Parameter Estimation and Adaptive Modeling Studies in Ocean Mixing

by

Eric Vincent Heubel

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2008

Author .....
Department of Mechanical Engineering
August 8, 2008

Certified by .....
Pierre F. J. Lermusiaux
Associate Professor
Thesis Supervisor

Accepted by .....
Lallit Anand
Chairman, Department Committee on Graduate Students

Parameter Estimation and Adaptive Modeling Studies in Ocean Mixing

by

Eric Vincent Heubel

Submitted to the Department of Mechanical Engineering
on August 8, 2008, in partial fulfillment of the
requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

In this thesis, we explore the different methods for parameter estimation in straightforward diffusion problems and develop ideas and distributed computational schemes for the automated evaluation of physical and numerical parameters of ocean models. This is one step of "adaptive modeling." Adaptive modeling consists of the automated adjustment of self-evaluating models in order to best represent an observed system. In the case of dynamic parameterizations, self-modifying schemes are used to learn the correct model for a particular regime as the physics change and evolve in time.

The parameter estimation methods are tested and evaluated on one-dimensional tracer diffusion problems. Existing state estimation methods and new filters, such as the unscented transform Kalman filter, are utilized in carrying out parameter estimation. These include the popular Extended Kalman Filter (EKF), the Ensemble Kalman Filter (EnKF) and other ensemble methods such as Error Subspace Statistical Estimation (ESSE) and Ensemble Adjustment Kalman Filter (EAKF), and the Unscented Kalman Filter (UKF). Among the aforementioned recursive state estimation methods, the so-called "adjoint method" is also applied to this simple study.

Finally, real data is examined for the applicability of such schemes in real-time forecasting using the MIT Multidisciplinary Simulation, Estimation, and Assimilation System (MSEAS). The MSEAS model currently contains the free surface hydrostatic primitive equation model from the Harvard Ocean Prediction System (HOPS), a barotropic tidal prediction scheme, and an objective analysis scheme, among other models and developing routines. The experiment chosen for this study is one which involved the Monterey Bay region off the coast of California in 2006 (MB06). Accurate vertical mixing parameterizations are essential in this well known upwelling region of the Pacific. In this realistic case, parallel computing will be utilized by scripting code runs in C-shell. The performance of the simulations with different parameters is evaluated quantitatively using Pattern Correlation Coefficient, Root Mean Squared error, and bias error. Comparisons quantitatively determined the most adequate model setup.

Thesis Supervisor: Pierre F. J. Lermusiaux
Title: Associate Professor

Acknowledgments

I would like to extend my gratitude to Professor Pierre Lermusiaux for his assistance during the course of this work, for his helpful direction and support. I thank Patrick Haley for his help in using the HOPS/MSEAS code, to Wayne Leslie for assistance with data conditioning, and to Oleg Logutov for the use of his tidal estimates. I am grateful for the help of the other students in the group in proofreading and discussion contributing to the organization of the thesis. I thank all my friends for their support and kindness through these past couple of years. Most of all, I wish to express my sincere appreciation to my family, for believing in me and helping me with their support, their love and faith.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background and Motivation

In the past several decades the stochastic methods of control have found more and more application in the fields of prediction and forecasting, parameter estimation, and model identification. With improvements in estimation methods and the growing complexity of existing models, it is necessary to establish the applicability of various schemes with respect to their complexity and computational efficiency. In the particular study, the uncertainty in the appropriate model, nonlinear nature of the parameter estimation problem for dynamically evolving systems, and the need for adequate means of measuring skill provide several issues to address in the selection of the ideal ocean modeling system. This, coupled with what resources are at hand and the desire for complete system automation, will set the proving ground for the system design of a fully automated adaptive model. The foundation which will be built upon is the Havard Ocean Prediction System (HOPS) Primitive Equation code developed at Harvard for regional ocean forecasts. The new MIT Multidisciplinary Simulation, Estimation, and Assimilation System (MSEAS) adds a new Objective Analysis package and barotropic tidal component model, among other packages. The software has been installed on a 266 CPU Verari system computer tower.

1.2 Goals

The main objectives in this thesis are to learn about parameter estimation methods through their implementation in simple idealized diffusion problems, then to implement a first version of the computational system for the automated performance evaluation of four-dimensional ocean models for various parameters using distributed computing. Specifically, the state estimation algorithms introduced in the next chapter are applied for the purpose of parameter estimation and results are compared to numerical and analytical solutions of straightforward test cases in one-dimensional tracer diffusion. The results of the simple application are utilized as guidance for the quantitative selection of physical and numerical parameters in the case of a four-dimensional ocean modeling system, MSEAS. Distributed computing software is utilized on advanced high performance computing machines in order to produce and analyze a variety of MSEAS ocean simulation options. For the analysis of the results MATLAB® software (Lermusiaux and Haley, Personal communication) is used and further developed as a means to compare complex four-dimensional ocean model output fields to irregularly-sampled, non-equispaced ocean data. These tools are used specifically to evaluate quality of barotropic tidal estimates in Monterey Bay 2006 regional ocean experiment. Results obtained will also help identify the quality of the set of numerical and physical parameters in Monterey Bay. The analysis of this particular region will aid in structuring a standard method in quantitative performance evaluation for selection of the best model parameters or models in various aspects of multidisciplinary ocean modeling.

Chapter 2

Parameter Estimation Methods

The purpose of this chapter is to explore the various existing methods of parameter estimation. In the future, these methods could be used for quantitatively selecting the most adequate closure, or sub-grid mixing model, in ocean simulations for adaptive modeling. The future goal is to reach the level at which the chosen model will be altered in accord with gathered observations as the dynamics evolve. Though this may be computationally intensive for the purpose of on-line adaptation, off-line results should at least identify the most prominent models for different regimes in various types of regions in the world oceans. The chosen parameter estimation methods and adaptive schemes are the popular Extended Kalman Filter (EKF), a recursive algorithm; the so-called "adjoint method," a type of "batch" algorithm; as well as Ensemble-based, and Unscented Transform methods. The first two require some linearization of the underlying dynamics of the system modeled. The latter methods, on the other hand, learn to represent the true nonlinear system by analyzing inputs and outputs of a variety of runs and parameters.

2.1 Background

As noted by Gelb et al. (1974) the problem of identifying constant parameters in a system of equations can be considered a special case of the general state estimation problem. The state of the system is augmented to include the unknown parameters of

interest in the dynamics. Naturally, such a procedure will make the state estimation problem non-linear, as will be made apparent in the following chapter. When utilizing this extension of state estimation methods, parameters involved in the system in question need not be constant; these may also be represented as a dynamic function of a stochastic forcing. By appropriately choosing the variance of this element of noise, the unknown parameter may be roughly limited to its expected range. Gelb et al. (1974) suggests a variance of the square of the expected range of deviation in the parameter divided by a characteristic time interval ($\mathbf{Q}_i = \frac{\Delta a_i^2}{\Delta t}$). An overview of the general state estimation methods is therefore in order.

2.1.1 Linear Dynamics

Continuous

If a linear dynamic system is concerned, a set of equations in the form of (2.1) is to be solved. Where the notation of Professor Harry Asada in his Identification, Estimation, and Learning course (MIT course ID 2.160) is used.

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{F}(t)\mathbf{x}(t) + \mathbf{G}(t)\mathbf{w}(t) + \mathbf{L}(t)\mathbf{u}(t) \\ \mathbf{z}(t) &= \mathbf{H}(t)\mathbf{x}(t) + \mathbf{v}(t) \end{aligned} \tag{2.1}$$

Where $\mathbf{z}$ is the observed output, $\mathbf{v}$ is measurement noise, $\mathbf{x}$ is the state, $\mathbf{w}$ is process noise, $\mathbf{u}$ is a control vector variable, which will quickly be ignored for the purposes of this study. The control term is dropped in the extension of the state estimation methods to parameter estimation as the goal is to minimize model uncertainty, rather than control the ocean response. To obtain the system state transition matrix of interest, the dynamic equations are first simplified. Assuming that all noise terms are zero and setting all control variables to zero, the homogeneous state equation remains (2.2).

$$\dot{\mathbf{x}}(t) = \mathbf{F}(t)\mathbf{x}(t) \tag{2.2}$$

Taking all inputs to be originally zero, a unit impulse is applied to the state

$$\mathbf{x}_i(t_0) = \begin{bmatrix} 0 \\ \vdots \\ \delta(t_0) \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} 1 \\ \vdots \\ i \\ \vdots \\ n \end{matrix}$$

and the system is integrated to obtain the response

$$\varphi_i(t, t_0) = \begin{bmatrix} x_1(t, t_0)_i \\ x_2(t, t_0)_i \\ \vdots \\ x_n(t, t_0)_i \end{bmatrix}$$

By properties of linearity, superposition and scaling can be used to determine the complete state of the system for any initial condition. These response vectors can be combined into a matrix as

$$\mathbf{\Phi} = \left[\varphi_1(t, t_0), \varphi_2(t, t_0), \ldots, \varphi_n(t, t_0)\right],$$

and so the full response is simplified to a matrix multiplication

$$\mathbf{x}(t) = \mathbf{\Phi}(t, t_0)\mathbf{x}(t_0).$$

In the above formulas $\mathbf{\Phi}(t, t_0)$ is known as the transition matrix. A few properties of this particular matrix are

$$\begin{gathered}
\frac{d\varphi_i(t,t_0)}{dt} = \mathbf{F}(t)\varphi_i(t,t_0) \\
\frac{d\mathbf{\Phi}(t,t_0)}{dt} = \mathbf{F}(t)\mathbf{\Phi}(t,t_0) \\
\mathbf{\Phi}(t,t) = \mathbf{I} \\
\mathbf{\Phi}(t_2,t_0) = \mathbf{\Phi}(t_2,t_1)\mathbf{\Phi}(t_1,t_0) \\
\mathbf{\Phi}(t,t_0)^{-1} = \mathbf{\Phi}(t_0,t)
\end{gathered}$$

for the homogeneous equation in the absence of external forcings. For stationary systems, $\mathbf{F}(t) = \mathbf{F}$ and is time invariant. In this case, the Taylor series expansion of $\mathbf{x}(t)$ about $\mathbf{x}(t_0)$ using $\dot{\mathbf{x}}(t_0) = \mathbf{F}\mathbf{x}(t_0)$, $\ddot{\mathbf{x}}(t_0) = \mathbf{F}\dot{\mathbf{x}}(t_0) = \mathbf{F}^2\mathbf{x}(t_0)$, and so forth leads to

$$\mathbf{x}(t) = \mathbf{x}(t_0) + \mathbf{F}(t-t_0)\mathbf{x}(t_0) + \frac{\mathbf{F}^2(t-t_0)^2}{2!}\mathbf{x}(t_0) + \ldots.$$

That is

$$\begin{aligned}
\mathbf{x}(t) &= \left[\mathbf{I} + \mathbf{F}(t-t_0) + \frac{\mathbf{F}^2(t-t_0)^2}{2!} + \ldots\right]\mathbf{x}(t_0) \\
&= \exp\{\mathbf{F}(t-t_0)\}\mathbf{x}(t_0).
\end{aligned}$$

So $\mathbf{\Phi}(t-t_0) = e^{\mathbf{F}(t-t_0)}$.

Discrete Time

Transitioning to the discrete case, the dynamical and measurement models are (2.3) in the absence of control term ($\mathbf{u}$) and taking the process noise propagation matrix as the identity.

$$\begin{gathered}
\mathbf{x}_k = \mathbf{\Phi}_{k-1}\mathbf{x}_{k-1} + \mathbf{w}_{k-1} \\
\mathbf{z}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k
\end{gathered} \tag{2.3}$$

Where $\mathbf{\Phi}_{k-1}$ is the discretized version of the transition or propagation matrix ($\mathbf{\Phi}_{k-1} = \mathbf{\Phi}(t_k, t_{k-1})$). A disruption of the homogeneous state equation occurs when a controlled input or noise is introduced to the system through $\mathbf{u}$ or $\mathbf{w}$ that excites the system response. Looking at (2.1) these inputs are disruptions of $\frac{d\mathbf{x}}{dt}$ and as such can be carried

through the integrator as perturbations in the state. The response to a control input, following the methodology presented in Gelb, 1974, is represented by

$$\Delta x_i(\tau) = \left(\mathbf{L}(\tau)\mathbf{u}(\tau)\right)_i \Delta\tau.$$

It is an impulse input in the state for a differential element of time $\Delta\tau$. The propagated effect can then be represented by carrying this impulse through the integration (represented as $\Delta\mathbf{x}(t)$).

$$\Delta\mathbf{x}(t) = \mathbf{\Phi}(t,\tau)\mathbf{L}(\tau)\mathbf{u}(\tau)\Delta\tau$$

The complete effect of the control input on the state can be viewed as a sum of short duration impulse disruptions and in the limit of $\Delta\tau \to 0$, this becomes an integral. For the forced system:

$$\mathbf{x}(t) = \int_{-\infty}^{t} \mathbf{\Phi}(t,\tau)\mathbf{L}(\tau)\mathbf{u}(\tau)d\tau$$

If an initial state is known prior to the start of the control input at t_0, then the solution can be represented as

$$\mathbf{x}(t) = \mathbf{x}(t_0) + \int_{t_0}^{t} \mathbf{\Phi}(t,\tau)\mathbf{L}(\tau)\mathbf{u}(\tau)d\tau$$

The continuous system can thus be easily rewritten in a discrete form where the dynamic equation in (2.1) is integrated and measurements are made discretely to obtain

$$\begin{gathered}\mathbf{x}(t) = \mathbf{\Phi}(t,t_0)\mathbf{x}(t_0) + \int_{t_0}^{t} \mathbf{\Phi}(t,\tau)\mathbf{G}(\tau)\mathbf{w}(\tau)d\tau + \int_{t_0}^{t} \mathbf{\Phi}(t,\tau)\mathbf{L}(\tau)\mathbf{u}(\tau)d\tau \\ \mathbf{x}_{k+1} = \mathbf{\Phi}_k\mathbf{x}_k + \mathbf{\Gamma}_k\mathbf{w}_k + \mathbf{\Lambda}_k\mathbf{u}_k \\ \mathbf{z}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k\end{gathered} \tag{2.4}$$

The integrals in (2.4) have been evaluated at t_{k+1}, where $\mathbf{\Phi}_k = \mathbf{\Phi}(t_{k+1}, t_k)$, $\mathbf{\Gamma}_k\mathbf{w}_k = \int_{t_k}^{t_{k+1}} \mathbf{\Phi}(t_{k+1},\tau)\mathbf{G}(\tau)\mathbf{w}(\tau)d\tau$, and $\mathbf{\Lambda}_k\mathbf{u}_k = \int_{t_k}^{t_{k+1}} \mathbf{\Phi}(t_{k+1},\tau)\mathbf{L}(\tau)\mathbf{u}(\tau)d\tau$ (Gelb et al., 1974).

2.1.2 Least Squares

In direct *least-squares estimation*, the minimum of the cost function

$$\mathbf{J} = (\mathbf{z} - \mathbf{H}\hat{\mathbf{x}})^T(\mathbf{z} - \mathbf{H}\hat{\mathbf{x}})$$

is sought, when provided with a perfect measurement $\mathbf{z}$ (i.e. $\hat{\mathbf{v}} = 0$). If the measurement vector is of equal or greater dimension than the state vector, this problem simplifies to

$$\frac{\partial \mathbf{J}}{\partial \hat{\mathbf{x}}} = 0$$

yielding

$$\hat{\mathbf{x}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{z}.$$

If the minimum of a weighted sum of squares (*weighted least square*) is desired instead, $\mathbf{J}$ becomes

$$\mathbf{J} = (\mathbf{z} - \mathbf{H}\hat{\mathbf{x}})^T\mathbf{R}^{-1}(\mathbf{z} - \mathbf{H}\hat{\mathbf{x}})$$

and

$$\hat{\mathbf{x}} = (\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}\mathbf{H}\mathbf{R}^{-1}\mathbf{z}.$$

The problem of solving for the minimum cost with this metric can be derived deterministically and is the manner in which the "adjoint method" performs state estimation (Bannister, 2001). Here, an analogy is drawn from the above mentioned performance metric (objective function) to the Kalman Filter. The Kalman Filter is derived to minimize the trace of the *a posteriori* error covariance matrix and does so recursively by only carrying forward information about the current estimate and error covariance. As such, both methods seek to minimize some form of an L2 norm.

2.1.3 Kalman Filter

In the Kalman filter approach, a better estimate is sought in a recursive manner by combining the current state estimate in a linear fashion with the current measurement in the form of (2.5).

$$\hat{\mathbf{x}}_k(+) = \mathbf{K}_k^1\hat{\mathbf{x}}_k(-) + \mathbf{K}_k^2\mathbf{z}_k \tag{2.5}$$

where $\hat{\mathbf{x}}(-)$ is the *a priori* state estimate and $\hat{\mathbf{x}}(+)$ the *a posteriori*. Taking the estimates to be a deviation about the truth ($\hat{\mathbf{x}}_k = \mathbf{x}_k^t + \tilde{\mathbf{x}}_k$), the equation can be rewritten for the estimation error as (2.6).

$$\tilde{\mathbf{x}}_k(+) = \left[\mathbf{K}_k^{1} + \mathbf{K}_k^{2}\mathbf{H}_k - \mathbf{I}\right]\mathbf{x}_k^t + \mathbf{K}_k^{1}\tilde{\mathbf{x}}_k(-) + \mathbf{K}_k^{2}\mathbf{v}_k \tag{2.6}$$

Having unbiased measurements sets $\mathbf{E}[\mathbf{v}_k] = 0$. Additionally, if the *a priori* error, $\tilde{\mathbf{x}}(-)$, is unbiased, the formulation requires that the *a posteriori* error also be unbiased, thus forcing the remaining nonzero term $\mathbf{x}_k^t$ to have a coefficient of zero, i.e. $\mathbf{K}^{1} = \mathbf{I} - \mathbf{K}^{2}\mathbf{H}$. Making this substitution into (2.5) and replacing $\mathbf{K}^{2}$ with $\mathbf{K}$, the equation is simplified.

$$\begin{gathered}\hat{\mathbf{x}}_k(+) = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\hat{\mathbf{x}}_k(-) + \mathbf{K}_k\mathbf{z}_k \\ \text{or} \\ \hat{\mathbf{x}}_k(+) = \hat{\mathbf{x}}_k(-) + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_k(-))\end{gathered} \tag{2.7}$$

By subtracting from (2.7) the true state the equation for the estimate error is obtained.

$$\begin{gathered}\underbrace{\tilde{\mathbf{x}}_k(+) + \mathbf{x}_k}_{\hat{\mathbf{x}}_k(+)} = \underbrace{\tilde{\mathbf{x}}_k(-) + \mathbf{x}_k}_{\hat{\mathbf{x}}_k(-)} + \mathbf{K}_k(\underbrace{\mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k}_{\mathbf{z}_k} - \mathbf{H}_k(\underbrace{\tilde{\mathbf{x}}_k(-) + \mathbf{x}_k}_{\hat{\mathbf{x}}_k(-)})) \\ \tilde{\mathbf{x}}_k(+) + \mathbf{x}_k = \tilde{\mathbf{x}}_k(-) + \mathbf{x}_k + \mathbf{K}_k(\mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k - \mathbf{H}_k(\tilde{\mathbf{x}}_k(-) + \mathbf{x}_k)) \\ \tilde{\mathbf{x}}_k(+) = \tilde{\mathbf{x}}_k(-) + \mathbf{K}_k(\mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k - \mathbf{H}_k\tilde{\mathbf{x}}_k(-) - \mathbf{H}_k\mathbf{x}_k) \\ \tilde{\mathbf{x}}_k(+) = \tilde{\mathbf{x}}_k(-) + \mathbf{K}_k(\mathbf{v}_k - \mathbf{H}_k\tilde{\mathbf{x}}_k(-))\end{gathered}$$

From this result, it is then possible to compute the new error covariance from the old by taking the expectation of this difference multiplied by its transpose (2.8).

$$\begin{aligned}\mathbf{E}\left[\tilde{\mathbf{x}}_k(+)\tilde{\mathbf{x}}_k(+)^T\right] &= \mathbf{E}\left\{[(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\tilde{\mathbf{x}}_k(-) + \mathbf{K}_k\mathbf{v}_k][(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\tilde{\mathbf{x}}_k(-) + \mathbf{K}_k\mathbf{v}_k]^T\right\} \\ \mathbf{P}_k(+) &= \mathbf{E}\Big\{(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\tilde{\mathbf{x}}_k(-)\tilde{\mathbf{x}}_k(-)^T(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T \\ &\quad + (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\tilde{\mathbf{x}}_k(-)\mathbf{v}_k^T\mathbf{K}_k^T + \mathbf{K}_k\mathbf{v}_k\tilde{\mathbf{x}}_k(-)^T(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T \\ &\quad + \mathbf{K}_k\mathbf{v}_k\mathbf{v}_k^T\mathbf{K}_k^T\Big\}\end{aligned} \tag{2.8}$$

Where the substitution of $\mathbf{E}[\tilde{\mathbf{x}}_k(-)\tilde{\mathbf{x}}_k(-)^T] = \mathbf{P}_k(-)$ can then be made, along with that of the error variance, $\mathbf{E}[\mathbf{v}_k\mathbf{v}_k^T] = \mathbf{R}_k$, and the simplifying assumption that measurement errors are uncorrelated with estimation errors, reducing (2.8) to (2.9).

$$\begin{aligned}
\mathbf{P}_k(+) &= (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k(-)(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \\
\mathbf{P}_k(+) &= (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k(-) - (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k(-)\mathbf{H}_k^T\mathbf{K}_k^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \\
\mathbf{P}_k(+) &= (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k(-) - \mathbf{P}_k(-)\mathbf{H}_k^T\mathbf{K}_k^T + \mathbf{K}_k\mathbf{H}_k\mathbf{P}_k(-)\mathbf{H}_k^T\mathbf{K}_k^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \\
\mathbf{P}_k(+) &= (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k(-) + [\mathbf{K}_k\mathbf{H}_k\mathbf{P}_k(-)\mathbf{H}_k^T + \mathbf{K}_k\mathbf{R}_k - \mathbf{P}_k(-)\mathbf{H}_k^T]\mathbf{K}_k^T \\
\mathbf{P}_k(+) &= (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k(-) + \{\mathbf{K}_k[\mathbf{H}_k\mathbf{P}_k(-)\mathbf{H}_k^T + \mathbf{R}_k] - \mathbf{P}_k(-)\mathbf{H}_k^T\}\mathbf{K}_k^T
\end{aligned} \tag{2.9}$$

Minimizing the earlier discussed least-squares cost function, $\mathbf{J}_k = \mathbf{E}[\tilde{\mathbf{x}}_k(+)^T\mathbf{S}\tilde{\mathbf{x}}_k(+)]$, weighted by any positive semidefinite scaling matrix $\mathbf{S}$ is equivalent to minimizing $\mathbf{J}_k = \text{trace}[\mathbf{P}_k(+)]$. Taking the derivative of the trace of (2.9) with respect to $\mathbf{K}$ and making use of the property $\frac{\partial}{\partial\mathbf{A}}[\text{trace}(\mathbf{ABA}^T)] = \mathbf{A}(\mathbf{B}+\mathbf{B}^T)$ the equation for $\mathbf{K}$, the Kalman gain, is obtained when $\frac{\partial\mathbf{J}}{\partial\mathbf{K}}$ is set to zero, (2.10).

$$\begin{aligned}
0 = -\mathbf{P}_k(-)^T\mathbf{H}_k^T - \mathbf{P}_k(-)\mathbf{H}_k^T + 2\mathbf{K}_k\mathbf{H}_k\mathbf{P}_k(-)\mathbf{H}_k^T + \mathbf{K}_k(\mathbf{R}_k+\mathbf{R}_k^T) \\
0 = -2(\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k(-)\mathbf{H}_k^T + 2\mathbf{K}_k\mathbf{R}_k \\
\mathbf{K}_k = \mathbf{P}_k(-)\mathbf{H}_k^T[\mathbf{H}_k\mathbf{P}_k(-)\mathbf{H}_k^T + \mathbf{R}_k]^{-1}
\end{aligned} \tag{2.10}$$

This Kalman gain is then substituted back into (2.9) for further simplification and to remove one of the variables relating the *a posteriori* covariance to the measurement matrix and *a priori* error covariance.

$$\begin{aligned}
\mathbf{P}_k(+) &= (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k(-) + \{\mathbf{P}_k(-)\mathbf{H}_k^T[\mathbf{H}_k\mathbf{P}_k(-)\mathbf{H}_k^T + \mathbf{R}_k]^{-1} \\
&\quad \times[\mathbf{H}_k\mathbf{P}_k(-)\mathbf{H}_k^T + \mathbf{R}_k] - \mathbf{P}_k(-)\mathbf{H}_k^T\}\mathbf{K}_k^T \\
\mathbf{P}_k(+) &= (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k(-) + [\mathbf{P}_k(-)\mathbf{H}_k^T - \mathbf{P}_k(-)\mathbf{H}_k^T]\mathbf{K}_k^T \\
\mathbf{P}_k(+) &= (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k(-) + 0\ \mathbf{K}_k^T \\
\mathbf{P}_k(+) &= (\mathbf{I}-\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k(-) \\
\mathbf{P}_k(+) &= (\mathbf{I}-\mathbf{P}_k(-)\mathbf{H}_k^T[\mathbf{H}_k\mathbf{P}_k(-)\mathbf{H}_k^T + \mathbf{R}_k]^{-1}\mathbf{H}_k)\mathbf{P}_k(-) \\
\mathbf{P}_k(+) &= \mathbf{P}_k(-) - \mathbf{P}_k(-)\mathbf{H}_k^T[\mathbf{H}_k\mathbf{P}_k(-)\mathbf{H}_k^T + \mathbf{R}_k]^{-1}\mathbf{H}_k\mathbf{P}_k(-)
\end{aligned} \tag{2.11}$$

From (2.3) the time-integrated estimate of the state for zero mean process noise, $\mathbf{w}$, is (2.12).

$$\hat{\mathbf{x}}_k(-) = \mathbf{\Phi}_{k-1}\hat{\mathbf{x}}_{k-1}(+) \tag{2.12}$$

Subtracting (2.12) from (2.3) and taking the expectation of this result transposed with itself, the extrapolated covariance becomes (2.13).

$$\mathbf{P}_k(-) = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1}(+)\mathbf{\Phi}_{k-1}^T + \mathbf{Q}_{k-1} \tag{2.13}$$

2.1.4 Nonlinear Systems

Up until this point, linear systems of equations have been explored. The application of the EKF and other nonlinear estimation (Data Assimilation or Parameter Estimation) schemes comes when dealing with dynamic equations that are not linear. Now instead of (2.1), the system of equations is of the form of (2.14)

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{f}(\mathbf{x}(t), t) + \mathbf{w}(t) \\ \mathbf{z}_k &= \mathbf{h}_k(\mathbf{x}(t_k)) + \mathbf{v}_k \end{aligned} \tag{2.14}$$

where the dynamics are continuous, and measurements are discrete. By integration, an equation for the future state can be obtained (2.15).

$$\mathbf{x}(t) = \mathbf{x}(t_{k-1}) + \int_{t_{k-1}}^{t} \mathbf{f}(\mathbf{x}(\tau), \tau)d\tau + \int_{t_{k-1}}^{t} \mathbf{w}(\tau)d\tau \tag{2.15}$$

Taking the expectation of (2.15) followed by a derivative in time the equation reduces to (2.16).

$$\begin{aligned} \frac{d}{dt}\mathbf{E}[\mathbf{x}(t)] &= \frac{d}{dt}\mathbf{E}\left[\mathbf{x}(t_{k-1}) + \int_{t_{k-1}}^{t} \mathbf{f}(\mathbf{x}(\tau), \tau)d\tau + \int_{t_{k-1}}^{t} \mathbf{w}(\tau)d\tau\right] \\ \frac{d}{dt}\mathbf{E}[\mathbf{x}(t)] &= \frac{d}{dt}\hat{\mathbf{x}}(t_{k-1}) + \frac{d}{dt}\int_{t_{k-1}}^{t} \mathbf{E}[\mathbf{f}(\mathbf{x}(\tau), \tau)]d\tau + \frac{d}{dt}\int_{t_{k-1}}^{t} \mathbf{E}\left[\mathbf{w}(\tau)\right]d\tau \\ \frac{d}{dt}\mathbf{E}[\mathbf{x}(t)] &= \mathbf{E}[\mathbf{f}(\mathbf{x}(t), t)] \\ \frac{d}{dt}\hat{\mathbf{x}}(t) &= \hat{\mathbf{f}}(\mathbf{x}(t), t) \end{aligned} \tag{2.16}$$

From the above equations, the expectation of $\mathbf{x}$ can be integrated with which the covariance can then be computed (2.17).

$$\begin{aligned}
\mathbf{P}(t) &\equiv \mathbf{E}\left[[\hat{\mathbf{x}}(t) - \mathbf{x}(t)][\hat{\mathbf{x}}(t) - \mathbf{x}(t)]^T\right] \\
\mathbf{P}(t) &= \mathbf{E}[\hat{\mathbf{x}}(t)\hat{\mathbf{x}}(t)^T] - \mathbf{E}[\hat{\mathbf{x}}(t)\mathbf{x}(t)^T] - \mathbf{E}[\mathbf{x}(t)\hat{\mathbf{x}}(t)^T] + \mathbf{E}[\mathbf{x}(t)\mathbf{x}(t)^T] \\
\mathbf{P}(t) &= \mathbf{E}[\mathbf{x}(t)\mathbf{x}(t)^T] - \hat{\mathbf{x}}(t)\hat{\mathbf{x}}(t)^T
\end{aligned} \tag{2.17}$$

The equation that defines the propagation of the state covariance is then defined based on (2.15) and (2.16).

$$\begin{aligned}
\frac{d}{dt}\mathbf{P}(t) &= \frac{d}{dt}\mathbf{E}\left[[\hat{\mathbf{x}}(t) - \mathbf{x}(t)][\hat{\mathbf{x}}(t) - \mathbf{x}(t)]^T\right] \\
\frac{d}{dt}\mathbf{P}(t) &= \frac{d}{dt}\mathbf{E}\left[\mathbf{x}(t_{k-1})\mathbf{x}(t_{k-1})^T\right] + \frac{d}{dt}\mathbf{E}\left[\mathbf{x}(t_{k-1})\int_{t_{k-1}}^{t}\mathbf{f}(\mathbf{x}(\tau),\tau)^T d\tau\right] \\
&\quad + \frac{d}{dt}\mathbf{E}\left[\mathbf{x}(t_{k-1})\int_{t_{k-1}}^{t}\mathbf{w}(\tau)^T d\tau\right] + \frac{d}{dt}\mathbf{E}\left[\int_{t_{k-1}}^{t}\mathbf{f}(\mathbf{x}(\tau),\tau)d\tau\mathbf{x}(t_{k-1})^T\right] \\
&\quad + \frac{d}{dt}\mathbf{E}\left[\int_{t_{k-1}}^{t}\mathbf{f}(\mathbf{x}(\tau),\tau)d\tau\int_{t_{k-1}}^{t}\mathbf{f}(\mathbf{x}(\tau),\tau)^T d\tau\right] \\
&\quad + \frac{d}{dt}\mathbf{E}\left[\int_{t_{k-1}}^{t}\mathbf{f}(\mathbf{x}(\tau),\tau)d\tau\int_{t_{k-1}}^{t}\mathbf{w}(\tau)^T d\tau\right] \\
&\quad + \frac{d}{dt}\mathbf{E}\left[\int_{t_{k-1}}^{t}\mathbf{w}(\tau)d\tau\mathbf{x}(t_{k-1})^T\right] \\
&\quad + \frac{d}{dt}\mathbf{E}\left[\int_{t_{k-1}}^{t}\mathbf{w}(\tau)d\tau\int_{t_{k-1}}^{t}\mathbf{f}(\mathbf{x}(\tau),\tau)^T d\tau\right] \\
&\quad + \frac{d}{dt}\mathbf{E}\left[\int_{t_{k-1}}^{t}\mathbf{w}(\tau)d\tau\int_{t_{k-1}}^{t}\mathbf{w}(\tau)^T d\tau\right] - \frac{d}{dt}\left(\hat{\mathbf{x}}(t)\hat{\mathbf{x}}(t)^T\right) \\
\frac{d}{dt}\mathbf{P}(t) &= \mathbf{E}\left[\mathbf{x}(t)\mathbf{f}(\mathbf{x}(t),t)^T\right] + \mathbf{E}\left[\mathbf{f}(\mathbf{x}(t),t)\mathbf{x}(t)^T\right] + \mathbf{Q}(t) \\
&\quad - \hat{\mathbf{f}}(\mathbf{x}(t),t)\hat{\mathbf{x}}(t)^T - \hat{\mathbf{x}}(t)\hat{\mathbf{f}}(\mathbf{x}(t),t)^T
\end{aligned} \tag{2.18}$$

2.2 Extended Kalman Filter

The EKF has become an algorithm that is often used and is well known in the field of system control (Julier and Uhlmann, 2004). It makes use of a linearization of the system dynamics to which it applies Rudolf E. Kalman's linear filter (Kalman, 1960). The algorithm is a set of equations designed to recursively minimize the trace

of the *a posteriori* covariance matrix of the state variables in question.

Whereas for linear systems

$$\begin{aligned}\hat{\mathbf{f}}(\mathbf{x}(t),t) &= \mathbf{E}[\mathbf{F}(t)\mathbf{x}(t)] \\ &= \mathbf{F}(t)\hat{\mathbf{x}}(t) \\ &= \mathbf{f}(\hat{\mathbf{x}}(t),t)\end{aligned}$$

in the case of nonlinear systems

$$\begin{aligned}\hat{\mathbf{f}}(\mathbf{x}(t),t) &= \int_{-\infty}^{\infty}\ldots\int_{-\infty}^{\infty}\mathbf{f}(\mathbf{x}(t),t)p(\mathbf{x},t)dx_1\ldots dx_n \\ &\neq \mathbf{f}(\hat{\mathbf{x}}(t),t)\end{aligned}$$

The EKF is obtained by simplifying these equations through the linearization of the dynamics via a first order Taylor expansion about the conditional mean of the state.

$$\mathbf{f}(\mathbf{x}(t),t) = \mathbf{f}(\hat{\mathbf{x}}(t),t) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}}(\mathbf{x}-\hat{\mathbf{x}}) + \ldots \tag{2.19}$$

In this fashion, the expectation of the dynamic equation reduces to $\hat{\mathbf{f}}(\mathbf{x}(t),t) = \mathbf{f}(\hat{\mathbf{x}}(t),t)$. Then, introducing these terms into (2.18) and using the simplified notation for the Jacobian

$$\begin{aligned}\mathbf{F}(\hat{\mathbf{x}}(t),t) &= \{\mathbf{f}_{ij}(\hat{\mathbf{x}}(t),t)\} \\ \mathbf{f}_{ij}(\hat{\mathbf{x}}(t),t) &= \left.\frac{\partial \mathbf{f}_i(\mathbf{x}(t),t)}{\partial \mathbf{x}_j(t)}\right|_{\mathbf{x}(t)=\hat{\mathbf{x}}(t)}\end{aligned}$$

The differential equation for the covariance matrix (the Riccati equation) can be evaluated.

$$\begin{aligned}\frac{d}{dt}\mathbf{P}(t) &= \mathbf{E}\left[\mathbf{x}(t)\mathbf{f}(\mathbf{x}(t),t)^T\right] + \mathbf{E}\left[\mathbf{f}(\mathbf{x}(t),t)\mathbf{x}(t)^T\right] + \mathbf{Q}(t) \\ &\quad -\hat{\mathbf{f}}(\mathbf{x}(t),t)\hat{\mathbf{x}}(t)^T - \hat{\mathbf{x}}(t)\hat{\mathbf{f}}(\mathbf{x}(t),t)^T \\ \frac{d}{dt}\mathbf{P}(t) &= \hat{\mathbf{x}}(t)\mathbf{f}(\hat{\mathbf{x}}(t),t)^T + \mathbf{E}\left[\mathbf{x}(t)(\mathbf{x}-\hat{\mathbf{x}})^T\mathbf{F}(\hat{\mathbf{x}}(t),t)^T\right] + \mathbf{f}(\hat{\mathbf{x}}(t),t)\hat{\mathbf{x}}(t)^T \\ &\quad +\mathbf{E}\left[\mathbf{F}(\hat{\mathbf{x}}(t),t)(\mathbf{x}-\hat{\mathbf{x}})\mathbf{x}(t)^T\right] + \mathbf{Q}(t) - \mathbf{f}(\hat{\mathbf{x}}(t),t)\hat{\mathbf{x}}(t)^T - \hat{\mathbf{x}}(t)\mathbf{f}(\hat{\mathbf{x}}(t),t)^T \\ \frac{d}{dt}\mathbf{P}(t) &= \mathbf{P}(t)\mathbf{F}(\hat{\mathbf{x}}(t),t)^T + \mathbf{F}(\hat{\mathbf{x}}(t),t)\mathbf{P}(t) + \mathbf{Q}(t)\end{aligned} \tag{2.20}$$

With these equations it is assumed that the propagated state and state covariance

can be obtained as an *a priori* estimate of the future state. It is then necessary to update these variables with measurement information to produce a better estimate. Using the same procedure as for the linear case, a linear function of the *a priori* and measurement values for the *a posteriori* updated value of the form of (2.5) is desired.

$$\hat{\mathbf{x}}_k(+) = \mathbf{K}_k^{\mathit{1}}\hat{\mathbf{x}}_k(-) + \mathbf{K}_k^{\mathit{2}}\mathbf{z}_k$$

Again used with the estimation errors prior to and after the update, defined as

$$\tilde{\mathbf{x}}_k(-) \equiv \hat{\mathbf{x}}_k(-) - \mathbf{x}_k$$
$$\tilde{\mathbf{x}}_k(+) \equiv \hat{\mathbf{x}}_k(+) - \mathbf{x}_k$$

substituting

$$\begin{aligned}
\hat{\mathbf{x}}_k(+) - \mathbf{x}_k &= \mathbf{K}_k^{\mathit{1}}\hat{\mathbf{x}}_k(-) + \mathbf{K}_k^{\mathit{2}}\mathbf{z}_k - \mathbf{x}_k \\
\tilde{\mathbf{x}}_k(+) &= \mathbf{K}_k^{\mathit{1}}\hat{\mathbf{x}}_k(-) + \mathbf{K}_k^{\mathit{2}}\mathbf{z}_k + \underbrace{\tilde{\mathbf{x}}_k(-) - \hat{\mathbf{x}}_k(-)}_{-\mathbf{x}_k} \\
\tilde{\mathbf{x}}_k(+) &= \mathbf{K}_k^{\mathit{1}}\hat{\mathbf{x}}_k(-) + \underbrace{\mathbf{K}_k^{\mathit{2}}\mathbf{h}_k(\mathbf{x}_k) + \mathbf{K}_k^{\mathit{2}}\mathbf{v}_k}_{\mathbf{K}_k^{\mathit{2}}\mathbf{z}_k} + \tilde{\mathbf{x}}_k(-) - \hat{\mathbf{x}}_k(-)
\end{aligned}.$$

An unbiased estimate *a posteriori* is required and $\mathbf{E}[\tilde{\mathbf{x}}_k(-)] = \mathbf{E}[\mathbf{v}_k] = 0$ is recalled. The expectation of the above equation then yields:

$$0 = \mathbf{K}_k^{\mathit{1}}\hat{\mathbf{x}}_k(-) + \mathbf{K}_k^{\mathit{2}}\hat{\mathbf{h}}_k(\mathbf{x}_k) - \hat{\mathbf{x}}_k(-)$$
$$\mathbf{K}_k^{\mathit{1}}\hat{\mathbf{x}}_k(-) = \hat{\mathbf{x}}_k(-) - \mathbf{K}_k^{\mathit{2}}\hat{\mathbf{h}}_k(\mathbf{x}_k)$$

Back-substituting and changing $\mathbf{K}_k^{\mathit{2}}$ to $\mathbf{K}_k$, as in the linear case produces the EKF version of the state update

$$\begin{aligned}
\hat{\mathbf{x}}_k(+) &= \hat{\mathbf{x}}_k(-) - \mathbf{K}_k\hat{\mathbf{h}}_k(\mathbf{x}_k) + \mathbf{K}_k\mathbf{z}_k \\
\hat{\mathbf{x}}_k(+) &= \hat{\mathbf{x}}_k(-) + \mathbf{K}_k[\mathbf{z}_k - \hat{\mathbf{h}}_k(\mathbf{x}_k)] \\
\tilde{\mathbf{x}}_k(+) &= \tilde{\mathbf{x}}_k(-) + \mathbf{K}_k[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k) + \mathbf{v}_k] \\
\tilde{\mathbf{x}}_k(+) &= \tilde{\mathbf{x}}_k(-) + \mathbf{K}_k[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)] + \mathbf{K}_k\mathbf{v}_k
\end{aligned} \tag{2.21}$$

Then the *a posteriori* state error covariance can be obtained from the above by taking the expectation of the $\tilde{\mathbf{x}}_k(+)$ multiplied by its transpose.

$$\begin{aligned}
\mathbf{P}_k(+) &= \mathbf{E}[\tilde{\mathbf{x}}_k(+)\tilde{\mathbf{x}}_k(+)^T] \\
\mathbf{P}_k(+) &= \mathbf{E}\Big\{\tilde{\mathbf{x}}_k(-)\tilde{\mathbf{x}}_k(-)^T + \mathbf{K}_k[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]\tilde{\mathbf{x}}_k(-)^T + \mathbf{K}_k\mathbf{v}_k\tilde{\mathbf{x}}_k(-)^T \\
&\quad +\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\mathbf{K}_k^T \\
&\quad +\mathbf{K}_k[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)][\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\mathbf{K}_k^T \\
&\quad +\mathbf{K}_k\mathbf{v}_k[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\mathbf{K}_k^T \\
&\quad +\tilde{\mathbf{x}}_k(-)\mathbf{v}_k^T\mathbf{K}_k^T + \mathbf{K}_k[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]\mathbf{v}_k^T\mathbf{K}_k^T + \mathbf{K}_k\mathbf{v}_k\mathbf{v}_k^T\mathbf{K}_k^T\Big\} \\
\mathbf{P}_k(+) &= \mathbf{P}_k(-) + \mathbf{K}_k\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]\tilde{\mathbf{x}}_k(-)^T\right] \\
&\quad +\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]\mathbf{K}_k^T \\
&\quad +\mathbf{K}_k\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)][\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]\mathbf{K}_k^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T
\end{aligned} \tag{2.22}$$

Where $\mathbf{P}_k(-) = \mathbf{E}[\tilde{\mathbf{x}}_k(-)\tilde{\mathbf{x}}_k(-)^T]$, $\mathbf{R}_k = \mathbf{E}[\mathbf{v}_k\mathbf{v}_k^T]$, and independence of $\mathbf{v}_k$ with respect to other terms has been assumed. As before, if the mean square error function

$$\mathbf{J}_k = \mathbf{E}[\tilde{\mathbf{x}}_k(+)^T\mathbf{S}\tilde{\mathbf{x}}_k(+)]$$

is to be minimized for any positive definite $\mathbf{S}$ this is the same as minimizing

$$\mathbf{J}_k = \mathbf{E}[\tilde{\mathbf{x}}_k(+)^T\tilde{\mathbf{x}}_k(+)] = \text{trace}[\mathbf{P}_k(+)].$$

To identify the Kalman gain that minimizes this covariance during the assimilation process, the derivative of this cost function with respect to $\mathbf{K}$ is set to zero.

$$\begin{aligned}
0 &\equiv \frac{\partial\mathbf{J}_k}{\partial\mathbf{K}_k} = \frac{\partial\text{trace}[\mathbf{P}_k(+)]}{\partial\mathbf{K}_k} \\
0 &= \mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right] + \mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right] \\
&\quad +2\mathbf{K}_k\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)][\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right] + 2\mathbf{K}_k\mathbf{R}_k \\
0 &= \mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right] \\
&\quad +\mathbf{K}_k\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)][\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right] + \mathbf{K}_k\mathbf{R}_k
\end{aligned}$$

yielding

$$\begin{aligned}\mathbf{K}_k &= -\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right] \\ &\times \left\{\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)][\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]+\mathbf{R}_k\right\}^{-1}\end{aligned} \tag{2.23}$$

Back-substituting into the original covariance update equation results in a simpler form of the EKF error covariance update.

$$\begin{aligned}\mathbf{P}_k(+) &= \mathbf{P}_k(-)+\mathbf{K}_k\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]\tilde{\mathbf{x}}_k(-)^T\right] \\ &\quad +\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]\mathbf{K}_k^T \\ &\quad +\mathbf{K}_k\left\{\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)][\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]+\mathbf{R}_k\right\}\mathbf{K}_k^T \\ \mathbf{P}_k(+) &= \mathbf{P}_k(-)+\mathbf{K}_k\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]\tilde{\mathbf{x}}_k(-)^T\right] \\ &\quad +\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]\mathbf{K}_k^T \\ &\quad +\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right] \\ &\quad \times\left\{\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)][\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]+\mathbf{R}_k\right\}^{-1} \\ &\quad \times\left\{\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)][\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]+\mathbf{R}_k\right\} \\ &\quad \times\left\{\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)][\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]+\mathbf{R}_k\right\}^{-T} \\ &\quad \times\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]^T \\ \mathbf{P}_k(+) &= \mathbf{P}_k(-)+\mathbf{K}_k\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]\tilde{\mathbf{x}}_k(-)^T\right] \\ &\quad +\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]\mathbf{K}_k^T+\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right] \\ &\quad \times\left\{\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)][\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]+\mathbf{R}_k\right\}^{-T} \\ &\quad \times\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]^T \\ \mathbf{P}_k(+) &= \mathbf{P}_k(-)+\mathbf{K}_k\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]\tilde{\mathbf{x}}_k(-)^T\right] \\ &\quad -\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right] \\ &\quad \times\left\{\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)][\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]+\mathbf{R}_k\right\}^{-T} \\ &\quad \times\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]^T+\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right] \\ &\quad \times\left\{\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)][\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]+\mathbf{R}_k\right\}^{-T} \\ &\quad \times\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]^T\right]^T \\ \mathbf{P}_k(+) &= \mathbf{P}_k(-)+\mathbf{K}_k\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k)-\hat{\mathbf{h}}_k(\mathbf{x}_k)]\tilde{\mathbf{x}}_k(-)^T\right]\end{aligned} \tag{2.24}$$

The extended Kalman filter algorithm then further simplifies the nonlinearity of the

measurement function (which depends on the probability density function of the state variable $\mathbf{x}$) by a truncation of the Taylor series expansion of this function.

$$\mathbf{h}_k(\mathbf{x}_k) = \mathbf{h}_k(\hat{\mathbf{x}}_k(-)) + \mathbf{H}_k(\hat{\mathbf{x}}_k(-))(\mathbf{x}_k - \hat{\mathbf{x}}(-)) + \ldots$$

Again, denoting the Jacobian of the nonlinear function $\mathbf{h}_k(\mathbf{x}_k)$ as $\mathbf{H}_k(\mathbf{x}_k)$.

$$\mathbf{H}_k(\hat{\mathbf{x}}_k(-)) = \frac{\partial \mathbf{h}_k(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}=\hat{\mathbf{x}}_k(-)}$$

Substituting these first two terms into the equations for the Kalman gain, $\mathbf{K}_k$ (2.23), and covariance update (2.24) gives the final form:

$$\begin{aligned}
\mathbf{K}_k &= -\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)[\mathbf{H}_k(\hat{\mathbf{x}}_k(-))(\mathbf{x}_k - \hat{\mathbf{x}}_k(-))]^T\right] \\
&\quad \times \left\{\mathbf{E}\left[[\mathbf{H}_k(\hat{\mathbf{x}}_k(-))(\mathbf{x}_k - \hat{\mathbf{x}}_k(-))][\mathbf{H}_k(\hat{\mathbf{x}}_k(-))(\mathbf{x}_k - \hat{\mathbf{x}}_k(-))]^T\right] + \mathbf{R}_k\right\}^{-1} \\
\mathbf{K}_k &= -\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)(\mathbf{x}_k - \hat{\mathbf{x}}_k(-))^T\mathbf{H}_k(\hat{\mathbf{x}}_k(-))^T\right] \\
&\quad \times \left\{\mathbf{E}\left[\mathbf{H}_k(\hat{\mathbf{x}}_k(-))(\mathbf{x}_k - \hat{\mathbf{x}}_k(-))(\mathbf{x}_k - \hat{\mathbf{x}}_k(-))^T\mathbf{H}_k(\hat{\mathbf{x}}_k(-))^T\right] + \mathbf{R}_k\right\}^{-1} \\
\mathbf{K}_k &= -\mathbf{E}\left[\tilde{\mathbf{x}}_k(-)(-\tilde{\mathbf{x}}_k(-))^T\right]\mathbf{H}_k(\hat{\mathbf{x}}_k(-))^T \\
&\quad \times \left\{\mathbf{H}_k(\hat{\mathbf{x}}_k(-))\mathbf{E}\left[(-\tilde{\mathbf{x}}_k(-))(-\tilde{\mathbf{x}}_k(-))^T\right]\mathbf{H}_k(\hat{\mathbf{x}}_k(-))^T + \mathbf{R}_k\right\}^{-1} \\
\mathbf{K}_k &= \mathbf{P}_k(-)\mathbf{H}_k(\hat{\mathbf{x}}_k(-))^T\left[\mathbf{H}_k(\hat{\mathbf{x}}_k(-))\mathbf{P}_k(-)\mathbf{H}_k(\hat{\mathbf{x}}_k(-))^T + \mathbf{R}_k\right]^{-1}
\end{aligned} \tag{2.25}$$

$$\begin{aligned}
\mathbf{P}_k(+) &= \mathbf{P}_k(-) + \mathbf{K}_k\mathbf{E}\left[[\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{h}}_k(\mathbf{x}_k)]\tilde{\mathbf{x}}_k(-)^T\right] \\
\mathbf{P}_k(+) &= \mathbf{P}_k(-) + \mathbf{K}_k\mathbf{E}\left[\mathbf{H}_k(\hat{\mathbf{x}}_k(-))(\mathbf{x}_k - \hat{\mathbf{x}}_k(-))\tilde{\mathbf{x}}_k(-)^T\right] \\
\mathbf{P}_k(+) &= \mathbf{P}_k(-) + \mathbf{K}_k\mathbf{H}_k(\hat{\mathbf{x}}_k(-))\mathbf{E}\left[(-\tilde{\mathbf{x}}_k(-))\tilde{\mathbf{x}}_k(-)^T\right] \\
\mathbf{P}_k(+) &= \mathbf{P}_k(-) - \mathbf{K}_k\mathbf{H}_k(\hat{\mathbf{x}}_k(-))\mathbf{P}_k(-) \\
\mathbf{P}_k(+) &= [\mathbf{I} - \mathbf{K}_k\mathbf{H}_k(\hat{\mathbf{x}}_k(-))]\mathbf{P}_k(-)
\end{aligned} \tag{2.26}$$

The EKF differs from the linearized Kalman filter in that this recursive algorithm uses the previous best estimate and linearizes the equations about this particular state to predict the *a priori* estimate. In contrast, the linearized Kalman filter simply uses an original state estimate about which it simplifies the complex dynamics to a set of linear equations.

Though a very common method, the EKF does have its limitations. The lin-

earization of the nonlinear system dynamics and the computation of the full error covariance matrix are significant computational expenses, especially for the purpose of on-line parameter estimation.

2.3 Adjoint Method

Adjoint models are used in optimal analysis, in sensitivity analysis, and in stability analysis. Unlike previous methods, use of a model adjoint allows increased computational speed and sensitivity measures that would otherwise require an ensemble of test cases. That is, for little added cost to a *tangent linear model* (TLM) the application of the so-called "adjoint method" can determine the sensitivity of a cost functional (through that of the model output) with respect to the model input, or model parameters. The TLM is termed as such because linearization is performed about each control input parameter at distinct time step; that is, the model consists of linear segments everywhere tangent to the trajectory the control. In this manner, a complex nonlinear system is made linear, at least in the piecewise sense. A fundamental assumption to the adjoint method is that the above linearity holds for the underlying dynamics of the model. Model solutions focus on determining and reducing a cost or objective function, $\mathbf{J}$, of outputs (forecasts) compared to measurements. What has been termed sensitivity analysis in the past is derived by comparing a control solution's cost functional for a specific input to that of perturbed input parameters, a (or state). In this manner, an approximation to the sensitivity (i.e. $\Delta\mathbf{J}/\Delta a$) is obtained. However, a perturbation, Δa, in the inputs may differ vastly from another introduced disturbance of similar amplitude and structure in a slightly different location; such is often the case in short term forecasts. The adjoint, on the other hand, starts again with a control solution, but instead of introducing a perturbation in the input, the sensitivity of the cost function with respect to the output is determined. This is often a simpler process as typically the cost functional is a simple user defined metric; whereas the dependence of the chosen statistic on input parameters is defined by the complex nature of the physics or dynamics represented by the model. The

sensitivity of the cost metric with respect to system outputs (or model data misfits) may then be found as easily as taking the derivative of $\mathbf{J}$ with respect to the output state, b. Then the change in $\mathbf{J}$ with respect to any perturbation location in b can be obtained by $\Delta\mathbf{J} \approx \sum_k \frac{\partial \mathbf{J}}{\partial b_k}\Delta b_k$ which is a first-order Taylor series approximation to $\Delta\mathbf{J}$. Yet this is not the solution sought. Interest lies in the changes in the cost function with respect to alterations in the input a (boundary and initial conditions, or other model parameters). That is, the equation $\Delta\mathbf{J} \approx \sum_k \frac{\partial \mathbf{J}}{\partial a_k}\Delta a_k$ is desired. To determine the relationship between $\frac{\partial \mathbf{J}}{\partial a}$ and $\frac{\partial \mathbf{J}}{\partial b}$, since b is obtained from a one simply needs to determine the function relating the two. A model is denoted as $b = B(a)$ with input a model operator B and output b. If a' represents a perturbation of the inputs, then $\Delta b_j \approx b'_j = \sum_k \frac{\partial b_j}{\partial a_k} a'_k$ is an approximation of the output perturbation in b. The vector first derivative $\left(\frac{\partial b_j}{\partial a_k}\right)$ is known as the Jacobian, in this case, of the model equation $B(a)$.

Considering the model runs in time with a sequence of operations $B(a) = B_n(B_{n-1}(\dots B_1(B_0(a))\dots))$, the chain rule allows the perturbation in b after n steps of the model run to be obtained (from the perturbation in a) with the following:

$$b'_j = b'^{(n)}_j; \quad b'^{(i)}_j \sum_k \frac{\partial b^{(i)}_j}{\partial b^{(i-1)}_k} b'^{(i-1)}_k = \left(\frac{\partial b^{(i)}_j}{\partial b^{(i-1)}}\right)^T b'^{(i-1)}; \quad b'^{(0)}_j = \sum_k \frac{\partial b^{(0)}_j}{\partial a_k} a'_k$$

Thus it is possible to compute

$$b'_j = \sum_k \left[\frac{\partial b^{(n)}_j}{\partial b^{(n-1)}_k} \sum_l \left(\frac{\partial b^{(n-1)}_k}{\partial b^{(n-2)}_l} \cdots \sum_k \left(\frac{\partial b^{(0)}_l}{\partial a_m} a'_m\right) \cdots\right)\right].$$

This allows the sufficient condition that the model only needs to be differentiable along the trajectory from a through $b^{(n)}$ (Errico, 1997).

Using the chain rule once more, then the desired relation between the cost function and input perturbations is:

$$\frac{\partial \mathbf{J}}{\partial a_j} = \sum_k \frac{\partial b_k}{\partial a_j} \frac{\partial \mathbf{J}}{\partial b_k} \tag{2.27}$$

Note that in this case $\frac{\partial b_k}{\partial a_j}$ is the transpose of the Jacobian (i.e. the model's adjoint operator). This function will map the sensitivity of the cost function with respect to system output backward in time to the input.

The advantage of the adjoint method arises when the original cost function is augmented into its associated Lagrangian through the introduction of Lagrange multipliers. These multipliers exist in the dual of the model domain and are termed adjoint variables. They are independent of the model parameters. The introduction of these Lagrange multipliers simplify the problem of finding stationary points of the gradient in the cost function abiding to model constraints to an unconstrained problem. The general procedure is explained in Plessix (2006). Each adjoint variable carries with it a global measure of the perturbation of the problem with respect to the state variables. In the case of least square, these variables are a measure of the misfits between the model and the observed truth. These are propagated backward through time with the use of the model adjoint operator, seen in (2.27), which corresponds to the transpose of the Jacobian. The back-propagation of this information is further simplified by the repeated use of adjoint operators as a result of the application of the chain rule on the linearized system. Reverting back to the previous notation and in minimizing the least squares cost function (2.28), the equations are summarized by Robinson et al. (1998)

$$\begin{aligned} \mathbf{J}_N &= \frac{1}{2}(\hat{\mathbf{x}}_0(+) - \hat{\mathbf{x}}_0(-))^T \mathbf{P}_0^{-1}(\hat{\mathbf{x}}_0(+) - \hat{\mathbf{x}}_0(-)) \\ &\quad + \sum_{k=1}^{N} \frac{1}{2}(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}(+)_k)^T \mathbf{R}_k^{-1}(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}(+)_k) \end{aligned} \tag{2.28}$$

The augmented Lagrangian form becomes

$$\begin{aligned} \mathbf{J}_N &= \frac{1}{2}(\hat{\mathbf{x}}_0(+) - \hat{\mathbf{x}}_0(-))^T \mathbf{P}_0^{-1}(\hat{\mathbf{x}}_0(+) - \hat{\mathbf{x}}_0(-)) \\ &\quad + \sum_{k=1}^{N} \frac{1}{2}(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}(+)_k)^T \mathbf{R}_k^{-1}(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}(+)_k) \\ &\quad + \sum_{k=1}^{N} \lambda_{k-1}^T(\hat{\mathbf{x}}(+)_k - \mathbf{\Phi}_{k-1}\hat{\mathbf{x}}(-)_{k-1}) \end{aligned} \tag{2.29}$$

where λ are the adjoint variables or Lagrange multipliers.

The state update follows from (2.30), the final Lagrange multiplier is taken to have no model data misfit (where no measurement is taken), and is back-propagated to λ_0, which is proportional to the gradient in $\mathbf{J}$ with respect to $\hat{\mathbf{x}}_0(+)$. Then (2.33) may be used to iteratively alter the initial guess $\hat{\mathbf{x}}_0(-)$ by assigning it the new value $\hat{\mathbf{x}}_0(+)$.

$$\hat{\mathbf{x}}(-)_k = \mathbf{\Phi}_{k-1}\hat{\mathbf{x}}(-)_{k-1} \tag{2.30}$$

$$\lambda_N = 0 \tag{2.31}$$

$$\lambda_{k-1} = \mathbf{\Phi}_{k-1}^T\lambda_k + \mathbf{H}_k^T\mathbf{R}_k^{-1}(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_k(-)) \tag{2.32}$$

$$\hat{\mathbf{x}}_0(+) = \hat{\mathbf{x}}_0(-) + \mathbf{P}_0\mathbf{\Phi}_0^T\lambda_0 \tag{2.33}$$

where data is assumed to be collected from time t_1 to t_N.

The adjoint model has vast application as adjoint operators can be derived for any equations having first derivatives (i.e. any model linearizable by first order Taylor expansion), but specific attention needs to be given to possible inaccuracies. Tangent linear and adjoint models make linearizations for use with infinitesimal inputs. The accuracy of tangent linear models and adjoint models depend on the approximations made in linearizing the dynamics of the problem as well as the size of the perturbation or step utilized. If the model is driven by highly nonlinear equations, large perturbations used at locations where the nonlinearity of the problem becomes significant (around critical points) will generate erroneous sensitivity results or output perturbations. How "large" the perturbations may be prior to the failure of the linearized model will depend on the particular application (Errico, 1997). If the control term in this method is set to the initial estimate, the TLM used in this parameter estimation scheme is comparable to the propagation matrix utilized by the linearized Kalman filter.

2.4 Ensemble-Bayesian Methods

From the previous sections regarding the Kalman filter, it was established that the optimal linear combination of the measurement and forecast is given by (2.7) with the gain, $\mathbf{K}_k$ defined by (2.10). Whereas the EKF utilizes a linearization of the system dynamics, the Ensemble Kalman Filter (EnKF) utilizes, as the name implies, ensemble statistics to obtain the covariance matrices involved in these equations. Computational cost is reduced for sparsely measured systems by limiting the calculation of the covariance matrix only to observed portions of the state. In this fashion, the product $\mathbf{H}_k\mathbf{P}_k(-)\mathbf{H}_k^T$ is treated as the expectation of the *a priori* state estimate mapped to the observation space multiplied with its transpose, i.e. $\mathbf{E}[\mathbf{y}_k(-)\mathbf{y}_k^T(-)]$ (where $\mathbf{y}_k(-) = \mathbf{H}_k\mathbf{x}_k(-)$ is the model output mapped onto the observation domain), and $\mathbf{P}_k(-)\mathbf{H}_k^T = \mathbf{E}[\mathbf{x}_k(-)\mathbf{y}_k^T(-)]$. The first real-time ensemble data assimilation done at seas was in the Strait of Sicily in 1996 utilizing an Error Subspace Statistical Estimation (ESSE) method that will be presented shortly (Lermusiaux, 1999).

Evensen (1994) applied this ensemble technique to a quasi-geostrophic ocean model, thus showing promising results in an alternative to the EKF with no closure problems in forecast error statistics, and in his particular study, to the benefit of a reduced computation cost. Houtekamer and Mitchell (1998) describe this filtering method and identify the ensemble statistic equations utilized in scheme presented by Evensen (1994). First, an ensemble is generated using an initial central estimate of the state to which a random field satisfying prior covariance conditions is added. Then, for each assimilation period, an ensemble is created about the available observation based on the best current representation of the observation error covariance. A twin experiment for the evaluation of this technique is examined by Houtekamer and Mitchell (1998). The central value of the first estimate is chosen based on the actual state used in the model dynamics, to which a realization of the desired noise characteristics is added. Observations are created from the simulated truth using an observation matrix $\mathbf{H}$ and observation noise. The computation of the error covariance

consists of

$$\begin{aligned}\mathbf{P}(-)\mathbf{H}^T &= \frac{1}{N-1}\sum_{i=1}^{N}(\hat{\mathbf{x}}_i(-)-\hat{\mathbf{x}}(-))[\mathbf{H}(\hat{\mathbf{x}}_i(-)-\hat{\mathbf{x}}(-))]^T \\ \mathbf{H}\mathbf{P}(-)\mathbf{H}^T &= \frac{1}{N-1}\sum_{i=1}^{N}\mathbf{H}(\hat{\mathbf{x}}_i(-)-\hat{\mathbf{x}}(-))[\mathbf{H}(\hat{\mathbf{x}}_i(-)-\hat{\mathbf{x}}(-))]^T\end{aligned} \tag{2.34}$$

and mean

$$\hat{\mathbf{x}}(-) = \frac{1}{N}\sum_{i=1}^{N}\hat{\mathbf{x}}_i(-). \tag{2.35}$$

The rank of the above covariance matrices is less than or equal the size of the ensemble used. By considering ensemble size larger than the number of observations, the rank deficiency problem is avoided (or alleviated for the case of small ensemble sizes) resulting in full rank covariance matrices.

Anderson describes an Ensemble Adjustment Kalman Filter (EAKF) for Data Assimilation, an apparent improvement to the traditional EnKF. The Ensemble Kalman Filter is typically derived from the Extended Kalman Filter equations, which hides the versatility of the EnKF in its ability to handle arbitrary probability distributions, which are non-Gaussian. The goal of the EAKF is to reduce the noise incorporated into the prior ensemble as a result of assimilating observations with distant correlation (in time and space). To alleviate this problem, the EAKF generates a set ensemble matching the state observation noise, which it utilizes for the remainder of the recursive inverse method. As such, the EAKF, unlike the EnKF, requires no generation of random vectors after initialization and becomes a deterministic filtering scheme from the start (Anderson, 2001). A means by which the cost of such recursive methods may be reduced is through the truncation of the structure present in the error covariance matrix. However, by using the method presented in the EAKF, this would fix the error statistics in time. It would be advantageous to diminish the complexity of this matrix without enforcing stationarity.

As opposed to using an ensemble set to determine the full covariance matrix, Lermusiaux 1999a, 1999b suggests reducing the analysis to a subspace of the error covariance in his Error Subspace Statistical Estimation (ESSE) scheme. In this

method, the first step is to obtain the dominant structures of the error covariance. These are obtained through the orthonormal decomposition of the matrix in question. Then the prominent vector, those corresponding to the largest singular values of the decomposition are multiplied to form a matrix of equivalent dimension as the original, but lacking in the less pronounced structures. Since the covariance matrix is by nature positive semi-definite, such a decomposition is equivalent to the eigenvalue decomposition. By limiting attention to the dominant errors, the computational cost can then be accordingly focused and lessened to capturing this reduced space. In the ESSE scheme, the melding criterion used consists of the linear Kalman update. That is, the state estimate update is as in (2.21), uncertainty update as in (2.26), and Kalman gain as in (2.25). These equations are then slightly altered in appearance by introducing the eigen-decomposed error covariance matrices (2.36)

$$\begin{aligned} \mathbf{P}(-) &= \mathbf{E}_{-}\mathbf{\Lambda}(-)\mathbf{E}_{-}^{T} \\ \mathbf{P}(+) &= \mathbf{E}_{+}\mathbf{\Lambda}(+)\mathbf{E}_{+}^{T} \end{aligned} \tag{2.36}$$

Where the subscripted $\mathbf{E}$ matrices are orthonormal matrices of eigenvectors because the uncertainty matrices from which they are derived, $\mathbf{P}$, are symmetric. Introducing these definitions into (2.10) and (2.11)

$$\begin{aligned} \mathbf{K} &= \mathbf{P}(-)\mathbf{H}^{T}[\mathbf{H}\mathbf{P}(-)\mathbf{H}^{T} + \mathbf{R}]^{-1} \\ \mathbf{K} &= \mathbf{E}_{-}\mathbf{\Lambda}(-)\mathbf{E}_{-}^{T}\mathbf{H}^{T}[\mathbf{H}\mathbf{E}_{-}\mathbf{\Lambda}(-)\mathbf{E}_{-}^{T}\mathbf{H}^{T} + \mathbf{R}]^{-1} \\ \mathbf{K} &= \mathbf{E}_{-}\mathbf{\Lambda}(-)\tilde{\mathbf{H}}^{T}[\tilde{\mathbf{H}}\mathbf{\Lambda}(-)\tilde{\mathbf{H}}^{T} + \mathbf{R}]^{-1} \end{aligned} \tag{2.37}$$

where the k subscripts have been omitted and the substitution of the definition $\tilde{\mathbf{H}} \equiv$

$\mathbf{HE}_-$ has been made.

$$\begin{aligned}
\mathbf{P}(+) &= \mathbf{P}(-) - \mathbf{P}(-)\mathbf{H}^T[\mathbf{HP}(-)\mathbf{H}^T + \mathbf{R}]^{-1}\mathbf{HP}(-) \\
\mathbf{E}_+\mathbf{\Lambda}(+)\mathbf{E}_+^T &= \mathbf{E}_-\mathbf{\Lambda}(-)\mathbf{E}_-^T - \mathbf{E}_-\mathbf{\Lambda}(-)\mathbf{E}_-^T\mathbf{H}^T[\mathbf{HE}_-\mathbf{\Lambda}(-)\mathbf{E}_-^T\mathbf{H}^T + \mathbf{R}]^{-1} \\
&\quad \times\mathbf{HE}_-\mathbf{\Lambda}(-)\mathbf{E}_-^T \\
\mathbf{E}_+\mathbf{\Lambda}(+)\mathbf{E}_+^T &= \mathbf{E}_-\mathbf{\Lambda}(-)\mathbf{E}_-^T - \mathbf{E}_-\mathbf{\Lambda}(-)\tilde{\mathbf{H}}^T[\tilde{\mathbf{H}}\mathbf{\Lambda}(-)\tilde{\mathbf{H}}^T + \mathbf{R}]^{-1}\tilde{\mathbf{H}}\mathbf{\Lambda}(-)\mathbf{E}_-^T \\
\mathbf{E}_+\mathbf{\Lambda}(+)\mathbf{E}_+^T &= \mathbf{E}_-\{\mathbf{\Lambda}(-) - \mathbf{\Lambda}(-)\tilde{\mathbf{H}}^T[\tilde{\mathbf{H}}\mathbf{\Lambda}(-)\tilde{\mathbf{H}}^T + \mathbf{R}]^{-1}\tilde{\mathbf{H}}\mathbf{\Lambda}(-)\}\mathbf{E}_-^T \\
\mathbf{E}_+\mathbf{\Lambda}(+)\mathbf{E}_+^T &\equiv \mathbf{E}_-\tilde{\mathbf{\Lambda}}(+)\mathbf{E}_-^T
\end{aligned} \tag{2.38}$$

with $\tilde{\mathbf{\Lambda}}(+) = \mathbf{\Lambda}(-) - \mathbf{\Lambda}(-)\tilde{\mathbf{H}}^T[\tilde{\mathbf{H}}\mathbf{\Lambda}(-)\tilde{\mathbf{H}}^T + \mathbf{R}]^{-1}\tilde{\mathbf{H}}\mathbf{\Lambda}(-)$ The eigen-decomposition of $\tilde{\mathbf{\Lambda}}(+)$ yields the same eigenvalues of $\mathbf{\Lambda}(+)$ for $\mathbf{P}(+)$ as in (2.39)

$$\tilde{\mathbf{\Lambda}}(+) = \mathbf{T}\mathbf{\Lambda}(+)\mathbf{T}^T \tag{2.39}$$

where the matrix $\mathbf{T}$ consists of orthonormal column vectors and transforms the eigenvectors of $\mathbf{E}_-$ into $\mathbf{E}_+$. Thus far, the equations presented only consist of a rewritten form with the eigen-decomposition of the actual covariance matrices. The eigen-decomposition of the sample covariances will be identified by the eigenvector and eigenvalue matrices $\mathbf{U}_-$ and $\mathbf{\Pi}(-)$ *a priori* and $\mathbf{U}_+$ and $\mathbf{\Pi}(+)$ *a posteriori*. Additionally, in the ESSE approach, the rank of the covariance is truncated to the dominant components. The reduced space is identified by the size of its rank in a superscript. The reduced rank is identified by p. An ensemble of size q unbiased state estimates is denoted by $\hat{\mathbf{x}}^i(-)$ *a priori*. The corresponding errors of these samples form a matrix $\mathbf{M}(-)$ of q state column vectors less their mean estimate. Here $\mathbf{d}^i$ $(i = 1, \ldots, q)$ denotes an ensemble of size q of observations perturbed by noise of covariance $\mathbf{R}$. The sample error covariance denoted by $\mathbf{P}^s$ is obtained by $\mathbf{P}^s = \mathbf{M}\mathbf{M}^T/q$. From (2.7) the update equation for the ensembles and the ensemble mean can be written.

$$\begin{aligned}
\hat{\mathbf{x}}^i(+) &= \hat{\mathbf{x}}^i(-) + \mathbf{K}^s[\mathbf{d}^i - \mathbf{H}\hat{\mathbf{x}}^i(-)] \\
\hat{\mathbf{x}}(+) &= \hat{\mathbf{x}}(-) + \mathbf{K}^s[\mathbf{d} - \mathbf{H}\hat{\mathbf{x}}(-)]
\end{aligned} \tag{2.40}$$

Through subtraction, the above becomes

$$\mathbf{M}(+) = (\mathbf{I} - \mathbf{K}^s\mathbf{H})\mathbf{M}(-) + \mathbf{K}^s\mathbf{D} \tag{2.41}$$

where $\mathbf{D} = [\mathbf{v}^j] = [\mathbf{d}^j - \mathbf{d}]$. Multiplying this equation through by its transpose and taking the expectation, or rather dividing through by the ensemble size q

$$\begin{aligned}
\mathbf{P}^s(+) &= (\mathbf{I} - \mathbf{K}^s\mathbf{H})\mathbf{P}^s(-)(\mathbf{I} - \mathbf{K}^s\mathbf{H})^T + \mathbf{K}^s\mathbf{R}^s\mathbf{K}^{sT} \\
&\quad +(\mathbf{I} - \mathbf{K}^s\mathbf{H})\mathbf{\Omega}^s\mathbf{K}^{sT} + \mathbf{K}^s\mathbf{\Omega}^{sT}(\mathbf{I} - \mathbf{K}^s\mathbf{H})^T \\
\mathbf{P}^s(+) &= \mathbf{P}^s(-) - \mathbf{K}^s\mathbf{H}\mathbf{P}^s(-) - \mathbf{P}^s(-)\mathbf{H}^T\mathbf{K}^{sT} + \mathbf{K}^s\mathbf{H}\mathbf{P}^s(-)\mathbf{H}^T\mathbf{K}^{sT} \\
&\quad +\mathbf{K}^s\mathbf{R}^s\mathbf{K}^{sT} + \mathbf{\Omega}^s\mathbf{K}^{sT} - \mathbf{K}^s\mathbf{H}\mathbf{\Omega}^s\mathbf{K}^{sT} + \mathbf{K}^s\mathbf{\Omega}^{sT} - \mathbf{K}^s\mathbf{\Omega}^{sT}\mathbf{H}^T\mathbf{K}^{sT}
\end{aligned} \tag{2.42}$$

where the definition of $\mathbf{P}^s$ has been used and $\mathbf{R}^s \equiv \mathbf{D}\mathbf{D}^T/q$ and $\mathbf{\Omega}^s \equiv \mathbf{M}(-)\mathbf{D}^T/q$ have been introduced. (In the limit of an infinite ensemble size, the matrices superscripted with s tend toward the actual covariance and cross-covariances). For the gain $\mathbf{K}^s$, which minimizes the trace of this expression, taking the derivative with respect to $\mathbf{K}^s$ (as with (2.22) in the EKF section) and setting to zero gives:

$$\begin{aligned}
0 &= 0 - \mathbf{P}^{sT}(-)\mathbf{H}^T - \mathbf{P}^s(-)\mathbf{H}^T + \mathbf{K}^s\mathbf{H}(\mathbf{P}^s(-) + \mathbf{P}^{sT}(-))\mathbf{H}^T \\
&\quad +\mathbf{K}^s(\mathbf{R}^s + \mathbf{R}^{sT}) + \mathbf{\Omega}^s - \mathbf{K}^s\mathbf{H}\mathbf{\Omega}^s - \mathbf{K}^s\mathbf{\Omega}^{sT}\mathbf{H}^T + \mathbf{\Omega}^s \\
&\quad -\mathbf{K}^s\mathbf{\Omega}^{sT}\mathbf{H}^T - \mathbf{K}^s\mathbf{H}\mathbf{\Omega}^s \\
0 &= -2\mathbf{P}^s(-)\mathbf{H}^T + 2\mathbf{K}^s\mathbf{H}\mathbf{P}^s(-)\mathbf{H}^T + 2\mathbf{K}^s\mathbf{R}^s \\
&\quad +2\mathbf{\Omega}^s - 2\mathbf{K}^s\mathbf{H}\mathbf{\Omega}^s - 2\mathbf{K}^s\mathbf{\Omega}^{sT}\mathbf{H}^T \\
\mathbf{P}^s(-)\mathbf{H}^T - \mathbf{\Omega}^s &= \mathbf{K}^s\mathbf{H}\mathbf{P}^s(-)\mathbf{H}^T + \mathbf{K}^s\mathbf{R}^s - \mathbf{K}^s\mathbf{H}\mathbf{\Omega}^s - \mathbf{K}^s\mathbf{\Omega}^{sT}\mathbf{H}^T \\
\mathbf{K}^s &= (\mathbf{P}^s(-)\mathbf{H}^T - \mathbf{\Omega}^s)[\mathbf{H}\mathbf{P}^s(-)\mathbf{H}^T + \mathbf{R}^s - \mathbf{H}\mathbf{\Omega}^s - \mathbf{\Omega}^{sT}\mathbf{H}^T]^{-1}
\end{aligned} \tag{2.43}$$

With the assumption that the measurement noise is uncorrelated to the dynamic process, as the ensemble size tends toward infinity, $\mathbf{\Omega}^s$ tends toward zero, leading to the simplified equations.

$$\mathbf{P}^s(+) = \mathbf{P}^s(-) - \mathbf{K}^s\mathbf{H}\mathbf{P}^s(-) - \mathbf{P}^s(-)\mathbf{H}^T\mathbf{K}^{sT} + \mathbf{K}^s\mathbf{H}\mathbf{P}^s(-)\mathbf{H}^T\mathbf{K}^{sT} + \mathbf{K}^s\mathbf{R}^s\mathbf{K}^{sT} \tag{2.44}$$

$$\mathbf{K}^s = \mathbf{P}^s(-)\mathbf{H}^T[\mathbf{H}\mathbf{P}^s(-)\mathbf{H}^T + \mathbf{R}^s]^{-1} \tag{2.45}$$

And, as in the case of (2.9) and (2.10), reduce to

$$\mathbf{P}^s(+) = (\mathbf{I} - \mathbf{K}^s\mathbf{H})\mathbf{P}^s(-) \tag{2.46}$$

The error subspace is derived from the dominant rank-p reduction of the sample space involved in the above equations. Lermusiaux 1999a identifies the singular value decomposition (SVD) as an efficient way of determining this reduce error space. By selecting the left-hand side singular vectors of the corresponding p highest singular values of the decomposition to generate a field of simplified structure.

$$\begin{aligned} \mathrm{SVD}_p[\mathbf{M}(-)] &= \mathbf{U}_-\mathbf{\Sigma}(-)\mathbf{V}_-^T \\ \mathrm{SVD}_p[\mathbf{M}(+)] &= \mathbf{U}_+\mathbf{\Sigma}(+)\mathbf{V}_+^T \end{aligned} \tag{2.47}$$

It is then easily seen that by the definition of the sample covariance, that the left singular vectors form the orthonormal eigenvectors of the reduced error space, and the reduced space eigenvectors correspond to

$$\begin{aligned} \mathbf{\Pi}(-) &= \mathbf{\Sigma}^2(-)/q \\ \mathbf{\Pi}(+) &= \mathbf{\Sigma}^2(+)/q. \end{aligned} \tag{2.48}$$

It is then possible to carry out an update with the substitution of this reduced space into (2.40). Noting the change by altering the superscripts from s to p, the ensemble mean estimate is then.

$$\hat{\mathbf{x}}(+) = \hat{\mathbf{x}}(-) + \mathbf{K}^p[\mathbf{d} - \mathbf{H}\hat{\mathbf{x}}(-)] \tag{2.49}$$

with the Kalman gain from (2.45)

$$\begin{aligned}
\mathbf{K}^p &= \mathbf{P}^p(-)\mathbf{H}^T[\mathbf{H}\mathbf{P}^p(-)\mathbf{H}^T + \mathbf{R}^s]^{-1} \\
\mathbf{K}^p &= \mathbf{U}_-\mathbf{\Pi}(-)\mathbf{U}_-^T\mathbf{H}^T[\mathbf{H}\mathbf{U}_-\mathbf{\Pi}(-)\mathbf{U}_-^T\mathbf{H}^T + \mathbf{R}^s]^{-1} \\
\mathbf{K}^p &= \mathbf{U}_-\mathbf{\Pi}(-)\tilde{\mathbf{H}}^{pT}[\tilde{\mathbf{H}}^p\mathbf{\Pi}(-)\tilde{\mathbf{H}}^{pT} + \mathbf{R}^s]^{-1} \\
\mathbf{K}^p &\equiv \mathbf{U}_-\tilde{\mathbf{K}}^p
\end{aligned} \tag{2.50}$$

where it was maintained that the covariance in observation remains at its full structure (not reduced) and the substitution $\tilde{\mathbf{H}}^p \equiv \mathbf{H}\mathbf{U}_-$ was introduced. The update of the covariance can be carried out in two steps: updating the eigenvalues and eigenvectors separately. From (2.46)

$$\begin{aligned}
\mathbf{P}^p(+) &= (\mathbf{I}^p - \mathbf{K}^p\mathbf{H})\mathbf{P}^p(-) \\
\mathbf{U}_-^T\mathbf{P}^p(+)\mathbf{U}_- &= \mathbf{U}_-^T(\mathbf{I}^p - \mathbf{K}^p\mathbf{H})\mathbf{P}^p(-)\mathbf{U}_- \\
\tilde{\mathbf{\Pi}}(+) &= (\mathbf{U}_-^T - \mathbf{U}_-^T\mathbf{K}^p\mathbf{H})\mathbf{U}_-\mathbf{\Pi}(-)\mathbf{U}_-^T\mathbf{U}_- \\
\tilde{\mathbf{\Pi}}(+) &= (\mathbf{U}_-^T\mathbf{U}_- - \mathbf{U}_-^T\mathbf{U}_-\tilde{\mathbf{K}}^p\mathbf{H}\mathbf{U}_-)\mathbf{\Pi}(-) \\
\tilde{\mathbf{\Pi}}(+) &= (\mathbf{I}^p - \tilde{\mathbf{K}}^p\tilde{\mathbf{H}}^p)\mathbf{\Pi}(-) \\
&\text{where} \\
\mathbf{P}^p(+) &= \mathbf{U}_+\mathbf{\Pi}(+)\mathbf{U}_+^T \\
\mathbf{P}^p(+) &= \mathbf{U}_-\tilde{\mathbf{\Pi}}(+)\mathbf{U}_-^T \\
\mathbf{U}_+ &= \mathbf{U}_-\mathbf{T} \\
\tilde{\mathbf{\Pi}}(+) &= \mathbf{T}\mathbf{\Pi}(+)\mathbf{T}^T
\end{aligned} \tag{2.51}$$

where $\mathbf{T}$ is as in (2.39). Lermusiaux 1999a concludes "Scheme A" at the update of the ensemble mean and ensemble covariance. After this point a new ensemble set has to be created. Scheme A utilizes these new equations to update the error subspace and ensemble estimate through the eigenvalues and eigenvectors (obtaining the covariance, Kalman gain, and new ensemble estimate). It does not update the ensemble itself. As a result, either a new ensemble should be generated, leading to resampling with the newly obtained covariance matrix, or "Scheme B" should be utilized. Lermusiaux 1999a introduces this extension to the above method to calculate the right hand side

singular vectors ($\mathbf{V}$). These can then be used to carry out an update of each ensemble member without the need of using another Monte Carlo resampling step.

2.5 Unscented Kalman Filter

Central to the EKF method are the assumptions that the noise present in the system dynamics and observations are Gaussian random variables of small amplitude and that the physics can adequately be represented through linearization. The EKF relies on the analytical propagation of the random variable information through a set of simplified state equations. Whereas a Monte Carlo ensemble approach would seek to improve the *a posteriori* estimate by propagating a large number of values representative of the noise through the nonlinear dynamics, a class of Kalman filters termed Sigma-Point Kalman Filter (SPKF) choose the representative set deterministically, reducing the required ensemble size to a minimal set capturing the properties of the distribution. Julier and Uhlmann (1996) discuss this novel method of deterministic sampling to calculate the terms in (2.7). This method allows a linearization of the system that accounts for the actual uncertainty. Where, as opposed to linearizing the dynamics, a linearization of the true nonlinear statistics is made.

Thus, considering a function $\mathbf{y} = \mathbf{g}(\mathbf{x})$, where $\mathbf{y}$ is a vector random variable output of the nonlinear transformation (through $\mathbf{g}(\cdot)$) of the vector random variable input $\mathbf{x}$, Bayesian estimation methods are applied. A set of r points (χ_i, υ_i) are evaluated as $\upsilon_i = \mathbf{g}(\chi_i)$. The sigma points, χ_i, are a deterministically chosen ensemble of vectors representative of $\mathbf{x}$, and υ_i are their nonlinearly transformed counterparts, representative of $\mathbf{y}$. The χ_i sigma points are selected so as to satisfy the mean and covariance of $\mathbf{x}$ (2.52)

$$\begin{aligned} \bar{\mathbf{x}} &= \sum_{i=1}^{r} w_i \chi_i \\ \mathbf{P_{xx}} &= \sum_{i=1}^{r} w_i (\chi_i - \bar{\mathbf{x}})(\chi_i - \bar{\mathbf{x}})^T \end{aligned} \tag{2.52}$$

after which output statistics can be computed in a similar linear fashion

$$\begin{aligned} \bar{\mathbf{y}} &= \sum_{i=1}^{r} w_i \upsilon_i \\ \mathbf{P_{yy}} &= \sum_{i=1}^{r} w_i (\upsilon_i - \bar{\mathbf{y}})(\upsilon_i - \bar{\mathbf{y}})^T \\ \mathbf{P_{xy}} &= \sum_{i=1}^{r} w_i (\chi_i - \bar{\mathbf{x}})(\upsilon_i - \bar{\mathbf{y}})^T \end{aligned} \tag{2.53}$$

where in the above equations $\sum_{i=1}^{r} w_i = 1$. van der Merwe and Wan (2003) present a few different filtering schemes related through their use of *weighted statistical linear regression* to compute the propagated uncertainty statistics. The aim is to find the linear relation $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b}$ which minimizes a statistical cost function $\mathbf{J}$ usually taken to be $\mathbf{J} = \mathbf{E}[w_i(\mathbf{e}_i)^2]$, or equivalently $\mathbf{J} = \text{trace}\{\mathbf{E}[\mathbf{e}_i \ \text{diag}(w_i) \ \mathbf{e}_i^T]\}$ (where $\mathbf{e}_i = \upsilon_i - (\mathbf{A}\chi_i + \mathbf{b})$). The matrix $\mathbf{A}$ is the UKF equivalent of the Kalman gain used in (2.7) and (2.9). By construct $\bar{\mathbf{y}} - \mathbf{A}\bar{\mathbf{x}} - \mathbf{b} = 0$.

$$\begin{aligned} \bar{\mathbf{e}} &= \sum_{i=1}^{r} w_i [\upsilon_i - (\mathbf{A}\chi_i + \mathbf{b})] \\ &= \sum_{i=1}^{r} w_i \upsilon_i - \sum_{i=1}^{r} w_i \mathbf{A}\chi_i - \sum_{i=1}^{r} w_i \mathbf{b} \\ &= \bar{\mathbf{y}} - \mathbf{A}\bar{\mathbf{x}} - \mathbf{b} \\ &= 0 \end{aligned}$$

As in the Kalman filter, the minimum of the trace in the *a posteriori* error covariance

is sought.

$$
\begin{aligned}
\mathbf{P_{ee}} &= \sum_{i=1}^{r} (\mathbf{e}_i - \bar{\mathbf{e}}_i) w_i (\mathbf{e}_i - \bar{\mathbf{e}}_i)^T \\
&= \sum_{i=1}^{r} [\upsilon_i - (\mathbf{A}\chi_i + \mathbf{b}) - (\bar{\mathbf{y}} - \mathbf{A}\bar{\mathbf{x}} - \mathbf{b})] w_i [\upsilon_i - (\mathbf{A}\chi_i + \mathbf{b}) - (\bar{\mathbf{y}} - \mathbf{A}\bar{\mathbf{x}} - \mathbf{b})]^T \\
&= \sum_{i=1}^{r} [(\upsilon_i - \bar{\mathbf{y}}) - (\mathbf{A}\chi_i - \mathbf{A}\bar{\mathbf{x}}) - (\mathbf{b} - \mathbf{b})] w_i [(\upsilon_i - \bar{\mathbf{y}}) - (\mathbf{A}\chi_i - \mathbf{A}\bar{\mathbf{x}}) - (\mathbf{b} - \mathbf{b})]^T \\
&= \sum_{i=1}^{r} [(\upsilon_i - \bar{\mathbf{y}}) - (\mathbf{A}\chi_i - \mathbf{A}\bar{\mathbf{x}})] w_i [(\upsilon_i - \bar{\mathbf{y}}) - (\mathbf{A}\chi_i - \mathbf{A}\bar{\mathbf{x}})]^T \\
&= \sum_{i=1}^{r} (\upsilon_i - \bar{\mathbf{y}}) w_i (\upsilon_i - \bar{\mathbf{y}})^T - (\upsilon_i - \bar{\mathbf{y}}) w_i (\chi_i - \bar{\mathbf{x}})^T \mathbf{A}^T \\
&\quad -\mathbf{A}(\chi_i - \bar{\mathbf{x}}) w_i (\upsilon_i - \bar{\mathbf{y}})^T + \mathbf{A}(\chi_i - \bar{\mathbf{x}}) w_i (\chi_i - \bar{\mathbf{x}})^T \mathbf{A}^T \\
&= \mathbf{P_{yy}} - \mathbf{P_{yx}}\mathbf{A}^T - \mathbf{A}\mathbf{P_{xy}} + \mathbf{A}\mathbf{P_{xx}}\mathbf{A}^T
\end{aligned}
$$

differentiating the trace of $\mathbf{P_{ee}}$ with respect to $\mathbf{A}$ and equating to zero yields the expression

$$
\begin{aligned}
0 &= -\mathbf{P_{yx}} - \mathbf{P_{xy}}^T + \mathbf{A}(\mathbf{P_{xx}} + \mathbf{P_{xx}}^T) \\
0 &= -2\mathbf{P_{yx}} + 2\mathbf{A}\mathbf{P_{xx}} \\
\mathbf{A} &= \mathbf{P_{yx}}\mathbf{P_{xx}}^{-1} \\
\mathbf{K_{UKF}} &= \mathbf{A}
\end{aligned}
\tag{2.54}
$$

where $\mathbf{K_{UKF}}$ can then be substituted for $\mathbf{K}$ in the Kalman filter equations, previously mentioned in Section 2.1.3. The difference across the various SPKFs lies in the weight assigned to each sigma-point and in the number of these created. van der Merwe and Wan (2003) describe the Unscented Kalman Filter (UKF), Central Difference Kalman Filter (CDKF) and their square root (SR) implementations. The UKF and CDKF are similar in that the number of sigma points are identical, additionally, the original UKF has the same weights as the CDKF when the parameters involved in computing these values are optimized for Gaussian priors. The Unscented Transformation is based on the intuition that approximating a Gaussian distribution based on a fixed number of parameters should be easier than making an approximation of an arbitrary nonlinear function (Julier and Uhlmann, 1996). And where an arbitrary sampling of points from a distribution might create spurious modes, a finite deterministic set can be created to captured the desired properties of the distribution in question. With

this assumption, a set of $2n$ points (where n is the size of the stochastic vector x) is created from the signed n columns of the matrix square root, $\mathbf{A}$, of the covariance, $n\mathbf{P}$ (where $\mathbf{A} = \sqrt{n\mathbf{P}}$ and $n\mathbf{P} = \mathbf{A}^T\mathbf{A}$). That is, the set is made symmetric about zero. To these values, the mean $\bar{\mathbf{x}}$ is added, thus capturing the first two moments of $\mathbf{x}$ deterministically with $2n$ sigma points (χ_i, $i = 1, \ldots, 2n$). To arrive to this formulation, definitions of ensemble statistics were used. From (2.34), when the mean is estimated by (2.35), the factor in the denominator is $N - 1$. If the true mean is used, this value is replaced by N. In computing the above mentioned set, the unique vectors created form n instances, reversing the sign, then the remaining $n + 1$ to $2n$ vectors are obtained. From the resulting set, and with the inverse procedure, the same equation can be applied to recompute the covariance, where N now becomes $2n$ as opposed to n. This value is simply a sum of the weights assigned to each instance of the random variable. As a result, it is possible to scale each deterministic occurrence by a particular weight to fine tune the properties of the sample set. In particular, an extra point, which is equivalent to the mean, can be added to adjust the higher order moments of the created sample distribution. In the case of a Gaussian distribution, a weight assigned to this central sigma point ($\chi_0 = \bar{\mathbf{x}}$) of $\kappa = 3 - n$ will resolve the fourth order moment of the distribution, the kurtosis ($\mathbf{E}[x^4] = 3$). As a result, the suggested sigma points are (van der Merwe and Wan, 2004)

$$\begin{aligned} \chi_0 &= \bar{\mathbf{x}} \\ \chi_i &= \bar{\mathbf{x}} + \left(\sqrt{(n+\kappa)\mathbf{P}_{\mathbf{xx}}}\right)_i \qquad i = 1, \ldots, n \\ \chi_i &= \bar{\mathbf{x}} - \left(\sqrt{(n+\kappa)\mathbf{P}_{\mathbf{xx}}}\right)_{i-n} \quad i = n+1, \ldots, 2n \end{aligned} \tag{2.55}$$

with weights

$$\begin{aligned} w_0 &= \frac{\kappa}{n+\kappa} \\ w_i &= \frac{1}{2(n+\kappa)} \quad i = 1, \ldots, 2n \end{aligned} \tag{2.56}$$

The overall method then consists of deterministically generating an ensemble set based on (2.55) with weights (2.56). Next, these input states are run through the nonlinear system to obtain the propagated state. Using (2.53) the *a priori* error

covariance and cross-covariance are computed ($\mathbf{y}$ in this case being representative of $\mathbf{h}_k(\mathbf{x}_k(-)$, and $\mathbf{x}$ of $\mathbf{x}_k(-)$). The Kalman gain, $\mathbf{K_{UKF}}$ is then obtained from (2.54). With all of these parameters then, the state and covariance updates are computed by applying the Kalman update equations.

$$\bar{\mathbf{x}}_k(+) = \bar{\mathbf{x}}_k(-) + \mathbf{K_{UKF}}_k(\mathbf{z}_k - \bar{\mathbf{y}}_k) \tag{2.57}$$

$$\mathbf{P_{xx}}(+) = \mathbf{P_{xx}}(-) - \mathbf{K_{UKF}}\mathbf{P_{yy}}\mathbf{K_{UKF}}^T \tag{2.58}$$

The performance of methods identified in this chapter is evaluated with simple test problems in the following section.

Chapter 3

Idealized One-Dimensional Diffusion Studies

As mentioned earlier, we are motivated by uncertainties in ocean models that arise due to vertical mixing uncertainties. As a result, one of our future interests is to compare and possibly improve existing parameterizations of vertical mixing. As a first simple step, we aim to implement and compare the methods described in Chapter 2 for the estimation of mixing coefficients in one-dimensional diffusion problems. This allows a simpler and relatively more rapid comparison of existing and developing parameter estimation methods for adaptive modeling. Reduced-dimensional ocean models are not unusual and are a useful tool in learning about/exploring new findings in aspects of numeric model representation and in physical phenomena. As an example, when Mellor (2001) tested his turbulence closure scheme on a one-dimensional ocean simulation, results suggested the parameterization was incorrect. However, the model has been tested with laboratory experiments and it was determined that in fact, the omission of the horizontal divergence of the tracer at the surface was the cause for the erroneous outcome, and a source/sink term proved mandatory for agreement.

3.1 Formulation of Test Problem

The first step to the adaptive modeling (parameterization estimation) problem is to optimize each parameterization through the estimation of the tunable parameters using sampled data. The model of interest here is the means by which diffusion in the ocean is parameterized in the vertical direction (across isopycnals). Current parameterizations consist of mixing models which are often analogous in form to the process of molecular diffusion, but with variable coefficients. They include Pacanowski and Philander (Pacanowski and Philander, 1981), Mellor and Yamada (Mellor, 2001, and references therein), K-Profile Parameterization (Li et al., 2000), and Niiler and Kraus. The original HOPS code can implement either the Pacanowski and Philander scheme, or the mixing model specified by Niiler and Kraus (1977) (Haley, Personal communication). For more information regarding the HOPS ocean model refer to Lermusiaux (2001). The above parameterizations assume a locally defined diffusivity constant. The difficulty in implementing the above in a one-dimensional problem lies in the dependence of diffusivity and viscosity constants on the local Richardson number that is obtained from the flow field and the stratification frequency. Therefore, in order to use the above mentioned models, an estimate of a representative Brunt-Väisälä frequency and current velocity profile has to be made. To begin, consider a diffusivity vector $\vec{\kappa}$ dependent on a parameter vector $\vec{\theta}$, for which various parameterization will later be derived.

The dynamics of interest for this one-dimensional case are then represented by the following equation, (3.1).

$$\frac{\partial C}{\partial t} = \frac{\partial}{\partial z}\left(\kappa \frac{\partial C}{\partial z}\right) \tag{3.1}$$

The tracer concentration is represented by the variable C which is a function of space and time. It will later be discretized to a vector in space evolving in time. Depth is represented by z, and κ is the diffusivity parameter, which, analogous to molecular diffusions, governs the rate of spread for the tracer and is also a function of space. The boundary conditions for this problem are homogeneous Neumann, thus no flux,

and as can be seen by the equation, there is no advection either. As such, the problem is conservative, in that no tracer is lost, since there are no source or sink terms either (at least at this stage). In order to have a time varying diffusivity κ, its evolution equation must be defined $\frac{\partial \kappa}{\partial t}$ which may be deterministically or stochastically driven, or both.

As discussed in the course Numerical Fluid Mechanics for Engineers (MIT course ID 2.29), a conservative scheme can be used in solving the dynamics numerically due to the nature of the problem. Therefore, the discretized formulation of the problem will be in terms of nested derivatives as the product $\kappa \frac{\partial C}{\partial z}$ will be conserved. Before proceeding with these simplifications, non-dimensional scales will first be computed for ease of implementation with models of various units and scales. As a result, the non-dimensional variables are defined as in (3.2).

$$\begin{aligned} C^* &= \frac{C}{\max\{C(t=0)\}} \\ \kappa^* &= \frac{\kappa \tilde{T}}{Z^2} \\ z^* &= [0,1] \\ t^* &= [0,1] \end{aligned} \tag{3.2}$$

where T is the total run time, and Z is the maximum depth. An adequate value for κ^* must then be chosen so that it will be representative of the vertical diffusivity in the real ocean. This value should be at least of the same order of magnitude. Additionally, the rate at which data is assimilated will need to be set non-dimensionally, as will the number of depth samples for the tracer.

In posing the parameter estimation problem, the original system is here expanded to include, as variables, the parameters of interest (Gelb et al., 1974). It is in this process that a system that is often originally linear becomes nonlinear through the interactions of the parameters with the variables in the system equations. This is where the difficulty in obtaining the optimal solution arises and the need for advanced techniques (advanced filtering techniques for the recursive process) becomes apparent. Inverse problems have been deemed as such for their efforts in obtaining information

about a system's original state given its output state. The process by which initial conditions and parameters are estimated is then a form of such a problem. In order to make an estimate of this type by only observing a part of the system in time, an underlying understanding of the driving dynamics is important. Therefore, it is necessary to have (at least assumed) a forward model.

3.1.1 Numerical Test Case Specifics

As a first test for the study, an idealized smooth diffusivity profile that exemplifies some of the expected features in an ocean medium will be used. For this, a profile in the shape of a hyperbolic tangent ($\vec{\kappa} = a_3 - a_2 \tanh(2\pi(z^* - a_1))$) has been chosen and is represented in Fig. 3-1 along with the tunable parameters, $a_1^t = 0.25$, $a_2^t = 0.01$, $a_3^t = 0.03$ (where the superscripted t identifies the "truth"). As the linearization of

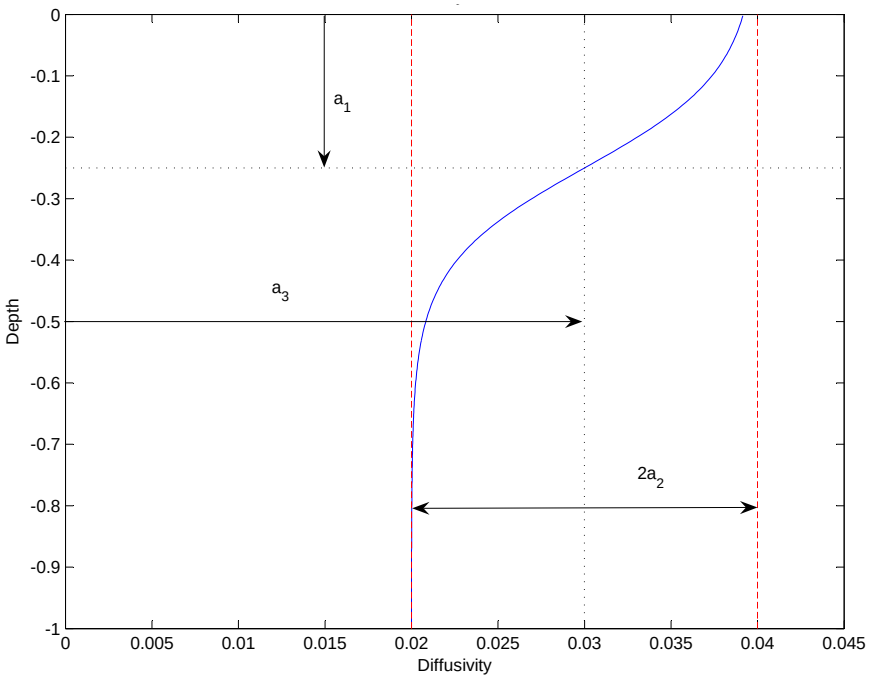

Figure 3-1: Abstract diffusivity profile
Hyperbolic tangent defined by three parameters ($a_1 = 0.25$, $a_2 = 0.01$, and $a_3 = 0.03$)

(3.1) will be necessary at least for two methods presented in Chapter 2, a numerical form of this formula will be derived. The original dynamics are therefore linearized in the following manner based on conservative discretization using the subscript i to

mark the vector index (3.3)

$$\begin{aligned}
\left(\frac{\partial C^*}{\partial t^*}\right)_i &= \left(\frac{\partial}{\partial z^*}\left(\kappa^*\frac{\partial C^*}{\partial z^*}\right)\right)_i \\
&= \left(\frac{\partial}{\partial z^*}\left(\frac{\kappa^*_{i+1}+\kappa^*_i}{2}\frac{C^*_{i+1}-C^*_i}{dz^*}\right)\right)_i \\
&= \frac{1}{dz^*}\left(\frac{\kappa^*_{i+1}+\kappa^*_i}{2}\frac{C^*_{i+1}-C^*_i}{dz^*} - \frac{\kappa^*_i+\kappa^*_{i-1}}{2}\frac{C^*_i-C^*_{i-1}}{dz^*}\right) \\
&= \frac{(\kappa^*_{i+1}+\kappa^*_i)C^*_{i+1} - (\kappa^*_{i+1}+2\kappa^*_i+\kappa^*_{i-1})C^*_i + (\kappa^*_i+\kappa^*_{i-1})C^*_{i-1}}{2dz^{*2}} \\
&= \frac{(\kappa^*_{i-1}+\kappa^*_i)C^*_{i-1} - (\kappa^*_{i-1}+2\kappa^*_i+\kappa^*_{i+1})C^*_i + (\kappa^*_i+\kappa^*_{i+1})C^*_{i+1}}{2dz^{*2}}
\end{aligned} \tag{3.3}$$

where in the above case a forward difference was utilized to compute the slope midway between points i and $i+1$ followed by a backward differencing scheme to calculate the value of the derivative between $i-\frac{1}{2}$ and $i+\frac{1}{2}$, i.e. at i. The result is a central difference scheme that is second order accurate. In order to reduce the computation expense, a staggered grid is utilized in defining the diffusivity vector. The descritized equation then becomes

$$\begin{aligned}
\left(\frac{\partial C^*}{\partial t^*}\right)_i &= \left(\frac{\partial}{\partial z^*}\left(\kappa^*\frac{\partial C^*}{\partial z^*}\right)\right)_i \\
&= \left(\frac{\partial}{\partial z^*}\left(\kappa^*_{i+\frac{1}{2}}\frac{C^*_{i+1}-C^*_i}{dz^*}\right)\right)_i \\
&= \frac{1}{dz^*}\left(\kappa^*_{i+\frac{1}{2}}\frac{C^*_{i+1}-C^*_i}{dz^*} - \kappa^*_{i-\frac{1}{2}}\frac{C^*_i-C^*_{i-1}}{dz^*}\right) \\
&= \frac{\kappa^*_{i+\frac{1}{2}}C^*_{i+1} - (\kappa^*_{i+\frac{1}{2}}+\kappa^*_{i-\frac{1}{2}})C^*_i + \kappa^*_{i-\frac{1}{2}}C^*_{i-1}}{dz^{*2}} \\
&= \frac{\kappa^*_{i-\frac{1}{2}}C^*_{i-1} - (\kappa^*_{i-\frac{1}{2}}+\kappa^*_{i+\frac{1}{2}})C^*_i + \kappa^*_{i+\frac{1}{2}}C^*_{i+1}}{dz^{*2}}
\end{aligned} \tag{3.4}$$

Of course, these equations do not hold at the boundary of the domain and will be treated later along with the discretization in time. Due to the numerical simplicity granted by (3.4), this form is utilized to derive a numerical solution to the forward problem.

After attempting the numerical solution of the diffusion problem with a forward Euler explicit method, added complexity was encountered by the need to abide to a stability condition ($\frac{\Delta x^2}{2\Delta t} > \kappa$). As the EKF will be using the same formulas, this poses

more of a problem in the recursive estimation rather than in the need to find a one-time solution (to simulate the truth). Therefore, instead of always ensuring that the values in the diffusivity vector have a maximum lower than the value prescribed by the stability criterion, an implicit scheme was adopted. The Crank-Nicolson scheme is stable and more accurate as it is second order time (unlike forward Euler which is first order accurate in time). Making use of this method to solve the equation numerically in time, the system is linearized into two first order accurate time stepping schemes, Classic Explicit (3.5) and Backward Implicit (3.6) schemes for a half time step, following the derivation of the Crank-Nicolson method as specified in Lapidus and Pinder (1982). The two first order schemes are then added to form (3.7). The superscript k is used to specify the time index.

$$\frac{C_i^{*k+\frac{1}{2}} - C_i^{*k}}{dt^*/2} = \frac{\kappa_{i-\frac{1}{2}}^{*k} C_{i-1}^{*k} - (\kappa_{i+\frac{1}{2}}^{*k} + \kappa_{i-\frac{1}{2}}^{*k}) C_i^{*k} + \kappa_{i+\frac{1}{2}}^{*k} C_{i+1}^{*k}}{dz^{*2}} \tag{3.5}$$

$$\frac{C_i^{*k+1} - C_i^{*k+\frac{1}{2}}}{dt^*/2} = \frac{\kappa_{i-\frac{1}{2}}^{*k+1} C_{i-1}^{*k+1} - (\kappa_{i+\frac{1}{2}}^{*k+1} + \kappa_{i-\frac{1}{2}}^{*k+1}) C_i^{*k+1} + \kappa_{i+\frac{1}{2}}^{*k+1} C_{i+1}^{*k+1}}{dz^{*2}} \tag{3.6}$$

and summing the two, the final equation becomes

$$\begin{aligned} 2\frac{C_i^{*k+1} - C_i^{*k}}{dt^*} \quad = \quad & \frac{\kappa_{i-\frac{1}{2}}^{*k+1} C_{i-1}^{*k+1} - (\kappa_{i+\frac{1}{2}}^{*k+1} + \kappa_{i-\frac{1}{2}}^{*k+1}) C_i^{*k+1} + \kappa_{i+\frac{1}{2}}^{*k+1} C_{i+1}^{*k+1}}{dz^{*2}} \\ & + \frac{\kappa_{i-\frac{1}{2}}^{*k} C_{i-1}^{*k} - (\kappa_{i+\frac{1}{2}}^{*k} + \kappa_{i-\frac{1}{2}}^{*k}) C_i^{*k} + \kappa_{i+\frac{1}{2}}^{*k} C_{i+1}^{*k}}{dz^{*2}} \end{aligned} \tag{3.7}$$

Separating the future time, t_{k+1}^*, from the current, the equality becomes

$$\begin{aligned} -\frac{dt^*}{2dz^{*2}} \kappa_{i-\frac{1}{2}}^{*k+1} C_{i-1}^{*k+1} + \left[1 + \frac{dt^*}{2dz^{*2}} (\kappa_{i+\frac{1}{2}}^{*k+1} + \kappa_{i-\frac{1}{2}}^{*k+1})\right] C_i^{*k+1} - \frac{dt^*}{2dz^{*2}} \kappa_{i+\frac{1}{2}}^{*k+1} C_{i+1}^{*k+1} = \\ \frac{dt^*}{2dz^{*2}} \kappa_{i-\frac{1}{2}}^{*k} C_{i-1}^{*k} + \left[1 - \frac{dt^*}{2dz^{*2}} (\kappa_{i+\frac{1}{2}}^{*k} + \kappa_{i-\frac{1}{2}}^{*k})\right] C_i^{*k} + \frac{dt^*}{2dz^{*2}} \kappa_{i+\frac{1}{2}}^{*k} C_{i+1}^{*k} \end{aligned} \tag{3.8}$$

This formula (3.8) is then simplified in the form of a matrix equation by the introduction of matrices $\mathbf{A}(\vec{\kappa^*})$ and $\mathbf{B}(\vec{\kappa^*})$ as functions of $\vec{\kappa^*}$.

$$\mathbf{A}(\vec{\kappa}^{*k+1}) \vec{C}^{*k+1} = \mathbf{B}(\vec{\kappa}^{*k}) \vec{C}^{*k} \tag{3.9}$$

In the above formula, matrix $\mathbf{A}(\vec{\kappa^*})$ computes the tracer concentration $C_i^{*k+\frac{1}{2}}$ numerically a half step backward in time from values of $\vec{C}$ at t_{k+1}. This is equated to the value obtained for $C_i^{*k+\frac{1}{2}}$ obtained explicitly from matrix $\mathbf{B}(\vec{\kappa}^*)$, which propagates the evolution in time of the concentration at $\vec{C}^k$ forward one half time step. Now the boundary conditions at time t_{k+1}^* must be accounted for. These will be incorporated into matrix $\mathbf{A}$. Again, the boundary conditions applied are the homogeneous Neumann boundary conditions on either side of the spatial domain, $z^* = 0$ and $z^* = 1$.

$$\begin{aligned} \left.\frac{\partial C^*}{\partial z^*}\right|_{z^*=0} &= \frac{-3C_1^* + 4C_2^* - 1C_3^*}{2dz^*} = 0 \quad \Rightarrow \quad C_1^* = \frac{4}{3}C_2^* - \frac{1}{3}C_3^* \\ \left.\frac{\partial C^*}{\partial z^*}\right|_{z^*=1} &= \frac{C_{n-2}^* - 4C_{n-1}^* + 3C_n^*}{2dz^*} = 0 \quad \Rightarrow \quad C_n^* = \frac{4}{3}C_{n-1}^* - \frac{1}{3}C_{n-2}^* \end{aligned} \tag{3.10}$$

Matrices $\mathbf{A}$ and $\mathbf{B}$ are then structured as

$$\mathbf{A}(\vec{\kappa}^{*k}) = \begin{bmatrix} 3 & -4 & 1 & \dots & 0 \\ \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & -\frac{dt^*}{2dz^{*2}}\kappa_{i-\frac{1}{2}}^{*k} & 1 + \frac{dt^*}{2dz^{*2}}(\kappa_{i+\frac{1}{2}}^{*k} + \kappa_{i-\frac{1}{2}}^{*k}) & -\frac{dt^*}{2dz^{*2}}\kappa_{i+\frac{1}{2}}^{*k} & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ 0 & \dots & 1 & -4 & 3 \end{bmatrix} \tag{3.11}$$

$$\mathbf{B}(\vec{\kappa}^{*k}) = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \frac{dt^*}{2dz^{*2}}\kappa_{i-\frac{1}{2}}^{*k} & 1 - \frac{dt^*}{2dz^{*2}}(\kappa_{i+\frac{1}{2}}^{*k} + \kappa_{i-\frac{1}{2}}^{*k}) & \frac{dt^*}{2dz^{*2}}\kappa_{i+\frac{1}{2}}^{*k} & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ 0 & \dots & 0 & 0 & 0 \end{bmatrix} \tag{3.12}$$

Forward stepping the state of the system (the concentration C) in time is then executed by

$$\vec{C}^{*k+1} = \mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k})\vec{C}^{*k} \tag{3.13}$$

Now prior to applying the state estimation methods described in Chapter 2 for

parameter estimation, the state must be augmented to include the parameters of interest. In doing so, the original system (3.1) becomes

$$\begin{pmatrix} \frac{\partial \vec{C}^*}{\partial t^*} \\ \frac{\partial \vec{\theta}}{\partial t^*} \end{pmatrix} = \begin{bmatrix} f(\vec{\theta}, t^*) & 0 \\ 0 & 0 \end{bmatrix} \begin{pmatrix} \vec{C}^* \\ \vec{\theta} \end{pmatrix} \tag{3.14}$$

where $f(\vec{\theta}, t^*)$ is a matrix function of t^* and the parameter vector $\vec{\theta}$ through $\vec{\kappa}^*$. Also apparent in this formula is the assumption that the parameters are constant in time (uncertainty in these values may be introduced later through process noise). The discrete version of these dynamics may also be written with the use of (3.11), (3.12) and (3.13).

$$\begin{pmatrix} \vec{C}^{*k+1} \\ \vec{\theta}^{k+1} \end{pmatrix} = \begin{bmatrix} \mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k}) & 0 \\ 0 & \mathbf{I} \end{bmatrix} \begin{pmatrix} \vec{C}^{*k} \\ \vec{\theta}^{k} \end{pmatrix} = \begin{pmatrix} \mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k})\vec{C}^{*k} \\ \vec{\theta}^{k} \end{pmatrix} \tag{3.15}$$

Once posed in such a fashion, parameter estimation may be carried out and attention will first be turned to the EKF.

3.1.2 Analytical Test Case

Another one-dimensional diffusion problem which will be used to evaluate the parameter estimation capabilities of the methods reviewed in Chapter 2 is derived analytically. Assuming a quadratic diffusivity profile, a solution is obtained from (3.1). The profile selected for this particular problem was chosen as $\kappa^*(z^*) = a(2 - z^*)^2$. The reason for the choice of the value 2 is to later simplify the calculation for the application of boundary conditions. Starting with the general form

$$\frac{\partial C}{\partial t^*} = \frac{\partial}{\partial z^*}\left(a(2 - z^*)^2 \frac{\partial C}{\partial z^*}\right) \tag{3.16}$$

and using separation of variables ($C(z^*, t^*)$ is assumed to have the form of $C(z^*, t^*) = \tau(t^*)\zeta(z^*)$).

$$\begin{aligned} (\tau(t^*)\zeta(z^*))_{t^*} &= (a(2-z^*)^2(\tau(t^*)\zeta(z^*))_{z^*})_{z^*} \\ \tau_t(t^*)\zeta(z^*) &= a(2-z^*)^2\tau(t^*)\zeta_{z^*z^*}(z^*) - 2a(2-z^*)\tau(t^*)\zeta_{z^*}(z^*) \end{aligned} \tag{3.17}$$

where the product rule has been applied and notation is simplified by representing partial derivatives with the use of subscripts (i.e. $\zeta_{z^*} = \frac{\partial \zeta}{\partial z^*}$). Separating the variables to opposite sides of the equality

$$\frac{\tau_{t^*}(t^*)}{\tau(t^*)} = \frac{a(2-z^*)^2\zeta_{z^*z^*}(z^*) - 2a(2-z^*)\zeta_{z^*}(z^*)}{\zeta(z^*)} \tag{3.18}$$

Since this equality must hold for any choice of variables, these ratios must be equivalent to a constant, which will be identified as $-\gamma$. The solution for the time dependent function becomes

$$\begin{aligned} \tau_{t^*}(t^*) &= -\gamma\tau(t^*) \\ \tau(t^*) &= e^{-\gamma t^*} \end{aligned} \tag{3.19}$$

The depth dependent portion of the solution is then derived from

$$\begin{aligned} \frac{a(2-z^*)^2\zeta_{z^*z^*}(z^*) - 2a(2-z^*)\zeta_{z^*}(z^*)}{\zeta(z^*)} &= -\gamma \\ a(2-z^*)^2\zeta_{z^*z^*}(z^*) - 2a(2-z^*)\zeta_{z^*}(z^*) + \gamma\zeta(z^*) &= 0 \end{aligned} \tag{3.20}$$

Introducing a variable substitution for z^* of the form $\alpha = \ln(2-z^*)$ into $\zeta(z^*)$, derivatives are obtained using the chain rule.

$$\begin{aligned} \zeta_{z^*}(z^*) &= \zeta_{z^*}(2-e^\alpha) \\ &= \tfrac{\partial \alpha}{\partial z^*}\zeta_\alpha(2-e^\alpha) \\ &= -\tfrac{1}{2-z^*}\zeta_\alpha(2-e^\alpha) \end{aligned}$$

$$\begin{aligned}
\zeta_{z^*z^*}(z^*) &= \zeta_{z^*z^*}(2-e^\alpha) \\
&= \left(\tfrac{\partial\alpha}{\partial z^*}\right)^2 \zeta_{\alpha\alpha}(2-e^\alpha) + \tfrac{\partial^2\alpha}{\partial z^{*2}}\zeta_\alpha(2-e^\alpha) \\
&= \left(-\tfrac{1}{2-z^*}\right)^2 \zeta_{\alpha\alpha}(2-e^\alpha) - \tfrac{1}{(2-z^*)^2}\zeta_\alpha(2-e^\alpha) \\
&= \tfrac{1}{(2-z^*)^2}(\zeta_{\alpha\alpha}(2-e^\alpha) - \zeta_\alpha(2-e^\alpha))
\end{aligned}$$

Substituting into (3.20) yields

$$\begin{aligned}
0 &= a(2-z^*)^2\left(\frac{1}{(2-z^*)^2}\left(\zeta_{\alpha\alpha}(2-e^\alpha) - \zeta_\alpha(2-e^\alpha)\right)\right) \\
&\quad -2a(2-z^*)\left(-\frac{1}{2-z^*}\zeta_\alpha(2-e^\alpha)\right) + \gamma\zeta(2-e^\alpha) \\
0 &= a\left(\zeta_{\alpha\alpha}(2-e^\alpha) - \zeta_\alpha(2-e^\alpha)\right) + 2a\zeta_\alpha(2-e^\alpha) + \gamma\zeta(2-e^\alpha) \\
0 &= a\zeta_{\alpha\alpha}(2-e^\alpha) + a\zeta_\alpha(2-e^\alpha) + \gamma\zeta(2-e^\alpha)
\end{aligned} \tag{3.21}$$

Another function substitution is then made replacing $\zeta(2-e^\alpha)$ with $e^{-\alpha/2}\beta(\alpha)$. The derivatives of ζ then become

$$\begin{aligned}
\zeta_\alpha(z^*) &= \zeta_\alpha(2-e^\alpha) \\
&= -\tfrac{1}{2}e^{-\alpha/2}\beta(\alpha) + e^{-\alpha/2}\beta_\alpha(\alpha)
\end{aligned}$$

$$\begin{aligned}
\zeta_{\alpha\alpha}(z^*) &= \zeta_{\alpha\alpha}(2-e^\alpha) \\
&= \tfrac{1}{4}e^{-\alpha/2}\beta(\alpha) - e^{-\alpha/2}\beta_\alpha(\alpha) + e^{-\alpha/2}\beta_{\alpha\alpha}(\alpha)
\end{aligned}$$

which when placed into (3.21) reduce the complexity further

$$\begin{aligned}
0 &= a\zeta_{\alpha\alpha}(2-e^\alpha) + a\zeta_\alpha(2-e^\alpha) + \gamma\zeta(2-e^\alpha) \\
0 &= a\left(\frac{1}{4}e^{-\alpha/2}\beta(\alpha) - e^{-\alpha/2}\beta_\alpha(\alpha) + e^{-\alpha/2}\beta_{\alpha\alpha}(\alpha)\right) \\
&\quad +a\left(-\frac{1}{2}e^{-\alpha/2}\beta(\alpha) + e^{-\alpha/2}\beta_\alpha(\alpha)\right) + \gamma e^{-\alpha/2}\beta(\alpha) \\
0 &= \frac{1}{4}a\beta(\alpha) + a\beta_{\alpha\alpha}(\alpha) - \frac{1}{2}a\beta(\alpha) + \gamma\beta(\alpha) \\
0 &= a\beta_{\alpha\alpha}(\alpha) - \frac{1}{4}a\beta(\alpha) + \gamma\beta(\alpha)
\end{aligned} \tag{3.22}$$

This leads to a function with a well known solution

$$\begin{aligned} \beta_{\alpha\alpha}(\alpha) &= \left(\tfrac{1}{4} - \tfrac{\gamma}{a}\right)\beta(\alpha) \\ \beta(\alpha) &= c_1 \cos\left(\sqrt{\tfrac{\gamma}{a} - \tfrac{1}{4}}\alpha\right) + c_2 \sin\left(\sqrt{\tfrac{\gamma}{a} - \tfrac{1}{4}}\alpha\right) \end{aligned} \tag{3.23}$$

where the constants of integration c_1 and c_2 remain to be determined. Back-substituting into $\zeta(2 - e^{\alpha})$

$$\zeta(2 - e^{\alpha}) = e^{-\alpha/2}\left(c_1 \cos\left(\sqrt{\frac{\gamma}{a} - \frac{1}{4}}\alpha\right) + c_2 \sin\left(\sqrt{\frac{\gamma}{a} - \frac{1}{4}}\alpha\right)\right) \tag{3.24}$$

and returning to the original space variable

$$\begin{aligned} \zeta(z^*) &= e^{-\ln(2-z^*)/2}\left(c_1 \cos\left(\sqrt{\tfrac{\gamma}{a} - \tfrac{1}{4}}\ln(2 - z^*)\right) + c_2 \sin\left(\sqrt{\tfrac{\gamma}{a} - \tfrac{1}{4}}\ln(2 - z^*)\right)\right) \\ \zeta(z^*) &= (2 - z^*)^{-1/2}\left(c_1 \cos\left(\sqrt{\tfrac{\gamma}{a} - \tfrac{1}{4}}\ln(2 - z^*)\right) + c_2 \sin\left(\sqrt{\tfrac{\gamma}{a} - \tfrac{1}{4}}\ln(2 - z^*)\right)\right) \end{aligned} \tag{3.25}$$

Now the general solution obtained, the constants of integration are sought by applying the no flux boundary conditions at $z^* = 0, 1$. Differentiating (3.25) with respect to z^* yields

$$\begin{aligned} \zeta_{z^*}(z^*) &= (2 - z^*)^{-1/2}\left(c_1 \sin\left(\sqrt{\frac{\gamma}{a} - \frac{1}{4}}\ln(2 - z^*)\right)\sqrt{\frac{\gamma}{a} - \frac{1}{4}}\frac{1}{2 - z^*}\right. \\ &\quad \left. - c_2 \cos\left(\sqrt{\frac{\gamma}{a} - \frac{1}{4}}\ln(2 - z^*)\right)\sqrt{\frac{\gamma}{a} - \frac{1}{4}}\frac{1}{2 - z^*}\right) \\ &\quad + \frac{(2 - z^*)^{-3/2}}{2}\left(c_1 \cos\left(\sqrt{\frac{\gamma}{a} - \frac{1}{4}}\ln(2 - z^*)\right) + c_2 \sin\left(\sqrt{\frac{\gamma}{a} - \frac{1}{4}}\ln(2 - z^*)\right)\right) \\ &= (2 - z^*)^{-3/2}\left[\left(c_1\sqrt{\frac{\gamma}{a} - \frac{1}{4}} + \frac{c_2}{2}\right)\sin\left(\sqrt{\frac{\gamma}{a} - \frac{1}{4}}\ln(2 - z^*)\right)\right. \\ &\quad \left. + \left(\frac{c_1}{2} - c_2\sqrt{\frac{\gamma}{a} - \frac{1}{4}}\right)\cos\left(\sqrt{\frac{\gamma}{a} - \frac{1}{4}}\ln(2 - z^*)\right)\right] \end{aligned} \tag{3.26}$$

Taking $\zeta_{z^*}(z^*) = 0$ at $z^* = 1$ the sine term in (3.26) vanishes due to the natural log of unity. This forces the constants to be related via

$$\frac{c_1}{2} - c_2\sqrt{\frac{\gamma}{a} - \frac{1}{4}} = 0$$

As a result, the constraint at the other boundary ($z^* = 0$, at the surface) reduces to

$$0 = 2^{-3/2}\left(2c_2\left(\frac{\gamma}{a} - \frac{1}{4}\right) + \frac{c_2}{2}\right)\sin\left(\sqrt{\frac{\gamma}{a} - \frac{1}{4}}\ln(2)\right)$$

where the substitution for c_1 has been made. For the above term to abide by the equality at $z^* = 0$, the sine term must equal zero at this location. Thus

$$\sqrt{\frac{\gamma}{a} - \frac{1}{4}}\ln(2) = \pi n \tag{3.27}$$

where n is any real integer. For convenience, the value of $n = 1$ will be used for this analytical test case. The initial concentration profile is therefore defined as

$$\zeta(z^*) = (2 - z^*)^{-1/2}\left(2c_2\frac{\pi}{\ln(2)}\cos\left(\frac{\pi}{\ln(2)}\ln(2 - z^*)\right) + c_2\sin\left(\frac{\pi}{\ln(2)}\ln(2 - z^*)\right)\right) \tag{3.28}$$

The constant c_2 will be used to normalize the initial condition. The sign will also be reversed to be somewhat representative of a thermocline or halocline, and an offset will be added to constrain values between 0 and 1. In this case the scaling is equivalent to

$$c_2 = -\frac{\ln(2)}{2\pi + \sqrt{2}\pi}$$

and the offset

$$\frac{2}{2 + \sqrt{2}}$$

is added.

The exponential decay rate, γ, is calculated from (3.27) with $n = 1$ as

$$\gamma = a\left[\frac{1}{4} - \left(\frac{\pi}{\ln(2)}\right)^2\right]$$

The parameter $a = 0.01$ is chosen to limit the diffusivity profile to a similar range of values as the hyperbolic tangent. This profile is shown in Fig. 3-2. The curve corresponds to $\kappa = az^{*2} - 4az^* + 4a$. It is therefore possible to split the profile into a function of three parameters a_1, a_2, and a_3 as the coefficients of each

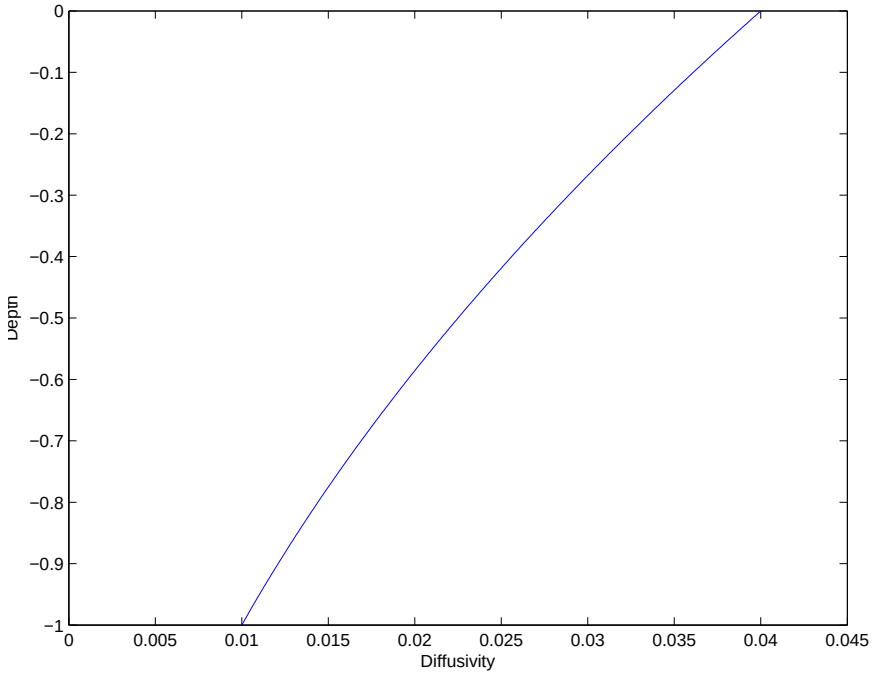

Figure 3-2: Quadratic diffusivity profile
The equation used for the generation of this profile is $\kappa^* = a(2 - z^*)^2 = a_1 z^{*2} + a_2 z^* + a_3$, with $a = 0.01$ yielding $a_1 = 0.01$, $a_2 = -0.04$, and $a_3 = 0.04$.

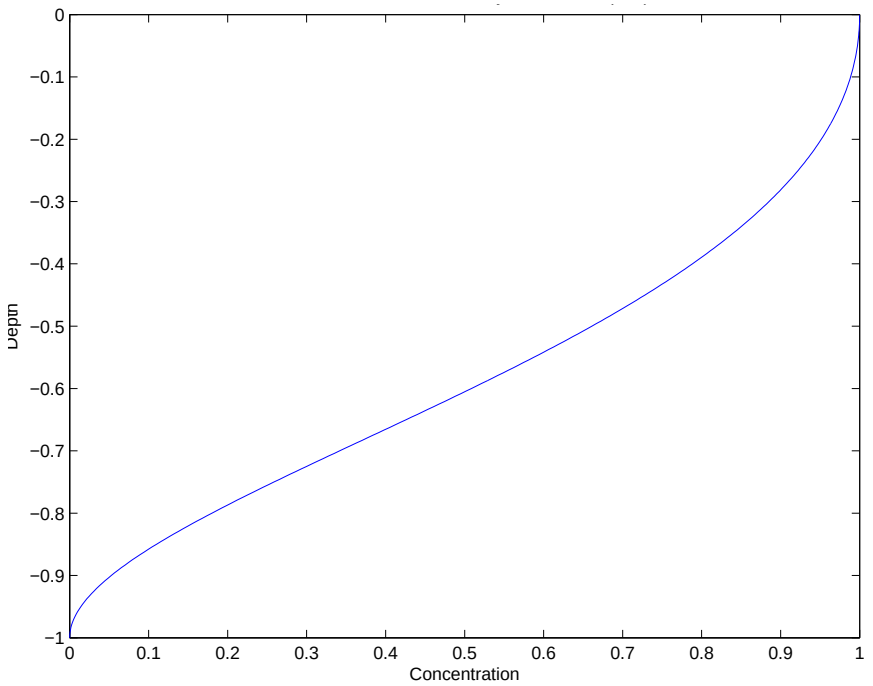

Figure 3-3: Initial concentration profile for analytic case
The first mode ($n = 1$) of the concentration profile for the analytical solution to the diffusion equation.

order of z^*, starting from highest to lowest. Thus the true parameter vector will be $\vec{\theta} = (0.01, -0.04, 0.04)^T$. The initial concentration profile obtained analytically is displayed in Fig. 3-3. Note that a diffusivity profile more similar to what is anticipated in the ocean would consist of a quadratically decreasing diffusivity in the surface layers (with an opposite curvature as the one used in this analytically case), transitioning to a linearly decreasing profile at depth.

3.2 Extended Kalman Filter

As explained previously, the Extended Kalman Filter (EKF) is based on the linearization of model dynamics locally in time. The Kalman Filter was originally developed for linear problems with uncertain forcings, that is for Ordinary Differential Equations (ODEs) of one variable (in this case time) with some inherent uncertainty. To apply the EKF to dynamics governed by Partial Differential Equations, the problem can be transformed into a set of ODEs. As such, the coupled set of equations in (3.14) must be made linear with respect to the augmented state $\begin{pmatrix} \vec{C}^* \\ \vec{\theta} \end{pmatrix}$. The Jacobian (in this case the derivative with respect to the augmented state) of (3.14) must be calculated. A numerical solution, discretized in time, will be the main form of the model used, the continuous equations will be set aside, and the Jacobian will be computed for the discrete form (3.15). This propagator will be denoted by F.

$$\begin{aligned} F &= \frac{\partial}{\partial \begin{pmatrix} \vec{C}^{*k} \\ \vec{\theta}^k \end{pmatrix}} \begin{pmatrix} \vec{C}^{*k+1} \\ \vec{\theta}^{k+1} \end{pmatrix} = \begin{bmatrix} \frac{\partial \vec{C}^{*k+1}}{\partial \vec{C}^{*k}} & \frac{\partial \vec{C}^{*k+1}}{\partial \vec{\theta}^k} \\ \frac{\partial \vec{\theta}^{k+1}}{\partial \vec{C}^{*k}} & \frac{\partial \vec{\theta}^{k+1}}{\partial \vec{\theta}^k} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k}) & \frac{\partial}{\partial \vec{\theta}^k}\left(\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k})\vec{C}^{*k}\right) \\ 0 & \mathbf{I} \end{bmatrix} \end{aligned} \tag{3.29}$$

Having established that $\vec{\kappa}^*$ is a function of $\vec{\theta}$, the resulting Jacobian for the vector function relating the two now needs to be specified as well. Thus, looking at the term in the upper right corner of the matrix in (3.29), it is possible to expand the

derivative in the following form

$$
\begin{aligned}
\frac{\partial \vec{C}^{*k+1}}{\partial \vec{\theta}^k} &= \frac{\partial}{\partial \vec{\theta}^k}\left(\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k})\vec{C}^{*k}\right) \\
&= \frac{\partial\left(\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\right)}{\partial \vec{\theta}^k}\mathbf{B}(\vec{\kappa}^{*k})\vec{C}^{*k} \\
&\quad +\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\frac{\partial\left(\mathbf{B}(\vec{\kappa}^{*k})\right)}{\partial \vec{\theta}^k}\vec{C}^{*k} \\
&\quad +\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k})\frac{\partial\left(\vec{C}^{*k}\right)}{\partial \vec{\theta}^k} \\
&= -\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^{*k+1})\right)}{\partial \vec{\theta}^k}\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k})\vec{C}^{*k} \\
&\quad +\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\frac{\partial\left(\mathbf{B}(\vec{\kappa}^{*k})\right)}{\partial \vec{\theta}^k}\vec{C}^{*k} \\
&\quad +\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k})\frac{\partial}{\partial \vec{\theta}^k}\left(\mathbf{A}^{-1}(\vec{\kappa}^{*k})\mathbf{B}(\vec{\kappa}^{*k-1})\vec{C}^{*k-1}\right) \\
&= -\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^{*k+1})\right)}{\partial \vec{\theta}^k}\vec{C}^{*k+1} \\
&\quad +\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\frac{\partial\left(\mathbf{B}(\vec{\kappa}^{*k})\right)}{\partial \vec{\theta}^k}\vec{C}^{*k} \\
&\quad +\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}^{-1}(\vec{\kappa}^{*k})\right)}{\partial \vec{\theta}^k}\mathbf{B}(\vec{\kappa}^{*k-1})\vec{C}^{*k-1} \\
&= \mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\left[-\frac{\partial\left(\mathbf{A}(\vec{\kappa}^{*k+1})\right)}{\partial \vec{\theta}^k}\vec{C}^{*k+1}+\frac{\partial\left(\mathbf{B}(\vec{\kappa}^{*k})\right)}{\partial \vec{\theta}^k}\vec{C}^{*k}\right. \\
&\quad \left.-\mathbf{B}(\vec{\kappa}^{*k})\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^{*k})\right)}{\partial \vec{\theta}^k}\mathbf{A}^{-1}(\vec{\kappa}^{*k})\mathbf{B}(\vec{\kappa}^{*k-1})\vec{C}^{*k-1}\right] \\
&= \mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\left[-\frac{\partial\left(\mathbf{A}(\vec{\kappa}^{*k+1})\right)}{\partial \vec{\theta}^k}\vec{C}^{*k+1}+\frac{\partial\left(\mathbf{B}(\vec{\kappa}^{*k})\right)}{\partial \vec{\theta}^k}\vec{C}^{*k}\right. \\
&\quad \left.-\mathbf{B}(\vec{\kappa}^{*k})\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^{*k})\right)}{\partial \vec{\theta}^k}\vec{C}^{*k}\right] \\
&= \mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\left[-\frac{\partial\left(\mathbf{A}(\vec{\kappa}^{*k+1})\right)}{\partial \vec{\kappa}^{*k+1}}\frac{\partial \vec{\kappa}^{*k+1}}{\partial \vec{\theta}^{k+1}}\frac{\partial \vec{\theta}^{k+1}}{\partial \vec{\theta}^k}\vec{C}^{*k+1}\right. \\
&\quad +\frac{\partial\left(\mathbf{B}(\vec{\kappa}^{*k})\right)}{\partial \vec{\kappa}^{*k}}\frac{\partial \vec{\kappa}^{*k}}{\partial \vec{\theta}^k}\vec{C}^{*k} \\
&\quad \left.-\mathbf{B}(\vec{\kappa}^{*k})\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^{*k})\right)}{\partial \vec{\kappa}^{*k}}\frac{\partial \vec{\kappa}^{*k}}{\partial \vec{\theta}^k}\vec{C}^{*k}\right] \\
&= \mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\left[-\frac{\partial\left(\mathbf{A}(\vec{\kappa}^{*})\right)}{\partial \vec{\kappa}^{*}}\vec{C}^{*k+1}+\frac{\partial\left(\mathbf{B}(\vec{\kappa}^{*})\right)}{\partial \vec{\kappa}^{*}}\vec{C}^{*k}\right. \\
&\quad \left.-\mathbf{B}(\vec{\kappa}^{*k})\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^{*})\right)}{\partial \vec{\kappa}^{*}}\vec{C}^{*k}\right]\frac{\partial \vec{\kappa}^{*}}{\partial \vec{\theta}^k}
\end{aligned}
\tag{3.30}
$$

Where the property $\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{X}} = -\mathbf{A}^{-1}\frac{\partial \mathbf{A}}{\partial \mathbf{X}}\mathbf{A}^{-1}$ was substituted in the above equation. Also, in the previous derivation $\frac{\partial \vec{\theta}^{k+1}}{\partial \vec{\theta}^{k}} = \mathbf{I}$ was used. In addition, the fact that $\vec{\kappa}^{*k}$ is solely a function of $\vec{\theta}^{k}$ (and does not vary with time, unless $\vec{\theta}^{k}$ is dynamic) was utilized. As a result, where appropriate, the time index was dropped on these variables. The derivative of $\vec{\kappa}^{*k}$ with respect to $\vec{\theta}^{k}$ will vary depending on the chosen parameterization. Therefore, the derivative of the matrix operators $\mathbf{A}$ and $\mathbf{B}$ will first be derived with respect to $\vec{\kappa}^{*}$. To avoid the need for representing three-dimensional arrays, the product of the derivative with the vector C^{*} is maintained, thus reducing the final derivation to a two-dimensional matrix.

$$
\begin{aligned}
\frac{\partial \mathbf{A}(\vec{\kappa}^{*})}{\partial \vec{\kappa}^{*}}\vec{C}^{*} &= \frac{\partial \mathbf{A}_{i,j}}{\partial \kappa^{*}_{k}}C^{*}_{j} \\
&= [\cdot]_{i,k} \\
&= \frac{\partial \mathbf{A}_{i,j}C^{*}_{j}}{\partial \kappa^{*}_{k}} \\
&= \frac{\partial}{\partial \vec{\kappa}^{*}_{k}}\left[-\frac{dt^{*}}{2dz^{*2}}\kappa^{*}_{i-\frac{1}{2}}C^{*}_{i-1} + \left(1 + \frac{dt^{*}}{2dz^{*2}}(\kappa^{*}_{i+\frac{1}{2}} + \kappa^{*}_{i-\frac{1}{2}})\right)C^{*}_{i} - \frac{dt^{*}}{2dz^{*2}}\kappa^{*}_{i+\frac{1}{2}}C^{*}_{i+1}\right]; \text{ for } i \neq 1, n \\
&= \frac{\partial}{\partial \vec{\kappa}^{*}_{k}}\left[-\frac{dt^{*}}{2dz^{*2}}\kappa^{*}_{i}C^{*}_{i-\frac{1}{2}} + \left(1 + \frac{dt^{*}}{2dz^{*2}}(\kappa^{*}_{i+1} + \kappa^{*}_{i})\right)C^{*}_{i+\frac{1}{2}} - \frac{dt^{*}}{2dz^{*2}}\kappa^{*}_{i+1}C^{*}_{i+\frac{3}{2}}\right]; \text{ for } i\exists[\tfrac{3}{2}, \tfrac{5}{2}, \ldots, n-\tfrac{3}{2}] \\
&= \frac{\partial}{\partial \vec{\kappa}^{*}_{k}}\left[\frac{dt^{*}}{2dz^{*2}}(-C^{*}_{i-\frac{1}{2}} + C^{*}_{i+\frac{1}{2}})\kappa^{*}_{i} + \frac{dt^{*}}{2dz^{*2}}(C^{*}_{i+\frac{1}{2}} - C^{*}_{i+\frac{3}{2}})\kappa^{*}_{i+1}\right] \\
&= \frac{dt^{*}}{2dz^{*2}}(-C^{*}_{i-\frac{1}{2}} + C^{*}_{i+\frac{1}{2}})\delta_{i,k} + \frac{dt^{*}}{2dz^{*2}}(C^{*}_{i+\frac{1}{2}} - C^{*}_{i+\frac{3}{2}})\delta_{i+1,k}; \text{ for } k\exists[\tfrac{3}{2}, \tfrac{5}{2}, \ldots, n-\tfrac{1}{2}] \\
&= \begin{cases} \frac{dt^{*}}{2dz^{*2}}(-C^{*}_{i-1} + C^{*}_{i})\delta_{i-\frac{1}{2},k-\frac{1}{2}} + \frac{dt^{*}}{2dz^{*2}}(C^{*}_{i} - C^{*}_{i+1})\delta_{i+\frac{1}{2},k-\frac{1}{2}} & \text{for } i\exists[2, n-1] \text{ and } k\exists[2, n] \\ 0 & \text{for } i = 1, n \text{ and } k\exists[2, n] \end{cases} \\
&= \begin{cases} \frac{dt^{*}}{2dz^{*2}}(-C^{*}_{i-1} + C^{*}_{i})\delta_{i-\frac{1}{2},k+\frac{1}{2}} + \frac{dt^{*}}{2dz^{*2}}(C^{*}_{i} - C^{*}_{i+1})\delta_{i+\frac{1}{2},k+\frac{1}{2}} & \text{for } i\exists[2, n-1] \text{ and } k\exists[1, n-1] \\ 0 & \text{for } i = 1, n \text{ and } k\exists[1, n-1] \end{cases} \\
&= \frac{dt^{*}}{2dz^{*2}}\begin{bmatrix} 0 & 0 & \cdots & 0 \\ \ddots & \ddots & \ddots & \vdots \\ 0 & -C^{*}_{i-1} + C^{*}_{i} & C^{*}_{i} - C^{*}_{i+1} & 0 \\ \vdots & \ddots & \ddots & \ddots \\ 0 & \cdots & 0 & 0 \end{bmatrix}
\end{aligned}
\tag{3.31}
$$

$$
\begin{aligned}
\frac{\partial \mathbf{B}(\vec{\kappa}^*)}{\partial \vec{\kappa}^*}\vec{C}^* &= \frac{\partial \mathbf{B}_{i,j}}{\partial \kappa^*_k} C^*_j \\
&= [\cdot]_{i,k} \\
&= \frac{\partial \mathbf{B}_{i,j} C^*_j}{\partial \kappa^*_k} \\
&= \frac{\partial}{\partial \vec{\kappa}^*_k}\left[\frac{dt^*}{2dz^{*2}}\kappa^*_{i-\frac{1}{2}} C^*_{i-1} + \left(1 - \frac{dt^*}{2dz^{*2}}(\kappa^*_{i+\frac{1}{2}} + \kappa^*_{i-\frac{1}{2}})\right) C^*_i + \frac{dt^*}{2dz^{*2}}\kappa^*_{i+\frac{1}{2}} C^*_{i+1}\right]; \text{ for } i \neq 1, n \\
&= \frac{\partial}{\partial \vec{\kappa}^*_k}\left[\frac{dt^*}{2dz^{*2}}\kappa^*_i C^*_{i-\frac{1}{2}} + \left(1 - \frac{dt^*}{2dz^{*2}}(\kappa^*_{i+1} + \kappa^*_i)\right) C^*_{i+\frac{1}{2}} + \frac{dt^*}{2dz^{*2}}\kappa^*_{i+1} C^*_{i+\frac{3}{2}}\right]; \text{ for } i \exists [\tfrac{3}{2}, \tfrac{5}{2}, \ldots, n - \tfrac{3}{2}] \\
&= \frac{\partial}{\partial \vec{\kappa}^*_k}\left[\frac{dt^*}{2dz^{*2}}(C^*_{i-\frac{1}{2}} - C^*_{i+\frac{1}{2}})\kappa^*_i - \frac{dt^*}{2dz^{*2}}(C^*_{i+\frac{1}{2}} - C^*_{i+\frac{3}{2}})\kappa^*_{i+1}\right] \\
&= \frac{dt^*}{2dz^{*2}}(C^*_{i-\frac{1}{2}} - C^*_{i+\frac{1}{2}})\delta_{i,k} - \frac{dt^*}{2dz^{*2}}(C^*_{i+\frac{1}{2}} - C^*_{i+\frac{3}{2}})\delta_{i+1,k}; \text{ for } k \exists [\tfrac{3}{2}, \tfrac{5}{2}, \ldots, n - \tfrac{1}{2}] \\
&= \begin{cases} \frac{dt^*}{2dz^{*2}}(C^*_{i-1} - C^*_i)\delta_{i-\frac{1}{2},k-\frac{1}{2}} - \frac{dt^*}{2dz^{*2}}(C^*_i - C^*_{i+1})\delta_{i+\frac{1}{2},k-\frac{1}{2}} & \text{for } i \exists [2, n-1] \text{ and } k \exists [2, n] \\ 0 & \text{for } i = 1, n \text{ and } k \exists [2, n] \end{cases} \\
&= \begin{cases} \frac{dt^*}{2dz^{*2}}(C^*_{i-1} - C^*_i)\delta_{i-\frac{1}{2},k+\frac{1}{2}} - \frac{dt^*}{2dz^{*2}}(C^*_i - C^*_{i+1})\delta_{i+\frac{1}{2},k+\frac{1}{2}} & \text{for } i \exists [2, n-1] \text{ and } k \exists [1, n-1] \\ 0 & \text{for } i = 1, n \text{ and } k \exists [1, n-1] \end{cases} \\
&= \frac{dt^*}{2dz^{*2}}\begin{bmatrix} 0 & 0 & \ldots & 0 \\ \ddots & \ddots & \ddots & \vdots \\ 0 & C^*_{i-1} - C^*_i & -C^*_i + C^*_{i+1} & 0 \\ \vdots & \ddots & \ddots & \ddots \\ 0 & \ldots & 0 & 0 \end{bmatrix} \\
&= -\frac{\partial \mathbf{A}(\vec{\kappa}^*)}{\partial \vec{\kappa}^*}\vec{C}^*
\end{aligned}
\tag{3.32}
$$

Then, substituting (3.31) and (3.32) into (3.30) yields

$$
\begin{aligned}
\frac{\partial \vec{C}^{*k+1}}{\partial \vec{\theta}^k} &= \frac{\partial}{\partial \vec{\kappa}^{*k}}\left(\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k})\vec{C}^{*k}\right) \\
&= \mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\frac{dt^*}{2dz^{*2}}\left\{\begin{bmatrix} 0 & 0 & \ldots & 0 \\ \ddots & \ddots & \ddots & \vdots \\ 0 & C^{*k}_{i-1} + C^{*k+1}_{i-1} - C^{*k}_i - C^{*k+1}_i & -C^{*k}_i - C^{*k+1}_i + C^{*k}_{i+1} + C^{*k+1}_{i+1} & 0 \\ \vdots & \ddots & \ddots & \ddots \\ 0 & \ldots & 0 & 0 \end{bmatrix}\right. \\
&\quad \left. + \mathbf{B}(\vec{\kappa}^{*k})\mathbf{A}^{-1}(\vec{\kappa}^{*k})\begin{bmatrix} 0 & 0 & \ldots & 0 \\ \ddots & \ddots & \ddots & \vdots \\ 0 & C^{*k}_{i-1} - C^{*k}_i & -C^{*k}_i + C^{*k}_{i+1} & 0 \\ \vdots & \ddots & \ddots & \ddots \\ 0 & \ldots & 0 & 0 \end{bmatrix}\right\}\frac{\partial \vec{\kappa}^*}{\partial \vec{\theta}^k}
\end{aligned}
\tag{3.33}
$$

where the sign in the last line of the above equation was carried through the difference matrix $\frac{\partial \mathbf{A}(\vec{\kappa}^*)}{\partial \vec{\kappa}^*}\vec{C}^{*k}$. The relationship between $\vec{\kappa}^*$ and the parameters in $\vec{\theta}$ will vary depending on the parameterization chosen and as such will have to be computed for every case, however, the main structure of the discrete propagation matrix has been derived with respect to $\vec{\kappa}^*$ for convenience when altering the model of choice. It is then possible to construct the Jacobian F in (3.29) from the previous equations

for use as the discrete transition matrix, $\mathbf{\Phi}_k$, in (2.12) and (2.13). Augmenting the observation matrix by the addition of zero columns for the unobserved parameters, then, all the elements necessary to run the recursive state estimation algorithm for parameter estimation (found in the now augmented state) are available. To apply this to any arbitrary one-dimensional test case only the derivative of $\vec{\kappa}$ with respect to $\vec{\theta}$ is required and multiplied through the derivatives previously evaluated to construct the discrete transition matrix.

3.3 Adjoint Method

The adjoint method differs from the EKF in that, instead of recursively changing the parameters at every step, the algorithm updates its original estimate (of the state) in a sort of "batch" process using the available observations at all times in the period of interest. Such a method may be more useful when a delay exists between the time data is collected and reported and when the future estimate is needed. The process, then, consists of running the entire forward model with the first set of parameter estimates while in the process collecting the misfits between the output state and the observed data. As a result, this method is not as useful in real-time application unless the time period of interest is decomposed into independent sets (batches) of given duration. Each forward model run is followed by a type of back propagation through the adjoint model with which the sensitivity of the cost function with respect to the parameters is determined providing a indication as to how to alter the parameters. This is performed by using a root finding method to set the sensitivity (derivative of the cost with respect to the parameters) to zero. The sensitivity is obtained using the algorithm specified in Chapter 2 from Robinson et al. (1998).

The root finding method chosen will determine the effectiveness and rapidity of convergence for the adjoint method. The use of a steepest descent method toward the minimum cost function requires determining an adequate step size by which to alter the parameters with respect to the computed gradient. This method requires the least amount of additional computations, but it is sensitive to the choice of the stepping

scale. A more rapid convergence rate may be obtain by the use of Newton's method. It is a more robust method but requires the calculation of the Hessian (second vector derivative) of the cost function. The second derivative of the system with respect to the parameters of interest is therefore needed, and as seen in the next derivation, the computational cost of the following operation would significantly increase the expense of implementing such an algorithm.

$$
\begin{aligned}
\frac{\partial^2 \vec{C}^{*k+1}}{(\partial\vec{\theta}^k)^2} &= \frac{\partial^2}{(\partial\vec{\theta}^k)^2}\left(\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k})\vec{C}^{*k}\right) \\
&= \frac{\partial}{\partial\vec{\theta}^k}\left\{\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\left[-\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\vec{C}^{*k+1}+\frac{\partial\left(\mathbf{B}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\vec{C}^{*k}\right.\right. \\
&\quad \left.\left.-\,\mathbf{B}(\vec{\kappa}^{*k})\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\vec{C}^{*k}\right]\frac{\partial\vec{\kappa}^*}{\partial\vec{\theta}^k}\right\} \\
&= -\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^{*k+1})\right)}{\partial\vec{\theta}^k}\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\left[-\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\vec{C}^{*k+1}\right. \\
&\quad \left.+\,\frac{\partial\left(\mathbf{B}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\vec{C}^{*k}-\mathbf{B}(\vec{\kappa}^{*k})\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\vec{C}^{*k}\right]\frac{\partial\vec{\kappa}^*}{\partial\vec{\theta}^k} \\
&\quad +\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\left[-\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\frac{\partial\vec{C}^{*k+1}}{\partial\vec{\theta}^k}+\frac{\partial\left(\mathbf{B}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\frac{\partial\vec{C}^{*k}}{\partial\vec{\theta}^k}\right. \\
&\quad -\frac{\partial\left(\mathbf{B}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\frac{\partial\vec{\kappa}}{\partial\vec{\theta}^k}\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\vec{C}^{*k} \\
&\quad +\mathbf{B}(\vec{\kappa}^{*k})\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\vec{\kappa}^*}{\partial\vec{\theta}^k}\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\vec{C}^{*k} \\
&\quad \left.-\,\mathbf{B}(\vec{\kappa}^{*k})\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\frac{\partial\vec{C}^{*k}}{\partial\vec{\theta}^k}\right]\frac{\partial\vec{\kappa}^*}{\partial\vec{\theta}^k} \\
&\quad +\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\left[-\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\vec{C}^{*k+1}\right. \\
&\quad \left.+\,\frac{\partial\left(\mathbf{B}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\vec{C}^{*k}-\mathbf{B}(\vec{\kappa}^{*k})\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\vec{C}^{*k}\right]\frac{\partial^2\vec{\kappa}^*}{(\partial\vec{\theta}^k)^2} \\
&= -2\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\frac{\partial\vec{\kappa}^*}{\partial\vec{\theta}^k}\frac{\partial\vec{C}^{*k+1}}{\partial\vec{\theta}^k}+\frac{\partial\vec{C}^{*k+1}}{\partial\vec{\kappa}^*}\frac{\partial^2\vec{\kappa}^*}{(\partial\vec{\theta}^k)^2} \\
&\quad +\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\frac{\partial\left(\mathbf{B}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\frac{\partial\vec{C}^{*k}}{\partial\vec{\theta}^k}\frac{\partial\vec{\kappa}^*}{\partial\vec{\theta}^k} \\
&\quad -\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k})\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\frac{\partial\vec{C}^{*k}}{\partial\vec{\theta}^k}\frac{\partial\vec{\kappa}^*}{\partial\vec{\theta}^k} \\
&\quad -\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\frac{\partial\left(\mathbf{B}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\frac{\partial\vec{\kappa}}{\partial\vec{\theta}^k}\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\vec{C}^{*k}\frac{\partial\vec{\kappa}^*}{\partial\vec{\theta}^k} \\
&\quad +\mathbf{A}^{-1}(\vec{\kappa}^{*k+1})\mathbf{B}(\vec{\kappa}^{*k})\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\mathbf{A}^{-1}(\vec{\kappa}^{*k})\frac{\partial\vec{\kappa}^*}{\partial\vec{\theta}^k}\frac{\partial\left(\mathbf{A}(\vec{\kappa}^*)\right)}{\partial\vec{\kappa}^*}\vec{C}^{*k}\frac{\partial\vec{\kappa}^*}{\partial\vec{\theta}^k}
\end{aligned}
\tag{3.34}
$$

3.4 Ensemble Methods

Here, the EnKF is applied for use as a recursive parameter estimation method which does not require linearization of the system equation. There are two options in the application of this particular ensemble method, much like the two schemes presented by Lermusiaux in his ESSE methods. Either a new ensemble is generated after every Kalman update in a resampling process utilizing the newly obtained estimate on the uncertainty (this would be analogous to Lermusiaux's Scheme A) or the original ensemble members are all individually updated (akin to Scheme B). In the case of resampling, the usual Kalman update can be carried out by adding process noise $\mathbf{Q}$ to the *a posteriori* estimate prior to reseeding a new ensemble set. On the other hand, if the same ensemble is utilized without reinitialization, the estimated process noise covariance has to first be used to generate an ensemble of the same size to add to the original. Anderson and Anderson (2001) refer to the Monte Carlo method based on a Gaussian prior distribution and with resampling after every update as a Gaussian Ensemble Filter. Here, the term EnKF will be used for both cases and focus will be placed on the resampling method.

3.5 Unscented Kalman Filter

The UKF does not require linearization of the system of equations, as such, the method described in Chapter 2 is applied using the optimal values suggested for Gaussian priors by Julier et al. (2000).

3.6 Results and Comparison of Methods

3.6.1 Numerical Test Case

Absence of Noise

The following results have been obtained with initial parameter estimates of $a_1 = 0.4$, $a_2 = 0.013$, and $a_3 = 0.027$ for the "true" parameters $a_1^t = 0.25$, $a_2^t = 0.01$,

$a_3^t = 0.03$ as identified in Fig. 3-1. The initial condition from which the simulation is started is a Gaussian bell, a normal distribution corresponding to $C^*(t_0) = \exp\left(\frac{-(z^*-0.5)^2}{(0.1)^2}\right)$. The maximum value of this profile is set to unity midway down the fictitious water column as a means to non-dimensionalize the tracer concentration. The estimated uncertainty in the initial, non-augmented state (the concentration) is of a variance of 10^{-8}. This same uncertainty is used for the measurements (measurement noise covariance). Realistically, though only limited in accuracy by machine precision, in this case 2.2204×10^{-16}, a value no less than a standard deviation of order 10^{-8} (variance of order 10^{-16}) should be utilized. Due to the sensitivity of certain state estimation methods, namely the EKF and UKF, to the inversion of this covariance matrix, a larger limiting value still should be utilized. For these simple cases, the previously mentioned variance led to stable responses. The corresponding uncertainties in the parameters are variances of 0.04, 0.0004, and 0.0004, respectively. In the absence of process noise, the $\mathbf{Q}$ matrix was set to zero. In this case, measured data is assimilated at every time step.

The results from using the three different recursive methods formerly discussed are presented in Figures 3-4, 3-5, and 3-6. In these plots, the recursion was carried for 100 times steps. It is evident from Fig. 3-4 that the EKF is a poor algorithm for tracking the true parameters in this highly nonlinear test case. The estimated covariance of the parameters is drastically underestimated resulting in rapid convergence of the filter to the invalid first estimate with slow adjustments thereafter. One solution to this behavior of the EKF is to "tune" the filter by introducing process noise, i.e. increasing the value of the $\mathbf{Q}$ matrix.

The EnKF performs remarkably better as is visible in Fig. 3-5. As a result of carrying the uncertainty through a Monte Carlo ensemble of parameters run through the nonlinear model, a better approximation of the true nonlinearly transformed covariance is obtained with ensemble statistics. It will be shown in the plot of normalized parameters (Fig. 3-7) that the first standard deviation about the estimate almost always contains the true value of the parameter. In fact, the parameters rarely deviate by more than ten percent from the true value. Another notable observation is

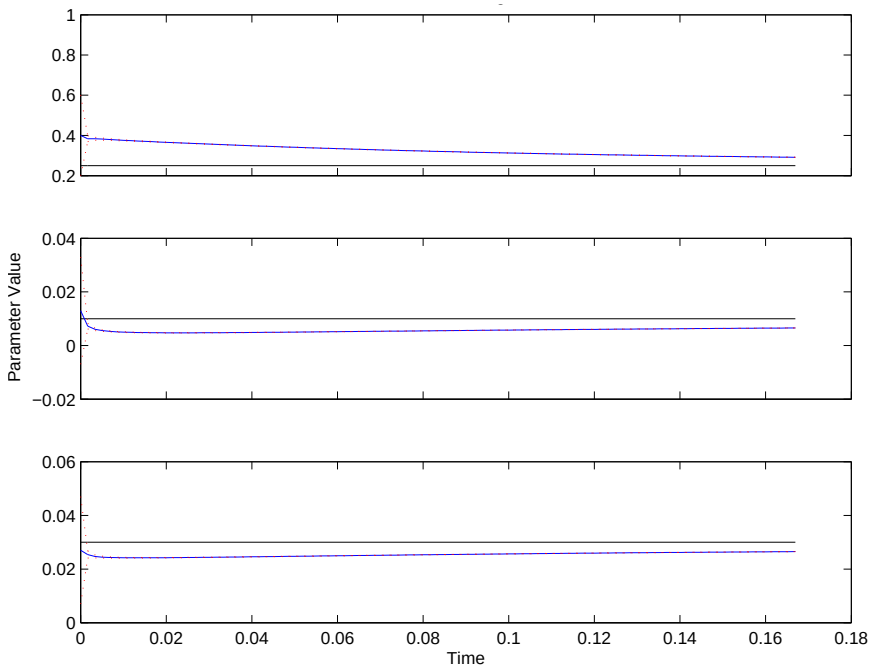

Figure 3-4: Parameter tracking performance of EKF without noise
Where from top to bottom the parameters are a_1, a_2, and a_3. The solid black line represents the true value, the solid blue curve is the estimated value, and the dotted red lines are the estimates +/- one standard deviation. The convention follows in the next figures.

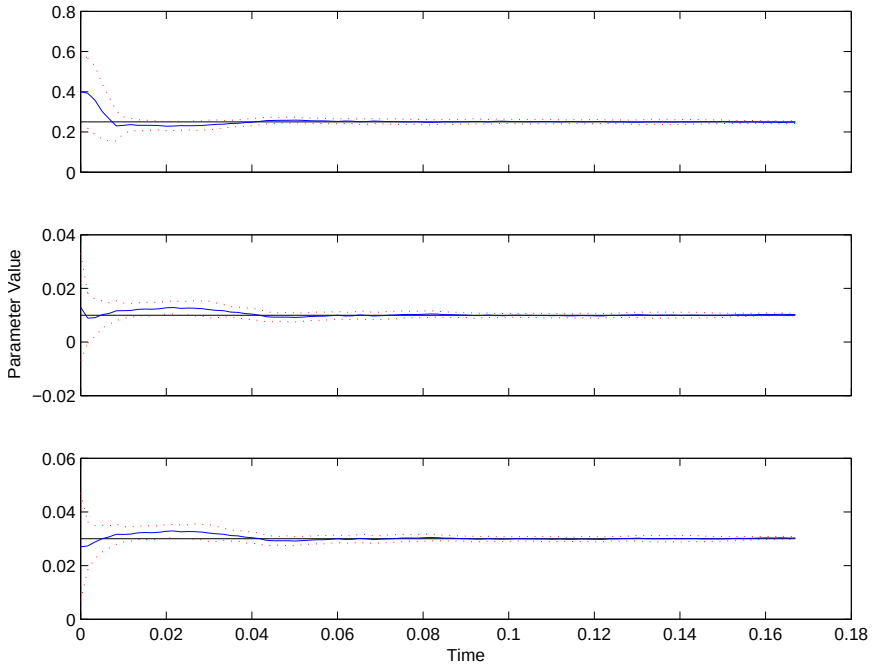

Figure 3-5: Parameter tracking performance of EnKF without noise
Here, an ensemble size of 100 members was used and resampling was performed after every Kalman update (assimilation of measurement data). Again, the convention is the same as in the previous figure.

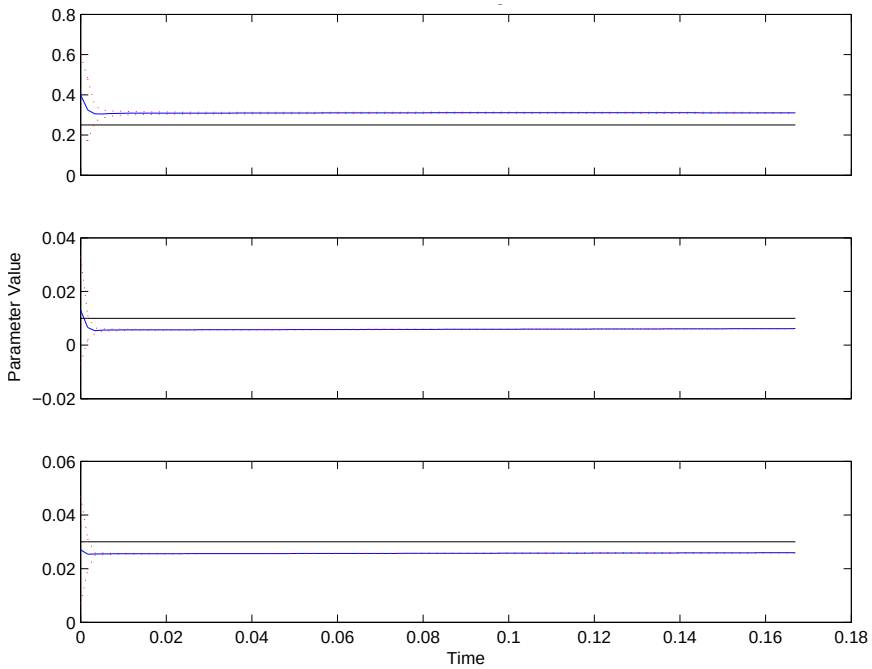

Figure 3-6: Parameter tracking performance of UKF without noise

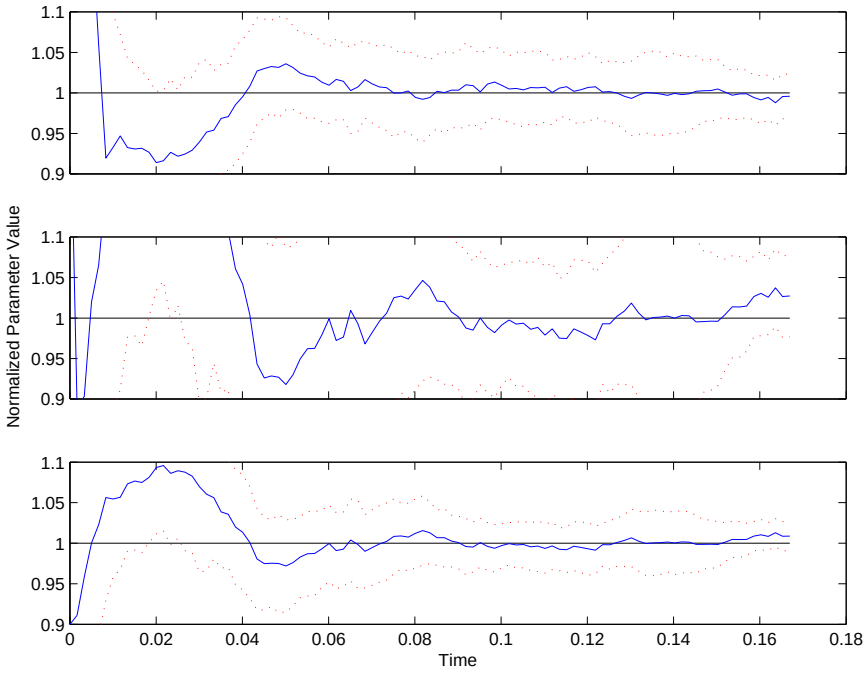

Figure 3-7: Normalized performance of EnKF without noise

the increased amplitude in the oscillations about the second parameter. This feature suggests a higher sensitivity for the variable a_2, which sets the range in the diffusivity value.

As pointed out by Julier, the UKF is expected to perform marginally better than the EKF without the need for linearization of the state dynamics. Looking at Fig. 3-6, the variance about the estimated parameters is in fact larger than that seen with the EKF, yet the filter converges to the wrong values. A possible reason for such an observation is that the selected sigma points, chosen optimally for a stochastic variable with an uncertainty represented by a Gaussian distribution, may not be ideal for this particular parameterization. It is likely that nonlocal effects are captured by the seven sigma points used to represent the original parameter uncertainty. One possible solution to this dilemma is the use of Julier's Scaled Unscented Transformation Julier (2002). Another option is to make sure that initial estimates are near the anticipated truth, which may not be an option over which one has any control.

The profile obtained after 100 time steps of the recursive filtering is very similar for both the UKF and the EKF. The results ("true", measured, forecast, and estimated

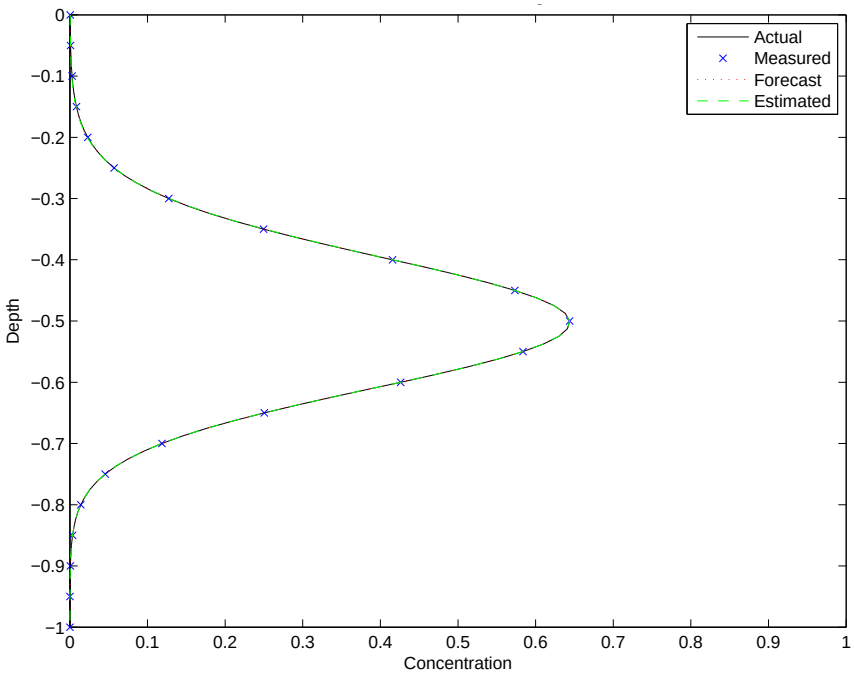

Figure 3-8: Tracer profile at 100 time steps using EKF without noise

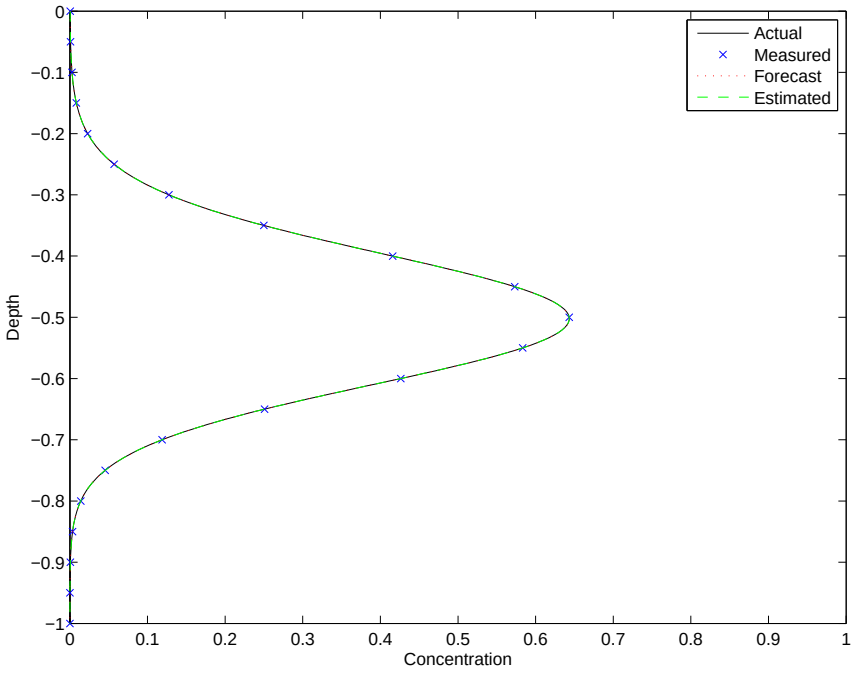

Figure 3-9: Tracer profile at 100 time steps using EnKF without noise

values) are shown, as described in the legend, for the EKF in Fig. 3-8. A faint deviation between the actual concentration and estimated concentration is visible in the upper portion of this curve where the diffusivity is highest (due to the particular parameterization utilized). Comparing this plot to the estimates obtained by the EnKF (Fig. 3-9), the advantage of the nonlinear propagation is apparent.

Measurement Noise

Introducing measurement noise to these one-dimensional simulations, the matrices used in the Kalman update scheme are also altered. In contrast to the previous runs without noise, a measurement error of variance 10^{-6} is introduced. This same value is used in forming the assumed error covariance matrix $\mathbf{R}$ and for the initial uncertainty matrix for the concentration vector at time t_0. The original parameter estimates and parameter covariances are maintained.

It is obvious that a deterioration in performance would exist for each method once this measurement uncertain is introduced. An interesting observation arises not from comparing the new plots (Figures 3-10, 3-11, and 3-12) to the previous results, but rather among one another.

A desired quality of the filter to be used is an adequately estimated variance; where the EKF changes its variance only slightly from the case without noise (Fig. 3-4) to this case (Fig. 3-10), the UKF variance is noticeable larger (Fig. 3-12). Still, both track the parameters relatively poorly.

An interesting observation is seen in the behavior of the Ensemble Kalman Filter in the addition of measurement noise. On certain applications, this filtering scheme exhibits poor tracking of the true values. This phenomenon is most evidently explained by the fact that the Monte Carlo seeding method for Ensemble-based filters can lead to significantly different results for various instances of these estimation methods. That is because such a filtering scheme is dependent on the number and nature of the pseudo-random parameter estimates generated. Notable still is the fact that the one standard deviation still manages, for the most part, to place the bounds on the estimates in a way that still includes the true parameter.

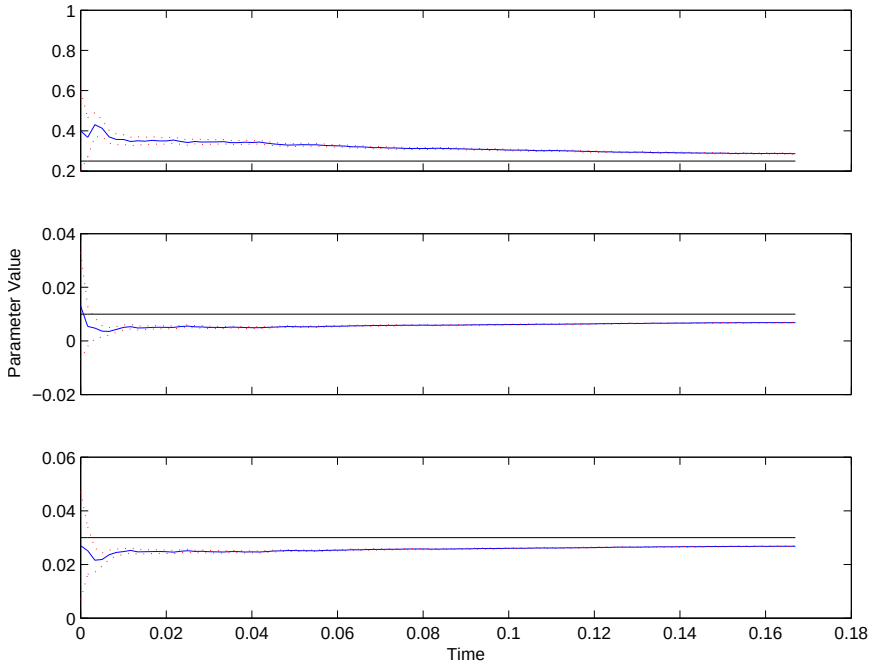

Figure 3-10: Parameter tracking performance of EKF with measurement noise

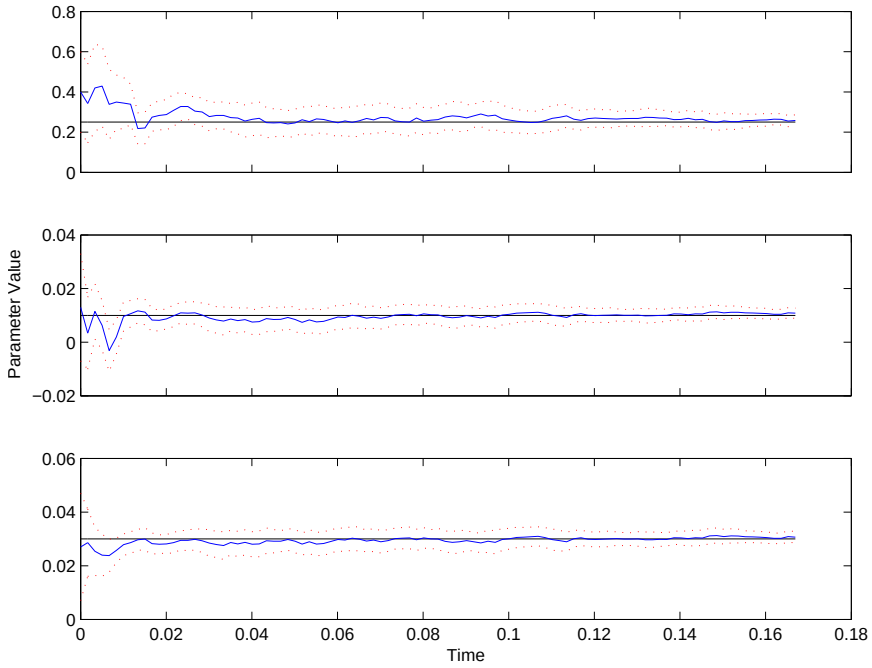

Figure 3-11: Parameter tracking performance of EnKF with measurement noise

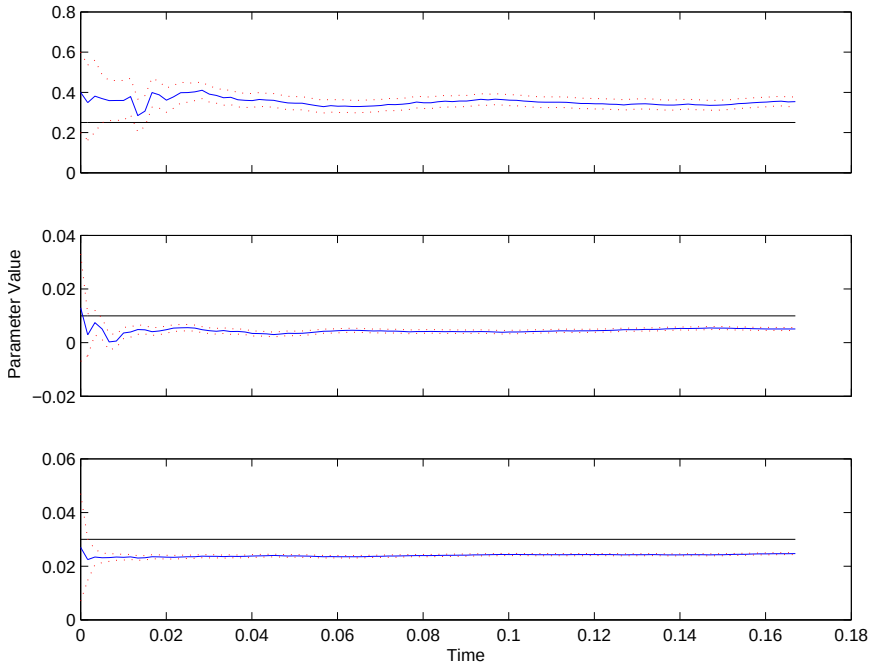

Figure 3-12: Parameter tracking performance of UKF with measurement noise

Process and Measurement Noise

Lastly, for this numerical one-dimensional simulation, process noise is introduced in parameter a_1, thus changing the depth over which the higher diffusivity constant is applied in a stochastic manner as time evolves. The measurement noise is maintained as in the last case with the added actual and estimated process covariance matrix $\mathbf{Q}$ whose sole non-zero value prescribes the variance of a_1 as 10^{-4}. In these last results, the similar performance in the EKF and UKF is observed in Figures 3-13 and 3-15, respectively. In these plots, the one standard deviation finally encloses the true parameter (at least for a_1) as the matrix $\mathbf{Q}$ has been increased.

The most promising methods, however, are still the EnKF and UKF which, when compared to the EKF, these filters maintain an adequate representation of the uncertainty for parameter a_1.

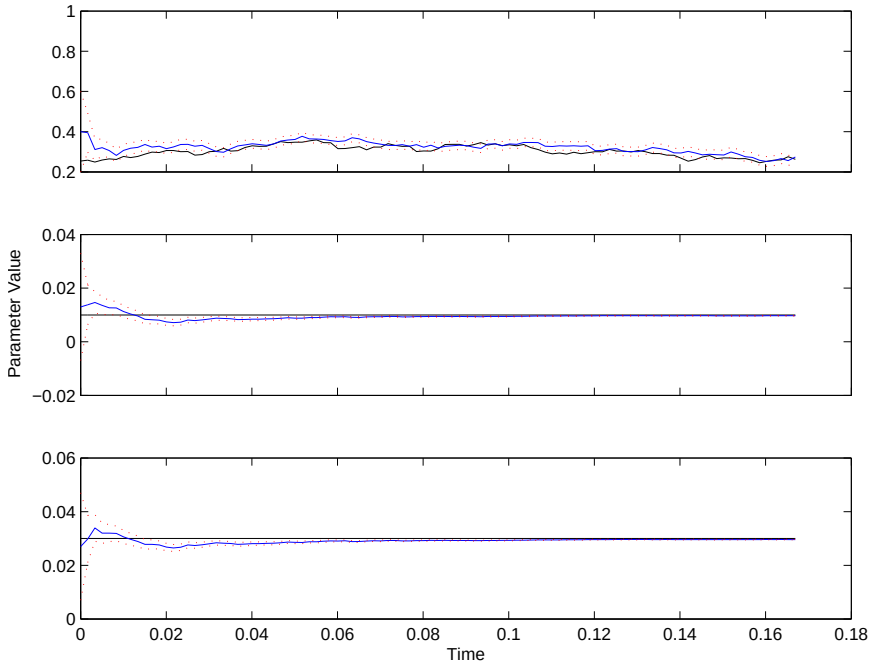

Figure 3-13: Parameter tracking performance of EKF with process and measurement noise

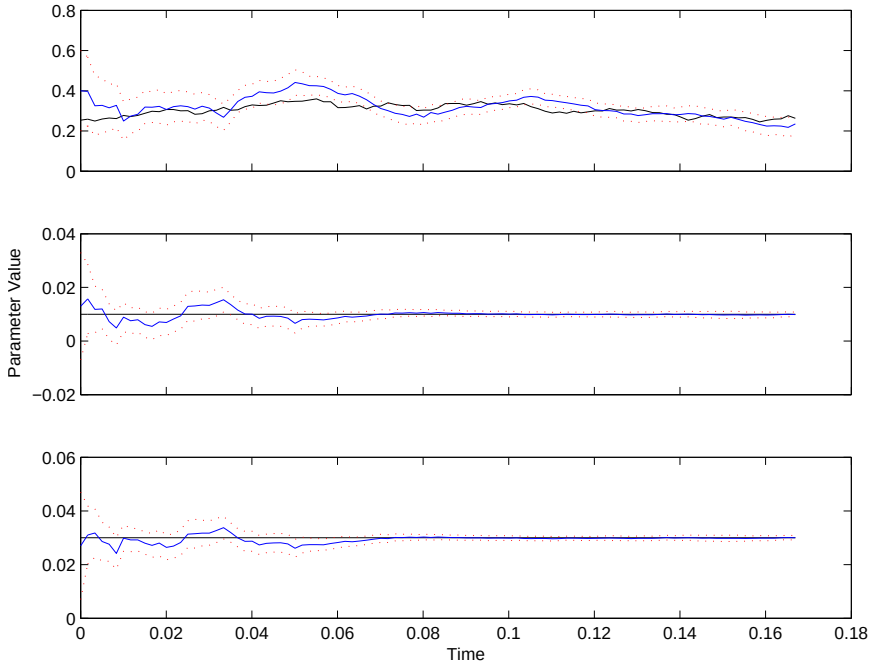

Figure 3-14: Parameter tracking performance of EnKF with process and measurement noise

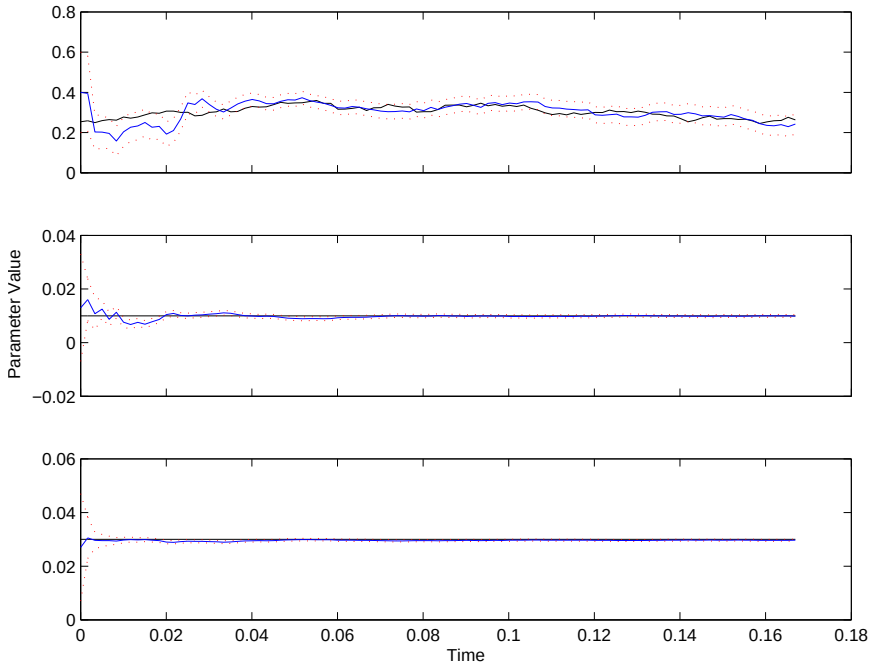

Figure 3-15: Parameter tracking performance of UKF with process and measurement noise

3.6.2 Analytical Test Case

For the parameters governing the analytically derived solutions, the true diffusivity profile was chosen as in Fig. 3-2. The estimates for the true parameters $a_1^t = 0.01$, $a_2^t = -0.04$, and $a_3^t = 0.04$ are $a_1 = 0.02$, $a_2 = -0.03$, and $a_3 = 0.05$ all with *a priori* variances of 10^{-4} (hence standard deviations of 0.01). In the first case, no noise is present, and as with the last test case, the measurement error for the concentration is estimated to have a variance of 10^{-8}. In the case with added measurement noise, the added random perturbations are assigned a variance of 10^{-6} and the same value is used as the estimated noise variance for the filtering algorithm in question. In this section the example with process noise is omitted as the solution was not derived for a stochastically forced system.

Absence of Noise

In the absence of measurement noise, the EKF performs better for this analytical case with quadratic diffusivity profile (Fig. 3-16). This phenomenon is a result of

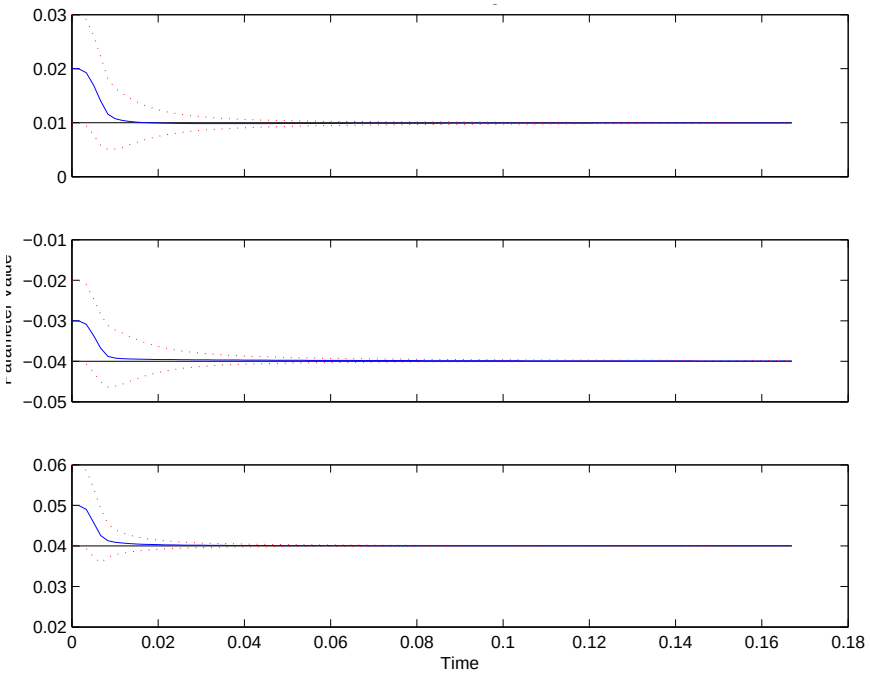

Figure 3-16: Parameter tracking performance of EKF for analytical case without noise

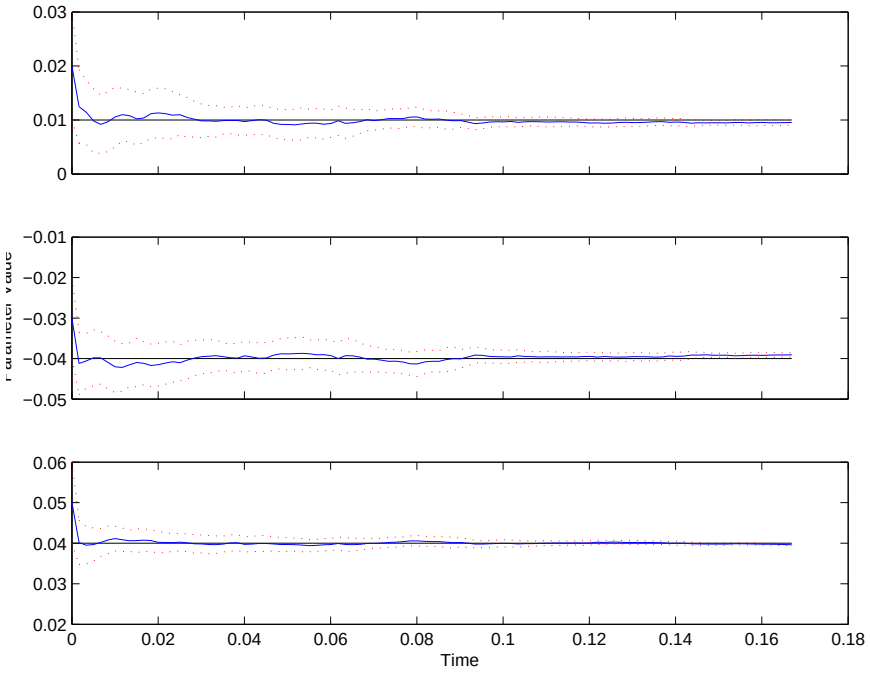

Figure 3-17: Parameter tracking performance of EnKF for analytical case without noise

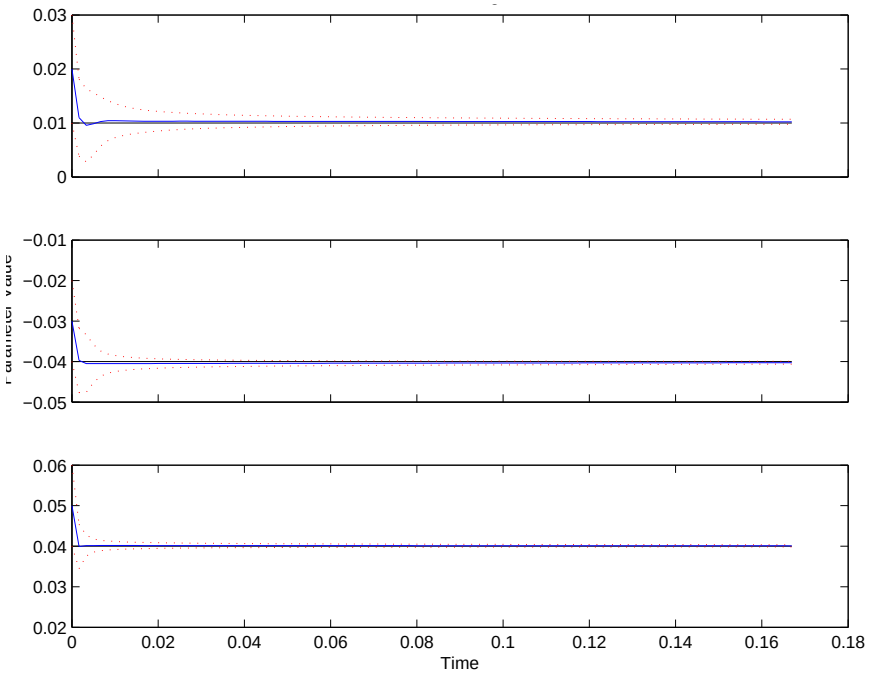

Figure 3-18: Parameter tracking performance of UKF for analytical case without noise

the nature of the problem being inherently less nonlinear than the numerical case with a hyperbolic tangent diffusivity. The UKF, therefore, experiences a similar improvement by the fact that the sigma points chosen to represent the *a priori* estimate and its uncertain is not as drastically distorted by the system dynamics as with the numerical test case. Still, the EnKF performs adequately, maintaining a suitable uncertainty as the estimated values oscillate about the true parameters (Fig. 3-17). Also, as noted earlier, the results from this method are dependent on the stochastic nature of the ensemble chosen.

Measurement Noise

As measurement noise is added to the observations, the EKF performance begins to deteriorate. In Fig. 3-19 the parameters are originally well within the error bounds of this filter, but as the uncertainty continues to be underestimated, the parameter values estimated slowly deviate from the truth. The EnKF and UKF (Figs. 3-20 and 3-21) continue with a similar performance as before, oscillating about the true

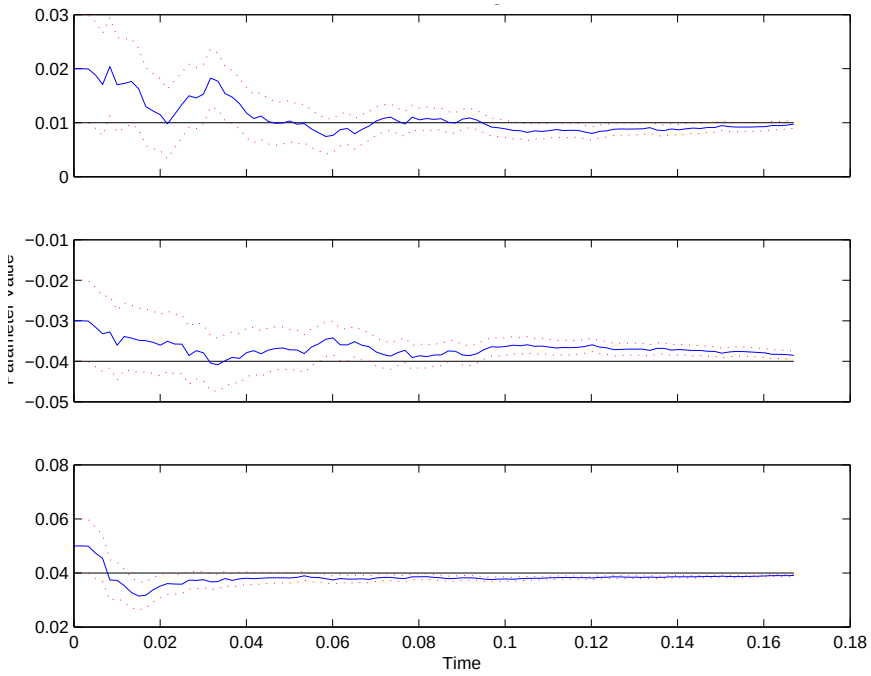

Figure 3-19: Parameter tracking performance of EKF for analytical case with measurement noise

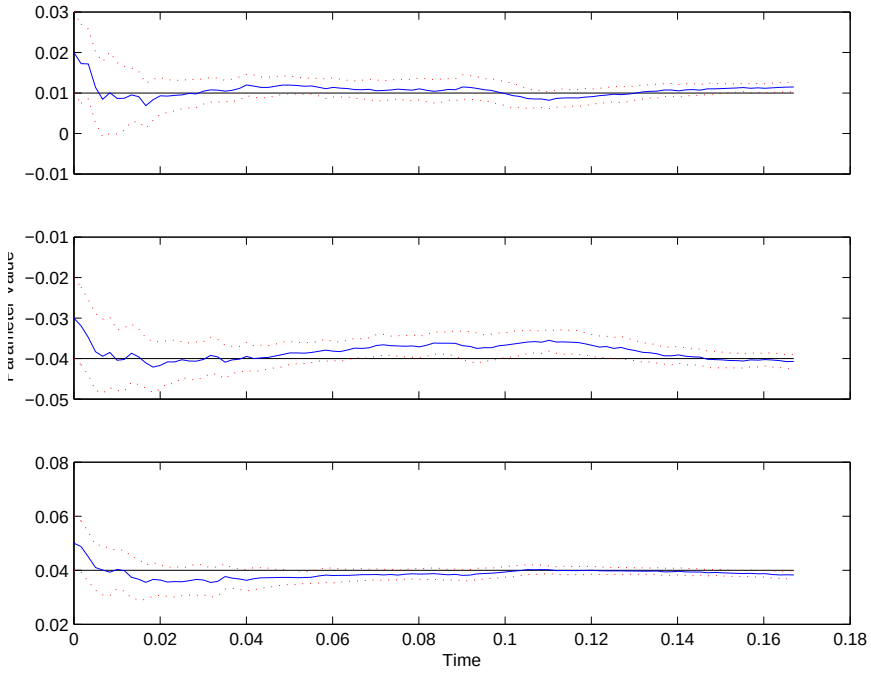

Figure 3-20: Parameter tracking performance of EnKF for analytical case with measurement noise

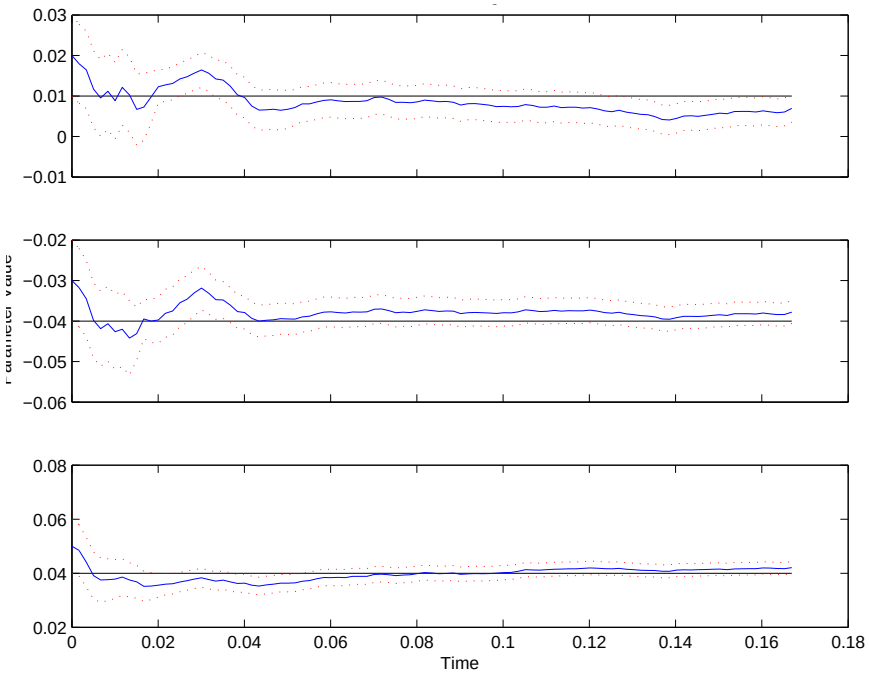

Figure 3-21: Parameter tracking performance of UKF for analytical case with measurement noise

parameters with adequate error bounds. Again it is noted that the UKF performs better in this analytical case due to the less nonlinear choice for the diffusivity profile.

Though computationally costly, the Ensemble Kalman Filter has the best performance of the three recursive methods present above for highly nonlinear systems of equations and in the presence of noise (as seen with the hyperbolic tangent diffusivity profile). Additionally, with the distributed computational capability of the MSEAS computer cluster, generating an Ensemble of simulations is the most conveniently implemented method for this portion of adaptive modeling without requiring the linearization of the complex primitive equation ocean model.

Chapter 4

Realistic Application: Monterey Bay 2006

The use of parameter estimation in adaptive modeling is in the application toward the identification of the most adequate numerical model for the purpose of real time forecasting. Parameter estimation is merely a small portion of the desired self-evaluating model. Such a system should be fully automated and able to identify, quantitatively, the best performing (in the statistical sense) of the available parameterizations. In order to utilize the methods described in Chapter 2 the need arises to quantitatively define the performance of a model. In the three-dimensional HOPS domain, several measures of skill have already been developed. A crucial step in adaptive modeling is to evaluate the performance of various models. The process of launching several runs using Condor on MSEAS computer cluster is discussed, followed by a description of the necessary steps to obtain an adequate comparison of the field estimates with objectively analyzed data and of pseudo-casts with observed casts. The objective is to automate or streamline the process of issuing and analyzing simulations with varied parameters of interest.

In coastal ocean modeling, there are different ways to evaluate the quality of a model setup, parameterizations, or parameter values. First, model outputs can be compared to independent data, either at fixed data points (e.g. a moored time series) or at varying data points (e.g. independent sea surface temperature data). Second,

the model can be started with some data assimilation initially (to achieve sufficiently accurate initial conditions) and then the assimilation stopped. The subsequent model predictions can then be compared to data collected after that point, up until the time the predictability limit is reached.

4.1 Computation Setup

A first task to automate the queuing of a large number of model runs is to generate the required input files with the variables of interest. A template for the input card (containing runtime parameters) is generated in the same format as the original HOPS code input ASCII file ("pe.in" file). This structured file will be read by a UNIX C-shell ("setupjob") routine to assign numerous permutations of possible parameter values. The permitted assignments for each parameter are written in separate ASCII files read by this same C-shell script. The code creates a new directory for each simulation with the chosen combination of parameters and writes the variable inputs to the "pe.in" file in each of the directories created. It also outputs a "day table" with a description of the variations across runs and writes a "submitPE" file which will be required to queue the multiple runs in a distributed fashion over the cluster using the "condor_submit," command. For this high performance computing, a Verari systems cluster of 133 blades on three towers is utilized. With the presence of at least two Central Processing Units on each blade, the simulations can consist of two nested domains and make use of Parallel Virtual Machine (PVM) software for communication between the two fields across processors. In using PVM in conjunction with Condor, simulations may be issued more rapidly with minimal changes to the current procedure. Also, as full or partial model linearization of the complex HOPS/MSEAS software requires a large time investment, and due to the amount of available CPUs, an ensemble based method is appropriate. Such methods also proved to be the best performing in the evaluation undertaken in Chapter 3. For these reasons ensemble methods for parameter estimation will be implemented here. Note that the UKF can be viewed as a type of deterministic ensemble method.

In creating the template for the input parameters, attention is placed on what inputs will be constant and which will vary across each simulation. To compare model runs at data locations a C-shell pre-processing (CPP) option, -Dpeprf, for extracting a profile from the Primitive Equation (PE) solution is called when building the original executable from HOPS Fortran code modules. This executable will be maintained across the input files for each simulation. The option to collect such pseudo-cast profiles requires an ASCII file, "sampling.dat," to specify the time and location from which model output data should be obtained. A MATLAB® code was written to generate such a file from an observed data ".mods" extension files. As the simulation proceeds, model output pseudo-casts are written to their own .mods file. In order to record the Current Meter (CM) and Conductivity Temperature Depth (CTD) data at the location of the MBARI moorings off the coast of California, both types of data format were specified in the "sampling.dat" file. As the HOPS/MSEAS code currently stands, both outputs are written in chronological order in *one* .mods file. As a result, another MATLAB® script was needed to separate the collected data by type for comparison with their respective observation data files.

Full field information on the model domain, when saved, is stored in a "pe_out.nc" file. In writing information to this NetCDF formatted output, the code is limited to a file two gigabytes in size. As a result, either the frequency at which data is saved must be reduced, the number of variables saved has to be truncated, or multiple simulation runs must be issued to collect all the information of interest. To alleviate this problem in the comparison of output files to observation data fields, a new option was introduced into the original HOPS code (Haley, Personal communication). This option consists of specifying a delay in the time at which data is first collected from the start of the simulation, thus permitting waiting to record outputs until after the end of a specified assimilation period.

4.2 Test Case Evaluation

The above techniques have been utilized in evaluating the performance of various tidal model forcings, which are an addition in the MSEAS ocean modeling from the former HOPS code. The selection of the period over which to issue a comparison was carried out based on the amount of available data and the days over which the respective forcings were most asynchronous. These, along with other performance analysis cases considered of interest are listed in the table seen in Fig. 4-1.

	Assimilation options								
	A Partial assim. vs. unassim. data			**B** Full assim. vs. ADCP & Mooring			**C** Model prediction vs. OA data		
Variations across Model runs / Period/ Dynamics	Single Parameter	Multiple Parameters	Tidal Forcing	Single Parameter	Multiple Parameters	Tidal Forcing	Single Parameter	Multiple Parameters	Tidal Forcing
1 Full Period	X	X	X	X	X	X	X	X	X
2 Upwelling/Relaxation	Chk Data	Chk Data	Not Nec.	After 08/06	After 08/06	Not Nec.	X	X	Not Nec.
3 Transition	Chk Data	Chk Data	Not Nec.	After 08/06	After 08/06	Not Nec.	X	X	Not Nec.
4 Tidal Disagreement	Not Nec.	Not Nec.	X	Not Nec.	Not Nec.	X	Not Nec.	Not Nec.	X

Not Nec.	Not Necessarily Informative
Chk Data	Check Amount or Temporal Availability of Data for spliting the OA
After 08/06	After initialization with data collected through August 6th, 2006
X	Possible Test Cases

Figure 4-1: Selection of simulation comparisons

Given the available experiments and the vast components of the current ocean modeling system a large set of possible test cases for performance evaluation were available. In order to choose which test to run, the above table was generated based on various questions of interest. Specifically, the table summarizes the limitations created by the availability of data, the time periods where focus can be placed on known issues in simulation (tides), and the periods of interest for mixing parameterization. The three subdivisions in each of the main columns correspond to parameter estimation of one mixing parameter, estimation of multiple mixing parameters (or possibly parameterizations), and selection of the best tidal forcing (in the statistical sense).

In column A of the table, comparisons are intended to be carried out between simulations with partial assimilation of the available data to the remaining measurements. This procedure requires the splitting of observations into two sets by sub-sampling the available measurements. Then, new objectively analyzed fields (hereafter OAs) must be generated for the two data sets, one for assimilation into the model, and the other

to generate a field for comparison with the model. The difficulty arises in selecting the means by which the data should be split for optimal coverage, and how much data is required for an effective model initialization. For the entire duration of the experiment (first row) this should be feasible. The following categories, with the aim to compare results during different events, will require selecting a time interval which occurs after a sufficiently long assimilation period. The "Chk Data" label identifies the fact that prior to choosing a time period over which upwelling or downwelling occurs for use in this type of comparison, the amount of data available for that time should be examined and deemed sufficient. The cells marked by "Not Nec." indicate combinations of model changes in periods of interest that may not be as informative as other options in the table.

Column B shows the option of running simulations with complete data assimilation of the full OA field already generated containing all of the field measurements. In this case, comparisons will be made between the model output and unassimilated data gathered at the M1 and M2 mooring locations presented in Fig. 4-2. In the case of upwelling and downwelling events, comparisons should probably be made for such oceanic responses occurring after the assimilation of the first Pt. Sur survey (Fig. 4-3), ending on August 6th, for an adequate model simulation initialization.

For column C, model simulations are initialized with objectively analyzed data through August 4th, 2006, after which the models issue forecasts for the fields of interest. Data collected beyond that day are used to evaluate the simulation predictions. The reason for limiting predictions (in column C runs) until after August 4th lies in the fact that data through this date covers the majority of the domain of interest, as shown in Fig. 4-3. This assimilation time period also leaves a significant portion of the data for model evaluation, as presented in Fig. 4-4. The entirety of the first Pt. Sur survey is not included in the assimilated data due to the fact that the time period of interest for the chosen test case, where the tidal disagreement (between old tides and new tides) is most prominent, occurs between the days of August 1st and 7th, 2006.

The intention of the study was primarily for identification of the appropriate

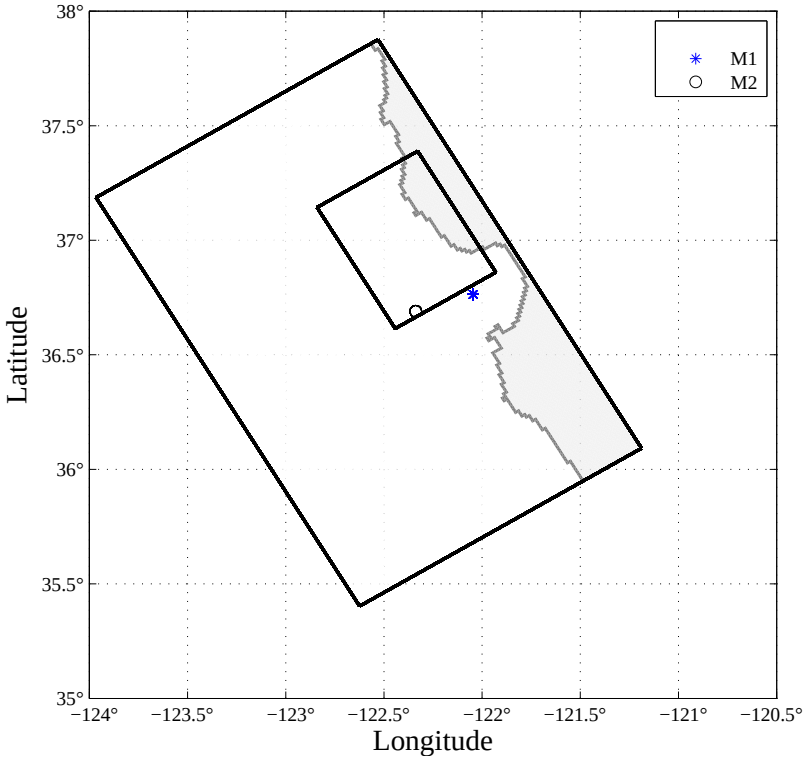

Figure 4-2: Mooring locations

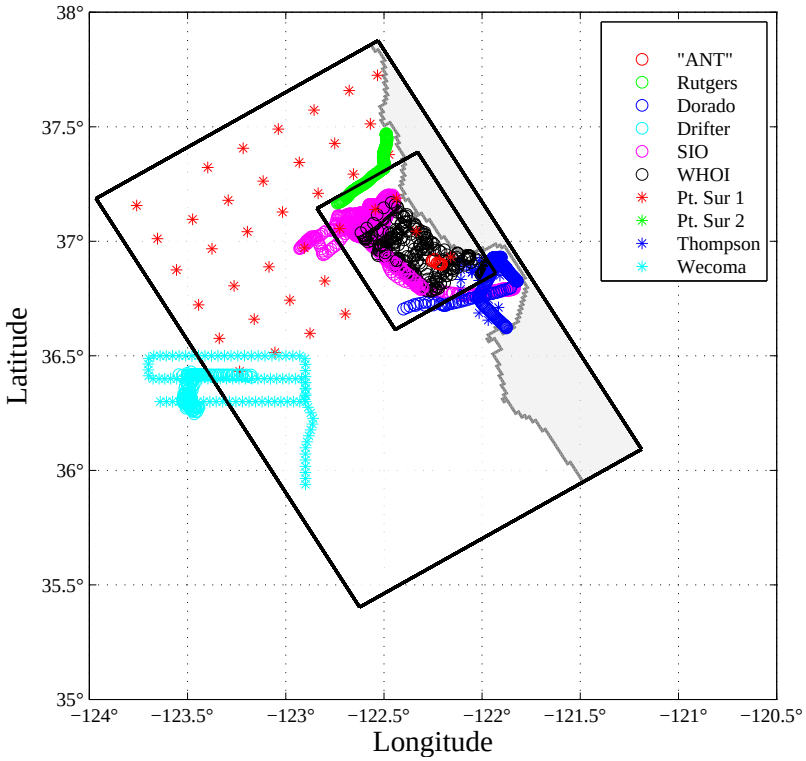

Figure 4-3: Data collected from July 16th through August 4th, 2006
The two outlined rectangles are the nested domains for Monterey Bay.

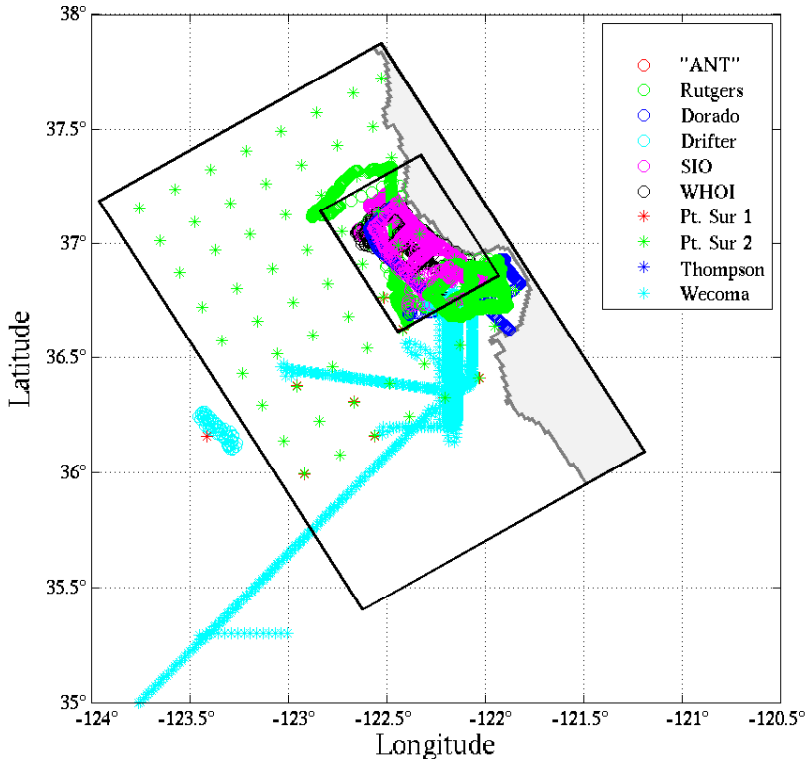

Figure 4-4: Data collected from August 5th through September 14th, 2006

parameters/parameterization of ocean mixing. Of particular interest is the desire to identify the most adequate parameterization for different regimes. Thus, identifying ocean regimes, e.g. upwelling or relaxation (Haley et al., 2008, Lermusiaux, 2007) was also a factor that entered the table in Fig. 4-1. Plots of the wind forcings with the times of expected upwelling and relaxation events, as well as the first appearance of cold surface waters and relaxed/warmed surface layers are presented in Fig. 4-5.

The following skill tools available will be applied in evaluating the tidal estimates in the Monterey Bay region for the 2006 experiment. Specifically, the comparisons reported in the table (Fig. 4-1) as X's in B4 and C4 are carried out, along with a reanalysis comparing outputs from B4 and C4.

A SkillTool for MATLAB® toolbox has been progressively built by Lermusiaux and Haley for comparison of data and models, these have been obtained and worked upon to generate quantitative output for each simulation's performance (Lermusiaux and Haley, Personal communication). Values for the Pattern Correlation Coefficient (PCC) can be obtained from comparing the full model output fields (hereafter re-

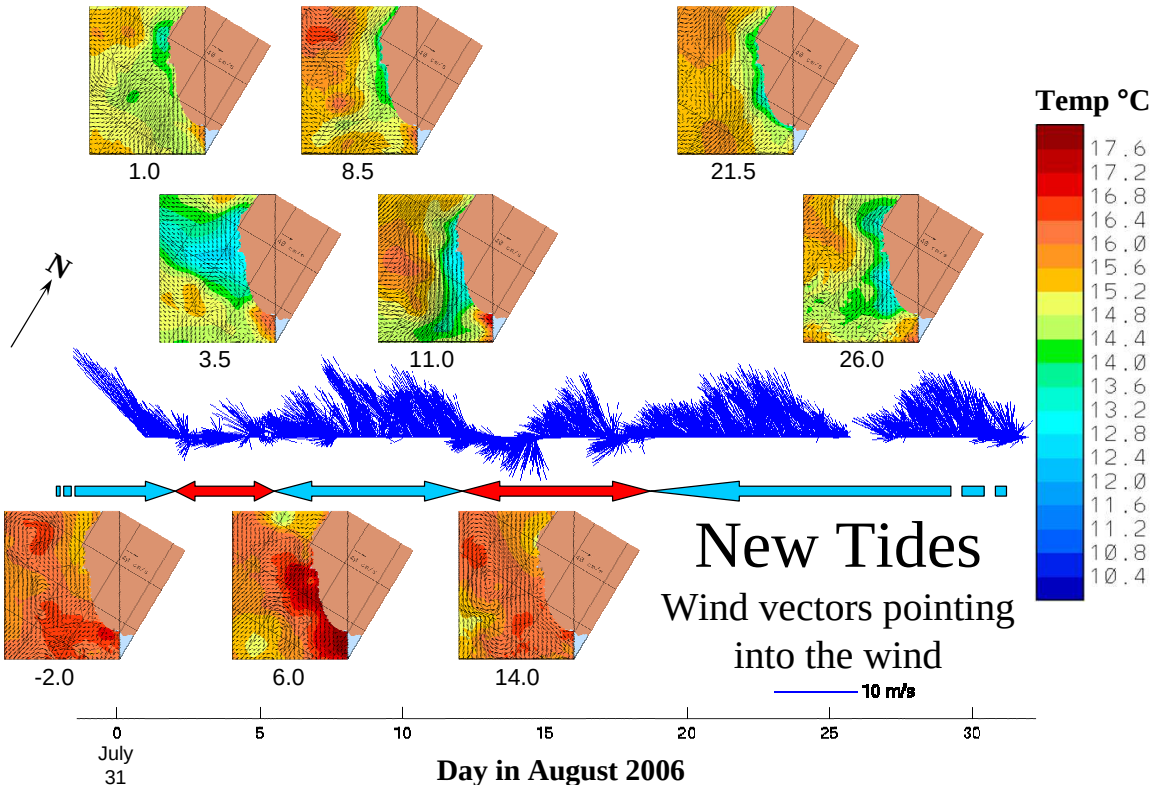

Figure 4-5: Simulation response to atmospheric forcings with new tides where the smaller nested Año Nuevo domain has been rotated (as indicated by true North) so that the coast runs near vertical on average. The blue arrows indicate upwelling favorable winds and red arrows are on average relaxation or transition favorable periods. The numbers below the temperature fields correspond to the day (matching the scale below).

ferred to as Primitive Equation, or PE, fields) to Objectively Analyzed (OA) data fields. From these PE fields, bias and Root Mean Squared (RMS) errors can also be computed. PCC reports the correlation between two fields and as a result can give an idea of how well certain scales are represented by the model. The scales compared by this metric depend on which background value is subtracted from each field, whether mean field, large scale field, assumed background field, etc. Additionally, the OA data mapping is itself dependent on the scales assumed in the true domain. The PCC is a measure analogous to the statistical cross correlation of random numbers. It is computed by the comparison of the product of differences in a horizontal cross-section of the two fields with the product of their standard deviations. The differences and standard deviations are computed with respect to a mean large scale field, thus comparing the mesoscale structure present in each horizontal domain (simulated and observed). This metric can be evaluated on several depth levels and averaged appropriately to provided a volumetric mean. The bias is merely the average difference in the two fields, again, taken over a horizontal section, but the average over several vertical levels will yield an appropriate volumetric estimate. The RMS error is obtained by squaring the bias at every field location, averaging this value over the domain and taking the square root. For a reference on these metrics see Lermusiaux (2007).

Bias and RMS error may also be evaluated at data locations; however, due to the necessity of field information in computing PCC, this metric cannot be computed at single observation locations. The choice to compare data at observation locations as opposed to through the use of an OA field lies in the fact that no assumption is made on the scales, and instead, the nearest simulation output to the true data location is utilized in evaluating errors. The bias error provides a good metric for evaluating disagreement in tides, where accurately capturing velocity direction is important. Whereas for tracer measurements, the RMS error may prove to be a better measure of performance. Both will be important in deciding the skill of various simulations. Optimal system performance will be established through a combination of these measures and PCC when comparing the model to observed data. These metrics may also be utilized in reanalysis, or post experiment evaluation of the model

performance.

A "compmods.m" code is used in evaluating the differences between observed data and pseudo-casts. The data available is best used when combined into one chronologically sorted .mods file, as such, separate observation periods have been concatenated to form a single ASCII file. Where appropriate, multiple data locations were also combined. To explore the data misfits, however, various data locations were kept separate. Unlike the limitations placed on the size of ".nc" files, ".mods" files can be of any size. Their downside is that, as of the moment, ".mods" files can only contain two or three pieces of information in each cast/profile. These are the depth, temperature, and possibly salinity for CTD data, and the depth, zonal, and meridional velocities for CM data. In this particular analysis the MBARI mooring data (M1 near shore at 122.046W, 36.764N and M2 at 122.338W, 36.670N offshore) were kept separate. Their locations are presented in Fig. 4-2. Model data agreement offshore is more difficult to achieve for the tidal model and added scrutiny will be placed on the errors at this location. In the next section, then, a set of simulations utilizing higher resolution tidal estimates is compared to a prior simulation run with lower resolution tides. The skill is then evaluated with the above described methods.

4.3 Numerical Setup

In utilizing the higher resolution tides, parameters in the primitive equation model must be altered to reflect the fact that smaller scales will now be represented and no longer be treated as sub-grid tidal mixing. A small ensemble of runs was therefore created with a select list of parameter values. The day table for the runs with the new, higher resolution tides is presented in Fig. 4-6. For comparison, the corresponding parameters for the coarse tidal forcings are presented in their "pe.in" card format in Fig. 4-7. In the day table, for the Domain (Dom) column, "OMG" refers to the large domain while "SmO" refers to the small, nested domain. The next two columns refer to parameters in Card 11 of the pe.in file. For bottom friction (BotFrc), the first value is the grid e-folding scale reported in number of vertical grid spacings

```
This directory tests a series of runs using Oleg's new inverse tides.  This
series bases the HOPS tidal velocities on Oleg's transports with the revised
B-grid Continuity.  AWACS parameterizations have been employed.

+-----+---+---------+---------+---+----+-------+-------+
| Dir |Dom|  BotFrc |  CstFrc | DT|WCFL| CDTID |TDMXFRC|
+-----+---+---------+---------+---+----+-------+-------+---------
|EVH01|OMG|1.5,14400|1.5, 7200|100| yes| 0.0025|  10.0 | cflT at TS 16782
+-----+---+---------+---------+---+----+-------+-------+  during day 20, Aug16
|EVH02|SmO|1.0,28800|3.0, 7200|100| yes| 0.0025|  10.0 |  (19.4236 days reached)
+-----+---+---------+---------+---+----+-------+-------+---------
|EVH03|OMG|1.0,14400|1.5, 7200|100| yes| 0.0025|  10.0 |
+-----+---+---------+---------+---+----+-------+-------+  during day 38, Sep03
|EVH04|SmO|1.0,14400|3.0, 7200|100| yes| 0.0025|  10.0 | cflT at TS 32288
+-----+---+---------+---------+---+----+-------+-------+---------
|EVH05|OMG|1.0,28800|1.5, 7200|100| no | 0.0025|  10.0 |
+-----+---+---------+---------+---+----+-------+-------+  during day 30, Aug26
|EVH06|SmO|1.0,28800|3.0, 7200|100| no | 0.0025|  10.0 | cflT at TS 25327
+-----+---+---------+---------+---+----+-------+-------+---------
|EVH07|OMG|1.5,14400|1.5, 7200|100| yes|0.00125|   5.0 |
+-----+---+---------+---------+---+----+-------+-------+  during day 37, Sep02
|EVH08|SmO|1.0,28800|3.0, 7200|100| yes|0.00125|   5.0 | cflT at TS 31801
+-----+---+---------+---------+---+----+-------+-------+---------
|EVH09|OMG|1.5,14400|1.5, 7200|100| yes|6.25e-4|   2.5 |
+-----+---+---------+---------+---+----+-------+-------+  during day 26, Aug22
|EVH10|SmO|1.0,28800|3.0, 7200|100| yes|6.25e-4|   2.5 | cflT at TS 22196
+-----+---+---------+---------+---+----+-------+-------+---------
|EVH11|OMG|1.5,14400|1.5,14400|100| no | 0.0025|  10.0 |
+-----+---+---------+---------+---+----+-------+-------+  during day 37, Sep02
|EVH12|SmO|1.0,28800|3.0,14400|100| no | 0.0025|  10.0 | cflT at TS 31568
+-----+---+---------+---------+---+----+-------+-------+---------
|EVH13|OMG|1.5,14400|1.0,14400|100| no | 0.0025|  10.0 |
+-----+---+---------+---------+---+----+-------+-------+  during day 36, Sep01
|EVH14|SmO|1.0,28800|2.0,14400|100| no | 0.0025|  10.0 | cflT at TS 30667
+-----+---+---------+---------+---+----+-------+-------+---------
|EVH15|OMG|1.0,28800|1.0,14400|100| yes|6.25e-4|   2.5 | cflT at TS 16832
+-----+---+---------+---------+---+----+-------+-------+  during day 20, Aug16
|EVH16|SmO|1.0,28800|2.0,14400|100| yes|6.25e-4|   2.5 |
+-----+---+---------+---------+---+----+-------+-------+---------
```

Figure 4-6: Day table for a small ensemble of simulations run with the new tides

(parameter DBTFRC in Fig. 4-7) and the second value is the temporal e-folding scale reported in seconds (parameter TBTFRC); the coastal friction parameters (in CstFrc) have the same meaning and correspond to DCSFRC and TCSFRC in Fig. 4-7. Column "DT" in the day table is the time step size in seconds, corresponding to the parameters reported in Card 2 of the input file. The following column labeled as "WCFL" states whether or not the model should check the CFL condition in the vertical (w corresponding to the vertical velocity) and corresponds to IOPT(6) in Card 12 where a 1 at this location would indicate "no" in the day table (disabling the vertical CFL check) and a 0 would imply "yes" (maintaining the vertical CFL check). Coincidentally, in all the runs issued for this test, none terminated as a result of reaching the CFL limit in the vertical direction. The tidal friction coefficient (CDTID) is similar in form to a drag coefficient and is found in Card 10 of the input file along with the limit placed on tidal mixing (TDMXFRC). Along with the large change in these last two parameters between using the old tides and new tides, the

```
1    NFIRST  NLAST  D0START  NNERGY  NTSOUT  NTSI  NMIX   NCON  NTDGN
       1     34560  13944.0    432     432    1     10     0      0
2    DTTS    DTUV     DTSF  (seconds)
      100     100      100
3    MIXVEL  MIXTRC  MIXZTD (mixing scheme: int. vel., tracers, vort|uhat)
       1       1       1
4    NORD NTIM NFRQ (int. vel., tracers, [vort|uhat], [trans|press] [& Pbndy])
     4 1 3   4 1 1   2 1 1   2 1 1   2 1 1
5    AM      AH
     1.E9    2.E7
6    AIDIF  FKPM  VVCLIM   WVMIX  FRICMX  FKPH   VDCLIM   WDMIX
     1.0    0.04  100.0     50.0   100.0  0.005   100.0     20.0
7    MLDOPT  MLDVAL   MLDMIN   MLDMAX   EKFAC   MCOEF   NCOEF  WSDFAC
        1    1.0E+4   1.0E+2   4.0E+3   0.081  0.3507  -9.578  0.0004
8   MXSCAN   SOR     CRIT      ACOR     TOLABS   TOLPCG  CGSTAT  FILLIN  GRELTL
     10000  1.625  1.0E-12  0.33333333  1.0e-25  1.0e-3     2      15    1.0E-8
9    CDBOT
     2.5e-3
10    CDTID     MTDDPTH   TDMXFRC   TDMXFAC   SADV
      0.125       30.0      400.0     200.0    0.2
11   DVBRLX   TVBRLX   DTBRLX   TTBRLX  DCSFRC  TCSFRC  DBTFRC  TBTFRC
      -1.0     -1.0     -1.0     -1.0    1.5    7200.0   1.5    7200.0
12   (1)  (2)  (3)  (4)  (5)  (6)  (7)  (8)  (9)  (10)   IOPT(I)
      7    7    7    7    0    1    0    2    2     0
13 (PSI) (Vt) (Vi) (Vb) (Vg)  (W@V) (W@T) (w@V) (w@T)  (KE) (VOR)   IOUT(01-11)
     0     1    0    0    0     0     0     0     0      0    0
14 (T) (S) (RHO) (Buoy) (MLD) (Vtide) (Stide) (Ttide) (TrcBal) (Err) IOUT(12-21)
    1   1    0       0     0      0       1       0       0       0
15   NLEV     LEV(nlev) in ascending numerical order
     30     1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
            21 22 23 24 25 26 27 28 29 30
16   TITLRUN (a80):  Application title.
ASAP:  BF(1.5,7200) CF(1.5,7200) ShP(211) ShUh(211) (run 492)
17   OUTNAME (a80): output PE fields NetCDF file name.
pe_out.nc
18   NRGNAME (a80): output PE energy and diagnostics NetCDF file name.
pe_nrg.nc
19   TRKNAME (a80): output Langrangian trajectories NetCDF file name, if any.
/dev/null
20   INPNAME (a80): input inital/boundary conditions NetCDF file name, if any.
/projects/asap/PE_initial/2006/Aug19/OMG30/Ic/pe_ini0727.nc
21   FRCNAME (a80): input forcing fields NetCDF file name, if any.
/projects/asap/PE_forcing/2006/Sep10/OLG01/pf0727_0912.nc
22   ASSNAME (a80): input assimilation fields NetCDF file name, if any.
/projects/asap/PE_initial/2006/Oct25/OMG30/AssL/pi_ass.nc
23   APARNAM (a80): input assimilation parameters ASCII file name, if any.
```

Figure 4-7: Partial input parameter card for the large domain with old tides. Note that the corresponding variables from the columns in Fig. 4-6 starting at column three and moving right are: DBTFRC and TBTFRC; DCSFRC and TCSFRC; DTTS, DTUV, and DTSF; IOPT(6); CDTID; TDMXFRC. Note also that the card for the smaller nested domain for the old tidal model differs only in the e-folding scales used in the larger domain shown here. Variables DBTFRC, TBTFRC, and DCSFRC are set to 1.0, 28800, and 3.0, respectively. In the process of changing tidal models, MTDDPTH was also increased to 50.0 for new tides.

maximum depth over which to apply tidal mixing was altered from a value of 30 meters (as seen in Card 10 as MTDDPTH in Fig. 4-7) to a depth of 50 meters for the new tides. Overall, the run with parameters nearest to the simulation using older tides consists of EVH01 and EVH02. In this domain pair, the only differences other than the maximum tidal friction and tidal mixing depth are the bottom friction temporal e-folding scale in the large domain and tidal friction coefficient in both domains. The first, TBTFRC, has been increased from 7200 to 14400, and the second, CDTID, has been reduced from 0.125 to 0.0025.

4.4 Results

After running on the MSEAS computer cluster for approximately nine hours, the output files for the simulations were created and the analysis of the performance could then be evaluated based on the previously discussed skill tools. As was reported in Section 4.2, the tidal disagreements in the barotropic forcings between the old and new tidal simulations were well out of phase between August 1st and 7th. For this reason, the velocity components were extracted from simulations using old and new tidal forcings at the M2 location 30 meters in depth and are shown in Figs. 4-8 and 4-9. These simulations were carried out with full assimilation, from the start of the experiment until its end, and therefore correspond to the B4 series of Fig. 4-1. The phase disagreement increases noticeably from the start (day 0 corresponding to July 27th, 2006) to a period between days 5 and 11 (August 1st through August 8th) where they are most out of synchronization. Finally, the forcings overlap once again around day 14, or August 11th. At the largest disagreement in phase, the tidal forcings are shown for the old tides in Fig. 4-10 and new tides in Fig. 4-11. It is evident in these figures as to why the M1 tidal gauge located in Monterey Bay (see Fig. 4-2) would contain less information of value than M2 further offshore.

For clarity, the following figures portraying the simulation performances have been divided into two sets of plots for each of the saved output fields, the large and small domains. This reduces the clutter and also allows the comparison of the (slightly

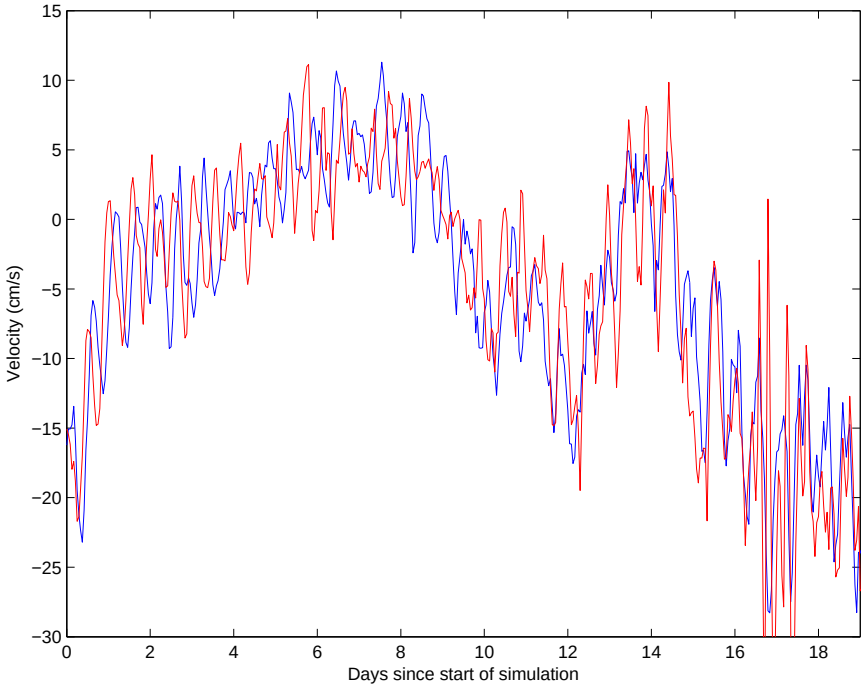

Figure 4-8: M2 30 meter Zonal velocities
The blue line is the zonal velocity obtained with the simulation using the older tides, the red curve is the measure of zonal velocities at M2 with the new tidal forcing for the EVH11 parameters (see Fig. 4-6).

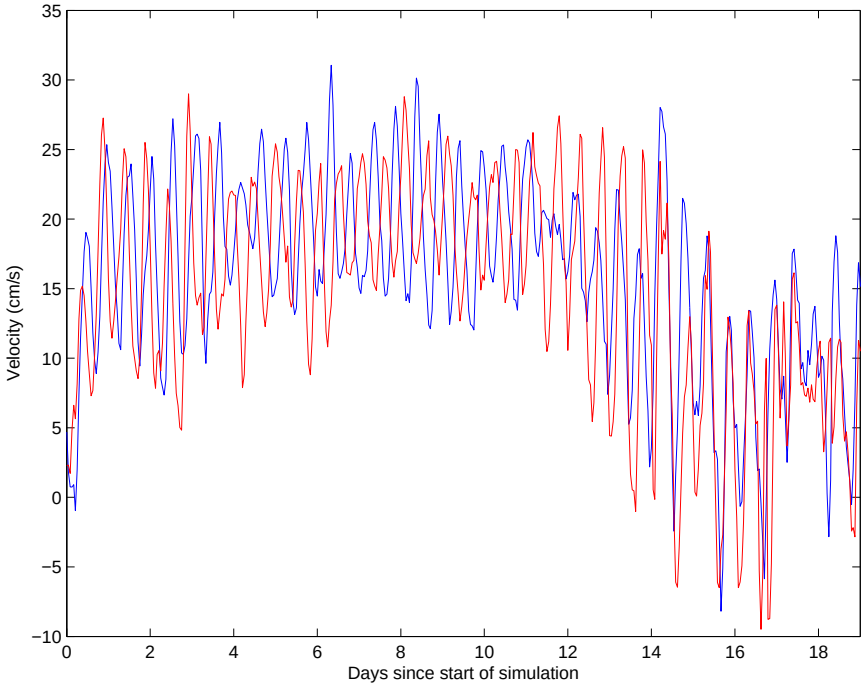

Figure 4-9: M2 30 meter Meridional velocities
Colors are indicative of the same simulations as the previous figure.

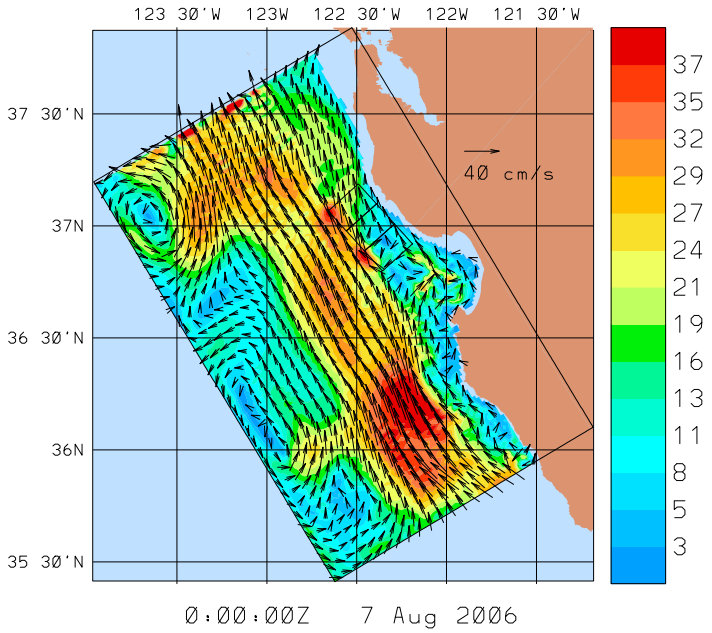

Figure 4-10: Old tide 30 meter velocities for August 7th

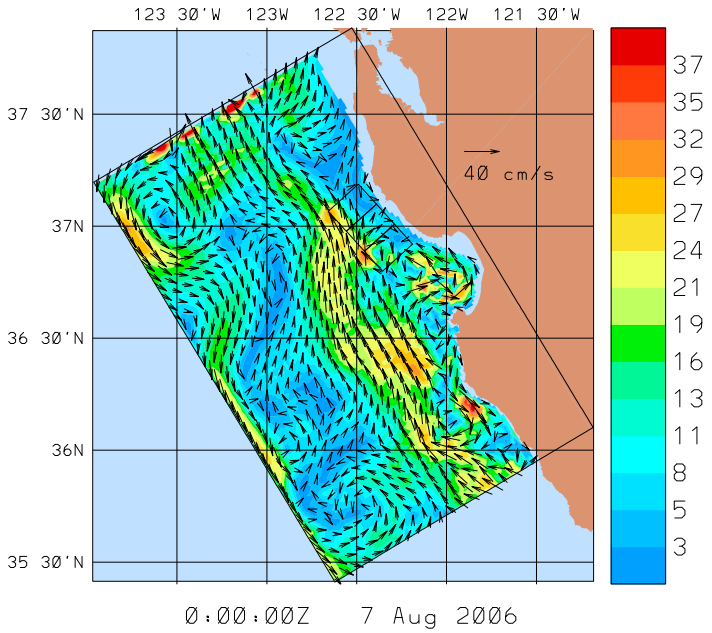

Figure 4-11: New tide 30 meter velocities for August 7th

varied) performance of the larger and smaller nested domains with their counterparts. For the correlation coefficient, a horizontal average of each field was subtracted as a background large scale, thus allowing the comparison of the mesoscales present in each field in question. A higher value is desired for the PCC, whereas, for the other two metrics (RMS and bias error) smaller values are sought. Another factor to keep in mind is that the predictability limit for HOPS simulations in the region is of approximately one week (Haley et al., 2008). The skill metrics are extended to ten days in the following figures as, when observing tidal disagreement, errors are expected to creep in as time proceeds and the forcings become more out of phase.

4.4.1 Performance Evaluation by Comparison to Objectively Analyzed Data

In the following plots, performance results for the large Monterey Bay domain are first presented, followed by the nested Año Nuevo domain. The distinction is made across each figure by utilizing a solid line for the large domain and dashed lines for the smaller domain. Additionally, when looking for the simulation with the old tides as compared to the new simulations, old tides are marked by a circle and new tides marked by quadrilaterals. The time scale in the graphs corresponds to days since the start of simulation. Data assimilation was discontinued at the end of day eight (August 4th, 2006) and therefore the next ten days are reported (simulation days 9 to 19) starting at the beginning of August 5th and ending at the end of August 14th (beginning of August 15th).

Temperature

In Figures 4-12 and 4-13, the performance of this output tracer field retains fairly similar quantities across all simulations for the first three or so days after ending assimilation, but as the forecast continues, the slight superiority of the simulation forced by old coarser-resolution tidal fields appears as the PCC stays highest for the majority of this run. As is apparent in Fig. 4-12, the old tidal forcings in this tracer

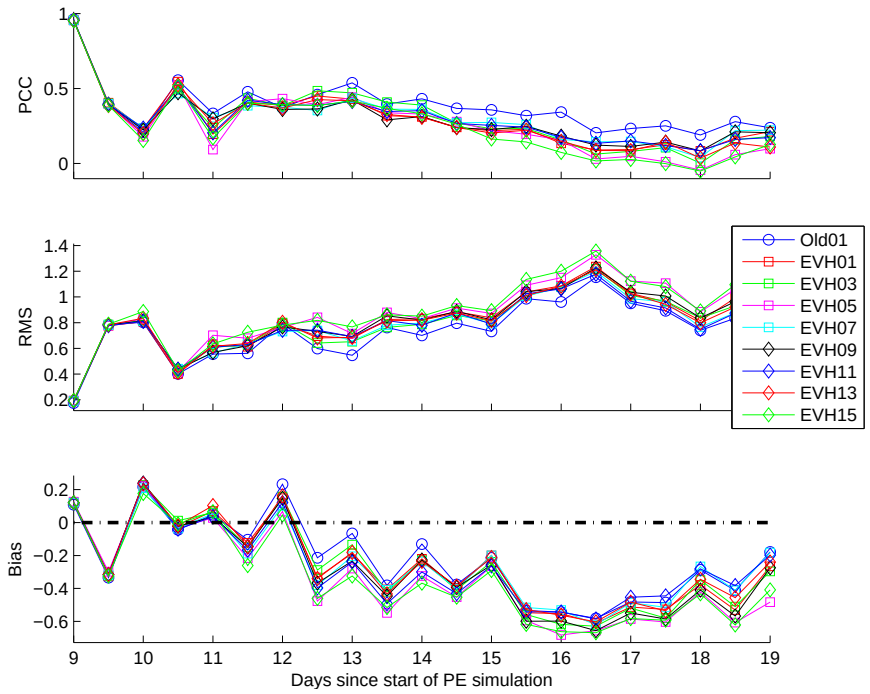

Figure 4-12: Error statistics for temperature in large domain
The thick dash-dotted line corresponds to the line of zero bias in the last subplot.

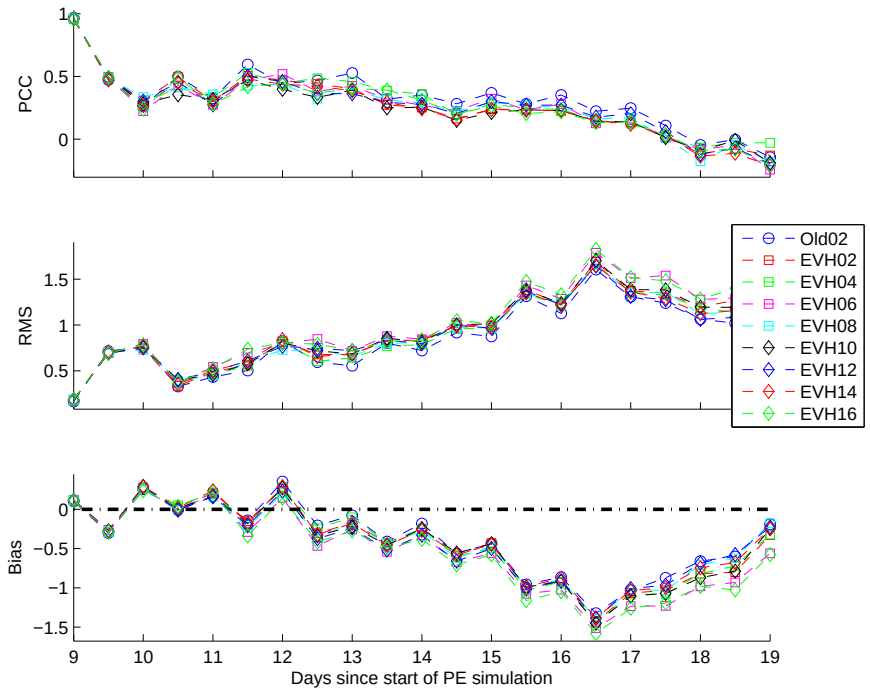

Figure 4-13: Error statistics for temperature in small domain

field dominate all three performance metrics with the absolute value in the bias and the RMS error remaining the lowest for the longest duration as compared to the new simulations. And similarly, for the small domain (Fig. 4-13) until the point where the PCC becomes negative in the tenth day of the forecast. Though oscillations are seemingly consistently present for all runs, these oscillations appear mainly in the top 30 meters of the water column, and are not as prominent or as frequent at greater depths. The volume averaged value reported in these plots thus contain evidence of these fluctuations in the bias from surface errors, yet are smoother as a result of including the deeper levels. The better old tides are only neared in performance by the EVH11 and EVH12 nested runs, followed closely by EVH13 and EVH14. Early in the analysis, EVH03 and EVH04 also show promising results, which deteriorate as time proceeds, surpassed by the prior mentioned around five days after the end of assimilation.

Salinity

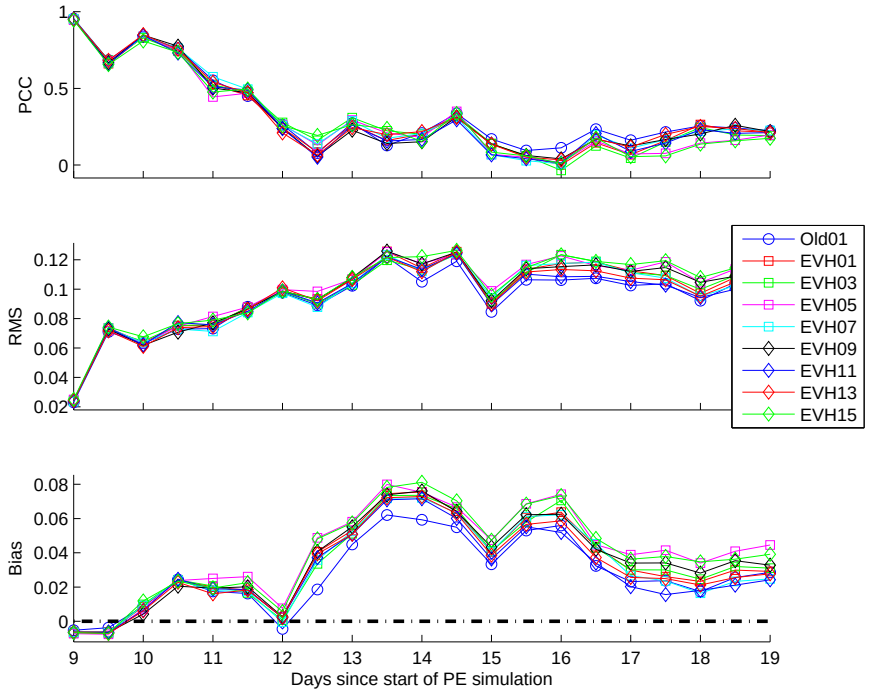

Figure 4-14: Error statistics for salinity in large domain

Salinity is a variable with less fluctuation in part due to the daily warming cycle

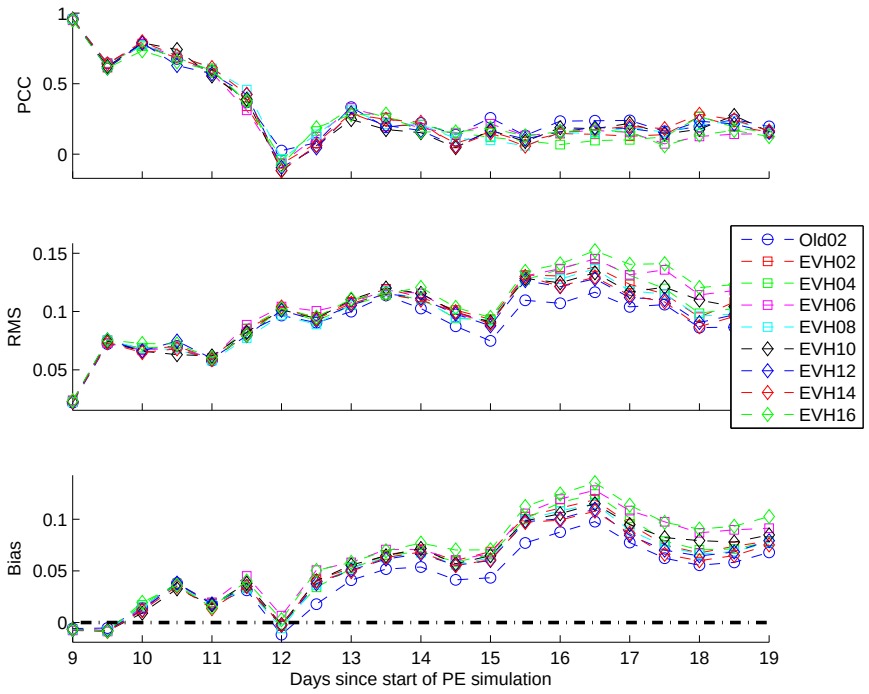

Figure 4-15: Error statistics for salinity in small domain

and maybe to internal tides/waves near the thermocline. As a result, when compared to temperature, it is slightly more difficult to use, at least visually, in acknowledging the better performing model setup. This output variable follows the same general trend as the previous temperature plots in that, for the first few days, all models have similar performance, and as time passes the original run with the older tidal forcings gives better results across the various metrics. In these plots, particularly Fig. 4-14, the match in performance by EVH11 is clearly seen in the RMS and bias for simulation days 16 through 19. And again, in Fig. 4-15 the nested Año Nuevo domain for the old tidal forcings (Old02) is a clear winner for simulation days beyond day 12, again neared most closely by EVH12 and EVH14.

Velocity

Note that the values reported in the following plots may not be as reliable as the information presented in Temperature and Salinity plots. This is because the velocity fields computed by each simulation are in part based on geostrophy, thus, are dependent on the above tracer variables. Furthermore, the OA velocity fields

have been obtained by the addition of the barotropic (external mode) terms and the baroclinic (internal mode) velocities, these are then compared to each simulation's output total velocity field.

Zonal Velocity

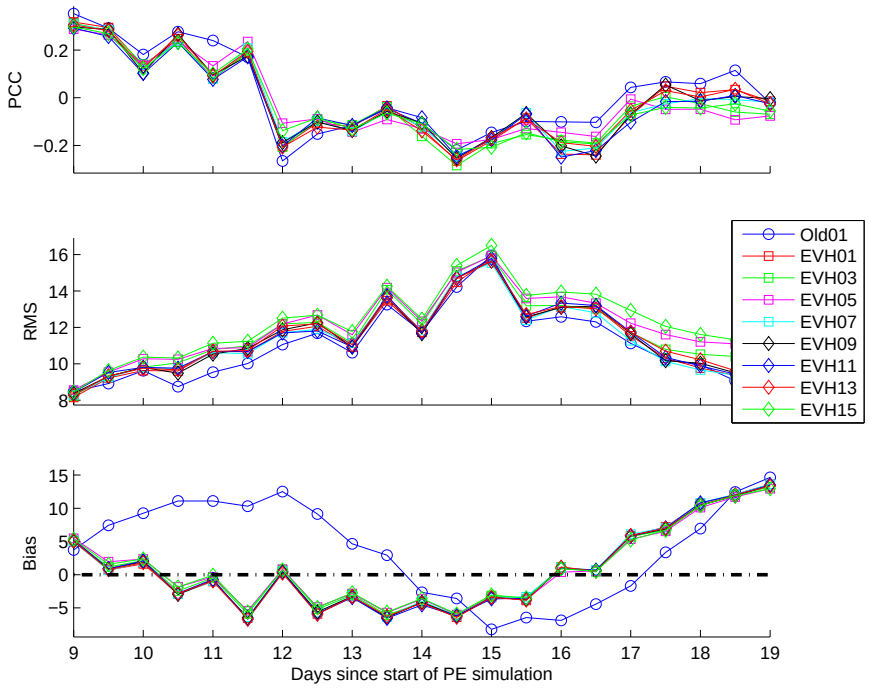

Figure 4-16: Error statistics for zonal velocity in large domain

For the first few days, in the large domain (Fig. 4-16), until about midway into simulation day 11, the old tidal forcings outperform the simulations using new tides with regard to PCC and RMS. Referring back to Fig. 4-8, this is approximately the time when the two forcings begin to line up in phase. They remain in agreement from simulation day 14 until around midday of the 16th simulation day, where at this point, the zonal velocity in the new simulation receives a jolt in amplitude leading to a relative improvement in the old simulation among the group, even in the bias. The bias plot shows another story. Simply looking at the values, it would appear that for the majority of the ten day forecast, the old simulation is a dramatically poorer representation of the zonal velocities. However, for the first five days of the prediction, when the data is compared every twelve hours, a noticeable change in the quality of the forecast is seen between noon and midnight for runs with new tides. On the other

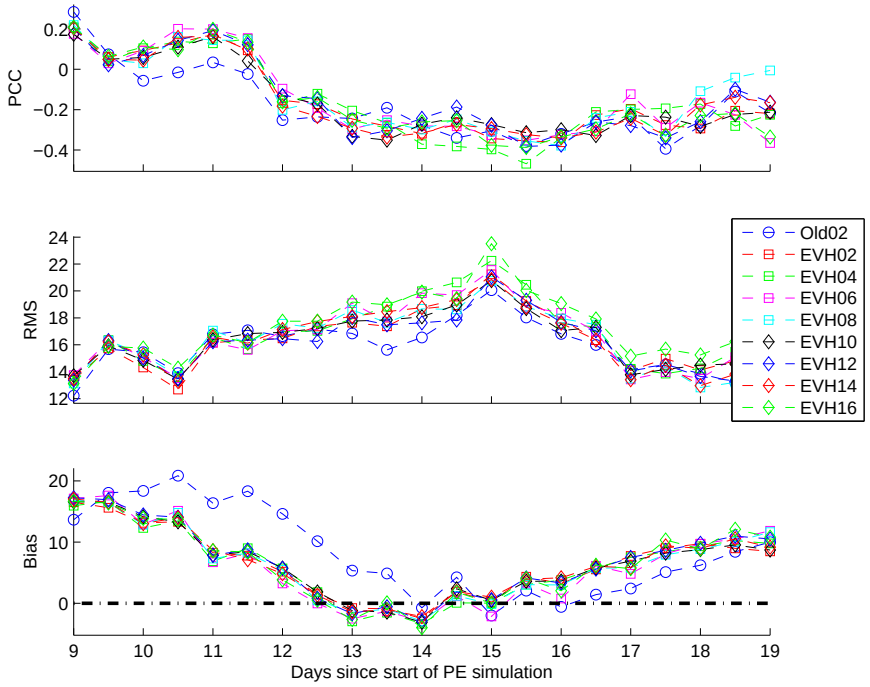

Figure 4-17: Error statistics for zonal velocity in small domain

hand, the observed bias in the old tides remains fairly steady. This observation may suggests a mismatch in the new tidal phases and the OA data and warrants further investigation. As for the small domain (Fig. 4-17) results suggest little as to which parameters or which tides perform best. PCC and RMS values remain relatively close in value, whereas the bias again shows favorable results in the new tides for the first five day forecast, but improved performance by the old forcings in the last four days of the ten day prediction. Such little amount of distinction should be expected in this Año Nuevo domain as, closer to the coast, tidal disagreement will have a less important effect on the observations in this region.

Meridional Velocity

Again, it is difficult to establish, by comparison of the PCC in the large domain seen in Fig. 4-18, which simulation is better. However, interesting observations can be made with regard to the RMS and bias in these plots when compared to the meridional velocities from Fig. 4-9. For the case of the RMS error, the simulation with old tides appears as a weaker contestant in the first four days of the simulation forecast (through simulation day 12). As the phases line up again, around the 14th day of

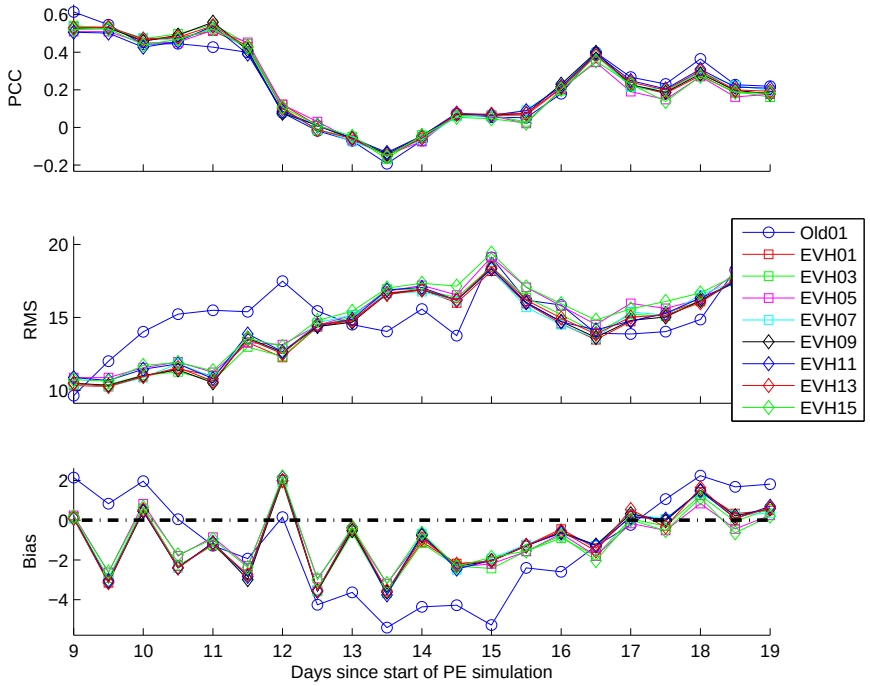

Figure 4-18: Error statistics for meridional velocity in large domain

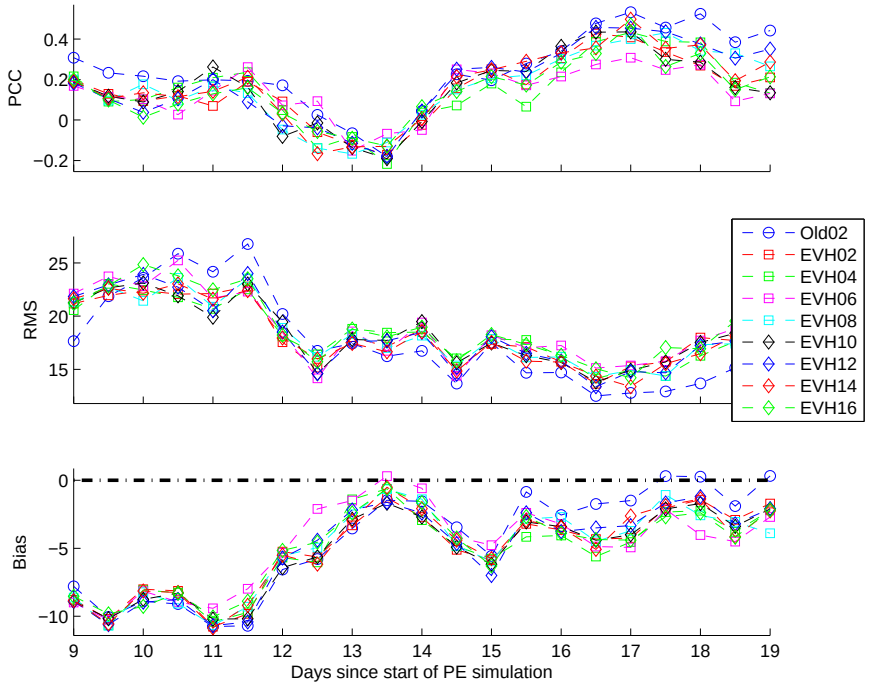

Figure 4-19: Error statistics for meridional velocity in small domain

simulation and through the end of the ten day prediction, all simulation seem to come into agreement. Additionally, the bias indicates an alternating best performer between old and new tidal forcings in the first two days (when the two barotropic tidal models are in highest disagreement) with the old model overestimating the meridional velocities, and the new model often underestimating these. A period where a large disagreement in the amplitude of the meridional velocity oscillations follows from around day 11 until day 14 of the simulation (Fig. 4-9). During this time fluctuations in the performance of the new simulations (those with the large oscillatory amplitude) increase, while the old simulation consistently, and almost uniformly, underestimates the meridional velocities. This is followed by all runs improving in bias error performance around day 16 up until 19 of the simulation, finishing with velocity estimates that are in phase. What this may suggest is that the phases of the old tides (at least the diurnal lunar tidal component) are in better agreement with observations, yet the amplitudes of these may not be in their best agreement. To address the possible causes for these results, it would be beneficial to carry out further skill metric analyses with more frequent simulation outputs or with other tidal estimates.

Once more, the distinction in performance for the various simulations in the smaller nested domain shown in Fig. 4-19 are not as prominent as those in the big domain, most likely resulting from the fact that the horizontal fields here are more sheltered from large tidal velocities offshore.

4.4.2 Performance Evaluation through Reanalysis

Here, another comparison is carried out to establish the performance of each model simulation. The prediction carried out by each model run is no longer compared to the OA data, rather, it is now compared to the same run with full data assimilation. This scenario is equivalent to comparing model real-time predictions (column C4 in Fig. 4-1) to model behavior incorporating validation data (column B4 in Fig. 4-1).

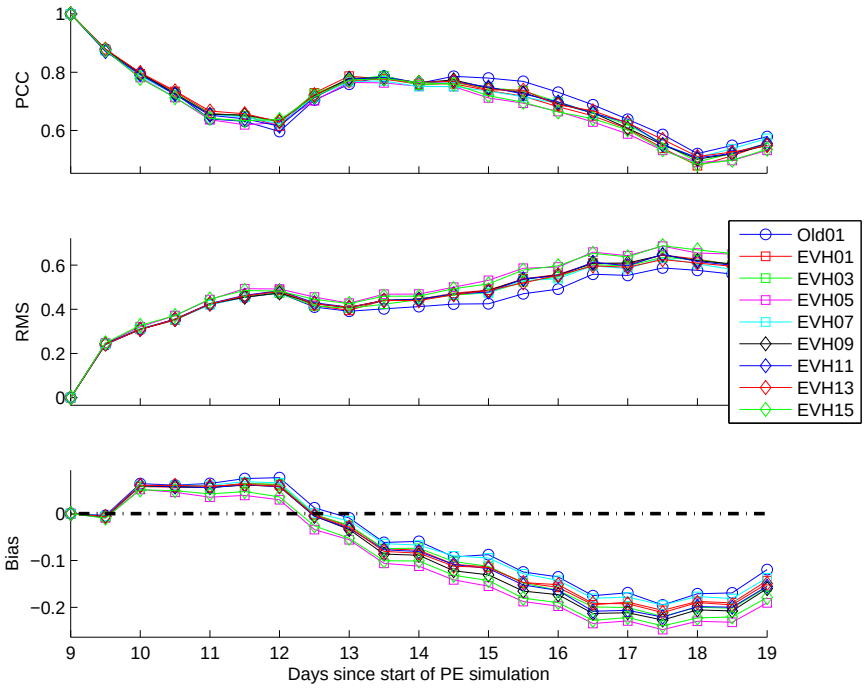

Figure 4-20: Error statistics for temperature in large domain

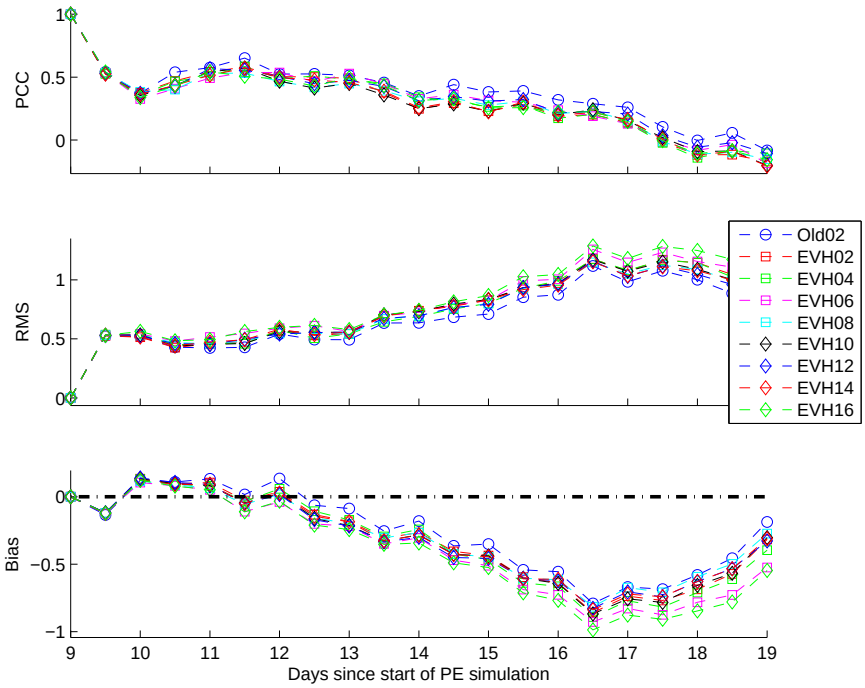

Figure 4-21: Error statistics for temperature in small domain

Temperature

The temperature performance plots for this comparison show a decrease in the error statistics and an increase in the correlation between the model prediction and the reference, as compared with the figures from the previous section. From Figures 4-20 and 4-21, the distinction between the performance of the simulation utilizing the old tidal forcings and those using the new tides more clearly identifies the superior performance of the old tides, at least when tracking its own predictions based on assimilated data. Additionally, this comparison shows the effect models have when incorporating objectively analyzed observation data. By comparing the curves in these figures (more noticeable with the large domain) to Figures 4-12 and 4-13, a clear reduction in the twelve hour oscillations is seen.

Salinity

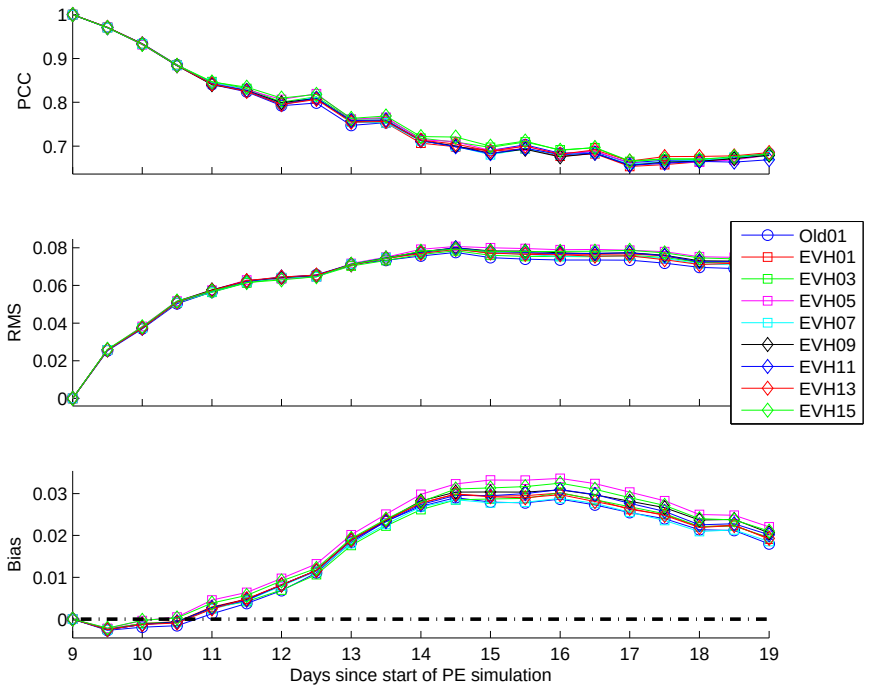

Figure 4-22: Error statistics for salinity in large domain

The performance in representing the salinity fields also suggests the first simulation using the older tidal forcings is better than those driven by new tides. Although in

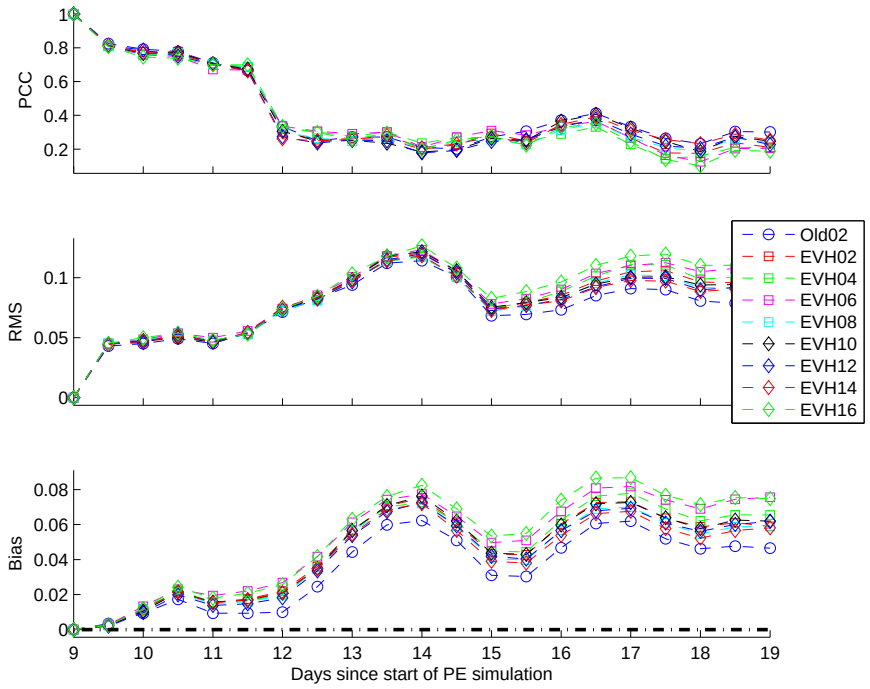

Figure 4-23: Error statistics for salinity in small domain

both domains the PCC is nearly indistinguishable from one run to the next, the two other metrics used for the model evaluations reveal a more accurate result is to be expected with the older barotropic tidal field. One anomaly that may warrant further investigation is the sudden loss in correlation between the predicted and reanalysis fields after three days without assimilation in the small domain (Fig. 4-23). A possible explanation for such a drastic drop may be tied to a change in data availability.

Velocity

Zonal Velocity

Again a large improvement in the metrics through comparison with models with complete assimilation is obviously apparent. The bias errors in Figures 4-24 and 4-25 are greatly reduced from those seen in Figures 4-16 and 4-17. Still, however, oscillations are apparent, specifically in the larger domain, which would potential suggest the existence of misrepresentation of tidal phases for all model runs, especially between days 13 and 18 of the simulation.

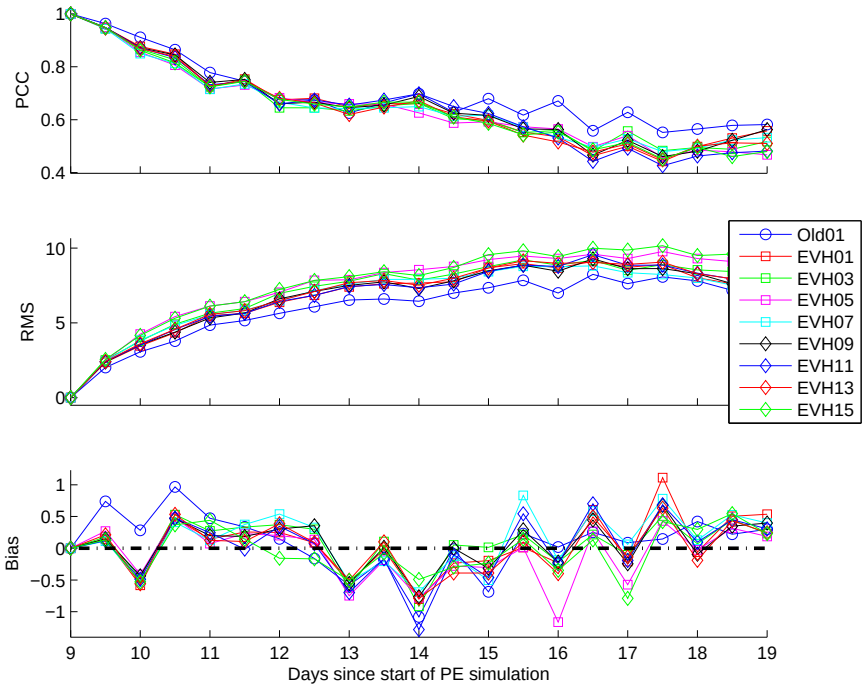

Figure 4-24: Error statistics for zonal velocity in large domain

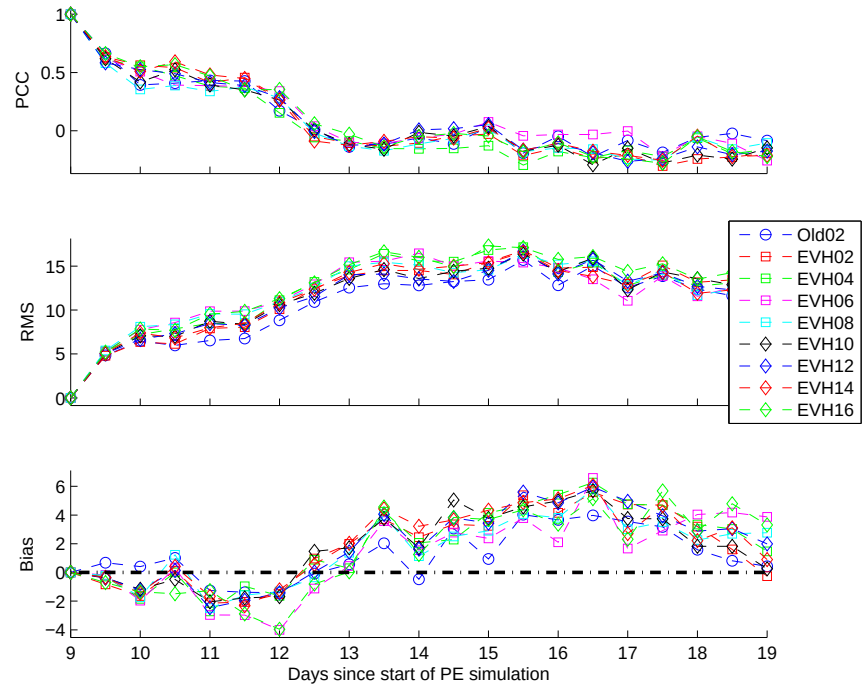

Figure 4-25: Error statistics for zonal velocity in small domain

Meridional Velocity

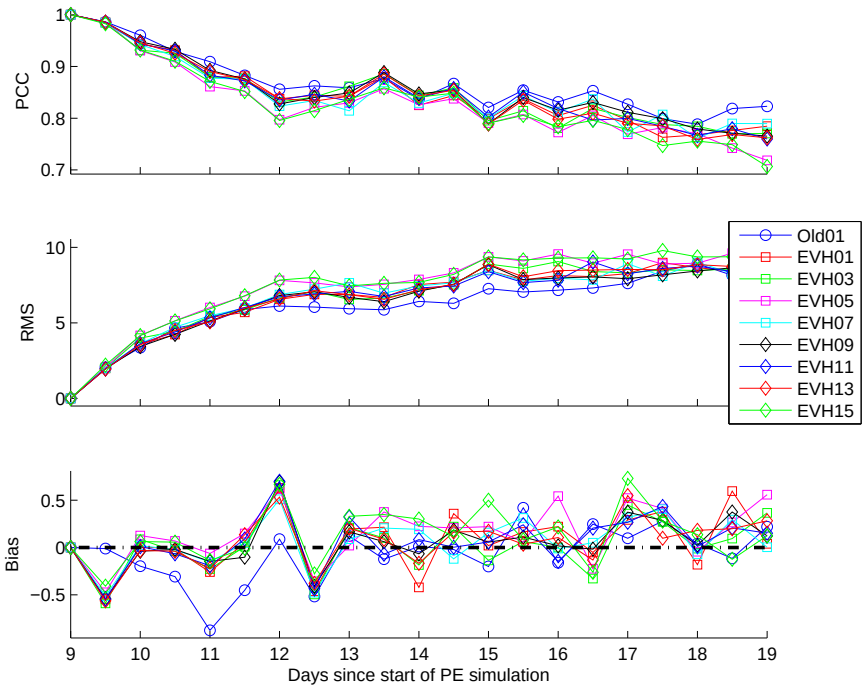

Figure 4-26: Error statistics for meridional velocity in large domain

In the meridional velocities, again, the story is the same as for the zonal measurements. Oscillations are once again present in the bias, unlike Fig. 4-24, however, Fig. 4-26 does not show the same similarity in phase. The effect of the various tidal parameters across the simulations using new tides is more eminent in this figure, with the two runs EVH11 and EVH13 having the smallest bias error during the length of the simulation. In the results for the small domain, Fig. 4-27, noticeable changes between new and old tidal forcings show that the older barotropic tides do not represent the velocity field in the small domain adequately between the days of August 8th through August 11th (simulation days 12 through the end of 15). Although the PCC increases again after this date, so does the bias error. These two effects occur just around the time the amplitude in the meridional velocities spike (Fig. 4-9) which may potentially have resulted from the assimilation of data in disagreement with the model prediction.

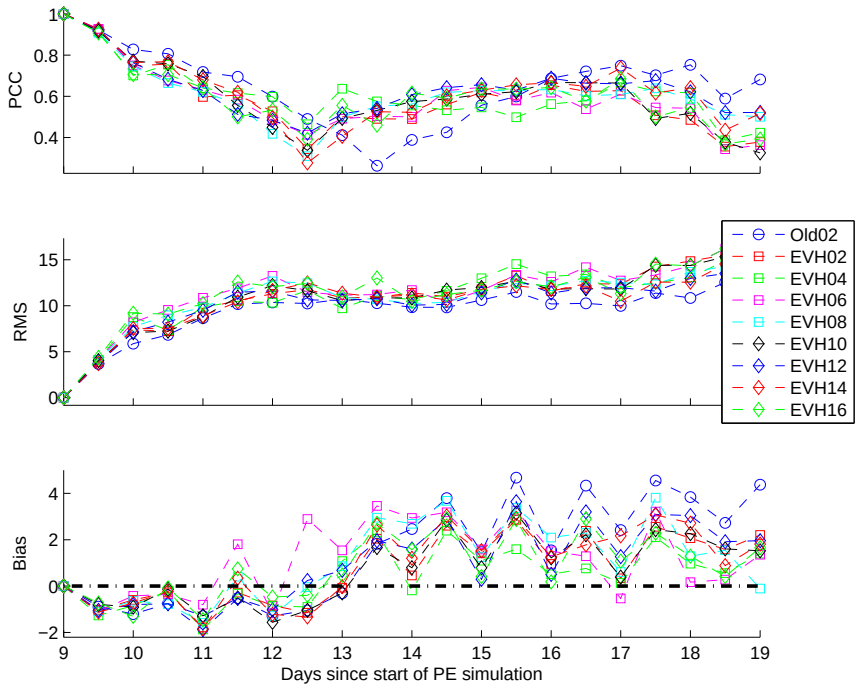

Figure 4-27: Error statistics for meridional velocity in small domain

4.5 Model Data Misfits at Mooring Locations

As presented in the first part of this chapter, another means of computing the performance metrics is by evaluating the errors at data location. In this section comparisons are carried out at the M1 and M2 mooring locations previously discussed. The data collected at these locations are CM and CTD data. As a result of the geographic location of these sensor within the region, tidal effects are expected to have a greater impact on the M2 mooring further offshore than on the M1 mooring. Therefore, more significant changes in the model data misfits are anticipated at the M2 location for the various model runs.

In the following plots, the M1 pseudo-casts are extracted from the large Monterey Bay domain, as they are not contained within the smaller nested domain. Also, since the M2 mooring location is so near the boundary separating the small Año Nuevo domain from the large domain, the pseudo-casts from either domain are nearly identical, as a result, M2 pseudo-casts are only reported from the Año Nuevo domain. The bias errors for the simulation with old tides are presented along with those for

the new runs EVH11-EVH12 and EVH13-EVH14. Though EVH01-EVH02 is most similar to the simulation using the old tidal forcings in terms of the model runtime parameters, it does not perform as well as the two previously mentioned runs where the temporal e-folding scales have been increased to maintain agreement between the large and small domains. As all new simulations show similar results, only the better performing are shown next.

4.5.1 M1 Current Meter Data Error Analysis

The bias in the velocities shown in these figures (Figs. 4-28, 4-29, and 4-30) for the M1 mooring location reveal, above all, the well established performance of the old tidal forcings. The models utilizing the new tidal estimates do not perform nearly as well as the simulation with older tides close to shore. With the new tidal model, data misfit plots show what appear to be fairly evenly spaced peaks every six hours (that is, positive peaks in the misfit every 12 hours, interspersed with negative peaks) which seem to indicate a major phase disagreement M2 lunar diurnal tides.

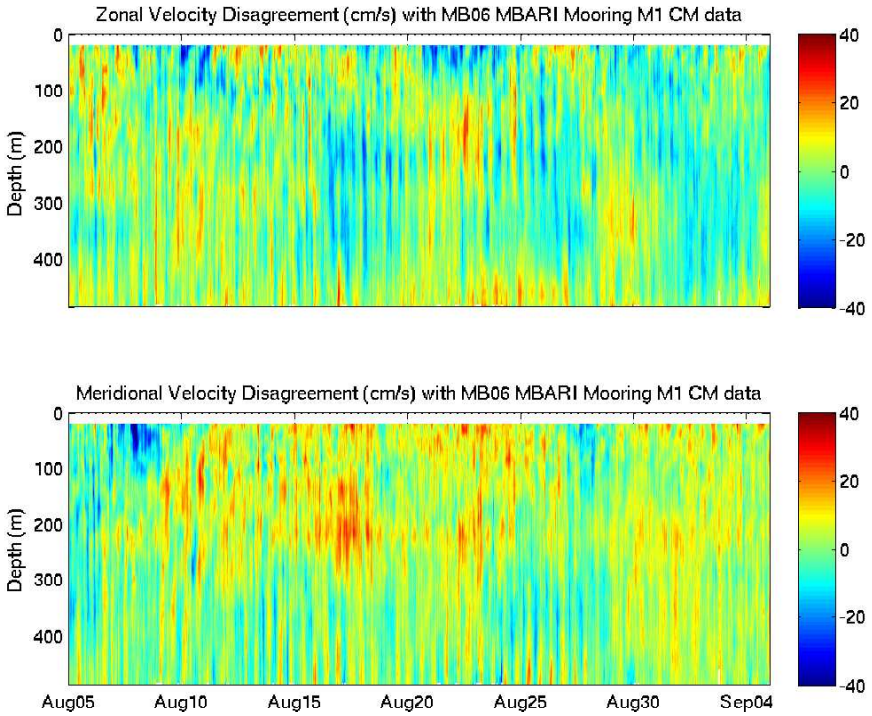

Figure 4-28: Error in M1 current meter data with old tides

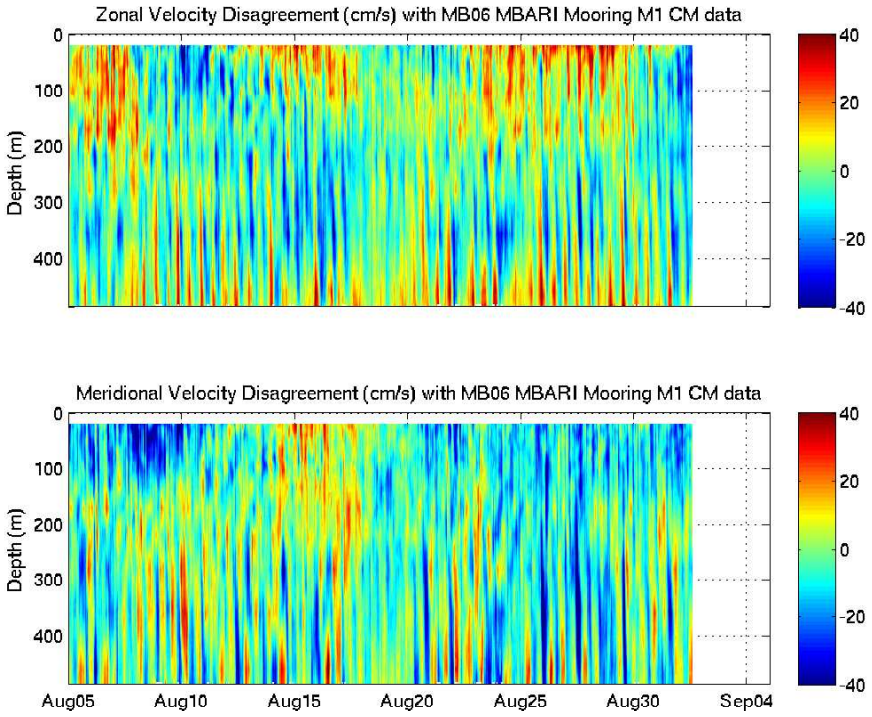

Figure 4-29: Error in M1 current meter data with new tides EVH11

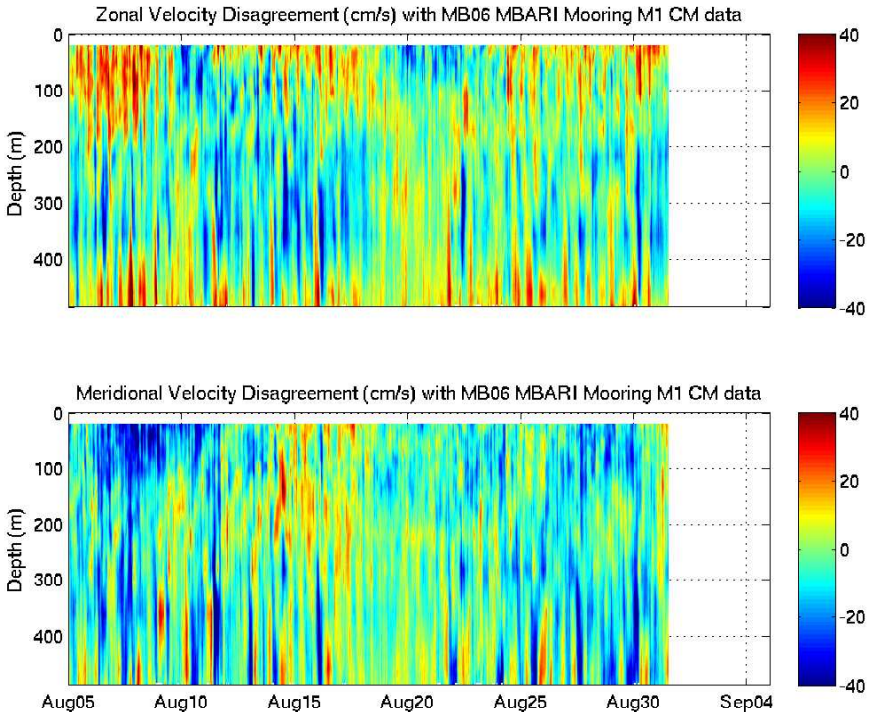

Figure 4-30: Error in M1 current meter data with new tides EVH13

4.5.2 M2 Current Meter Data Error Analysis

The mooring further offshore shows misfits with much less disagreement for the two tidal forcings available. The bias seen in the first few days after the end of the initialization period in Fig. 4-31 reveal less striations than in the errors seen in Figures 4-32 and 4-33. Such an observation would indicate a better representation of the higher frequency components of the barotropic forcings in the older representation of tidal velocities, or at least an initial agreement in phase which deteriorates after around five or six days. It should be noted in these figures that the predictability limit is reached by the third time axis label (August 15) which is where the comparisons stopped for the quantitative metrics present in the previous Section 4.4. After this date, there is a noticeable increase in the disagreement in meridional velocity, as for the next 16 days, all runs overestimate this component of velocity.

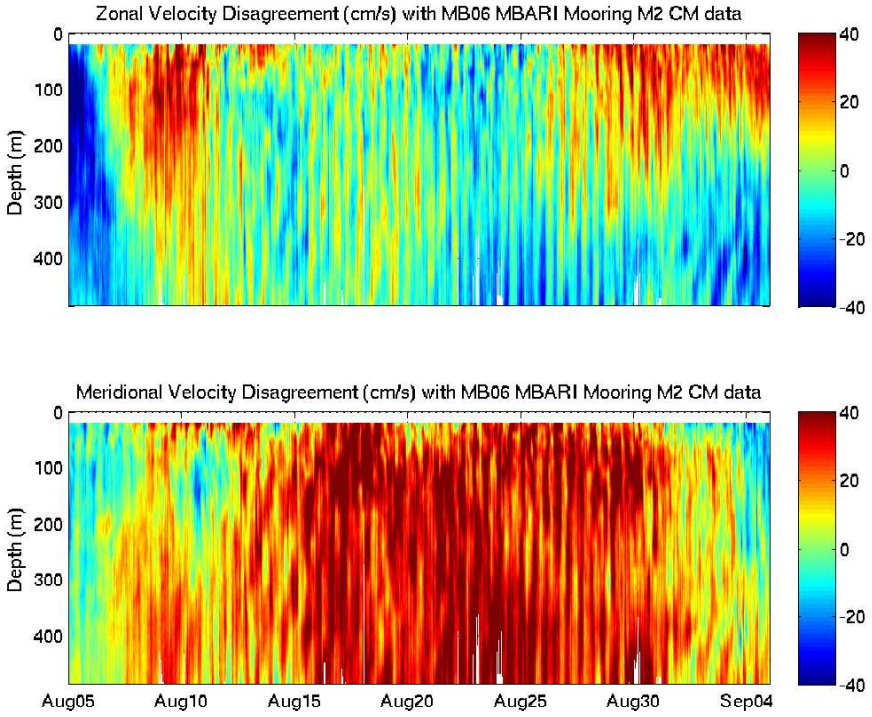

Figure 4-31: Error in M2 current meter data with old tides

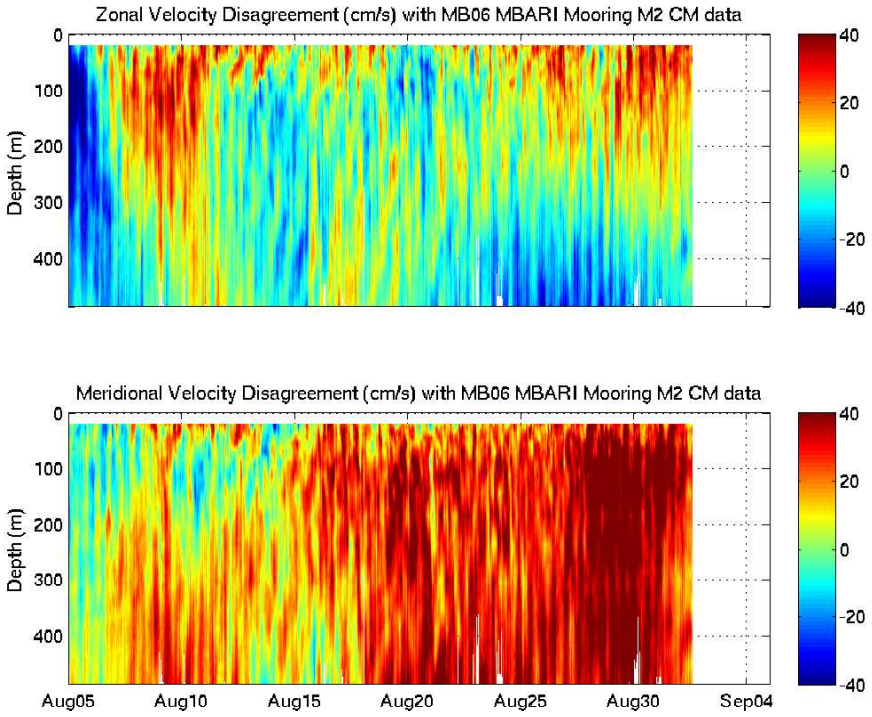

Figure 4-32: Error in M2 current meter data with new tides EVH12

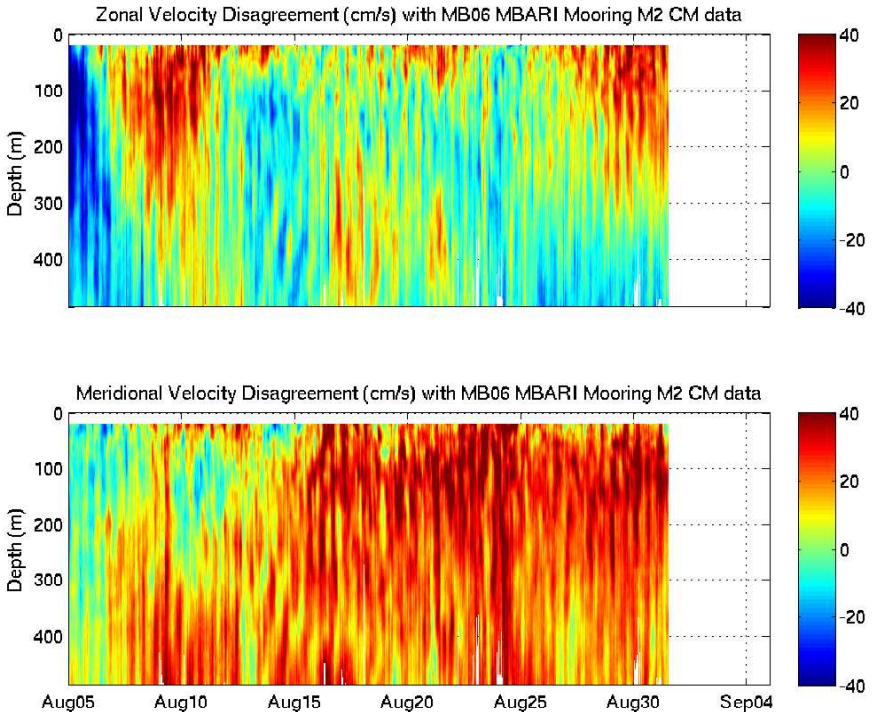

Figure 4-33: Error in M2 current meter data with new tides EVH14

Through the use of the skill metrics, comparisons of model simulations using different tidal estimates and varied parameters were carried out. Through the results obtained, the old barotropic tidal forcings shown more accurate. This observation suggests that the new, higher resolution tides be reevaluated with model alterations. Additionally, comparison among the new simulations led to the identification of the better parameters, where a larger temporal e-folding scale, maintained across the nested domains is recommended for the new forcings along with the larger of the tested tidal friction coefficients for the new simulations.

Chapter 5

Conclusion

In the course of this work, a particular aspect of adaptive modeling has been examined. Parameter estimations methods utilizing algorithms from stochastic control theory were evaluated and tested on straightforward diffusion problems. The performance of implemented methodologies provided insight with which to initialize a four-dimensional ocean simulation model using MSEAS for the purpose of assessing tunable aspects of the model using an ensemble approach. Distributed runs were issued over a high performance 266 CPU computer cluster. Model simulation results were analyzed quantitatively with the use of error metrics specified in Lermusiaux (2007).

The application of the adjoint method, EKF, EnKF, and UKF were tested on a numerical and analytical one-dimensional diffusion problem. In using these methods for parameter estimation with these simple test cases, the adjoint method proved impractical and computationally costly for highly nonlinear systems of equations. Though the EKF revealed good performance in certain applications, it too is not easily generalized to high dimensional, largely nonlinear models, and may require a substantial computational overhead cost. The EnKF showed adequate performance and ensemble methods are more readily generalized to complex simulations. Though the UKF can also be conveniently applied to nonlinear models, it may require more tuning through the related scaled UKF (Julier, 2002). Also, this method may be viewed as a type of deterministic ensemble method. As a result, of these simple case

evaluations, an ensemble approach was determined fitting for use with the realistic ocean model.

The Monterey Bay 2006 experiment was utilized for an evaluation of barotropic tidal modeling in the region. To this extent, C-shell scripts were written, and MATLAB® scripts were improved upon for the quantitative analysis of model simulation outputs with scarcely sampled oceanic fields. Comparisons drawn from these results suggested a decrease in performance with the use of the new higher-resolution barotropic tidal forcings. This behavior may be due to several reasons: the higher resolution tidal model may be creating features which on a coarser grid remained unresolved and were not adequately dissipated when the high-resolution barotropic tides were computed; the Dirichlet boundary conditions utilized at the open boundaries may require revision to allow for advection out of the domain when resolution is increased (Lermusiaux, Haley, and Logutov, Personal communication). New mixed von Neumann and Dirichlet open boundary conditions have since been implemented in the barotropic tidal model.

In evaluating the barotropic tidal forcings, numerous other parameters were examined. Comparisons among these runs with the new higher resolution tides suggested that with the increased tidal resolution a weaker coastal friction (a larger temporal e-folding scale) should be used while maintaining the same spatial e-folding scale in coastal friction as the simulation with the older tidal forcings. Choices for other tidal friction parameters resulted in less distinct effects and as such did not allow for unequivocal conclusions to be drawn from them. These preliminary results could be used to investigate other parameter values, or other parameters altogether. Still, this method can be used to quantitatively rate each aspect of the model setup.

A package was developed for the evaluation of model performance of an ensemble of simulations. The results obtained from the simulation outputs were used to produce a set of valuable quantitative metrics with which to identify how well model options or parameters pair when compared to observations or other references. In the future, a method to automatically update the parameters in the models based on the respective results obtained from the original ensemble will be sought. Once

parameters of each parameterization are sufficiently fitter, the final step will be to evaluate the performance of the various model parameterizations or model options themselves, in the same manner as the parameters, for a fully automated adaptive modeling algorithm. A final question to address will be to determine the necessity of fitting the parameters of parameterizations prior to evaluating their performance, or if only a partial fit of their parameter values would suffice to distinguish among parameterizations.

Bibliography

Anderson, J. L. (2001). An ensemble adjustment kalman filter for data assimilation. *Monthly Weather Review*, 129:2884–2903.

Anderson, J. L. and Anderson, S. L. (2001). A monte carlo implementation of the nonlinear filtering problem to produce ensemble assimilations and forecasts. *Monthly Weather Review*, 127:2741–2758.

Bannister, R. N. (2001). Elementary 4d-var. Technical report.

Errico, R. M. (1997). What is an adjoint model? *Bulletin of the American Meteorological Society*, 78(11):2577–2592.

Evensen, G. (1994). Sequential data assimilation with a nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics. *Journal of Geophysical Research*, 99(C5):10143–10162.

Gelb, A., Joseph F. Kasper, J., Raymond A. Nash, J., Price, C. F., and Arthur A. Sutherland, J. (1974). *Applied Optimal Estimation*. The M.I.T Press, Cambridge, Massachusetts.

Haley, P. J., Lermusiaux, P. F. J., Robinson, A. R., Leslie, W. G., Logutov, O., Cossarini, G., Liang, X. S., Moreno, P., Ramp, S. R., Doyle, J. D., Bellingham, J., Chavez, F., and Johnston, S. (2008). Forecasting and reanalysis in the monterey bay/california current region for the autonomous ocean sampling network-ii experiment. Deep Sea Research, Part II. In Press.

Houtekamer, P. L. and Mitchell, H. L. (1998). Data assimilation using an ensemble kalman filter technique. *Monthly Weather Review*, 126:796–811.

Jazwinski, A. H. (1970). *Stochastic Processes and Filtering Theory*. Academic Press, New York.

Julier, S., Uhlmann, J., and Durrant-Whyte, H. F. (2000). A new method for the nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on Automatic Control*, 45(3):477–482.

Julier, S. and Uhlmann, J. K. (1996). A general method for approximating nonlinear transformations of probability distributions. Technical report.

Julier, S. J. (2002). The scaled unscented transformation. In *Proceedings of the American Control Conference*, pages 4555–4559.

Julier, S. J. and Uhlmann, J. K. (2004). Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422.

Kalman, R. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME-Journal of Basic Engineering*, 82(D):35–45.

Lapidus, L. and Pinder, G. F. (1982). *Numerical Solutions of Partial Differential Equations in Science and Engineering*. John Wiley & Sons, New York.

Lermusiaux, P. F. J. (1999). Estimation and study of mesoscale variability in the strait of sicily. *Dynamics of Atmospheres and Oceans*, 29:255–303.

Lermusiaux, P. F. J. (2001). Evolving the subspace of the three-dimensional multiscale ocean variability: Massachusetts bay. *Journal of Marine Systems*, 29:385–422.

Lermusiaux, P. F. J. (2007). Adaptive modeling, adaptive data assimilation and adaptive sampling. *Physica D*, 230:172–196.

Lermusiaux, P. F. J. and Robinson, A. R. (1999a). Data assimilation via error subspace statistical estimation. part i: Theory and schemes. *Monthly Weather Review*, 127:1385–1407.

Lermusiaux, P. F. J. and Robinson, A. R. (1999b). Data assimilation via error subspace statistical estimation. part ii: Middle atlantic bight shelfbreak front simulations and esse validation. *Monthly Weather Review*, 127:1408–1432.

Li, X., Chao, Y., McWilliams, J. C., and Fu, L.-L. (2000). A comparison of two vertical mixing schemes in a pacific ocean general circulation model. *Journal of Climate*.

Mellor, G. L. (2001). One-dimensional, ocean surface layer modeling: A problem and a solution. *Journal of Physical Oceanography*, 31(3):790–809.

Niiler, P. and Kraus, E. B. (1977). One-dimensional models of the upper ocean. In Kraus, E. B., editor, *Modelling and Predictions of the Upper Layers of the Ocean.*, pages 143–172. Pergamon, New York.

Pacanowski, R. C. and Philander, S. G. H. (1981). Parameterization of vertical mixing in numerical models of tropical oceans. *Journal of Physical Oceanography*, 11(11):1443–1451.

Plessix, R. E. (2006). A review of the adjoint-state method for computing the gradient of a functional with geophysical applications. *Geophysical Journal International*, 167:495–503.

Robinson, A. R., Lermusiaux, P. F. J., and III, N. Q. S. (1998). Data assimilation. *The Sea*, 10:541–594.

van der Merwe, R. and Wan, E. A. (2003). Sigma-point kalman filters for probabilistic inference in dynamic state-space models. In *Workshop on Advances in Machine Learning, http://www.iro. umontreal.ca/ kegl/CRMWorkshop/program.html.*

van der Merwe, R. and Wan, E. A. (2004). Sigma-point kalman filters for integrated navigation. In *Proceedings of the 60th Annual Meeting of The Institute of Navigation (ION)*, Dayton, Ohio.