Adaptive Observation Strategies for Forecast Error Minimization

Nicholas Roy[1], Han-Lim Choi[2], Daniel Gombos[3], James Hansen[4], Jonathan How[2],
and Sooho Park[1]

[1] Computer Science and Artificial Intelligence Lab
Massachusetts Institute of Technology
Cambridge, MA 02139
[2] Aerospace Controls Lab
Massachusetts Institute of Technology
Cambridge, MA 02139
[3] Department of Earth and Planetary Sciences
Massachusetts Institute of Technology
Cambridge, MA 02139
[4] Marine Meteorology Division
Naval Research Laboratory
Monterey, CA 93943

Abstract. Using a scenario of multiple mobile observing platforms (UAVs) measuring weather variables in distributed regions of the Pacific, we are developing algorithms that will lead to improved forecasting of high-impact weather events. We combine technologies from the nonlinear weather prediction and planning/control communities to create a close link between model predictions and observed measurements, choosing future measurements that minimize the expected forecast error under time-varying conditions.

We have approached the problem on three fronts. We have developed an information-theoretic algorithm for selecting environment measurements in a computationally effective way. This algorithm determines the best discrete locations and times to take additional measurement for reducing the forecast uncertainty in the region of interest while considering the mobility of the sensor platforms. Our second algorithm learns to use past experience in predicting good routes to travel between measurements. Experiments show that these approaches work well on idealized models of weather patterns.

1 Introduction

Recent advances in numerical weather prediction (NWP) models have greatly improved the computational tractability of long-range prediction accuracy. However, the inherent sensitivity of these models to their initial conditions has further increased the need for accurate and precise measurements of the environmental conditions. Deploying an extensive mobile observation network is likely to be costly, and measurements of the current conditions may produce different results in terms of improving forecast performance [1,2]. These facts have led to the development of observation strategies where additional sensors are deployed to achieve the best performance according to some

Y. Shi et al. (Eds.): ICCS 2007, Part I, LNCS 4487, pp. 1138–1146, 2007.

measures such as expected forecast error reduction and uncertainty reduction [3]. One method for augmenting a fixed sensor network is through the use of “adaptive” or “targeted” observations where mobile observing platforms are directed to areas where observations are expected to maximally reduce forecast error under some norm (see, for example, NOAA’s Winter Storm Reconnaissance Program [4]). The hypothesis is that these directed measurements provide better inputs to the weather forecasting system than random or gridded use of the observing assets.

This paper describes an adaptive observation strategy that integrates nonlinear weather prediction, planning and control to create a close link between model predictions and observed measurements, choosing future measurements that minimize the expected forecast error under time-varying conditions. The main result will be a new framework for coordinating a team of mobile observing assets that provides more efficient measurement strategies and a more accurate means of capturing spatial correlations in the system dynamics, which will have broad applicability to measurement and prediction in other domains. We first describe the specific non-linear weather prediction model used to develop our adaptive observation strategy, and then describe a global targeting algorithm and a local path planner that together choose measurements to minimize the expected forecast error.

2 Models of Non-linear Weather Prediction

While there exist large-scale realistic models of weather prediction such as the Navy’s Coupled Ocean Atmosphere Prediction System (COAMPS), our attention will be restricted to reduced models in order to allow computationally tractable experiments with different adaptive measurement strategies. The Lorenz-2003 model is an extended model of the Lorenz-95 model [1] to address multi-scale feature of the weather dynamics in addition to the basic aspects of the weather motion such as energy dissipation, advection, and external forcing. In this paper, the original one-dimensional model is extended to two-dimensions representing the mid-latitude region $(20 - 70\,\mathrm{deg})$ of the northern hemisphere. The system equations are

$$\begin{aligned}\dot{y}_{ij} =& - \xi_{i-2\alpha,j}\xi_{i-\alpha,j} + \frac{1}{2\lfloor\alpha/2\rfloor+1}\sum_{k=-\lfloor\alpha/2\rfloor}^{k=+\lfloor\alpha/2\rfloor}\xi_{i-\alpha+k,j}y_{i+k,j}\\ &- \mu\eta_{i,j-2\beta}\eta_{i,j-\beta} + \frac{\mu}{2\lfloor\beta/2\rfloor+1}\sum_{k=-\lfloor\beta/2\rfloor}^{k=+\lfloor\beta/2\rfloor}\eta_{i,j-\beta+k}y_{i,i+k}\\ &- y_{ij} + F\end{aligned} \tag{1}$$

where

$$\xi_{ij} = \frac{1}{2\lfloor\alpha/2\rfloor+1}\sum_{k=-\lfloor\alpha/2\rfloor}^{k=+\lfloor\alpha/2\rfloor} y_{i+k,j}, \quad \eta_{ij} = \frac{1}{2\lfloor\beta/2\rfloor+1}\sum_{k=-\lfloor\beta/2\rfloor}^{k=+\lfloor\alpha/2\rfloor} y_{i,j+k}, \tag{2}$$

where $i = 1, \ldots, L_{on}, j = 1, \ldots, L_{at}$. The subscript i denotes the west-to-eastern grid index, while j denotes the south-to-north grid index. The dynamics of the (i, j)-th grid point depends on its longitudinal 2α-interval neighbors (and latitudinal 2β) through the advection terms, on itself by the dissipation term, and on the external forcing ($F = 8$ in this work). When $\alpha = \beta = 1$, this model reduces to the two-dimension Lorenz-95 model [3]. The length-scale of this model is proportional to the inverse of α and β in each direction: for instance, the grid size for $\alpha = \beta = 2$ amounts to 347 km × 347 km. The time-scale is such that 0.05 time units are equivalent to 6 hours in real-time.

2.1 State Estimation

A standard approach to state estimation and prediction is to use a Monte Carlo (ensemble) approximation to the extended Kalman Filter, in which each ensemble member presents an initial state estimate of the weather system. These ensembles are propagated (for a set forecast time) through the underlying weather dynamics and the estimate (i.e., the mean value of these ensembles) is refined by measurements (i.e., updates) that are available through the sensor network. The particular approximation used in this work is the sequential ensemble square root filter [5] (EnSRF). In the EnSRF, the propagation of the mean state estimate and covariance matrix amounts to a nonlinear integration of ensemble members, improving the filtering of non-linearities compared to standard EKF techniques and mitigating the computational burden of maintaining a large covariance matrix [6,5]. The ensemble mean corresponds to the state estimate, and the covariance information can be obtained from the perturbation ensemble,

$$\mathbf{P} = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^T/(L_E - 1), \qquad \tilde{\mathbf{X}} \in \mathbb{R}^{L_S \times L_E} \tag{3}$$

where L_S is the number of state variables and L_E is the ensemble size. $\tilde{\mathbf{X}}$ is the perturbation ensemble defined as

$$\tilde{\mathbf{X}} = \eta \left(\mathbf{X} - \bar{\mathbf{x}} \times \mathbf{1}^T \right) \tag{4}$$

where $\mathbf{X}$ is the ensemble matrix, a row concatenation of each ensemble member, and $\bar{\mathbf{x}}$ is the ensemble mean, the row average of the ensemble matrix. $\eta(\geq 1)$ is the covariance inflation factor introduced to avoid underestimation of the covariance by finite ensemble size. The propagation step for EnSRF is the integration

$$\mathbf{X}^f(t + \Delta t) = \int_t^{t+\Delta t} \dot{\mathbf{X}} dt, \qquad \mathbf{X}(t) = \mathbf{X}^a(t), \tag{5}$$

with $\mathbf{X}^f$ and $\mathbf{X}^a$ denoting the forecast and analysis ensemble, respectively. The measurement update step for the EnSRF is

$$\bar{\mathbf{x}}^a = \bar{\mathbf{x}}^f + \mathbf{K}(\mathbf{y} - \mathbf{H}\bar{\mathbf{x}}^f) \tag{6}$$

$$\tilde{\mathbf{X}}^a = (\mathbf{I} - \mathbf{K}\mathbf{H})\tilde{\mathbf{X}}^f \tag{7}$$

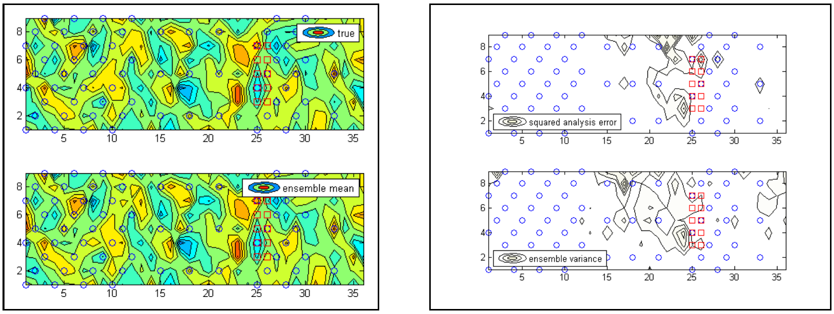
where $\mathbf{y}$ denotes the observation vector and $\mathbf{H}$ is the linearized observation matrix. $\mathbf{K}$ denotes the appropriate Kalman gain, which can be obtained by solving a nonlinear matrix equation stated in terms of $\mathbf{X}$[5]. The sequential update process avoids solving a nonlinear matrix equation and provides a faster method for determining $\mathbf{K}$. The ensemble update by the m-th observation is

$$\tilde{\mathbf{X}}^{m+1} = \tilde{\mathbf{X}}^{m} - \alpha_m \beta_m \mathbf{p}_i^m \tilde{\xi}_i^m, \tag{8}$$

$$\alpha_m = 1/\left(1 + \sqrt{\beta_m R_i}\right), \beta_m = 1/\left(\mathbf{P}_{ii}^m + R_i\right) \tag{9}$$

where measurement is taken for i-th state variable. $\mathbf{p}_i^m$, $\tilde{\xi}_i^m$, and $\mathbf{P}_{ii}^m$ are the i-th column, the i-th row, and the (i,i) element of the prior perturbation matrix $\mathbf{P}^m$ respectively. α_m is the factor for compensating the mismatch of the serial update and the batch update, while $\beta_m \mathbf{p}_l^m$ amounts to the Kalman gain.

Figure 1(a) shows an example true state of a Lorenz model (top) over the 36×9 state variables. The bottom frame shows the estimated state at the same time. This estimate is computed from an EnSRF using 200 ensemble members. Observations are taken at 66 fixed (routine) locations represented by blue circles; note that there are regions where routine observations are sparse, representing areas such as open ocean where regular measurements are hard to acquire. Figure 1(b) (top) shows the squared analysis error between true state and ensemble estimates from the upper figure, that is, the actual forecast error. The lower panel shows the ensemble variance, that is, the expected squared forecast error. Note that the expected and true error are largely correlated; using 200 ensemble members was enough to estimate the true model with reasonable error as shown in the figures.

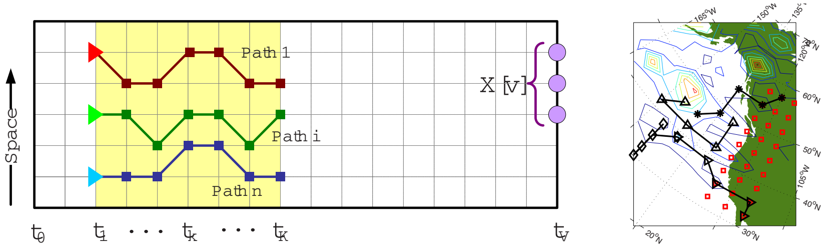


(a) True vs. Estimated State

(b) Performance Analysis

Fig. 1. (a) Top panel: the true state of the Lorenz system, where the intensity correlates with the state value. Lower panel: The estimated state of the system, using 200 ensemble members. (b) Top panel: the actual forecast error. Lower panel: the ensemble variance.

For the purposes of forecast error, we are typically interested in improving the forecast accuracy for some small region such as the coast of California, rather than the entire Pacific. A verification region is specified as $\mathbf{X}[v]$ and verification time t_v in our experiments, as shown by the red squares in Figure 1. Our goal is therefore to choose measurements of $\mathbf{X}$ at time t to minimize the forecast error at $\mathbf{X}[v]$ at time t_v.



(a) Targeting in grid space-time

(b) Four example targeting plans

Fig. 2. (a) Multi-UAV targeting in the grid space-time. (b) Targeting of four sensor platforms for the purpose of reducing the uncertainty of 3-day forecast over the west coast of North America.

3 A Targeting Algorithm for Multiple Mobile Sensor Platforms

The targeting problem is how to assign multiple sensor platforms (e.g. UAVs) to positions in the finite grid space-time (Figure 2a) in order to reduce the expected forecast uncertainty in the region of interest $\mathbf{X}[v]$. We define the targeting problem as selecting n paths consisting of K (size of the targeting time window) points that maximize the information gain at $\mathbf{X}[v]$ of the measurements taken along the selected paths.

A new, computationally efficient *backward selection* algorithm forms the backbone of the targeting approach. To address the computation resulting from the expense of determining the impact of each measurement choice on the uncertainty reduction in the verification site, the backward selection algorithm exploits the commutativity of mutual information. This enables the contribution of each measurement choice to be computed by propagating information backwards from the verification space/time to the search space/time. This significantly reduces the number of times that computationally expensive covariance updates must be performed. In addition, the proposed targeting algorithm employs a branch-and-bound search technique to reduce computation required to calculate payoffs for suboptimal candidates, utilizing a simple cost-to-go heuristics based on the diagonal assumption of the covariance matrix that provides an approximate upper bound of the actual information gain. The suggested heuristic does not guarantee an optimal solution; nevertheless, in practice it results in a substantial reduction in computation time while incurring minimal loss of optimality, which can be improved by relaxing a bounding constraint.

Figure 2(b) depicts an illustrative solution of the four-agent (black $\Diamond$, $\triangle$, $\triangleright$, $*$) targeting problem for enhancing the 3-day forecast of the west coast of North America (red $\Box$); the time interval between the marks is three hours.

The computation time of the targeting algorithm grows exponentially with the number of sensor platforms and the size of the targeting window increase, in spite of the reduction in computational cost. Thus, further approximations that decompose the computation and decision making into different topologies and choices on the planning horizon will be explored. These have been shown to avoid the combinatorial explosion of the computation time, and the performance of the approximation scheme turns out to depend highly on the communication topology between agents. The existence of inter-agent sharing of the up-to-date covariance information has also been shown to be essential to achieve performance.

4 Trajectory Learning

Given a series of desired target locations for additional measurements, an appropriate motion trajectory must be chosen between each pair of locations. Rather than directly optimizing the trajectory based on the current state of the weather system, the system will learn to predict the best trajectory that minimizes the forecast by examining past example trajectories. The advantage to this approach is that, once the predictive model is learned, each prediction can be made extremely quickly and adapted in real time as additional measurements are taken along the trajectory. The second advantage is that by careful selecting the learning technique, a large number of factors can be considered in both the weather system and the objective function, essentially optimizing against a number of different objectives, again without incurring a large computational penalty.

The problem of learning a model that minimizes the predicted forecast error is that of reinforcement learning, in which an agent takes actions and receives some reward signal. The goal of the agent is to maximize over its lifetime the expected received reward (or minimize the received cost) by learning to associate actions that maximize reward in different states. Reinforcement learning algorithms allow the agent to learn a policy $\pi : x \rightarrow a$, mapping state x to action a in order to maximize the reward.

In the weather domain, our cost function is the norm of the forecast error at the verification state variables ($\mathbf{X}[v]$) at the verification time t_v, so that the optimal policy π^* is

$$\pi^*(\mathbf{X}) = \operatorname*{argmin}_{\pi \in \Pi} E_{\mathbf{X}_{t_v}[v]} \left[\left\| (\tilde{\mathbf{X}}_{t_v}[v] | h(\pi), \mathbf{X}) - \mathbf{X}_{t_v}[v] \right\| \right] \tag{10}$$

If our set of actions is chosen to be a class of paths through space, such as polynomial splines interpolating the target points, then the policy attempts to choose the best spline to minimize our expected forecast error. Notice that this policy maps the current state $\mathbf{X}$ to the action a; however, the policy does not have access to the current weather state but only the current estimate of the weather given by the EnSRF. The learner therefore computes the policy that chooses actions based on the current estimate given by the mean $\tilde{\mathbf{X}}$ and covariance Σ of the ensemble.

In order to find the optimal policy π^*, a conventional reinforcement learning algorithm spends time trying different trajectories under different examples of weather conditions, and modelling how each trajectory predicts a different forecast error. The learning problem then becomes one of predicting, for a given EnSRF estimate $\tilde{\mathbf{X}}$ and Σ, the expected forecast error $\xi \in \mathbb{R}$ for each possible trajectory $a \in A$:

$$(\tilde{\mathbf{X}}, \Sigma) \times A \rightarrow \mathbb{R}. \tag{11}$$

Once this functional relationship is established, the controller simply examines the predicted error ξ for each action a given the current state estimate and chooses the action with the least error.

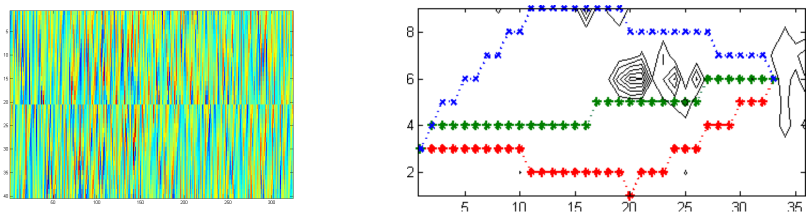
With access to a weather simulator such as the Lorenz model, we can simplify this learning problem by turning our reinforcement learning problem into a “supervised” learning problem, where the goal of the learner is not to predict the forecast error ξ of each possible trajectory conditioned on the current weather estimate, but rather to predict the best trajectory, a converting the regression problem of equation (11) into a classification problem, that is,

$$(\tilde{\mathbf{X}}, \Sigma) \rightarrow A. \tag{12}$$

Although regression and classification are closely linked (and one can often be written in terms of another), we can take advantage of some well-understood classification algorithms for computing policies. The classification algorithm used is the multi-class Support Vector Machine [7], which assigns a label (i.e., our optimal action) to each initial condition. The SVM is a good choice to learn our policy for two reasons: firstly, the SVM allows us to learn a classifier over the continuous state space $(\tilde{\mathbf{X}}, \Sigma)$. Secondly, the SVM is generally an efficient learner of large input spaces with a small number of samples; the SVM uses a technique known as the “kernel trick” [7] to perform classification by projecting each instance to a high-dimensional, non-linear space in which the inputs are linearly separable according to their class label.

4.1 Experimental Results

Training data for the Lorenz model was created by randomly generating initial conditions of the model, creating a set of ensemble members from random perturbations to the initial conditions and then propagating the model. Figure 3(a) shows a plot of 40 initial conditions used as training data created by running the model forward for several days and sampling a new initial condition every 6 hours, re-initializing the model every 5 days. Each row corresponds to a different training datum $\mathbf{X}_t$, and each column corresponds to a state variable X^i. While the data are fairly random, the learner can take advantage of the considerable temporal correlation; notice the clear discontinuity in the middle of the data where the model was re-initialized.

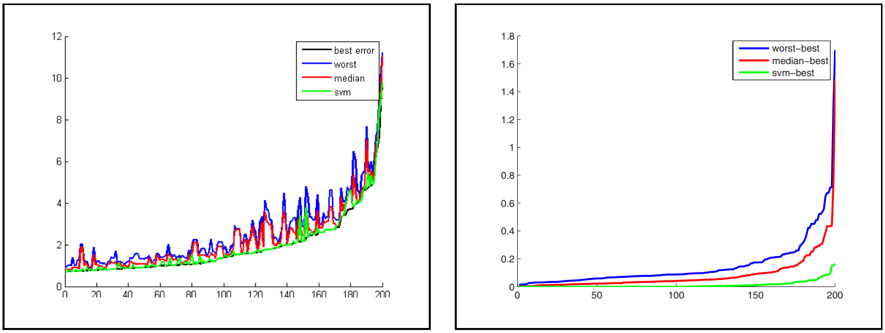


(a) Training Data

(b) Example trajectories

Fig. 3. (a) A plot of 40 training instances $\mathbf{X}$. Each column corresponds to a state variable X^i and each row is a different $\mathbf{X}_t$. (b) Three example trajectories from our action space A. Our action space consisted of 5 trajectories that span the region from the same start and end locations.

Each training instance was labelled with the corresponding optimal trajectory. We restricted the learner to a limited set of 5 actions or candidate trajectories, although this constraint will be relaxed in future work. All candidate trajectories started from the same mid-point of the left side of the region and ended at the same mid-point of the right side of the region. Three examples of the five trajectories are shown in Figure 3(b); the trajectories were chosen to be maximal distinct through the area of sparse routine observations in the centre of the region. From the initial condition, the model and ensemble were propagated for each trajectory. During this propagation, routine observations were taken every 5 time units and then a forecast was generated by propagating the ensemble for time equivalent to 2 and 4 days, without taking additional observations. The forecast error was then calculated by the difference between the ensemble estimate and the true value of the variables of the verification region. Each initial condition was labelled with the trajectory that minimized the resultant forecast error.



(a) Forecast Error

(b) Forecast Error Loss

Fig. 4. (a) Forecast error at the verification region after 2 days for the 200 largest losses in test data, sorted from least to greatest. (b) Forecast error loss, where the loss is taken with respect to the forecast error of the best trajectory.

Figure 4(a) shows the forecast error of best, median, worst and SVM trajectory for the 200 most difficult (highest forecast error) initial conditions in terms of the forecast error in the verification region. Notice that the forecast error of the SVM trajectory tracks the best trajectory relatively closely, indicating good performance. Figure 4(b) is an explicit comparison between the worst, median and SVM trajectories compared to the best trajectory for the same 200 most difficult training instances. Again, the SVM has relatively little loss (as measured by the difference between the forecast error of the SVM and the forecast error of the best trajectory) for many of these difficult cases.

In training the learner, two different kernel (non-linear projections in the SVM) were tested, specifically polynomial and radial basis function (RBF) kernels. Using cross-validation and a well-studied search method to identify the best kernel fit and size, a surprising result was that a low-order polynomial kernel resulted in the most accurate prediction of good trajectories. A second surprising result is that in testing different combinations of input data, such as filter mean alone, compared to filter mean and filter covariance, the filter covariance had relatively little effect on the SVM performance. This effect may be related to the restricted action class, but further investigation is warranted.

5 Conclusion

The spatio-temporal character of the data and chaotic behavior of the weather model makes adaptive observation problem challenging in the weather domain. We have described two adaptive observation techniques, including a targeting algorithm and a learning path planning algorithm. In the future, we plan to extend these results using the Navy's Coupled Ocean Atmosphere Prediction System (COAMPS), a full-scale regional weather research and forecasting model.

References

1. Lorenz, E.N., Emanuel, K.A.: Optimal sites for supplementary weather observations: Simulation with a small model. Journal of the Atmospheric Sciences **55**(3) (1998) 399–414
2. Morss, R., Emanuel, K., Snyder, C.: Idealized adaptive observation strategies for improving numerical weather prediction. Journal of the Atmospheric Sciences **58**(2) (2001)
3. Choi, H.L., How, J., Hansen, J.: Ensemble-based adaptive targeting of mobile sensor networks. In: Proc. of the American Control Conference (ACC). (To appear. 2007)
4. : http://www.aoc.noaa.gov/article_winterstorm.htm. Available online (last accessed June 2005)
5. Whitaker, J., Hamill, H.: Ensemble data assimilation without perturbed observations. Monthly Weather Review **130**(7) (2002) 1913–1924
6. Evensen, G., van Leeuwen, P.: Assimilation of altimeter data for the agulhas current using the ensemble kalman filter with a quasigeostrophic model. Monthly Weather Review **123**(1) (1996) 85–96
7. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines. Cambridge University Press, Cambridge, UK (2000)