Predicting Flow Reversals in a Computational Fluid Dynamics Simulated Thermosyphon using Data Assimilation

Andrew Reagan,[1] Yves Dubief,[2] Peter Sheridan Dodds,[1] and Christopher M. Danforth[1]

[1]*Department of Mathematics & Statistics, Vermont Complex Systems Center, Computational Story Lab, & the Vermont Advanced Computing Core, The University of Vermont, Burlington, VT 05401*

[2]*School of Engineering, Vermont Complex Systems Center & the Vermont Advanced Computing Core, The University of Vermont, Burlington, VT 05401*

(Dated: September 23, 2015)

A thermal convection loop is a circular chamber filled with water, heated on the bottom half and cooled on the top half. With sufficiently large forcing of heat, the direction of fluid flow in the loop oscillates chaotically, forming an analog to the Earth's weather. As is the case for state-of-the-art weather models, we only observe the statistics over a small region of state space, making prediction difficult. To overcome this challenge, data assimilation (DA) methods, and specifically ensemble methods, use the computational model itself to estimate the uncertainty of the model to optimally combine these observations into an initial condition for predicting the future state. First, we build and verify four distinct DA methods, and then we perform a twin model experiment with the computational fluid dynamics simulation of the loop using the Ensemble Transform Kalman Filter (ETKF) to assimilate observations and predict flow reversals. We show that using adaptively shaped localized covariance outperforms static localized covariance with the ETKF, and allows for the use of less observations in predicting flow reversals. We also show that a Dynamic Mode Decomposition (DMD) of the temperature and velocity fields recovers the low dimensional system underlying reversals, finding specific modes which together are predictive of reversal direction.

PACS numbers:

I. INTRODUCTION

Prediction of the future state of complex systems is integral to the functioning of our society. Some of these systems include weather [18], health [13], the economy [35], marketing [2], and engineering [34]. For weather in particular, this prediction is made using supercomputers integrating numerical weather models, projecting our current best guess of the weather into the future. The accuracy of these predictions depend on the accuracy of the models themselves, and the quality of our knowledge of the current state of the atmosphere.

Model accuracy has improved with better meteorological understanding of weather processes and advances in computing technology [3]. To solve the initial value problem, techniques developed over the past 50 years are now broadly known as *data assimilation*. Formally, data assimilation is the process of using all available information, including short-range model forecasts and physical *observations, to estimate the current state of a system as* accurately as possible [39]. This best-guess of the current state is often referred to as the *analysis* state.

We employ a fluid dynamics experiment as a test-bed for improving numerical weather prediction algorithms, focusing specifically on data assimilation methods. This approach is inspired by the historical development of current methodologies, and provides a tractable system for rigorous analysis. The experiment is a thermal convection loop, which by design simplifies our problem into the prediction of convection. The dynamics of thermal convection loops have been explored under both periodic [25] and chaotic [6, 7, 9, 10, 15, 16, 22, 33, 38–40] regimes. A full characterization of the computational behavior of a loop under flux boundary conditions by Louisos et. al. describes four regimes: chaotic convection with reversals, high Ra aperiodic stable convection, steady stable convection, and conduction/quasi-conduction [28]. For the remainder of this work, we focus on the chaotic flow regime.

Computational simulations of the thermal convection loop are performed with the open-source finite volume C++ library OpenFOAM [21]. The open-source nature of this software enables its integration with the data assimilation framework that this work provides.

II. PHYSICAL EXPERIMENT AND COMPUTATIONAL MODEL

The thermosyphon, a type of natural convection loop or non-mechanical heat pump, can be likened to a toy model of climate [17].

The reduced order system describing a thermal convection loop was originally derived by Gorman [16] and Ehrhard and Müller [10]. Here we present this three dimensional system in non-dimensionalized form. In Appendix B, we present a more complete derivation of these equations, following the derivation of Harris [17]. For $x_{\{1,2,3\}}$ the mean fluid velocity, temperature difference ΔT_{3-9}, and deviation from conductive temperature

Typeset by REVTEX

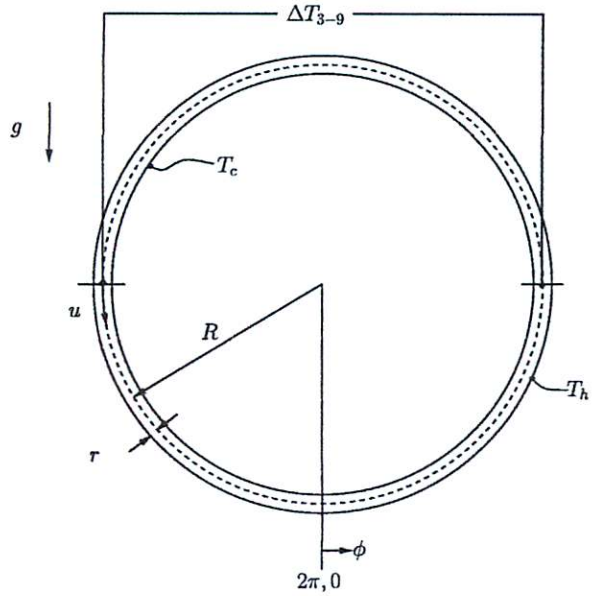

FIG. 1: Schematic of the experimental, and computational, setup from Harris et al. (2012). The loop radius is given by R and inner radius by r. The top temperature is labeled T_c and bottom temperature T_h, gravity g is defined downward, the angle ϕ is prescribed from the 6 o'clock position, and *temperature difference* ΔT_{3-9} *is labeled.*

profile, respectively, these equations are:

$$\frac{dx_1}{dt} = \alpha(x_2 - x_1), \tag{1}$$

$$\frac{dx_2}{dt} = \beta x_1 - x_2(1 + Kh(|x_1|)) - x_1 x_3, \tag{2}$$

$$\frac{dx_3}{dt} = x_1 x_2 - x_3(1 + Kh(|x_1|)). \tag{3}$$

The parameters α and β, along with scaling factors for time and each model variable can be fit to data using standard parameter estimation techniques.

Operated by Dave Hammond, UVM's Scientific Electronics Technician, the experimental thermosyphons access the chaotic regime of state space found in the principled governing equations. We quote the detailed setup from Darcy Glenn's undergraduate thesis [14]:

> The [thermosyphon] was a bent semi-flexible plastic tube with a 10-foot heating rope wrapped around the bottom half of the upright circle. The tubing used was light-transmitting clear THV from McMaster-Carr, with an inner diameter of 7/8 inch, a wall thickness of 1/16 inch, and a maximum operating temperature of 200F. The outer diameter of the circular thermosyphon was 32.25 inches. This produced a ratio of about 1:36 inner tubing radius to outside thermosyphon radius. There were 1 inch 'windows' when the heating cable was coiled in a helix pattern around the outside of the tube, so the heating is not exactly uniform. The bottom half was then insulated using aluminum foil, which allowed fluid in the bottom half to reach 176F. A forcing of 57 V, or 105 Watts, was required for the heating cable so that chaotic motion was observed. Temperature was measured at the 3 o'clock and 9 o'clock positions using unsheathed copper thermocouples from Omega.

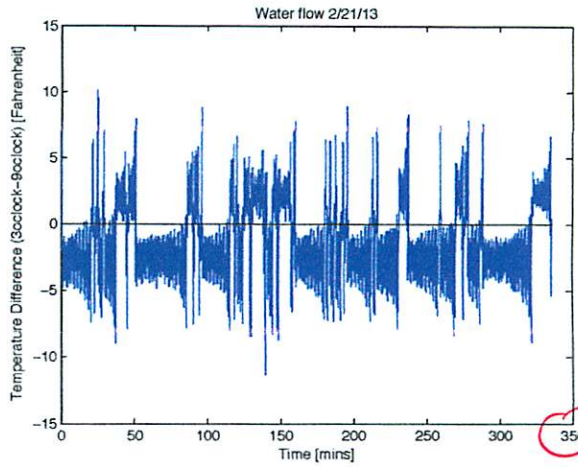

FIG. 2: A time series of the physical thermosyphon, from the Undergraduate Honor's Thesis of Darcy Glenn [14]. The temperature difference (plotted) is taken as the difference between temperature sensors in the 3 and 9 o'clock positions. The sign of the temperature difference indicates the flow direction, where positive values are clockwise flow.

We first test our ability to predict this experimental thermosyphon using synthetic data.

A. Computational Setup

We consider the incompressible Navier-Stokes equations with the Boussinesq approximation to model the flow of water inside a thermal convection loop. For brevity, we omit the equations themselves, and include them in the Appendix. The solver in OpenFOAM that we use, with some modification, is `buoyantBoussinesqPimpleFoam`. Solving is accomplished by the Pressure-Implicit Split Operator (PISO) algorithm [20]. Modification of the code was necessary for laminar operation.

Both 2-dimensional and 3-dimensional meshes were created using OpenFOAM's native meshing utility `blockMesh`. After creating a mesh, we consider refining the mesh near the walls to capture boundary layer phenomena and renumbering the mesh for solving speed. To refine the mesh near walls, we used the

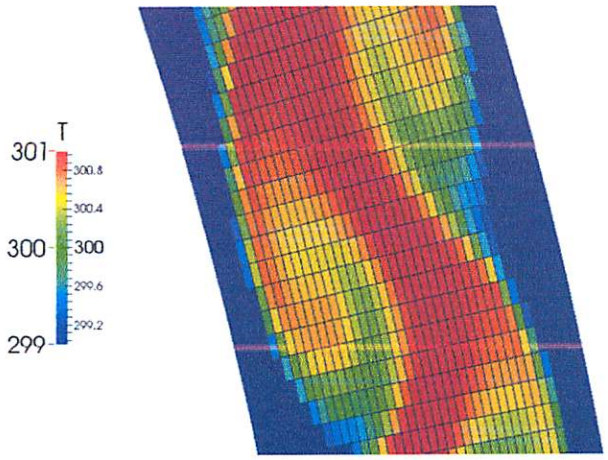

FIG. 3: A snapshot of the mesh used for CFD simulations. Shown is an initial stage of heating for a fixed value boundary condition, 2D, laminar simulation with a mesh of 40000 cells including wall refinement with walls heated at 340K on the bottom half and cooled to 290K on the top half.

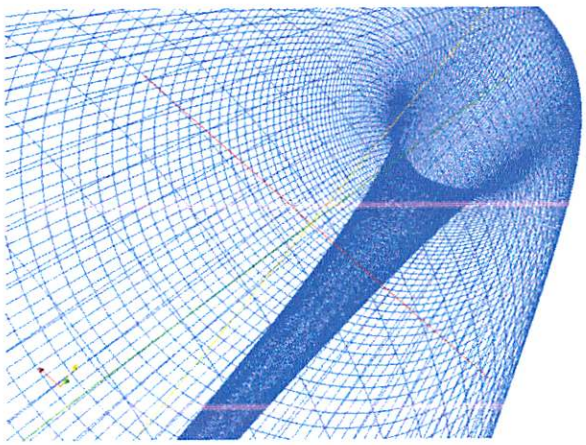

FIG. 4: The 3D mesh viewed as a wire-frame from within. Here there are 900 cells in each slice (not shown), for a total mesh size of 81,000 cells. Simulations using this computational mesh are prohibitively expensive for use in a real time ensemble forecasting system, but are possible offline.

`refineWallMesh` utility. Lastly, the mesh is renumbered using the `renumberMesh` utility, which implements the Cuthill-McKee algorithm to minimize the adjacency matrix bandwidth, defined as the maximum distance from diagonal of nonzero entry. The 2D mesh contains *40,000 points (40 across the diameter and 1000 around).*

Available boundary conditions (BCs) that were found to be stable in OpenFOAM's solver were constant gradient, fixed value conditions, and turbulent heat flux. Simulations with a fixed flux BC is implemented through the `externalWallHeatFluxTemperature` library were unstable and resulted in physically unrealistic results. Constant gradient simulations were stable, but the behavior was empirically different from our physical system. While it is possible that a fixed value BC is acceptable due to the thermal diffusivity and thickness of the walls of the experimental setup, we find that this is also inadequate. We employ the third-party library `groovyBC` to use a gradient condition that computes a flux using a fixed external temperature and fixed wall heat transfer coefficient.

With the mesh, BCs, and solver chosen, we now simulate the flow. From the data of T, ϕ, u, v, w and p that are saved at each timestep, we extract the mass flow rate and average temperature at the 12, 3, 6 and 9 o'clock positions on the loop. Since ϕ is saved as a face-value flux, we compute the mass flow rate over the cells i of top (12 o'clock) slice as

$$\sum_i \phi_{f(i)} \cdot v_i \cdot \rho_i \tag{4}$$

where $f(i)$ corresponds the face perpendicular to the loop angle at cell i and ρ is reconstructed from the Boussinesq approximation $\rho = \rho_{\text{ref}}(1 - \beta(T - T_{\text{ref}}))$.

III. METHODS

A. Data Assimilation

Tests of the data assimilation algorithms described here are performed with the Lorenz '63 system, which is analogous to the above equations with Lorenz's $\beta = 1$, and $K = 0$. The canonical choices of $\sigma = 10, \beta = 8/3$ and $\rho = 28$ produce the well known butterfly attractor, and are used for all examples here. From these tests, we have found the optimal data assimilation parameters (inflation factors) for predicting time series with this system. Having done so, we focus our efforts on making prediction using computational fluid dynamics models.

We first implement the 3D-Var filter. Simply put, 3D-Var is the variational (cost-function) approach to finding the analysis. It has been shown that 3D-var solves the same statistical problem as optimal interpolation (OI) [27]. The usefulness of the variational approach comes from the computational efficiency, when solved with an iterative method. The multivariate 3D-Var is thus finding the $\mathbf{x}_a$ that minimizes the cost function

$$J(\mathbf{x}) = (\mathbf{x}-\mathbf{x}_b)^T\mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b)+(\mathbf{y}_o+H(\mathbf{x}))^T\mathbf{R}(\mathbf{y}_o-H(\mathbf{x})). \tag{5}$$

Next, we implemented the "gold-standard" Extended Kalman Filter (EKF). The tangent linear model (TLM) is precisely the model (written as a matrix) that transforms a perturbation at time t to a perturbation at time $t + \Delta t$, analytically equivalent to the Jacobian of the model. Using the notation of Kalnay [23], this amounts to making a forecast with the nonlinear model M, and updating the error covariance matrix $\mathbf{P}$ with the TLM L, and adjoint model L^T:

$$\mathbf{x}^f(t_i) = M_{i-1}[\mathbf{x}^a(t_{i-1})]$$
$$\mathbf{P}^f(t_i) = L_{i-1}\mathbf{P}^a(t_{i-1})L_{i-1}^T + \mathbf{Q}(t_{i-1})$$

where $\mathbf{Q}$ is the noise covariance matrix (model error). In the experiments with Lorenz '63 presented in this section, $\mathbf{Q} = 0$ since our model is perfect. In NWP, $\mathbf{Q}$ must be approximated, e.g., using statistical moments on the analysis increments [8, 26].

The analysis step is then written as (for H the observation operator):

$$\mathbf{x}^a(t_i) = \mathbf{x}^f(t_i) + \mathbf{K}_i\mathbf{d}_i \quad (6)$$
$$\mathbf{P}^a(t_i) = (\mathbf{I} - \mathbf{K}_i\mathbf{H}_i)\mathbf{P}^f(t_i) \quad (7)$$

where

$$\mathbf{d}_i = \mathbf{y}_i^o - \mathbf{H}[x^f(t_i)]$$

is the innovation. The Kalman gain matrix is computed to minimize the analysis error covariance P_i^a as

$$\mathbf{K}_i = \mathbf{P}^f(t_i)\mathbf{H}_i^T[\mathbf{R}_i + \mathbf{H}_i\mathbf{P}^f(t_i)\mathbf{H}^T]^{-1}$$

where $\mathbf{R}_i$ is the observation error covariance. Since we are making observations of the truth with random normal errors of standard deviation ϵ, the observational error covariance matrix $\mathbf{R}$ is a diagonal matrix with ϵ along the diagonal. The most difficult (and most computationally expensive) part of the EKF is deriving and integrating the TLM. For this reason, the EKF is not used operationally, and later we will turn to statistical approximations of the EKF using ensembles of model forecasts. With our CFD model we have no such TLM, and we present more detail on the TLM approachas applicable to the Lorenz '63 system in Appendix C.

The computational cost of the EKF is mitigated through the approximation of the error covariance matrix $\mathbf{P}_f$ from the model itself, without the use of a TLM. One such approach is the use of a forecast ensemble, where a collection of models (ensemble members) are used to statistically sample model error propagation. With ensemble members spanning the model analysis error space, the forecasts of these ensemble members are then used to estimate the model forecast error covariance.

The only difference between this approach and the EKF, in general, is that the forecast error covariance $\mathbf{P}^f$ is computed from the ensemble members, without the need for a tangent linear model:

$$\mathbf{P}^f \approx \frac{1}{K-2}\sum_{k\neq l}\left(\mathbf{x}_k^f - \overline{\mathbf{x}}_l^f\right)\left(\mathbf{x}_k^f - \overline{\mathbf{x}}_l^f\right)^T.$$

The ETKF introduced by Bishop is one type of square root filter, and we present it here to provide background for the formulation of the LETKF [4]. For a square root filter in general, we begin by writing the covariance matrices as the product of their matrix square roots. Because $\mathbf{P}_a\mathbf{P}_f$ are symmetric positive-definite (by definition), we can write

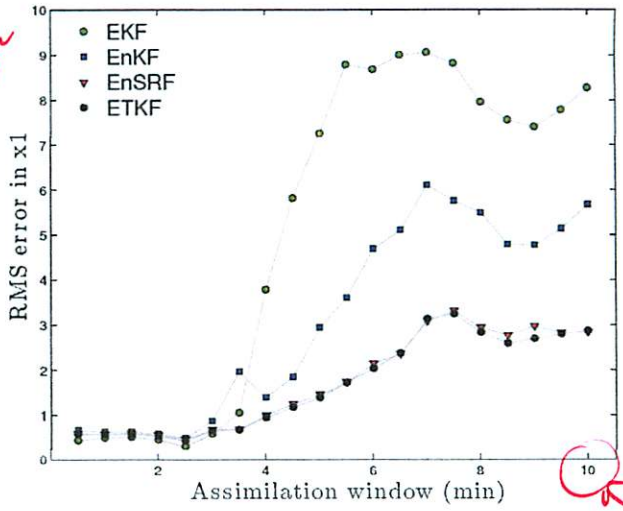

FIG. 5: The RMS error (not scaled by climatology) is reported for our EKF and EnKF filters, measured as the difference between forecast and truth at the end of an assimiliation window for the latter 2500 assimiliation windows in a 3000 assimilation window model run. Error is measured in the only observed variable, x_1. Increasing the assimilation window led to an decrease in predictive skill, as expected. Additive and multiplicative covariance inflation is the same as Harris et. al (2011).

$$\mathbf{P}_a = \mathbf{Z}_a\mathbf{Z}_a^T \quad , \quad \mathbf{P}_f = \mathbf{Z}_f\mathbf{Z}_f^T \quad (8)$$

for $\mathbf{Z}_a, \mathbf{Z}_f$ the matrix square roots of $\mathbf{P}_a, \mathbf{P}_f$, respectively. We are not concerned that this decomposition is not unique, and note that $\mathbf{Z}$ must have the same rank as $\mathbf{P}$ which will prove computationally advantageous. The power of the SRF is now seen as we represent the columns of the matrix $\mathbf{Z}_f$ as the difference from the ensemble members from the ensemble mean, to avoid forming the full forecast covariance matrix $\mathbf{P}_f$. The ensemble members are updated by applying the model M to the states $\mathbf{Z}_f$ such that an update is performed by

$$\mathbf{Z}_f = M\mathbf{Z}_a. \quad (9)$$

To summarize, the steps for the ETKF are to (1) form $\mathbf{Z}_f^T\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{Z}_f$, assuming $\mathbf{R}^{-1}$ is easy, and (2) compute its eigenvalue decomposition, and apply it to $\mathbf{Z}_f$.

The LEKF implements a strategy that becomes important for large simulations: localization. Namely, the analysis is computed for each grid-point using only local observations, without the need to build matrices that represent the entire analysis space. This localization removes long-distance correlations from $\mathbf{B}$ and allows greater flexibility in the global analysis by allowing different linear combinations of ensemble members at different spatial locations [24]. The general formulation of

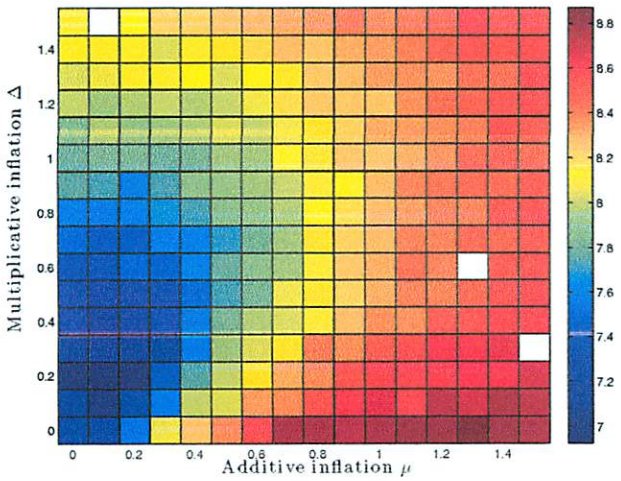

FIG. 6: The RMS error averaged over 100 model runs of length 1000 windows is reported for the ETKF for varying additive and multiplicative inflation factors Δ and μ. Each of the 100 model runs starts with a random IC, and the analysis forecast starts randomly. The window length here is 390 seconds. The filter performance RMS is computed as the RMS value of the difference between forecast and truth at the assimilation window for the latter 500 windows, allowing a spin-up of 500 windows.

the LEKF by Ott [29] goes as follows, quoting directly:

1. Globally advance each ensemble member to the next analysis timestep. Steps 2-5 are performed for each grid point.
2. Create local vectors from each ensemble member.
3. Project that point's local vectors from each ensemble member into a low dimensional subspace as represented by perturbations from the mean.
4. Perform the data assimilation step to obtain a local analysis mean and covariance.
5. Generate local analysis ensemble of states.
6. Form a new global analysis ensemble from all of the local analyses.
7. Wash, rinse, and repeat.

Proposed by Hunt in 2007 with the stated objective of computational efficiency, the LETKF is named from its most similar algorithms from which it draws [19]. With the formulation of the LEKF and the ETKF given, the LETKF can be described as a synthesis of the advantages of both of these approaches. This is the method that was sufficiently efficient for implementation on the full OpenFOAM CFD model of 240,000 model variables, and so we present it in more detail and follow the notation of Hunt et al (2007). As in the LEKF, we explicitly perform the analysis for each grid point of the model. The choice of observations to use for each grid point can be selected a priori, and tuned adaptively. Starting with a collection of background forecast vectors $\{\mathbf{x}_{b(i)} : i = 1..k\}$, we perform steps 1 and 2 in a global variable space, then steps 3-8 for each grid point:

1. Apply H to $\mathbf{x}_{b(i)}$ to form $\mathbf{y}_{b(i)}$, average the $\mathbf{y}_b$ for $\overline{\mathbf{y}_b}$, and form $\mathbf{Y}b$.
2. Similarly form $\mathbf{X}_b$ (now for each grid point).
3. Form the local vectors.
4. Compute $\mathbf{C} = (\mathbf{Y}_b)^T\mathbf{R}^{-1}$ (perhaps by solving $\mathbf{R}\mathbf{C}^T = \mathbf{Y}_b$.
5. Compute $\tilde{\mathbf{P}}_a = ((k-1)\mathbf{I}/\rho + \mathbf{C}\mathbf{Y}_b)^{-1}$ where $\rho > 1$ is a tun-able covariance inflation factor.
6. Compute $\mathbf{W}_a = \left((k-1)\tilde{\mathbf{P}}_a\right)^{1/2}$.
7. Compute $\overline{\mathbf{w}}_a = \tilde{\mathbf{P}}_a\mathbf{C}\,(\mathbf{y}_o - \overline{\mathbf{y}}_b)$ and add it to the column of $\mathbf{W}_a$.
8. Multiply $\mathbf{X}_b$ by each $\mathbf{w}_{a(i)}$ and add $\bar{\mathbf{x}}_b$ to get $\{\mathbf{x}_{a(i)} : i = 1..k\}$ to complete each grid point.
9. combine all of the local analysis into the global analysis.

The LETKF is implemented on our mesh using the full 40 cells across with zone sizes of center 10, and sides 15, resulting in 1600 local variables for 100 zones. In parallel, these 100 local computations can all be carried out simultaneously over an arbitrary number of processors.

B. Limited observation with adaptive local covariance

An initial test of prediction skill with limited observations in a twin model experiment showed that we needed 1000 spatial measurements of the temperature to predict flow reversals within 1 assimilation window. In an attempt to decrease the required observations to a experimentally realizable number, we implement a simple, adaptively localized covariance for data assimilation. Using the "square" sections of the loop to localize, we shift the zone to the left or right to follow the dominate flow direction at the center of that local window. Since we first saw a modest improvement in the prediction skill with full temperature observations, we hope that this improvement increases and is sufficient to get down to needing as few as 32 observations to predict reversals 1 assimilation window (of length 10 seconds) into the future.

In Figure 7 we see that the over an assimilation of 200 seconds, the ensemble converges on the hidden, true state. To test the performace of flow reversal prediction, we take the average of the ensemble flow direction (the average of each value of ϕ) as the predicted flow direction, and count how often we predict reversals both when

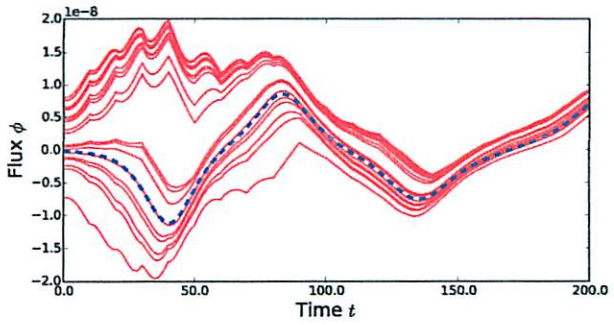

FIG. 7: Convergence of 20 ensembles using sliding windows, starting from initially random states. Here, as in most of the experiments, only temperature is observered and assimilated. Flux is computed as in Equation 4, on the left hand side of the thermosyphon. Assimilation takes place every 10 model seconds.

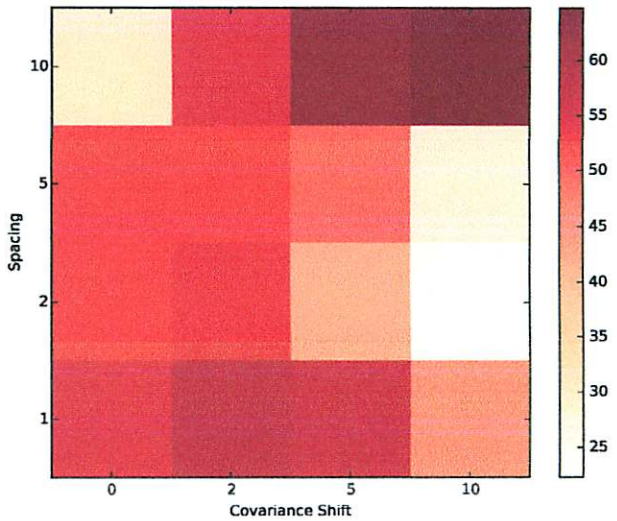

FIG. 8: Prediction skill as fraction of reversals that we correctly predicted across different numbers of observations and sliding windows of localized covariance.

they do and do not occur. Varying both the number of model variables and the strength of convariance shifting in Figure 8, we find that covariance shifting improves flow reversal prediction skill even when overservations are decreased. With full observations (spacing of 1), we have the best predictions with a covariance shift of 2. For 1/2 and 1/5 observations (a spacing of 2 (5) to obvserve every other (fifth) variable), we again have the best predictions with a shift of 2. And for a spacing of 10, observing every 10th variable, we achieve greater prediction skill with a covariance shift of 10.

Computing the average flow direction inside a localized covariance zone is straighforward, and computationally easy since the velocity is immediately available, making incorporation of this scheme into any data assimilation method easy. Since observations are also sparse in large weather models, we expect that using an adaptive local covariance scheme could lead to improved prediction skill.

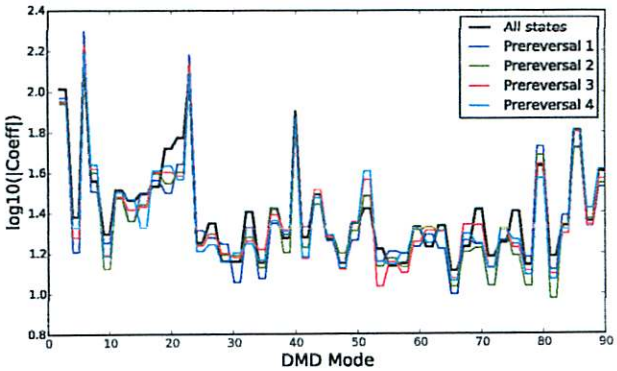

FIG. 9: The $\log_{10}$ average projection onto each DMD mode for different sets of model states. DMD constructed as snapshots every 10 seconds for the first 900 seconds of model time, and model states from the first 2000 seconds are all projected onto the DMD modes. All states average shown in black, and the average of the subset of states that occur 1 second, 3 seconds, 5 seconds, and 7 seconds before a reversal are shown in other colors. The symmetry of the loop generates modes that often come in pairs.

C. DMD experiment

To incorporate limited observations into a high-dimensional CFD simulation, we combine ideas from both CFD literature and data assimilation to make predictions. A Dynamic Mode Decomposition is performed using the "standard" algorithm of Tu [37], using snapshots every 10 seconds for the first 900 seconds of model time. A full picture of this timeseries can be found Appendix C, Figure S5. The "standard" algorithm is as follows with X and Y taken as the first and last $N-1$ columns of the snapshot matrix D:

$X = U\Sigma V$, take SVD of X:

$\tilde{A} = U^T Y V \Sigma^{-1}$, build the A matrix:

$\tilde{A}w = \lambda w$, compute eigenvectors and values:

$\hat{\theta}w = Uw$, compute corresponding modes:

In this reduced space, we extract the modes that correspond to the instability leading to flow reversals. For modes 21 and 79, we directly observe in Figure 9 that the average projection from states just 1 second before rever-

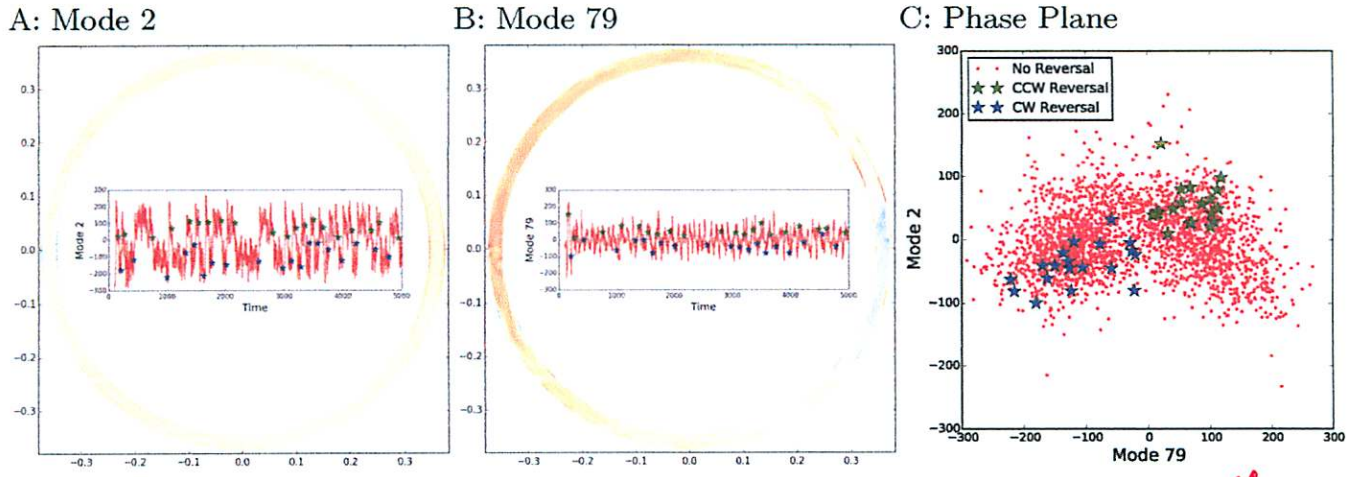

FIG. 10: Panel A: The temperature profile of the thermosyphon of Mode 2, with inset of the projection of timeseries states onto Mode 2 (the projection coefficient). Panel B: Likewise, the temperature profile of the thermosyphon of Mode 79, with inset of the projection of timeseries states onto Mode 79 (the projection coefficient). Panel C: This butterfly-shaped phase plane shows the value of the projection onto modes 2 and 79 for each time in the first 2000 timesteps of our ground truth model run. In blue and green stars the states that occur directly before a flow are highlighted, and are isolated into separate quadrants of phase space.

sal is the most different from the average stat projection, and the further away from the reversal, the more similiar the states become to the average. In Figure 10 we see that the dominant dynamics from mode 2 plotted with those of mode 79 are able to strongly separate reversal.

IV. RESULTS

The first output of this work is a general data assimilation framework for MATLAB and Julia. By utilizing an object-oriented (OO) design, the model and data assimilation algorithm code are separate and can be changed independently. The principal advantage of this approach is the ease of incorporation of new models and DA techniques (code available at `https://github.com/andyreagan/julia-openfoam`).

We first present the results pertaining to the accuracy of forecasts for synthetic data (twin model experiments). There are many possible experiments given the choice of assimilation window, data assimilation algorithm, localization scheme, model resolution, observational density, observed variables, and observation quality. We focused on considering the effect of observations and observational locations on the resulting forecast skill, and found that there was a threshold for the required number of observations to make useful predictions. In general, we see that increasing observational density leads to improved forecast accuracy. With too few observations, the data assimilation is unable to recover the underlying dynamics. Using adaptively localized covariance holds promise for data assimilation with data-scarce models, to overcome the lack of data.

The ability of DMD to recover the lower dimensional dynamics was expected but with 120000 variables is nonetheless an accomplishment. When modeling systems for which there are unknown but useful dimension reductions, as demonstrated here, DMD can be a useful tool.

The numerical coupling of CFD to experiment by DA should be generally useful to improve the skill of CFD predictions in even data-poor experiments, which can provide better knowledge of unobservable quantities of interest in fluid flow.

Acknowledgments

This work was made possible by funding from the Vermont Space Grant Consortium, NASA EPSCoR, NSF-DMS Grant No. 0940271, the Mathematics & Climate Research Network and the Vermont Advanced Computing Center.

[1] Anderson, M. J. (2001). A new method for non-parametric multivariate analysis of variance. *Austral Ecology 26*(1), 32–46.

[2] Asur, S. and B. A. Huberman (2010). Predicting the future with social media. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, Volume 1, pp. 492–499. IEEE.

[3] Bauer, P., A. Thorpe, and G. Brunet (2015). The quiet revolution of numerical weather prediction.