

The bleeps, the sweeps, and the creeps: Convergence rates for dynamic observer patterns via data assimilation for the 2D Navier–Stokes equations

Trenton Franz^a, Adam Larios^{b,*}, Collin Victor^b

^a School of Natural Resources, University of Nebraska-Lincoln, Lincoln, NE 68588-0130, United States of America

^b Department of Mathematics, University of Nebraska-Lincoln, Lincoln, NE 68588-0130, United States of America

Received 19 October 2021; received in revised form 20 January 2022; accepted 22 January 2022

Available online 12 February 2022

Abstract

We adapt a continuous data assimilation scheme, known as the Azouani–Olson–Titi (AOT) algorithm, to the case of moving observers for the 2D incompressible Navier–Stokes equations. We propose and test computationally several movement patterns (which we refer to as “the bleeps, the sweeps and the creeps”), as well as Lagrangian motion and combinations of these patterns, in comparison with static (i.e. non-moving) observers. In several cases, order-of-magnitude improvements in terms of the time-to-convergence are observed. We end with a discussion of possible applications to real-world data collection strategies that may lead to substantial improvements in predictive capabilities.

© 2022 Elsevier B.V. All rights reserved.

Keywords: Continuous data assimilation; Azouani–Olson–Titi algorithm; Navier–Stokes equations; Moving observers; Lagrangian particles; Observer interpolation

“I’m having trouble with the radar, sir. [...] I’ve lost the bleeps, I’ve lost the sweeps, and I’ve lost the creeps.”

[Michael Winslow as Radar Technician in *Spaceballs* [1]]

1. Introduction

One difficulty in the simulation of many physically interesting dynamical systems is that initial data is often incomplete or inaccurate. However, one can circumvent this difficulty by using the tools of *data assimilation*, which is a class of techniques used to increase accuracy by combining time-dependent observational data together with a physical model, driving simulations to converge to the “true” solution. In this work, we concentrate on a simple yet powerful data assimilation tool known as the Azouani–Olson–Titi (AOT) algorithm, since it readily lends itself to the case of *dynamic observation points*, which is the major focus of this manuscript. In a previous work [2], two of the authors first proposed the idea of dynamic observers for the AOT algorithm, and showed that in a simplified 1D model (the Allen–Cahn reaction–diffusion equations), moving the observation points in time resulted

* Corresponding author.

E-mail addresses: trenton.franz@unl.edu (T. Franz), alarios@unl.edu (A. Larios), collin.victor@huskers.unl.edu (C. Victor).

in order-of-magnitude increases in the time to convergence to machine precision error. In the present work, we extend the idea of continuous data assimilation via moving observers to the 2D Navier–Stokes equations, and show similar gains in convergence rates. Moreover, we propose and investigate several movement schemes for observers, and compare these to the case of static (i.e., non-moving) observers. During the completion of this manuscript, we learned of another work [3], which shows convergence of the AOT algorithm for the 2D Navier–Stokes equations with observers in a moving square patch. However, the purpose of the present work is to examine the effectiveness of several different, arguably more realistic, movement patterns with the aim of real-world data collection strategies.

Since its inception in [4,5], the AOT algorithm has been the subject of much recent study in both analytical studies [3,6–38] and computational studies [2,39–47].

In this work we investigate the AOT algorithm as applied to the 2D Navier–Stokes equations (NSE). Writing the NSE in an abstract form¹ we have the following equation

$$\frac{d}{dt}u = F(u). \quad (1.1)$$

Note that in (1.1), we do not know the initial condition for this system. However, we assume that we can observe the solution at certain spatial locations, which may be sparsely distributed, say with some characteristic spatial distance between observations $h > 0$. Using these measurements, the AOT algorithm [4,5] constructs the system

$$\begin{cases} \frac{d}{dt}v = F(v) + \mu(I_h(u) - I_h(v)) \\ v(0) = v_0 \end{cases} \quad (1.2)$$

where $I_h(u)$ is a suitable interpolation of the observed values of a strong solution u of (1.2), and $\mu > 0$ is a constant, sometimes called the “nudging parameter”. It was proven in [4] that for sufficiently large μ and sufficiently small h that $\|u(t) - v(t)\|_{L^2} \rightarrow 0$ and $\|\nabla u(t) - \nabla v(t)\|_{L^2} \rightarrow 0$ exponentially fast in time as $t \rightarrow \infty$ for any choice of $v_0 \in L^2$ such that v_0 is divergence-free. Note that in many works including the present one, for simplicity data is assumed to be observed continuously in time and without noise. However, similar exponential convergence results for discrete-in-time observations were proven in [22] and examined computationally in [50], and for noisy data and stochastic forcing in [7,40].

An important aspect to consider when capturing physical phenomena via data assimilation is where to place observers. Observers can be thought of as physical objects such as weather stations in the case of predicting the weather. These observers may require financial investment and manpower to place, thus it is crucial to minimize the number of observers required for convergence of the method. Computational tests in the case of 2D NSE indicated exponential convergence in time using a static uniform grid of observers [44]. The rate of convergence can typically be sped up by adding more observers, however computational studies [2,51] seem to suggest that the rate of convergence can be improved by moving existing observers instead. In a previous work we found in the case of the 1D Allen–Cahn equations that the number of observers can be drastically reduced by replacing the static uniform grid with a mobile cluster of fine length scale [2]. Recently in [51] it was shown computationally that mobile local data assimilation in the case of 2D NSE achieved fast convergence rates when compared to a non-mobile local data assimilation algorithm. There mobile local data assimilation was investigated in the single regime of continuously moving a square subdomain around the domain. In this work we investigate computationally the effect of movement strategies for observers on convergence rates for 2D NSE for both local and global data assimilation.

The effect of mobile observers has been seen recently in the reduction of accuracy in weather forecasting models due to the elimination of cross Atlantic air travel. For example, it was shown in [52] the 50%–75% reduction in air flights in 2020 has lead to a decrease of 2°C in air temperature forecasts over Greenland and Siberia and a sustained reduction in flights will lead to further degradation of the air pressure and wind speed forecast skill.

In the present work, we focus on the 2D incompressible Navier–Stokes equations, given by:

$$u_t + u \cdot \nabla u - \nu \Delta u = \nabla p + f, \quad (1.3a)$$

$$\nabla \cdot u = 0. \quad (1.3b)$$

¹ After solving for the pressure, one can write $F(u) = \nu \Delta u - u \cdot \nabla u - \nabla \Delta^{-1} \nabla \cdot (u \otimes u)$ (see, e.g., [48,49] for details), but this specific form is not especially relevant for the present discussion.

Here $u(x, t) = (u_1(x, t), u_2(x, t))$ is the velocity, $\nu > 0$ is the viscosity, p is the pressure and f is a forcing term. Applying the continuous data assimilation algorithm (1.2) to this system yields the following equations:

$$v_t + v \cdot \nabla v - \nu \Delta v = \nabla p + f + \mu(I_{h,t}(u) - I_{h,t}(v)), \quad (1.4a)$$

$$\nabla \cdot v = 0. \quad (1.4b)$$

Here, $I_{h,t}$ is a linear operator representing nodal piecewise-linear interpolation. In the static case, more general interpolation operators were considered in [4], but for the sake of simplicity, we only consider the case of piecewise-linear interpolation. Note that, in contrast to the usual AOT algorithm for static observers, here, we allow the observer locations to depend on time, as denoted by the subscript t in $I_{h,t}$.

This paper is laid out as follows. In Section 2, we describe the various observer movement patterns that we consider. In Section 3, we describe our computational methods and results, but to keep the discussion concise, we only refer to the graphics, which are in the appendix. In Section 4, we conclude with a comparative discussion on the merits of the various observer movement patterns, and contrast them with the results of the static observer case.

2. Movement patterns

For this study we computationally examined different methods of moving observers dynamically. We group these into three general movement patterns that we call “The Bleeps”, “The Sweeps” and “The Creeps”. In addition to these movement patterns we also investigate a fourth pattern given by observers as Lagrangian particles suspended in the fluid.

2.1. The bleeps

“The Bleeps” refers to the strategy of observing data at random locations that change over time. Observers are initially placed at locations that are determined randomly and moved to new random locations at every time-step. These random locations are determined by picking uniformly random integers $x_i \in \{0, 1, \dots, N\}$ with locations given by $\vec{x} = (x_1 \cdot \Delta x - \pi, x_2 \cdot \Delta y - \pi)$. Here $\Delta x = \Delta y = 2\pi/N$, where N is the resolution of the underlying spatial grid used in simulations. The locations are chosen in this manner so that their coordinates align with the underlying spatial grid.

While we do not claim that there is a physically-realistic scenario corresponding to this movement strategy, studying “The Bleeps” can give some idea of how non-static observations can influence convergence rates in the absence of any coherent spatial structure. Thus, we examine this case as guideline or comparative test to shed light on the importance of domain coverage vs. certain geometric patterns in observer motion.

2.2. The sweeps

“The Sweeps” refers to the general strategy of moving a group of observers at a constant velocity periodically over the domain. For this method we investigated three separate movement strategies referred to as thin, thick, and random sweeps.

“Thin-sweeps” refers to the strategy of utilizing a large cluster of observers placed in a thin rectangular strip of the domain that moves continuously to the right at a constant velocity. The rectangular strip utilized by this strategy has length $x = a\Delta x$ and width $y = L_y = 2\pi$ and moves at constant velocity $b\frac{\Delta x}{\Delta t}$ for some $a, b \in \mathbb{N}$. Note that the form of the velocity and x -dimension were chosen specifically so that the rectangular region would align with the underlying spatial grid at every timestep. In all of the simulations conducted we choose values $a \in \{1, \dots, 20\}$ and additionally with $b = a$. In simulations this rectangular box contains observers placed at each gridpoint along the spatial grid, meaning that the full solution of u is observed locally in this rectangular region. Outside of this rectangle, $I_{h,t}(u - v) = 0$ for any given time t . Thus $I_{h,t}$ is given as follows:

$$I_{h,t}(u - v) = (u - v)\chi_{R(t)}$$

Here $R(t)$ refers to the sliding rectangular box of observers at the t (see Fig. 2.1).

“Thick-sweeps” refers to the strategy of utilizing a uniform grid of observers placed on a rectangular region that is fitted with a coarse grid of observers (see Fig. 2.1). We fix the rectangular region to be a box containing the

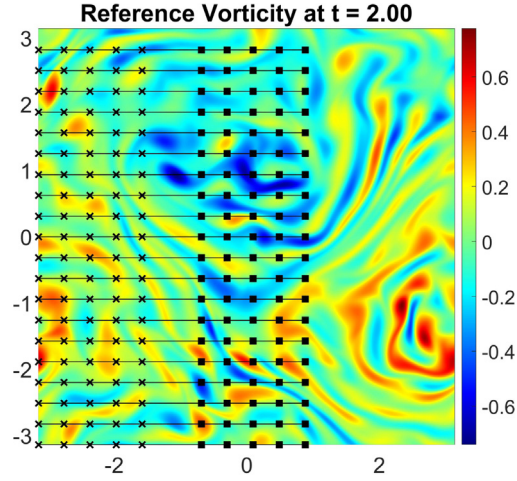


Fig. 2.1. Graph displaying movement of observers using the thick sweeps movement strategy displayed over the vorticity of the reference solution at time $t = 2$. Observers were initially placed at locations indicated by 'x' and travel to the right along the black paths with ending locations marked by '■'.

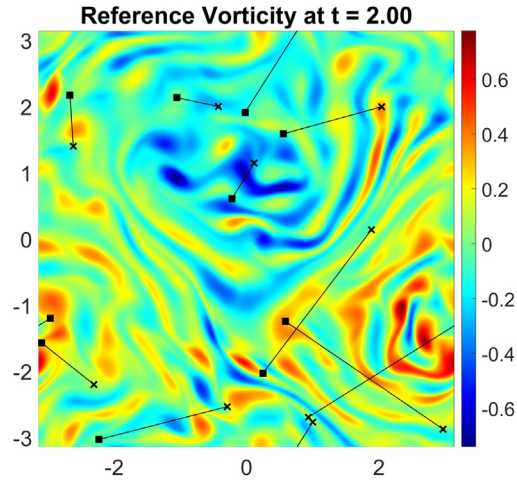


Fig. 2.2. Graph displaying movement of observers using the random sweeps movement strategy displayed over the vorticity of the reference solution at time $t = 2$. Observers were initially placed at 10 locations indicated by 'x' and travel along random trajectories indicated by black paths with ending locations marked by '■'.

fourth of the domain given by $R = \{(x, y) \in \mathbb{T}^2 : -\pi \leq x \leq -\frac{\pi}{2}\}$. This rectangular region moves to the right at the constant velocity $b \frac{\Delta x}{\Delta t}$. In all of the simulations whose results are presented here we used $b = 1$. For certain velocities one can view this method as a uniform static grid with only a fourth of the observers active at any given time. $I_{h,t}$ is given as follows:

$$I_{h,t}(u - v) = (I_h(u - v))\chi_{R(t)}$$

“Random-sweeps” refers to the strategy of placing observers at random locations with an additional random velocity (see Fig. 2.2). A major difference between this and all of the previous methods is that the location of the observers are not restricted to the underlying spatial grid. This means that the value of $u - v$ must be interpolated from the spatial grid at each timestep in order to obtain approximate values for the observer to observe. The velocities of each observer are given by vectors with each component having a random uniformly distributed value from the

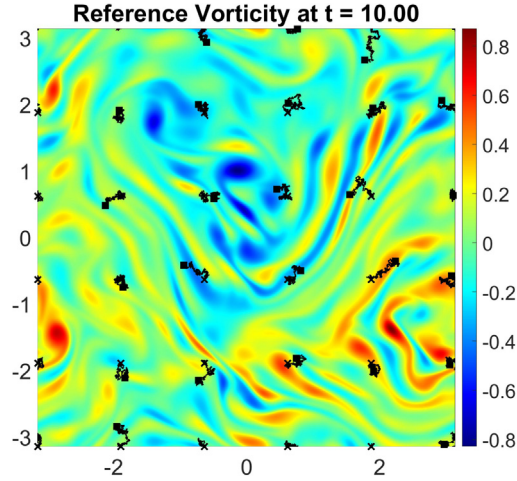


Fig. 2.3. Graph displaying movement of observers using the creeps movement style displayed over the vorticity of the reference solution at time $t = 10$. Observers were initially placed at 25 locations indicated by ‘x’ and travel along a random walk indicated by black paths with ending locations marked by ‘■’.

interval $(-1, 1)$. $I_{h,t}$ is given as follows:

$$I_{h,t}(u - v) = I_{h,t} \circ J_{\Delta x,t}(u - v)$$

Here J is a linear interpolation operator applied to $u - v$ in order to get the approximate values for the observers with I interpolating these observed values onto \mathbb{T}^2 . We note that random-sweeps can be viewed as a hybridization of bleeps and sweeps. We have elected to list it as a variation of sweeps as the observers move continuously in time (see Fig. 2.2).

We study each of these sweep-movement strategies as each can be potentially realized in physical scenarios. The thin-sweep observers can be used to model observations given by, e.g., a laser sweeping through a domain in particle image velocimetry (PIV) applications, where a velocity profile is taken as an observation. Thick sweeps can be used to model observers on mobile platforms that move at a fixed velocity. Random sweeps can be thought of as modeling observers with a fixed velocity, e.g., aircraft or satellites.

2.3. The creeps

“The creeps” refers to the strategy of having observers move according to random walks (“creeping” along) (see Fig. 2.3). After a given amount of time, each observer will (uniformly) randomly pick an integer $j \in \{1, 2, 3, 4, 5\}$. These integers determine the direction of motion $d(j)$ as follows:

$$d(j) = \begin{pmatrix} x_1 \Delta x \\ x_2 \Delta y \end{pmatrix} \quad \text{where} \quad \begin{cases} x_1 = 1, x_2 = 0, & \text{if } j = 1, \\ x_1 = -1, x_2 = 0, & \text{if } j = 2, \\ x_1 = 0, x_2 = 1, & \text{if } j = 3, \\ x_1 = 0, x_2 = -1, & \text{if } j = 4, \\ x_1 = 0, x_2 = 0 & \text{if } j = 5. \end{cases}$$

That is, they randomly move in a cardinal direction, or remain still, with uniform probability (see Fig. 2.3).

2.4. Lagrangian particles

The final movement strategy we considered is given by observers following Lagrangian particles trajectories (see Fig. 2.4). Observers are initialized into a uniform grid of spatial resolution $h > 0$ and then follow the Lagrangian

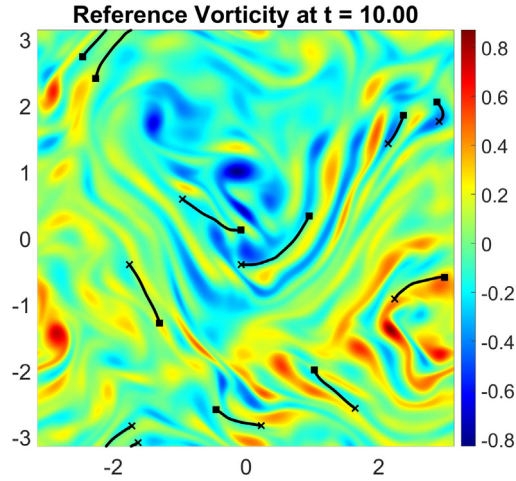


Fig. 2.4. Graph displaying movement of observers given by Lagrangian particles displayed over the vorticity of the reference solution at time $t = 10$. Observers were initially placed at 10 locations indicated by ‘x’ and travel along trajectories indicated by black paths with ending locations marked by ‘■’.

trajectory $\ell(t)$ as determined by

$$\frac{d\ell}{dt} = u(\ell(t), t). \quad (2.1)$$

This is a very natural type of dynamic observer, as the movement of the observer comes exactly from the behavior of the system that we are attempting the capture in simulations of v . For instance, one can consider sensors attached to buoys (e.g., Argo floats) set adrift in the ocean as observers.

Lagrangian trajectories for observers were evolved forward using third-order Adams–Bashforth method with values of u at locations $(\ell(t), t)$. As in the case for random-sweeps, observers in the case of Lagrangian particles the observations do not in general occur at location on the underlying spatial grid. Therefore we must approximate the value $u(\ell(t), t)$ via a linear interpolation of u . In this case we can consider $I_{h,t}(u - v)$ as follows:

$$I_{h,t}(u - v) = I_{\ell(t),t} \circ J_{\ell(t)}(u - v)$$

Here $J_{\ell(t)}(u - v)$ is a piecewise-linear interpolant used to approximate the values of $u - v$ in order to find the observational data and $I_{\ell(t),t}$ is another piecewise-linear interpolant used to extend the observational data to the numerical grid (see Fig. 2.4).

3. Computational results

All of our computations were done on the 2D NSE in the vorticity-stream function formulation using a fully dealiased pseudospectral code with physical domain $\mathbb{T}^2 = [-\pi, \pi]^2$. As in [51] we used viscosity $\nu = 10^{-4}$ with forcing as given in [44] that was multiplied by a scalar constant to obtain a forcing f with Grashof number $G = 10^6$. Additionally, consistent with [51,53,54], we used a third-order Adams–Bashforth method with an integrating factor to solve for the linear term exactly. We used step size $\Delta t = 0.005$, and our initial data was generated by evolving zero initial data out to time $t = 25,000$. The spectrum of the initial data for the reference solution u is plotted in Fig. 3.1 and one can see that spectrum is well-resolved (it also remained well-resolved for all times). We note that the spectrum of the assimilation solution v was temporarily slightly under-resolved in all of our simulations. Additionally, we used spatial resolution $N^2 = 1,024^2$ in our simulations in order to fully resolve the energy spectrum to machine precision (roughly 2.22×10^{-16}) before the $2/3$ ’s dealiasing cutoff (see, e.g., Fig. 3.1).

Remark 3.1. In simulations it was found that the choice of initial data v_0 did not affect convergence rates significantly. For our simulations we chose $v_0 = 0$ at all points in the domain, but another natural choice of initial data is $v_0(x) = I_h(u_0)$. We found that the choice of initial data only effectively lowered the initial error without

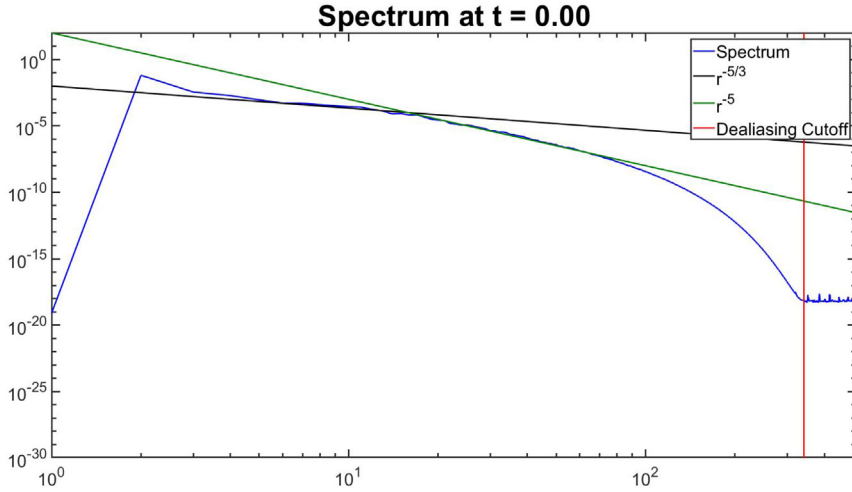


Fig. 3.1. Energy spectrum of the initial data with $\nu = 10^{-4}$, $G = 10^6$, and $\Delta t = 0.005$. The vertical red line is the $2/3$'s dealiasing cutoff at $\frac{2}{3} \frac{N}{2} = 341.3$.

increasing the rate of convergence at larger times. Due to the lower initial error, simulations with the interpolated observational data as the initial data converged between 0.5 to 2 units of time faster than zero initial data. The exact speedup depends on the number of observers used for the simulation. We elected to use zero initial data to avoid potentially favoring methods with globally distributed observers.

Note that while the equation was simulated using the vorticity-stream function formulation, the interpolation term was calculated at the velocity level using nodal interpolation in physical space. To preserve the periodicity in the interpolation across the periodic “boundary” we interpolated using observed values over $[-3\pi, 3\pi] \times [-3\pi, 3\pi]$ instead of just over $\mathbb{T}^2 = [-\pi, \pi] \times [-\pi, \pi]$, and then truncated the result back to the physical domain. Since we are simulating the equations at the vorticity-stream function level, we calculate $\|\psi - \tilde{\psi}\|_{L^2}$ as the error between the simulated and reference solutions. In particular, we look at the error with the smallest number of observers needed to obtain convergence by roughly time $t \approx 70$ (see Figs. 3.2(a) and 3.3(a)), and the time to convergence for approximately the same number of observers (see Figs. 3.4(a) and 3.7(a)). We perform similar computations both $\|\omega - \tilde{\omega}\|_{L^\infty}$ (see Figs. 3.5, 3.6, 3.8 and 3.9) and $\|\omega - \tilde{\omega}\|_{L^2}$ (see Figs. 3.2(b), 3.3(b), 3.4(b) and 3.7(b)). Here ψ denotes the stream function of the u equation with ω being the vorticity of u , and $\tilde{\cdot}$ denoting the corresponding quantity in system (1.4).

Using the numerical methods discussed above, which are described in more detail in Appendix A, we simulated dynamic observers that move in accordance with the movement styles. We found that the choice of μ varied for different movement methods. For the uniform static grid it was found that $\mu = 10$ was the optimal choice of μ . It appears that the choice of larger μ values is restricted by a CFL condition on the time-step; namely $\Delta t < 2/\mu$. We observed, similar to our previous work, [2] that highly mobile movement schemes, such as thin-sweeps, do not appear to be subject to the same restriction on μ , perhaps due to the numerically destabilizing effect of large μ -values being only briefly concentrated in any given spatial location. For simulations of sweeps, we used the value $\mu = 30$, which was unstable in the case of the static uniform grid of observers. In our simulations we observed that larger values of μ tended to lead to faster convergence in the case of thin-sweeps. In simulations, we tested thick-sweeps and found that the method did allow for larger choices of μ when the speed of the observer movement was increased to that of the thin-sweeps (from $1 \frac{\Delta x}{\Delta t}$ to $3 \frac{\Delta x}{\Delta t}$), further corroborating the “brief local destabilization” explanation discussed above. While thick-sweeps did allow for a larger choice of μ we did not observe faster convergence rates in these trials. We conjecture that, at least typically, the larger values of μ are optimal for very fast moving observer regimes in which any given point will only be observed for a short amount of time.

In preliminary simulations, we observed that certain μ values for which the static uniform grid remained stable could introduce instabilities in mobile observer schemes (except in the case of thin-sweeps and thick-sweeps). This is perhaps due to a CFL condition restricting the length-scale h , the minimum length between observers. Since h is not

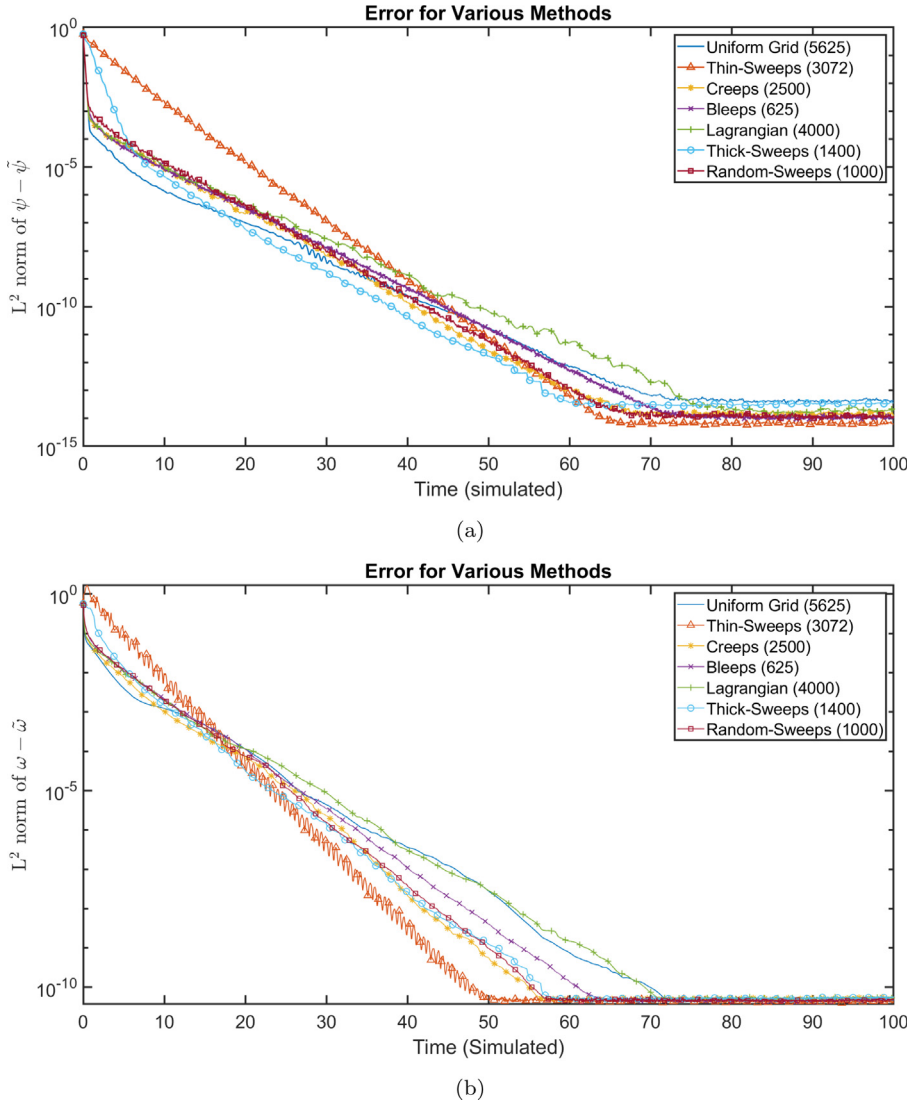


Fig. 3.2. Comparison of stream function error 3.2(a) and vorticity error 3.2(b) in the L^2 -norm over time for static uniform grid and dynamic observer strategies with number of observers optimized to ensure convergence around the time $t = 70$ (log-linear plot). See legend entries for the number of observers used in each method. $\mu = 10$ for all methods except thin-sweeps, which uses $\mu = 30$. Thick-sweeps has constant x velocity $3 \frac{\Delta x}{\Delta t}$.

restricted for most of the mobile observer paradigms, μ values should be smaller to avoid this instability. We found that the value $\mu = 10$ was optimal for the uniform static grid and did not cause instability in the mobile observers for our choice of spatial resolution $N^2 = 1024^2$. In preliminary simulations with spatial resolution $N^2 = 512^2$ we saw that the mobile observer methods required μ values smaller than the optimal choice of μ for the uniform static grid to avoid instability. However this instability was not seen in any of the higher resolution tests. While μ can potentially be further optimized, as generally higher μ values coincide with faster decay in the error, it is worth noting that our results showed faster rates of convergence using the same μ values, but with smaller numbers of observers required (see Figs. 3.2(a) and 3.3(a)).

To test these methods computationally, we first ran simulations for the uniform static grid case to establish a baseline for comparison. We found that, for our choices of parameters, a uniform grid of 5625 observers would converge to an error of $\approx 10^{-14}$ (close to machine precision) by approximately time $t = 70$ (see Figs. 3.2(a), 3.2(b),

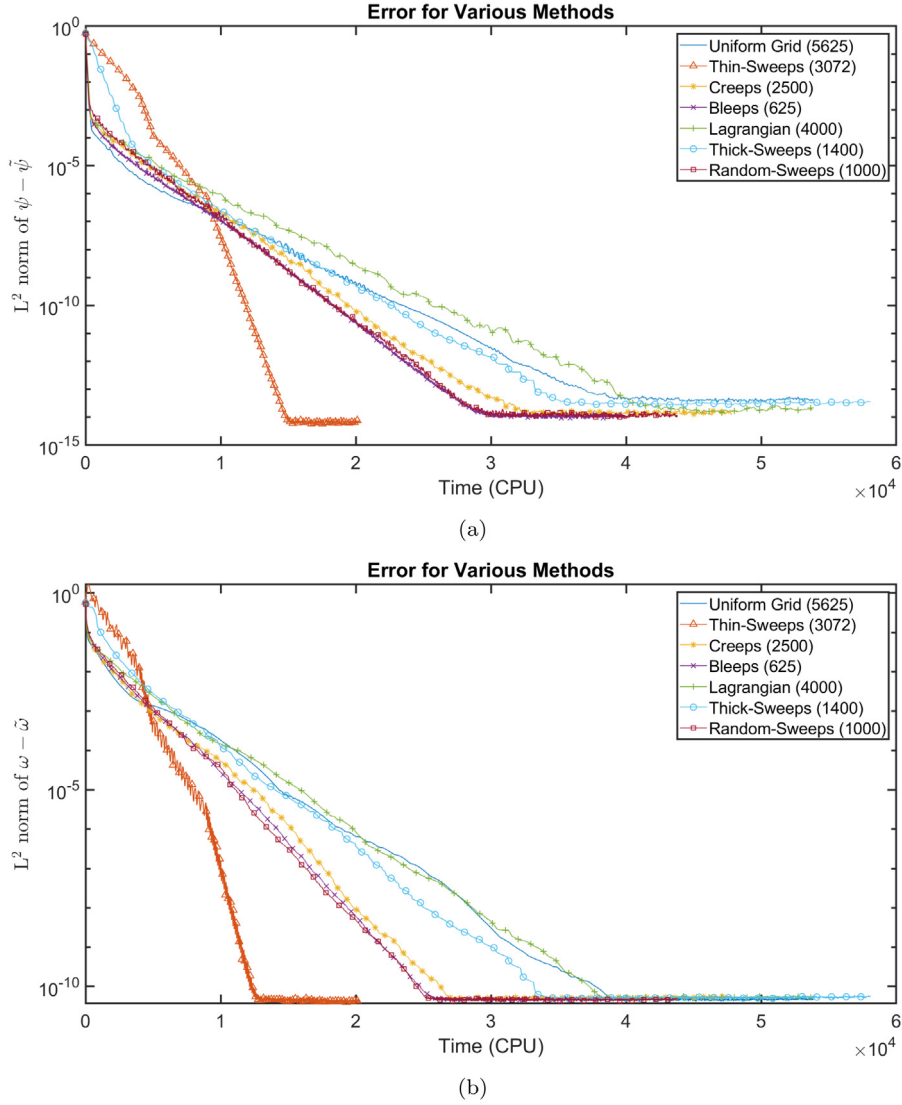


Fig. 3.3. Comparison of stream function error 3.3(a) and vorticity error 3.3(b) in the L^2 norm over CPU time for static uniform grid and dynamic observer strategies with number of observers optimized to ensure convergence around the time $t = 70$ (log-linear plot). See legend entries for the number of observers used in each method. $\mu = 10$ for all methods except thin-sweeps, which uses $\mu = 30$. Thick-sweeps has constant x velocity $3 \frac{\Delta x}{\Delta t}$.

3.3(a), 3.3(b), 3.4(a), 3.4(b), 3.5, 3.6, 3.7(a), 3.7(b), 3.8 and 3.9). We found in our simulations that approximately 2500 observers were required to achieve convergence to the true solution for the times simulated. We saw that for simulations with fewer numbers of observers, either numerical instability arose, or the simulated solution converged to an incorrect solution with the error staying approximately constant during the simulation times. Simulations with larger numbers of observers demonstrated faster rates of decay in the error, and thus we chose to use a uniform grid with 5625 observers in our trials as simulations converged relatively quickly without utilizing extremely large numbers of observers. Using this as a baseline, we varied the number of observers in the case of the bleeps, the sweeps, and the creeps in order to obtain convergence at approximately the same time ($t = 70$) for the sake of comparison. The results from these studies can be seen in Figs. 3.2(a), 3.3(a), 3.2(b), 3.3(b), 3.5 and 3.8. Here, Figs. 3.2(a), 3.2(b) and 3.5 all detail the results of these simulations plotting the error in different norms against the simulation time. Figs. 3.3(a), 3.3(b) and 3.8 plot the same errors as before, but against the CPU time to better

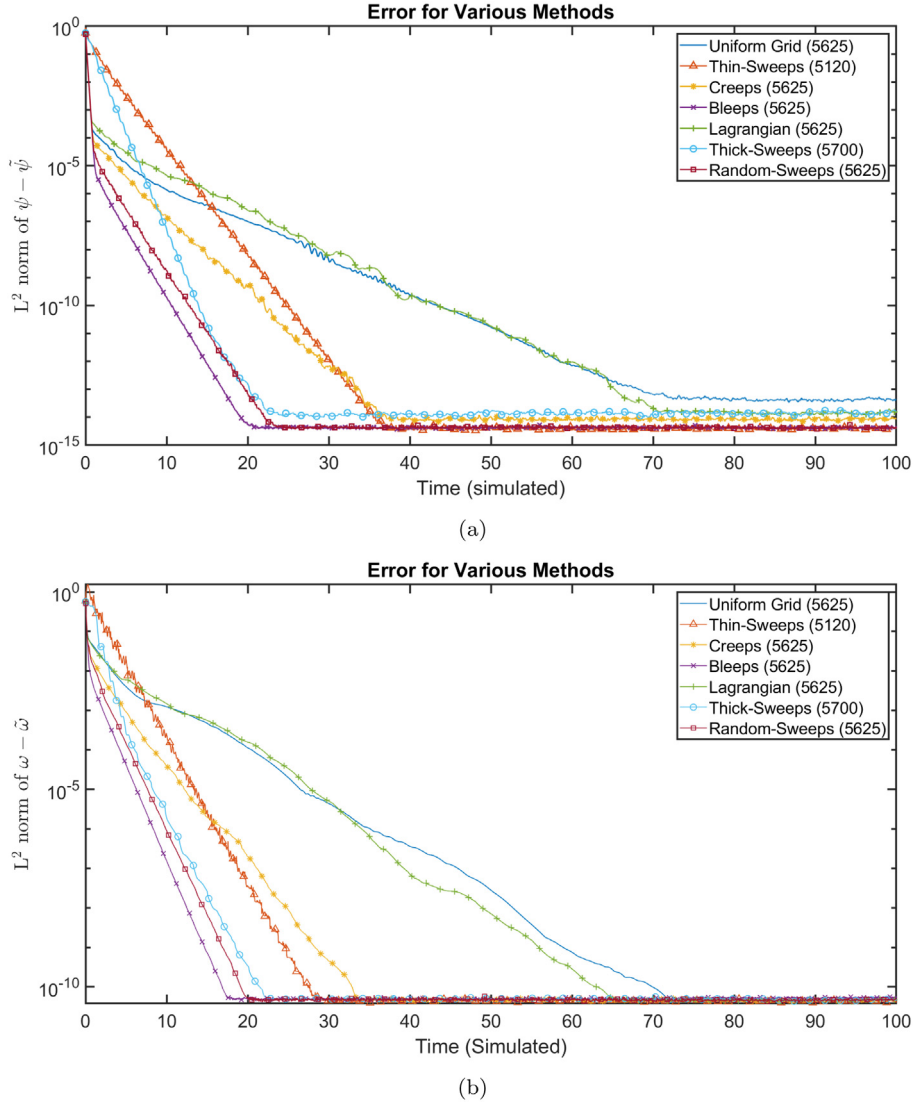


Fig. 3.4. Comparison of stream function error 3.4(a) and vorticity error 3.4(b) in the L^2 norm over time for static uniform grid and dynamic observer strategies all initialized using approximately 5625 observers (log-linear plot). See legend entries for the number of observers used in each method. $\mu = 10$ for all methods except thin-sweeps, which uses $\mu = 30$. Thick-sweeps has constant x velocity $3 \frac{\Delta x}{\Delta t}$.

judge the computational advantages of each method (although we note that different implementations, optimizations, software, hardware, data I/O speeds, etc., may have a significant effect on CPU times). As we see from these results, all three of these methods enjoy exponential convergence to the reference solution in the each norm plotted. Moreover, all three methods converge to machine precision faster than the uniform grid using significantly fewer observers. The bleeps in particular demonstrated almost an order-of-magnitude improvement in the number of observers compared to the uniform grid.

Next, we ran simulations utilizing mobile observers with each initialized to approximately 5625 observers to compare with the static uniform grid of 5625 observers. As expected, we observed that all of the mobile observer methods converged exponentially fast to the reference solution u at a faster rate than the uniform static grid. Our results can be seen in Figs. 3.4(a), 3.4(b), 3.7(a), 3.7(b), 3.6 and 3.9. Here Figs. 3.4(a), 3.4(b) and 3.6 all detail the results of these simulations plotting the error in different norms against the simulation time. Figs. 3.7(a), 3.7(b) and

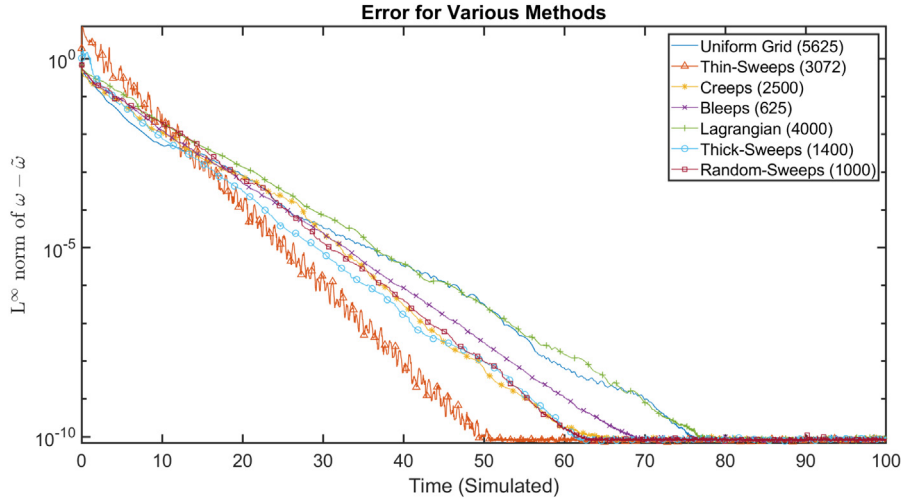


Fig. 3.5. Comparison of vorticity error in the L^∞ norm over time for static uniform grid and dynamic observer strategies with number of observers optimized to ensure convergence around the time $t = 70$ (log-linear plot). See legend entries for the number of observers used in each method. $\mu = 10$ for all methods except thin-sweeps, which uses $\mu = 30$. Thick-sweeps has constant x velocity $3 \frac{\Delta x}{\Delta t}$.

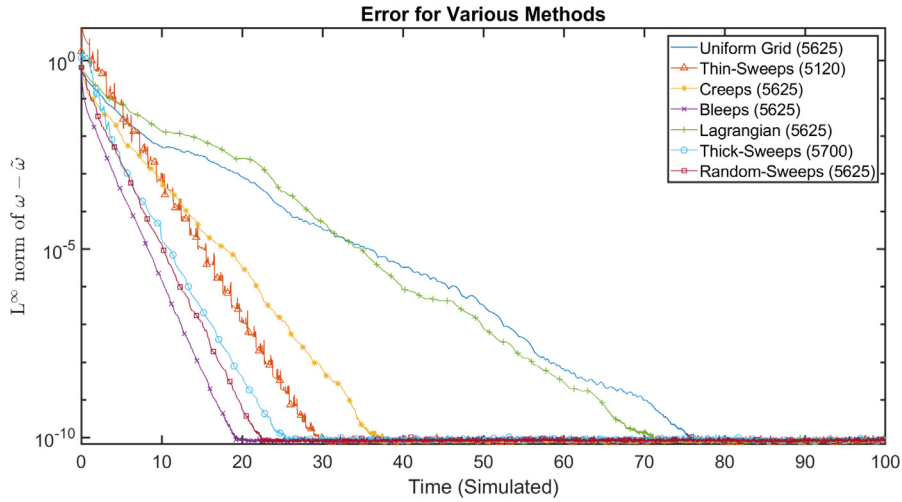


Fig. 3.6. Comparison of vorticity error in the L^∞ norm over time for static uniform grid and dynamic observer strategies all initialized using approximately 5625 observers (log-linear plot). See legend entries for the number of observers used in each method. $\mu = 10$ for all methods except thin-sweeps, which uses $\mu = 30$. Thick-sweeps has constant x velocity $3 \frac{\Delta x}{\Delta t}$.

3.9 are plotting the same errors as before but against the CPU time to better judge the computational advantages of each method.

One additional metric we used for comparison with the uniform static grid was the CPU time required to obtain convergence to machine precision. It is important to check that the extra complexity built into simulating moving observers does not make these methods prohibitively expensive to simulate when compared to the static uniform grid. It is arguable that this additional complexity is not an issue, as this will not be seen when conducting data assimilation live. In all of the simulations described above we also captured the CPU time. Graphs of the error vs the CPU time can be seen in Figs. 3.7(a), 3.7(b) and 3.9 for our simulations in which the number of observers were chosen to achieve convergence at simulation time 70, and Figs. 3.3(a), 3.3(b) and 3.8 for our simulations with each method utilizing approximately 5625 observers. Note that the CPU time required to update the locations is included in these calculations of total CPU time. This includes the time to randomly determine locations, move the

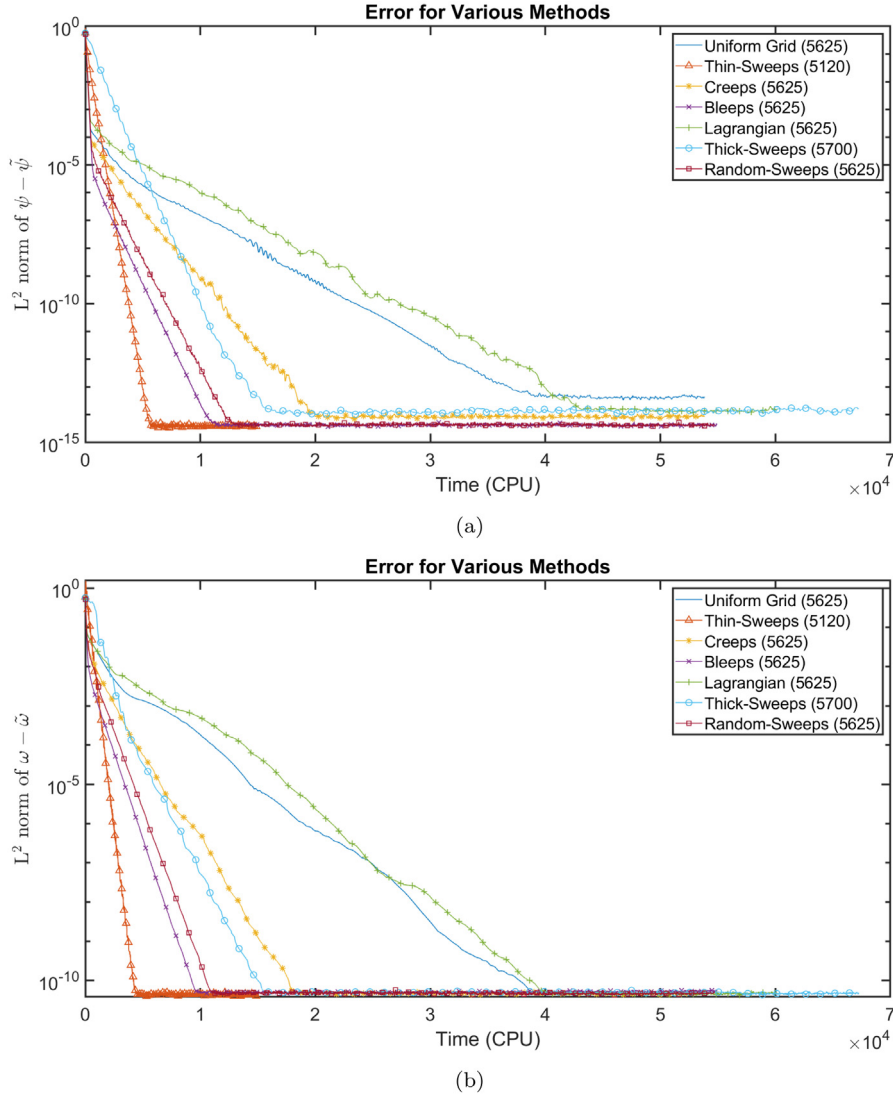


Fig. 3.7. Comparison of stream function error 3.7(a) and vorticity error 3.7(b) over CPU time for static uniform grid and dynamic observer strategies all initialized using approximately 5625 observers (log-linear plot). See legend entries for the number of observers used in each method. $\mu = 10$ for all methods except thin-sweeps, which uses $\mu = 30$. Thick-sweeps has constant x velocity $3 \frac{\Delta x}{\Delta t}$.

rectangular subdomain, and determine random walks for bleeps, sweeps, and creeps, respectively (this would likely not be needed when working with real-world data, but on the other hand, working real-world data may bring other CPU-intensive operations, such as I/O costs or off-grid interpolation on non-uniform grids). Additionally, note that since the thin-sweeps strategy use the exact solution over a small rectangular subdomain there is no interpolation performed. This leads to a very low CPU time in those figures. In the plots, we found that when the number of observers is chosen to ensure convergence by (roughly) time $t = 70$, all of the dynamic observer movement strategies except Lagrangian particles obtain convergence significantly faster in CPU time than the uniform grid case. This is perhaps due to the fewer number of observers used, simplifying the calculations of the interpolation term. Additionally, when all of the methods are simulated using approximately 5625 observers, the mobile observer methods still obtain faster convergence than the static uniform grid. In this case, it is likely due to the rate of convergence, which can be seen in Figs. 3.4(a), 3.4(b) and 3.6.

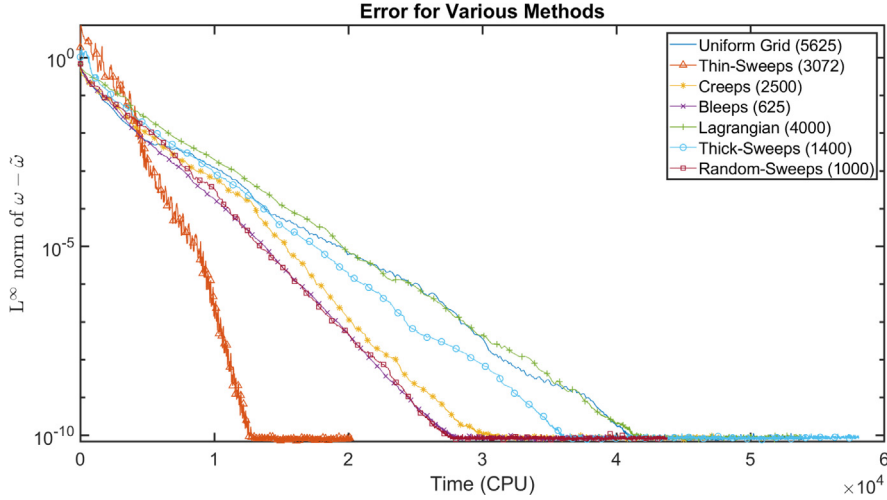


Fig. 3.8. Comparison of vorticity L^∞ error over CPU time for static uniform grid and dynamic observer strategies with number of observers optimized to ensure convergence around the time $t = 70$ (log-linear plot). See legend entries for the number of observers used in each method. $\mu = 10$ for all methods except thin-sweeps, which uses $\mu = 30$. Thick-sweeps has constant x velocity $3 \frac{\Delta x}{\Delta t}$.

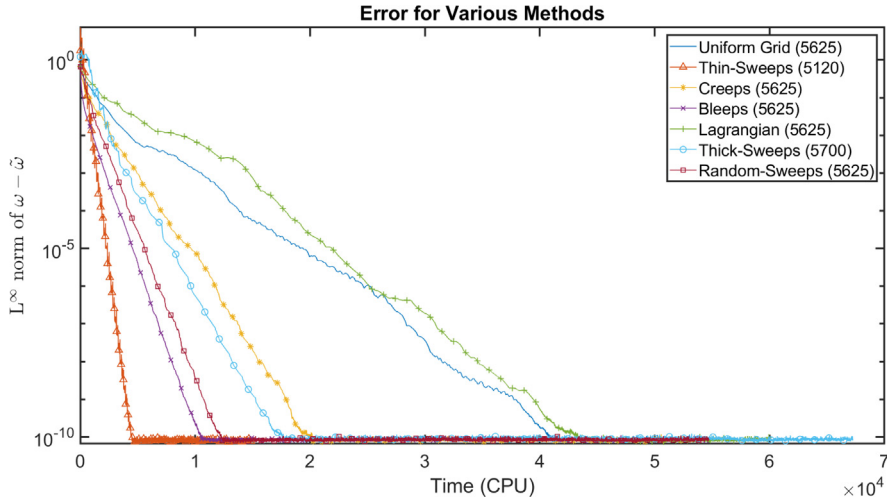


Fig. 3.9. Comparison of error over CPU time for static uniform grid and dynamic observer strategies all initialized using approximately 5625 observers (log-linear plot). See legend entries for the number of observers used in each method. $\mu = 10$ for all methods except thin-sweeps, which uses $\mu = 30$. Thick-sweeps has constant x velocity $3 \frac{\Delta x}{\Delta t}$.

We note that all of the data assimilation methods, except thin and thick sweeps, show a steep decay in the error initially until time $t \approx 1$. This steep decay is possibly due to the global distribution of observers across the domain. This steeper decay can be seen in the case of thick-sweeps when initialized with 1400 observers moving with constant velocity $3 \frac{\Delta x}{\Delta t}$ in the x direction until approximately time $t = 8$ (see Fig. 3.2(a)). Moreover we note that this steep decay was not seen for thick-sweeps with the constant velocity $1 \frac{\Delta x}{\Delta t}$ in the x direction (see Fig. 3.10(a)).

In the above simulations it appears that performing thin-sweeps is a good strategy for mobile data assimilation however it requires the exact solution in a small rectangle. All of the other movement schemes are adaptable to higher resolution with no difficulties, but with thin-sweeps for a fixed rectangular region the number of observers scales with the resolution. Physically this may be feasible if the data collection comes from a device that gathers a velocity profile along one dimension, however it is not feasible if the velocity is measured using observers placed at discrete points. This lead to the generalization of thin-sweeps to thick-sweeps, which allows for mobile observers

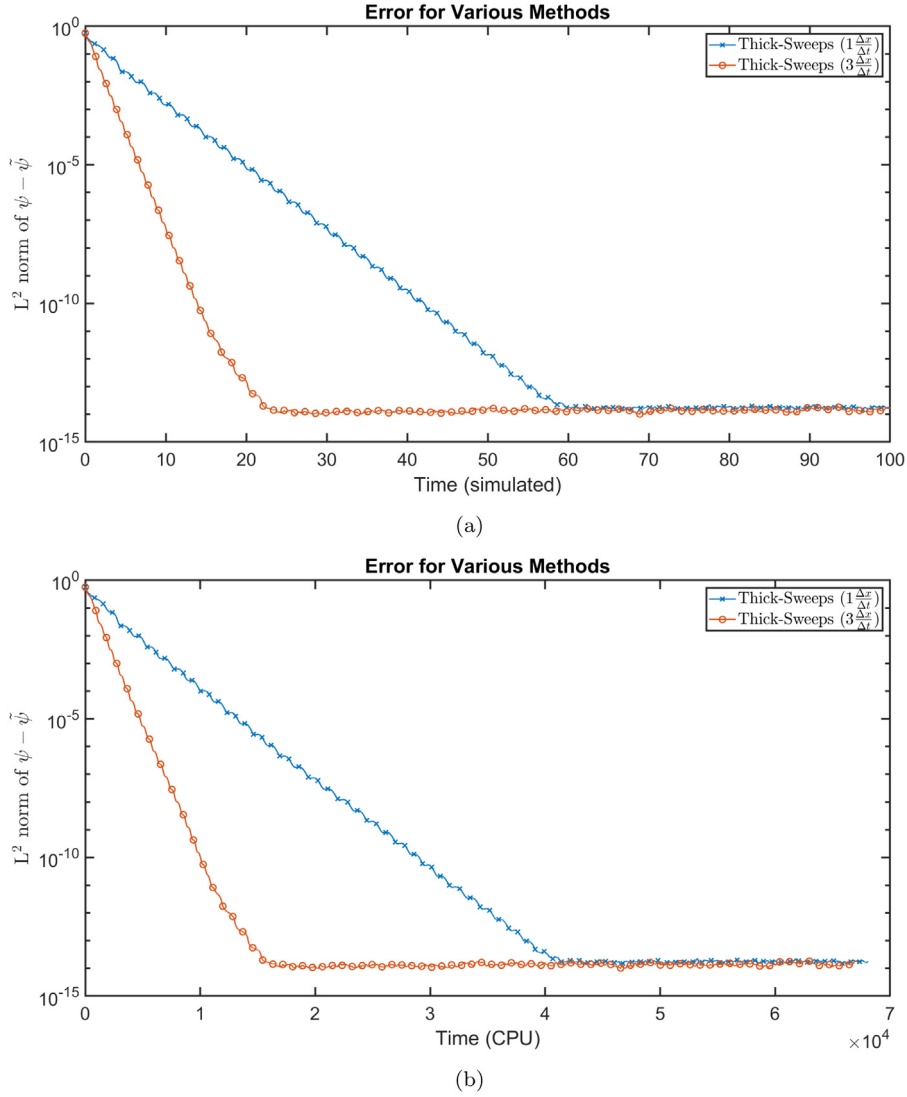


Fig. 3.10. Comparison of stream function error in the L^2 norm over simulation time 3.10(a) and over CPU time 3.10(b) for Thick-Sweep methods initialized with different speeds (log-linear plot). Both trials were initialized with the 5700 observers, $\mu = 10$, moving with constant velocity in the x direction as given in legend parentheses.

that sample locally in space and move continuously in time without requiring a number of observers proportional to the spatial resolution of the underlying numerical grid. Here, instead of having access to the full solution in a rectangle, we extend the rectangular area to cover a quarter of the domain and interpolate locally over a uniform grid of observers distributed locally on this subdomain. In our simulations we found that the velocity of the observers was an important factor in the convergence rate (see Figs. 3.10(a) and 3.10(b)). One may be able to optimize the velocity of the observers based on the size of the domain being sampled to improve convergence rates further, but we do not explore this in the present work.

4. Conclusions

Our simulations of mobile observers provide evidence that highly mobile observers such as the bleeps give the fastest convergence rates using the fewest number of observers when compared to a static uniform grid of observers. All mobile methods demonstrated convergence faster in simulation time than the static uniform grid when initialized

with the same number of observers; however, the methods that demonstrated the highest convergence rates were the bleeps and random-sweeps, both of which require some level of randomness. While sensors with random movement may be difficult to implement physically, this does demonstrate the need for observing data throughout the domain.

While all of the methods converged to machine precision faster in simulation time than the static uniform grid, the thick-sweeps and Lagrangian mobile observer movement schemes performed poorly in comparison to the rest of the mobile methods. In particular, neither thick-sweeps nor Lagrangian particle observers demonstrate the improvements in CPU time seen in the other mobile methods. In the case of thick-sweeps, the poor performance relative to the other mobile methods is likely due to lack of mobility. While thick-sweeps do move observers continuously, all of the simulations done in this study used $b = 1$ which results in slow movement across the domain. Larger b values would likely make thick-sweeps behave similarly to thin-sweep in convergence rates. Similarly, we believe that the Lagrangian particle observers perform poorly due to the slow movement of fluid particles and the tendency for particles to become trapped in vorticity structures (e.g., eddies).

An additional point of note is the performance of thin-sweeps. Recall that thin-sweeps measures the full solution within a small rectangle and inserts this directly into the v equation without interpolation of any sort. This leads to a drastic decrease in CPU time when compared to the other methods which can be seen in Figs. 3.2(a), 3.2(b), 3.3(a), 3.3(b), 3.4(a), 3.4(b), 3.5, 3.6, 3.7(a), 3.7(b), 3.8 and 3.9. Additionally, due to the requirement of measuring the full solution inside a small rectangle, the number of observers is misleading. As the resolution increases, the thin-sweep strategy will need an increasing amount of observers if the rectangular area remains the same size. The need to measure the full solution essentially means that the thin-sweep strategy requires an infinite number of observers. As currently implemented, the number of observers determines the width of the rectangular area, which for 3072 is $3072/N\Delta x = 3\Delta x$. Another restriction of thin-sweeps is that this requirement that the number of observers must be a multiple of the resolution N .

Due to the shortcomings of both thin-sweeps and thick-sweeps, we propose that a hybrid of the two methods would be practical to implement and would have some of the beneficial properties of both while being more readily implementable physically. This hybrid method would replace the rectangular area of thick-sweeps with the much smaller rectangle from thin-sweeps. Additionally, as in thin-sweeps, this hybrid method would keep the movement of the rectangular area proportional to its width. However, unlike thin-sweeps, we would actually interpolate over the area within the rectangle instead of requiring the full solution. We will explore these ideas in a future work.

From our simulations it appears that the bleeps method requires the smallest number of observers to achieve convergence comparable to the uniform static grid. In the description of the bleeps movement strategy, we noted that the locations of observers are changed at each timestep. In our previous simulations, it was found that the bleep strategy also demonstrates exponential convergence to the reference solution when observers are placed in random locations that changed after a specified amount of time. We noted that the error decreased sharply for a short amount of time after the observers were moved. In order to maintain this steep rate of decay, we chose to randomize the observer locations at every timestep. This suggests that one could optimize the randomization time, which will be the subject of a future work.

When considering the viability of mobile methods, it is important to consider the feasibility of physical implementations. In particular, one should ask if one can realize these methods using actual sensors physically moving in time. Moreover, in physical implementations, mobile observers may be more expensive than non-mobile observers and more prone to break down and potentially suffer measurement error. Therefore, factoring in cost of number of observers should be taken into account in real-world applications, but this is beyond the scope of the present work. In this study, we assumed that all observers had the same level of accuracy (namely, they are assumed to have perfect accuracy). Although studies of the AOT algorithm have been carried out in the context of noisy data [7,40], showing that the algorithm with static observations is robust with respect noisy measurements, factoring in different levels of accuracy for different observers may be crucial in real-world implementations. (For example, sensors on aircraft tend to be much more accurate, but more expensive than ground-based sensors.)

We note that, although we made an effort to follow good coding practices, no specific effort was made to optimize the computation of the various moving observer schemes, nor their interpolation, nor the nudging constants μ , except in the case of static observers. However, the moving-observer schemes still outperform the static observer case in simulation time. When measured in CPU time the moving-observer schemes outperform the static observer case except in the case of Lagrangian observers. It appears that Lagrangian tracer particles tend to get stuck in eddies, decreasing their global coverage. However, the random sampling locations of the bleeps demonstrated

significantly better error. Therefore, for example, in the case of observational devices attached to buoys, it may be worth considering mechanisms to “kick” the buoys into a different location. For example, a small motor or wind sail attached to Argo buoys could lead to significant decreases the error.

In summary, this work demonstrates the enormous potential for improving data assimilation speed, accuracy, and the number of observers required by allowing for observer to move through the domain in certain patterns, sampling wider regions of the flow. These results open the door to many new possibilities, including optimizing movement patterns (using, e.g., machine learning strategies), combining data from moving and non-moving sources (using, e.g., data from ground-based stationary sensors and fast-moving drones), heterogeneous data sources (e.g., temperature and salinity of the ocean), and many others. We will explore these ideas in future works.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Author T.E.F. acknowledges the financial support of the USDA National Institute of Food and Agriculture Hatch, United States of America project #1020768 and project #2019-67021-29312. Author A.L. acknowledges the financial support of National Science Foundation, United States of America (NSF) grants DMS-1716801 and CMMI-1953346. Author C.V. acknowledges the financial support of NSF GRFP, United States of America grant DMS-1610400.

Appendix A. Description of numerical algorithms

All simulations were carried out in Matlab (R2019b) using our own code, a version of which we have made freely available at:

https://github.com/cvictor2/AOT_CDA_2D_NSE

In all our simulations, we used pseudo-spectral methods (see, e.g., [55–57] for textbook descriptions of these methods); namely, derivatives were computed using appropriate multipliers in Fourier space, using Matlab’s implementation of the *n*-dimensional fast Fourier transform (`fft`), which relies on FFTW). Computations were done at the stream function level, where, e.g., the 2D Navier–Stokes equations (1.3) are written in the form

$$\begin{aligned} \psi_t + \Delta^{-1}(u \cdot \nabla \omega) &= \nu \Delta \psi + \Delta^{-1} \nabla^\perp \cdot f \\ u &:= \nabla^\perp \psi = \begin{pmatrix} \partial_y \psi \\ -\partial_x \psi \end{pmatrix}, \quad \omega := \partial_x u_2 - \partial_y u_1 = -\Delta \psi \end{aligned}$$

where the inverse Laplacian Δ^{-1} is taken with respect to periodic boundary conditions and the mean-free condition. To compute the nonlinear term, the standard 2/3’s dealiasing rule was applied, zeroing out the modes above wavenumber $N/3$, where N is number of gridpoints in one direction of the periodic box. Moreover, to reduce the number of `fft` operations, we used the Basdevant formula $u \cdot \nabla \omega = (\partial_x^2 - \partial_y^2)(u_1 u_2) + \partial_{xy}(u_2^2 - u_1^2)$, and one uses $\nabla \cdot u := \partial_x u_1 + \partial_y u_2 = 0$ (see [58,59] for further details on the Basdevant formula). Time stepping was carried out using an implicit/explicit scheme. Namely, the linear diffusion term was handled implicitly using an integrating factor in Fourier space, and the solution was then updated using the Adams–Bashforth-3 (AB-3) method² adapted to allow for an integrating factor. To initialize AB-3, we used Runge–Kutta-4 for the ψ solution (where there is no need to interpolate data in time), and identically zero data for the ψ solution.

The interpolation operators were created using periodic versions of the observational data combined with Matlab’s built-in `ScatteredInterpolant` function. To create periodic observational data we duplicated the observational data from the periodic box $\mathbb{T}^2 = [-\pi, \pi] \times [-\pi, \pi]$ over a larger $[-3\pi, 3\pi] \times [-3\pi, 3\pi]$ domain. This was done by taking each observer and including 8 more sister observers that share the same data at locations that

² We used AB-3 for the sake of consistency with [44], which used AB-3 for AOT simulations. We note that multi-stage methods, such as Runge–Kutta methods, are not suitable for implementation of the AOT algorithm, due to the need to interpolate data in time. See [53] for a discussion of these issues.

have been periodically shifted outside of \mathbb{T}^2 into the larger domain. For example, if there was an observer at the origin, $(0, 0)$, we would duplicate the values of this observer to the points $(0, -2\pi)$, $(0, 2\pi)$, $(-2\pi, 0)$, $(2\pi, 0)$, $(-2\pi, -2\pi)$, $(-2\pi, 2\pi)$, $(2\pi, -2\pi)$, and $(2\pi, 2\pi)$. We used all of these extra observers to preserve periodicity across the “boundary” when the `ScatteredInterpolant` generated is evaluated on the original domain \mathbb{T}^2 .

For the Lagrangian particle observers as well as the random-sweeps, there was an extra interpolation step required. Since these mobile methods were not in general aligned with the underlying spatial grid, interpolation of nearby points was required to generate the data that was observed by these methods. To generate the data for these types of observers, we interpolated using the values from the nearest four points on the spatial grid. Once the data for all of the observers was generated in this fashion, we proceeded as described previously to create the interpolation operator using the observational data.

References

- [1] *Spaceballs*, Mel Brooks (producer and director), 1987, Metro-Goldwyn-Mayer, Brookfilms.
- [2] A. Larios, C. Victor, Continuous data assimilation with a moving cluster of data points for a reaction diffusion equation: a computational study, *Commun. Comput. Phys.* 29 (4) (2021) 1273–1298, <http://dx.doi.org/10.4208/cicp.0a-2018-0315>.
- [3] A. Biswas, Z. Bradshaw, M.S. Jolly, Data assimilation for the Navier–Stokes equations using local observables, *SIAM J. Appl. Dyn. Syst.* 20 (4) (2021) 2174–2203.
- [4] A. Azouani, E. Olson, E.S. Titi, Continuous data assimilation using general interpolant observables, *J. Nonlinear Sci.* 24 (2) (2014) 277–304, <http://dx.doi.org/10.1007/s00332-013-9189-y>.
- [5] A. Azouani, E.S. Titi, Feedback control of nonlinear dissipative systems by finite determining parameters—a reaction-diffusion paradigm, *Evol. Equ. Control Theory* 3 (4) (2014) 579–594, <http://dx.doi.org/10.3934/eect.2014.3.579>.
- [6] D.A. Albanese, H.J. Nussenzveig Lopes, E.S. Titi, Continuous data assimilation for the three-dimensional Navier–Stokes- α model, *Asymptotic Anal.* 97 (1–2) (2016) 139–164.
- [7] H. Bessaih, E. Olson, E.S. Titi, Continuous data assimilation with stochastically noisy data, *Nonlinearity* 28 (3) (2015) 729–753, <http://dx.doi.org/10.1088/0951-7715/28/3/729>.
- [8] A. Biswas, C. Foias, C.F. Mondaini, E.S. Titi, Downscaling data assimilation algorithm with applications to statistical solutions of the Navier–Stokes equations, in: *Annales De L’Institut Henri Poincaré C, Analyse Non LinÉAire*, Elsevier, 2019, pp. 295–326.
- [9] A. Biswas, V.R. Martinez, Higher-order synchronization for a data assimilation algorithm for the 2D Navier–Stokes equations, *Nonlinear Anal. RWA* 35 (2017) 132–157, <http://dx.doi.org/10.1016/j.nonrwa.2016.10.005>.
- [10] A. Biswas, R. Price, Continuous data assimilation for the three-dimensional Navier–Stokes equations, *SIAM J. Math. Anal.* 53 (6) (2021) 6697–6723.
- [11] E. Carlson, J. Hudson, A. Larios, Parameter recovery for the 2 dimensional Navier-Stokes equations via continuous data assimilation, *SIAM J. Sci. Comput.* 42 (1) (2020) A250–A270, <http://dx.doi.org/10.1137/19M1248583>.
- [12] E. Carlson, A. Larios, Sensitivity analysis for the 2D Navier-Stokes equations with applications to continuous data assimilation, *J. Nonlinear Sci.* 31 (5) (2021) <http://dx.doi.org/10.1007/s00332-021-09739-9>, Paper No. 84, 30.
- [13] E. Celik, E. Olson, E.S. Titi, Spectral filtering of interpolant observables for a discrete-in-time downscaling data assimilation algorithm, *SIAM J. Appl. Dyn. Syst.* 18 (2) (2019) 1118–1142, <http://dx.doi.org/10.1137/18M1218480>.
- [14] N. Chen, Y. Li, E. Lunasin, An efficient continuous data assimilation algorithm for the Sabra shell model of turbulence, *Chaos* 31 (10) (2021) 103123, <http://dx.doi.org/10.1063/5.0057421>.
- [15] A.E. Diegel, L.G. Rebholz, Continuous data assimilation and long-time accuracy in a C^0 interior penalty method for the Cahn-Hilliard equation, [arXiv:2106.14744](https://arxiv.org/abs/2106.14744), submitted for publication.
- [16] Y.J. Du, M.-C. Shiue, Analysis and computation of continuous data assimilation algorithms for Lorenz 63 system based on nonlinear nudging techniques, *J. Comput. Appl. Math.* 386 (2021) 113246, <http://dx.doi.org/10.1016/j.cam.2020.113246>.
- [17] A. Farhat, M.S. Jolly, E.S. Titi, Continuous data assimilation for the 2D Bénard convection through velocity measurements alone, *Physica D* 303 (2015) 59–66, <http://dx.doi.org/10.1016/j.physd.2015.03.011>.
- [18] A. Farhat, E. Lunasin, E.S. Titi, Abridged continuous data assimilation for the 2D Navier–Stokes equations utilizing measurements of only one component of the velocity field, *J. Math. Fluid Mech.* 18 (1) (2016) 1–23.
- [19] A. Farhat, E. Lunasin, E.S. Titi, Data assimilation algorithm for 3D Bénard convection in porous media employing only temperature measurements, *J. Math. Anal. Appl.* 438 (1) (2016) 492–506.
- [20] A. Farhat, E. Lunasin, E.S. Titi, On the Charney conjecture of data assimilation employing temperature measurements alone: the paradigm of 3D planetary geostrophic model, *Math. Clim. Weather Forecast.* 2 (1) (2016).
- [21] A. Farhat, E. Lunasin, E.S. Titi, Continuous data assimilation for a 2D Bénard convection system through horizontal velocity measurements alone, *J. Nonlinear Sci.* (2017) 1–23, <http://dx.doi.org/10.1007/s00332-017-9360-y>.
- [22] C. Foias, C.F. Mondaini, E.S. Titi, A discrete data assimilation scheme for the solutions of the two-dimensional Navier–Stokes equations and their statistics, *SIAM J. Appl. Dyn. Syst.* 15 (4) (2016) 2109–2142, <http://dx.doi.org/10.1137/16M1076526>.
- [23] K. Foyash, M.S. Dzholli, R. Kravchenko, E.S. Titi, A unified approach to the construction of defining forms for a two-dimensional system of Navier–Stokes equations: the case of general interpolating operators, *Uspekhi Mat. Nauk* 69 (2(416)) (2014) 177–200.
- [24] B. García-Archilla, J. Novo, E.S. Titi, Uniform in time error estimates for a finite element method applied to a downscaling data assimilation algorithm for the Navier-Stokes equations, *SIAM J. Numer. Anal.* 58 (1) (2020) 410–429, <http://dx.doi.org/10.1137/19M1246845>.

- [25] B. García-Archilla, J. Novo, Error analysis of fully discrete mixed finite element data assimilation schemes for the Navier-Stokes equations, *Adv. Comput. Math.* 46 (4) (2020) <http://dx.doi.org/10.1007/s10444-020-09806-x>, Paper No. 61, 33.
- [26] M. Gardner, A. Larios, L.G. Rebholz, D. Vargun, C. Zervas, Continuous data assimilation applied to a velocity-vorticity formulation of the 2D Navier-Stokes equations, *Electron. Res. Arch.* 29 (3) (2021) 2223–2247, <http://dx.doi.org/10.3934/era.2020113>.
- [27] H.A. Ibdah, C.F. Mondaini, E.S. Titi, Fully discrete numerical schemes of a data assimilation algorithm: uniform-in-time error estimates, *IMA J. Numer. Anal.* 40 (4) (2020) 2584–2625.
- [28] M.S. Jolly, V.R. Martinez, E.J. Olson, E.S. Titi, Continuous data assimilation with blurred-in-time measurements of the surface quasi-geostrophic equation, *Chinese Ann. Math. Ser. B* 40 (5) (2019) 721–764, <http://dx.doi.org/10.1007/s11401-019-0158-0>.
- [29] M.S. Jolly, V.R. Martinez, E.S. Titi, A data assimilation algorithm for the subcritical surface quasi-geostrophic equation, *Adv. Nonlinear Stud.* 17 (1) (2017) 167–192, <http://dx.doi.org/10.1515/ans-2016-6019>.
- [30] M.S. Jolly, T. Sadigov, E.S. Titi, A determining form for the damped driven nonlinear Schrödinger equation—Fourier modes case, *J. Differential Equations* 258 (8) (2015) 2711–2744, <http://dx.doi.org/10.1016/j.jde.2014.12.023>.
- [31] A. Larios, Y. Pei, Approximate continuous data assimilation of the 2D Navier-Stokes equations via the voigt-regularization with observable data, *Evol. Equ. Control Theory* 9 (3) (2020) 733–751, <http://dx.doi.org/10.3934/eect.2020031>.
- [32] A. Larios, C. Victor, Improving convergence rates of continuous data assimilation for 2D Navier-Stokes using observations that are sparse in space and time, (preprint).
- [33] P.A. Markowich, E.S. Titi, S. Trabelsi, Continuous data assimilation for the three-dimensional brinkman-forchheimer-extended Darcy model, *Nonlinearity* 29 (4) (2016) 1292–1328, <http://dx.doi.org/10.1088/0951-7715/29/4/1292>.
- [34] C.F. Mondaini, E.S. Titi, Uniform-in-time error estimates for the postprocessing Galerkin method applied to a data assimilation algorithm, *SIAM J. Numer. Anal.* 56 (1) (2018) 78–110.
- [35] B. Pachev, J.P. Whitehead, S.A. McQuarrie, Concurrent multi-parameter learning demonstrated on the Kuramoto–Sivashinsky equation, 2021.
- [36] Y. Pei, Continuous data assimilation for the 3D primitive equations of the ocean, *Commun. Pure Appl. Anal.* 18 (2) (2019) 643–661, <http://dx.doi.org/10.3934/cpaa.2019032>.
- [37] L.G. Rebholz, C. Zervas, Simple and efficient continuous data assimilation of evolution equations via algebraic nudging, *Numer. Methods Partial Differ. Equations* (2021) 1–25, <http://dx.doi.org/10.1002/num.22751>.
- [38] C. Zervas, L.G. Rebholz, M. Schneier, T. Iliescu, Continuous data assimilation reduced order models of fluid flow, *Comput. Methods Appl. Mech. Engrg.* 357 (2019) 112596, 18, <http://dx.doi.org/10.1016/j.cma.2019.112596>.
- [39] M.U. Altaf, E.S. Titi, O.M. Knio, L. Zhao, M.F. McCabe, I. Hoteit, Downscaling the 2D benard convection equations using continuous data assimilation, *Comput. Geosci.* 21 (3) (2017) 393–410.
- [40] E. Carlson, J. Hudson, A. Larios, V.R. Martinez, E. Ng, J.P. Whitehead, Dynamically learning the parameters of a chaotic system using partial observations, *arXiv:2108.08354*, submitted for publication.
- [41] E. Carlson, L. Van Roekel, M. Petersen, H.C. Godinez, A. Larios, CDA algorithm implemented in MPAS-O to improve eddy effects in a mesoscale simulation, 2021, <http://dx.doi.org/10.1002/essoar.10507378.1>, submitted for publication.
- [42] P. Clark Di Leonì, A. Mazzino, L. Biferale, Inferring flow parameters and turbulent configuration with physics-informed data assimilation and spectral nudging, *Phys. Rev. Fluids* 3 (10) (2018) 104604.
- [43] S. Desamsetti, H. Dasari, S. Langodan, O. Knio, I. Hoteit, E.S. Titi, Efficient dynamical downscaling of general circulation models using continuous data assimilation, *Q. J. R. Meteorol. Soc.* (2019) <http://dx.doi.org/10.1002/qj.3612>.
- [44] M. Gesho, E. Olson, E.S. Titi, A computational study of a data assimilation algorithm for the two-dimensional Navier–Stokes equations, *Commun. Comput. Phys.* 19 (4) (2016) 1094–1110.
- [45] A. Larios, Y. Pei, Nonlinear continuous data assimilation *arXiv:1703.03546*, submitted for publication.
- [46] A. Larios, L.G. Rebholz, C. Zervas, Global in time stability and accuracy of IMEX-FEM data assimilation schemes for Navier-Stokes equations, *Comput. Methods Appl. Mech. Engrg.* 345 (2019) 1077–1093, <http://dx.doi.org/10.1016/j.cma.2018.09.004>.
- [47] E. Lunasin, E.S. Titi, Finite determining parameters feedback control for distributed nonlinear dissipative systems—a computational study, *Evol. Equ. Control Theory* 6 (4) (2017) 535–557, <http://dx.doi.org/10.3934/eect.2017027>.
- [48] P. Constantin, C. Foias, *Navier–Stokes Equations*, in: *Chicago Lectures in Mathematics*, University of Chicago Press, Chicago, IL, 1988, p. x+190.
- [49] R. Temam, *Navier–Stokes Equations And Nonlinear Functional Analysis*, Second, in: *CBMS-NSF Regional Conference Series in Applied Mathematics*, vol. 66, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1995, p. xiv+141.
- [50] A. Larios, C. Victor, Intermittent data assimilation for the 2D Navier-Stokes equations, 2021, in preparation.
- [51] A. Biswas, Z. Bradshaw, M.S. Jolly, Data assimilation for the Navier-Stokes equations using local observables, *SIAM J. Appl. Dyn. Syst.* 20 (4) (2021) 2174–2203, <http://dx.doi.org/10.1137/20M136058X>.
- [52] Y. Chen, COVID-19 pandemic imperils weather forecast, *Geophys. Res. Lett.* 47 (15) (2020) e2020GL088613.
- [53] E. Olson, E.S. Titi, Determining modes and grashof number in 2D turbulence: a numerical case study, *Theor. Comput. Fluid Dyn.* 22 (5) (2008) 327–339.
- [54] E. Olson, E.S. Titi, Determining modes for continuous data assimilation in 2D turbulence, *J. Stat. Phys.* 113 (5–6) (2003) 799–840, <http://dx.doi.org/10.1023/A:1027312703252>, *Progress in statistical hydrodynamics* (Santa Fe, NM, 2002).
- [55] C. Canuto, M.Y. Hussaini, A. Quarteroni, T.A. Zang, *Spectral Methods*, in: *Scientific Computation*, Springer-Verlag, Berlin, 2006, p. xxii+563, *Fundamentals in single domains*.
- [56] R. Peyret, *Spectral Methods for Incompressible Viscous Flow*, Vol. 148, Springer Science & Business Media, 2013.
- [57] J. Shen, T. Tang, L.-L. Wang, *Spectral Methods*, in: *Springer Series in Computational Mathematics*, vol. 41, Springer, Heidelberg, 2011, p. xvi+470, <http://dx.doi.org/10.1007/978-3-540-71041-7>, *Algorithms, analysis and applications*.

- [58] C. Basdevant, Technical improvements for direct numerical simulation of homogeneous three-dimensional turbulence, *J. Comput. Phys.* 50 (2) (1983) 209–214.
- [59] P. Emami, J.C. Bowman, On the global attractor of 2D incompressible turbulence with random forcing, *J. Differential Equations* 264 (6) (2018) 4036–4066, <http://dx.doi.org/10.1016/j.jde.2017.12.004>.