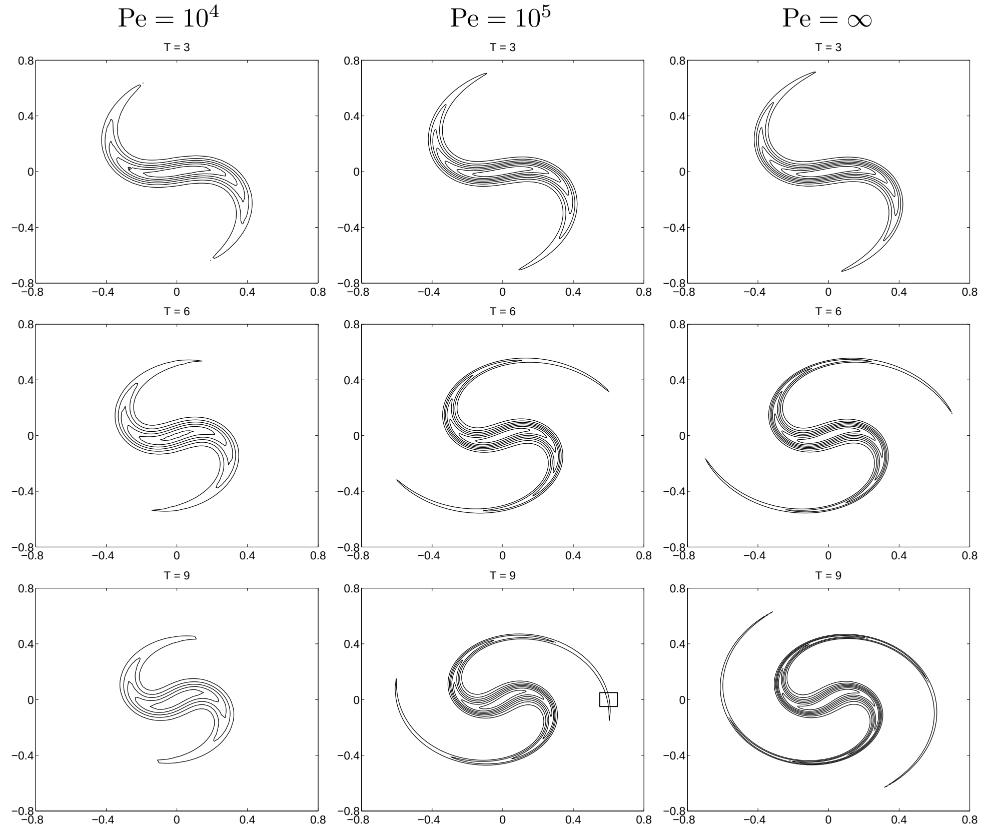

FIG. 3.1. *Solutions for* Pe $= 10^4$ *(left),* Pe $= 10^5$ *(center) and* Pe $= \infty$ *(right). The contour increment is 4 units of* ρ *per division. The differential rotation of the underlying flow intensifies gradients while diffusion suppresses gradients. The long, slender filaments at high Peclét number pose a challenge to any numerical method. The area of detail indicated at the bottom center is expanded in Figure 4.2.*

for Pe $= 10^4$ and 10^5, respectively, depending upon the stability requirements of the spectral differentiation matrix. Refinement studies indicate that all computations are temporally resolved. While the purpose of this paper is to examine the tradeoffs between the Axisymmetric Blob Method and the Deforming Blob Method, it is worthwhile to discuss three notable distinctions between the spectral reference solution and the Lagrangian solutions.

- The most important difference is that we should achieve spectral accuracy in space from the spectral code since the exact solution is smooth for all finite time. That is, we expect the spatial accuracy to scale like C^{-N} where N is the number of modes used in the computation. The Lagrangian methods are second- $O(\sigma^2)$ and fourth-order $O(\sigma^4)$ in space, respectively. However, for very high Peclét numbers, the scalar filaments can become quite sharp, and the resulting large gradients make the spectral convergence rate very slow.
- The spectral code is designed to take advantage of the simple geometry of the test problem whereas the Lagrangian code is general and is applicable to any flow geometry. For instance, if we were to superpose a straining flow such as $\psi_2 = Cxy$ into the flow field, we could directly apply the same

Lagrangian code whereas the spectral code would need to be modified for a new computational domain.
- The spectral method operates on a finite domain where $\widehat{\rho} = 0$ at $r = 1$ in polar coordinates whereas the Lagrangian code operates on an unbounded domain and the choice of basis function automatically satisfies $\widehat{\rho} \to 0$ as $r \to \infty$. Since (3.2) decays rapidly and the Peclét number is high, solutions from either method can be compared to one another.

Solutions to all calculations are compared on a uniform grid that is sufficiently fine to resolve differences down to four decimal places. For Pe $= 10^4$, a 100×100 is sufficient, but for Pe $= 10^5$ and Pe $= \infty$, a 400×400 mesh is used. For the Lagrangian computations, $\widehat{\rho}$ is computed directly at each mesh point. For the spectral reference solution, we acquire regularly spaced data using the corresponding spectral projection. To measure the difference between two solutions, u and v, we use an l_2 approximation to the L_2 norm:

$$\|u - v\|_2 = \sqrt{\sum_{i,j} (v_{i,j} - v_{i,j})^2 \Delta x \Delta y}, \tag{3.4}$$

where the sum is over all mesh points, and Δx and Δy are the width and height of the regular adjoint grid cells. Thus, the error in a test calculation is the l_2 difference between the test solution and the reference solution.

4. Results. It is no surprise that both Lagrangian methods achieve the predicted spatial convergence rate as seen in Figure 4.1 and reported in [16] because the only the flow parameters and initial data have been changed. Examining (2.2) and (2.3), we see that the core sizes for the deforming elements will always be at least as large as those of the axisymmetric system. In Figure 4.1, we see that for the same initial widths, the deforming elements grow substantially larger. Yet, despite having wider elements, the Deforming Blob scheme realizes better results because elements conform to the local flow geometry as seen in Figure 4.2. If one extrapolates to a hypothetical situation where the initial widths are zero, one can see that the final state will correspond to the Deforming Blob Method having wider elements, but a more accurate solution. In other words, while both techniques are inconsistent if left uncorrected by remeshing or adaptive refinement techniques, the Deforming Blob Method suffers less for it.

To measure the performance of both methods, the overall precision of the computation can be seen as a function of computational effort expended to achieve the resulting accuracy. Typically, the deforming elements require five times the computational effort of an axisymmetric blob because there are more degrees of freedom and the evolution of σ^2 is coupled to a^2 so neither can be solved analytically as is the case for the axisymmetric elements. In Figure 4.1, we see that the tradeoff between complexity and accuracy occurs even at small problem sizes of about 10,000 (axisymmetric) elements at Pe $= 10^4$, and the improvements are realized at increasingly large problem sizes as we increase the Peclét number. This does not correspond to a loss of performance at large Pe because accurate solutions to larger Pe flow will require a larger number of computational elements to resolve it. Since it is not unusual for high performance Lagrangian computations to use millions of computational elements, these results indicate that the use of deforming elements would benefit many investigations, should lead to significant improvements in accuracy, and will make aggressive calculations more accessible with modest computational resources.

As a final note, it is also meaningful to report upon the spectral method used to generate reference solutions. Though the references themselves are not the focal point

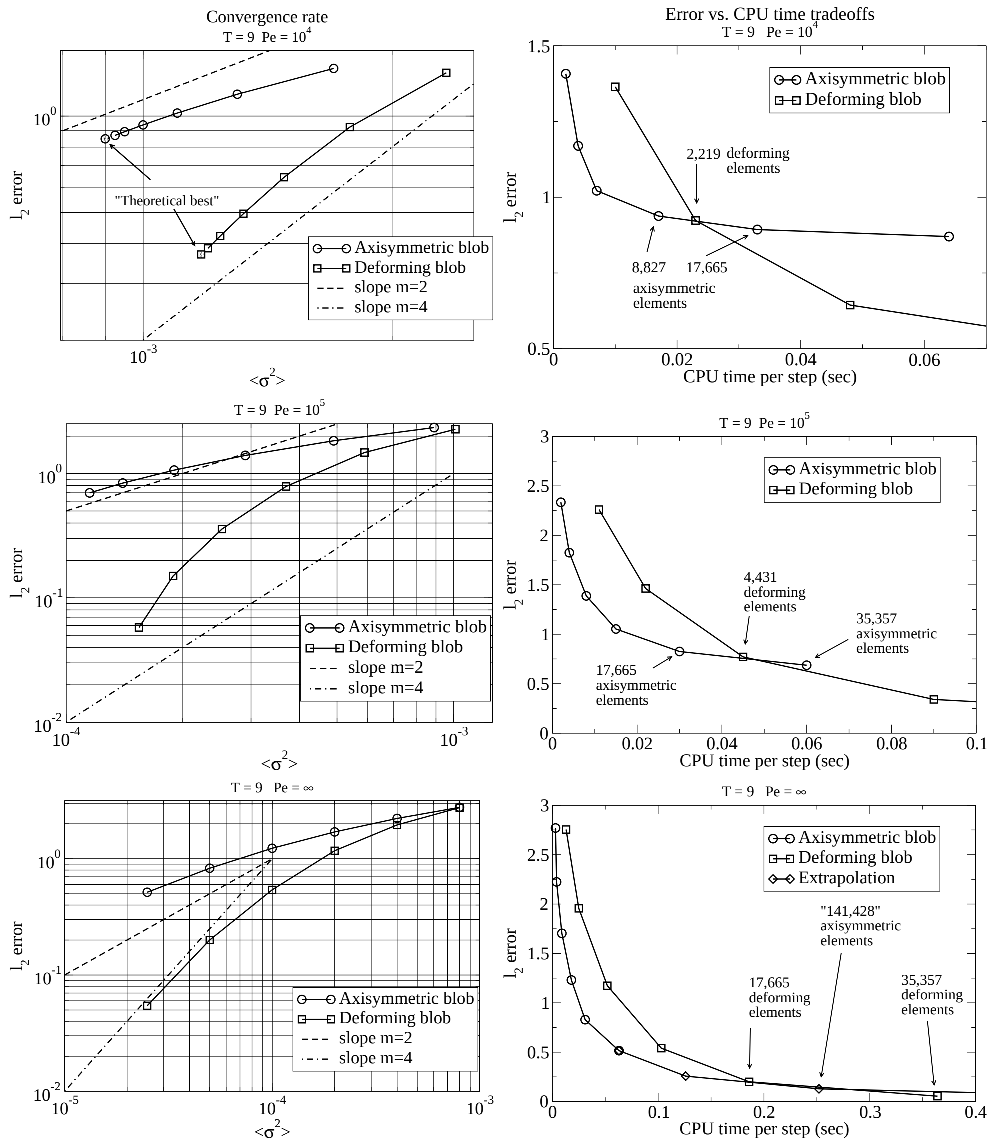

FIG. 4.1. *Convergence rates using logarithmic scales (left) and performance data (right) for test problems using the Axisymmetric Blob and Deforming Blob Methods. Theoretical convergence rates are achieved for both Lagrangian methods, and the Deforming Blob Method works better. In the absence of remeshing or adaptive refinement, both methods are inconsistent meaning that $\langle\sigma^2\rangle$ has a lower bound. The "theoretical best" value is extrapolated back to a position where initial core widths would be zero for the $\mathrm{Pe} = 10^4$ calculation. By examining the accuracy gained as a function of ticks of the CPU clock, we can measure the tradeoff between the simpler low-order method and the more complex high-order method. One can see that the trade-off depends upon the Peclét number.*

of the paper, the efforts required generate them merit some attention. At $\mathrm{Pe} = 10^4$, the spectral computation ran with about 100 modes azimuthally and radially in less than two hours to produce a reference solution that was well below the theoretical best solution that could be obtained with either Lagrangian method. Like the Lagrangian methods that were being tested, the spectral algorithm struggled with computations

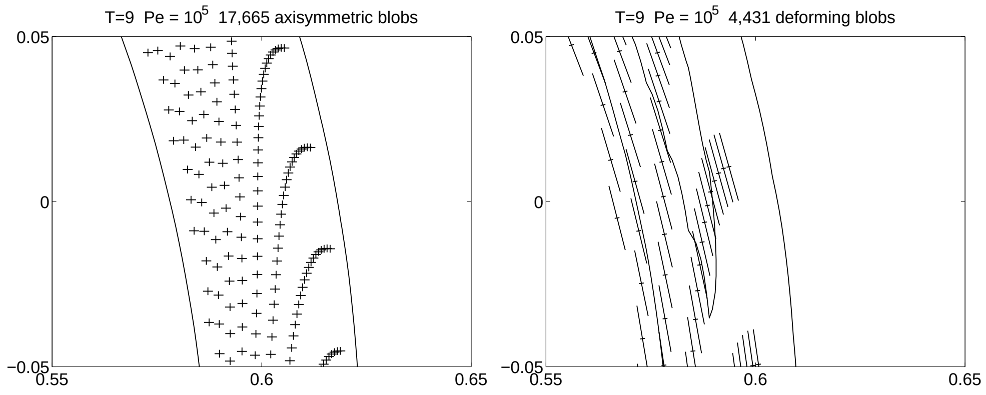

FIG. 4.2. *Sample particle positions using the Axisymmetric (left) and Deforming Blob Methods (right). Particle positions are indicated by oriented X's whose arms correspond to $\sigma a/10$ and $\sigma/a/10$. Scalar concentrations are indicated by the solid contour line. The area of detail is indicated in Figure 3.1 in the bottom center figure. The axisymmetric calculation uses more blobs to achieve slightly less accurate results than the deforming computation on the right. The deforming elements are more than twice as wide, but they naturally conform to the geometry of the computed field.*

at Pe $= 10^5$ and Pe $= \infty$. The spectral reference solution at Pe $= 10^5$ required more 250 modes azimuthally and radially and needed roughly 28 days of CPU time. Part of the problem is that the spatial resolution required a small time step of 6×10^{-5} for the time-stepping to be stable. Before being critical about this performance, one should consider that the Lagrangian method could never reach this accuracy at all without remeshing or refinement. Also, a crafty spectral guru would know of some tricks to save memory and clock cycles, or do a better job avoid stability boundaries. For instance, one might remap the domain to allocate more resolution away from the outer boundary and possibly dedicate more nodes closer to the filaments (see [6] for an example). When Pe $= \infty$, the Lagrangian methods are consistent, and the standard spectral scheme is prohibitively expensive both in memory and time. The deforming Lagrangian method was used to generate its own reference solution. Even with close to 150,000 particles, it ran in three and a half hours, and so was the natural choice for generating a reference solution.

5. Conclusions.

A number of common themes emerge from this quantitative comparison of low and high order Lagrangian method. The general conclusion is that there are significant performance advantages in using the Deforming Blob Method over the Axisymmetric Blob Method for problems of general interest that are accessible with currently available computing devices. The more specific findings are:

- The performance tradeoff depends upon the Peclét number of the problem, so the tradeoffs occur at larger problem sizes when diffusion is less dominant. The other side of this result is that one might be better off using methods with axisymmetric blobs if the resolution requirements are low.
- While both the Axisymmetric and Deforming Blob Methods are inconsistent unless one applies a correction method, the Deforming Blob Method has a better limiting accuracy. Therefore, one can argue that the range of applicability of the Deforming Blob Method is greater than the Axisymmetric Blob Method, and the Deforming Blob Method requires fewer corrections if one desires consistency.

- Both Lagrangian methods compare very favorably to the reference spectral scheme at high Peclét numbers, and this really highlights the advantage of natural adaptivity in Lagrangian schemes. As a general tool for fluid flows, the Lagrangian algorithm dedicates resources more effectively where the field is substantial. This is not intended to criticize the use of spectral methods for high Peclét number flows, but rather to point out that it might require some apriori knowledge of the solution to tune the spectral scheme. The results in this paper suggest that these Lagrangian methods are high quality general tools for flow problems where the field of interest occupies a fraction of the domain of interest.

These results point toward closely related projects. To go beyond the convection-diffusion equations, one might solve the nonlinear vorticity equations where ρ would represent the vorticity of a fluid flow, but one must extract the velocity field from $\widehat{\rho}$. Some progress in this area has already been made by determining a fourth order asymptotic expression for the Biot-Savart integral of an elliptical Gaussian [15]. Along different lines, one might examine the three-dimensional convection-diffusion equations using deforming ellipsoidal Gaussian blobs. Many of the ideas and techniques in this paper and in [16] would apply. Finally, most of the work on remeshing and adaptive refinement has focused on axisymmetric elements. Generalizations to these techniques for highly anisotropic blobs should be explored to make methods like these fully consistent in flow calculations with finite diffusivity.

6. Acknowledgements. The author would like to thank Prof. Toby Driscoll at the University of Delaware for his helpful advise on the practicalities of implementing the spectral code used in this paper.

REFERENCES

[1] G.-H. Cottet and P. D. Koumoutsakos, *Vortex methods: Theory and practice*, Cambridge University Press, Cambridge; New York, 2000.

[2] C. Greengard, *The core spreading vortex method approximates the wrong equation*, J. Comp. Phys., 61 (1985), pp. 345–348.

[3] T. Kida, *Theoretical and numerical results of a deterministic two-dimensional vortex method*, Sadhana, 23 (1998), pp. 419–441.

[4] T. Kida and T. Nakajima, *Core spreading vortex methods in two-dimensional viscous flows.*, Comput. Methods Appl. Mech. Engrg., 160 (1998), pp. 273–298.

[5] T. Kida, T. Nakajima, and H. Suemitsu, *Second order core spreading vortex method in two-dimensional viscous flows*, JSME Int. J. Series B - Fluids and Therm. Engg., 41 (1998), pp. 441–446.

[6] D. Kosloff and H. Tal-Ezer, *A modified chebyshev pseudospectral method with an $O(N^{-1})$ time step restriction*, J. Comp. Phys., 104 (1993), pp. 457–469.

[7] B. Legras and D. G. Dritschel, *A comparison of the contour surgery and pseudo-spectral methods*, J. Comp. Phys., 104 (1993), pp. 287–302.

[8] A. Leonard, *AIAA 97-0204: Large-eddy simulation of chaotic convection and beyond*, in 35th Aerospace Sciences Meeting & Exhibit, American Institute of Aeronautics and Astronautics, 1997.

[9] J. S. Marshall and J. R. Grant, *A method for determining the velocity induced by highly anisotropic vorticity blobs*, J. Comp. Phys., 126 (1996), pp. 286–298.

[10] E. Meiburg, *Incorporation and test of diffusion and strain effects in the two-dimensional vortex blob technique*, J. Comp. Phys., 82 (1989), pp. 85–93.

[11] P. Moeleker and A. Leonard, *Lagrangian methods for the tensor-diffusivity subgrid model*, J. Comp. Phys., 167 (2001), pp. 1–21.

[12] W. H. Press, S. A. Teukolsky, W. T. Vettering, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, Cambridge, United Kingdom, 1992.

[13] L. F. Rossi, *Resurrecting core spreading methods: A new scheme that is both deterministic and convergent*, SIAM J. Sci. Comp., 17 (1996), pp. 370–397.

[14] ———, *Merging computational elements in Lagrangian simulations*, SIAM J. Sci. Comp., 18 (1997), pp. 1014–1027.

[15] ———, *High order vortex methods with deforming elliptical Gaussian blobs 1: Derivation and validation.*, Tech. Rep. 2001-11, University of Delaware, 2002.

[16] ———, *Achieving high-order convergence rates with deforming basis functions.*, Mathematical Sciences 2003-6, University of Delaware, 2003. Submitted to SIAM J. Sci. Comput. Available online at http://www.math.udel.edu/research/techrept/tech_2003.html.

[17] L. F. Rossi, J. F. Lingevitch, and A. J. Bernoff, *Quasi-steady monopole and tripole attractors for relaxing vortices*, Physics of Fluids, 9 (1997), pp. 2329–2339.

[18] Z.-H. Teng, *Elliptic-vortex method for incompressible flow at high Reynolds number.*, J. Comp. Phys., 46 (1982), pp. 54–68.

[19] ———, *Variable-elliptical-vortex method for incompressible flow simulation.*, J. Comp. Math., 4 (1986), pp. 255–262.

[20] ———, *Convergence of the variable-elliptic-vortex method for Euler equations*, SIAM J. Num. Anal., 32 (1995), pp. 754–774.

[21] L. N. Trefethen, *Spectral Methods in Matlab*, Society for Industrial and Applied Mathematics, 2000.