Proceedings of the
46th IEEE Conference on Decision and Control
New Orleans, LA, USA, Dec. 12-14, 2007

Multiscale Retrograde Estimation and Forecasting of Chaotic Nonlinear Systems

J. Cessna, C. Colburn, and T.R. Bewley

Abstract— **Chaotic systems are characterized by long-term unpredictability. Previous methods designed to estimate and forecast such systems, such as extended Kalman filtering [a matrix-based approach] and 4Dvar [aka Moving-Horizon Estimation (MHE), a vector-based approach], are essentially based on the assumption that Gaussian uncertainties in the initial state estimate and Gaussian disturbances to the state and measurements lead to uncertainty on the state estimate at later times that is well described by a Gaussian model. This assumption is not valid in chaotic nonlinear systems. A new method is thus proposed which revisits past measurements in order to reconcile them with more recent measurements of the system. This new approach, which we refer to as Model Predictive Estimation (MPE), is a straightforward extension of 4Dvar/MHE, an operational algorithm recently adopted by the weather forecasting community. Our new method leverages backwards-in-time (aka,"retrograde") time marches of the system, a receding-horizon optimization framework, and adaptive adjustment of the optimization horizon based on the quality of the estimate at each iteration.**

I. INTRODUCTION

The key assumption upon which the Kalman filter is based is that Gaussian initial uncertainty of the state estimate and Gaussian disturbances and measurement noise result in Gaussian uncertainty of later state estimates. (This allows the Kalman filter to summarize all past measurements with a single state estimate of dimension N, and covariance estimate of dimension N^2.) Though true for infinitesimal uncertainties in smooth nonlinear systems, this assumption fails dramatically for the estimation errors which are typical in such systems. Thus, a new approach is warranted to handle this class of problems. We have developed such a new approach and conclusively demonstrated its effectiveness on the simplest of all chaotic systems (the Lorenz equation) and a slightly more complicated 1D chaotic PDE (the Kuramoto-Sivashinsky equation). We present this method as a sequence of straightforward modifications to the existing 4Dvar/MHE algorithm (recently adopted by the weather forecasting community). Essentially, it is a rearrangement of the order in which calculations are performed that leads to no more algorithm complexity. Thus, we begin by highlighting the key features of this existing algorithm.

A. Background: The 4D-Variational Assimilation Method

The 4D-Variational Assimilation Method (4Dvar) is a gradient-based approach. It is designed to harness observations distributed in both time and space to minimize the estimate of the state constrained to be on a trajectory of the system. For sufficiently large systems, it is preferable to traditional matrix-based methods (such as Extended Kalman Filtering) because it avoids expensive costs of propagating covariance (and resulting gain) matrices. Due to its structure, these calculations are implicitly and iteratively computed using only the integration of vectors in time. Unlike the Extended Kalman Filter, however, the 4Dvar algorithm is not an infinite-time-horizon algorithm. Consequently, estimation calculations cannot be done online.

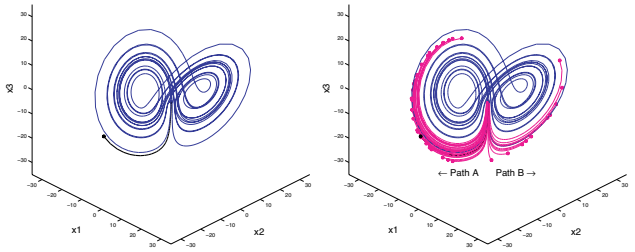

Fig. 1. Demonstration of the nominal trajectory of the state estimate (left, dark), and several perturbed trajectories (right) initiated at a point in the Lorenz system characterized by a large local Lyapunov exponent. In this test, the initial perturbations of the perturbed trajectories are very small and distributed in a Gaussian fashion. The final distribution of the perturbed trajectories, however, is highly non-Gaussian.

The 4Dvar algorithm is initiated with an estimate of the state, $\mathbf{u}$, at a past time, $t=-T$ (where $t=0$ is the current time). The best unbiased estimator minimizes the cost (a function of the estimate, $\mathbf{u}$, at $t=-T$),

$$J(\mathbf{u}) = (\mathbf{u}-\mathbf{u}_B)^* B^{-1}(\mathbf{u}-\mathbf{u}_B) + \int_{-T}^{0} [\mathbf{y}(t)-H\hat{\mathbf{x}}(t)]^* R^{-1}[\mathbf{y}(t)-H\hat{\mathbf{x}}(t)]dt, \quad (1)$$

where $\mathbf{u}_B$ is the background (or initial) estimate of the state at $t=-T$, B is its associated covariance, and $\hat{\mathbf{x}}(t)$ is the trajectory of system with initial conditions $\hat{\mathbf{x}}(-T)=\mathbf{u}$.

This minimizer is iteratively found using the gradient of the cost function and a steepest descent method. Obtaining the gradient of the cost function requires the introduction of the adjoint operator.

For a general nonlinear system, the estimator equation is given by:

$$\dot{\hat{\mathbf{x}}} = n(\hat{\mathbf{x}}) \quad \text{on} \quad t \in [-T, 0] \quad (2)$$

Taking perturbations ($\mathbf{u} \leftarrow \mathbf{u}+\mathbf{u}'$, $\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}}+\hat{\mathbf{x}}'$, $J \leftarrow J+J'$) and linearizing about the trajectory $\hat{\mathbf{x}}(t)$ gives:

$$\dot{\hat{\mathbf{x}}}' = A\hat{\mathbf{x}}' \quad \Leftrightarrow \quad L\hat{\mathbf{x}}' = 0 \quad (3)$$

$$\hat{\mathbf{x}}'(-T) = \mathbf{x}' \quad (4)$$

$$J'(\mathbf{u}') = (\mathbf{u}-\mathbf{u}_B)^* B^{-1}\mathbf{u}' - \int_{-T}^{0} [\mathbf{y}(t)-H\hat{\mathbf{x}}(t)]^* R^{-1}\hat{\mathbf{x}}'(t)\, dt. \quad (5)$$

J. Cessna is with the Department of of Mechanical and Aerospace Engineering, University of California - San Diego jcessna@ucsd.edu
C. Colburn is with the Department of of Mechanical and Aerospace Engineering, University of California - San Diego ccolburn@ucsd.edu
T. Bewley is with the Faculty of Mechanical and Aerospace Engineering, University of California - San Diego bewley@ucsd.edu

1-4244-1498-9/07/$25.00 ©2007 IEEE.

Defining the adjoint identity (6), allows us to derive the adjoint operator as follows:

$$\langle\ \mathbf{r}, \hat{\mathbf{x}}'\ \rangle = \int_{-T}^{0} \mathbf{r}(t)^*\ \hat{\mathbf{x}}'(t)\,dt, \quad (6)$$

$$\langle\ \mathbf{r}, L\hat{\mathbf{x}}'\ \rangle = \langle\ L^*\mathbf{r}, \hat{\mathbf{x}}'\rangle + b \quad (7)$$

$$\Longrightarrow \quad L^* = -(d/dt + A^*), \quad b = [\ \mathbf{r}^*\hat{\mathbf{x}}'\ ]_{t=-T}^{t=0} \quad (8)$$

which gives the final adjoint equation as:

$$L^*\mathbf{r} = -H^*(\mathbf{y} - H\hat{\mathbf{x}}) \quad \Longleftrightarrow$$
$$-\dot{\mathbf{r}} = A^*\mathbf{r} - H^*(\mathbf{y} - H\hat{\mathbf{x}}) \quad \text{where} \quad \mathbf{r}(0) = 0 \quad (9)$$

Hence, a single forward integration of the estimate and a single backward integration of the adjoint, over the estimation window, provide the gradient of (1) as follows:

$$J'(\mathbf{u}') = \left[(\mathbf{u} - \mathbf{u}_B)^* B^{-1} - \mathbf{r}(-T)\right]\ \mathbf{u}' \quad (10)$$

$$J'(\mathbf{u}') = \left[\bigtriangledown J(\mathbf{u})\right]\ \mathbf{u}' \quad (11)$$

$$\Longrightarrow \bigtriangledown J(\mathbf{u}) = (\mathbf{u} - \mathbf{u}_B)^* B^{-1} - \mathbf{r}(-T) \quad (12)$$

where r is the adjoint operator. At each iteration, the suitable forward and backward integrations are performed to find the gradient, and then a line search,

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \alpha\mathbf{p}_k \qquad \text{where} \qquad \mathbf{p}_k = -\bigtriangledown J(\mathbf{u}_k), \quad (13)$$

is performed to best update the estimate for the next iteration. This process is then repeated until convergence.

For a linear system, it can be shown that 4Dvar converges to the optimal solution, in the sense that it best balances the observations taken (and their associated covariance) with the previous knowledge of the state. Therefore, it is equivalent to the Kalman Smoothing algorithm performed over the same interval with identical initial conditions.

II. Model Predictive Estimation Algorithm

While the advantages (over traditional matrix-based methods) of 4Dvar have been conclusively demonstrated for large order systems, it is evident that for highly chaotic systems (and for systems where the perturbations to the state estimate are likely large,) 4Dvar falls short of the performance required.

A. Receding Optimization Horizon

A property of chaotic systems is the exponential divergence of the trajectories of perturbed initial conditions. The first fundamental flaw of the standard 4Dvar approach is that, if the computer speed is finite (say, if the computer can simulate the evolution of the system model less than 1000 times faster than the system itself actually evolves), then by the time the involved iteration process converges, the optimization window has slid so far into the past that the problem solved is no longer a problem of interest. Even if this estimate is known very accurately, any forecast developed from this estimate will already have begun to diverge from the truth before being propagated out to the present time.

One modification to reduce this problem is to redefine the optimization window before each iteration begins in order to stay as close to the real time operation of the system as possible. That is, if an iteration of the algorithm is performed on the window $t = [-T\ ,\ 0\ ]$, this window will fall back in time by an amount Δ. This elapsed time is a function of the computing speed of the computer. Now before beginning the next iteration we will shift our optimization window forward to $t = [-T + \Delta\ ,\ \Delta\ ]$ (where the current time is now $t = \Delta$). Additionally, our estimate must be propagated forward (via the system equations) by an amount Δ.

B. Retrograde Time Marches

The receding optimization horizon previously described serves the purpose of keeping the optimization window valid. However, even with the addition of a receding optimization horizon, the 4Dvar algorithm is optimizing an estimate that is always a distance $t = -T$ into the past. A forecast from this estimate still experiences exponential divergence prior to becoming a 'true' forecast. That is, the forecast diverges from the known measurements in the past before propagating into the future to provide useful information. This presents a difficult problem when deciding upon the width (T) of the optimization window. It must be sufficiently small such that the exponential divergence over $t = [-T\ ,\ 0\ ]$ is not severe, but it must be sufficiently large such that the cost function computed over this interval is based on a sufficient number of (noisy) measurements to accurately reflect the problem we need to solve.

Hence, it is desirable to solve a new problem where the estimate being optimized is on the most recent end of the optimization window. To do this, we define a modified cost function,

$$J_R(\mathbf{u}) = \int_0^{-T} [\ \mathbf{y}(t) - H\,\hat{\mathbf{x}}(t)\ ]^*\ R^{-1}\ [\ \mathbf{y}(t) - H\,\hat{\mathbf{x}}(t)\ ]\ dt, \quad (14)$$

where the optimization window remains $t = [-T\ ,\ 0\ ]$, but now the estimate, $\mathbf{u}$, is taken at $t = 0$ (the current time) as opposed to $t = -T$. The retrograde cost function is then defined by propagating the estimate at the front of the interval *backwards* in time over the observations.

Additionally, one should note the disappearance of the background term from the original cost function, (1). Due to the nature of chaotic systems, some spurious behavior can be eliminated by removing the 'inertia' from the previous estimate. When it becomes apparent that our estimate is bifurcating from the observations, we don't want to place a penalty on a large update to the estimate to correct for this. While it is noted that the removal of the background term can introduce other instabilities into the algorithm, we seek to alleviate any adverse behavior by including sufficient observations in our optimization window. On a side note, the propagation of the estimate covariance is, at this time, ill-defined for the receding horizon problem being framed.

Derivation of the gradient of the retrograde cost function requires a similar introduction of the adjoint as previously described, however, the retrograde adjoint is initialized to $r = 0$ at $t = -T$ and marched *forward* through the optimization window. The resulting gradient is similar to the standard

4Dvar case (minus the contributions from the background term):

$$\bigtriangledown J_R(\mathbf{u}) \;=\; -\mathbf{r}(0). \tag{15}$$

The update to the estimate is still of the form:

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \alpha \mathbf{p}_k \qquad \text{where} \qquad \mathbf{p}_k = -\bigtriangledown J_R(\mathbf{u}_k), \tag{16}$$

and requires a suitable line search minimization.

Note that, in a system dominated by convection-like behavior, such as the Lorenz system, the exponential rate of divergence of perturbed trajectories (along the attractor) when we march the system forward in time is commensurate with the exponential rate of divergence of perturbed trajectories (perpendicular to the attractor) when we march the system backward in time. That is, for short time marches, it is essentially no more difficult to march the system backward in time than it is to march the system forward in time.

Computationally, these retrograde time marches require the same amount of effort as the traditional time marches in the 4Dvar algorithm. The net effect, though, is that the algorithm is working to minimize an estimate at the current time, so that any forecasts from the estimate are, in fact, forecasts of the system beyond the current time. It is worth noting, however, that for a linear system, we know that 4Dvar (and accordingly our new algorithm) converge to a global minimizer of the cost function. Therefore, the optimum estimate of the state is found along all values in the optimization window, and all points in the window are equivalent in time (i.e. they all lie along the same trajectory of the system). As a result, a forecast from any one of these points will result in the identical future prediction. For the chaotic systems of interest, though, it is unlikely that either algorithm will converge to anything other than a local minimum, thus the resulting forecasts from these estimates will begin to diverge immediately–which gives prudence to the objective of optimizing the estimate of the current time.

C. Multiscale Analysis in Time

With the traditional 4Dvar algorithm, the width of the optimization window must be determined offline. This is due to the fact that the estimate is taken at $t = -T$ and the cost function is computed by integrating forward in time. As discussed previously, the selection of this interval width is crucial to the performance of the algorithm: too small and not enough information is gained to make an accurate update, too large and the computations take too long to be of any significant value.

An added advantage of the retrograde time marches is that the retrograde cost function, (14), is evaluated by integrating the estimate backward in time from $t = 0$ to $t = -T$. Hence, the width of the optimization window does not need to be determined prior to the start of the backwards propagation of the estimate. In fact, the width can be continually determined on the fly. This allows us to march the estimate backward until enough information is obtained about its quality. At that point we can stop the integration, initialize the adjoint, and march the adjoint forward to obtain the gradient. Conceptually, this is done by looking at the deviation between our estimated trajectory and the observations. If the estimate seems to be nearly unbiased as it passes backwards through the observation sequence, then we can extend the width of our optimization window to capture more data points. However, if our estimation begins to diverge from the observations, then the two will become quickly uncorrelated, and we can no longer gain valuable information by extending the width of the window. At this point we stop the backwards march.

Mathematically, we define a bias measure, $B(t)$, as

$$B(t) = \left| \int_0^{-t} [\; \mathbf{y} - H\hat{\mathbf{x}}(\tau) \;]\; d\tau \right|_{L^1} \quad \text{where} \quad |\; \mathbf{y} \;|_{L^1} = |y_1| + \cdots + |y_n|. \tag{17}$$

We define (through experimentation) a critical bias value ($\bar{B}$), such that we deem the estimate 'sufficiently' uncorrelated from the observations, and at that point the estimation is stopped. This defines the width (T) of the estimation window as

$$-T = \max\{\; t \mid B(t) > \bar{B} \;\}. \tag{18}$$

Fig. 2 shows a graphical representation of the accumulated bias. The trajectories begin to significantly depart at about $t = -0.5$. After about $t = -0.8$ the estimation is completely diverged from the observations; this is the point at which the backwards march would be halted. For this example, the critical bias is $\bar{B} = 1.25$ and the width of the optimization window is $T = 0.8$.

A discussion of how this online multiscale analysis window affects the performance of the new algorithm (specifically the optimization surface) is left for a latter section.

D. Summary of the Proposed Algorithm

When all the modifications are included together, the resulting algorithm can be described as follows. Begin by initializing the iteration count, $k = 0$, and a preliminary estimate of the state at the current time, $\mathbf{u}_0$.

1) March the estimate, $\mathbf{u}_k$, backward until sufficient divergence (i.e. $B(t) > \bar{B}$). Save the estimated trajectory.
2) Use the saved trajectory to march the adjoint, $\mathbf{r}_k$, forward from $t = -T$ to $t = 0$.
3) Compute the gradient, $\bigtriangledown J_R(\mathbf{u}_k) = -\mathbf{r}_k(0)$, and the direction of descent, $\mathbf{p}_k = -\bigtriangledown J_R(\mathbf{u}_k)$.
4) Perform a suitable line search on α^k so as to minimize the cost function, $J_R(\mathbf{u}_k + \alpha \mathbf{p}_k)$. Update the estimate, $\mathbf{u}_{k+1} = \mathbf{u}_k + \alpha_k \mathbf{p}_k$.
5) Determine the elapsed time since the start of the iteration, Δ_k, and march the current estimate, $\mathbf{u}_{k+1}$ up to the current time. Increment $k \leftarrow k+1$.
6) Repeat from step 1 (algorithm will not converge).

On a semantic note, one might call the resulting algorithm *multiscale retrograde receding-horizon model predictive estimation* (MPE) in order to draw attention to its close

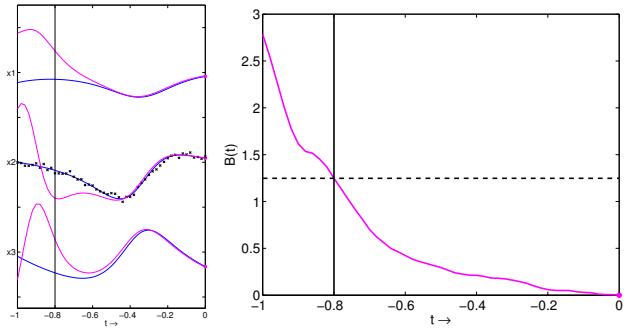

Fig. 2. A graphical representation of the diverging estimated trajectory from the observations. The plot on the left shows the estimate (light line) diverging from the truth model and observations. The plot on the right shows the value of $B(t)$ as the divergence between the estimate and observations.

analogy with the well-known strategy of model predictive control (MPC). The former approach consistently starts from the present time and scans backward in time to determine the best state estimate, whereas the latter approach consistently starts from the present time and scans forward in time to determine the best control distribution.

III. Model Predictive Estimation Advantages

As presented, Model Predictive Estimation (MPE) is defined as a natural sequence of modifications to the existing data assimilation method known as 4Dvar. Thus, implicitly, MPE retains the basic advantage of 4Dvar over Kalman filtering and its tangents: MPE is a vector based method. Computationally, this is a huge advantage over any matrix based method for sufficiently large systems. In practice, all systems of interest are large enough to make any matrix-based method infeasible. Unlike 4Dvar, though, MPE has the added advantages of begin able to revisit past measurements in light of new data, update an estimate of the current time, and dynamically change the optimization surface.

A. Revisit Measurements in Light of New Data

At the sacrifice of the background term, MPE is able to repeatedly revisit know observations to help shape the optimization problem. Measurements that at one time seemed noisy and spurious, could at a later time help define the bifurcation of the estimate from the truth model. The retrograde time marches along with the multiscale optimization window allow the algorithm to go back and utilize these observations as deemed necessary. In contrast, 4Dvar remains iterating on the same fixed window until convergence, and then it moves on–never to revisit the observations in that time window again.

B. Update Estimate of Current Time

By definition, chaotic systems will always be limited in their ability to be predicted. Due to observation noise, model noise, and many other complex factors, an exact estimate of the state is impossible to obtain. Therefore any forecast of the estimate will immediately begin to diverge from the truth and will always be limited in length. Hence, it follows naturally that the forecast should start from the present time; doing anything else places some of the useful width of the forecast in the past. The MPE algorithm, along with sufficient computational power, achieves this real-time goal.

C. Dynamic Optimization Surface

The traditional 4Dvar method makes no guarantee of global convergence, and, in fact, for highly nonlinear system (such as chaotic systems) it is extremely likely that 4Dvar will get hung up in local minima. The optimization surface defined by the cost function (1) is static throughout the iteration process.

Perhaps one of the more subtle, yet significant consequences of the MPE algorithm, is its effect on the convergence of the estimate. In fact, the MPE algorithm is guaranteed to *never* converge. At each iteration, because of the receding horizon optimization framework and the multiscale analysis, the optimization surface is modified. In a sense, the MPE algorithm does not iterate each surface until convergence, but rather takes one steepest descent step, and then redefines the problem. One can think of the MPE algorithm as a continuous stepping towards the bottom of a dynamically changing surface. Consequently, the estimate never converges to the minimum because the minimum is constantly changing.

This dynamic optimization is desired because it keeps the problem being solved valid, but it also keeps the estimate out of local minima. When 4Dvar converges to a local minima, the algorithm is complete, and–good or bad–no more information can be gained from repeated iterations. With MPE, the estimate never converges to any minima (local or global), so when it approaches a local minima, the optimization surface eventually changes enough to force the estimate out of the local area.

In addition to the optimization surface changing with time, it is also highly dependent on the width of the optimization window. Small windows tend to yield much smoother optimization surfaces with fewer local minima. But, the global minimum of these types of surfaces also tends to be inaccurate. Larger windows produce a more accurate global minimum, but they produce highly irregular optimization surfaces, fullof local minima. Thus, shorter windows are preferred to more easily pull the estimate close to the solution, and longer windows are used to improve the accuracy of an already sufficiently close estimate. The difference in these surfaces is depicted in Fig. 3.

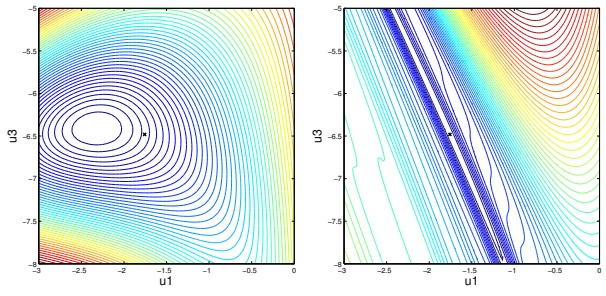

Fig. 3. Demonstration of how the multiscale approach preconditions the optimization problem. The figure at left corresponds to the short optimization interval $T = 0.2$, whereas the figure at the right corresponds to the long optimization interval $T = 0.8$. In both figures, the curves denote isosurfaces of the optimization surface, assuming $u_2 = x_2(0)$ is known precisely for the purpose of making an intelligible plot, and the $\times$ denotes the actual value of the underlying (but unobserved) "truth model" of the system, $x_1(0)$ and $x_3(0)$.

IV. The Lorenz Equation

As a logical first test case, we examine the simplest of all chaotic systems: the Lorenz equation. This ODE models the dynamic convection of a fluid cell being warmed from below and cooled from above. A deep understanding of our newly-proposed forecasting strategy can be obtained in this low dimensional setting, where the chaotic attractor is easily visualized. For purposes of reference, the system dynamics are given by:

$$\dot{\mathbf{x}} = n(\mathbf{x}) \quad \text{where} \quad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}, \quad n(\mathbf{x}) = \begin{pmatrix} \sigma(x_2 - x_1) \\ -x_2 - x_1 x_3 \\ -b x_3 + x_1 x_2 - br \end{pmatrix} \tag{19}$$

where σ, b, and r are constant parameters of the system.

An experiment was constructed to compare the new MPE method against the traditional 4Dvar assimilation. Both methods were initialized with noisy measurements of only the second state. The 4Dvar window was selected to be one time unit in length, and at the onset of the experiment, this window was placed as near as possible to the current time. As the experiment ran, both MPE and 4Dvar iterated at the same rate, with the 4Dvar window sliding back into the past as governed by the computational speed of the computer. MPE was performed as highlighted in Section II-D. The iterations of both algorithms were stopped when 4Dvar converged to a solution (remember that MPE will never converge) and then forecasts were computed out from each estimate. For comparison of these forecasts, we define an accumulated error in (20).

$$E(t) = \frac{1}{\gamma} \int_{t_0}^{t} \| \mathbf{x}_f(\tau) - \mathbf{x}_t(\tau) \|_2^2 \, d\tau \qquad \text{for} \quad t \geq t_0, \tag{20}$$

where t_0 is the time the forecast began, $\mathbf{x}_f(t)$ is the time-history of the forecast, $\mathbf{x}_t(t)$ is the time-history of the truth-model, and γ is a constant normalizing parameter.

Fig. 4 shows a typical result from this experiment. As expected, both forecasts begin to exponentially diverge. However, The forecast from the MPE algorithm remains

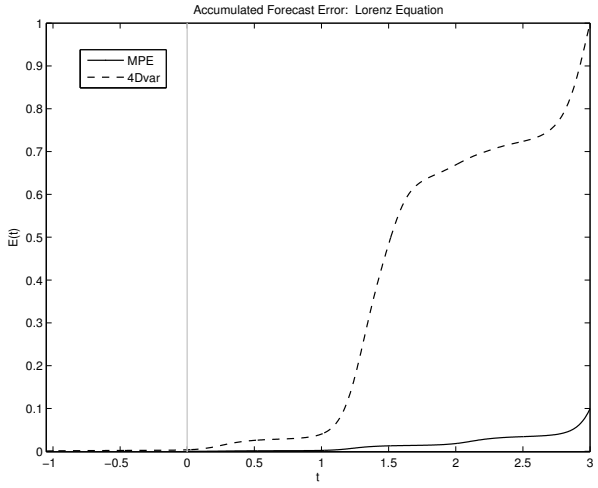

Fig. 4. Accumulated forecast error for parallel 4Dvar and MPE assimilation example. Both algorithms were run for the same system and terminated upon 4Dvar convergence. Forecasts were done from the computed estimate and compared against the known truth model. MPE, by construction, forecasts out from the current time ($t = 0$, light vertical line), while 4Dvar forecasts out from the left edge of the optimization window ($t = -1.05$).

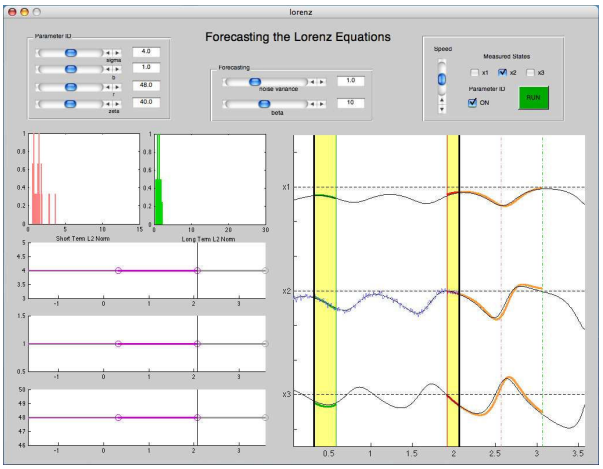

Fig. 5. Cartoon depicting the MPE forecasting algorithm synthesized with a modified parameter ID algorithm, indicating the relation of the time intervals used for identifying the parameters and obtaining the estimate of the current time.

valid until 2.5-3 time units into the future. In contrast, the 4Dvar forecast is useless after 1 time unit into the future. Remember that the 4Dvar forecast starts from the estimate in the past (which for this example is at $t = -1.05$). The 4Dvar algorithm provides an excellent forecast over the window $t = [-1.05,\ 0]$, but this forecast of the past is essentially wasted information. Note that at the onset of the algorithm, the 4Dvar estimate was at the back edge of the window ($t = -1$), so with this simple system, convergence happened relatively quick ($\Delta t = 0.05$). Even so, the width of the 4Dvar window places the estimate far enough back in time to significantly impact the validity of its forecast. In addition to the shift in estimate times, one can also note that the length of valid forecast time for MPE (2.5-3 time units) is longer than that of 4Dvar (2 time units). This can be attributed to the other algorithm modifications (receding horizon, multiscale analysis, etc.).

A GUI has been developed to illustrate the MPE algorithm working on the Lorenz system. Additionally, it incorporates a receding horizon, multiscale parameter ID analysis to estimate the quasi-constants (σ, b, and r) of the system.

V. The 1D Kuramoto-Sivashinksky Equation

The 1D Kuramoto-Sivashinksky (KS) equation is an ideal case study because it is both well known by researchers and non-trivial in most applications. The KS equation describes the following chaotic systems: hydrodynamic instability in a laminar flame [6], diffusion induced chaos in reaction systems [3], and the growth of thermodynamically unstable crystal surfaces [1]. For the scope of this paper the 1D KS system with periodic boundary conditions is considered:

$$u_t + u u_x + (u + u_{xx})_{xx} = 0 \quad \text{where} \quad u_x = \frac{\partial u}{\partial x}, \tag{21}$$

$$u(t,0) = u(t,L), \quad \text{and} \quad u(0,x) = f(x). \tag{22}$$

The MPE algorithm is performed on this system as has been previously described. Thus, extracting the gradient for the steepest descent method becomes a march of two partial differential equations. However, some care must be taken when doing retrograde marches of a PDE system.

A. Regularization of PDE Analysis

Regularization, or "smoothing," in the adjoint analysis of PDE systems is a subtle issue. The advantages gained in performing the PDE adjoint analysis forward in time requires that the PDE state equations be marched backward in time. This algorithm construction is ill-posed. To stabilize this system, a mixed time-space derivative must be introduced to dampen any dominant unstable terms. This topic, called quasi-reversibility, is well known (and regularly applied) throughout the forecasting community. One might claim that because this regularization technique is required to ensure a well-posed problem, the accuracy of the forecast is reduced. In fact, even in well-posed forecasts of multiscale systems on coarse grids, a time-space derivative is added to artificially dampen length-scale fluctuations and stabilize the system. Additionally, the adverse effects of reversibility are further minimized by the multiscale horizon. So, where other assimilation techniques would blindly use inaccurate states, MPE has the ability to reduce the number of estimates to a smaller, more accurate set. The 1D Kuramoto-Sivashinksky equation is dominated by an unstable 4th order linear term. As a result of the system being ill-posed these regularization techniques are required.

B. Comparison to 4Dvar

An identical case study to the Lorenz system described in Section IV was done for the KS equation. Fig. 6 shows error comparisons of each algorithm after providing the same initial/boundary conditions to each system. A typical run shows that MPE provides forecasts which are significantly more accurate than 4DVar.

Note that, just as in the Lorenz case, both forecasts are exponentially diverging from the actual system states, but the MPE forecast remains valid for a much longer forecast length. The accumulated error of 4DVar becomes significant as the forecast reaches into the actual future ($t \geq 0$), whereas MPE can accurately forecast up to 25 time units ahead (this is approximately 80 times larger than the optimization window used by the algorithm.)

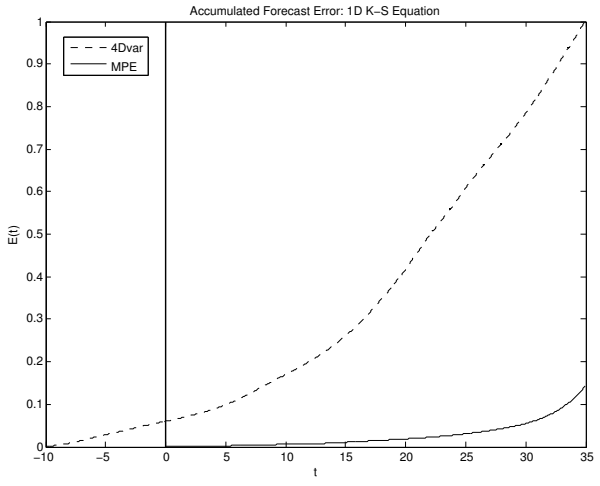

Fig. 6. Accumulated forecast error for parallel 4Dvar and MPE assimilation example. Both algorithms were run for the same system and terminated upon 4Dvar convergence. Forecasts were done from the computed estimate and compared against the known truth model. MPE, by construction, forecasts out from the current time ($t = 0$, light vertical line), while 4Dvar forecasts out from the left edge of the optimization window ($t = -10$).

VI. CONCLUSIONS AND FUTURE WORK

A. Conclusions

A new method for data assimilation has been presented, aptly named Model Predictive Estimation (MPE). MPE is developed as a sequence of straightforward modifications to an existing vector-based method known as 4Dvar. The new method utilizes a receding-horizon optimization along with retrograde time marches and a multiscale analysis to work towards an improved forecast of a nonlinear chaotic system. MPE has been developed to handle these types of systems where gaussian initial uncertainties of the state estimate and gaussian disturbances and measurement noise do not result in gaussian uncertainties of later state estimates. This is accomplished via revisiting past measurements in light of new data, optimizing an estimate of the most recent time, and creating a dynamic optimization surface. The benefits of such an algorithm have been conclusively demonstrated for two well known chaotic systems.

B. Future Work

From this point, this research will explore two important directions. First off, we intend to extend our MPE algorithm to more complex PDE systems. This is in an effort to work towards a real-life application, such as a forecast of the 2D shallow water equation and potentially the Navier-Stokes equation.

Another area that we will explore actively is the relation between the proposed retrograde approach (with backward-in-time state marches) and the Kalman smoothing approach (with backward-in-time sweeps of the relevant Ricatti equation linearized each time about updated state trajectories). This is the natural counterpart to the relation between the traditional forward in time (4Dvar) approach and Kalman filtering, and is a subject area that is quite intriguing and promising, at least for systems in which matrix-based analyses (or reduced rank approximations thereof) are numerically tractable.

REFERENCES

[1] A. A. Golovin, A. A. Nepomnyashchy, S. H. Davis, and M. A. Zaks, "Convective Cahn-Hilliard Models: From Coarsening to Roughening", *in Physical Review Letters*, vol. 86, no. 8, 2001.

[2] J.Kim and T.R.Bewley, "A Linear Systems Approach to Flow Control", *Annual Rev. Fluid Mech.*, 2007.

[3] Y. Kuramoto, Diffusion induced chaos in reaction systems, *in Progress of Theoretical Physics.*, 1978, Supplements 64, pp. 346-367.

[4] Z. Li and I. M. Navon, "Optimality of 4D-Var and its relationship with the Kalman filter and Kalman smoother", *Quarterly Journal of the Royal Meteorological Society*, Vol.127 No. 572 - January 2001 Part B, 2001, pp. 661-684.

[5] C.V. Rao, J.B. Rawlings, and D.Q. Mayne, "Constrained state estimation for nonlinear discrete-time systems: stability and moving horizon approximations", *IEEE Trans. on Auto. Control*, vol.48, no.2, 2003.

[6] G. I. Sivashinsky, "Nonlinear Analysis of Hydrodynamic Instability in Laminar Flames - I. Derivation of Baisc Equations", *Acta Astronautica*, vol. 4, 1977, pp 1177-1206.