International Journal of Pure and Applied Mathematics
Volume 71 No. 2 2011, 289-307

AN ERROR-BASED CURVE TRACKING ALGORITHM FOR 2-DIMENSIONAL FLOWS

Paul von Dohlen[1 §], Patrick D. Miller[2]
[1]Department of Mathematics
William Paterson University
Wayne, NJ 07470, USA
[2]Department of Mathematical Sciences
Stevens Institute of Technology
Hoboken, NJ 07030, USA

Abstract: Computer simulations that track the flow of particles under the action of a time-dependent velocity field are often used to visualize the dynamics of phase-space transport. When the velocity field has two space variables, it is often sufficient to track the behavior of a curve of initial conditions, rather than a cloud of particles. Tracking a closed curve of initial particles can be used to accurately follow the evolution of a closed region in the phase space. The work presented here investigates methods for performing particle-tracking simulations that are 1) more rigorous with respect to accuracy and 2) computationally more efficient in the way in which the manifold (curve) is represented. A novel feature is to use the linear variational flow to track the first derivatives of the manifold, making it possible to construct a C^1 representation for the manifold. We use a local Hermite interpolation to define a globally C^1 curve. Error estimates for the interpolating polynomials are used as criteria to determine where additional nodes are needed (refinement) and where nodes can be removed (coarsening).

AMS Subject Classification: 37M05, 65D99
Key Words: curve tracking, flow simulation

Received: June 20, 2011

[§]Correspondence author

1. Introduction

The study of nonlinear dynamics investigates the time evolution of complicated systems. We will study the behavior of certain coherent structures in these dynamical systems. By definition we will consider a dynamical system to be either: a system of ordinary differential equations (ODE) such as

$$\frac{d\mathbf{x}}{dt} = f\left(\mathbf{x}, t\right) \qquad \mathbf{x} \subset \mathbb{R}^n, t \in \mathbb{R}^1 \tag{1}$$

or a map or difference equation of the form

$$\mathbf{x} \mapsto g\left(x\right) \qquad \mathbf{x} \subset \mathbb{R}^n \tag{2}$$

We consider the initial value problem (IVP) $\mathbf{x}(t_0) = \mathbf{x}_0$ where the unique solution represents the trajectory or orbit of a passive particle in $\mathbb{R}^n$. This work focuses on two-dimensional time-dependent velocity fields as in (1). The solution to the IVP is represented as $\mathbf{x}(t) = \varphi(t; \mathbf{x}_0, t_0)$ and the ODE generates a one-parameter family of maps (called the flow) on the phase space,

$$\varphi_t : \mathbb{R}^2 \to \mathbb{R}^2; \quad \varphi_t\left(\mathbf{x}_0\right) = \varphi\left(t; \mathbf{x}_0, t_0\right). \tag{3}$$

The general approach is to grow or evolve the relevant geometric structure under the flow generated by equation (1). In other words, we start with a well-defined representation of the structure as a set of initial particles and an interpolation rule for defining the entire structure. The structure is then mapped from t_0 to $t_0 + \Delta t$ under the flow map φ, defined as the solution to the ODE in equation 1, $\varphi(\mathbf{x}_0) = \varphi(t_0 + \Delta t; \mathbf{x}_0, t_0)$. The representation for the updated structure is evaluated to determine if additional particles need to be tracked from $t = t_0$, a step we refer to as *refinement*. A more challenging problem is to effectively identify and remove nodes from the representation that are no longer necessary, a process called *coarsening*. At each time step that this process is repeated, we end up with a geometric structure defined as a set of nodes and an interpolation rule.

While such an approach can and has been used to analyze invariant manifolds, it is important to note that the utility of these techniques is not restricted to identifying stable and unstable manifolds. Any curve of interest can be tracked and studied by employing these techniques. For example, using a simple closed curve as the initial structure is a way to track the evolution of a closed region of the phase space. This is considerably more accurate and more efficient than densely seeding the region with initial particles and then tracking the evolution of this cloud of particles.

Previous methods for evolving curves in two-dimensional phase space have focused on the accurate evolution of the individual trajectories but use low-order interpolation schemes to create the structures. In most cases the points are connected via straight line segments (linear interpolation) or perhaps using a global cubic spline to increase the smoothness and accuracy. In this work, we present a novel approach to particle tracking simulations that are based on C^1 representations of the geometric structures and include mapping first derivative information via the linear variational equation associated with the ODE in equation 1. For tracking these curves, techniques are developed that use error estimates for the interpolation scheme to determine an efficient and accurate representation of the curve at each timestep. This includes both adding nodes to improve the accuracy (refinement) and removing nodes when they are judged to be unnecessary (coarsening).

2. Parametrization

We will characterize the curve at any time by a mapping from a one-dimensional parameter space to the two-dimensional state space (see Figure 1). The parametrization approach allows for curves which cannot be globally defined as a function over one variable and it also provides a well-defined space for carrying out refinement and coarsening procedures.

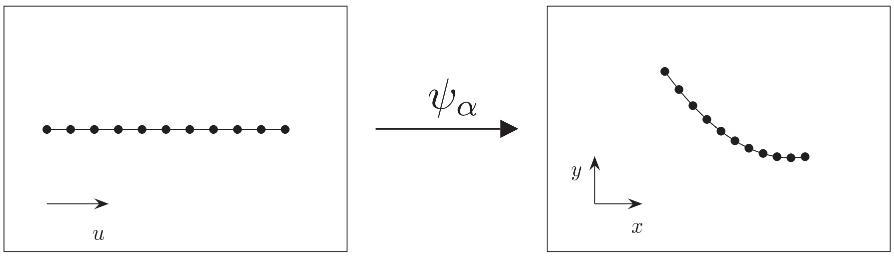


Figure 1: Curve Parametrization

The mapping ψ defines the curve as a set of nodes and an interpolation procedure over those nodes. Hence we have:

$$\begin{aligned} &\psi : P \to \mathbb{R}^2 \qquad \text{where} \quad P \subset \mathbb{R} \\ &\psi\,(u) = (\ x\,(u)\ ,\ y(u)\ ) \qquad u \in P \end{aligned} \tag{4}$$

To clearly illustrate the dependence of ψ on the current set of nodes, $\{u\}_\alpha$, we can write ψ as

$$\psi = \psi\,(u\;;\;\{u\}_\alpha) = \psi_\alpha \tag{5}$$

and here we should note that we are suppressing any reference to the specific interpolation scheme (which will be fixed throughout the simulation).

We can then evolve the curve by mapping the nodes under the flow φ in the state space, as shown in Figure 2. By composition we have a mapping $\tilde{\psi}_\alpha$ from the parameter space to the evolved curve in state space. Our goal is

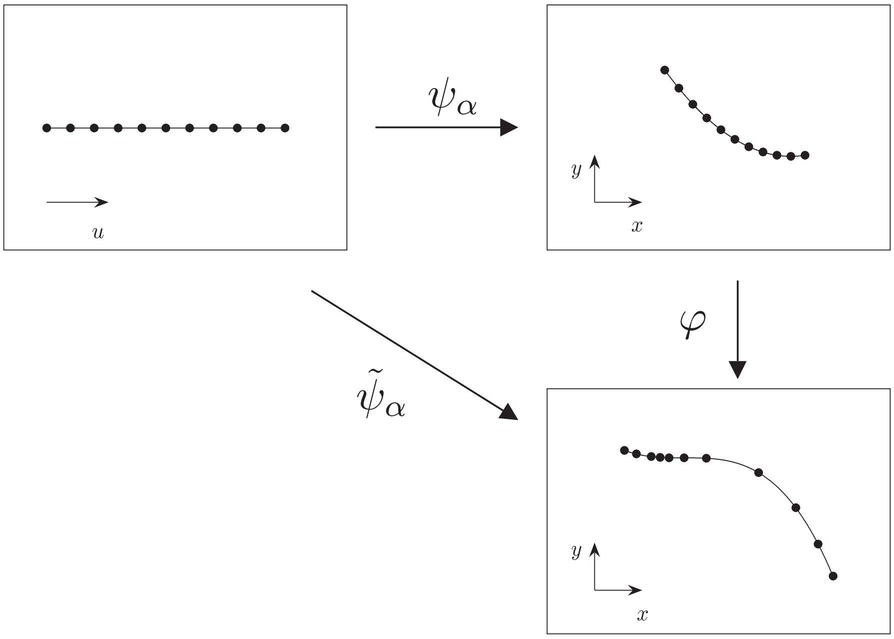


Figure 2: Curve Parametrization and Flow

to have the most accurate and efficient mapping $\tilde{\psi}_\alpha$ for the evolved curve in order to proceed with the further time evolution of the curve representation. We ultimately accomplish this using a C^1 Hermite interpolant and a refinement and coarsening algorithm to add or remove nodes as needed. Before we detail the algorithm, we need to first establish some interpolation error estimates and then also describe the procedure for mapping first derivatives, which will be necessary for our choice of interpolation.

3. Interpolation Error Estimates

In order to use interpolation error as a criterion for either refinement or coarsening we need to derive some error estimates for piecewise polynomial interpolation techniques. Here we address linear interpolation and cubic Hermite interpolation.

Let $\mathbf{x}(t)$ denote the exact curve with respect to the parameter t on the interval $a \leq t \leq b$, and $\mathbf{P}(t)$ an interpolating polynomial for $\mathbf{x}(t)$, also with domain $[a, b]$. In the context of discussing specific classes of interpolating polynomials, $\mathbf{P}(t)$ will refer to piecewise Lagrange polynomials (C^0) and $\mathbf{H}(t)$ refers to the cubic Hermite polynomial (C^1).

Letting $L(t) = |\mathbf{x}'(t)|$ and $L_P(t) = |\mathbf{P}'(t)|$, the unit tangent vectors are given by

$$\mathbf{T}(t) = \frac{\mathbf{x}'(t)}{L(t)} \quad \text{and} \quad \mathbf{T}_P(t) = \frac{\mathbf{P}'(t)}{L_P(t)}.$$

We are interested in controlling the errors $|\mathbf{x}(t) - \mathbf{P}(t)|$ and, in the case of C^1 interpolations, $|\mathbf{T}(t) - \mathbf{T}_P(t)|$. To simplify the calculation of the error in the tangent vector, it is helpful to approximate $L_P(t)$ with $L(t)$. More precisely,

$$|\mathbf{T}(t) - \mathbf{T}_P(t)| - \frac{|\mathbf{x}'(t) - \mathbf{P}'(t)|}{L(t)} = \mathcal{O}\left(\frac{L(t) - L_P(t)}{L(t)}\right)$$

Let the subscript j denote the components of the vector valued functions, for $j = 1, \dots, n$. The supremum norm will be used on the following error functions defined for each $t \in [a, b]$:

$$\begin{aligned}
\mathcal{E}_0(t) &= \big|\mathbf{x}(t) - \mathbf{P}(t)\big| &&= \Big(\sum_j \mathcal{E}_{0,j}^2\Big)^{1/2} \\
&\text{where } \mathcal{E}_{0,j}(t) = \big|x_j(t) - P_j(t)\big| \\
\mathcal{E}_1(t) &= \frac{\big|\mathbf{x}'(t) - \mathbf{P}'(t)\big|}{L(t)} &&= \Big(\sum_j \mathcal{E}_{1,j}^2\Big)^{1/2} \\
&\text{where } \mathcal{E}_{1,j}(t) = \frac{\big|x_j'(t) - P_j'(t)\big|}{L(t)} \\
E_0 &= \max_{t\in[a,b]} \mathcal{E}_0(t) \qquad && E_1 = \max_{t\in[a,b]} \big(\mathcal{E}_0(t) + \mathcal{E}_1(t)\big)
\end{aligned} \tag{6}$$

3.1. Linear interpolation

Let $\mathbf{P}(t) = [P_1(t), \dots, P_n(t)]$ be the vector-valued polynomial that interpolates $\mathbf{x}(t)$ on the interval $t_1 \leq t \leq t_2$, that is, $\mathbf{P}(t_1) = \mathbf{x}(t_1)$ and $\mathbf{P}(t_2) = \mathbf{x}(t_2)$. The error in the jth component is,

$$x_j(t) - P_j(t) = \frac{(t-t_1)(t-t_2)x_j''(\tau_j)}{2} \qquad \text{for some } \tau_j \in [t_1, t_2] \tag{7}$$

Letting $M_j[t_1,t_2] = \max\limits_{t\in[t_1,t_2]} \left|x_j''(t)\right|$, we get the following error bound on the interval $[t_1, t_2]$,

$$\begin{aligned} |x_j(t) - P_j(t)| &\le \frac{M_j[t_1,t_2]}{4\cdot 2}(t_2-t_1)^2 \\ \mathcal{E}_0(t) = \left|\mathbf{x}(t) - \mathbf{P}(t)\right| &\le \frac{M[t_1,t_2]}{8}(t_2-t_1)^2 \\ \text{where } \ M[t_1,t_2] &= \left(\sum_j M_j^2[t_1,t_2]\right)^{1/2} \end{aligned} \tag{8}$$

3.2. Cubic Hermite Interpolation

Let $\mathbf{H}(t) = [H_1(t), \ldots, H_n(t)]$ be the vector-valued polynomial that interpolates $\mathbf{x}(t)$ on the interval $[t_1, t_2]$, that is, $\mathbf{H}^{(\nu)}(t_1) = \mathbf{x}^{(\nu)}(t_1)$ and $\mathbf{H}^{(\nu)}(t_2) = \mathbf{x}^{(\nu)}(t_2)$ for $\nu = 0, 1$. The error in the jth component satisfies,

$$x_j(t) - H_j(t) = \frac{(t-t_1)^2(t-t_2)^2 x_j^{(4)}(\tau_j)}{4!} \qquad \text{for some } \tau_j \in [t_1, t_2] \tag{9}$$

Letting $M_j[t_1,t_2] = \max\limits_{t\in[t_1,t_2]} \left|x_j^{(4)}(t)\right|$, we get the following error bound on the interval $[t_1, t_2]$,

$$\begin{aligned} |x_j(t) - H_j(t)| &\le \frac{M_j[t_1,t_2]}{16\cdot 4!}(t_2-t_1)^4 \\ \mathcal{E}_0(t) = \left|\mathbf{x}(t) - \mathbf{H}(t)\right| &\le \frac{M[t_1,t_2]}{384}(t_2-t_1)^4 \\ \text{where } \ M[t_1,t_2] &= \left(\sum_j M_j^2[t_1,t_2]\right)^{1/2}. \end{aligned} \tag{10}$$

In order to find the error in the derivative, we note that the function $x(t) - H(t)$ has roots at $t = t_1, t_2$, so its derivative must have a root at some point $t_m \in (t_1, t_2)$, $x'(t_m) - H'(t_m) = 0$. Construct the function $G(s) = x'(s) - H'(s) - K\eta(s)$, where $\eta(s) = (s-t_1)(s-t_m)(s-t_2)$. This function has roots at $s \in \{t_1, t_m, t_2\}$. Assume t is any point in the interval (t_1, t_2) other than $t = t_m$

(no need to consider $t = t_1, t_m, t_2$ since $x' = H'$ there). Since $\eta(t) \neq 0$, we can choose $K = (x'(t) - H'(t))/\eta(t)$. With this choice of K the function G has at least four roots in $[a, b]$, at $s = \{t_1, t_2, t_m, t\}$. It follows that the third derivative, $G^{(3)}(s) = f^{(4)}(s) - H^{(4)}(s) - K\eta^{(3)}(s)$, has at least one root in $[t_1, t_2]$, say at $s = \tau$. Evaluating at $s = \tau$ with $H^{(4)}(s) \equiv 0$ and $\eta^{(3)}(s) = 3!$ yields $K = x^{(4)}(\tau)/3!$. Substituting the previous definition of K,

$$x'(t) - H'(t) = K\eta(t) = \frac{\eta(t)x^{(4)}(\tau)}{3!} \tag{11}$$

It is not too difficult to see that the maximum of $|\eta(s)|$ on the interval $t_1 \leq s \leq t_2$, and for all possible choices of $t_m \in [t_1, t_2]$ must occur when t_m coincides with either of the endpoints. Setting $t_m = t_1$ yields the upper bound $|\eta(s)| \leq 4\big(t_2 - t_1\big)^3/27$. Thus, an error estimate for the first derivative of the jth component is:

$$\left|x_j'(t) - H_j'(t)\right| \leq \frac{4M_j[t_1, t_2]}{27 \cdot 3!}\left(t_2 - t_1\right)^3$$

and an estimate on the tangent vector is given by,

$$\mathcal{E}_1(t) = \frac{\left|\mathbf{x}'(t) - \mathbf{H}'(t)\right|}{L(t)} \leq \frac{2M[t_1, t_2]}{81L(t)}\left(t_2 - t_1\right)^3 \tag{12}$$

where $M[t_1, t_2]$ is an upper bound on $\left|x^{(4)}(s)\right|$ for $s \in [t_1, t_2]$.

4. Mapping the First Derivative

The choice of cubic Hermite interpolation will require first derivative information in building the interpolating polynomial. In order to carry along that information at our nodes we devise a procedure for mapping the first derivative using the linear variational equation. This method should provide a more accurate account of the derivative information than estimation techniques.

Consider our system of first-order ordinary differential equations,

$$\frac{d\mathbf{x}}{dt} = f(\mathbf{x}, t) \tag{13}$$

where $\mathbf{x} \in \mathbb{R}^n$, $t \in \mathbb{R}$, and f is of class C^1 in both t and $\mathbf{x}$. Let $\varphi(t; \mathbf{x}_0, t_0)$ denote the solution to the initial value problem, equation 13 with $\mathbf{x}(t_0) = \mathbf{x}_0$. Fixing t_0 and t, this defines a map,

$$\varphi_t : \mathbb{R}^n \to \mathbb{R}^n \qquad \text{by} \ \ \varphi_t(\mathbf{x}_0) = \varphi(t; \mathbf{x}_0, t_0). \tag{14}$$

The derivative of this map at $\mathbf{x}_0$, $D_{\mathbf{x}}\varphi_t(\mathbf{x}_0)$, is a linear map from the tangent space at $\mathbf{x}_0$ to the tangent space at $\varphi_t(\mathbf{x}_0)$. In particular, suppose there is a curve $\gamma(s)$ with $\gamma(s_0) = \mathbf{x}_0$ and $\gamma'(s_0) = \vec{u}$. Under the map φ_t, a new curve is created, $\sigma(s) = \varphi_t(\gamma(s))$, with the tangent vector $\sigma'(s_0)$ given by

$$\sigma'(s_0) = D_{\mathbf{x}}\varphi_t(\gamma(s_0))\gamma'(s_0) = D_{\mathbf{x}}\varphi_t(\mathbf{x}_0)\vec{u}. \tag{15}$$

Linearizing $f(\mathbf{x}, t)$ about the trajectory φ yields a linear ODE on $\mathbb{R}^n$,

$$\frac{dz}{dt} = D_{\mathbf{x}}f(t, \varphi(t; \mathbf{x}_0, t_0))z \tag{16}$$

called the linear variational equation relative to φ. If $z(t_0)$ is just the identity matrix on $\mathbb{R}^n$, the solution $z(t)$ is the entire matrix representation for the linear map $D_{\mathbf{x}}\varphi_t(\mathbf{x}_0)$. In particular, letting $z(t_0) = \gamma'(s_0)$, the solution $z(t)$ is the tangent vector, $\sigma'(s_0) = D_{\mathbf{x}}\varphi_t(\mathbf{x}_0)\gamma'(s_0)$.

By including the linear variational equation in the calculations, it is possible to map the new location of the nodes as well as the tangent vectors associated with the new parametrization. This is a considerable advantage where derivative information is being used to construct the interpolant at each time step.

5. Algorithm

Now that we have established interpolation error estimates, we will show how these estimates are used as a criterion for implementing refinement and coarsening. The curve representation improvement process involves both criteria to be regularly checked and a procedure for carrying out the adjustment if the criteria evaluation determines that modification is necessary. By using this process we ensure that the curve representation remains accurate at each time-step of the evolution under the flow.

As stated previously, we have $\psi_\alpha = \psi\left(u\ ;\ \{u\}_\alpha\right)$ as our mapping from the one-dimensional parameter space to the two-dimensional state space. Thus we begin with ψ_{α_0} based on the initial set of nodes, u_{α_0}. Applying the flow map, φ, gives

$$\varphi \circ \psi_{\alpha_0} : P \to \mathbb{R}^2 \tag{17}$$

as a map from the parameter space to the evolved state space. We should note that $\varphi \circ \psi_{\alpha_0}$ can be calculated to *any* degree of accuracy (within the limits of the ODE solver) by seeding enough points, but the goal is to determine a set of nodes, u_{α_f}, (along with an interpolation method) which provides an accurate and efficient representation of the curve in the evolved state space. Hence we begin with

$$\tilde{\psi}_{\alpha_0}(u) = (\varphi \circ \psi_{\alpha_0})(u\,;\; u_{\alpha_0}) \tag{18}$$

as a polynomial approximation to $\varphi \circ \psi_{\alpha_0}$ and the procedure amounts to the evolution of u_{α_0} to some u_{α_f} whereby we improve the approximation based on error bounds. During the refinement stage, we have the following

$$\tilde{\psi}_{\alpha_0} \xrightarrow{\text{refine}} \tilde{\psi}_{\alpha_1} \xrightarrow{\text{refine}} \tilde{\psi}_{\alpha_2} \cdots \xrightarrow{\text{refine}} \tilde{\psi}_{\alpha_r} \tag{19}$$

resulting in a set of nodes, u_{α_r}, and a mapping, $\tilde{\psi}_{\alpha_r}$, which satisfies

$$\left|\varphi \circ \psi_{\alpha_0} - \tilde{\psi}_{\alpha_r}\right|_{\infty} < \mathcal{E} \tag{20}$$

where $\mathcal{E}$ is a pre-determined error bound as previously discussed. Then, during the coarsening stage, we have

$$\tilde{\psi}_{\alpha_r} \xrightarrow{\text{coarsen}} \tilde{\psi}_{\alpha_{r+1}} \xrightarrow{\text{coarsen}} \tilde{\psi}_{\alpha_{r+2}} \cdots \xrightarrow{\text{coarsen}} \tilde{\psi}_{\alpha_f} \tag{21}$$

resulting in a set of nodes, u_{α_f}, and a mapping, $\tilde{\psi}_{\alpha_f}$, which still satisfies

$$\left|\varphi \circ \psi_{\alpha_0} - \tilde{\psi}_{\alpha_f}\right|_{\infty} < \mathcal{E} \tag{22}$$

but such that any further coarsening would violate the error criterion. The resulting mapping, $\tilde{\psi}_{\alpha_f}$, from the parameter space to the evolved state space, having been deemed accurate and efficient, allows for the continued evolution of the curve to the next timestep.

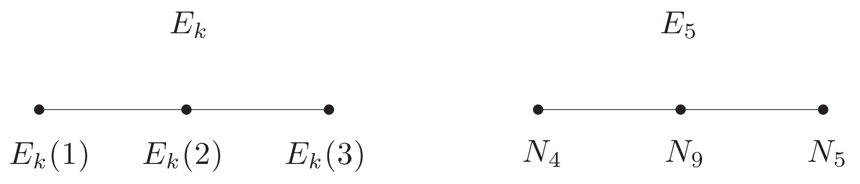
In order to carry out the procedure, we need to set up a structure for the parametric representation of the curve which will serve as the framework for any modifications to the representation. We begin by defining the nodes.

Definition 1. Let a node, N_i, refer to the ith indexed point with coordinate $u(i)$ in the parameter space and $\tilde{N}_i$ denote the corresponding point $(x(u), y(u))$ in the state space under the bijective mapping ψ; i.e., $\tilde{N}_i = \psi(N_i)$.

Definition 2. Let N be the set of all nodes in the curve representation.

We then use the nodes to define the element structure by grouping every three nodes into an element.

Definition 3. Let an element, E_k, refer to the kth indexed, ordered set consisting of three consecutive nodes. (Note that consecutive refers the ordering of nodes in the parameter space, not the indexing of the nodes.) Let E be the set of all elements.



(a) General Element (b) Example Element

Figure 3: Element

Thus, the element in Figure 3b is identified as: $E_5 = \{N_4\,,\ N_9\,,\ N_5\}$. Each element consists of two subelements determined by the grouping of the first and second nodes and the second and third nodes. For example, again referring to Figure 3b, element E_5 consists of the subelements $\{N_4\,,\ N_9\}$ and $\{N_9\,,\ N_5\}$.

The interpolating polynomial will be calculated over each element as the piecewise Hermite cubic as determined by the nodal values and the values of the derivative at each node. The quintic polynomial determined by the three nodes (and derivatives) of the element is calculated in order to estimate the bound on the fourth derivative needed for the error estimates.

5.1. Refinement

Before we analyze the criteria for refinement, let us investigate the specific procedure used for refinement if the criteria evaluation determines that refining the representation is necessary. If an element is to be refined, then two new nodes are inserted, one at each of the midpoints of the subelements in the parameter space domain. This transforms each of the subelements into new elements. The new nodes are mapped under ψ to the pre-evolved state space and then mapped under φ to the evolved state space. The newly created elements are again checked against the criteria and the process is repeated if necessary. Figures 4 and 5 illustrate both the identification and the procedure of refinement.

The first criterion for refinement is a simple element length based criterion. If the length of an element in the state space (as calculated by the sum of the linear lengths of the subelements) exceeds some predetermined maximum allowable length then the element is designated as requiring refinement. The idea here is that even if such an element could be interpolated within a desired accuracy the large distance between the nodes could cause the representation to miss subtle changes in the flow field. Using this criterion the curve representation remains well-seeded from a pure distance perspective regardless of any

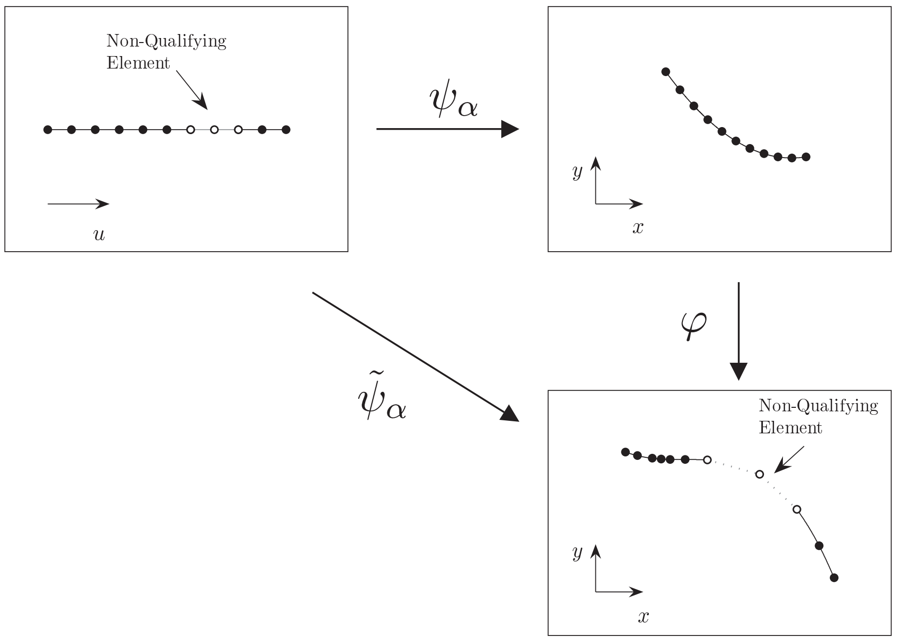


Figure 4: Identification of Refinement

other criteria which are imposed.

The second criterion for refinement is based on the interpolation errors derived previously. Our method involves the use of the C^1 cubic Hermite interpolant, with the quintic used to estimate values of the fourth derivative as needed for error estimates. At every time-step of the flow evolution, the curve representation is checked to ensure that both the error in the interpolant and the error in the derivative remain within specified error bounds. Thus for each element, E_k we require,

$$\mathcal{E}_0\left(E_k\right) \equiv \frac{M\left(E_k\right)}{384}\left(E_k\left(3\right) - E_k\left(1\right)\right)^4 < I_{\max}$$

and

$$\mathcal{E}_1\left(E_k\right) \equiv \frac{2M\left(E_k\right)}{81}\left(E_k\left(3\right) - E_k\left(1\right)\right)^3 < D_{\max}$$

$$\text{where}\left\{\begin{array}{l} I_{\max} = \text{ maximum allowable error in the interpolant} \\ D_{\max} = \text{ maximum allowable error in the derivative} \\ M\left(E_k\right) = \max_{u\in U_k}\{\max|x^{(4)}\left(u\right)|, \max|y^{(4)}\left(u\right)|\} \end{array}\right.$$

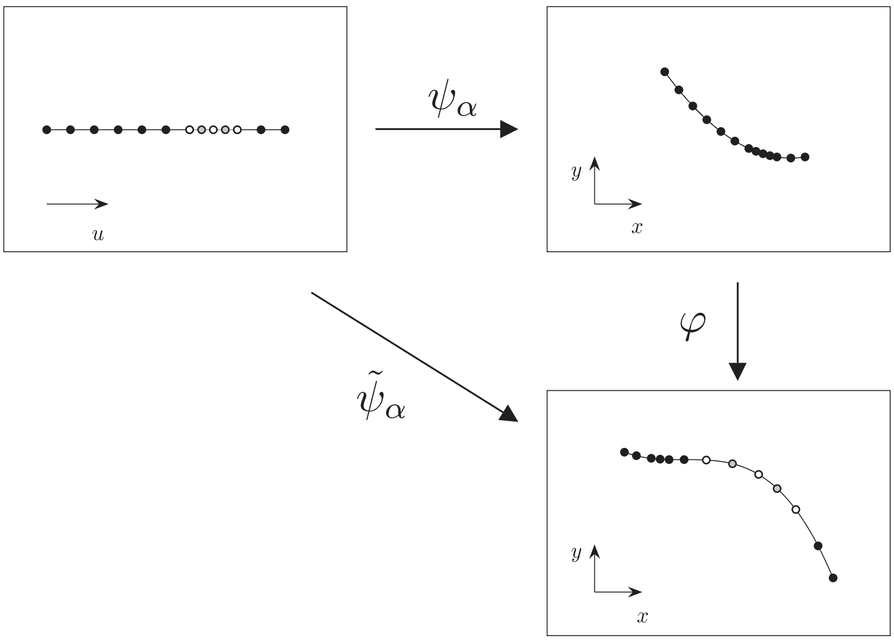


Figure 5: Refinement Procedure

We use the quintic determined by the three nodes of the element to determine the maximum value of the fourth derivative. Any element which fails to meet the error criteria is refined under the previously defined procedure and the resulting new elements are subsequently verified.

5.2. Coarsening

The insertion of additional nodes is necessary in order to keep the curve representation accurate in areas of extensive stretching as well as areas in which the flow causes the curve to bend sharply. Over time, though, nodes may accumulate in regions of strong contraction causing the curve to be over-resolved there. The unnecessarily large number of nodes leads to excessive calculations and compromises the efficiency of the entire technique. We therefore propose an approach for addressing curve coarsening via merging cells. This method of merging cells is a straightforward approach which examines neighboring elements to determine if the representation would be sufficient if some of the elements could be merged together.

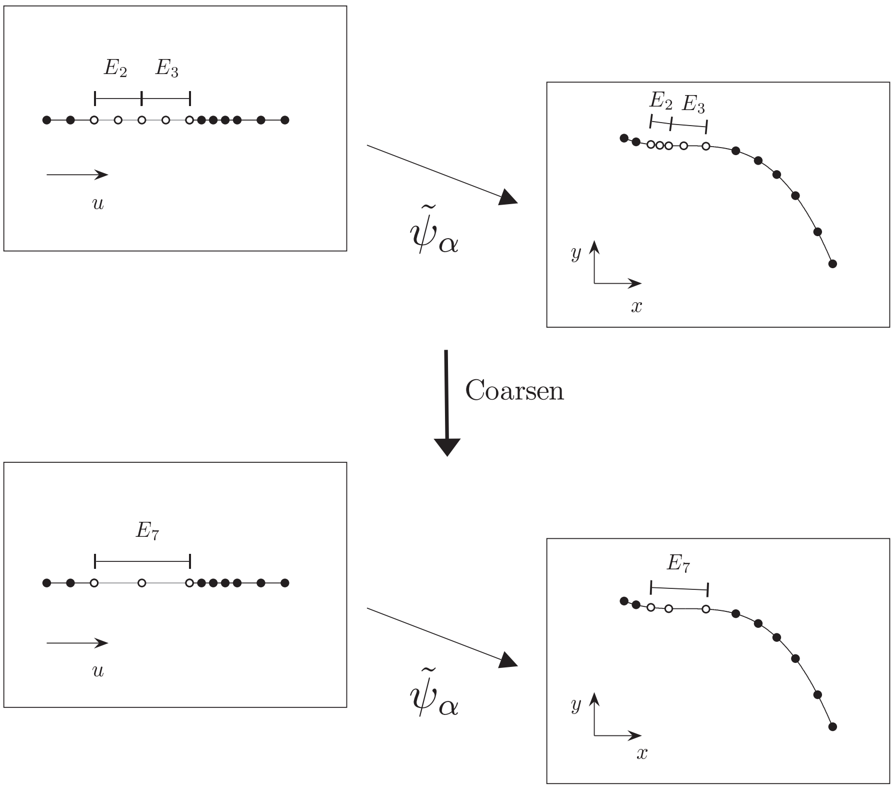


Figure 6: Coarsening Procedure (Merging)

The cell merging method for coarsening involves examining an existing, valid curve representation to determine if there are over-resolved regions of the curve which could be represented accurately by combining some neighboring cells. Thus, it is a local procedure which adjusts a given representation rather than a global procedure which would use the information from the current representation to create an entirely new representation.

Once a sufficiently refined curve representation has been established, we check each cell with its neighboring cell to see if the resulting merged cell would still qualify under the length and error criteria established for refinement. If the merged cell qualifies then that cell is used in the representation instead of the two smaller cells. Note that unlike in the refinement process, this procedure does not involve mapping a node to the pre-evolved state space as this would be unnecessary because we are starting with a curve representation which is

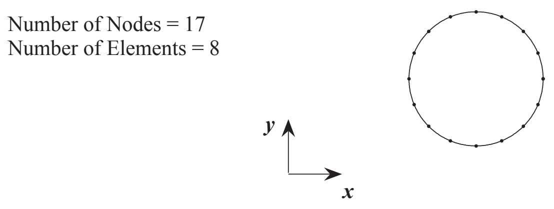


Figure 7: Initial Closed Curve (Example)

sufficiently defined, just possibly over-resolved in some regions. The process is iteratively repeated until the original representation has been reduced to the optimal coarsened representation. Figure 6 shows how two elements (E_2 and E_3) could be replaced by the single element E_7 thus reducing the number of nodes and elements without any significant change to the resulting interpolated curve.

This cell merging method has the benefit that it only affects regions which are over-resolved and thus is not subject to any possible accumulating computational error in already well-represented regions. As an added benefit, it also extends naturally to the higher dimensional case where triangular elements will be merged in a similar manner.

6. Results

As an illustrative example of the refinement and coarsening procedure, we consider the evolution of a curve in a model flow. The vector field used for this simulation is given by:

$$\begin{aligned} x' &= x^2 \\ y' &= y + \frac{1}{x} \end{aligned} \tag{23}$$

The exact solution to this initial value problem is:

$$\begin{aligned} x\,(t) &= \frac{x_0}{1 - t\,x_0} \\ y\,(t) &= \left(y_0 + \frac{1}{x_0} - 1\right) e^t - \frac{1}{x_0} + t - 1 \end{aligned} \tag{24}$$

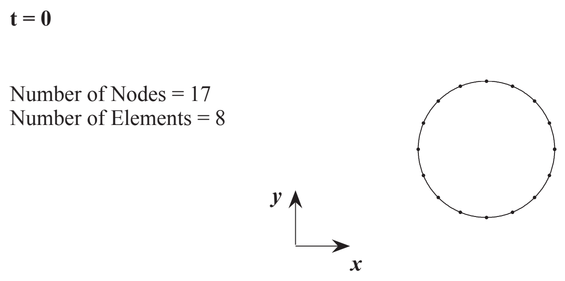


Figure 8: Evolution of Closed Curve ($t = 0$)

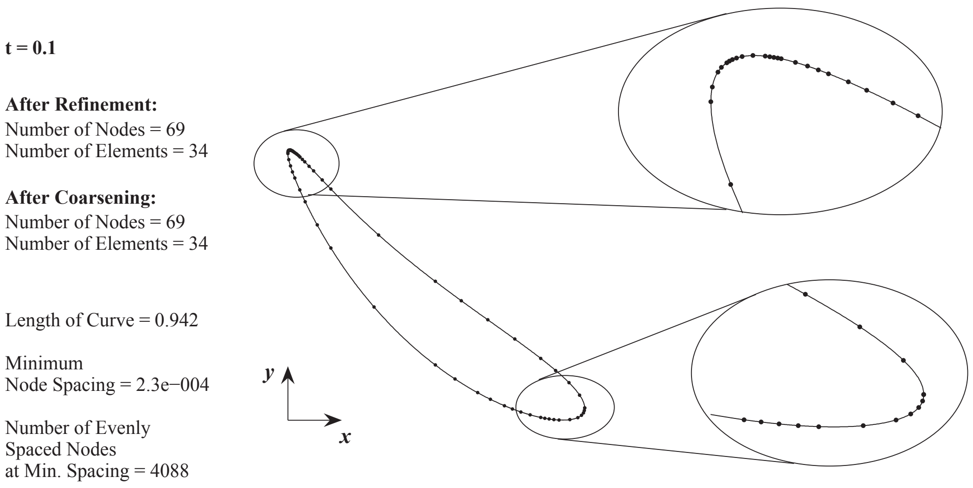
This example shows the evolution of a closed curve (initially defined as a circle) under the flow generated by the system in equation 23. Figure 7 illustrates the representation of the curve as the mapping of 17 nodes from the parameter space to a circle in the physical space.

Figure 8 illustrates the final curve representation at selected time-steps of the evolution with zoomed views of high curvature regions where we would expect to have the most difficulty producing a good representation. The figures illustrates the method's ability to keep these regions well-resolved.

As another illustration of the effectiveness of this algorithm, we ran the same initial curve using only a length-based criterion with maximum node separation distance set to 0.025 (based on the 0.0245 separation for the initial set of nodes). The number of nodes at timesteps 0.1,...,0.5 are 56, 99,170, 225 and 314, respectively. By $t = 0.5$ the length-based method already requires more than three times the number of nodes as the Hermite interpolant. Moreover, comparison with the exact curve clearly illustrates the inability of the length-based method to adequately resolve the high curvature regions.

Figure 9 shows a zoomed comparison of the two methods for $t = 0.5$ of the closed curve example. The difference is especially pronounced because of the high curvature of the sharp bend in the curve in this region.

Furthermore, for this solvable analytic example we can compare both methods with an exact solution. Figure 10 shows a comparison on a zoomed region of the curve. Here again, we note the inability of the length-based method to capture the true curvature of the region while the proposed method achieves, at least visually, excellent correspondence with the true curve.



t = 0.1

After Refinement:
Number of Nodes = 69
Number of Elements = 34

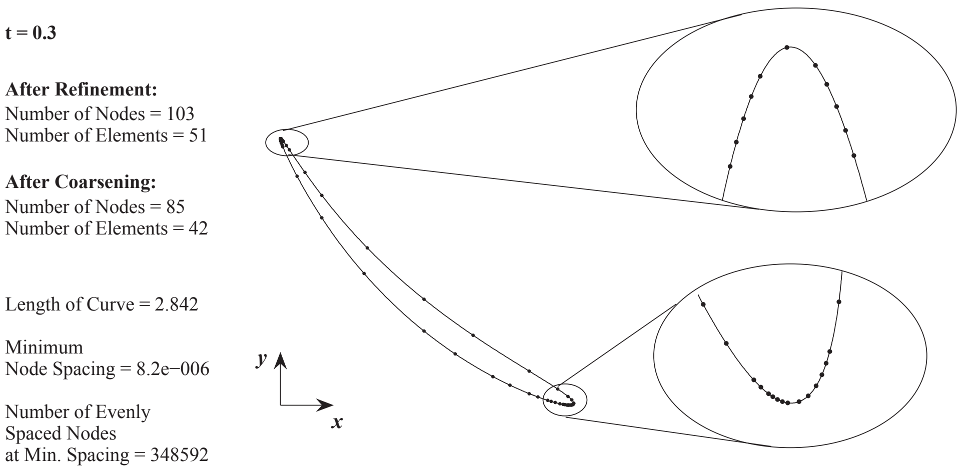
After Coarsening:
Number of Nodes = 69
Number of Elements = 34

Length of Curve = 0.942

Minimum
Node Spacing = 2.3e−004

Number of Evenly
Spaced Nodes
at Min. Spacing = 4088

Figure 8: Evolution of Closed Curve ($t = 0.1$)



t = 0.3

After Refinement:
Number of Nodes = 103
Number of Elements = 51

After Coarsening:
Number of Nodes = 85
Number of Elements = 42

Length of Curve = 2.842

Minimum
Node Spacing = 8.2e−006

Number of Evenly
Spaced Nodes
at Min. Spacing = 348592

Figure 8: Evolution of Closed Curve ($t = 0.3$)

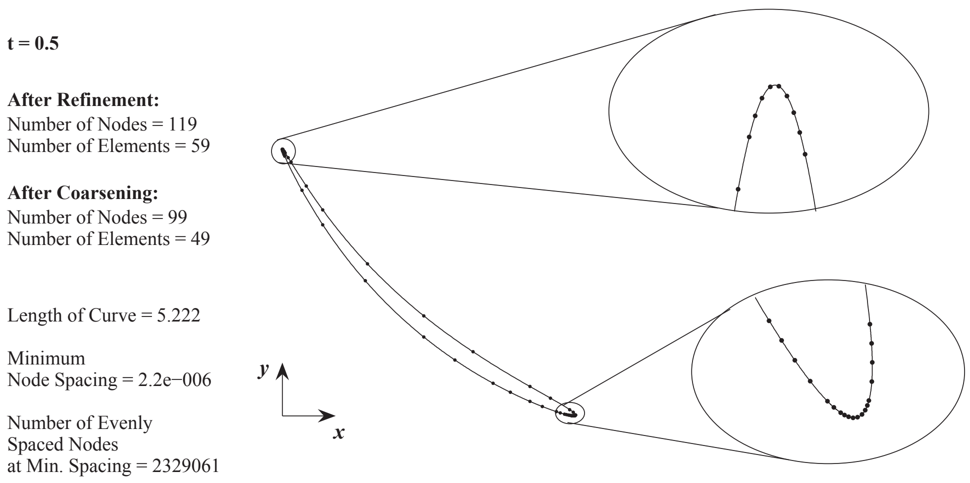


Figure 8: Evolution of Closed Curve ($t = 0.5$)

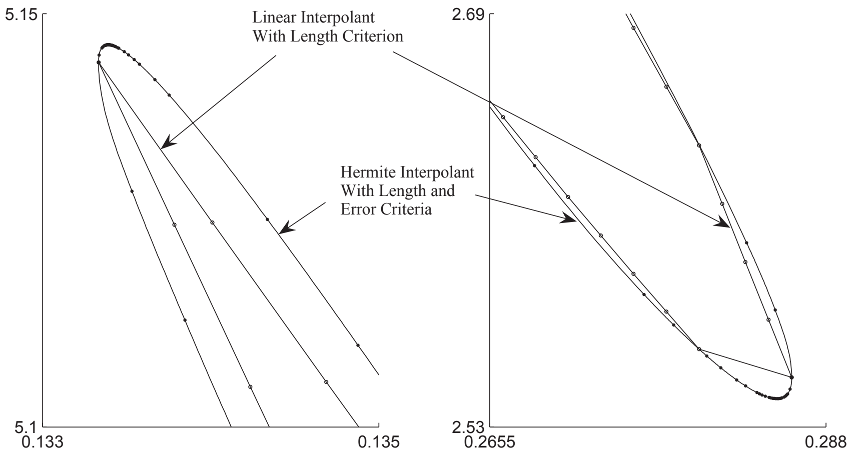


Figure 9: Method Comparison

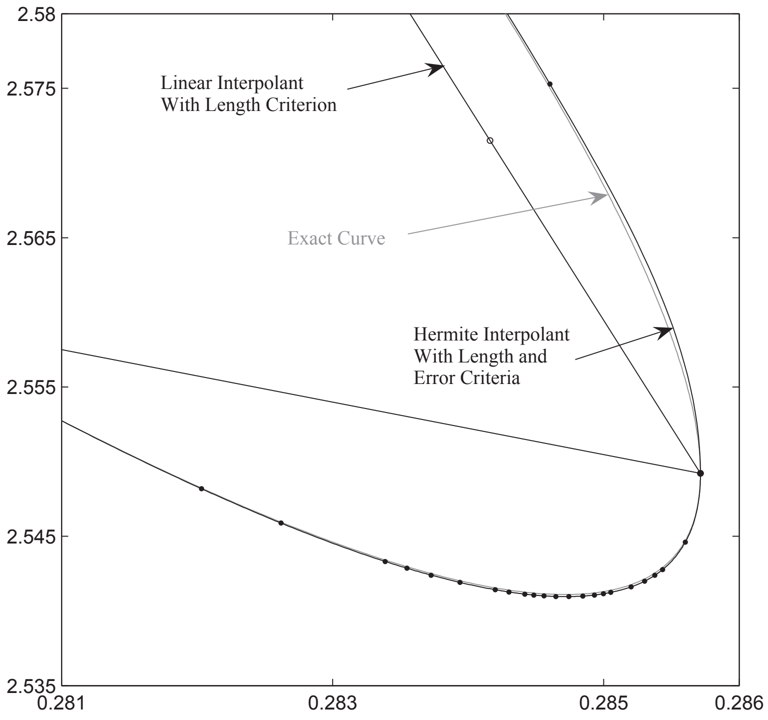


Figure 10: Comparison With Exact Curve

7. Conclusion

Numerical simulations of particle tracking serve as a useful tool for visualizing the flow associated with systems of nonlinear ordinary differential equations. Rather than simply tracking clouds of individual particles, this work has focused on tracking coherent geometric structures (curves). One important contribution of this work is that the structures are represented as globally C^1 manifolds. The curves are defined parametrically and the C^1 representation comes from a local polynomial interpolation using the position and tangent vectors on a set of nodes. By simultaneously solving the nonlinear ODE and its associated linear variational equation, we map both the position and tangent vectors at t_0 to their new values at $t_0 + \Delta t$.

In tracking the curves, error estimates for the Hermite polynomial interpolation are used to determine where nodes should be added or removed. Specifically, error estimates for a Hermite interpolation on a pair of adjacent intervals are used to determine where refinement should take place. The method of coarsening involves merging adjacent elements when deemed appropriate from the error estimates. Using this process of refinement and coarsening, we were able to demonstrate accurate representations of curves after several iterations of the flow, using a minimal number of nodes.

References

[1] Kendall E. Atkinson, *An Introduction to Numerical Analysis*, Second Edition, John Wiley and Sons Inc., New York (1989).

[2] John Guckenheimer, Philip Holmes, *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields*, Applied Mathematical Sciences, **42**, Springer-Verlag, New York (1983), xvi+453.

[3] G. Haller, Finding finite-time invariant manifolds in two-dimensional velocity fields, *Chaos*, **10** (2000), 99-108.

[4] Morris W. Hirsch, Stephen Smale, *Differential Equations, Dynamical Systems, and Linear Algebra*, Pure and Applied Mathematics, Volume 60, Academic Press, New York-London (1974), xi+358.

[5] D. Hobson, An efficient method for computing invariant manifolds of planar maps, *JCP*, **104**, No. 1 (1993), 14-22.

[6] Peter Lancaster, Kestutis Salkauskas, *Curve and Surface Fitting: An Introduction*, Academic Press, London (1986).

[7] Patrick D. Miller, Lawrence J. Pratt, Karl R. Helfrich, Christopher K.R.T. Jones, Chaotic transport of mass and potential vorticity for an island recirculation, *Journal of Physical Oceanography*, **32**, No. 1 (2002), 80-102.

[8] J.M. Ottino, Mixing, chaotic advection, and turbulence, *ARFM*, **22** (1990), 207-253.

[9] A.M. Rogerson, P.D. Miller, L.J. Pratt, C.K.R.T. Jones, Lagrangian motion and fluid exchange in a barotropic meandering jet, *JPO*, **29**, No. 10 (1999), 2635-2655.