UNIVERSITY OF CALIFORNIA, SAN DIEGO

Adaptive Observation with Vehicle Dynamics

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Engineering Science with specialization in Computational Science

by

David Da Zhang

Committee in charge:

Professor Thomas Bewley, Chair
Doctor Bruce Cornuelle
Doctor Jules Jaffe
Professor Ryan Kastner
Professor Miroslav Krstic
Professor Daniel Tartakovsky

2011

The dissertation of David Da Zhang is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____

_____

_____
Chair

University of California, San Diego

2011

DEDICATION

To my parents, who sacrificed so much to get me where I am today.
My achievements are as much as theirs.

EPIGRAPH

"We choose to go to the moon in this decade and do the other things, not because they are easy, but because they are hard..."

—President John F. Kennedy

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to thank my office-mates Joseph Cessna and Christopher Colburn for the brain-storming sessions, which give me new ideas and made this thesis possible. I would also like to thank my advisor Professor Thomas Bewley for the thesis topic. A lot of the numerical results cannot be done within a reasonable amount of time without the unlimited access to the Greenlight supercomputer cluster run by Calit2 and UCSD, managed by Christopher Misleh.

Most credits goes especially to Christopher Colburn, where in Chapter 2, the solution for solving the infinite-time sensor placement problem in continuous time was first found by Chris. I simply extended the idea for sensor placement in discrete-time, and the equivalent extension for actuator placement. Chris is also credited for finding the different measures used in Information Theory to quantify estimation quality; he also implementing the theory on the Ginzburg-Landau equation. In Chapter 3 Chris is credited for implementing the stochastic EnKF on the Navier-Stoke's equation and gathering time-averaged performance statistics on the performances of different sampling methods in the environmental plume numerical experiment. All-in-all, the materials I've used in this thesis taken from papers and journals Chris and I co-authored are

- Colburn, C.H., Zhang, D., Bewley, T.R "*Adjoint-based Gradient Calculation for Infinite-time Optimal Sensor/Actuator Placement*", under preparation.
- Zhang, D., Colburn, C.H. Bewley, T.R. "*Estimation and Adaptive Observation of Environmental Plumes*", Proceedings of American Control Conference 2011, San Francisco, USA.
- Zhang, D., Colburn, C.H., Bewley, T.R. "*Estimation and Adaptive Observation of Environmental Plumes*", under preparation.

At the early stage of my research there were many obstacles and intellectual dry spells; I often question my decision to pursue a Ph.D and the urge to quit was strong. However it is the continuing encouragement and support from my loving parents and dear friends that kept me going; so that as I slowly grind toward the finish, I could see the light at the end of the tunnel. **And at last, here I am.**

VITA

2006	Bachelor of Science in Mechanical Engineering, minor in Mathematics, University of California, San Diego
2009	Master of Science in Engineering Science (Mechanical Engineering), University of California, San Diego
2011	Doctor of Philosophy in Engineering Science with specialization in Computational Science, University of California, San Diego

PUBLICATIONS

Zhang, D., Colburn, C.H., & Bewley, T.R. "Estimation and Adaptive Observation of Environmental Plumes", *Proceedings of American Control Conference 2011, San Francisco, USA*.

Kang, K., Zhang, D., & Bitmead, R.R. "Disturbance Rejection Control in Coordinated Systems", *Proceedings of IFAC World Congress 2008, Seoul, Korea*.

ABSTRACT OF THE DISSERTATION

Adaptive Observation with Vehicle Dynamics

by

David Da Zhang

Doctor of Philosophy in Engineering Science with specialization in Computational Science

University of California, San Diego, 2011

Professor Thomas Bewley, Chair

Technological advances in unmanned sensor vehicles allows the possibility to solving the Adaptive Observation problem with moving sensors. This problem is addressed in this thesis. Toward applying the solution in real-world problems, the derivation of the solution is broken down into three incremental steps. The first step involves solving a simpler problem at infinite time with stationary sensors. The second step take the theory establish in step one and augment it with vehicle dynamics in finite time. The last step takes a look at implementation issues when implement the theory develop in step two, and make appropriate modifications in order to derive a feasible and practical algorithm.

Chapter 1

Motivation

This research is funded by the Los Alamos National Laboratory *Contaminant Plume Identification, Estimation and Forecasting* project. The original problem statement was framed around national defense and homeland security, where feasible Unmanned Aerial Vehicles (UAVs, such as Figure 1.1(a)) flight trajectories are sought in order to improve forecast on airborne chemical/biological agent released in an urban area (represented as smoke in Figure 1.1(b)). Note in this setting, the evolution of the plume is predominately convective rather than diffusive; instead of diffusing in the air, the plume movement is forced by local wind structures and thus typically exhibits chaotic movement. Furthermore, because these wind structures have dynamic time-scale comparable to the sensor vehicle dynamic, this problem cannot simply be treated as quasi-static. Finally solution to this problem can lend itself to environmental problems such as the Icelandic volcanic ash plume, Gulf oil spill, and radioactive plume from the post-tsunami Fukushima Dai-ichi nuclear power plant in Japan.

The motion planning of sensor vehicle in order to improve forecast is called Adaptive Observation (AO); detail discussion and existing work on AO will be presented later. Toward solving the AO problem and implement the solution in practice, the solution is incrementally broken into three steps, with each step appropriately described in its respective chapter. In Chapter 2 the AO problem is simplified to an *infinite-time* sensor placement problem, where the objective is to find the optimal *static* sensor locations that improve a forecast quality metric at

(a) An environmental-sensor equipped UAV

(b) Smoke plume release representing airborne chemical/biological agent

(c) UAV gathering environmental data.

Figure 1.1: A mock experiment representing real-time deployment of mobile sensors (Figure 1.1(a)) to gather environmental data (Figure 1.1(c)) in order to forecast airborne contaminant (figure 1.1(b))

infinite-time the most. The theories established in Chapter 2 is adapted to solve the full AO problem considered in this thesis in Chapter 3, where the optimal sensor vehicle *trajectories*, conformal to vehicle dynamic, that improves forecast quality over a *finite-time* interval is computed. The resulting algorithm is called the Dynamic Adaptive Observation (DAO). There are several implementation issues when applying DAO to real-life problems, therefore approximations and new theories are developed to ensure feasible DAO implementation. This modified algorithm is called the Ensemble Variational Adaptive Observation (EnVO), and it is discussed in Chapter 4.

Chapter 2

Infinite-time Sensor Placement

In this chapter the Adaptive Observation (AO) problem is reduced to a sensor placement problem at infinite-time without considering vehicle dynamics. Sensor placement techniques for state estimation have broad applications in environmental studies, finance, and engineering. Classic applications include: sensor placement in environmental applications (Majumdar *et al.*, 2002), explosion detection and contaminant plume tracking (Zhang *et al.*, 2011), and estimation/control of chemical production/mixing procedures (Alonso *et al.*, 2004). Although it may seem obvious that results in these estimation settings are strongly dependent on sensor locations, efforts toward developing model-based optimal sensor placement algorithms have been little.

Of the works done, the majority of model-based sensor placement algorithms considers scalar measures of the Fisher Information Matrix (FIM), which is defined as the covariance of the score of a probability distribution, and is particularly useful when comparing different measures of a random process. In Information Theory, the score describes the information provided from observing a random variable, and it is the gradient of the log-likelihood function (the log of the probability distribution) with respect to some unknown variable. The FIM is written

$$\mathbf{I}_F = \mathrm{E}\left\{ \left(\frac{\partial \ln p(X|\theta)}{\partial \theta} \right)^T \left(\frac{\partial \ln p(X|\theta)}{\partial \theta} \right) \right\}, \tag{2.1}$$

where $p(x|\theta), x \in X$ is the probability distribution of a random variable X con-

ditioned on the parameter θ. The inverse of the FIM lower-bounds the singular values of the estimation error covariance matrix $\mathbf{P}$ (to be defined in §2.1) by the Cramér-Rao inequality

$$\mathbf{P} \geq \mathbf{I}_F^{-1}. \tag{2.2}$$

For convenience, a derivation of this bound and its relationship to standard linear filtering methods can be found in Appendix A.

Relevant scalar measures of the FIM are typically considered when optimizing sensor locations. The three most common measures are

- The A-optimality (trace) criterion

$$J_A(\mathbf{I_F}) \triangleq \mathrm{tr}(\mathbf{I_F}^{-1}) \tag{2.3a}$$

- The D-optimality (determinant) criterion

$$J_D(\mathbf{I_F}) \triangleq -\ln\det(\mathbf{I_F}) \tag{2.3b}$$

- The E-optimality (eigenvalue) criterion

$$J_E(\mathbf{I_F}) \triangleq \lambda_{\max}(\mathbf{I_F}^{-1}). \tag{2.3c}$$

Minimizing the D, E, and A optimality criteria respectively minimizes the uncertainty ellipsoid volume, dominant principle axis of said ellipsoid, and the average variance (Uciński, 2005, p. 16).

There have been a broad investigations into how these measures can be utilized in determining optimal sensor placement. For example, Martínez & Bullo (2006) effectively found methods for minimizing D-optimality criterion in target tracking problems. Similarly, Faulds & King (2000) propose an A-type criterion to analyze (not optimize) a model-free method for placing sensors in the domain of the 2D heat equation using Centroidal Voronoi Tesselations. These results are particularly attractive because (i) no model is required, and (ii) it can be calculated in a distributed framework (Cortes *et al.*, 2004; Bullo & Cortes, 2004; Kwok & Martinez, 2010). However computational experiments (Zhang *et al.*, 2011) with the 2D Navier-Stokes Equations have demonstrated that this is sub-optimal if

model information can be used to plan measurement sequences in the optimization process. Porat & Nehorai (1996) propose a source-seeking estimation/tracking problem where they seek to reduce the expected contaminant source location estimation error after each measurement update by performing a global search over feasible future measurement locations. Because this method scales poorly computationally, they augment the algorithm by calculating gradients to the Cramér-Rao bound at select locations within the feasible set for each sensor. The authors propose this optimization in a receding horizon setting where measurement locations eventually converge to stationary points in the domain.

Although the Cramér-Rao bound provides a mean to compute the best-possible performance achievable by any estimation algorithms, the well-known Kalman Filter (KF) only reach this performance limit when model uncertainty and process noise is neglected (see Appendix A). In practice these disturbances are typically built into the KF because ignoring them substantially effects the uncertainty distribution and has been found to cause filter divergence. Therefore the Cramér-Rao bound is rarely reached in practice, and thus it is more practical to minimize the measures based on $\mathbf{P}$ rather than $\mathbf{I}_F$.

Variational methods, on the other hand, are used widely in optimization problems for extracting gradient/sensitivity information of linear and nonlinear systems (Bewley *et al.*, 2001; Bewley & Protas, 2004). Adjoint optimization have been used for airfoil/aerodynamic shape optimization (Jameson *et al.*, 1998; Giles & Pierce, 2000), but has not been used to optimize sensor locations. Only a few papers have used adjoint methods to evaluate the sensitivity of Riccati equations, which is the basis behind the KF. Specifically Kenney & Hewer (1990) examined how Riccati solutions change as a result of modeling errors in the actuation/measurement covariance matrix.

The remainder of this chapter illustrates how adjoint analysis is used to extract the local gradient of a relevant measure with respect to the sensor positions, and be used by iterative optimization methods. §2.1 outlines the theoretical analysis for optimizing sensor placement. In §2.1.1 the analysis performed in continuous-time while §2.1.2 mirror the same analysis in discrete-time. When

performing the discrete-time analysis there is an ambiguity about how the relevant measures should be calculated; therefore the ambiguity is treated as two separate cases and analysis is performed on each. A numerical experiment is performed as an example in §2.1.4 to show how the theory established in §2.1.1 is applied. Because in linear control theory the controls problem is the dual of the estimation problem, the theories in §2.1 is adapted into a control setting in §2.2 where the optimal actuator placement is sought. Again the continuous- and discrete-time version of the analysis is performed, and they are presented respectively in §2.2.1 and §2.2.2. The same numerical experiment used in §2.1 is modified to demonstrate actuator placement optimization in §2.2.4.

2.1 Optimal Sensor Placement

2.1.1 Continuous-time Analysis

Consider a continuous-time Linear Time Invariant (LTI) system described by

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{A}\mathbf{x} + \mathbf{B}_\mathbf{a}(\mathbf{q}_\mathbf{a})\mathbf{u} + \mathbf{B}_\mathbf{w}\mathbf{w}, \tag{2.4a}$$

$$\mathbf{y} = \mathbf{C}(\mathbf{q}_\mathbf{s})\mathbf{x} + \mathbf{v}, \tag{2.4b}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the state, $\mathbf{u}(t) \in \mathbb{R}^c$ is the control, $\mathbf{y}(t) \in \mathbb{R}^m$ is the measurement, $\mathbf{w}(t)$ is the state disturbance, $\mathbf{v}(t)$ is the measurement noise, $\mathbf{q}_\mathbf{a} \in \mathbb{R}^{l_a}$ describes actuator states, and $\mathbf{q}_\mathbf{s} \in \mathbb{R}^{l_s}$ describes sensor states (e.g. positions). The dependence of the bounded operators $\mathbf{B}_\mathbf{a}$ and $\mathbf{C}$ on $\mathbf{q}_\mathbf{a}$ and $\mathbf{q}_\mathbf{s}$ is made explicit here to emphasize that some aspects of $\mathbf{B}_\mathbf{a}$ and $\mathbf{C}$ can be controlled through changes in actuator and sensor states; however this notation is dropped from hereon for simplicity. The interpretation of $\mathbf{C}$ and $\mathbf{q}_\mathbf{s}$ is perhaps most intuitive if one thinks about $\mathbf{q}_\mathbf{s}$ as the *positions* of various sensors which measures $\mathbf{x}$ to yield $\mathbf{y}$. If these sensors measure local quantities (in general this applies to most sensors), then $\mathbf{C}$ is dependent on $\mathbf{q}_\mathbf{s}$. Similarly, $\mathbf{q}_\mathbf{a}$ can be interpreted as actuator positions.

From standard LTI estimation theory, if $\mathbf{w}(t)$ and $\mathbf{v}(t)$ are *uncorrelated zero-mean, nearly-white continuous-time random processes* with respective covariance

$\mathbf{W} \geq \mathbf{0}$ and $\mathbf{R} > \mathbf{0}$, then an estimator for the state estimate $\hat{\mathbf{x}}(t)$ of the form

$$\frac{\mathrm{d}\hat{\mathbf{x}}}{\mathrm{d}t} = (\mathbf{A} - \mathbf{LC})\hat{\mathbf{x}} + \mathbf{B_a u} + \mathbf{Ly}, \qquad \mathbf{L} = \mathbf{PC}^T\mathbf{R}^{-1}, \tag{2.5}$$

with the estimation error covariance $\mathbf{P}$ satisfying the *Differential Riccati Equation* (DRE)

$$\mathbf{P}(t) = \mathrm{E}\{\tilde{\mathbf{x}}(t)\tilde{\mathbf{x}}^T(t)\} \geq 0, \qquad \tilde{\mathbf{x}} \triangleq \mathbf{x} - \hat{\mathbf{x}}, \tag{2.6a}$$

$$\frac{\mathrm{d}\mathbf{P}}{\mathrm{d}t} = \mathbf{AP} + \mathbf{PA}^T + \mathbf{W} - \mathbf{LRL}^T, \tag{2.6b}$$

is the *Best Linear Unbiased Estimator* (BLUE) (see Bucy & Joseph, 1968; Jazwinski, 1970). This is typically called the *Kalman-Bucy* Filter.

The time-dependence of $\mathbf{P}(t)$ demonstrates that the estimation uncertainty is not static; however provided the system is detectable, if (2.5) and (2.6) are allowed to run for a long time, then $\tilde{\mathbf{x}}$ converges to zero and $\mathbf{P}$ converges to some infinite-time value. The infinite-time solution can be computed directly by setting $\mathrm{d}\mathbf{P}/\mathrm{d}t = 0$, transforming it into the *Continuous-time Algebraic Riccati Equation* (CARE)

$$\mathbf{0} = \mathbf{AP} + \mathbf{PA}^T + \mathbf{W} - \mathbf{LRL}^T, \qquad \mathbf{L} = \mathbf{PC}^T\mathbf{R}^{-1}, \quad \mathbf{P}^T = \mathbf{P}. \tag{2.7}$$

This implies, from a filter design perspective, that the converged performance limit of the filter designed according to (2.5) and (2.6) can be known *ahead of time*. Therefore, a natural question is how should (2.6) be manipulated so that the filter achieves the best performance when converged.

In this work, matrices $\mathbf{A}$, $\mathbf{W}$, and $\mathbf{R}$ are assumed to be constant; therefore (2.6) is only manipulated through variations in $\mathbf{C}$, which is implicitly determined by changes of $\mathbf{q_s}$. Thus, an optimization problem can be posed where the optimal $\mathbf{q_s}$ that minimizes a scalar cost of $\mathbf{P}$ in (2.7). In particular, the optimal stationary $\mathbf{q_s}$ is sought such that

$$\min_{\mathbf{q_s}} \; J(\mathbf{q_s}) = \mathrm{trace}(\mathbf{P}), \tag{2.8a}$$

$$\text{s.t. } \; \mathbf{0} = \mathbf{AP} + \mathbf{PA}^T + \mathbf{W} - \mathbf{LRL}^T, \qquad \mathbf{L} = \mathbf{PC}^T\mathbf{R}^{-1}, \tag{2.8b}$$

where in the spirit of the A-optimality criterion, the more practical trace($\mathbf{P}$) rather than trace($\mathbf{I}_F^{-1}$) is used as the cost. Alternative cost metrics are discussed in §2.1.3.

Since closed-form solution to the CARE can only be found for systems with exceptionally simple dynamics, differentiate (2.8a) analytically is not possible; therefore, iterative methods must be used. In an iterative method an initial $\mathbf{q_s}$ is arbitrarily selected and local gradient of the cost function with respect to this initial condition,$\nabla_{\mathbf{q_s}} J$,is calculated; the local gradient information is used to find the next $\mathbf{q_s}$ that produces a lower cost. This is iterated until a minimizing solution is found. Note if the cost function is nonlinear and/or non-convex, it is very likely the optimal solution only converges to a local minimum; thus global optimality is not guaranteed with iterative methods.

Because the number of iterations it takes for the solution to converge to a minimum varies greatly, a critical step is in efficiently finding the gradient $\nabla_{\mathbf{q_s}} J$. This gradient is commonly approximated using finite-difference methods (the process of measuring how J changes from small perturbations in $\mathbf{q_s}$). This type of gradient approximation is very sensitive to the perturbation step-size. Ideally one would choose a perturbation step-size as small as possible to ensure the gradient magnitude and direction is accurate; however as the step-size becomes very small, the gradient calculation is corrupted from finite precision floating-point arithmetics. Furthermore, one would need to evaluate (2.8) l_s times to approximate a single $\nabla_{\mathbf{q_s}} J$ for a first-order finite-difference approximation; if higher order approximation is used (such as central-difference), the number of evaluations increases further. For these reasons it is better to seek an analytic expression for $\nabla_{\mathbf{q_s}} J$, which cuts down computation time and increase accuracy (since the gradient computation is exact).

The following analysis demonstrates how the local gradient can be analytically derived. Suppose an initial $\mathbf{q_s}$ with the corresponding $\mathbf{C}$, associated CARE solution, and J are evaluated. Now if perturbations are applied to $\mathbf{q_s}$, then $\mathbf{C}$, the CARE solution, and J are also perturbed. Subtracting the perturbed equations

with the originals yields the first-order perturbations:

$$J' = \text{trace}(\mathbf{P}'), \tag{2.9a}$$

$$\begin{aligned} \mathbf{0} &= \mathbf{A}\mathbf{P}' + \mathbf{P}'\mathbf{A}^T - \mathbf{L}'\mathbf{R}\mathbf{L}^T - \mathbf{L}\mathbf{R}(\mathbf{L}')^T, \\ \mathbf{L}' &= \mathbf{P}'\mathbf{C}^T\mathbf{R}^{-1} + \mathbf{P}(\mathbf{C}')^T\mathbf{R}^{-1}, \end{aligned} \tag{2.9b}$$

$$\mathbf{C}' = \left(\frac{\mathrm{d}\mathbf{C}}{\mathrm{d}\mathbf{q_s}}\right)^T \mathbf{q_s}'. \tag{2.9c}$$

Note that $\mathrm{d}\mathbf{C}/\mathrm{d}\mathbf{q_s}$ is a rank-3 tensor that contracts to a rank-2 matrix $\mathbf{C}'$ by the inner-product with $\mathbf{q}'$. Also note that since process noise $\mathbf{w}(t)$ and measurement noise $\mathbf{v}(t)$ are not affected by the perturbations, $\mathbf{W}' = \mathbf{0}$ and $\mathbf{R}' = \mathbf{0}$. For notational purposes (2.9b) is reposed as two linear operators $\mathrm{L}(\mathbf{P}')$ and $\mathrm{M}(\mathbf{C}')$ which are defined as

$$\begin{aligned} \mathrm{L}(\mathbf{P}') &= \mathrm{M}(\mathbf{C}'), \\ \mathrm{L}(\mathbf{P}') &\triangleq \mathbf{A}\mathbf{P}' + \mathbf{P}'\mathbf{A}^T - \mathbf{P}'\mathbf{C}^T\mathbf{L}^T - \mathbf{L}\mathbf{C}\mathbf{P}', \\ \mathrm{M}(\mathbf{C}') &\triangleq \mathbf{P}(\mathbf{C}')^T\mathbf{L}^T + \mathbf{L}\mathbf{C}'\mathbf{P}. \end{aligned} \tag{2.10}$$

From the Taylor expansion of J about the initial $\mathbf{q_s}$, J' is defined as

$$J' = (\nabla_{\mathbf{q_s}} J)^T \mathbf{q_s}'. \tag{2.11}$$

Therefore, if (2.9a) can be reposed to the form in (2.11), then $\nabla_{\mathbf{q_s}} J$ can be readily extracted. To perform this conversion an appropriate matrix inner-product

$$\langle \mathbf{X}, \mathbf{Y} \rangle \triangleq \text{trace}(\mathbf{X}^T\mathbf{Y}) \tag{2.12}$$

is defined, along with a matrix adjoint variable $\mathbf{S}$, and the adjoint operator $\mathrm{L}^*(\mathbf{S})$ such that

$$\begin{aligned} \langle \mathbf{S}, \mathrm{L}(\mathbf{P}') \rangle &= \langle \mathrm{L}^*(\mathbf{S}), \mathbf{P}' \rangle, \\ \mathrm{L}^*(\mathbf{S}) &= \mathbf{A}^T\mathbf{S} + \mathbf{S}\mathbf{A} - \mathbf{S}\mathbf{L}\mathbf{C} - \mathbf{C}^T\mathbf{L}^T\mathbf{S} \\ &= (\mathbf{A} - \mathbf{L}\mathbf{C})^T\mathbf{S} + \mathbf{S}(\mathbf{A} - \mathbf{L}\mathbf{C}), \end{aligned} \tag{2.13}$$

where the RHS of $\mathrm{L}^*(\mathbf{S})$ is derived by substituting (2.10) into (2.13) and using the trace identity $\text{trace}(\mathbf{A}\mathbf{B}) = \text{trace}(\mathbf{A}^T\mathbf{B}^T) = \text{trace}(\mathbf{B}\mathbf{A})$ to shift $\mathbf{P}'$ to the right.

Recognizing (2.9a) can be expressed using (2.12), it is immediately clear from (2.10) and (2.13) that if $\mathrm{L}^*(\mathbf{S}) = \mathbf{I}$ then (2.9a) is exactly

$$\begin{aligned} J' &= \operatorname{trace}(\mathbf{P}') \\ &= \langle \mathbf{I}, \mathbf{P}' \rangle \\ &= \langle \mathrm{L}^*(\mathbf{S}), \mathbf{P}' \rangle \\ &= \langle \mathbf{S}, \mathrm{L}(\mathbf{P}') \rangle \\ &= \langle \mathbf{S}, \mathrm{M}(\mathbf{C}') \rangle \\ &= \operatorname{trace}\left(2\mathbf{PSL}\frac{\mathrm{d}\mathbf{C}}{\mathrm{d}\mathbf{q_s}}\mathbf{q_s'} \right) \qquad \text{shift } \mathbf{C}' \text{ to the right.} \end{aligned} \tag{2.14}$$

Thus, the gradient of the cost function (2.8a) with respect to the i-th element of the sensor state vector $\mathbf{q_s}$, q_s^i, is

$$\nabla_{q_s^i} J \triangleq \operatorname{trace}\left(2\mathbf{PSL}\frac{\mathrm{d}\mathbf{C}}{\mathrm{d}q_s^i} \right), \qquad \left[\nabla_{q_s^1} J, \quad \cdots \quad , \nabla_{q_s^{l_s}} J \right] \triangleq \nabla_{\mathbf{q_s}} J \tag{2.15}$$

where $\mathbf{S}$ satisfies the *Continuous-time Algebraic Lyapunov Equation* (CALE)

$$(\mathbf{A} - \mathbf{LC})^T \mathbf{S} + \mathbf{S}(\mathbf{A} - \mathbf{LC}) = \mathbf{I}. \tag{2.16}$$

2.1.2 Discrete-time Analysis

To be rigorous, a discrete time derivation equivalent to §2.1.1, is presented to clarify the subtle differences between the continuous- and discrete- time formulation of the optimal sensor placement problem. Consider a discrete-time linear system described by

$$\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k + \mathbf{G}(\mathbf{q_a})\mathbf{u}_k + \mathbf{w}_k, \tag{2.17a}$$

$$\mathbf{y}_k = \mathbf{H}(\mathbf{q_s})\mathbf{x}_k + \mathbf{v}_k, \tag{2.17b}$$

where $\mathbf{x}_k$, $\mathbf{u}_k$, $\mathbf{w}_k$, $\mathbf{y}_k$, and $\mathbf{v}_k$ are the discrete-time equivalents of the same variables in (2.4), and $\mathbf{F}$, $\mathbf{G}(\mathbf{q_a})$, and $\mathbf{H}(\mathbf{q_s})$ are discretizations of $\mathbf{A}$, $\mathbf{B}(\mathbf{q_a})$, and $\mathbf{C}(\mathbf{q_s})$ in (2.4) respectively. Similarly, $\mathbf{q_a}$ and $\mathbf{q_s}$ are the states of the actuators and sensors. For notational convenience, $\mathbf{G}$ and $\mathbf{H}$ represent $\mathbf{G}(\mathbf{q_a})$ and $\mathbf{H}(\mathbf{q_s})$ respectively.

If $\mathbf{w}_k$ and $\mathbf{v}_k$ are *uncorrelated zero-mean, white, discrete-time random processes* with respective covariance $\mathbf{W} \geq 0$ and $\mathbf{R} > 0$, then the discrete-time version

of the Kalman-Bucy filter is described as a two step process: a time-update and a measurement-update. The state estimate $\hat{\mathbf{x}}_k$ evolution equations are written

$$\hat{\mathbf{x}}_{k+1}^- = \mathbf{F}\hat{\mathbf{x}}_k^+ + \mathbf{G}\mathbf{u}_k, \tag{2.18a}$$

$$\begin{aligned} \hat{\mathbf{x}}_k^+ &= \hat{\mathbf{x}}_k^- + \mathbf{L}_k(\mathbf{y}_k - \mathbf{H}\hat{\mathbf{x}}_k^-), \\ \mathbf{L}_k &= \mathbf{P}_k^-\mathbf{H}^T(\mathbf{H}\mathbf{P}_k^-\mathbf{H}^T + \mathbf{R})^{-1}. \end{aligned} \tag{2.18b}$$

The covariance evolution follows a similar two-step process:

$$\mathbf{P}_{k+1}^- = \mathbf{F}\mathbf{P}_k^+\mathbf{F}^T + \mathbf{W}, \tag{2.19a}$$

$$\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{L}_k\mathbf{H})\mathbf{P}_k^-. \tag{2.19b}$$

The notation $(\,)_k^+$ and $(\,)_k^-$ represents the best estimate at time t_k given measurements up t_k and t_{k-1}, respectively. In particular, $\hat{\mathbf{x}}_k^-$ and $\mathbf{P}_k^-$ are often called the prior estimate and prior covariance, whereas $\hat{\mathbf{x}}_k^+$ and $\mathbf{P}_k^+$ are often called the posterior estimate and the posterior covariance.

Similar to §2.1.1 the trace of the covariance $\mathbf{P}$ is utilized for minimization, but because the two step process it is unclear whether the prior or the posterior covariance should be used. For completeness, both results are presented, and applications should dictate which covariance should be used.

Prior Covariance Optimizations

The infinite-time prior covariance of (2.19) is computed by substituting (2.19b) into (2.19a), and letting $\mathbf{P} = \mathbf{P}_{k+1|k} = \mathbf{P}_{k|k-1}$:

$$\begin{aligned} \mathbf{P} &= \mathbf{F}\left[(\mathbf{I} - \mathbf{L}\mathbf{H})\mathbf{P}\right]\mathbf{F}^T + \mathbf{W}, \qquad \mathbf{L} = \mathbf{P}\mathbf{H}(\mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R})^{-1} \\ &= \mathbf{F}\mathbf{P}\mathbf{F}^T - \mathbf{F}\mathbf{P}\mathbf{H}^T(\mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R})^{-1}\mathbf{H}\mathbf{P}\mathbf{F}^T + \mathbf{W}, \qquad \mathbf{P}^T = \mathbf{P}. \end{aligned} \tag{2.20}$$

Equation (2.20) is called the *Discrete-time Algebraic Riccati Equation* (DARE). The discrete-time equivalent to (2.8) is proposed:

$$\begin{aligned} &\min_{\mathbf{q_s}} \; J(\mathbf{q_s}) = \operatorname{trace}(\mathbf{P}), \\ &\quad \text{s.t. } \mathbf{P} = \mathbf{F}\mathbf{P}\mathbf{F}^T - \mathbf{F}\mathbf{P}\mathbf{H}^T(\mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R})^{-1}\mathbf{H}\mathbf{P}\mathbf{F}^T + \mathbf{W}, \end{aligned} \tag{2.21}$$

where the first-order perturbations are

$$J' = \text{trace}(\mathbf{P}') = \langle \mathbf{I}, \mathbf{P}' \rangle, \tag{2.22a}$$

$$\begin{aligned} \mathbf{P}' = {} & \mathbf{F}\mathbf{P}'\mathbf{F}^T - \mathbf{F}\mathbf{P}'\mathbf{H}^T\mathbf{L}^T - \mathbf{L}\mathbf{H}\mathbf{P}'\mathbf{F}^T \\ & + \mathbf{F}\mathbf{L}(\mathbf{H}'\mathbf{P}\mathbf{H}^T + \mathbf{H}\mathbf{P}'\mathbf{H}^T + \mathbf{H}\mathbf{P}(\mathbf{H}')^T)\mathbf{L}^T\mathbf{F}^T, \end{aligned} \tag{2.22b}$$

$$\mathbf{H}' = \left(\frac{\mathrm{d}\mathbf{H}}{\mathrm{d}\mathbf{q_s}}\right)^T \mathbf{q_s}', \tag{2.22c}$$

and once again $\mathbf{W}' = \mathbf{0}$ and $\mathbf{R}' = \mathbf{0}$. Note the perturbation of the inverse formula from Petersen & Pedersen (2008), $\mathbf{\Phi}^{-1}$ is $(\mathbf{\Phi}^{-1})' = -\mathbf{\Phi}^{-1}\mathbf{\Phi}'\mathbf{\Phi}^{-1}$, is used to perform perturbation to $(\mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R})^{-1}$:

$$\left((\mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R})^{-1}\right)' = (\mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R})^{-1}(\mathbf{H}'\mathbf{P}\mathbf{H}^T + \mathbf{H}\mathbf{P}'\mathbf{H}^T + \mathbf{H}\mathbf{P}(\mathbf{H}')^T)(\mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R})^{-1}. \tag{2.23}$$

The local gradient derivation is similar to §2.1.1. After defining the matrix variable $\mathbf{S}$ and inner product (2.12), and performing the necessary rearrangements, one ultimately arrives at the *Discrete-time Algebraic Lyapunov Equation* (DALE) that $\mathbf{S}$ satisfies:

$$\underbrace{\mathbf{S} - (\mathbf{I} - \mathbf{H}^T\mathbf{L}^T)\mathbf{F}^T\mathbf{S}\mathbf{F}(\mathbf{I} - \mathbf{L}\mathbf{H})}_{\mathrm{L}^*(\mathbf{S})} = \mathbf{I}, \tag{2.24}$$

and the gradient of J with respect to the i-th element of $\mathbf{q_s}$ is

$$\nabla_{q_s^i} J = \text{trace}\left(2\mathbf{P}(\mathbf{H}^T\mathbf{L}^T - \mathbf{I})\mathbf{F}^T\mathbf{S}^T\mathbf{F}\mathbf{L}\frac{\mathrm{d}\mathbf{H}}{\mathrm{d}q_s^i}\right), \qquad \left[\nabla_{q_s^1} J, \quad \cdots \quad , \nabla_{q_s^{l_s}} J\right] \triangleq \nabla_{\mathbf{q_s}} J. \tag{2.25}$$

Note that (2.25) is the gradient of (2.21), which derives from the infinite-time solution of the prior covariance $\mathbf{P}_k^-$. However, one may argue it is better to minimize the infinite-time solution of the posterior covariance $\mathbf{P}_k^+$ instead.

Posterior covariance optimization

The infinite-time posterior covariance of (2.19) is computed by substituting (2.19a) into (2.19b):

$$\begin{aligned} \mathbf{P}_{k|k} &= (\mathbf{I} - \mathbf{L}_k\mathbf{H})(\mathbf{F}\mathbf{P}_{k-1|k-1}\mathbf{F}^T + \mathbf{W}), \\ \mathbf{L}_k &= \mathbf{P}_{k|k-1}\mathbf{H}^T(\mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^T + \mathbf{R})^{-1}, \\ \mathbf{P}_{k|k-1} &= \mathbf{F}\mathbf{P}_{k-1|k-1}\mathbf{F}^T + \mathbf{W}. \end{aligned} \tag{2.26}$$

After defining $\mathbf{P} = \mathbf{P}_{k+1|k+1} = \mathbf{P}_{k|k}$, and leveraging the Matrix Inversion Lemma

$$\mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{C}(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1}\mathbf{B}\mathbf{D}^{-1} = (\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1}, \tag{2.27}$$

equation (2.26) is best written as

$$\mathbf{P}^{-1} = (\mathbf{F}\mathbf{P}\mathbf{F}^T + \mathbf{W})^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}, \qquad \mathbf{P}^{-T} = \mathbf{P}^{-1}. \tag{2.28}$$

Applying perturbations, the first-order perturbation is

$$-\mathbf{P}^{-1}\mathbf{P}'\mathbf{P}^{-1} = -(\mathbf{F}\mathbf{P}\mathbf{F}^T+\mathbf{W})^{-1}\mathbf{F}\mathbf{P}'\mathbf{F}^T(\mathbf{F}\mathbf{P}\mathbf{F}^T+\mathbf{W})^{-1}+(\mathbf{H}^T)'\mathbf{R}^{-1}\mathbf{H}+\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}', \tag{2.29}$$

where once again the matrix inverse perturbation formula is used.

Using the similar procedure as before to define $\mathbf{S}$ and the appropriate adjoint inner product, one could show the new constrain equation for $\mathbf{S}$

$$\underbrace{-\mathbf{P}^{-1}\mathbf{S}\mathbf{P}^{-1} + \mathbf{F}^T(\mathbf{F}\mathbf{P}\mathbf{F}^T + \mathbf{W})^{-1}\mathbf{S}(\mathbf{F}\mathbf{P}\mathbf{F}^T + \mathbf{W})^{-1}\mathbf{F}}_{\mathrm{L}^*(\mathbf{S})} = \mathbf{I}, \tag{2.30a}$$

can be reëxpressed as a DALE

$$\mathbf{S} - \mathbf{P}\mathbf{F}^T(\mathbf{F}\mathbf{P}\mathbf{F}^T + \mathbf{W})^{-1}\mathbf{S}(\mathbf{F}\mathbf{P}\mathbf{F}^T + \mathbf{W})^{-1}\mathbf{F}\mathbf{P} + \mathbf{P}^2 = \mathbf{0}, \tag{2.30b}$$

with corresponding gradient for the i-th element of $\mathbf{q_s}$

$$\nabla_{q_s^i} J = \mathrm{trace}\left(2\mathbf{S}\mathbf{H}^T\mathbf{R}^{-1}\frac{\mathrm{d}\mathbf{H}}{\mathrm{d}q_s^i}\right), \qquad \left[\nabla_{q_s^1} J, \quad \cdots \quad, \nabla_{q_s^{l_s}} J\right] \triangleq \nabla_{\mathbf{q_s}} J. \tag{2.31}$$

The gradient in (2.25) and (2.31) are different because the cost they are minimizing are different. While (2.31) minimizes the cost with respect to the infinite-time solution of the posterior covariance, denote now as $\mathbf{P}^+$, (2.25) minimizes the cost with respect to the prior covariance infinite-time solution $\mathbf{P}^-$. Using (2.26), the cost function relationship between the two cases becomes clear:

$$J^- = \mathrm{trace}(\mathbf{P}^-) = \mathrm{trace}(\mathbf{F}\mathbf{P}^+\mathbf{F}^T + \mathbf{W}) \tag{2.32a}$$

$$J^+ = \mathrm{trace}(\mathbf{P}^+) \tag{2.32b}$$

Since $\mathbf{W}$ is constant, both costs are equivalent when $\mathrm{trace}(\mathbf{F}\mathbf{P}^+\mathbf{F}^T) = \mathrm{trace}(\mathbf{P}^+)$, which is true when $\mathbf{F}^T\mathbf{F} = \mathbf{I}$.

2.1.3 Different Costs

In order to compute $\nabla_{\mathbf{qs}} J$ analytically, one must solve a single algebraic Riccati equation and a single algebraic Lyapunov equation. After the analysis performed, it is clear that the right-hand-side forcing to $\mathrm{L}^*(\mathbf{S})$ is determined completely by the cost function. Appropriately new versions of the candidate cost functions described in (2.3) can be defined where the estimation error covariance replaces the Fisher Information Matrix. A summary outlining the relationship between choice of cost function and right-hand-side forcing to $\mathrm{L}^*(\mathbf{S})$ is shown:

- The A-optimality (trace) criterion:

$$\begin{aligned} J_A &= \mathrm{trace}(\mathbf{P}), \\ J'_A &= \mathrm{trace}(\mathbf{P}') = \langle \mathbf{I}, \mathbf{P}' \rangle, \\ \mathrm{L}^*(\mathbf{S}) &= \mathbf{I}. \end{aligned} \tag{2.33a}$$

- The D-optimality (determinant) criterion:

$$\begin{aligned} J_D &= \ln \det(\mathbf{P}), \\ J'_D &= \mathrm{trace}(\mathbf{P}^{-1}\mathbf{P}') = \langle \mathbf{P}^{-1}, \mathbf{P}' \rangle, \\ \mathrm{L}^*(\mathbf{S}) &= \mathbf{P}^{-1}. \end{aligned} \tag{2.33b}$$

- The E-optimality (eigenvalue, where $\mathbf{r}$ is the eigenvector of $\lambda_{\max}$) criterion:

$$\begin{aligned} J_E &= \lambda_{\max}(\mathbf{P}), \\ J'_E &= \mathrm{trace}(\mathbf{r}\mathbf{r}^T\mathbf{P}') = \langle \mathbf{r}\mathbf{r}^T, \mathbf{P}' \rangle, \\ \mathrm{L}^*(\mathbf{S}) &= \mathbf{r}\mathbf{r}^T. \end{aligned} \tag{2.33c}$$

Matrix perturbation formula from Petersen & Pedersen (2008) is used to compute (2.33b), and simple eigenvalue perturbation theory from Horn & Johnson (1990) is used to compute (2.33c).

Although theoretically possible, note the computational effort involved in computing J at each optimization step for (2.33b) and (2.33c) are significantly higher compared to the A-optimality metric considered in (2.33a).

2.1.4 Numerical Experiment

Setup

The Ginzburg-Landau (GL) equation shares many of the iconic characteristics exhibited by the Navier-Stokes Equation (NSE), and for this reason it has been a favorite model for model-based control/estimation of fluid systems. The most notable similarities are the exhibition of transient energy growth (attributed to the non-normality of the eigenvectors), bounded oscillating solutions (yet to be proved for the NSE), and well defined stability criterion (where quantitative thresholds for convective/global instabilities have been identified). Furthermore, the relative simplicity required to analyze a complex-valued 1D PDE and the equivalence with a broad range of physical phenomena perhaps explain its common usage. The 1D linear GL equation for a field ϕ on the x-axis is written

$$\frac{\partial \phi}{\partial t} = \mathbf{A}(x)\phi = \left(-\nu \frac{\partial}{\partial x} + \mu(x) + \gamma \frac{\partial^2}{\partial x^2} \right) \phi \tag{2.34}$$

where ν, γ are complex coefficients which parameterize the convective and dissipative properties of the flow respectively, and $\mu(x)$ characterizes local stability properties. This parameterization allows for stable, convectively unstable ("sub-critical"), and globally unstable ("super-critical") flows to be observed. A review by Bagheri *et al.* (2009) details these conditions, and provides an intuitive presentation of each flow state.

Bagheri *et al.* (2009) have also provided a code-base for analyzing the GL equation in this recent (and possibly most thorough) review of the application of control/estimation theory to the GL equation. This code-base provides a straightforward mechanism for consistent analysis of the sub/super-critical GL equation flows when discretized using a Hermite polynomial expansion. In this setting (2.34) and be rewritten

$$\frac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}t} = \mathbf{A}(x)\boldsymbol{\phi} + \mathbf{B_a}(\mathbf{q_a}, x)\mathbf{u} + \mathbf{B_w}(\mathbf{q_w}, x)\mathbf{w} \tag{2.35}$$

$$\mathbf{y} = \mathbf{C}(\mathbf{q_s}, x)\boldsymbol{\phi} + \mathbf{v} \tag{2.36}$$

where $\boldsymbol{\phi} \in \mathbb{R}^n$ is the discretization of ϕ, $\mathbf{u} \in \mathbb{R}^c$, $\mathbf{y} \in \mathbb{R}^m$, $\mathbf{A}(x)$ is a linear spatially-varying operator defined in (2.34), $\mathbf{B}(\mathbf{q_a}, x)$ and $\mathbf{B}(\mathbf{q_w}, x)$ are constant spatially

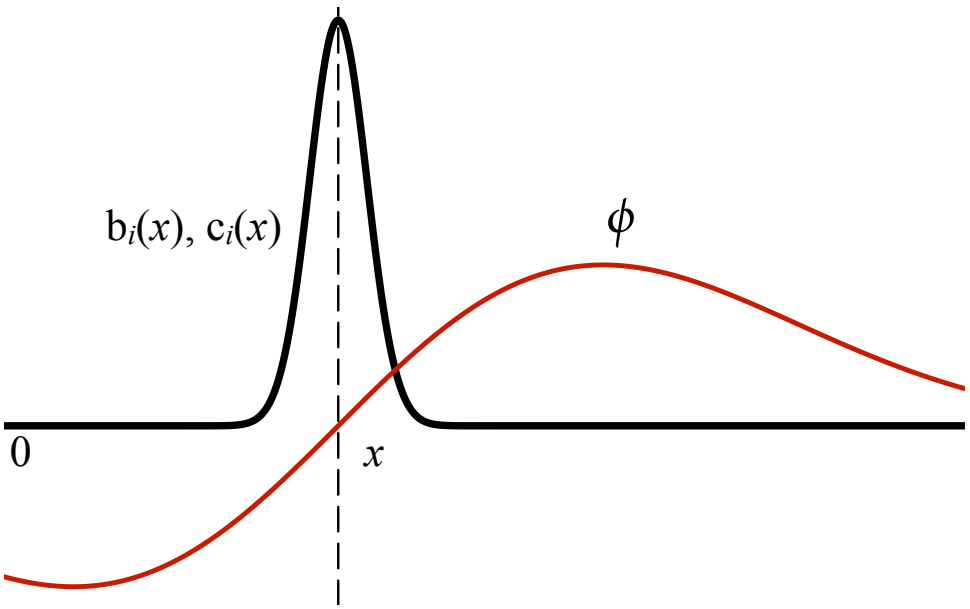


Figure 2.1: Graphical representation of the functions $\mathbf{b}_i(x)$ and $\mathbf{c}_i(x)$. The functions are Gaussian-shaped masking functions in space, thus each operation on ϕ is effectively localized to where $q_i^* = x, * = \{\mathbf{a}, \mathbf{w}, \mathbf{s}\}$.

distributed forcing operators centered at $\mathbf{q_a}$ and $\mathbf{q_w}$ respectively, $\mathbf{C}(\mathbf{q_s}, x)$ describes the measurement operator for a sensor centered at $\mathbf{q_s}$, and random vectors $\mathbf{w}$ and $\mathbf{v}$ are normally distributed with variance $\mathbf{W}$ and $\mathbf{R}$, respectively. In this experiment, $\mathbf{q_a}$, $\mathbf{q_w}$ and $\mathbf{q_s}$ are considered to contain only the positions of the actuators, disturbance, and sensors, q_i^a, q_i^w, and q_i^s, respectively. In this framework the matrices $\mathbf{B}(\mathbf{q_a}, x)$, $\mathbf{B}(\mathbf{q_w}, x)$, and $\mathbf{C}(\mathbf{q_s}, x)$ are explicitly written as the collection of Gaussian functions (see Figure E.18)

$$\mathbf{B}(\mathbf{q_a}, x) = \begin{bmatrix} \mathbf{b}_1, & \mathbf{b}_2, & \cdots & \mathbf{b}_c \end{bmatrix}, \qquad \mathbf{b}_i(x) = \exp(-(q_a^i - x)^2/2\sigma_a^2), \quad (2.37a)$$

$$\mathbf{B}(\mathbf{q_w}, x) = \begin{bmatrix} \mathbf{b}_1, & \mathbf{b}_2, & \cdots & \mathbf{b}_{size(\mathbf{w})} \end{bmatrix}, \quad \mathbf{b}_i(x) = \exp(-(q_w^i - x)^2/2\sigma_w^2), \quad (2.37b)$$

$$\mathbf{C}(\mathbf{q_s}, x) = \mathbf{M}^T \begin{bmatrix} \mathbf{c}_1^T, & \mathbf{c}_2^T, & \cdots & \mathbf{c}_m^T \end{bmatrix}^T, \quad \mathbf{c}_i(x) = \exp(-(q_s^i - x)^2/2\sigma_c^2). \quad (2.37c)$$

The matrix $\mathbf{M}$ in (2.37c) reflects the additional conditioning required to properly integrate the effects of n orthogonal Hermite polynomials.

In the following $m = 2$ and $size(\mathbf{w}) = 1$, and $\sigma_w^2 = \sigma_a^2 = 0.5$, which corresponds to a 2 sensors optimal placement problem. The remaining parameters were selected to coincide with that of the sub-critical case studied by Bagheri *et al.*

(2009) and Chen & Rowley (2010) where $q_1^w = -11.0$, $\nu = 2 + 0.2i$, $\gamma = 1 - i$, and $\mu(x) = 0.38 - 0.01x^2/2$. The resulting variable-coefficient PDE has an unstable domain, $\mu(x) > 0$, for all $x \in (-8.7178, 8.7178)$.

Results

Before analyzing the influence that observations have on the estimation error covariance, it's important to understand the infinite-time statistics of the system. Figure 2.2 visualizes $\mathbf{P}$ where the GL dynamics (2.34) are forced but no measurements are used for estimation. Although not visible in the figure, the disturbance decays slightly at first (because the disturbance originates in a stable region of the flow), grows exponentially through the unstable region, and then stabilizes itself as the flow is advected out of the unstable domain. It is this locally unable but globally stable behavior that ensures $\mathbf{P}$ does not blow up at infinite-time.

Finding the optimal placement for a single sensor on the GL equation is relatively simple – a line search will result in finding the global minimum. The problem becomes substantially more challenging when considering the placement of two or more sensors because each new sensor increases the dimension of the optimization space, making the accurate gradient computation critical.

Figures 2.2 and 2.3 summarize the difference in uncertainty reduction for different sensor placement strategies considered. The comparison of Figure 2.2 with Figure 2.3 demonstrates an improvement in performance when measurements are chosen appropriately, as seen through the orders-of-magnitude uncertainty reduction when measurements are taken. Sensor locations in Figure 2.3(a) are chosen heuristically (where sensors are placed sequentially at the locations where maximum uncertainty is observed), whereas sensor locations in Figure 2.3(b) are chosen using the algorithm described in §2.1.1. Another order-or-magnitude improvement compared to the heuristic method is observed when sensor placements are determined through the gradient method. Figure 2.4 is an image of the optimization surface, and includes the path taken during the optimization process. Notice that the converged solution is $q_s^1 = -10.65$ and $q_s^2 = 2.12$, and that this solution is

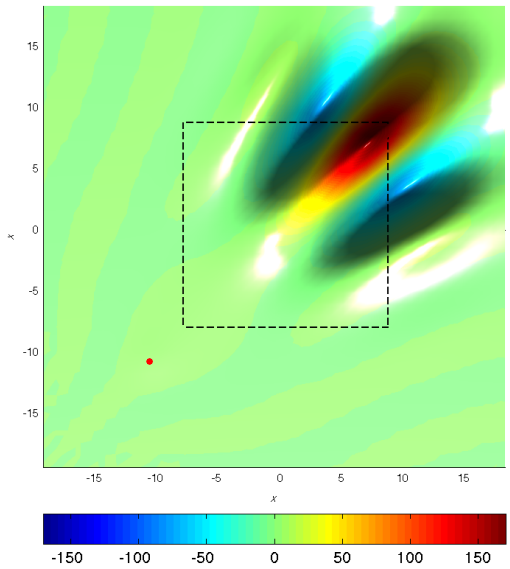
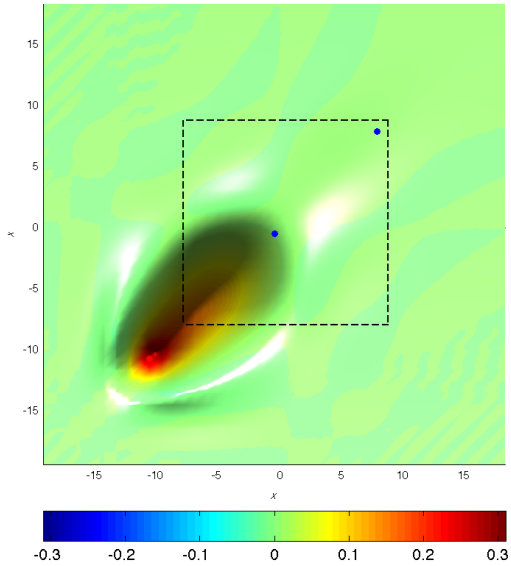
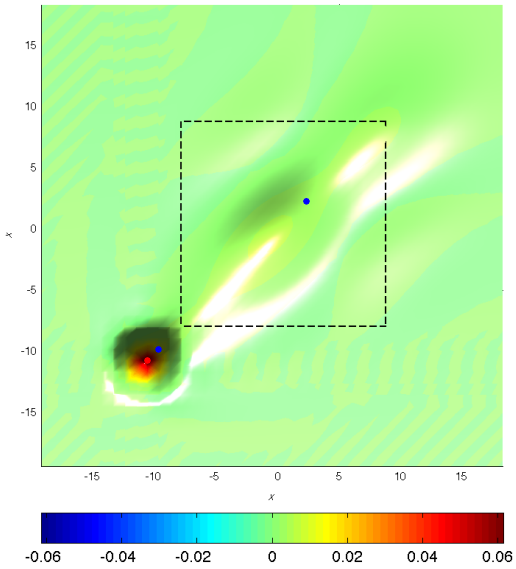


Figure 2.2: The infinite-time $\mathbf{P}$ of the Ginzburg-Landau equation when stochastic forcing is applied with no measurement information. Image reproduced from Bagheri *et al.* (2009). The image diagonal represent the variances of $\mathbf{P}$ and the off-diagonal represent the covariances. The red circle signifies the forcing location ($q_w^1 = -11.0$), and the dashed box marks the region of instability.

symmetric about the line $q_s^2 = q_s^1$ because the sensor positions are interchangeable. Because there is only one minimum in the cost (since sensor positions are interchangeable), the solution converges to a global minimum. Had the sensors have different noise attributes (e.g. $\mathbf{R} = \text{diag}(R_1, R_2)$, where $R_1 \neq R_2$), then one sensor would have been favored over the other, and the symmetry would have been broken. It is unclear whether in this case there would be one minimum or multiple minima.



(a) Actuators placed by choosing locations of maximal uncertainty



(b) Actuators placed with gradient calculation

Figure 2.3: The infinite-time **P** of the Ginzburg-Landau equation when stochastic forcing is applied and measurements are taken where sensor locations are chosen heuristically (Figure 2.3(a)) and chosen using the gradient method described in Section 2.1.1 (Figure 2.3(b)). Note the amplitude of the covariance is reduced by nearly 4 orders-of-magnitude as compared to Figure 2.2, and another order-of-magnitude by using the gradient method.

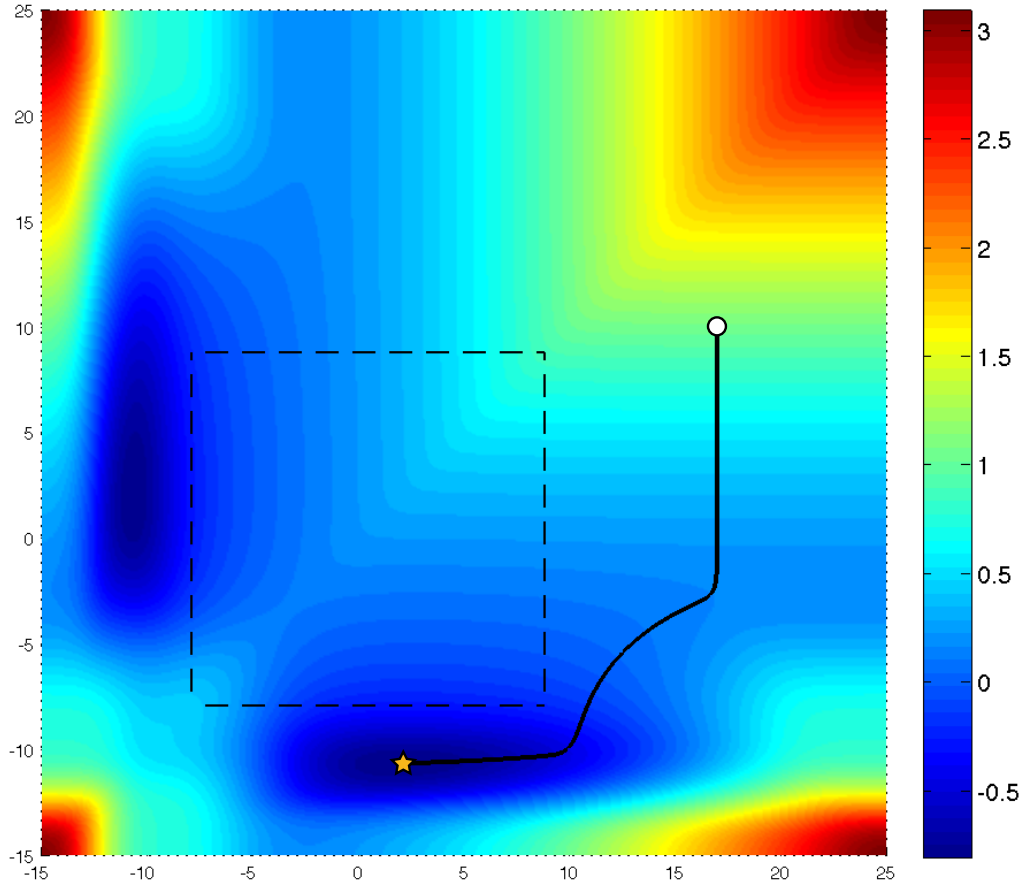


Figure 2.4: A $\log_{10}$ plot of the optimization surface for 2 sensor placement problem in the GL equation. White dot indicates the arbitrary initial $\mathbf{q_s}$ at the beginning of the optimization, and orange star indicates the converged solution $q_s^1 = -10.65$ and $q_s^2 = 2.12$. The x and y axes represent the positions of the first and second sensors, respectively. Since the 2 sensor positions are interchangeable a symmetry exists about the $y = x$ line.

2.2 Optimal Actuator Placement

2.2.1 Continuous-time Analysis

In linear system theory, the estimation problem as described in §2.1 is the *dual* of the control problem. Hence similar to §2.1, control theory provides a rigorous derivation for the optimal full-state feedback control policy given the linear system (2.4). If a cost metric J is defined such that

$$J = \lim_{T\to\infty} \frac{1}{2} \int_0^T \mathbf{x}^T \mathbf{Q_x} \mathbf{x} + \mathbf{u}^T \mathbf{Q_u} \mathbf{u} \, \mathrm{d}t, \tag{2.38}$$

then the optimal linear control $\mathbf{u}(t)$ which minimizes J is

$$\mathbf{u}(t) = -\mathbf{K}\mathbf{x}(t) \tag{2.39a}$$

$$= -\mathbf{Q_u}^{-1}\mathbf{B}^T\mathbf{Y}\mathbf{x}(t) \tag{2.39b}$$

where

$$\mathbf{0} = \mathbf{A}^T\mathbf{Y} + \mathbf{Y}\mathbf{A} - \mathbf{Y}\mathbf{B}\mathbf{Q_u}^{-1}\mathbf{B}^T\mathbf{Y} + \mathbf{Q_x}, \qquad \mathbf{Y}^T = \mathbf{Y}. \tag{2.40}$$

The time-invariant solution of $\mathbf{K}$ is obtained by solving the infinite-time solution of (2.40), which is a CARE. A linear system that implements the linear feedback law described in (2.39b) can be shown to have the optimal cost

$$J = \frac{1}{2}\mathbf{x}_0^T\mathbf{Y}\mathbf{x}_0, \tag{2.41}$$

where $\mathbf{x}_0$ is any inital condition. This implies one could manipulate $\mathbf{B}$ through $\mathbf{q_a}$ to minimize (2.41), which can be achieved by minimizing the eigenvalues of $\mathbf{Y}$. There are many ways to minimize different aspects of the $\mathbf{Y}$ eigenvalues (see §2.2.3), to be consistent to §2.1 a cost function analogous to (2.8a) is proposed:

$$\begin{aligned} &\min_{\mathbf{q_a}} J(\mathbf{q_a}) = \mathrm{trace}(\mathbf{Y}), \\ &\quad \text{s.t. } \mathbf{0} = \mathbf{A}^T\mathbf{Y} + \mathbf{Y}\mathbf{A} - \mathbf{Y}\mathbf{B}\mathbf{Q_u}^{-1}\mathbf{B}^T\mathbf{Y} + \mathbf{Q_x}. \end{aligned} \tag{2.42}$$

Using analysis similar to §2.2.1 the optimal $\mathbf{q_a}$ can be computed iteratively. When perturbations are applied to an initial nominal $\mathbf{q_a}$, $\mathbf{B}$, $\mathbf{Y}$, and J are also perturbed.

The first-order perturbations are:

$$J' = \text{trace}(\mathbf{Y}') = \langle \mathbf{I}, \mathbf{Y}' \rangle, \tag{2.43a}$$

$$(\mathbf{A} - \mathbf{BK})^T \mathbf{Y}' + \mathbf{Y}'(\mathbf{A} - \mathbf{BK}) = \mathbf{YB}'\mathbf{K} + \mathbf{K}^T(\mathbf{B}')^T\mathbf{Y}, \tag{2.43b}$$

$$\mathbf{B}' = \left(\frac{\mathrm{d}\mathbf{B}}{\mathrm{d}\mathbf{q_a}}\right)^T \mathbf{q_a}', \tag{2.43c}$$

where it is assumed that $\mathbf{Q_x}' = \mathbf{0}$ and $\mathbf{Q_u}' = \mathbf{0}$. Equation (2.43b) is simplified by introducing the linear operators $\mathrm{L}(\mathbf{Y}')$ and $\mathrm{M}(\mathbf{B}')$:

$$\mathrm{L}(\mathbf{Y}') = \mathrm{M}(\mathbf{B}'), \tag{2.44a}$$

$$\mathrm{L}(\mathbf{Y}') \triangleq (\mathbf{A} - \mathbf{BK})^T \mathbf{Y}' + \mathbf{Y}'(\mathbf{A} - \mathbf{BK}), \tag{2.44b}$$

$$\mathrm{M}(\mathbf{B}') \triangleq \mathbf{YB}'\mathbf{K} + \mathbf{K}^T(\mathbf{B}')^T\mathbf{Y}. \tag{2.44c}$$

By using the matrix adjoint variable $\mathbf{S}$, inner product (2.12), and trace identities it can be shown that $\mathbf{S}$ satisfies the CALE

$$\underbrace{(\mathbf{A} - \mathbf{BK})\mathbf{S} + \mathbf{S}(\mathbf{A} - \mathbf{BK})^T}_{\mathrm{L}^*(\mathbf{S})} = \mathbf{I}. \tag{2.45}$$

Therefore (2.43a) can be rewritten into

$$J' = \text{trace}\left(2\mathbf{KSYB}'\right), \tag{2.46a}$$

$$\nabla_{q_a^i} J \triangleq \text{trace}\left(2\mathbf{KSY}\frac{\mathrm{d}\mathbf{B}}{\mathrm{d}q_a^i}\right), \qquad \left[\nabla_{q_a^1} J, \quad \cdots \quad, \nabla_{q_a^{l_a}} J\right] \triangleq \nabla_{\mathbf{q_a}} J. \tag{2.46b}$$

Note the duality with the continuous optimal sensor placement solution.

2.2.2 Discrete-time Analysis

A discrete-time control formulation for the gradient, similar to §2.2.1, can be found by defining the appropriate discrete-time cost function

$$J = \lim_{T\to\infty} \frac{1}{2} \sum_{k=1}^{T} \mathbf{x}_k^T \mathbf{Q_x} \mathbf{x}_k + \mathbf{u}_k^T \mathbf{Q_u} \mathbf{u}_k. \tag{2.47}$$

The optimal control sequence $\mathbf{u}_k$ which minimizes this cost satisfies

$$\mathbf{u}_k = -\mathbf{K}\mathbf{x}_k \tag{2.48a}$$

$$= -\left(\mathbf{Q_u} + \mathbf{G}^T\mathbf{YG}\right)^{-1} \mathbf{G}^T\mathbf{YF}\mathbf{x}_k, \tag{2.48b}$$

where $\mathbf{Y}$ is the solution to the following DARE.

$$\mathbf{Y} = \mathbf{F}^T\mathbf{Y}\mathbf{F} - \mathbf{F}^T\mathbf{Y}\mathbf{G}(\mathbf{Q_u} + \mathbf{G}^T\mathbf{Y}\mathbf{G})^{-1}\mathbf{G}^T\mathbf{Y}\mathbf{F} + \mathbf{Q_x}, \qquad \mathbf{Y}^T = \mathbf{Y}. \tag{2.49}$$

For the same reasons listed in §2.2.1, the infinite-time cost function

$$\begin{aligned} &\min_{\mathbf{q_a}} J(\mathbf{q_a}) = \text{trace}(\mathbf{Y}), \\ &\quad \text{s.t. } \mathbf{Y} = \mathbf{F}^T\mathbf{Y}\mathbf{F} - \mathbf{F}^T\mathbf{Y}\mathbf{G}(\mathbf{Q_u} + \mathbf{G}^T\mathbf{Y}\mathbf{G})^{-1}\mathbf{G}^T\mathbf{Y}\mathbf{F} + \mathbf{Q_x}, \end{aligned} \tag{2.50}$$

is minimized. Applying perturbation analysis, defining the same matrix variable $\mathbf{S}$ and inner product (2.12), performing the necessary rearrangement using the trace identity and substitutions, one could show that $\mathbf{S}$ must satisfy a DALE

$$\underbrace{\mathbf{S} - (\mathbf{F} + \mathbf{G}\mathbf{K})\mathbf{S}(\mathbf{F}^T + \mathbf{K}^T\mathbf{G}^T)}_{\mathrm{L}^*(\mathbf{S})} = \mathbf{I}, \tag{2.51}$$

and the local gradient is defined as

$$\nabla_{q_a^i} J = \text{trace}\left(2\mathbf{K}\mathbf{S}(\mathbf{F}^T + \mathbf{K}^T\mathbf{G}^T)\mathbf{Y}\frac{\mathrm{d}\mathbf{G}}{\mathrm{d}q_a^i}\right), \qquad \left[\nabla_{q_a^1} J, \quad \cdots \quad, \nabla_{q_a^{l_a}} J\right] \triangleq \nabla_{\mathbf{q_a}} J. \tag{2.52}$$

2.2.3 Different Costs

The different costs considered in §2.1.3 can equally applied in the control case, except the interpretations to the costs are now different. Assuming $\mathbf{x}_0$ is unit length, then performing eigen-decomposition on $\mathbf{Y}$ and decompose $\mathbf{x}_0$ as linear combinations of the eigenvectors, it is clear from (2.41) that

$$\begin{aligned} J &= \frac{1}{2}[a_1\mathbf{e}_1 + \cdots + a_n\mathbf{e}_n]^T\mathbf{Y}[a_1\mathbf{e}_1 + \cdots + a_n\mathbf{e}_n] \\ &= \frac{1}{2}[a_1, \cdots, a_n]\text{diag}(\lambda_1, \cdots, \lambda_n)[a_1, \cdots, a_n]^T, \\ &= \frac{1}{2}\sum_{i=1}^{n} a_i^2\lambda_i, \end{aligned} \tag{2.53}$$

where $\mathbf{Y}\mathbf{e}_i = \lambda_i\mathbf{e}_i$, $a_i = \mathbf{x}_0^T\mathbf{e}_i$, and $\mathbf{e}_i \perp \mathbf{e}_j, i \neq j$ because $\mathbf{Y}$ is symmetric.

Therefore minimizing the A-optimality criterion means minimizing a conservative upper bound of (2.53), where $a_i = 1$. The E-optimality criterion on the

other hand minimizes the maximum eigenvalue, which in this context minimizes J with respect to the *worst* possible $\mathbf{x}_0$; using this criterion intrinsically build-in robustness against initial conditions. Currently it is unclear what is the physical interpretation of the D-optimality criterion in this context.

2.2.4 Numerical Experiment

Setup

The same GL equation in §2.1.4 is used to perform optimal actuator placement. To mirror the 2-sensor placement problem, a 2-actuator placement problem is solved here.

Results

Results for the actuator placement problem are summarized by Figure 2.5 and Figure 2.6. Figure 2.5 visualizes the optimization surface for various actuator locations, and the path taken by the algorithm to minimize (2.41). Similar to the estimation problem this result is symmetric about the $q_a^2 = q_a^1$ line, and the solution is a global minimizer because there is only one minimum. If the penalty matrices $\mathbf{Q_x}$ or $\mathbf{Q_u}$ are chosen such that one actuator is biased against the other (e.g. different state or control penalties), the symmetry would be broken. The optimal solution places both actuators inside the convectively unstable region. This suggests (and intuition validates) that actuators should be placed so that they allow the dissipative dynamics of the GL system to dampen the disturbances before introducing actuation within the unstable region. This interpretation is verified by figure 2.6 which shows the square-root of the variance of $\mathbf{P}$ of the controlled GL equation (with optimal sensor placement solution from §2.1.4 for estimation) throughout the domain. The two actuators are placed ahead of the convectively stable region, so that control effort is reduced by taking advantage of the natural dynamics of the GL equation to reduce the uncertainty even further.

It is known the combined optimal estimation/control problem in linear system theory can be solved by designing the optimal estimator and controller *sepa-*

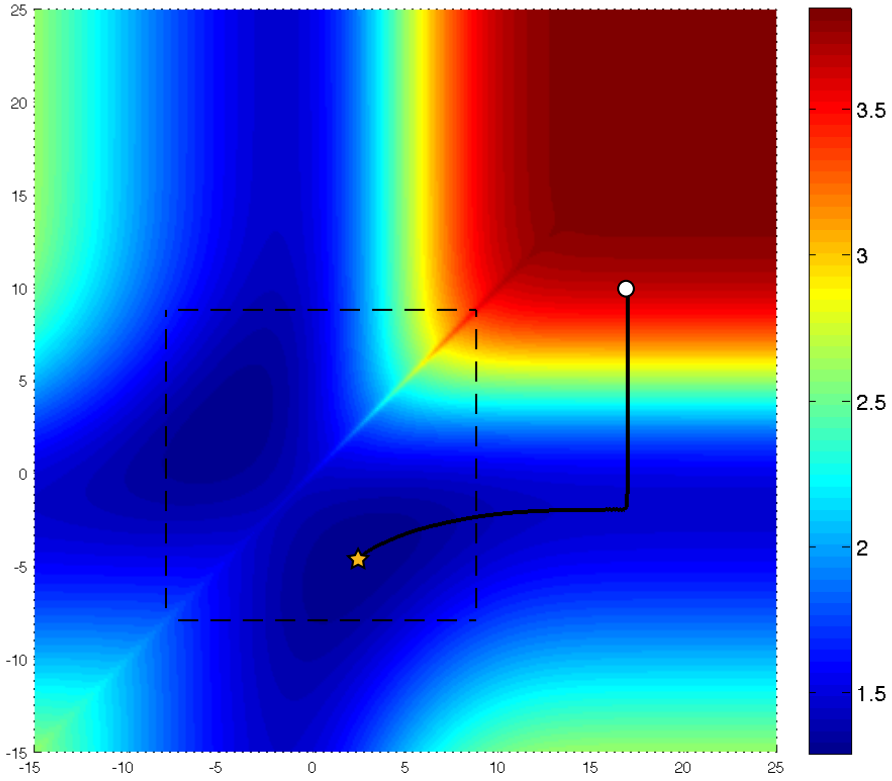


Figure 2.5: A $\log_{10}$ plot of the optimization surface for 2 actuator placement problem in the GL equation. Similarly to Fig. 2.4 the x and y axes represent the positions of the first and second actuator, respectively. White dot indicates the arbitrary initial $\mathbf{q_s}$ at the beginning of the optimization, and orange star indicates the converged solution at $q_a^1 = -4.6571$ and $q_a^2 = 2.3560$

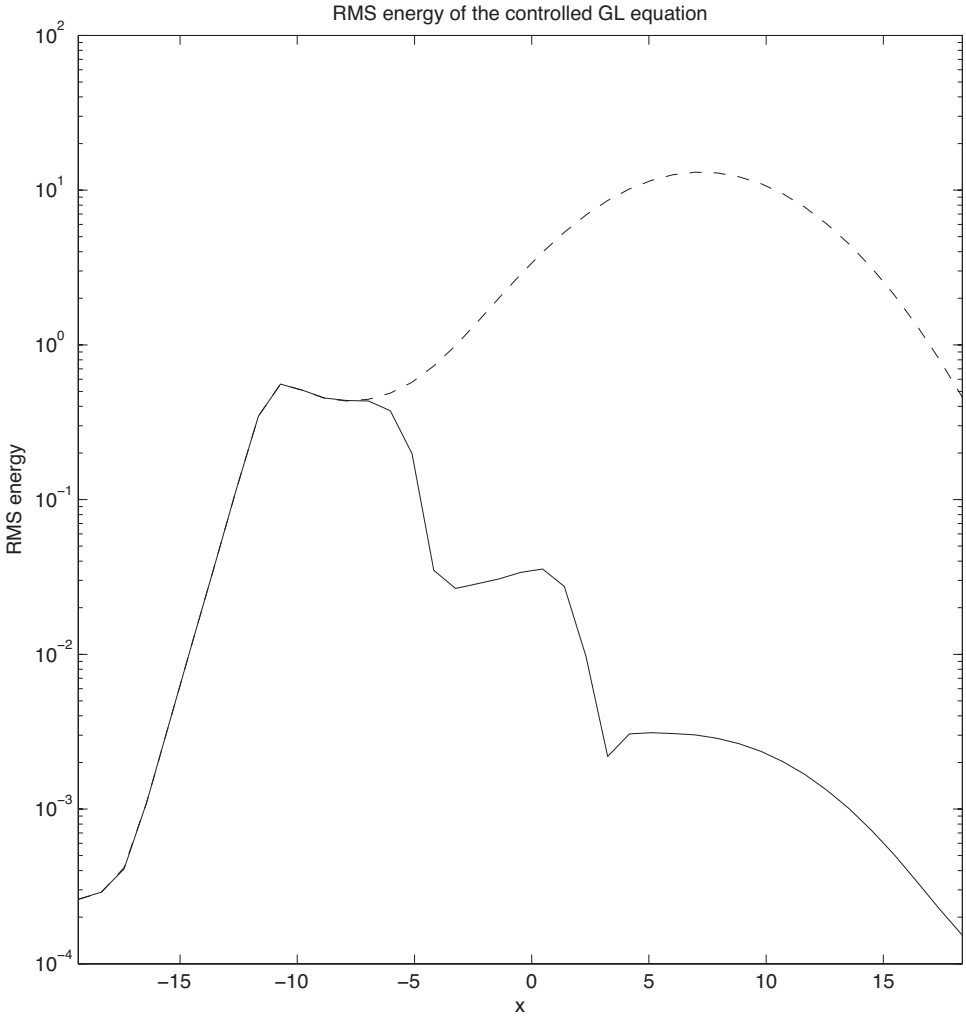


Figure 2.6: Square-root of the variance of **P** for the uncontrolled GL equation (dash line, the square-root of the diagonal of Figure 2.2) and the controlled GL equation (solid). Two sensors are placed at the solution from §2.1.4 for estimation, with stochastic disturbance at $q_w^1 = -11.0$.

rately; this is commonly referred as the *separation principle*. Similarly as shown in this example, the separation principle could also be employed to separately computing the optimal sensor/actuator placements. Chen & Rowley (2010) have considered a similar optimal sensor/actuator placement for subcritical GL equation. However there the authors found minimizing solutions of norms of continuous-time transfer functions, in the L_2/H_2 sense, by simultaneously optimizing *both* measurement sensor and actuator position. This is substantially different from the optimization functions considered in §2.1.1, where the measurement and actuation locations are optimized separately. It is likely that moving sensors to suboptimal estimation positions might lead to better performance in the controller, which would explain the dissimilarity between the solutions published by Chen & Rowley (2010). The method of extracting gradients, proposed in the previous section §2.1.1, could be applied to the work of Chen & Rowley (2010).

2.3 Summary/Discussion

In this chapter the infinite-time optimal sensor placement problem is solved by analytically computing the local gradient of some cost quantifying estimation quality with respect to the sensor states. The resulting gradient information is used for gradient-based optimization algorithms to iteratively converge a nominal solution toward an optimal solution. For completeness, the local gradient analysis are performed for both continuous- and discrete-time; also, various estimation quality measures are discussed and incorporated into the analysis. Numerical experiment with the Ginzburg-Landau (GL) equation is performed to find the optimal sensor placement, and the results show great improvements.

Because the control problem is the dual of the estimation problem in linear control theory, the same analysis is applied for finding the optimal actuator placement in both continuous- and discrete-time. The same GL equation is used to perform optimal actuator placement, and the results are sensible. Since separation principle allows computing the optimal estimator and controller separately, it suggests the optimal sensor and actuator placement could also be performed

separately.

Toward solving the actual AO problem as discussed in Chapter 1, sensor movement is required; furthermore a finite time interval formulation of the theories established in this chapter is in order. Both will be address in Chapter 3.

2.4 Acknowledgments

The solution for solving the infinite-time sensor placement problem in continuous time was first found by Chris Colburn. I simply extended the idea for sensor placement in discrete-time, and the equivalent extension for actuator placement. Chris is also credited for finding the different measures used in Information Theory to quantify estimation quality; he also implementing the theory on the Ginzburg-Landau equation from the code-base provided by Bagheri *et al.* (2009). The works in this chapter are taken from

- Colburn, C.H., Zhang, D., Bewley, T.R "*Adjoint-based Gradient Calculation for Infinite-time Optimal Sensor/Actuator Placement*", under preparation.

Chapter 3

Dynamic Adaptive Observation (DAO)

Toward solving the full Adaptive Observation (AO) problem presented in Chapter 1, in this chapter the theories established in Chapter 2 is build upon to incorporate sensor movements subjected to vehicle dynamics within a finite time interval. Before moving onto the actual formulation, existing AO algorithms are reviewed in §3.1, and the AO problem is formally introduced in §3.2. The actual analysis with performed in §3.3, with various generalizations to the resulting algorithm, dubbed Dynamic Adaptive Observation (DAO), presented in §3.4. In §3.5 two examples on how the DAO algorithm can be implemented are illustrated.

3.1 AO Algorithms Survey

The task of Adaptive Observation (AO) is to determine best future sensor positions/trajectories for estimation/forecast uncertainty reduction. This class of problems is considered a hybrid of problems from control and estimation theory, and proposed methods are either distributed or centralized in nature.

In typical distributed AO algorithms (see Martínez *et al.*, 2007; Laventall & Cortés, 2009; Stanković & Stipanović, 2009; Zhang & Leonard, 2010), each sensor vehicle has little knowledge of the sensed system, and deployment is planned locally. The hope is that simple local rules might lead to vehicle motions that

samples the system in such a way that improve estimation and forecast. Because decisions are made locally, these algorithms are easily deployable, scale nicely, and derive controls that inherently satisfy vehicle dynamic constraints. The majority of existing distributed AO algorithms reduce the AO problem to an optimal coverage, extremum seeking, or level-set tracking problem, Figure 3.1 through Figure 3.3 illustrate graphically the principle behind each approach. While these algorithms work adequately for certain applications, their performances degrade in large domains likely encountered in environmental flow problems. Furthermore since environmental flows are mainly driven by convection, state-couplings must be taken into account in order to produce accurate forecast. Such couplings are not considered in distributed AO algorithms. For example, if extremum seeking and level-set tracking are concurrently implemented on the BP Gulf spill disaster in 2011 in Figure 3.4, one would get a good estimate of where the oil plume is; however there is insufficient data to produce a good forecast because ocean current data outside of the oil plume is also required.

Therefore, it is beneficial to plan the sensor distributions more deliberately with a centralized AO strategy, where the underlying system model is exploited to target sensor positions that maximally reduce the estimation/forecast uncertainty. As a consequence, the centralized AO strategy is computationally intensive (since typically the underlying system model is very complex and require high fidelity simulation), and thus cannot be computed locally on the individual vehicles. Rather, the bulk of necessary computations must be done centrally on a supercomputer cluster, and the optimized vehicle trajectories (or waypoints selected along these trajectories) are periodically sent back to the sensor vehicles. Good examples on how AO is applied in real-life problems can be found in Snyder (1996); Langland *et al.* (1999); Szunyogh *et al.* (2000); Lermusiaux (2007). More detailed reviews on some popular centralized AO methods in the atmospheric science community are shown in Bishop (2000 (submitted)), while Lermusiaux (2007) reviews some in the oceanographic community. Since the problem-of-interest in this thesis deals with environmental problems that benefits from centralize AO, the remainder of the thesis shall focus on centralized AO.

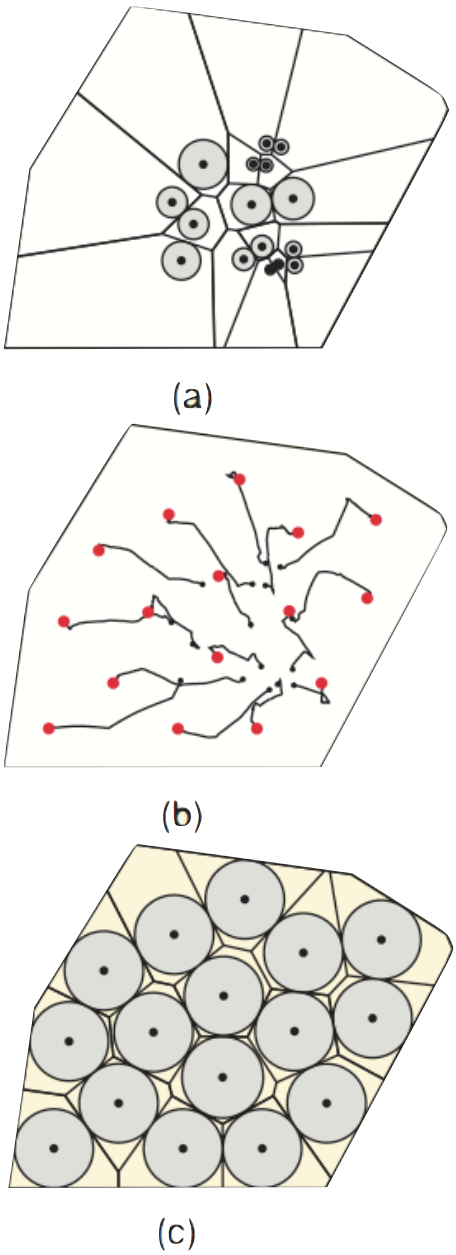


Figure 3.1: The optimal coverage approach. Initially the sensor vehicles are clumped together (a). As the algorithm progresses each vehicle acts like charged particles and repel each other (b). Eventually the vehicles are distributed evenly in the domain such that the coverage (the union of the circles in the figure) maximally cover the domain (c).

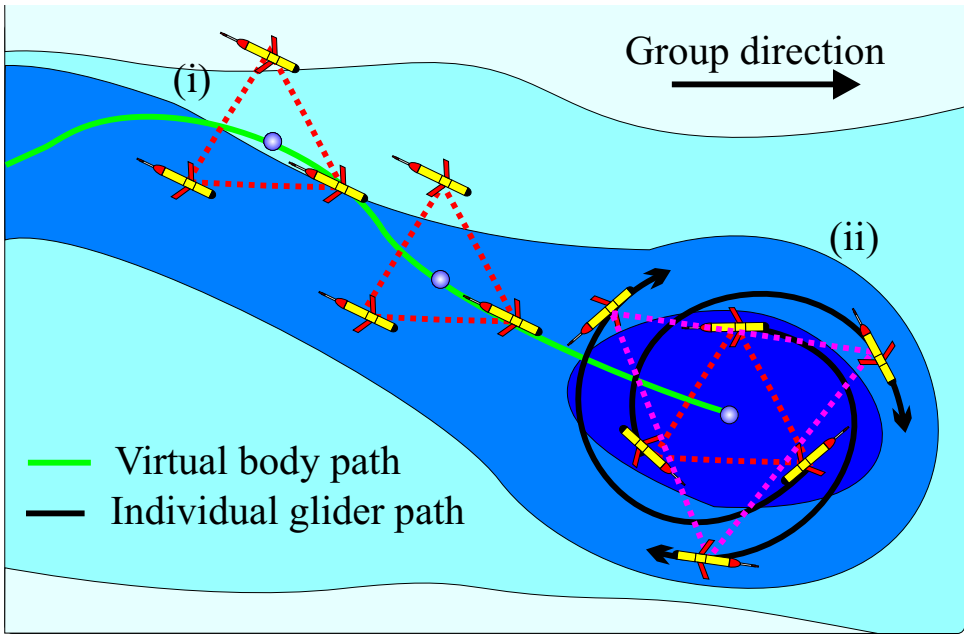


Figure 3.2: From the sensor group readings the local field gradient can be approximated. The vehicle fleet then move together such that the group centroid moves along the gradient toward the field extremum. The formation changes to loitering mode once the extremum is reached.

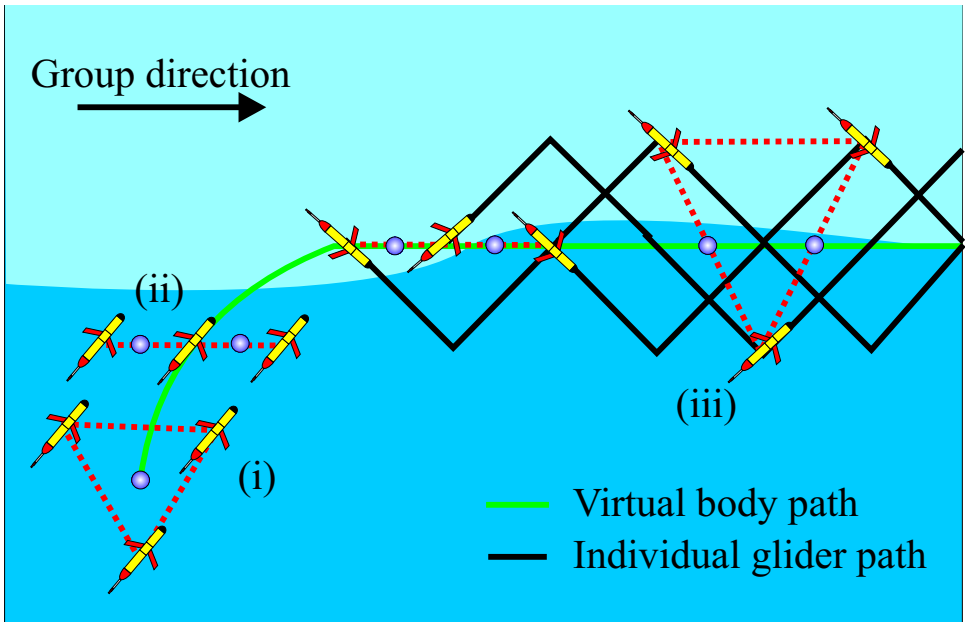


Figure 3.3: Instead of going toward the extremum, the group centroid could instead move *orthogonal* to the gradient and track a particular field level-set.

Figure 3.4: Satellite image of the 2010 BP oil spill off the Gulf of Mexico.

Centralized AO algorithms can be further categorized as either uncertainty-based or sensitivity-based. In sensitivity-based AO (see Langland & Rohaly, 1996; Buizza & Montani, 1999), adjoint analysis is used to reveal sensitive region of the domain that significantly influence a estimation/forecast uncertainty metric; these sensitive regions are then tagged for sensor vehicle deployment. One of the strengths of these algorithms centers on the relative speed and computational efficiency required for the calculation of the adjoint, as the sensitivity computation cost is on the same order as the forward system propagation computational cost. Another strength is that these algorithms leverage system flow-dependencies. However because the sensitivity analysis is performed about a particular forecast, the analysis is susceptible to forecast errors resulting from chaotic nonlinearities in the models. As a consequence, the computed sensitive regions maybe incorrect due to a wrong forecast. Furthermore, these algorithms do not address how the sensitive regions should be optimally probed by sensing vehicles.

Uncertainty-based AO algorithms (see Bishop *et al.*, 2001; Khare, 2004) take a different approach, where they seek a measurement location sequence that minimizes the estimation/forecast uncertainty. The uncertainty is typically quantified using error covariance; also in general only the low-rank approximation of the error covariance is used for computational feasibility. To this end, the Ensemble Kalman Filter (EnKF) (Evensen, 1994, 2003) is typically used to propagate and update the uncertainty. In particular, to achieve deterministic behavior and repeatability, the deterministic version of the EnKF, the Ensemble Square-Root Filter (ESRF) (Whitaker & Hamill, 2002) is used. Tippett *et al.* (2003) shows the Ensemble Transform Kalman Filter (ETKF) in Bishop *et al.* (2001) and the Ensemble Adjusted Kalman Filter (EAKF) in Khare (2004) are variants of the ESRF. The best measurement location sequence is found by searching a set of all possible sequences. The anticipated uncertainty associated with measurement location sequence in the set is computed, and the one that produces the lowest uncertainty is used. Performing this search may take significant amount of time if the searching set is large. For example, there are $\binom{100}{3}^2 > 26 \times 10^9$ possible sequences for 3 vehicles with 2 measurement times in a domain of 100 possible

measurement points. Bishop *et al.* (2001) commented that since each forecast uncertainty is independently evaluated, the evaluations are *embarrassingly parallel*; thus the total computation time is inversely proportional to the computational resources available. Furthermore with sub-optimality, Bishop *et al.* also noted that the set can be significantly reduced by performing the search *serially*; that is, find the locations one at a time, assuming the previously placed measurement locations are fixed. In Yilmaz *et al.* (2008), set reduction is achieved through adding linear constraints that simulate vehicle dynamics in the searching set, so that only the neighborhoods of the vehicles are exhaustively searched.

There are also hybrid AO algorithms that use ideas from both the sensitivity- and uncertainty-based AO methods. One such algorithm is the Ensemble Sensitivity presented in Ancell & Hakim (2007), where ensemble sensitivity is computed using the uncertainty information. Another method is presented in Bishop (2000 (submitted)), which extends the works in Baker & Daley (2000) such that sensitivity information is used to compute waypoints that yields the least forecast uncertainty.

All aforementioned AO algorithms do not fully consider vehicle dynamics. This is perhaps because the various environmental flow models to which centralized AO algorithms have traditionally been applied are essentially static when compared with the motion time scales of the vehicles. In order to overcome this problem, for example when ETKF is used to select weather reconnaissance aircraft flight path in the Winter Storm Reconnaissance (WSR) programs, Majumdar *et al.* (2002) considered about 40 pre-approved feasible flight paths and select among them. Vehicle dynamics are partially incorporated in Yilmaz *et al.* (2008) as linear constraints, but because linear constraints cannot model complex vehicle dynamics, overly conservative constraints are imposed to ensure a dynamically feasible solution, resulting in unnecessarily sluggish vehicle trajectories. As discussed in Chapter 1, the problem in this thesis has comparable dynamics between the underlying system and sensor vehicles; therefore a quasi-static assumption is invalide and existing AO algorithms do not work. Thus, a new centralized AO method algorithm that incorporates vehicle dynamics is needed to solved the problem.

3.2 AO Problem Formulation

The AO problem with moving sensor vehicles is formally established in full here. Suppose a continuous-time nonlinear evolution equation that describes certain physical phenomenon $\mathbf{x}(t) \in \mathbb{R}^n$ within a domain:

$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = f(\mathbf{x}(t), \mathbf{d}(t), \mathbf{w}(t)), \qquad \mathbf{w}(t) \sim N(0, \mathbf{W}), \tag{3.1}$$

where $\mathbf{w}(t)$ models random external disturbance and model uncertainties, and $\mathbf{d}(t)$ is the known external input. There are also M sensor vehicles within the same domain. The i-th vehicle's continuous-time dynamics, with vehicle state $\mathbf{q}^i(t) \in \mathbb{R}^l$ and control $\mathbf{u}^i(t) \in \mathbb{R}^c$, is

$$\frac{\mathrm{d}\mathbf{q}^i(t)}{\mathrm{d}t} = g(\mathbf{q}^i(t), \mathbf{u}^i(t)). \tag{3.2}$$

Each vehicle move about the domain *continuously*, while taking measurements $\mathbf{y}_k^i$ at *discrete* times t_k:

$$\mathbf{y}_k^i = h_k^i(\mathbf{x}_k, \mathbf{q}_k^i) + \mathbf{v}_k^i, \qquad \mathbf{v}_k^i \sim N(0, \mathbf{R}_k^i(\mathbf{q}^i)). \tag{3.3}$$

Equation (3.3) reflects that vehicle states such as position, heading, and velocity typically affect how the measurements are taken and measurement noise statistics. For convenience, the vehicle state dependence notation is dropped from hereon, with understanding that dependence is implied when using $h_k^i(\mathbf{x}_k)$ and $\mathbf{R}_k^i$. The collection of measurements and corresponding statics from all vehicles are defined as:

$$\mathbf{y}_k = \begin{bmatrix} \mathbf{y}_k^1 \\ \vdots \\ \mathbf{y}_k^M \end{bmatrix}, \quad h_k(\mathbf{x}_k) = \begin{bmatrix} h_k^1(\mathbf{x}_k) \\ \vdots \\ h_k^M(\mathbf{x}_k) \end{bmatrix}, \quad \mathbf{R}_k = \begin{bmatrix} \mathbf{R}_k^1 & & 0 \\ & \ddots & \\ 0 & & \mathbf{R}_k^M \end{bmatrix}. \tag{3.4}$$

For practical purposes, a hybrid Kalman Filter (KF) which utilizing the continuous-time KF during time-update phase and the discrete-time KF during measurement update phase, is assumed for estimation. The state estimate of the filter is $\hat{\mathbf{x}}$, and the estimation error covariance is $\mathbf{P}$. $\mathbf{P}$ propagates between measurements in continuous-time according to continuous-time KF covariance propagation

formula

$$\frac{\mathrm{d}\mathbf{P}(t)}{\mathrm{d}t} = \mathbf{A}\mathbf{P}(t) + \mathbf{P}(t)\mathbf{A}^T + \mathbf{B}\mathbf{W}\mathbf{B}^T, \tag{3.5}$$

where $\mathbf{A}$ and $\mathbf{B}$ are respectively (3.1) linearized about $\hat{\mathbf{x}}(t)$ and $\mathrm{E}\{\mathbf{w}(t)\} = \mathbf{0}$. Without loss of generality, from hereon it is assumed $\mathbf{B} = \mathbf{I}$. $\mathbf{P}$ is updated sequentially with each additional measurement according to the discrete-time KF update formula discussed in §2.1.2

$$\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{L}_k\mathbf{H}_k)\mathbf{P}_k^-, \qquad \mathbf{L}_k = \mathbf{P}_k^-\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k)^{-1}, \tag{3.6}$$

where $\mathbf{H}_k$ is $h_k(\mathbf{x}_k)$ in (3.4) linearized about $\hat{\mathbf{x}}_k$, and $\mathbf{L}_k$ is the optimal filter gain.

The AO problem in this thesis is framed as followed:

At the start of time t_0, the initial vehicle state $\mathbf{q}_0^i$ and estimation error covariance $\mathbf{P}_0$ are known. Design a control trajectory $\mathbf{u}^i(t)$ for each vehicle over the time window $[t_0, t_K]$ to minimize a cost function balancing control effort and forecast quality at the final time t_F, where $t_F \geq t_K$, conditioned on the measurements taken by the vehicles at times $\{t_1, t_2, \cdots, t_K\}$.

For simplicity, a quadratic measure for control effort is chosen and a modified A-optimality criterion to quantify forecast accuracy summed together to form a cost function:

$$J = \mathrm{trace}(\mathbf{T}\mathbf{P}_F) + \frac{1}{2}\sum_{i=1}^{M}\int_0^{t_K} \mathbf{u}^i(t)^T\mathbf{Q}_\mathbf{u}\mathbf{u}^i(t)\,\mathrm{d}t, \tag{3.7}$$

where $\mathbf{T}$ is a diagonal matrix and $\mathbf{Q}_\mathbf{u}$ is a positive-definite symmetric matrix. Together with the trace( ) operator, $\mathbf{T}$ weights the diagonal of $\mathbf{P}_F$, which physically can be interpreted as masking specific uncertainty regions in the domain (such as urban areas). Extensions to (3.7) are discussed in §3.4. Figure 3.5 illustrates the mixed continuous/discrete time formulation, and the relationships between the several quantities involved. Note, as $\mathbf{P}(t)$ is updated during each measurement, the trajectory $\mathbf{P}(t)$ is *piecewise smooth*. Also note, since $\mathbf{u}^i(t)$ affects the cost function non-linearly, J is in general non-convex with many local minimum; global optimization of this cost function cannot be guaranteed with a computationally tractable algorithm. Therefore, the minimizing solution is sought via an iterative approach by initially assuming a nominal control trajectory for each vehicle and

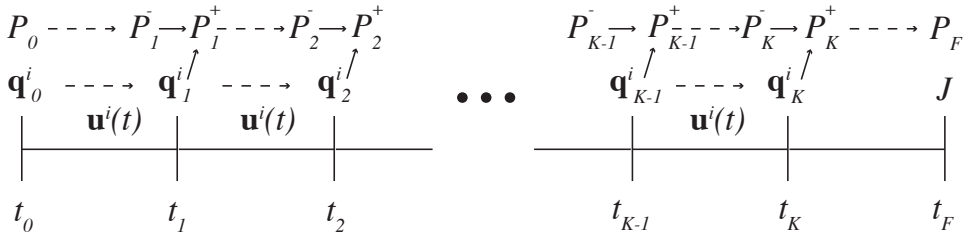


Figure 3.5: Cartoon illustrating the problem formulation. $\mathbf{u}^i(t)$ affect the *continuous-time* evolution of sensor vehicle trajectories $\mathbf{q}^i(t)$. In turn, the sensor vehicle positions at the measurement times, $\mathbf{q}^i_k$, affect the *discrete-time* update of the estimation error covariance $\mathbf{P}_k$; this covariance otherwise evolves continuously between the measurements, and between t_K and t_F. The cost J depends on $\mathbf{P}_F$ and $\mathbf{u}^i(t)$ within time window $[t_0, t_K]$; a set of controls $\mathbf{u}^i(t)$ is sought to minimize this cost. Dashed arrows denote continuous-time propagations; solid arrows denote discrete-time updates.

compute a local gradient $\nabla_{\mathbf{u}^i(t)} J$; the gradient is used in gradient-based iterative optimization routines. The following shows how such gradient can be computed using *adjoint analysis*.

3.3 Computing the Gradient

The principle idea behind the procedure to analytically derive $\nabla_{\mathbf{u}^i(t)} J$ is the same as Chapter 2, where perturbations are performed and adjoint analysis is used to extract the gradient. Applying perturbations to a set of nominal control trajectories causes a chain reaction that perturbs other variables; the first-order

perturbations of these variables are:

$$\frac{\mathrm{d}\mathbf{q}^i(t)'}{\mathrm{d}t} = \mathbf{F}^i\mathbf{q}^i(t)' + \mathbf{G}^i\mathbf{u}^i(t)', \qquad \mathbf{q}_0^i{}' = 0, \tag{3.8a}$$

$$\frac{\mathrm{d}\mathbf{P}(t)'}{\mathrm{d}t} = \mathbf{A}\mathbf{P}(t)' + \mathbf{P}(t)'\mathbf{A}^T, \qquad P_0' = 0, \tag{3.8b}$$

$$\begin{aligned}(\mathbf{P}_k^+)' = (\mathbf{P}_k^-)' - ((\mathbf{P}_k^-)'\mathbf{H}_k^T + \mathbf{P}_k^-(\mathbf{H}_k')^T)\mathbf{L}_k^T - \mathbf{L}_k(\mathbf{H}_k(\mathbf{P}_k^-)' + \mathbf{H}_k'\mathbf{P}_k^-) \\ + \mathbf{L}_k(\mathbf{H}_k'\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{H}_k(\mathbf{P}_k^-)'\mathbf{H}_k^T + \mathbf{H}_k\mathbf{P}_k^-(\mathbf{H}_k')^T + \mathbf{R}_k')\mathbf{L}_k^T,\end{aligned} \tag{3.8c}$$

$$J' = \mathrm{trace}(\mathbf{T}\mathbf{P}_F') + \sum_{i=1}^{M}\int_0^{t_K} \mathbf{u}^i(t)^T\mathbf{Q}_\mathbf{u}\mathbf{u}^i(t)'\,\mathrm{d}t, \tag{3.8d}$$

$$\mathbf{R}_k' = \begin{bmatrix}(\mathbf{R}_k^1)' & & 0 \\ & \ddots & \\ 0 & & (\mathbf{R}_k^M)'\end{bmatrix}, \quad \mathbf{H}_k' = \begin{bmatrix}(\mathbf{H}_k^1)' \\ \vdots \\ (\mathbf{H}_k^M)'\end{bmatrix}, \tag{3.8e}$$

$$(\mathbf{R}_k^i)' = \left(\frac{\mathrm{d}\mathbf{R}_k^i}{\mathrm{d}\mathbf{q}_k^i}\right)^T(\mathbf{q}_k^i)', \quad (\mathbf{H}_k^i)' = \left(\frac{\mathrm{d}\mathbf{H}_k^i}{\mathrm{d}\mathbf{q}_k^i}\right)^T(\mathbf{q}_k^i)', \tag{3.8f}$$

where $\mathbf{F}^i$ and $\mathbf{G}^i$ (recycled variables, not to be confused with the discretization of $\mathbf{A}$ and $\mathbf{B}$ in §2.1.2) are (3.2) linearized about $\mathbf{q}^i(t)$ and $\mathbf{u}^i(t)$ respectively, and $(\mathbf{q}_0^i)'$, $\mathbf{P}_0'$, and $\mathbf{W}'$ are zero because they are not affected by perturbations in $\mathbf{u}^i(t)$. Note that $\mathrm{d}\mathbf{R}_k^i/\mathrm{d}\mathbf{q}_k^i$ and $\mathrm{d}\mathbf{H}_k^i/\mathrm{d}\mathbf{q}_k^i$ are rank-3 tensors that contract by the inner product with $(\mathbf{q}_k^i)'$ to yield matrices $(\mathbf{R}_k^i)'$ and $(\mathbf{H}_k^i)'$. The purpose of the perturbations is to repose (3.8d) such that

$$J' = \sum_{i=1}^{M}\int_0^{t_K} \left(\nabla_{\mathbf{u}^i(t)}J\right)^T\mathbf{u}^i(t)'\,\mathrm{d}t. \tag{3.9}$$

Equation (3.9) is similar to (3.8d) except for the $\mathrm{trace}(\mathbf{T}\mathbf{P}_F')$ term; the rest of the formulation focuses on rewriting this in a similar form.

The descriptions in (3.8a) and (3.8b) are simplified by introducing linear operators $\mathrm{L}(\mathbf{P}')$, $\mathrm{M}(\mathbf{q}')^i$, and $\mathrm{B}(\mathbf{u}')^i$:

$$\mathrm{L}(\mathbf{P}') \triangleq \frac{\mathrm{d}\mathbf{P}(t)'}{\mathrm{d}t} - \mathbf{A}\mathbf{P}(t)' - \mathbf{P}(t)'\mathbf{A}^T, \tag{3.10a}$$

$$\mathrm{M}(\mathbf{q}')^i \triangleq \frac{\mathrm{d}\mathbf{q}^i(t)'}{\mathrm{d}t} - \mathbf{F}^i\mathbf{q}^i(t)', \tag{3.10b}$$

$$\mathrm{B}(\mathbf{u}')^i \triangleq \mathbf{G}^i\mathbf{u}^i(t)', \tag{3.10c}$$

so that $\mathrm{L}(\mathbf{P}') = 0$ by (3.8b) and $\mathrm{M}(\mathbf{q}')^i = \mathrm{B}(\mathbf{u}')^i$ by (3.8a). An adjoint variable $\mathbf{S}(t) \in \mathbb{R}^{n\times n}$ is defined over the time window $[t_K^+, t_F]$ and an adjoint identity based on a relevant inner product is defined:

$$\begin{aligned} \langle \mathbf{S}, \mathrm{L}(\mathbf{P}')\rangle_{t_K^+,t_F} &= \langle \mathrm{L}^*(\mathbf{S}), \mathbf{P}'\rangle_{t_K^+,t_F} + a, \\ \langle \mathbf{X}, \mathbf{Y}\rangle_{t_K^+,t_F} &\triangleq \int_{t_K^+}^{t_F} \mathrm{trace}(\mathbf{X}(t)^T\mathbf{Y}(t))\,\mathrm{d}t. \end{aligned} \tag{3.11}$$

Using integration by parts, it can be shown that

$$\mathrm{L}^*(\mathbf{S}) = -\frac{\mathrm{d}\mathbf{S}(t)}{\mathrm{d}t} - \mathbf{A}^T\mathbf{S}(t) - \mathbf{S}(t)\mathbf{A}, \tag{3.12a}$$

$$a = \mathrm{trace}\left(\mathbf{S}_F^T\mathbf{P}_F'\right) - \mathrm{trace}\left((\mathbf{S}_K^+)^T(\mathbf{P}_K^+)'\right). \tag{3.12b}$$

Taking $\mathrm{L}^*(\mathbf{S}) = \mathbf{0}$ and $\mathbf{S}_F^T = \mathbf{T}$, (3.8d) is rewritten using relationships established in (3.11) and (3.12) into

$$J' = \mathrm{trace}((\mathbf{S}_K^+)^T(\mathbf{P}_K^+)') + \sum_{i=1}^{M}\int_0^{t_K} \mathbf{u}^i(t)^T\mathbf{Q}_\mathbf{u}\mathbf{u}^i(t)'\,\mathrm{d}t. \tag{3.13}$$

Note by setting $\mathrm{L}^*(\mathbf{S}) = \mathbf{0}$ and $\mathbf{S}_F^T = \mathbf{T}$, this is equivalent to defining a backward-in-time evolution equation for $\mathbf{S}(t)$ such that

$$\frac{\mathrm{d}\mathbf{S}(t)}{\mathrm{d}t} = -\mathbf{A}^T\mathbf{S}(t) - \mathbf{S}(t)\mathbf{A}, \tag{3.14}$$

with starting condition $\mathbf{S}_F^T = \mathbf{T}$. By the special structure of (3.14) and $\mathbf{S}_F$, it is clear that $\mathbf{S}_K^+$ is also symmetric in (3.13).

Substituting (3.8c), (3.8e), and (3.8f) into $(\mathbf{P}_K^+)'$ in (3.13) and leveraging the trace identity $\mathrm{trace}(\mathbf{AB}) = \mathrm{trace}(\mathbf{BA}) = \mathrm{trace}(\mathbf{A}^T\mathbf{B}^T)$, the $(\mathbf{P}_K^-)'$ and $(\mathbf{q}_K^i)'$ terms are gathered to the right. Equation (3.13) now becomes

$$\begin{aligned} J' = &\,\mathrm{trace}\left((\mathbf{I} - \mathbf{H}_K^T\mathbf{L}_K^T)\mathbf{S}_K^+(\mathbf{I} - \mathbf{L}_K\mathbf{H}_K)(\mathbf{P}_K^-)'\right) \\ &+ \mathrm{trace}\left(2\mathbf{P}_K^-(\mathbf{H}_K^T\mathbf{L}_K^T - \mathbf{I})\mathbf{S}_K^+\mathbf{L}_K\mathbf{H}_K'\right) \\ &+ \mathrm{trace}\left(\mathbf{L}_K^T\mathbf{S}_K^+\mathbf{L}_K\mathbf{R}_K'\right) + \sum_{i=1}^{M}\int_0^{t_K} \mathbf{u}^i(t)^T\mathbf{Q}_\mathbf{u}\mathbf{u}^i(t)'\,\mathrm{d}t, \end{aligned} \tag{3.15a}$$

and leveraging the block matrix structure of $\mathbf{H}'_K$ and $\mathbf{R}'_K$ in (3.8e), it could further be simplified to

$$\begin{aligned} J' = &\operatorname{trace}\left((\mathbf{I}-\mathbf{H}_K^T\mathbf{L}_K^T)\mathbf{S}_K^+(\mathbf{I}-\mathbf{L}_K\mathbf{H}_K)(\mathbf{P}_K^-)'\right) \\ &+\sum_{i=1}^{M}\operatorname{trace}\left([2\mathbf{P}_K^-(\mathbf{H}_K^T\mathbf{L}_K^T-\mathbf{I})\mathbf{S}_K^+\mathbf{L}_K]_i\left(\frac{\mathrm{d}\mathbf{H}_K^i}{\mathrm{d}\mathbf{q}_K^i}\right)\right)^T(\mathbf{q}_K^i)' \\ &+\sum_{i=1}^{M}\operatorname{trace}\left((\mathbf{L}_K^T\mathbf{S}_K^+\mathbf{L}_K)_{ii}\left(\frac{\mathrm{d}\mathbf{R}_K^i}{\mathrm{d}\mathbf{q}_K^i}\right)\right)^T(\mathbf{q}_K^i)' \\ &+\sum_{i=1}^{M}\int_0^{t_K}\mathbf{u}^i(t)^T\mathbf{Q}_\mathbf{u}\mathbf{u}^i(t)'\,\mathrm{d}t, \end{aligned} \tag{3.15b}$$

where $(\mathbf{L}_K^T\mathbf{S}_K^+\mathbf{L}_K)_{ii}$ denotes the (i,i) block of $M\times M$ block matrix $\mathbf{L}_K^T\mathbf{S}_K^+\mathbf{L}_K$ and $[2\mathbf{P}_K^-(\mathbf{H}_K^T\mathbf{L}_K^T-\mathbf{I})\mathbf{S}_K^+\mathbf{L}_K]_i$ denotes the i-th column block of $1\times M$ block matrix $2\mathbf{P}_K^-(\mathbf{H}_K^T\mathbf{L}_K^T-\mathbf{I})\mathbf{S}_K^+\mathbf{L}_K$. Note the $(\,)^-$ superscript on $(\mathbf{q}_K^i)'$ is dropped because the $\mathbf{q}^i(t)$ trajectory is smooth and not prior or posterior distinction is required.

If the same inner product and adjoint identity as in (3.11) are defined, but now over the time window $[t_{K-1}^+, t_K]$, and if in addition M adjoint vectors $\mathbf{r}^i(t)\in\mathbb{R}^l$ are defined over the same time window with the appropriate adjoint identity

$$\begin{aligned} \langle\!\langle \mathbf{r}^i, \mathrm{M}(\mathbf{q}')^i\rangle\!\rangle_{t_{K-1}^+,t_K} &= \langle\!\langle \mathrm{M}^*(\mathbf{r})^i, \mathbf{q}^{i\,\prime}\rangle\!\rangle_{t_{K-1}^+,t_K} + b^i, \\ \langle\!\langle \mathbf{x},\mathbf{y}\rangle\!\rangle_{t_{K-1}^+,t_K} &\triangleq \int_{t_{K-1}^+}^{t_K}\mathbf{x}(t)^T\mathbf{y}(t)\,\mathrm{d}t, \end{aligned} \tag{3.16a}$$

$$\mathrm{M}^*(\mathbf{r})^i = -\frac{\mathrm{d}\mathbf{r}^i(t)}{\mathrm{d}t} - (\mathbf{F}^i)^T\mathbf{r}^i(t), \tag{3.16b}$$

$$b^i = (\mathbf{r}_K^i)^T(\mathbf{q}_K^i)' - (\mathbf{r}_{K-1}^{i+})^T(\mathbf{q}_{K-1}^i)', \tag{3.16c}$$

then by letting

$$\mathrm{L}^*(\mathbf{S}) = \mathbf{0}, \qquad \mathrm{M}^*(\mathbf{r})^i = \mathbf{0}, \tag{3.17a}$$

$$\mathbf{S}_K^- = (\mathbf{I}-\mathbf{H}_K^T\mathbf{L}_K^T)\mathbf{S}_K^+(\mathbf{I}-\mathbf{L}_K\mathbf{H}_K), \tag{3.17b}$$

$$\begin{aligned} \mathbf{r}_K^i = &\operatorname{trace}\left([2\mathbf{P}_K^-(\mathbf{H}_K^T\mathbf{L}_K^T-\mathbf{I})\mathbf{S}_K^+\mathbf{L}_K]_i\left(\frac{\mathrm{d}\mathbf{H}_K^i}{\mathrm{d}\mathbf{q}_K^i}\right)\right) \\ &+\operatorname{trace}\left((\mathbf{L}_K^T\mathbf{S}_K^+\mathbf{L}_K)_{ii}\left(\frac{\mathrm{d}\mathbf{R}_K^i}{\mathrm{d}\mathbf{q}_K^i}\right)\right), \end{aligned} \tag{3.17c}$$

and substitute into (3.15b), it is transformed by using the trace identity to:

$$J' = \text{trace}\left((\mathbf{S}_{K-1}^{+})^T(\mathbf{P}_{K-1}^{+})'\right) + \sum_{i=1}^{M}\int_0^{t_K} \mathbf{u}^i(t)^T\mathbf{Q}_\mathbf{u}\mathbf{u}^i(t)'\,\mathrm{d}t + \sum_{i=1}^{M}\Big(\int_{t_{K-1}^{+}}^{t_K} \mathbf{r}^i(t)^T\mathrm{B}(\mathbf{u}')^i\,\mathrm{d}t + (\mathbf{r}_{K-1}^{i+})^T(\mathbf{q}_{K-1}^{i})'\Big). \quad (3.18)$$

Note the trace( ) operator only contract two of the three dimensions in (3.17c), resulting a vector of length l.

Equation (3.18) bears a strong resemblance to (3.13), except for the shifted time index on the first term and the additional third and fourth terms. In general for a given measurement interval $[t_{k-1}^{+}, t_k^{+}]$ where the initial J' equation is

$$J' = \text{trace}\left((\mathbf{S}_{k}^{+})^T(\mathbf{P}_{k}^{+})'\right) + \sum_{i=1}^{M}\int_0^{t_K} \mathbf{u}^i(t)^T\mathbf{Q}_\mathbf{u}\mathbf{u}^i(t)'\,\mathrm{d}t + \sum_{i=1}^{M}\Big(\int_{t_{k}^{+}}^{t_K} \mathbf{r}^i(t)^T\mathrm{B}(\mathbf{u}')^i\,\mathrm{d}t + (\mathbf{r}_{k}^{i+})^T(\mathbf{q}_{k}^{i})'\Big), \quad (3.19)$$

if the following is enforced:

$$\mathbf{S}_k^{-} = (\mathbf{I} - \mathbf{H}_k^T\mathbf{L}_k^T)\mathbf{S}_k^{+}(\mathbf{I} - \mathbf{L}_k\mathbf{H}_k), \quad (3.20a)$$

$$\mathbf{r}_k^{i-} = \text{trace}\left([2\mathbf{P}_k^{-}(\mathbf{H}_k^T\mathbf{L}_k^T - \mathbf{I})\mathbf{S}_k^{+}\mathbf{L}_k]_i\left(\frac{\mathrm{d}\mathbf{H}_k^i}{\mathrm{d}\mathbf{q}_k^i}\right)\right) + \text{trace}\left((\mathbf{L}_k^T\mathbf{S}_k^{+}\mathbf{L}_k)_{ii}\left(\frac{\mathrm{d}\mathbf{R}_k^i}{\mathrm{d}\mathbf{q}_k^i}\right)\right), \quad (3.20b)$$

$$\mathbf{S}_k^{-} \to \mathbf{S}_{k-1}^{+} \qquad \text{via } \frac{\mathrm{d}\mathbf{S}(t)}{\mathrm{d}t} = -\mathbf{A}^T\mathbf{S}(t) - \mathbf{S}(t)\mathbf{A}, \quad (3.20c)$$

$$\mathbf{r}_k^{i-} \to \mathbf{r}_{k-1}^{i+} \qquad \text{via } \frac{\mathrm{d}\mathbf{r}^i(t)}{\mathrm{d}t} = -(\mathbf{F}^i)^T\mathbf{r}^i(t), \quad (3.20d)$$

where $\to$ denotes propagation, then (3.19) is rewritten into

$$J' = \text{trace}\left((\mathbf{S}_{k-1}^{+})^T(\mathbf{P}_{k-1}^{+})'\right) + \sum_{i=1}^{M}\int_0^{t_K} \mathbf{u}^i(t)^T\mathbf{Q}_\mathbf{u}\mathbf{u}^i(t)'\,\mathrm{d}t + \sum_{i=1}^{M}\Big(\int_{t_{k-1}^{+}}^{t_K} \mathbf{r}^i(t)^T\mathrm{B}(\mathbf{u}')^i\,\mathrm{d}t + (\mathbf{r}_{k-1}^{i+})^T(\mathbf{q}_{k-1}^{i})'\Big). \quad (3.21)$$

Through the structure of (3.20a) and (3.20c), it is clear that in fact the entire $\mathbf{S}(t)$ trajectory is symmetric.

By defining the variables according to (3.20), the time index in the J' equation is iteratively shifted toward t_0. Eventually, the J' transformation reaches time t_0, where from earlier result $\mathbf{P}'_0 = \mathbf{0}$ and $(\mathbf{q}^i_0)' = \mathbf{0}$. Hence the final J' equation is

$$\begin{aligned} J' &= \sum_{i=1}^{M} \int_0^{t_K} \mathbf{r}^i(t)^T \underbrace{\mathrm{B}(\mathbf{u}')^i}_{\mathbf{G}^i \mathbf{u}^i(t)'} + \mathbf{u}^i(t)^T \mathbf{Q}_\mathbf{u} \mathbf{u}^i(t)' \, \mathrm{d}t \\ &= \sum_{i=1}^{M} \int_0^{t_K} \underbrace{((\mathbf{G}^i)^T \mathbf{r}^i(t) + \mathbf{Q}_\mathbf{u} \mathbf{u}^i(t))}_{\nabla_{\mathbf{u}^i(t)} J}{}^T \mathbf{u}^i(t)' \, \mathrm{d}t, \end{aligned} \tag{3.22}$$

which is in the necessary form to obtain the local gradient information. This gradient information is ready to be used by an iterative optimization method. After the optimal solution is found, either the optimal control trajectories $\mathbf{u}^i(t)^*$ or the resulting optimal vehicle trajectories from propagating (3.2) with $\mathbf{u}^i(t)^*$ can be sent to the sensor vehicles.

To recap, the first step toward deriving the analytical solution $\nabla_{\mathbf{u}^i(t)} J$ is by perturbing the cost J, which yields (3.8d); this is not in the correct form as in (3.9) to obtain the local gradient information. Through defining the proper adjoint identities (3.11), (3.12), and (3.16), leveraging (3.8c), (3.8e), (3.8f), and the trace identity, and correctly setting $\mathrm{L}^*(\mathbf{S})$, $\mathbf{S}_k^-$, $\mathrm{M}^*(\mathbf{r})^i$, and $\mathbf{r}_k^{i-}$ in (3.20), (3.8d) is transformed iteratively until the final form in (3.22). The local gradient can then be extracted at this point and used in iterative optimization methods (e.g. Steepest Decent, Conjugate Gradient, limited memory BFGS). These steps are dubbed the Dynamic Adaptive Observation (DAO) algorithm.

Note that $\mathbf{P}(t)$ is updated at each measurement time in the forward march; it is thus natural that the adjoint variables $\mathbf{S}(t)$ and $\mathbf{r}^i(t)$ are similarly updated at each measurement time in the backward march. Also note that even though the initial $\mathbf{u}^i(t)$ and $\mathbf{q}^i(t)$ are continuous, because the update to $\mathbf{u}^i(t)$ is derived from $\mathbf{r}^i(t)$, which is now piece-wise continuous like $\mathbf{P}(t)$, the newly updated $\mathbf{u}^i(t)$ in the second iteration of the optimization will also be piece-wise continuous, and

thus so will $\mathrm{d}\mathbf{q}^i(t)/\mathrm{d}t$. Nevertheless, $\mathbf{q}^i(t)$ is still continuous, and the continuity assumption on $\mathbf{q}^i(t)$ in the formulation still holds.

3.4 DAO extensions

3.4.1 Generalization and Clarifications

For clarity sake, the DAO formulation in §3.3 was restricted to a specific cost function and identical dynamics and sensors in all vehicles. Furthermore, when linearizing (3.1) and (3.2) with respect to the future $\hat{\mathbf{x}}(t)$, how the future $\hat{\mathbf{x}}(t)$ is predicted was not discussed. These issues are addressed here.

Generalize Cost Function

The vehicle penalty portion in (3.7) is not restricted to be quadratic and penalizes only $\mathbf{u}^i(t)$, other types of penalties can be incorporated. In general,

$$J = \mathrm{trace}(\mathbf{T}\mathbf{P}_F) + \sum_{i=1}^{M} \Big(\int_0^{t_K} a^i(\mathbf{q}^i(t), \mathbf{u}^i(t))\,\mathrm{d}t + b^i(\mathbf{q}_K^i) \Big). \tag{3.23}$$

Note if $a^i(\cdot,\cdot)$ and $b^i(\cdot)$ are quadratic, one would have the standard Linear Quadratic Regulator (LQR) familiar in the controls community, where $a^i(\cdot,\cdot)$ is the state and control trajectory penalties and $b^i(\cdot)$ the terminal state penalty. Also this formulation allows the possibility to penalize each vehicle differently. Without re-deriving, the modifications to the DAO algoritm are described in the following.

Suppose the perturbation of $a^i(\mathbf{q}^i(t), \mathbf{u}^i(t))$ and $b^i(\mathbf{q}_K^i)$ can be written as

$$a^i(\mathbf{q}^i(t), \mathbf{u}^i(t))' = \left(\frac{\partial a^i(\mathbf{q}^i(t), \mathbf{u}^i(t))}{\partial \mathbf{q}^i(t)} \right)^T \mathbf{q}^i(t)' + \left(\frac{\partial a^i(\mathbf{q}^i(t), \mathbf{u}^i(t))}{\partial \mathbf{u}(t)^i} \right)^T \mathbf{u}^i(t)', \tag{3.24a}$$

$$b^i(\mathbf{q}_K^i)' = \left(\frac{\mathrm{d}b^i(\mathbf{q}_K^i)}{\mathrm{d}\mathbf{q}_K^i} \right)^T (\mathbf{q}_K^i)'. \tag{3.24b}$$

The local cost function gradient is now expressed as

$$\nabla_{\mathbf{u}^i(t)} J = (\mathbf{G}^i)^T \mathbf{r}^i(t) + \frac{\partial a^i(\mathbf{q}^i(t), \mathbf{u}^i(t))}{\partial \mathbf{u}^i(t)}. \tag{3.25}$$

The terminal state penalty simply changes the starting condition of $\mathbf{r}_K^i$, and the state trajectory penalty introduces an additional forcing to the $\mathbf{r}^i$ evolution equation. The evolution equation for $\mathbf{r}^i(t)$ is now

$$\frac{\mathrm{d}\mathbf{r}^i(t)}{\mathrm{d}t} = -(\mathbf{F}^i)^T\mathbf{r}^i(t) - \frac{\partial a^i(\mathbf{q}^i(t),\mathbf{u}^i(t))}{\partial \mathbf{u}^i(t)}, \tag{3.26a}$$

$$\begin{aligned}\mathbf{r}_K^i = \frac{\mathrm{d}b^i(\mathbf{q}_K^i)}{\mathrm{d}\mathbf{q}_K^i} &+ \operatorname{trace}\left((\mathbf{L}_K^T\mathbf{S}_K^+\mathbf{L}_K)_{ii}\left(\frac{\mathrm{d}\mathbf{R}_K^i}{\mathrm{d}\mathbf{q}_K^i}\right)\right)\\ &+ \operatorname{trace}\left([2\mathbf{P}_K^-(\mathbf{H}_K^T\mathbf{L}_K^T - \mathbf{I})\mathbf{S}_K^+\mathbf{L}_K]_i\left(\frac{\mathrm{d}\mathbf{H}_K^i}{\mathrm{d}\mathbf{q}_K^i}\right)\right).\end{aligned} \tag{3.26b}$$

Similarly, the forecast quality metrics considered in §2.1 can be used here as well, where the initialization of $\mathbf{S}_F$ differs depending on the metric used. For the D-optimality criterion, $\mathbf{S}_F = \mathbf{P}_F^{-1}$; and for the E-optimality criterion, $\mathbf{S}_F = \mathbf{r}\mathbf{r}^T$, where $\mathbf{P}\mathbf{r} = \lambda_{\max}\mathbf{r}$.

Routine Measurements

In some situations supplemental routine measurements are also available in the future. These measurements typically come from existing stationary sensor networks that makes routine measurements (e.g. sensor buoys), while some other times these measurements come from non-controllable sources (i.e. wind data from boats). Routine measurements should be incorporated into the AO formulation to avoid redundant measurements. This is done by augmenting H_k and R_k such that

$$\mathbf{H}_k \triangleq \begin{bmatrix}\mathbf{H}_k^r\\ \mathbf{H}_k^{AO}\end{bmatrix}, \qquad \mathbf{R}_k \triangleq \begin{bmatrix}\mathbf{R}_k^r & 0\\ 0 & \mathbf{R}_k^{AO}\end{bmatrix}, \tag{3.27}$$

where the superscript r and AO denotes routine and AO sources. Note when performing perturbation analysis, the perturbation of $\mathbf{H}_k^r$ and $\mathbf{R}_k^r$ to $\mathbf{q}_k^i$ are zero since the routine measurement placements are not controllable.

Future $\hat{\mathbf{x}}(t)$

During the DAO formulation, (3.1) and (3.2) are linearized about the forecast trajectories of $\mathbf{q}^i(t)$, $\mathbf{u}^i(t)$, and $\hat{\mathbf{x}}(t)$. Assuming the initial vehicle states $\mathbf{q}_0^i$ and control trajectories $\mathbf{u}^i(t)$ are know precisely, forecasting $\mathbf{q}^i(t)$ is trivial. However

without knowing the future measurements, it is impossible to forecast $\hat{\mathbf{x}}(t)$; the future measurements must be predicted.

Since the best estimate at current time is $\hat{\mathbf{x}}_0^+$, naturally a forecast made from $\hat{\mathbf{x}}_0^+$ would be used as $\hat{\mathbf{x}}(t)$. Furthermore, the best guess on the i-th vehicle's future measurements $\mathbf{y}_1^i$ is obtained from the open-loop $\hat{\mathbf{x}}(t)$ forecast at t_1, $\hat{\mathbf{x}}_1^-$, and apply the output operator $h_1^i(\hat{\mathbf{x}}_1^-)$ to predict $\hat{\mathbf{y}}_1^i$. Using this predicted measurement in the state update equation would yield $\hat{\mathbf{x}}_1^+ = \hat{\mathbf{x}}_1^-$, since the innovation would be zero. In general if $\hat{\mathbf{x}}_k^+$ is given, one would follow similar logics to forecast $\hat{\mathbf{x}}_k^+$ and update with $\hat{\mathbf{y}}_{k+1}^i = h_{k+1}^i(\hat{\mathbf{x}}_{k+1}^-)$, and show that $\hat{\mathbf{x}}_{k+1}^+ = \hat{\mathbf{x}}_{k+1}^-$. Thus by induction it is clear to see a forecast from $\hat{\mathbf{x}}_0^+$ is the best prediction of the future $\hat{\mathbf{x}}(t)$.

3.4.2 Reducing to Pure Discrete- and Continuous-time

The mixed continuous-/discrete-time DAO formulation developed in §3.3 can be reduced to a pure discrete- or continuous-time setting. The standard cost in (3.7) is used for the derivation; however note that in both cases, the generalizations in §3.4.1 equally applies.

Discrete-time DAO

If the continuous propagation of $\mathbf{P}(t)$ in (3.5) and $\mathbf{q}^i(t)$ in (3.2) from time t_k to t_{k+1} are allowed to be propagated discretely in one time-step, then the propagation and update can be combined into a discrete evolution equation for the posterior estimation covariance and vehicle states. The conversion to discrete time propagation is done through an Explicit Euler approximation of the continuous time propagation. Similarly, this could be done for the adjoint propagations. Using the same Explicit Euler approximation and bearing in mind the propagations are *backward* in time, the combined adjoint propagation and update equations are

$$\mathbf{S}_{k-1} = \mathbf{A}_D^T(\mathbf{I} - \mathbf{H}_k^T\mathbf{L}_k^T)\mathbf{S}_k(\mathbf{I} - \mathbf{L}_k\mathbf{H}_k)\mathbf{A}_D, \tag{3.28a}$$

$$\begin{aligned}\mathbf{r}_{k-1}^i = (\mathbf{F}_D^i)^T\mathbf{r}_k^i &+ \operatorname{trace}\left((\mathbf{L}_k^T\mathbf{S}_k\mathbf{L}_k)_{ii}\left(\frac{\mathrm{d}\mathbf{R}_k^i}{\mathrm{d}\mathbf{q}_k^i}\right)\right) \\ &+ \operatorname{trace}\left([2\mathbf{A}_D\mathbf{P}_k\mathbf{A}_D^T(\mathbf{H}_k^T\mathbf{L}_k^T - \mathbf{I})\mathbf{S}_k\mathbf{L}_k]_i\left(\frac{\mathrm{d}\mathbf{H}_k^i}{\mathrm{d}\mathbf{q}_k^i}\right)\right),\end{aligned} \tag{3.28b}$$

where $(\,)_D$ denotes the discretized version of the continuous-time counterpart. Note the $(\,)^+$ and $(\,)^-$ designations are dropped between they have combined into a single equation. The cost function gradient is also appropriately redefined as

$$J' = \sum_{i=1}^{M} \sum_{k=0}^{K-1} (\nabla_{\mathbf{u}_k^i} J)^T \mathbf{u}_k^{i\,\prime}, \tag{3.29a}$$

$$\nabla_{\mathbf{u}_k^i} J = (\mathbf{G}_D^i)^T \mathbf{r}_k^i + \mathbf{Q}_{\mathbf{uD}} \mathbf{u}_k^i, \tag{3.29b}$$

where final time index is $K-1$ because the zero-order-hold assumption for the control $\mathbf{u}_k^i$.

Equation (3.28) only holds within the time interval $[t_0, t_K]$, where measurement updates are performed. It is necessary to propagate $\mathbf{S}_F$ to $\mathbf{S}_K$ using the discretized version of (3.20c)

$$\mathbf{S}_{k-1} = \mathbf{A}_D^T \mathbf{S}_k \mathbf{A}_D \tag{3.30}$$

and set $\mathbf{S}_{K-1} = \mathbf{S}_K$ before using (3.28). The starting conditions for $\mathbf{S}_F$ and $\mathbf{r}_{K-1}^i$ remain the same. Note the starting conditions for $\mathbf{S}_{K-1}$ and $\mathbf{r}_{K-1}^i$ are one time-step off from their respective mix-time counterpart in §3.3. This is due to a theoretical gap that exists when converting a continuous-time adjoint equation to discrete-time. This inconsistency vanishes as the time-step becomes small, as $\mathbf{S}_{K-1}$ and $\mathbf{r}_{K-1}^i$ approach $\mathbf{S}_K$ and $\mathbf{r}_K^i$.

Continuous-time DAO

The continuous-time Kalman Filter is rarely used in practice for practical reasons. Nevertheless, for theoretical completeness a DAO formulation also exists for a continuous-time KF. There are two formulation approaches to this problem. The first approach is to perform the entire DAO formulation presented before with the continuous-time KF propagation of $\mathbf{P}(t)$. A perhaps simpler approach, inspired by the work in Smith & Robers (1978) which reconciles the discrete- and continuous-time KF, is by substituting the discrete-time components with

$$\mathbf{A}_D \to \mathbf{I} + \Delta t \mathbf{A}, \qquad \mathbf{F}_D^i \to \mathbf{I} + \Delta t \mathbf{F}^i,$$

$$\mathbf{H}_k \to \mathbf{H}(t_k), \qquad \mathbf{R}_k \to \mathbf{R}(t_k)/\Delta t,$$

into (3.28). After pulling the $\mathbf{S}_k - \mathbf{S}_{k-1}$ and $\mathbf{r}_k^i - \mathbf{r}_{k-1}^i$ terms to the LHS and divide by Δt, the limit as $\Delta t \to 0$ is taken to yield the continuous-time equivalent of the DAO algorithm:

$$\frac{\mathrm{d}\mathbf{S}}{\mathrm{d}t} = -\mathbf{A}^T\mathbf{S} - \mathbf{S}\mathbf{A} + \mathbf{S}\mathbf{P}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{P}\mathbf{S}, \tag{3.31a}$$

$$\begin{aligned}\frac{\mathrm{d}\mathbf{r}^i}{\mathrm{d}t} = &-(\mathbf{F}^i)^T\mathbf{r}^i + \operatorname{trace}\left([2\mathbf{P}\mathbf{S}\mathbf{P}\mathbf{H}^T\mathbf{R}^{-1}]_i\left(\frac{\mathrm{d}\mathbf{H}^i}{\mathrm{d}\mathbf{q}^i}\right)\right) \\ &- \operatorname{trace}\left((\mathbf{R}^{-1}\mathbf{H}\mathbf{P}\mathbf{S}\mathbf{P}\mathbf{H}^T\mathbf{R}^{-1})_{ii}\left(\frac{\mathrm{d}\mathbf{R}^i}{\mathrm{d}\mathbf{q}^i}\right)\right),\end{aligned} \tag{3.31b}$$

and the local gradient is the same as (3.22). Again (3.31) only holds within the time interval $[t_0, t_K]$. It is necessary to propagate $\mathbf{S}_F$ to $\mathbf{S}_K$ using (3.20c). Like the discrete-time case, the starting conditions for $\mathbf{S}_F$ and $\mathbf{r}_F^i$ remain the same.

3.5 Experiments and Results

The capabilities of the DAO algorithm shall be demonstrated through two numerical experiments. The first experiment—a surveillance problem—uses the DAO algorithm to find two vehicle trajectories for camera-equipped sensor vehicles in order to minimize a growing stationary uncertainty field. The second experiment—an environmental plume problem—uses DAO to plan sensor vehicle waypoints for improving estimation of a convection driven contaminant plume (moving uncertainty field).

3.5.1 Surveillance

Setup

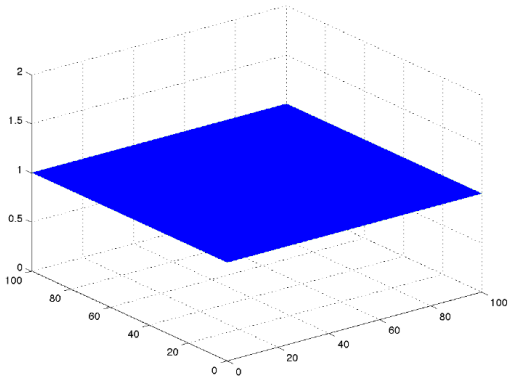
Suppose two ($M = 2$) sensor vehicles start at the bottom-left of a square domain of size 100×100. Each vehicle is equipped with a specialized imaging system which takes a 360-degree-view image of the domain. Like real-world imaging systems, the image quality gets progressively worse as the distance between the objects and the camera increases (poor pixel resolution). Furthermore, there are Regions Of Interests (ROIs) within the domain which additional visual information

is preferred. This is modeled as an uncertainty field growing with a constant rate, where the ROIs corresponds to regions with high uncertainty and thus grow the quickest if left unexplored. In particular, three types of uncertainty fields (defined as the variance of the domain estimate) are considered, as illustrated in Figure 3.6. Three types of surveillance situations common in practice are modeled in Figure 3.6. Figure 3.6(a) represents the first common surveillance situation, where a new domain is surveyed for the first time; therefore the uncertainty of the entire domain is uniformed. The second common surveillance situation is when a domain is recently surveyed, and several ROIs with different priorities are identified; therefore the vehicle objective should focus only on those ROIs, which are modeled in Figure 3.6(b) as Guassian bumps with various amplitudes and attenuations. The uncertainty field in Figure 3.6(c) is the combination of Figure 3.6(a) and 3.6(b), which models a previously, but not necessarily recent, surveyed domain; therefore in addition to those ROI, the entire domain is preferably resurveyed. The objective of this problem is to find the optimal vehicle trajectories over a time window, subjected to vehicle dynamic, that minimizes the uncertainty field at the end of the time window.

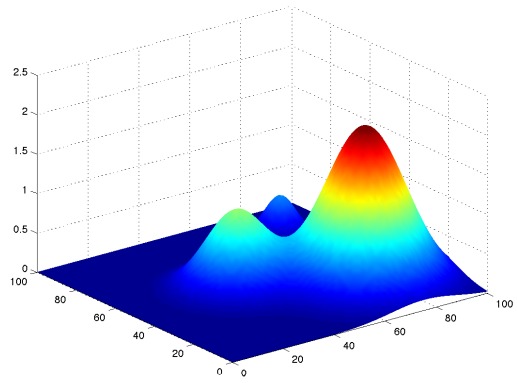
Given the nature of the problem, the measurements are taken frequently when compared with the dynamics of the vehicle; therefore the discrete-time version of DAO discussed in §3.4.2 is used (the continuous-time version in §3.4.2 could in principle be used as well, but because discrete system formulations are more prevalent in practice, the discrete-time version is more favored), where a measurement is taken at each time-step. Furthermore, the vehicle dynamic description is simplified by assuming the vehicles are point-masses with damping

$$\begin{bmatrix} \dot{x}^i \\ \ddot{x}^i \\ \dot{y}^i \\ \ddot{y}^i \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \end{bmatrix} \underbrace{\begin{bmatrix} x^i \\ \dot{x}^i \\ y^i \\ \dot{y}^i \end{bmatrix}}_{\mathbf{q}^i} + \underbrace{\begin{bmatrix} 0 \\ u_x^i \\ 0 \\ u_y^i \end{bmatrix}}_{\mathbf{u}^i}, \tag{3.32}$$

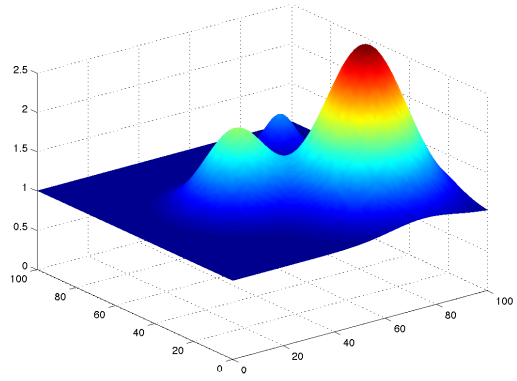
where (x^i, y^i) is the i-th vehicle's position inside the domain. The domain and the uncertainty fields are discretized into $101 \times 101 = 10201$ grid points. For simplicity,



(a) Unit uncertainty.



(b) Gaussian bumps.



(c) Sum of first two.

Figure 3.6: The three different uncertainty fields considered in this experiment. (a) is an uniform uncertainty, which represent a region completely unexplored. (b) is composed of 3 Gaussian bumps of different amplitude and attenuations, which represents a previously-explored region with three different high-value targets. (c) is the sum of (a) and (b). These fields are used to initialize the diagonals of $\mathbf{P}_0$.

the optimization time window is chosen to be $t_K = t_F = 300$ (i.e. no additional forecast from $\mathbf{P}_K^+$ to $\mathbf{P}_F$) and the uncertainty fields are assumed to be the diagonals for a diagonal $\mathbf{P}_0$. For the cost function, the entire domain is targeted ($\mathbf{T} = \mathbf{I}$), $\mathbf{Q_u} = \mathbf{I}$ for the control trajectories penalty, and both vehicle state trajectories and terminal state penalties are not considered.

Since the camera can "see" the entire domain, $h_k^i(\mathbf{x}_k) = \mathbf{H}_k^i = \mathbf{I}$. The distance-dependent image resolution is modeled with $\mathbf{R}_k^i$, where the measurement noise variance at each grid point is proportional to the distance to the vehicle's location squared:

$$\mathbf{R}_k^i = \frac{\beta}{2}\text{diag}\big((x_k^i\mathbf{1} - \mathbf{z}_x) \bullet (x_k^i\mathbf{1} - \mathbf{z}_x) + (y_k^i\mathbf{1} - \mathbf{z}_y) \bullet (y_k^i\mathbf{1} - \mathbf{z}_y) + \epsilon\mathbf{1}\big), \qquad (3.33)$$

where β is a proportional constant, $\mathbf{z}_x$ and $\mathbf{z}_y$ are the discretized grid point locations, $\mathbf{1}$ is a vector of 1s of the appropriate size, and $\bullet$ denotes element-wise multiplication (Schur product). The ϵ term serves two purposes: first it models the camera intrinsic digital background noise such as quantization error, and second it ensures the positive-definiteness of $\mathbf{R}_k^i$ required by the Kalman Filter. In this experiment, $\beta = 0.1$ and $\epsilon = 0.001$. $\mathbf{R}_k^i$ can be visualized as a quadratic bowl centered at vehicle i's position with β affecting the curvature. Note when the vehicle position coincides with a grid point, then for a small ϵ (typically this is true since quantization error is much smaller than the signal), the state estimate variance at that grid point is driven to nearly zero *regardless of the initial variance*.

To model the growing uncertainty field, a simple model with a diagonal A is chosen:

$$\mathbf{P}_{k+1} = \mathbf{A}\mathbf{P}_k\mathbf{A}^T, \qquad \mathbf{A} = \alpha\mathbf{I}. \qquad (3.34)$$

This model also significantly simplifies the computations performed. It can be shown that together with the initially diagonal $\mathbf{P}_0$, the structure of $\mathbf{H}_k$ in this problem, diagonal $\mathbf{R}_k^i$, and a diagonal $\mathbf{A}$, the DAO algorithm can be carried out by tracking only the diagonals of $\mathbf{P}_k$ and $\mathbf{S}_k$ (see Appendix B). $\alpha = \sqrt{1.01}$ is chosen so that if no measurement updates are performed, at the end of the time window the uncertainty field would have grown by $1.01^{300} \approx 20$ times.

The one last piece of information needed to start the DAO algorithm is a set of initial nominal optimal control sequences. The initial control sequences $\mathbf{u}_k^1$

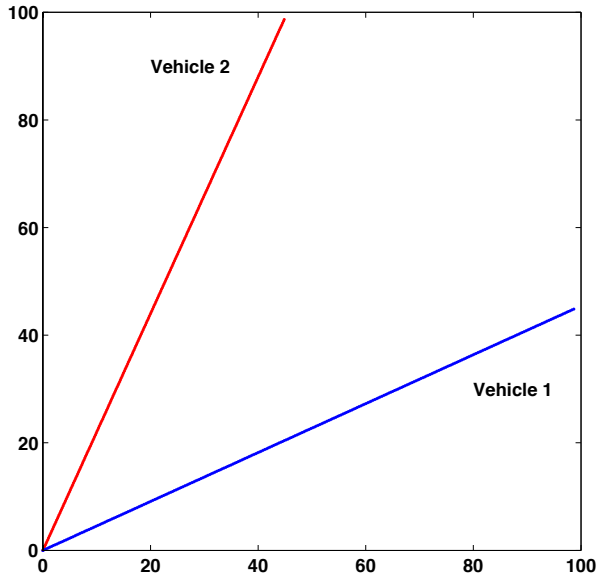
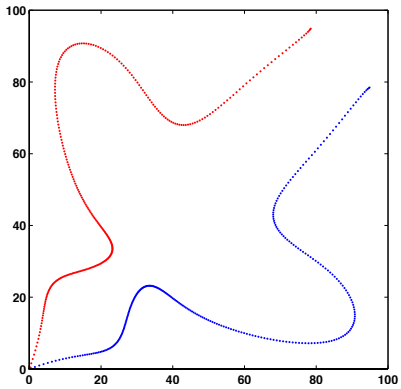


Figure 3.7: Initial vehicle trajectories created by initial control sequences $\mathbf{u}_k^1$ and $\mathbf{u}_k^2$. The trajectories are symmetric about the domain diagonal.

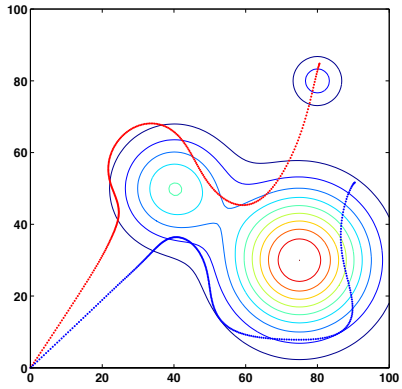
and $\mathbf{u}_k^2$ that produce initial the vehicle trajectories shown in Figure 3.7 are used. Note the vehicle trajectories are symmetric about the diagonal.

Results

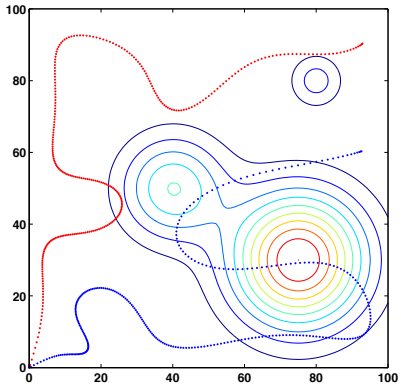
The converged optimal vehicle trajectories for all three initial uncertainties in Figure 3.6 with $\beta = 0.1$ are shown in Figure 3.8. As seen in Figure 3.8(a), when the uncertainty is uniform the resulting optimal solution take on an uniform-coverage approach. The symmetry in the converged trajectories is due to the symmetric initial vehicle trajectories. Running the same optimization with non-symmetric vehicle initial conditions, the solution is shown in Figure 3.9. While the symmetry is destroyed, the uniform-coverage attribute remains. When the initial uncertainty consist just the three Gaussian bumps, the solution is completely different. As seen in Figure 3.8(b), the vehicles head right into the bumps without deviation; the vehicles are also moving at a higher velocity toward the bumps than when they are when inside, since there are incentives to be near the bumps. Since the uncertainty field in Figure 3.6(c) is the sum of Figure 3.6(a) and 3.6(b), the converged vehicle trajectories embody characteristics from both.



(a) DAO solution for the uniform uncertainty.



(b) DAO solution for the Gaussian bumps uncertainty.



(c) DAO solution for the uniform background with Gaussian bumps uncertainty.

Figure 3.8: Converged vehicle trajectories for all three uncertainty fields, $\beta = 0.1$.

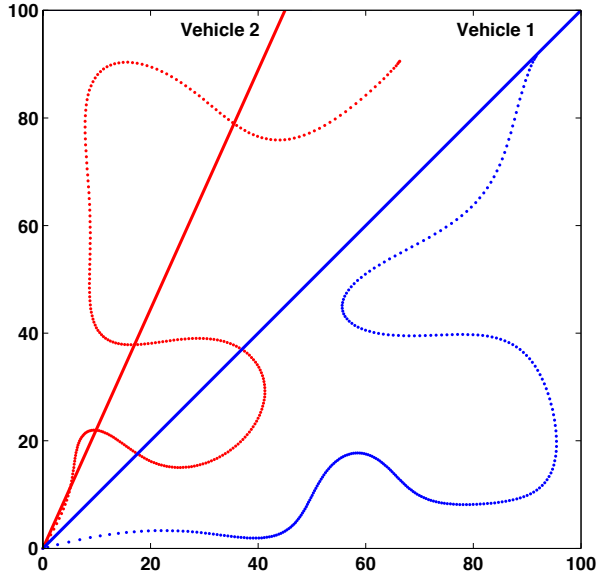


Figure 3.9: Optimal vehicle trajectories for the uniform uncertainty with non-symmetric vehicle initial conditions. Solid lines indicate the initial vehicle trajectory for both vehicles.

3.5.2 Environmental Plume

Setup

In the second experiment, a representative problem modeling the problem-of-interest of this thesis is used. The system considered is the 2D Navier-Stokes Equation (NSE), with additive low-frequency forcing, coupled with a passive (does not affect the flow field) scalar with source near the center of the physical domain. The governing equations are

$$\frac{\partial \mathbf{x}}{\partial t} = -\mathbf{x} \cdot \nabla \mathbf{x} + \nu \nabla^2 \mathbf{x} + \frac{1}{\rho} \nabla p + \mathbf{f}_{\mathbf{u}}, \tag{3.35a}$$

$$\frac{\partial \phi}{\partial t} = -\mathbf{x} \cdot \nabla \phi + \kappa \nabla^2 \phi + \mathbf{f}_{\phi}, \tag{3.35b}$$

$$h_k^i(\mathbf{x}_k) = \mathbf{H}_k^i \mathbf{x}_k, \tag{3.35c}$$

with density ρ, kinematic viscosity ν, pressure p, and diffusion constant κ. $\mathbf{x}$ is a velocity vector field containing the horizontal and vertical velocities, ϕ is the passive scalar field simulating an environmental plume. The i-th measurement operator $\mathbf{H}_k^i$ measures the local quantity of $\mathbf{x}$ and ϕ at the i-th sensor position contained in

the vehicle state $\mathbf{q}_k^i$. Again, each sensor vehicle has the same dynamics as (3.32).

Numerical simulation of (3.35) uses the pseudo-spectral code developed in Bewley *et al.* (2001) on a 64×64 uniform square grid. In all numerical experiments the "truth" simulation uses an identical model (with different initial conditions and random forcing) running in parallel with an Ensemble Kalman Filter (EnKF, to be described in Chapter 4). The simulations are done in a periodic domain non-dimensionalized with width $L = 2$. To simulate estimating a non-periodic domain, the estimation is isolated to a subregion of width $R = 1.35$ centered in the domain. The targeting matrix $\mathbf{T}$ in (3.7) is chosen to focus on the entire estimation subregion. For flow computation stability, a relatively small marching time-step ($\Delta t = 0.005$ time unit) is used, and measurements are taken every $\tau_{meas} = 0.15$ time units. Because of the difference between the marching step size and measurement interval, the mixed continuous/discrete DAO formulation is used. Again for simplicity $t_F = t_K$.

In this experiment the optimization objective is to find the vehicle waypoints subjected to vehicle dynamics over six measurement times, thus the event horizon is $t_F = \tau_{meas} \times 6 = 0.9$. During each measurement the DAO algorithm receives the current covariance estimate from the EnKF and updates the optimization problem. The optimization lasts 3 measurements, and when the time is up the optimal control sequence iterate (whether the optimization has converged or not) is used to forecast future optimal vehicle waypoints. The future six-vehicle-waypoints sequences are sent to the vehicles, but only the first three are used; the last three waypoints serve as backup should the vehicles fail to receive the next set of waypoint sequence from DAO for any reason (although this is not necessary for the numerical simulation, it is useful for any future physical implementations). The control sequence for the last three waypoints are used as part of the initial control sequence for the next optimization. Figure 3.10 illustrates how the DAO algorithm and the EnKF interacts.

Rather than propagating $\mathbf{P}(t)$ using the fully model dynamics, the same "growing" model and diagonal $\mathbf{P}(t)$ as in the surveillance example is adopted

$$\frac{\mathrm{d}\mathbf{P}(t)}{\mathrm{d}t} = 0.1\mathbf{P}(t) + 0.1\mathbf{P}(t)^T = 0.2\mathbf{P}(t), \tag{3.36}$$

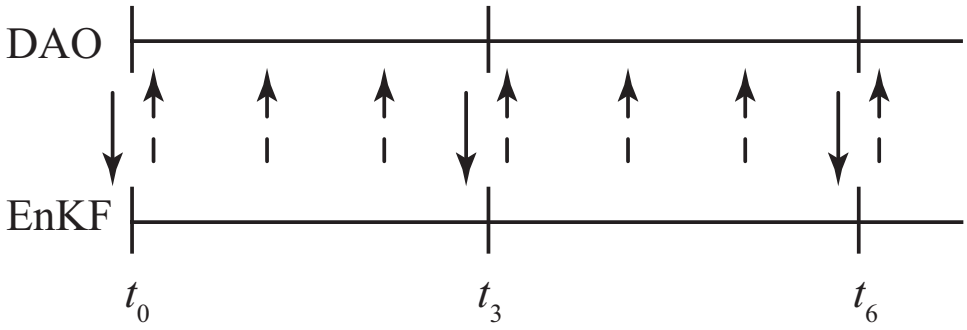


Figure 3.10: Cartoon illustrating how DAO and EnKF interacts in time. During each measurement the EnKF sends the new estimation covariance to DAO (dashed up arrow) to update the optimization problem. The optimization lasts 3 measurements and the future vehicle waypoint sequences are sent to the vehicles (solid down arrow).

This approximation is justified by the small event horizon t_F considered in the present simulation, where numerical experiment data collected on (3.35) (not shown) suggests the equations are essentially linear with little cross-correlations between states within this event horizon. The second reason for this approximation is to achieve computational feasibility, since as mentioned in §3.5.1 only the diagonals of $\mathbf{P}(t)$ and $\mathbf{S}(t)$ are needed to propagate and stored. Chapter 4 will attempt to address this problem.

To quantify the estimate quality, the steady-state, infinite-time averaged absolute error, defined by Bewley & Protas (2004) as the difference between the estimate and the truth squared, integrated over the estimation subregion is considered:

$$\mathrm{Errn}(\hat{\mathbf{x}}, \mathbf{x}_{tru}) = \int_R (\hat{\mathbf{x}} - \mathbf{x}_{tru})^2 \, dR, \tag{3.37}$$

where $(\,)_{tru}$ corresponding the "truth" values. Long-time averages of this measure applied to both the velocity field and the scalar are used to approximate the expected value.

Results

Figure 3.11 and Figure 3.12 compare the time-averaged error within a time interval using three different adaptive observation strategies: sensors following a

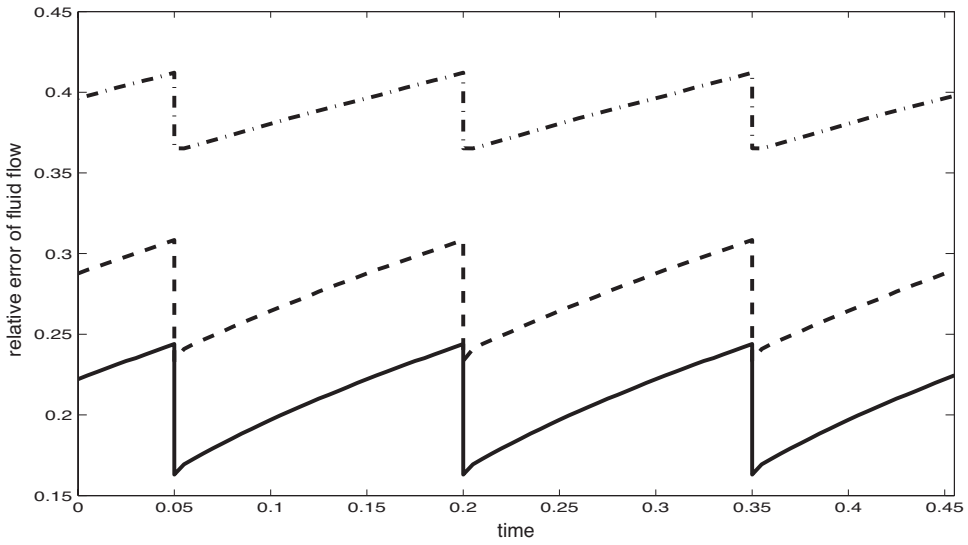


Figure 3.11: Time-averaged absolute error of the flow velocities within a time interval, following three AO strategies: (1) sensors following a random walk, average 13.4 (dot-dashed) (2) stationary sensors uniformly distributed over the estimation subregion, average 9.5 (dashed), and (3) sensor trajectories provided by DAO, average 7.1 (solid). The error increases between measurements, and decreases at the EnKF measurement updates, thus creating the "saw-tooth" shape in the error plot.

random walk, sensors distributed uniformly in the estimation subregion, and the present DAO algorithm. As a baseline to compare against, when no measurements are taken, the average estimation error as defined in (3.37) for the flow velocity is 34.5, and 4.06 for the scalar. Figure 3.13 provides a typical example of the truth simulation compared against the EnKF estimate, and the waypoints optimized by the DAO algorithm.

These results in Fig. 3.11 and Fig. 3.12 demonstrate that significant improvements in the estimate can be accomplished via path planning. The DAO algorithm is able to achieve a 47% reduction for the flow and 42% reduction for the scalar estimation error compare to the random walk scheme, and 25% and 17% respectively compared against uniformly distributed sensors. Furthermore, these results also suggests deliberate sensor placements is more important than unorganized movement, as demonstrated by the performance of uniformly distributed sensors against random walk.

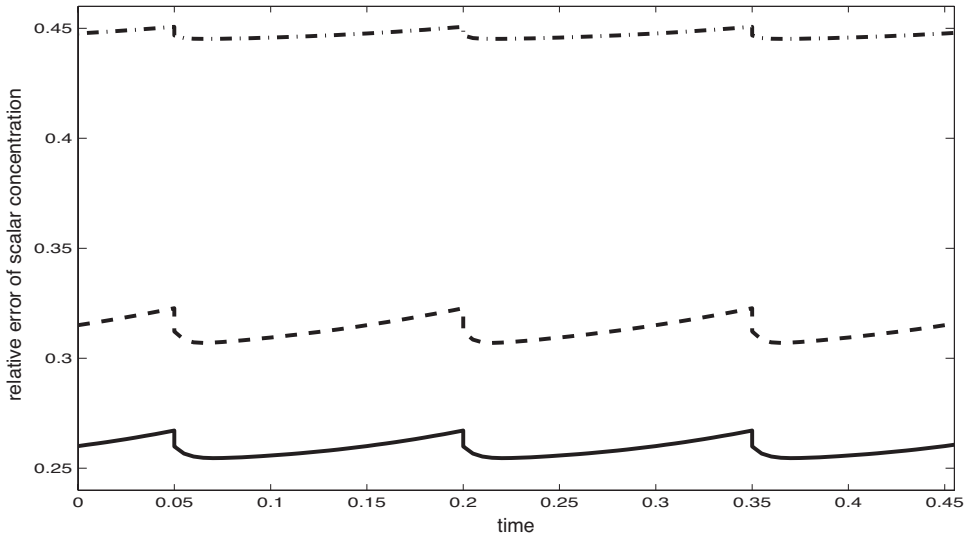


Figure 3.12: Time-averaged absolute error of the scalar within a time interval, with the sensor motion as described in Figure 3.11. Because the scalar evolution is primarily driven by the flow velocities, the scalar estimate absolute error dips slightly after each measurement update, due to the improved velocity estimate. The average errors are 1.82 for random walk, 1.28 for uniformly distributed stationary sensors, and 1.06 for DAO.

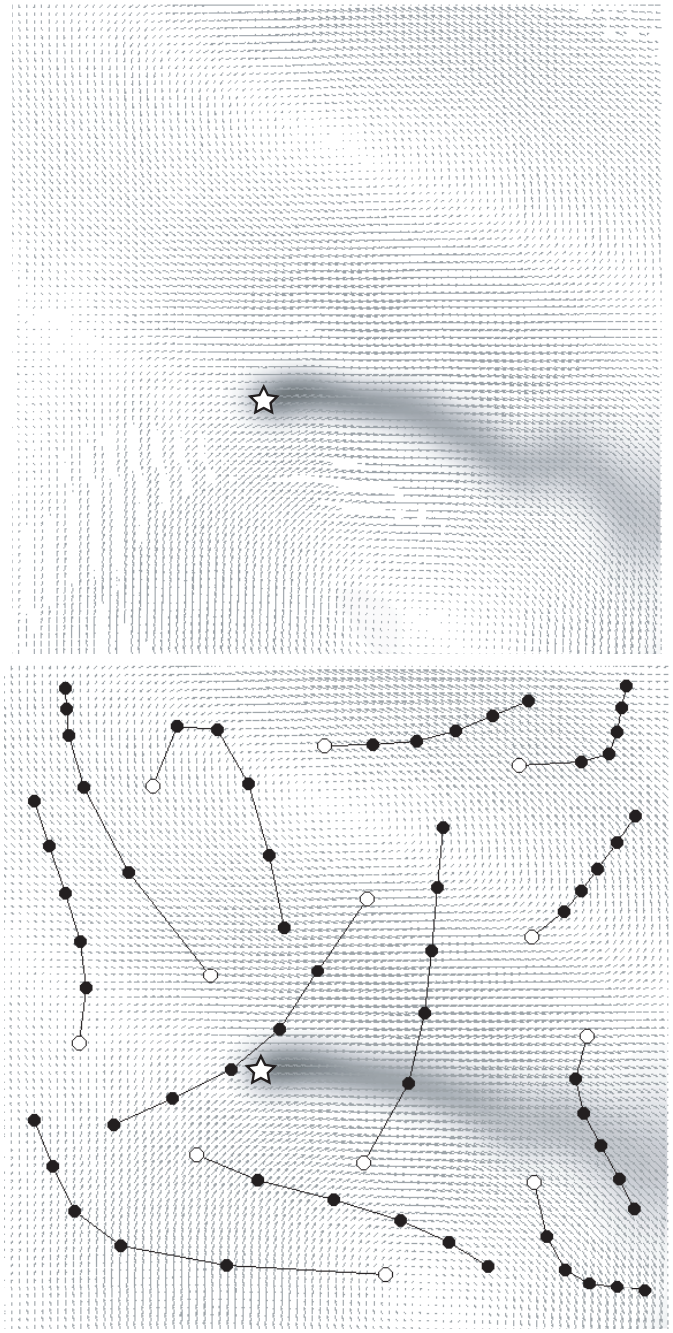


Figure 3.13: A typical example of the truth simulation (left) and sensor waypoints optimized by DAO, overlaying the velocity field and passive scalar estimated by the EnKF (right). The passive scalar source (star) is located near the center of the domain. Empty circles denote the initial vehicle positions with five subsequent optimized vehicle waypoints denote as black circles.

3.6 Summary/Discussion

A new Adaptive Observation (AO) algorithm, dubbed the Dynamic Adaptive Observation (DAO) algorithm, is presented in this chapter. Unlike existing AO algorithms, the DAO algorithm incorporates vehicle dynamics and compute the optimal vehicle waypoints that minimizes estimation/forecast uncertainty. By minimizing a cost containing both a measure of the forecast quality and vehicle control effort penalties *iteratively* using adjoint analysis, the DAO algorithm is able to balance control effort with uncertainty reduction. Generalizations such as vehicle penalties, forecast quality metrics, and incorporating routine measurements are discussed; also for theoretical completeness the fully continuous- and discrete-time DAO formulation is derived from the mixed-time analysis.

Two numerical experiments are performed. The first experiment applies the DAO algorithm to a surveillance problem, and the resulting optimized vehicle trajectories are sensible, where an uniform coverage solution is produced for uniform uncertainty, localized coverage solution is produced for localized uncertainty, and a combined coverage solution when the two uncertainties are combined. The second experiment combines the Ensemble Kalman Filter (EnKF) and the DAO algorithm, and applies them to estimate an environmental flow represented with a passive scalar emanating from a source and driven convectively in a 2D randomly-forced flow. The results demonstrates a significant reduction in the estimation error over less deliberate sensor routing strategies.

Because the DAO algorithm is formulated with the Kalman Filter, one disadvantage is that propagation of the full covariance matrix is required, which is not practical due to the problem size. For example, in typical environmental flow system, to achieve sufficient model fidelity the flow system is routinely discretized with state size ($n \geq 100,000$); in the DAO algorithm, this means the covariance and adjoint matrix $\mathbf{P}$ and $\mathbf{S}$ both contains more than 10 billion entries, and the propagation and storage of these matrices are currently computationally intractable. This issue is side-stepped in the numerical experiments by assuming a simplified model of the covariance evolution that reduces the necessary computations significantly. For better performance, the underlying model should be fully leveraged;

Figure 3.14: One of the sensor vehicles used in a parking-lot plume forecasting experiments performed at UCSD during 2010.

hence a computationally feasible version of the DAO algorithm, developed based on the EnKF, will be presented in Chapter 4.

A new Hybrid variational / Ensemble Kalman Smoother (HEnS) algorithm for state estimation has recently been proposed by the UCSD Flow Control and Coordinated Robotics group. Preliminary tests have show that this new algorithm outperforms the EnKF in the presence of substantial non-Gaussian uncertainties. Steps are taken to combine HEns with DAO in the near future, with an ultimate goal that one day, sensor vehicles such as the ones depicted in Figure 3.14 would be deployed in an environmental contaminant disaster (represented as colored smoke in the photo) as first-responders to adaptively take environmental readings and facilitate contaminant movement forecast in real-time.

3.7 Acknowledgments

I would like to thank Chris Colburn for implementing the stochastic EnKF on the Navier-Stoke's equation and gathering time-averaged performance statistics on the performances of different sampling methods in the environmental plume numerical experiment. The materials presented in this chapter are taken from

- Zhang, D., Colburn, C.H. Bewley, T.R. "*Estimation and Adaptive Observation of Environmental Plumes*", Proceedings of American Control Conference 2011, San Francisco, USA.

- Zhang, D., Colburn, C.H., Bewley, T.R. "*Estimation and Adaptive Observation of Environmental Plumes*", under preparation.

Chapter 4

Ensemble Variational Adaptive Observation (EnVO)

In Chapter 3 the Dynamic Adaptive Observation (DAO) algorithm is introduced. The DAO algorithm is a centralized Adaptive Observation (AO) algorithm that incorporates the sensor vehicle dynamics in the formulation, and compute the optimal vehicle control to move the vehicles in such a way that the estimation/forecast quality of the system is improved.

The DAO algorithm uses the Kalman Filter (KF) to predicts the future estimation/forecast error covariance and computes the best control, subjected to the vehicle dynamic constraints, to minimize this covariance. This is achieved by minimizing a cost function containing a estimation/forecast quality metric with a vehicle control penalty metric. Because explicit optimal control formulation with respect to the cost function is difficult to derive analytically, adjoint analysis is used to calculate the local gradient, and iterate on the optimal control. While initial numerical experimental results seem promising, the DAO algorithm cannot be reasonably applied to real-world problems because it requires full covariance matrix propagation. This issue is side-stepped in Chapter 3 by assuming special structures in the underlying system evolution and measurement operator. Toward applying DAO to real-world AO problems, in this chapter the DAO algorithm is reformulated to use the Ensemble Kalman Filter (EnKF) for computational efficiency. The resulting algorithm is the Ensemble Variational Observation (EnVO)

algorithm, and there are two flavors to this algorithm: stochastic and deterministic.

In order to be self-contained, the entire AO problem is restated in §3.2. Although it is recommended that the readers should familiarize themselves with Chapter 3 before proceeding to this chapter, it is not necessary because many concepts introduced in Chapter 3 will be reintroduced in this chapter. The stochastic EnVO algorithm is derived by first introducing the stochastic EnKF in §4.2.1, the corresponding EnVO algorithm is formulated in §4.2.2. Similarly, the deterministic EnVO is formulated by first introducing the deterministic Ensemble Square-Root Filter (ESRF) in §4.3.1, followed by analysis in §4.3.2. Typically when the EnKF is used in practice, it is implemented in conjunction with ad-hoc covariance conditioning techniques such as covariance inflation and covariance localization. These two conditioning techniques are reviewed in §4.5, and how the EnVO algorithms could also incorporate these techniques in its formulation is discussed. Finally, a numerical experiment is performed with the 1D Kuramoto-Sivashinksy (KS) equation as the underlying system to test the stochastic EnVO performance against two other ad-hoc sampling methods in §4.6.

4.1 The AO problem

The adaptive observation problem considered in this chapter is restated from Chapter 3, with some minor differences. Suppose at initial time t_0 one wish to make a forecast of $\mathbf{x}(t)$ at t_F, where $\mathbf{x}(t) \in \mathbb{R}^n$ is the discretization of a PDE system over n grid points. $\mathbf{x}(t)$ evolves with the underlying dynamics

$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = f(\mathbf{x}(t)). \tag{4.1}$$

External disturbances and model uncertainties can also be modeled in (4.1), however for simplicity this is neglected in the present discussion. There is a series of K discrete future measurement times $\{t_0, t_1 \cdots t_K\}$, $t_K \leq t_F$, where M sensor vehicles can be deployed to gather additional measurements about the system. The dynamical equation of the i-th vehicle state and control, denoted as $\mathbf{q}^i(t)$ and $\mathbf{u}^i(t)$ respectively, is:

$$\frac{\mathrm{d}\mathbf{q}^i(t)}{\mathrm{d}t} = g(\mathbf{q}^i(t), \mathbf{u}^i(t)). \tag{4.2}$$

As the vehicles move, vehicle states such as position, heading, and velocity may affect the measurement results; also depending on the sensor types, the vehicle state may influence the measurement noise statistics. Assuming the influence is local, the measurement operator $h_k^i(\mathbf{x}_k)$ and measurement noise covariance $\mathbf{R}^i$ are dependent (nonlinearly) to their respective vehicle state $\mathbf{q}^i(t)$. Therefore the measurement vector $\mathbf{y}_k^i$ taken from the i-th vehicle at time t_k is

$$\mathbf{y}_k^i = h_k^i(\mathbf{x}_k, \mathbf{q}_k^i) + \mathbf{v}_k^i, \qquad \mathbf{v}_k^i \sim N(0, \mathbf{R}^i(\mathbf{q}_k^i)). \tag{4.3}$$

For convenience, from hereon the explicit dependencies of $h_k^i(\mathbf{x}_k, \mathbf{q}_k^i)$ and $\mathbf{R}^i(\mathbf{q}_k^i)$ are dropped with the understanding that it is implied when $h_k^i(\mathbf{x}_k)$ and $\mathbf{R}_k^i$ are used instead. The collective $\mathbf{R}_k^{AO}$, $h_k^{AO}(\mathbf{x}_k)$, and $\mathbf{y}_k^{AO}$ from all AO vehicles are

$$\mathbf{R}_k^{AO} = \begin{bmatrix} \mathbf{R}_k^1 & & 0 \\ & \ddots & \\ 0 & & \mathbf{R}_k^M \end{bmatrix}, \quad h_k^{AO}(\mathbf{x}_k) = \begin{bmatrix} h_k^1(\mathbf{x}_k) \\ \vdots \\ h_k^M(\mathbf{x}_k) \end{bmatrix}, \quad \mathbf{y}_k^{AO} = \begin{bmatrix} \mathbf{y}_k^1 \\ \vdots \\ \mathbf{y}_k^M \end{bmatrix}. \tag{4.4}$$

Typically in addition to AO measurements, there are also routine measurements from external sources (denote with superscript r); thus the overall measurement operator, noise covariance, and measurement vector are defined as

$$\mathbf{R}_k \triangleq \begin{bmatrix} \mathbf{R}_k^r & 0 \\ 0 & \mathbf{R}_k^{AO} \end{bmatrix}, \quad h_k(\mathbf{x}_k) \triangleq \begin{bmatrix} h_k^r(\mathbf{x}_k) \\ h_k^{AO}(\mathbf{x}_k) \end{bmatrix}, \quad \mathbf{y}_k = \begin{bmatrix} \mathbf{y}_k^r \\ \mathbf{y}_k^{AO} \end{bmatrix}. \tag{4.5}$$

Assuming an EnKF (reviewed in §4.2.1 and 4.3.1) is used to assimilate future measurements, the forecast covariance $\mathbf{P}_F$ can be predicted. The AO problem is stated as followed:

At time t_0, the vehicle states $\mathbf{q}_0^i$ and an (approximate) estimation error covariance $\mathbf{P}_0$ are specified. Design a set of control trajectories $\mathbf{u}^i(t)$ for all sensor vehicles over the time window $[t_0, t_K]$ that balances the control effort and forecast quality at time t_F, conditioned on the measurements taken by the vehicles at times $\{t_1, t_2, \cdots, t_K\}$.

Using terminology established in the atmospheric science community, the measurement times $\{t_0, t_1 \cdots t_K\}$ are called target times and the vehicle positions at those times are called target sites. t_F is called the verification time. Normally, one would want to improve forecast quality over some important regions (such as urban areas); these regions are called the verification sites.

This AO objective is succinctly described by an optimization problem where a (scalar) cost metric J is minimized through the optimization variable $\mathbf{u}^i(t)$:

$$J = \frac{1}{2}\text{trace}\left(\mathbf{T}\mathbf{P}_F\right) + \sum_{i=1}^{M}\left(\int_{t_0}^{t_K} a^i(\mathbf{q}^i(t), \mathbf{u}^i(t))\,\mathrm{d}t + b^i(\mathbf{q}_K^i)\right), \tag{4.6}$$

where $\mathbf{T}$ masks $\mathbf{P}_F$ to reveal the target sites, $a^i(\cdot,\cdot)$ is the vehicle state and control trajectory penalty function, and $b^i(\cdot)$ penalizes the terminal vehicle state at t_K. For now the forecast quality metric is restricted to (4.6), extension to other cost metrics will be discussed in §4.4. Note that since $\mathbf{u}^i(t)$ affects the cost function nonlinearly, J is in general non-convex with many local minimum.

4.2 Stochastic EnVO

In this section the stochastic EnVO algorithm will be derived by first introducing the stochastic EnKF in §4.2.1, followed by the corresponding EnVO derivation in §4.2.2.

4.2.1 Stochastic EnKF

Suppose there are N ensemble members $\mathbf{x}^j$ where each member represent an estimate of the true state $\mathbf{x}$. Each ensemble member can be thought of as an expert, so a good estimate of $\mathbf{x}$ among these ensemble members is the ensemble mean, defined as the mean of all ensemble members $\hat{\mathbf{x}} = 1/N\sum_{j=1}^{N}\mathbf{x}^j$. The sample covariance $\mathbf{P}$ approximates the true covariance through the the ensemble spread

$$\mathbf{P} = \frac{1}{N-1}\sum_{j=1}^{N}(\mathbf{x}^j - \hat{\mathbf{x}})(\mathbf{x}^j - \hat{\mathbf{x}})^T. \tag{4.7}$$

The readers should not confuse the full covariance in Chapter 2 and 3 with the sample covariance here in (4.7). In an effort to achieve computational feasibility, the ensemble covariance approximation of the true covariance is used in this chapter; but to draw parallel with Chapter 2 and 3, the variable $\mathbf{P}$ is recycled here to denote the ensemble covariance.

Equation (4.7) is frequently expressed as $\mathbf{P} = \mathbf{Z}(\mathbf{Z})^T$ or $\mathbf{P} = \mathbf{X}\mathbf{Q}\mathbf{X}^T$, where the j-th column of $\mathbf{Z}$ and $\mathbf{X}$ contains respectively $(\mathbf{x}^j - \hat{\mathbf{x}})/\sqrt{N-1}$ and $\mathbf{x}^j$, and $\mathbf{Q} = (\mathbf{I} - \mathbf{1}_N)/(N-1)$ with $\mathbf{I}$ being the identity matrix and $\mathbf{1}_N$ is a $N \times N$ matrix with each element being $1/N$. The matrix $\mathbf{I} - \mathbf{1}_N$ is a symmetric and idempotent matrix with rank $N-1$; thus $(\mathbf{I} - \mathbf{1}_N)(\mathbf{I} - \mathbf{1}_N)^T = \mathbf{I} - \mathbf{1}_N$. It's easy to show that $\mathbf{Z} = \sqrt{N-1}\mathbf{X}\mathbf{Q}$.

Similar to the traditional KF time-update phase, each ensemble member in $\mathbf{X}$ is propagated using the full nonlinear model dynamics between measurement updates. At the end of the propagation the prior ensemble covariance $\mathbf{P}_k^-$ is computed from the prior ensemble $\mathbf{X}_k^-$ using (4.7). During the measurement-update phase, an observation $\mathbf{y}_k$ is taken and an ensemble of measurements is generated, where the j-th ensemble measurement $\mathbf{y}_k^j$ is defined as

$$\mathbf{y}_k^j \sim N(\mathbf{y}_k, \mathbf{R}_k), \qquad \mathbf{Y}_k = \begin{bmatrix} \mathbf{y}_k^1 & \mathbf{y}_k^2 & \cdots & \mathbf{y}_k^N \end{bmatrix}. \tag{4.8}$$

$\mathbf{X}_k^-$ is updated with this measurement ensemble to yield the *posterior* ensemble $\mathbf{X}_k^+$ using the update equation

$$\mathbf{X}_k^+ = \mathbf{X}_k^- + \mathbf{P}_k^- \mathbf{H}_k^T \left(\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k\right)^{-1} \left(\mathbf{Y}_k - \mathbf{H}_k \mathbf{X}_k^-\right), \tag{4.9}$$

where $\mathbf{H}_k$ is $h_k(\mathbf{x}_k)$ linearized about $\hat{\mathbf{x}}_k$. In this chapter the special case when $h_k(\mathbf{x})$ is linear in $\mathbf{x}_k$ is considered, so that $h_k(\mathbf{x}) = \mathbf{H}_k\mathbf{x}$. In practice this is not a strict assumption as the state (e.g. velocity and concentration) is typically directly measured. The readers are reminded that through earlier definition, $\mathbf{H}_k$ is dependent on $\mathbf{q}_k^i$. Note (4.9) can be interpreted as each ensemble member performs an independent KF update, except with the differences that the measurement $\mathbf{y}_k^j$ and the sample covariance $\mathbf{P}_k^-$ are used instead. The posterior covariance $\mathbf{P}_k^+$ can be calculated from $\mathbf{X}_k^+$.

Equation (4.9) is called the stochastic EnKF because $\mathbf{Y}_k$ is generated from applying random perturbation to $\mathbf{y}_k$. This is necessary in order for the stochastic EnKF to be statistically consistent with the Kalman Filter (Burgers *et al.*, 1998). Butala *et al.* (2008) formally shows in the limit of an infinitely large ensemble, the stochastic EnKF estimate asymptotically converge to the Kalman estimate with linear systems and Gaussian noise.

4.2.2 Formulation

Assume the initial posterior state ensemble $\mathbf{X}_0^+$, initial vehicle state $\mathbf{q}_0^i$, and a set of nominal vehicle control trajectories $\mathbf{u}^i(t)$ are given. A set of future vehicle trajectories $\mathbf{q}^i(t)$ is computed by applying $\mathbf{u}^i(t)$ to (4.1). From $\mathbf{q}^i(t)$ the future target sites are extracted, allowing the computation of future $\mathbf{H}_k$, so that $\mathbf{X}_0^+$ can be propagated to $\mathbf{X}_F$ through (4.9) and compute $\mathbf{P}_F$. From $\mathbf{P}_F$ and $\mathbf{u}^i(t)$ one could evaluate J using (4.6).

One consequence of using the stochastic EnKF to forecast the covariance is that the future measurement values are needed. Clearly, the future measurements cannot be known in advance; the next best thing is to predict the most likely measurements. Since at current time t_0 the best state estimate is $\hat{\mathbf{x}}_0^+$, the best future measurement prediction at t_k is made by applying the measurement operator to a forecast made from $\hat{\mathbf{x}}_0^+$. To this end, the predicted measurement is $\hat{\mathbf{y}}_k = \mathbf{H}_k\mathbf{A}_{0,k}\hat{\mathbf{x}}_0^+ = \mathbf{H}_k\hat{\mathbf{x}}_k^-$, where $\mathbf{A}_{0,k}$ is the forward propagator mapping $\mathbf{x}_0$ to $\mathbf{x}_k$. Applying random perturbations to $\hat{\mathbf{y}}_k$ and substitute the resulting measurement ensemble into (4.9) to compute the posterior ensemble mean, it follows that $\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^-$ because the measurement ensemble is zero-mean. This shows the stochastic EnKF remains unbiased when predicted measurements are used, which is consistent with the KF when the innovation becomes zero. Note that although the ensemble mean does not change, each ensemble member is updated.

Now imagine small perturbations are applied to the nominal $\mathbf{u}^i(t)$, from the first-order perturbation to J, J', is

$$J' = \sum_{i=1}^{M} \int_{t_0}^{t_K} \left(\nabla_{\mathbf{u}^i(t)} J\right)^T \mathbf{u}^i(t)' \, \mathrm{d}t, \tag{4.10}$$

where $\nabla_{\mathbf{u}^i(t)} J$ is gradient information necessary for any gradient-based optimization methods to optimize $\mathbf{u}^i(t)$. Computing $\nabla_{\mathbf{u}^i(t)} J$ is not straight forward, since (4.6) is also directly dependent on $\mathbf{P}_F$ and $\mathbf{q}^i(t)$, which indirectly dependent on $\mathbf{u}^i(t)$ as depicted in Figure 3.5 in Chapter 3. The adjoint analysis shall be used to make the dependencies on $\mathbf{u}^i(t)$ more apparent.

The perturbations in $\mathbf{u}^i(t)$ causes a chain reaction that also perturb $\mathbf{P}_F$ (or

equivalently, $\mathbf{X}_F$) and $\mathbf{q}^i(t)$, the first-order perturbation to J is

$$\begin{aligned} J' &= \operatorname{trace}\left(\left(\mathbf{T}\mathbf{X}_F\mathbf{Q}\right)^T \mathbf{X}_F'\right) \\ &\quad + \sum_{i=1}^{M}\left(\int_{t_0}^{t_K} \left(\nabla_{\mathbf{q}} a^i\right)^T \mathbf{q}^i(t)' + \left(\nabla_{\mathbf{u}} a^i\right)^T \mathbf{u}^i(t)'\,\mathrm{d}t + \left(\nabla_{\mathbf{q}} b^i\right)^T {\mathbf{q}_K^i}'\right), \end{aligned} \tag{4.11a}$$

$$\nabla_{\mathbf{q}} a^i \triangleq \frac{\partial a^i(\mathbf{q}^i(t),\mathbf{u}^i(t))}{\partial \mathbf{q}^i(t)}, \quad \nabla_{\mathbf{u}} a^i \triangleq \frac{\partial b^i(\mathbf{q}^i(t),\mathbf{u}^i(t))}{\partial \mathbf{u}^i(t)}, \quad \nabla_{\mathbf{q}} b^i \triangleq \frac{\mathrm{d}b^i(\mathbf{q}_K^i)}{\mathrm{d}\mathbf{q}_K^i}, \tag{4.11b}$$

where the trace identity is used to pull $\mathbf{X}_F'$ to the right in (4.11a). An ensemble of state adjoint vectors $\mathbf{s}^j \in \mathbb{R}^n$ are defined, with the resulting state adjoint ensemble defined as $\mathbf{S} \in \mathbb{R}^{n\times N}$. $\mathbf{S}_k$ can be interpreted as the sensitivity of $\mathbf{X}_k$ to some cost J, so that $J' = \operatorname{trace}\left(\mathbf{S}_k^T\mathbf{X}_k'\right)$. To this end, from (4.11a) $\mathbf{S}_F = T\mathbf{X}_F\mathbf{Q}$. The ensemble mean and spread concept equally apply to $\mathbf{S}$, where $\hat{\mathbf{s}} = 1/N\sum_{j=1}^{N}\mathbf{s}^j$ is the best sensitivity estimate, and $\mathbf{S}\mathbf{Q}\mathbf{S}^T$ represents the uncertainty in the sensitivity estimate. This is the main difference between the EnVO algorithm from existing sensitivity-based AO algorithms, where the uncertainty on the sensitivity calculation is quantified through an adjoint ensemble.

Since $\mathbf{x}_F^j$ is dynamically connected to $\mathbf{x}_K^{j+}$ through (4.1), the adjoint operator $\mathbf{A}_{K,F}^T$ can be used to relate $\mathbf{S}_F'$ to ${\mathbf{S}_K^+}'$. Equation (4.11a) now becomes

$$\begin{aligned} J' = \operatorname{trace}\Big(\Big(\underbrace{\left(\mathbf{A}_{K,F}\right)^T\mathbf{S}_F}_{\mathbf{S}_K^+}\Big)^T {\mathbf{X}_K^+}'\Big) \\ + \sum_{i=1}^{M}\left(\int_{t_0}^{t_K} \left(\nabla_{\mathbf{q}} a^i\right)^T \mathbf{q}^i(t)' + \left(\nabla_{\mathbf{u}} a^i\right)^T \mathbf{u}^i(t)'\,\mathrm{d}t + \left(\nabla_{\mathbf{q}} b^i\right)^T {\mathbf{q}_K^i}'\right). \end{aligned} \tag{4.12}$$

Like traditional adjoint analysis, in (4.12) each state adjoint ensemble member is propagated backward in time using the adjoint operator; the only difference here is that an ensemble of adjoints are propagated instead of one.

At each measurement time $\mathbf{X}_k^-$ is updated to $\mathbf{X}_k^+$ using (4.9); in essence, the state ensemble experiences a secondary evolution equation at each measurement time. Therefore in order to propagate $\mathbf{S}_K^+$ further, adjoint analysis must be applied to (4.9) in order to "update" $\mathbf{S}_K^+$ into $\mathbf{S}_K^-$. The EnKF update equation (4.9) is restated here, with the addition of predicted measurement and additional

shorthand definitions $\mathbf{\Psi}_k$, $\mathbf{\Phi}_k$, and $\mathbf{\Theta}_k$:

$$\mathbf{X}_k^+ = \mathbf{X}_k^- + \underbrace{\mathbf{P}_k\mathbf{H}_k^T}_{\mathbf{\Psi}_k}\underbrace{(\mathbf{H}_k\mathbf{P}_k\mathbf{H}_k^T + \mathbf{R}_k)^{-1}}_{\mathbf{\Phi}_k^{-1}}\underbrace{(\mathbf{H}_k\mathbf{X}_k^-\mathbf{1}_N + \mathbf{E}_k - \mathbf{H}_k\mathbf{X}_k^-)}_{\mathbf{\Theta}_k}, \tag{4.13}$$

where $\mathbf{H}_k\mathbf{X}_k^-\mathbf{1}_N + \mathbf{E}_k$ is the predicted measurement and $\mathbf{E}_k$ is the ensemble of measurement perturbations with distribution $N(\mathbf{0}, \mathbf{R}_k)$. The same perturbation in $\mathbf{u}^i(t)$ also perturbs (4.13), the first-order perturbation is

$$\mathbf{X}_k^{+\prime} = \mathbf{X}_k^{-\prime} + \mathbf{\Psi}_k'\mathbf{\Phi}_k^{-1}\mathbf{\Theta}_k + \mathbf{\Psi}_k(\mathbf{\Phi}_k^{-1})'\mathbf{\Theta}_k + \mathbf{\Psi}_k\mathbf{\Phi}_k^{-1}\mathbf{\Theta}_k', \qquad \text{where} \tag{4.14a}$$

$$\mathbf{\Psi}_k' = \mathbf{P}_k'\mathbf{H}_k^T + \mathbf{P}_k(\mathbf{H}_k')^T, \tag{4.14b}$$

$$\mathbf{P}_k' = \mathbf{X}_k^{-\prime}\mathbf{Q}(\mathbf{X}_k^-)^T + \mathbf{X}_k^-\mathbf{Q}(\mathbf{X}_k^{-\prime})^T, \tag{4.14c}$$

$$\mathbf{H}_k' = \sum_{i=1}^{M}\sum_{p=1}^{size(\mathbf{q}^i)}\left(\frac{\mathrm{d}\mathbf{H}_k}{\mathrm{d}\mathbf{q}_k^{i,p}}\right)^T \mathbf{q}_k^{i,p\prime}, \qquad \mathbf{q}_k^{i,p} \text{ denotes the } p\text{-th element of } \mathbf{q}_k^i. \tag{4.14d}$$

$$\begin{aligned}(\mathbf{\Phi}_k^{-1})' &= -\mathbf{\Phi}_k^{-1}\mathbf{\Phi}_k'\mathbf{\Phi}_k^{-1} \qquad \text{taken from Petersen \& Pedersen (2008).}\\ &= -\mathbf{\Phi}_k^{-1}(\mathbf{H}_k'\mathbf{P}_k\mathbf{H}_k^T + \mathbf{H}_k\mathbf{P}_k'\mathbf{H}_k^T + \mathbf{H}_k\mathbf{P}_k(\mathbf{H}_k')^T + \mathbf{R}_k')\mathbf{\Phi}_k^{-1},\end{aligned} \tag{4.14e}$$

$$\mathbf{R}_k' = \sum_{i=1}^{M}\sum_{p=1}^{size(\mathbf{q}^i)}\left(\frac{\mathrm{d}\mathbf{R}_k}{\mathrm{d}\mathbf{q}_k^{i,p}}\right)^T \mathbf{q}_k^{i,p\prime}, \tag{4.14f}$$

$$\mathbf{\Theta}_k' = (\mathbf{H}_k'\mathbf{X}_k^- + \mathbf{H}_k\mathbf{X}_k^{-\prime})(\mathbf{1}_N - \mathbf{I}), \tag{4.14g}$$

where $\mathbf{E}_k' = \mathbf{0}$ because it is not affected by the perturbations. Substituting (4.14) into (4.12) with $k = K$, and gather all the perturbation terms to the right using

the trace identity, (4.12) now becomes

$$J' = \text{trace}\left((\mathbf{S}_K^-)^T \mathbf{X}_K^{-\,\prime}\right) + \sum_{i=1}^{M} \sum_{p=1}^{size(\mathbf{q}^i)} \Upsilon^{i,p}(\mathbf{S}_K^+)\mathbf{q}_K^{i,p\,\prime} + \sum_{i=1}^{M} \left(\int_{t_0}^{t_K} \left(\nabla_{\mathbf{q}} a^i\right)^T \mathbf{q}^i(t)' + \left(\nabla_{\mathbf{u}} a^i\right)^T \mathbf{u}^i(t)'\,\mathrm{d}t + \left(\nabla_{\mathbf{q}} b^i\right)^T \mathbf{q}_K^{i\,\prime} \right), \tag{4.15a}$$

$$\begin{aligned} \mathbf{S}_K^- = \mathbf{S}_K^+ &+ \mathbf{H}_K^T \mathbf{\Phi}_K^{-1} \mathbf{\Psi}_K^T \mathbf{S}_K^+ (\mathbf{1}_N - \mathbf{I}) \\ &+ \mathbf{S}_K^+ \mathbf{\Theta}_K^T \mathbf{\Phi}_K^{-1} \mathbf{H}_K \mathbf{X}_K^- \mathbf{Q} + \mathbf{H}_K^T \mathbf{\Phi}_K^{-1} \mathbf{\Theta}_K (\mathbf{S}_K^+)^T \mathbf{X}_K^- \mathbf{Q} \\ &- \mathbf{H}_K^T \mathbf{\Phi}_K^{-1} \mathbf{\Psi}_K^T \mathbf{S}_K^+ \mathbf{\Theta}_K^T \mathbf{\Phi}_K^{-1} \mathbf{H}_K \mathbf{X}_K^- \mathbf{Q} \\ &- \mathbf{H}_K^T \mathbf{\Phi}_K^{-1} \mathbf{\Theta}_K (\mathbf{S}_K^+)^T \mathbf{\Psi}_K \mathbf{\Phi}_K^{-1} \mathbf{H}_K \mathbf{X}_K^- \mathbf{Q}, \end{aligned} \tag{4.15b}$$

$$\begin{aligned} \Upsilon^{i,p}(\mathbf{S}_K^+) = \text{trace}\Big(&\mathbf{P}_K \mathbf{S}_K^+ \mathbf{\Theta}_K^T \mathbf{\Phi}_K^{-1} \frac{\mathrm{d}\mathbf{H}_K}{\mathrm{d}\mathbf{q}_K^{i,p}} + \mathbf{X}_K^+ (\mathbf{1}_N - \mathbf{I})(\mathbf{S}_K^+)^T \mathbf{\Psi}_K \mathbf{\Phi}_K^{-1} \frac{\mathrm{d}\mathbf{H}_K}{\mathrm{d}\mathbf{q}_K^{i,p}} \\ &- \mathbf{\Psi}_K \mathbf{\Phi}_K^{-1} \mathbf{\Theta}_K (\mathbf{S}_K^+)^T \mathbf{\Psi}_K \mathbf{\Phi}_K^{-1} \frac{\mathrm{d}\mathbf{H}_K}{\mathrm{d}\mathbf{q}_K^{i,p}} - \mathbf{\Psi}_K \mathbf{\Phi}_K^{-1} \mathbf{\Psi}_K^T \mathbf{S}_K^+ \mathbf{\Theta}_K^T \mathbf{\Phi}_K^{-1} \frac{\mathrm{d}\mathbf{H}_K}{\mathrm{d}\mathbf{q}_K^{i,p}} \\ &- \mathbf{\Phi}_K \mathbf{\Theta}_K (\mathbf{S}_K^+)^T \mathbf{\Psi}_K \mathbf{\Phi}_K^{-1} \frac{\mathrm{d}R_K}{\mathrm{d}\mathbf{q}_K^{i,p}} \Big). \end{aligned} \tag{4.15c}$$

The adjoint update equation (4.15b) that maps the adjoint ensemble $\mathbf{S}_K^+$ to $\mathbf{S}_K^-$ (and more generally, for all subsequent adjoint ensemble updates $\mathbf{S}_k^+$ to $\mathbf{S}_k^-$) have been derived. During each adjoint ensemble update, additional sensitivity dependence on $\mathbf{q}_k^i$ is picked up in the form of (4.15c). This is expected since the update from $\mathbf{X}_k^-$ to $\mathbf{X}_k^+$ not only depends on $\mathbf{X}_k^-$, but also depends on $\mathbf{q}_k^i$.

Equation (4.15a) bears strong similarity to (4.11a), because technically speaking $\mathbf{X}_F \triangleq \mathbf{X}_F^-$; the only differences are the shifted time index in the first term, and the addition of the second term in (4.15a). Therefore similar to (4.11a), $\mathbf{S}_K^-$ can be propagated to t_{K-1} using the adjoint operator $\mathbf{A}_{K-1,K}^T$ to become $\mathbf{S}_{K-1}^+$, where it is update to $\mathbf{S}_{K-1}^-$ using (4.15b) with time index shift to $K-1$ and pick up an additional $\mathbf{q}_{K-1}^i$ dependency. This process is repeated until $\mathbf{S}_k$ is propagated

to $\mathbf{S}_0^+$, at which the perturbed cost now has the following form

$$J' = \operatorname{trace}\Big((\mathbf{S}_0^+)^T \underbrace{\mathbf{X}_0^{+\,\prime}}_{0}\Big) + \sum_{k=1}^{K}\sum_{i=1}^{M}\sum_{p=1}^{size(\mathbf{q}^i)} \Upsilon^{i,p}(\mathbf{S}_k^+)\mathbf{q}_k^{i,p\,\prime} + \sum_{i=1}^{M}\left(\int_{t_0}^{t_K} \left(\nabla_{\mathbf{q}} a^i\right)^T \mathbf{q}^i(t)' + \left(\nabla_{\mathbf{u}} a^i\right)^T \mathbf{u}^i(t)'\,\mathrm{d}t + \left(\nabla_{\mathbf{q}} b^i\right)^T \mathbf{q}_K^{i\,\prime}\right). \quad (4.16)$$

Since the initial ensemble $\mathbf{X}_0^+$ is given, it is not affected by the perturbation and thus is zero; hence the first term in (4.16) vanishes. The original perturbed cost (4.11a), which is dependent on the perturbations $\mathbf{X}'_F$, $\mathbf{q}^i(t)'$, and $\mathbf{u}^i(t)'$, has now being re-written as a function that is dependent only on perturbations $\mathbf{q}^i(t)'$ and $\mathbf{u}^i(t)'$. Further efforts are needed to repose the $\mathbf{q}^i(t)'$ dependence to $\mathbf{u}^i(t)'$.

Perturbing the vehicle dynamical equation (4.2) to explore the relationship between $\mathbf{q}^i(t)'$ and $\mathbf{u}^i(t)'$:

$$\mathrm{L}(\mathbf{q}^i(t)') = \mathbf{G}^i(t)\mathbf{u}^i(t)', \quad (4.17a)$$

$$\mathrm{L} \triangleq \frac{\mathrm{d}}{\mathrm{d}t} - \mathbf{F}^i(t), \quad \mathbf{F}^i(t) \triangleq \frac{\partial g(\mathbf{q}^i(t), \mathbf{u}^i(t))}{\partial \mathbf{q}^i(t)}, \quad \mathbf{G}^i(t) \triangleq \frac{\partial g(\mathbf{q}^i(t), \mathbf{u}^i(t))}{\partial \mathbf{u}^i(t)}. \quad (4.17b)$$

M vehicle adjoint vectors $\mathbf{r}^i(t)$ are defined and an adjoint identity is framed based on a relevant inner product:

$$\langle \mathbf{r}^i, \mathrm{L}(\mathbf{q}^{i\,\prime})\rangle_{a,b} = \langle \mathrm{L}^*(\mathbf{r}^i), \mathbf{q}^{i\,\prime}\rangle_{a,b} + c_{a,b}, \qquad \langle \mathbf{x}, \mathbf{y}\rangle_{a,b} \triangleq \int_{t_a}^{t_b} \mathbf{x}(t)^T\mathbf{y}(t)\,\mathrm{d}t. \quad (4.18)$$

Using integration-by-parts, it can be shown that $\mathrm{L}^* = -\mathrm{d}/\mathrm{d}t - \mathbf{F}^i(t)^T$ and $c_{a,b} = (\mathbf{r}^i(t_b))^T\mathbf{q}^i(t_b)' - (\mathbf{r}^i(t_a))^T\mathbf{q}^i(t_a)'$. Equation (4.18) is defined between the time interval $[t_{K-1}, t_K]$ so that together with (4.17a):

$$\langle \mathbf{r}^i, \mathbf{G}^i\mathbf{u}^{i\,\prime}\rangle_{K-1,K} = \langle \mathrm{L}^*(\mathbf{r}^i), \mathbf{q}^{i\,\prime}\rangle_{K-1,K} + (\mathbf{r}_K^i)^T\mathbf{q}_K^{i\,\prime} - (\mathbf{r}_{K-1}^i)^T\mathbf{q}_{K-1}^{i\,\prime}. \quad (4.19)$$

Comparing (4.19) and (4.16), if

$$\mathrm{L}^*(\mathbf{r}^i) = \nabla_{\mathbf{q}} a^i, \qquad \mathbf{r}_K^i = \nabla_{\mathbf{q}} b^i + \mathbf{T}_K^i, \quad (4.20)$$

where $\mathbf{T}^i \in \mathbb{R}^{size(\mathbf{q}^i)}$ and the p-th element of $\mathbf{T}_K^i$ is $\Upsilon^{i,p}(\mathbf{S}_K^+)$, then equation (4.16)

can be transformed into

$$J' = \sum_{k=1}^{K-1}\sum_{i=1}^{M}\sum_{p=1}^{size(\mathbf{q}^i)} \Upsilon^{i,p}(\mathbf{S}_k^+)\mathbf{q}_k^{i,p\,\prime} + \sum_{i=1}^{M}\Bigg(\int_{t_0}^{t_{K-1}} \left(\nabla_{\mathbf{q}}a^i\right)^T \mathbf{q}^i(t)' + \left(\nabla_{\mathbf{u}}a^i\right)^T \mathbf{u}^i(t)'\,\mathrm{d}t + \langle(\mathbf{G}^i)^T\mathbf{r}^i + \nabla_{\mathbf{u}}a^i, \mathbf{u}^{i\,\prime}\rangle_{K-1,K} + (\mathbf{r}_{K-1}^i)^T\mathbf{q}_{K-1}^{i\,\prime}\Bigg). \quad (4.21)$$

Note when $\mathrm{L}^*(\mathbf{r}^i)$ and $\mathbf{r}_K^i$ are defined in (4.20), an adjoint dynamical equation and the starting condition at time t_K for $\mathbf{r}^i(t)$ have been defined:

$$\frac{\mathrm{d}\mathbf{r}^i}{\mathrm{d}t} = -\mathbf{F}^i(t)^T\mathbf{r}^i + \nabla_{\mathbf{q}}a^i, \qquad \mathbf{r}_K^i = \nabla_{\mathbf{q}}b^i + \mathbf{T}_K^i. \quad (4.22)$$

Equation (4.21) bears strong resemblance to (4.16); the differences are the shifted time index from t_K to t_{K-1}, $\nabla_{\mathbf{q}}b^i$ is replaced with $\mathbf{r}_{K-1}^i$, and the additional $\langle(\mathbf{G}^i)^T\mathbf{r}^i + \nabla_{\mathbf{u}}a^i, \mathbf{u}^{i\,\prime}\rangle_{K-1,K}$ term. Thus naturally if the steps between (4.18) and (4.20) are repeated by defining new vehicle adjoint starting conditions and propagate them for all subsequent measurement time intervals $[t_{k-1}, t_k]$, and recognizing that $\mathbf{q}_0^{i\,\prime} = 0$, (4.21) would eventually be re-expressed as

$$\begin{aligned} J' &= \sum_{i=1}^{M}\sum_{k=1}^{K}\langle(\mathbf{G}^i)^T\mathbf{r}^i + \nabla_{\mathbf{u}}a^i, \mathbf{u}^{i\,\prime}\rangle_{k-1,k} \\ &= \sum_{i=1}^{M}\int_{t_0}^{t_K}\Bigg(\underbrace{(\mathbf{G}^i(t))^T\mathbf{r}^i(t) + \nabla_{\mathbf{u}}a^i}_{\nabla_{\mathbf{u}^i(t)}J}\Bigg)^T\mathbf{u}^i(t)'\,\mathrm{d}t, \end{aligned} \quad (4.23)$$

$$\mathrm{L}^*(\mathbf{r}^i) = \nabla_{\mathbf{q}}a^i, \qquad (\mathbf{r}_k^i)^- = \begin{cases} \nabla_{\mathbf{q}}b^i + \mathbf{T}_K^i, & k = K \\ (\mathbf{r}_k^i)^+ + \mathbf{T}_k^i, & k \neq K \end{cases}. \quad (4.24)$$

Because additional sensitivity on $\mathbf{q}_k^i$ is picked up during each adjoint ensemble update, the vehicle adjoints also experience an update during each target time, hence the prior and posterior distinction in (4.24). The gradient information $\nabla_{\mathbf{u}^i(t)}J$ could now be readily extracted from equation (4.23) for gradient-based optimization methods.

Note due to the measurement perturbations, $\mathbf{\Theta}_k$ is a random variable; therefore J and $\nabla_{\mathbf{u}^i(t)}J$ are also random variables. Like the stochastic EnKF, this also gives the resulting EnVO algorithm a stochastic flavor. This also means given the same $\mathbf{u}^i(t)$, J and $\nabla_{\mathbf{u}^i(t)}J$ are slightly different for each optimization iteration.

4.3 Deterministic EnVO

In this section the deterministic EnVO algorithm will be derived by first introducing the deterministic ESRF in §4.3.1, followed by the corresponding EnVO derivation in §4.3.2.

4.3.1 Deterministic EnKF

Deterministic EnKF such as the Ensemble Transform Kalman Filter (ETKF, Bishop *et al.*, 2001) and the Ensemble Adjust Kalman Filter (EAKF, Khare, 2004) are variations of the Ensemble Square-Root Filter (ESRF, Tippett *et al.*, 2003), which derives its idea from the Kalman Square-Root Filter.

Unlike the stochastic EnKF, the ESRF focuses on the covariance evolution and update implicitly through $\mathbf{Z}$. The sample covariance evolution equations for the ESRF are

$$\mathbf{P}_k^- = \mathbf{A}_{k-1,k}\mathbf{P}_{k-1}^+\mathbf{A}_{k-1,k}^T, \tag{4.25a}$$

$$\mathbf{P}_k^+ = \left(\mathbf{I} - \mathbf{P}_k^-\mathbf{H}_k^T\left(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k\right)^{-1}\mathbf{H}_k\right)\mathbf{P}_k^-, \tag{4.25b}$$

where $\mathbf{A}_{k-1,k}$ is the tangent linear operator (like a state transition matrix) mapping the $\mathbf{x}_{k-1}$ to $\mathbf{x}_k$. Using the definition $\mathbf{P} = \mathbf{Z}\mathbf{Z}^T$ and substitute into (4.25), the square-root version emerges:

$$\mathbf{Z}_k^- = \mathbf{A}_{k-1,k}\mathbf{Z}_{k-1}^+, \tag{4.26a}$$

$$\mathbf{Z}_k^+ = \mathbf{Z}_k^-\mathbf{V}_k\mathbf{U}_k, \tag{4.26b}$$

where $\mathbf{V}_k$ is defined as

$$\mathbf{V}_k\mathbf{V}_k^T = \mathbf{I} - (\mathbf{Z}_k^-)^T\mathbf{H}_k^T\left(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k\right)^{-1}\mathbf{H}_k\mathbf{Z}_k^-, \tag{4.27}$$

and $\mathbf{U}_k$ is an arbitrary unitary matrix. Note that by the Matrix Inversion Lemma,

$$\mathbf{I} - (\mathbf{Z}_k^-)^T\mathbf{H}_k^T\left(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k\right)^{-1}\mathbf{H}_k\mathbf{Z}_k^- = \left(\mathbf{I} + (\mathbf{Z}_k^-)^{-1}\mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{H}_k\mathbf{Z}_k^-\right)^{-1}, \tag{4.28}$$

thus as long as $\mathbf{R}_k$ is positive-definite, the RHS of (4.27) is also positive-definite; therefore $\mathbf{V}_k$ could always be computed using the Cholesky factorization, and that

$\mathbf{V}_k^{-1}$ exists. In this framework, the $\mathbf{Z}$ propagation and update (and thus implicitly the $\mathbf{P}$ propagation and update) does not require knowing the actual measurement or measurement ensemble, hence measurement prediction is not necessary. Also, because the ESRF is designed to be consistent with the KF second-order statistics, no stochastic perturbations of any sort are necessary; this is why the term "deterministic" is coined for the ESRF.

Similar to the unbiased property in the stochastic EnKF when using the predicted measurement, the same property is also implicit for ESRF. As mentioned in Livings *et al.* (2008), the EAKF is unbiased while the original ETKF formulation presented in Bishop *et al.* (2001) is not. However Livings *et al.* noted that an unbiased version of the ETKF is presented in Wang *et al.* (2004, appendix Ca).

4.3.2 Formulation

The analysis performed on the stochastic EnKF which yields the stochastic EnVO algorithm can equally be applied to the deterministic EnKF and give rise to the deterministic EnVO algorithm. Since there are various flavors to the deterministic EnKF, the following formulation shall be restricted to the ESRF, which is the basis to other deterministic EnKFs. For convenience notations defined in §4.2.2 will be used here without explicit definition, and several variable names will be recycled here to draw parallel against the stochastic EnVO algorithm.

Assume the same $\mathbf{X}_0^+$, $\mathbf{q}_0^i$, and $\mathbf{u}^i(t)$ data are given; it is easy to convert $\mathbf{X}_0^+$ to $\mathbf{Z}_0^+$ by $\mathbf{Z}_0^+ = \sqrt{N-1}\mathbf{X}_0^+\mathbf{Q}$. From these data a forecast of the future vehicle trajectories can be made, from which the future target site information are gathered, and the forecast from $\mathbf{Z}_0^+$ to $\mathbf{Z}_F$ is made to compute $\mathbf{P}_F$; the cost J can then be evaluated.

Now the nominal vehicle control trajectories $\mathbf{u}^i(t)$ are perturbed. Using the alternative representation $\mathbf{P}_F = \mathbf{Z}_F\mathbf{Z}_F^T$, the perturbation in the nominal $\mathbf{u}^i(t)$

causes a perturbation in J:

$$\begin{aligned} J' = &\operatorname{trace}\big((\underbrace{\mathbf{T}\mathbf{Z}_F}_{\mathbf{M}_F})^T \mathbf{Z}_F'\big) \\ &+ \sum_{i=1}^{M} \left(\int_{t_0}^{t_K} \left(\nabla_{\mathbf{q}} a^i\right)^T \mathbf{q}^i(t)' + \left(\nabla_{\mathbf{u}} a^i\right)^T \mathbf{u}^i(t)' \,\mathrm{d}t + \left(\nabla_{\mathbf{q}} b^i\right)^T {\mathbf{q}_K^i}' \right). \end{aligned} \tag{4.29}$$

An ensemble of state adjoint vectors $\mathbf{m}^j \in \mathbb{R}^n$ are defined, with the collection of ensemble members denoting $\mathbf{M} \in \mathbb{R}^{n \times N}$. Similar to §4.2.2, $\mathbf{M}_k$ is the sensitivity of $\mathbf{Z}_k$; the same ensemble mean and spread concept also applies to $\mathbf{M}$. Therefore, $\mathbf{M}_F = \mathbf{T}\mathbf{Z}_F$. From (4.26), it is clear to see the same adjoint operator $\mathbf{A}_{K,F}^T$ can be applied here to propagate $\mathbf{M}_F$ to $\mathbf{M}_K^+$, where the new J' is now

$$\begin{aligned} J' = &\operatorname{trace}\big((\underbrace{\mathbf{A}_{K,F}^T \mathbf{M}_F}_{\mathbf{M}_K^+})^T {\mathbf{Z}_K^+}'\big) \\ &+ \sum_{i=1}^{M} \left(\int_{t_0}^{t_K} \left(\nabla_{\mathbf{q}} a^i\right)^T \mathbf{q}^i(t)' + \left(\nabla_{\mathbf{u}} a^i\right)^T \mathbf{u}^i(t)' \,\mathrm{d}t + \left(\nabla_{\mathbf{q}} b^i\right)^T {\mathbf{q}_K^i}' \right). \end{aligned} \tag{4.30}$$

Perturbation analysis to (4.26b) is needed to update ${\mathbf{M}_K^+}'$ to ${\mathbf{M}_K^-}'$; the perturbation to (4.26b) is

$${\mathbf{Z}_k^+}' = {\mathbf{Z}_k^-}' \mathbf{V}_k \mathbf{U}_k + \mathbf{Z}_k^- \mathbf{V}_k' \mathbf{U}_k, \tag{4.31}$$

where because $\mathbf{U}_k$ is assumed to be chosen arbitrary, it's not affected by the perturbation. Substitute (4.31) into (4.30) with $k = K$, and using the trace identity yields

$$\begin{aligned} J' = &\operatorname{trace}\big(\mathbf{V}_K \mathbf{U}_K (\mathbf{M}_K^+)^T {\mathbf{Z}_K^-}' + \mathbf{U}_K (\mathbf{M}_K^+)^T \mathbf{Z}_K^- \mathbf{V}_K'\big) \\ &+ \sum_{i=1}^{M} \left(\int_{t_0}^{t_K} \left(\nabla_{\mathbf{q}} a^i\right)^T \mathbf{q}^i(t)' + \left(\nabla_{\mathbf{u}} a^i\right)^T \mathbf{u}^i(t)' \,\mathrm{d}t + \left(\nabla_{\mathbf{q}} b^i\right)^T {\mathbf{q}_K^i}' \right). \end{aligned} \tag{4.32}$$

Clearly the first term in (4.32) is already in the right form, but the second term requires more work to re-express $\mathbf{V}_K'$ in terms of ${\mathbf{Z}_K^-}'$.

The perturbation to the definition of $\mathbf{V}_k$ in (4.27) is

$$\underbrace{\mathbf{V}_k'\mathbf{V}_k^T + \mathbf{V}_k\mathbf{V}_k^{T\,\prime}}_{\mathrm{D}(\mathbf{V}_k')} = \mathbf{C}_k', \tag{4.33a}$$

$$\begin{aligned}
\mathbf{C}_k &= (\mathbf{I} - (\mathbf{Z}_k^-)^T\mathbf{H}_k^T\mathbf{\Phi}_k^{-1}\mathbf{H}_k\mathbf{Z}_k^-),\\
\mathbf{C}_k' &= -(\mathbf{Z}_k^{-\,\prime})^T\mathbf{H}_k^T\mathbf{\Phi}_k^{-1}\mathbf{H}_k\mathbf{Z}_k^- - (\mathbf{Z}_k^-)^T\mathbf{H}_k^T\mathbf{\Phi}_k^{-1}\mathbf{H}_k\mathbf{Z}_k^{-\,\prime}\\
&\quad - (\mathbf{Z}_k^-)^T(\mathbf{H}_k')^T\mathbf{\Phi}_k^{-1}\mathbf{H}_k\mathbf{Z}_k^- - (\mathbf{Z}_k^-)^T\mathbf{H}_k^T\mathbf{\Phi}_k^{-1}\mathbf{H}_k'\mathbf{Z}_k^-\\
&\quad + (\mathbf{Z}_k^-)^T\mathbf{H}_k^T\mathbf{\Phi}_k^{-1}\left(\mathbf{H}_k'\mathbf{\Psi}_k + \mathbf{\Psi}_k^T(\mathbf{H}_k')^T + \mathbf{R}_k'\right)\mathbf{\Phi}_k^{-1}\mathbf{H}_k\mathbf{Z}_k^-\\
&\quad + (\mathbf{Z}_k^-)^T\mathbf{H}_k^T\mathbf{\Phi}_k^{-1}\left(\mathbf{H}_k\left(\mathbf{Z}_k^{-\,\prime}(\mathbf{Z}_k^-)^T + \mathbf{Z}_k^-(\mathbf{Z}_k^{-\,\prime})^T\right)\mathbf{H}_k^T\right)\mathbf{\Phi}_k^{-1}\mathbf{H}_k\mathbf{Z}_k^-.
\end{aligned} \tag{4.33b}$$

Note that from (4.33a), $\mathbf{V}_k'$ cannot be written as an explicit function of $\mathbf{C}_k'$; therefore the directly substitution scheme (equation (4.12) through (4.15a)) used in deriving the stochastic EnVO algorithm does not work here. Just as the ensemble update is viewed as a separate dynamic that occurs only during measurement times, so shall the Cholesky factorization in (4.27). A new matrix adjoint variable $\mathbf{N} \in \mathbb{R}^{N\times N}$ is defined along with a new adjoint identity

$$\langle\!\langle \mathbf{N}_k, \mathrm{D}(\mathbf{V}_k')\rangle\!\rangle = \langle\!\langle \mathrm{D}^*(\mathbf{N}_k), \mathbf{V}_k'\rangle\!\rangle, \qquad \langle\!\langle \mathbf{X}, \mathbf{Y}\rangle\!\rangle \triangleq \mathrm{trace}(\mathbf{X}^T\mathbf{Y}), \tag{4.34}$$

where $\mathrm{D}(\mathbf{V}_k')$ is defined in (4.33a). Substitute (4.33a) into (4.34), it is easy to see $\mathrm{D}^*(\mathbf{N}_k) = (\mathbf{N}_k + \mathbf{N}_k^T)\mathbf{V}_k$. Therefore, if $\mathrm{D}^*(\mathbf{N}_K) = (\mathbf{Z}_K^-)^T\mathbf{M}_K^+\mathbf{U}_K^T$, then (4.32) can be rewritten by leveraging (4.34) and noting $\mathrm{D}(\mathbf{V}_k') = \mathbf{C}_k'$ as

$$\begin{aligned}
J' = \mathrm{trace}\big(&\mathbf{V}_K\mathbf{U}_K(\mathbf{M}_K^+)^T\mathbf{Z}_K^{-\,\prime} + \mathbf{N}_K^T\mathbf{C}_K'\big)\\
&+ \sum_{i=1}^{M}\left(\int_{t_0}^{t_K}\left(\nabla_{\mathbf{q}}a^i\right)^T\mathbf{q}^i(t)' + \left(\nabla_{\mathbf{u}}a^i\right)^T\mathbf{u}^i(t)'\,\mathrm{d}t + \left(\nabla_{\mathbf{q}}b^i\right)^T\mathbf{q}_K^{i\,\prime}\right),
\end{aligned} \tag{4.35}$$

where

$$\mathbf{N}_K + \mathbf{N}_K^T = (\mathbf{Z}_K^-)^T\mathbf{M}_K^+\mathbf{U}_K^T\mathbf{V}_K^{-1}. \tag{4.36}$$

Note by construction, the LHS of (4.36) is symmetric; however it is not immediately obvious that the RHS is also symmetric. If $\mathbf{M}_K^+$ is expanded to contain $\mathbf{Z}_K^-$:

$$\begin{aligned}
(\mathbf{Z}_K^-)^T\mathbf{M}_K^+\mathbf{U}_K^T\mathbf{V}_K^{-1} &= (\mathbf{Z}_K^-)^T\mathbf{A}_{K,F}^T\mathbf{M}_F\mathbf{U}_K^T\mathbf{V}_K^{-1} = (\mathbf{Z}_K^-)^T\mathbf{A}_{K,F}^T T\mathbf{Z}_F\mathbf{U}_K^T\mathbf{V}_K^{-1}\\
&= (\mathbf{Z}_K^-)^T\mathbf{A}_{K,F}^T T\mathbf{A}_{K,F}\mathbf{Z}_K^+\mathbf{U}_K^T\mathbf{V}_K^{-1} = (\mathbf{Z}_K^-)^T\mathbf{A}_{K,F}^T T\mathbf{A}_{K,F}\mathbf{Z}_K^-,
\end{aligned} \tag{4.37}$$

it shows the the RHS of (4.36) is indeed symmetric. In fact it can be shown all subsequent $(\mathbf{Z}_k^-)^T\mathbf{M}_k^+\mathbf{U}_k^T\mathbf{V}_k^{-1}$ are symmetric (see Appendix A). Also note the solution to $\mathbf{N}_K$ is not unique, as any anti-symmetric matrix added to $\mathbf{N}_K$ also satisfies (4.36). However, from (4.35) if $\mathbf{N}_K$ is interpreted as the sensitivity of $\mathbf{C}_K$, then $\mathbf{N}_K$ must be symmetric because matrix square-root for $\mathbf{C}_K$ perturbed by a non-symmetric $\mathbf{N}_K$ does not exist. Thus, the only symmetric solution

$$\mathbf{N}_k = \frac{1}{2}(\mathbf{Z}_k^-)^T\mathbf{M}_k^+\mathbf{U}_k^T\mathbf{V}_k^{-1} \tag{4.38}$$

is taken during all computations of $\mathbf{N}_k$.

Finally, substitute (4.33b) into (4.35) with $k = K$ and use trace identity to pull the perturbation terms to the right, (4.35) becomes

$$\begin{aligned} J' = \operatorname{trace}\big((\mathbf{M}_K^-)^T\mathbf{Z}_K^{-\prime}\big) + \sum_{i=1}^{M}\sum_{p=1}^{size(\mathbf{q}^i)} \Upsilon^{i,p}(\mathbf{M}_K^+)\mathbf{q}_K^{i,p\,\prime} \\ + \sum_{i=1}^{M}\left(\int_{t_0}^{t_K} \left(\nabla_{\mathbf{q}}a^i\right)^T\mathbf{q}^i(t)' + \left(\nabla_{\mathbf{u}}a^i\right)^T\mathbf{u}^i(t)'\,\mathrm{d}t + \left(\nabla_{\mathbf{q}}b^i\right)^T\mathbf{q}_M^{i\,\prime}\right), \end{aligned} \tag{4.39a}$$

$$\begin{aligned} \mathbf{M}_K^- &= \mathbf{M}_K^+\mathbf{U}_K^T\mathbf{V}_K^T - 2(\mathbf{H}_K^T\mathbf{\Phi}_K^{-1}\mathbf{H}_K\mathbf{Z}_K^-\mathbf{N}_K)\underbrace{(\mathbf{I} - (\mathbf{Z}_K^-)^T\mathbf{H}_K^T\mathbf{\Phi}_K^{-1}\mathbf{H}_K\mathbf{Z}_K^-)}_{\mathbf{V}_K\mathbf{V}_K^T} \\ &= (\mathbf{I} - \mathbf{H}_K^T\mathbf{\Phi}_K^{-1}\mathbf{\Psi}_K^T)\mathbf{M}_K^+\mathbf{U}_K^T\mathbf{V}_K^T, \end{aligned} \tag{4.39b}$$

$$\begin{aligned} \Upsilon^{i,p}(\mathbf{M}_K^+) &= \operatorname{trace}\Big(-2\mathbf{Z}_K^-\underbrace{(\mathbf{I} - (\mathbf{Z}_K^-)^T\mathbf{\Phi}_K^{-1}\mathbf{H}_K\mathbf{Z}_K^-)}_{\mathbf{V}_K\mathbf{V}_K^T}(\mathbf{N}_K(\mathbf{Z}_K^-)^T\mathbf{H}_K^T\mathbf{\Phi}_K^{-1})\frac{\mathrm{d}\mathbf{H}_K}{\mathrm{d}\mathbf{q}_K^{i,p}}\Big) \\ &\quad + \operatorname{trace}\Big(\mathbf{\Phi}_K^{-1}\mathbf{H}_K^T\mathbf{Z}_K^-\mathbf{N}_K(\mathbf{Z}_K^-)^T\mathbf{H}_K^T\mathbf{\Phi}_K^{-1}\frac{\mathrm{d}\mathbf{R}_K}{\mathrm{d}\mathbf{q}_K^{i,p}}\Big) \\ &= \operatorname{trace}\Big(-\mathbf{Z}_K^-\mathbf{V}_K\mathbf{U}_K(\mathbf{M}_K^+)^T\mathbf{\Psi}_K\mathbf{\Phi}_K^{-1}\frac{\mathrm{d}\mathbf{H}_K}{\mathrm{d}\mathbf{q}_K^{i,p}}\Big) \\ &\quad + \operatorname{trace}\Big(\mathbf{\Phi}_K^{-1}\mathbf{H}_K^T\mathbf{Z}_K^-\mathbf{N}_K(\mathbf{Z}_K^-)^T\mathbf{H}_K^T\mathbf{\Phi}_K^{-1}\frac{\mathrm{d}\mathbf{R}_K}{\mathrm{d}\mathbf{q}_K^{i,p}}\Big). \end{aligned} \tag{4.39c}$$

The second equality in (4.39b) is derived from performing the simplification shown in the first equality, together with the definition of $\mathbf{N}_K$ in (4.38) substituted. Similarly, the second equality in (4.39c) is derived by performing the simplification shown in the first equality, together with substituting the transpose of (4.38) while noting $\mathbf{N}_K$ is symmetric.

Equation (4.39a) is similar to (4.29), and like in the stochastic EnVO algorithm derivation $\mathbf{M}_K^-$ can be propagated backward to t_{K-1} using the adjoint operator $\mathbf{A}_{K-1,K}^T$, update (with appropriate change of time index) using (4.39b), and pick up an additional $\mathbf{q}_{K-1}^i$ dependency through (4.39c). The process is repeated until $\mathbf{M}_k$ is propagated to $\mathbf{M}_0^+$, where the perturbed cost now has a similar form as (4.16), except $\mathbf{S}$ is replaced with $\mathbf{M}$ and $\mathbf{X}_0^{+\prime}$ is replaced with $\mathbf{Z}_0^{+\prime}$, which is also zero because it is pre-specified. The same vehicle adjoints are defined and the derivations are the same as the stochastic EnVO algorithm, thus it is neglected here. The only difference is that in (4.24) $\mathbf{T}_k^i$ is modified with the new definition of $\Upsilon^{i,p}(\mathbf{M}_k^+)$ defined in (4.39c). At the end of the vehicle adjoint propagation the local gradient definition is identical to (4.23), which again could be extracted to be used by the gradient-based optimization algorithms.

4.4 Different Costs

The EnVO algorithm formulation is sufficiently general to accommodate different sensor vehicles with various dynamics; also the vehicle penalty portion in (4.6) can encompass various vehicle state and control penalties. Furthermore, the different estimation quality measures considered in §2.1.3 can equally be used here; however due to the low rank approximation of $\mathbf{P}$, there will be modifications to these measures. The modifications are discussed in the following.

The forecast quality measure used in (4.6) is the A-optimality criterion and is similar to the one in §3.2 (with the exception of the scaling by 1/2 for convenience); therefore it will be discussed further here. Taking the results from §2.1.3 for the D- and E-optimality criteria and fit them to the square-root framework of $\mathbf{P}$, it could be shown this is equivalent to setting the starting condition for $\mathbf{S}_F$ and

$\mathbf{M}_F$ as followed:

$$\text{D optimality:} \qquad \mathbf{S}_F = \mathbf{P}_F^{-1}\mathbf{X}_F\mathbf{Q} \qquad \mathbf{M}_F = \mathbf{P}_F^{-1}\mathbf{Z}_F, \tag{4.40a}$$

$$\text{E optimality:} \qquad \mathbf{S}_F = \mathbf{r}\mathbf{r}^T\mathbf{X}_F\mathbf{Q} \qquad \mathbf{M}_F = \mathbf{r}\mathbf{r}^T\mathbf{Z}_F. \tag{4.40b}$$

In practice $\mathbf{P}_F$ is rarely full, as it would requires at least $n+1$ ensemble members; hence $\mathbf{P}_F^{-1}$ does not exist in (4.40a). One could make an approximation by using the pseudo-inverse of $\mathbf{P}_F$, $\mathbf{P}_F^{\#}$; thus $\mathbf{M}_F$ in (4.40a) becomes

$$\mathbf{M}_F = (\mathbf{Z}_F\mathbf{Z}_F^T)^{\#}\mathbf{Z}_F = (\mathbf{Z}_F^T)^{\#}\mathbf{Z}_F^{\#}\mathbf{Z}_F = (\mathbf{Z}_F^T)^{\#}\mathbf{Z}_F^T(\mathbf{Z}_F^T)^{\#} = (\mathbf{Z}_F^T)^{\#}. \tag{4.41}$$

Because $\mathbf{Z}_F$ is $\mathbf{X}_F\mathbf{Q}$ scaled by $\sqrt{N-1}$, for the stochastic EnVO algorithm $\mathbf{S}_F$ is simply $\mathbf{M}_F/\sqrt{N-1}$.

Since $\mathbf{P}_F$ is constructed from the outer-product of $\mathbf{Z}_F$, one could show that $\mathbf{r}$ is simply the left singular vector associated with the largest singular value σ_{max} of $\mathbf{Z}_F$. Therefore if the reduced Singular Value Decomposition of $\mathbf{Z}_F$ is

$$\mathbf{Z}_F = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \qquad \mathbf{U}^T\mathbf{U} = \mathbf{I}, \quad \mathbf{V}^T\mathbf{V} = \mathbf{I} \tag{4.42}$$

then $\mathbf{M}_F$ in (4.40b) becomes

$$\mathbf{M}_F = \mathbf{r}\mathbf{r}^T\mathbf{Z}_F = \mathbf{r}\mathbf{r}^T\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \mathbf{r}\sigma_{max}\mathbf{v}^T, \tag{4.43}$$

where $\mathbf{v}$ is the corresponding right singular vector of σ_{max}. This corresponds to setting the ith ensemble adjoint column to $\mathbf{r}$ scaled by $\sigma_{max}\mathbf{v}^i$, where $\mathbf{v}^i$ is the i-th element of $\mathbf{v}$. Like before, $\mathbf{S}_F = \mathbf{M}_F/\sqrt{N-1}$.

4.5 Covariance Localization and Inflation

For highly nonlinear systems, the EnKF significantly outperform traditional KF or Extended Kalman Filter (EKF) even with limited ensemble members. However for large systems such as oceanographic or atmospheric models, the ensemble size is typically too small to be statically representative of the system. The undersampled ensemble typically lead to covariance underestimation, filter divergence, and long-range spurious correlations in the covariance.

Filter divergence typically goes hand-in-hand with covariance underestimation. During each measurement update the ensemble covariance decreases in light of new information; however, an under-sampled ensemble would tend to underestimate the covariance due to insufficient ensemble size that represent the statistics faithfully. This over-confidence decreases the effect of new measurements on the covariance during measurement updates, thus enforcing the over-confidence further. Eventually the ensemble covariance collapses sufficiently that the filter ignores the measurements altogether, creating filter divergence. A simple ad-hoc method called covariance inflation (Anderson & Anderson, 1999) is used to mitigate this problem by artificially inflate the covariance during each measurement update. Anderson (2007) introduces an adaptive method to tune the inflation factor at run-time to achieve better filter performance.

Long-range spurious correlation in the ensemble covariance could perhaps best understood by noting the ensemble covariance $\mathbf{P}$ is severely rank deficient from the outer-product of the ensemble perturbations. Therefore there are elements in $\mathbf{P}$ that represents the correlation between states physically far apart, which may be argued on physical grounds to have zero correlation, that are nonzero. Covariance localization (Hamill *et al.*, 2001) is an ad-hoc method addressing this issue. How covariance inflation and localization can be incorporated into the EnVO algorithm formulation is investigated here.

4.5.1 Covariance Inflation

In covariance inflation, the prior covariance is inflated by scaling with a parameter greater than 1. So for the stochastic EnKF this translate to inflating the ensembles about the ensemble mean:

$$\mathbf{X}^-_{inflate} = \alpha(\mathbf{X}^- - \mathbf{X}^-\mathbf{1}_N) + \mathbf{X}^-\mathbf{1}_N = \mathbf{X}^-(\alpha\mathbf{I} + (1-\alpha)\mathbf{1}_N), \tag{4.44}$$

and for the ESRF this simply means $\mathbf{Z}^-_{inflate} = \alpha\mathbf{Z}^-$. Typically α is chosen to be slightly greater than 1.0 (e.g. $\alpha = 1.01$, or 1 percent inflation).

The covariance inflation could be viewed as another discontinuity in the ensemble forecast, where the ensemble forecast now looks like: propagate $\Rightarrow$ inflate

$\Rightarrow$ update. Accordingly since the state adjoint ensemble propagate backward in time, the state adjoint propagation now looks like: adjoint update $\Rightarrow$ adjoint of inflation $\Rightarrow$ adjoint propagation. From how covariance inflation is defined for both the EnKF and ESRF, it's clear that covariance inflation is a linear operation and self-adjoint. Therefore during the stochastic EnVO adjoint inflation phase:

$$\mathbf{S}^{-}_{inflate} = \mathbf{S}^{-}(\alpha\mathbf{I} + (1-\alpha)\mathbf{1}_N), \tag{4.45}$$

and during the deterministic EnVO adjoint inflation phase $\mathbf{M}^{-}_{inflate} = \alpha\mathbf{M}^{-}$.

4.5.2 Localization

Houtekamer & M. (2001) proposed applying Schur (element-wise) product to the ensemble covariance so the stochastic EnKF gain $\mathbf{K}$ in (4.9) now defined as

$$\mathbf{K} = \rho \bullet \mathbf{P}\mathbf{H}^T\big(\mathbf{H}(\rho \bullet \mathbf{P})\mathbf{H}^T + \mathbf{R}\big)^{-1}, \tag{4.46}$$

where $\bullet$ denotes Schur product and ρ is a matrix of the same size as $\mathbf{P}$ with element values defined by a correlation function with maximum of 1.0 and minimum of 0. To remove long-range spurious correlation between states, typically this correlation function determine the proper scaling based on the Euclidean distance between the said states. When the distance is zero (state is correlating with itself), the scaling is 1.0; as the distance increases, the scaling diminishes to 0. A popular covariance localization scaling function is discussed in Gaspari & Cohn (1999), which approximates an Gaussian function but with compact support.

It is not practical to scale the element of $\mathbf{P}$ due to its size. For computational feasibility, (4.46) is approximated by

$$\mathbf{K} \approx \rho_s \bullet (\mathbf{P}\mathbf{H}^T)\big(\rho_m \bullet (\mathbf{H}\mathbf{P}\mathbf{H}^T) + \mathbf{R}\big)^{-1}, \tag{4.47}$$

where elements of ρ_s are functions of distances between state and measurement location, and ρ_m the distances between measurement locations. An added side-effect of covariance localization is that the effective rank of $\mathbf{P}$ increases, simulating an increase of ensemble members and helps mitigating covariance underestimation and filter divergence.

Note inherently the idea behind covariance localization is to scale $\mathbf{P}$ element-wise to remove long-range spurious correlation. This idea does not directly translate to ESRF because $\mathbf{P}$ is implicitly represented through $\mathbf{Z}$. However one could achieve a similar localization effect by sequential processing of batches of measurements as described in Houtekamer & M. (2001). Such method processes groups of local measurements that are separated by a critical distance $2r_1$ as defined in Gaspari & Cohn (1999). The sequential processing of measurements implicitly suppresses the long-range spurious correlations by assuming the correlations between these groups of measurements are zero. Petrie & Dance (2010) propose a covariance localization method for the ESRF by assuming ρ can be approximated as $\rho = \hat{\rho}\hat{\rho}^T$, and through more approximations incorporate $\hat{\rho}$ into the ETKF measurement update; however numerical experiment shows the method has detrimental affect on the covariance. The deterministic EnVO algorithm could easily account for localization in the ESRF if sequential batch measurement processing is used. Since this type of measurement processing is a series of measurement updates, the deterministic EnVO algorithm can be modified to perform a series of state adjoint updates; after all state adjoint updates are serially processed at a target time, the vehicle adjoint update is computed.

The remaining analysis derives the new stochastic EnVO algorithm taking covariance localization into account. Substitute (4.46) into (4.13) and perform perturbation analysis, the new perturbed EnKF covariance update equation is very similar to (4.14), except now

$$\mathbf{\Psi}_k = \rho \bullet \mathbf{P}_k \mathbf{H}_k^T, \tag{4.48a}$$

$$\mathbf{\Phi}_k^{-1} = \mathbf{H}_k(\rho \bullet \mathbf{P}_k)\mathbf{H}_k^T + \mathbf{R}_k, \tag{4.48b}$$

$$\mathbf{\Psi}'_k = \rho \bullet \mathbf{P}'_k \mathbf{H}_k^T + \rho \bullet \mathbf{P}_k (\mathbf{H}'_k)^T, \tag{4.48c}$$

$$(\mathbf{\Phi}_k^{-1})' = -\mathbf{\Phi}_k^{-1}(\mathbf{H}'_k \mathbf{P}_k \mathbf{H}_k^T + \mathbf{H}_k(\rho \bullet \mathbf{P}'_k)\mathbf{H}_k^T + \mathbf{H}_k \mathbf{P}_k (\mathbf{H}'_k)^T + \mathbf{R}'_k)\mathbf{\Phi}_k^{-1}. \tag{4.48d}$$

Applying these new changed into (4.14a), and leverage the trace identity and the property $\text{trace}(\mathbf{A}(\rho \bullet \mathbf{B})\mathbf{C}) = \text{trace}(\rho^T \bullet (\mathbf{CA})\mathbf{B})$ (see Appendix B), the $\mathbf{X}_k^-{}'$ terms

are gathered. The new adjoint update equation becomes

$$\begin{aligned}\mathbf{S}_k^- = \mathbf{S}_k^+ &+ \mathbf{H}_k^T\mathbf{\Phi}_k^{-1}\mathbf{\Psi}_k^T\mathbf{S}_k^+(\mathbf{1}_N - \mathbf{I}) \\ &+ \rho \bullet (\mathbf{S}_k^+\mathbf{\Theta}_k^T\mathbf{\Phi}_k^{-1}\mathbf{H}_k)\mathbf{X}_k^-\mathbf{Q} + \rho \bullet (\mathbf{H}_k^T\mathbf{\Phi}_k^{-1}\mathbf{\Theta}_k(\mathbf{S}_k^+)^T)\mathbf{X}_k^-\mathbf{Q} \\ &- \rho \bullet (\mathbf{H}_k^T\mathbf{\Phi}_k^{-1}\mathbf{\Psi}_k^T\mathbf{S}_k^+\mathbf{\Theta}_k^T\mathbf{\Phi}_k^{-1}\mathbf{H}_k)\mathbf{X}_k^-\mathbf{Q} \\ &- \rho \bullet (\mathbf{H}_k^T\mathbf{\Phi}_k^{-1}\mathbf{\Theta}_k(\mathbf{S}_k^+)^T\mathbf{\Psi}_k\mathbf{\Phi}_k^{-1}\mathbf{H}_k)\mathbf{X}_k^-\mathbf{Q}\end{aligned} \tag{4.49a}$$

$$\begin{aligned}\Upsilon^{i,p}(\mathbf{S}_k^+) = \mathrm{trace}\Big(&\rho \bullet \mathbf{P}_k^-\mathbf{S}_k^+\mathbf{\Theta}_k^T\mathbf{\Phi}_k^{-1}\frac{d\mathbf{H}_k}{d\mathbf{q}_k^{i,p}} + \mathbf{X}_k^+(\mathbf{1}_N - \mathbf{I})(\mathbf{S}_k^+)^T\mathbf{\Psi}_k\mathbf{\Phi}_k^{-1}\frac{d\mathbf{H}_k}{d\mathbf{q}_k^{i,p}} \\ &- \rho \bullet \mathbf{P}_k^-\mathbf{H}_k^T\mathbf{\Phi}_k^{-1}\mathbf{\Psi}_k^T\mathbf{S}_k^+\mathbf{\Theta}_k^T\mathbf{\Phi}_k^{-1}\frac{d\mathbf{H}_k}{d\mathbf{q}_k^{i,p}} \\ &- \rho \bullet \mathbf{P}_k^-\mathbf{H}_k^T\mathbf{\Phi}_k^{-1}\mathbf{\Theta}_k(\mathbf{S}_k^+)^T\mathbf{\Psi}_k\mathbf{\Phi}_k^{-1}\frac{d\mathbf{H}_k}{d\mathbf{q}_k^{i,p}} \\ &- \mathbf{\Phi}_k\mathbf{\Theta}_k(\mathbf{S}_k^+)^T\mathbf{\Psi}_k\mathbf{\Phi}_k^{-1}\frac{d\mathbf{R}_k}{d\mathbf{q}_k^{i,p}}\Big).\end{aligned} \tag{4.49b}$$

In (4.49) because correlation between any two states are mutually the same, $\rho^T = \rho$. It's clear if localization is not implemented, (4.15b), it is consistent with the non-localized version of the stochastic EnVO algorithm.

Similarly, substituting (4.47) into (4.13) and perform perturbation analysis,

one would have

$$\boldsymbol{\Psi}_k = \rho_s \bullet (\mathbf{P}_k \mathbf{H}_k^T), \tag{4.50a}$$

$$\boldsymbol{\Phi}_k^{-1} = \rho_m \bullet (\mathbf{H}_k \mathbf{P}_k \mathbf{H}_k^T) + \mathbf{R}_k, \tag{4.50b}$$

$$\boldsymbol{\Psi}_k' = \rho_s \bullet (\mathbf{P}_k' \mathbf{H}_k^T + \mathbf{P}_k (\mathbf{H}_k')^T), \tag{4.50c}$$

$$(\boldsymbol{\Phi}_k^{-1})' = -\boldsymbol{\Phi}_k^{-1}(\rho_m \bullet (\mathbf{H}_k' \mathbf{P}_k \mathbf{H}_k^T + \mathbf{H}_k \mathbf{P}_k' \mathbf{H}_k^T + \mathbf{H}_k \mathbf{P}_k (\mathbf{H}_k')^T) + \mathbf{R}_k') \boldsymbol{\Phi}_k^{-1}, \tag{4.50d}$$

$$\begin{aligned} \mathbf{S}_k^- = \mathbf{S}_k^+ &+ \mathbf{H}_k^T \boldsymbol{\Phi}_k^{-1} \boldsymbol{\Psi}_k^T \mathbf{S}_k^+ (\mathbf{1}_N - \mathbf{I}) \\ &+ \rho_s \bullet (\mathbf{S}_k^+ \boldsymbol{\Theta}_k^T \boldsymbol{\Phi}_k^{-1}) \mathbf{H}_k \mathbf{X}_k^- \mathbf{Q} + \mathbf{H}_k^T \rho_s^T \bullet (\boldsymbol{\Phi}_k^{-1} \boldsymbol{\Theta}_k (\mathbf{S}_k^+)^T) \mathbf{X}_k^- \mathbf{Q} \\ &- \mathbf{H}_k^T \rho_m \bullet (\boldsymbol{\Phi}_k^{-1} \boldsymbol{\Psi}_k^T \mathbf{S}_k^+ \boldsymbol{\Theta}_k^T \boldsymbol{\Phi}_k^{-1}) \mathbf{H}_k \mathbf{X}_k^- \mathbf{Q} \\ &- \mathbf{H}_k^T \rho_m \bullet (\boldsymbol{\Phi}_k^{-1} \boldsymbol{\Theta}_k (\mathbf{S}_k^+)^T \boldsymbol{\Psi}_k \boldsymbol{\Phi}_k^{-1}) \mathbf{H}_k \mathbf{X}_k^- \mathbf{Q} \end{aligned} \tag{4.50e}$$

$$\begin{aligned} \Upsilon^{i,p}(\mathbf{S}_k^+) = \operatorname{trace}\Big(&\mathbf{P}_k^- \rho_s \bullet (\mathbf{S}_k^+ \boldsymbol{\Theta}_k^T \boldsymbol{\Phi}_k^{-1}) \frac{d\mathbf{H}_k}{d\mathbf{q}_k^{i,p}} + \mathbf{X}_k^+ (\mathbf{1}_N - \mathbf{I}) (\mathbf{S}_k^+)^T \boldsymbol{\Psi}_k \boldsymbol{\Phi}_k^{-1} \frac{d\mathbf{H}_k}{d\mathbf{q}_k^{i,p}} \\ &- \mathbf{P}_k^- \mathbf{H}_k^T \rho_m \bullet (\boldsymbol{\Phi}_k^{-1} \boldsymbol{\Psi}_k^T \mathbf{S}_k^+ \boldsymbol{\Theta}_k^T \boldsymbol{\Phi}_k^{-1}) \frac{d\mathbf{H}_k}{d\mathbf{q}_k^{i,p}} \\ &- \mathbf{P}_k^- \mathbf{H}_k^T \rho_m \bullet (\boldsymbol{\Phi}_k^{-1} \boldsymbol{\Theta}_k (\mathbf{S}_k^+)^T \boldsymbol{\Psi}_k \boldsymbol{\Phi}_k^{-1}) \frac{d\mathbf{H}_k}{d\mathbf{q}_k^{i,p}} \\ &- \boldsymbol{\Phi}_k \boldsymbol{\Theta}_k (\mathbf{S}_k^+)^T \boldsymbol{\Psi}_k \boldsymbol{\Phi}_k^{-1} \frac{d\mathbf{R}_k}{d\mathbf{q}_k^{i,p}} \Big). \end{aligned} \tag{4.50f}$$

In (4.50) because correlation between any two measurement locations are mutually the same, $\rho_m^T = \rho$; however note ρ_s is not square, thus $\rho_s^T \neq \rho_s$. Once again, (4.50) is consistent with the non-localized stochastic EnVO algorithm.

4.6 Numerical Experiment

Setup

The stochastic EnVO algorithm is tested on a 1D Kuramoto-Sivashinsky (KS) equation. The KS equation is one of the simplest equation for numerical simulation of turbulence. It is commonly used for describing wave processes in active and dissipative environments. In particular, the KS equations used here is

in the form

$$\frac{\partial f}{\partial t} + f\frac{\partial f}{\partial x} + \frac{\partial^2 f}{\partial x^2} + \frac{\partial^4 f}{\partial x} = 0 \tag{4.51}$$

over the domain of length L with periodic boundary conditions. While the second spacial derivative excites the system at low frequencies and the fourth spacial derivative dissipate energy at high frequencies, the nonlinear term scatters energy across the frequencies. Thus for sufficiently large initial condition and L, (4.51) neither blows up nor settle to an equilibrium; in this case, the system approaches to a chaotic attractor, like the Navier-Stokes equation.

$f(x,t)$ is discretized over $L = 50$ with 128 evenly-spaced grid points ($n = 128$) to form the state variable $\mathbf{x}$, and the KS equation is propagated spectrally, so that $f(x,t)$ can be evaluated at non-grid points by spectral interpolation. 5 "sensors" having damped point-mass vehicle dynamics with control input $u^i(t)$

$$\frac{\mathrm{d}^2 q^i}{\mathrm{d}t^2} + \mu\frac{\mathrm{d}q^i}{\mathrm{d}t} + q^i = u^i(t), \qquad \mu = 0.1, \tag{4.52}$$

are initially distributed evenly along L at $x = \{0, 10, 20, 30, 40\}$. Periodic boundary condition is also enforced so that sensor warps around when q^i exceeds 50 or below 0. The sensors measures the local value of f at q^i every 0.2 time units, while both (4.51) and (4.52) propagates with a low-storage, third-order Runge Kutta scheme (RKW3) using time step of 0.01.

At time t_0 there is an estimate of $f(x,t_0)$ from an ensemble of 128 ensemble estimates. Two scenarios are chosen for testing, one for a short verification time ($t_F = 1$) and the other a long verification time ($t_F = 5$); as a point of reference, the error doubling time is around 9 time units. For simplicity the verification time coincides with the last target time, where $t_K = t_F$. The cost J is defined as the sum of the A-optimality criterion and a quadratic vehicle control penalty:

$$J = \frac{1}{2}\mathrm{trace}(\mathbf{P}_F) + \frac{1}{2}\sum_{i=1}^{5}\int_{t_0}^{t_F} u^i(t)^T \mathbf{Q_u} u^i(t)\,\mathrm{d}t, \tag{4.53}$$

where $\mathbf{Q_u}$ is a diagonal matrix with elements set to 0.01.

The EnVO algorithm performance is compared against two other ad-hoc sampling methods. The first method considers the initially-evenly-spaced sensors to remain stationary; this is to simulate a stationary uniform coverage sampling

approach. In the second method, constant $u^i(t)$ is applied to each vehicle such that in the $t_F = 1$ case, the vehicles reaches the initial position of their neighboring vehicle immediately to the right, and in the $t_F = 5$ case, the vehicles would have made a round-trip about the domain and arrive back at its initial starting position. This control strategy is chosen to simulate a moving uniform coverage sampling approach. Clearly there are pros and cons to each method. Because vehicles are not moving in the stationary method, there would be zero vehicle control penalty, but forecast quality suffers; while in the constant control method, vehicle movement should improve forecast quality at the expense of high vehicle control penalty.

A time-averaged framework is used to quantify the average performance of all three methods. At each verification time the next future target site sequences are computed for each method, with evenly spaced and at rest initial vehicle state. Measurements of $f(x,t)$ are taken at these future target sites and assimilated by three independent stochastic EnKFs, each dedicated for a sampling method. Each stochastic EnKF assimilates measurements until the verification time is reached, at which the estimation error statistics against the truth is calculated. The updated ensembles at verification time are used as initial ensembles for the next run, and all vehicle states are reset to as before. This way, the ensembles are propagated and updated throughout the entire attractor, and the error statistics collected would give a average performance independent of initial conditions. The experiments are performed with 1,000 spin-up runs, and time averaged statistics is gather for the subsequent 20,000 runs.

Results

The averaged statistics for 128 ensemble members are recorded in Table 4.1 and Table 4.2. Not surprisingly, the constant control method performs the best amongst the three in terms of estimation performance, because it has the best coverage of the domain; however such aggressive action is penalized in terms of substantially higher control cost. The stationary method has the worst estimation performance, however it doesn't incur any control penalty. As the time interval gets bigger, the vehicles are allowed to have a smaller acceleration to achieve uniform

Table 4.1: Time averaged result for $t_F = 1$, with ensemble size $N = 128$.

method	estimation error	estimation error covariance	control cost
constant control	53.2e-3	21.0	10.5
stationary	1.5	346.0	0
EnVO	306.4e-3	86.9	557.6e-3

Table 4.2: Time averaged result for $t_F = 5$, with ensemble size $N = 128$.

method	estimation error	estimation error covariance	control cost
constant control	269.1e-6	112.6e-3	2.7
stationary	1.6	348.2	0
EnVO	1.0e-3	449.6e-3	44.6e-3

domain coverage, which translates to smaller control effort. This is evident from the tables where the control cost for $t_F = 1$ is greater than that of $t_F = 5$, despite longer time interval to accumulate penalty; this implies much of the control cost is incurred from the initial vehicle acceleration. The EnVO algorithm is able to strike a good balance between control effort and estimation performance. In both verification time scenarios, the EnVO algorithms is able to improve estimation performance (compared to the stationary method) while maintaining small control cost (compared to the constant control method).

The previous experiment is performed with ensemble size comparable to the state size. However in practice this rarely is the case, therefore the same experiment is repeated with only 8 ensemble members to simulate real-life situations where the ensemble size is much smaller than the state size. The results are illustrated in Table 4.3 and Table 4.4. Comparing these results with the one in Table 4.1 and Table 4.2, it's clear that while there are little changes to the constant control and stationary method (both with estimation error and covariance increased, as expected from a smaller ensemble size), the estimation error and covariance increase significantly while control cost decreased for the EnVO algorithm. This can be explained by the small ensemble size, where because the ensemble size is smaller, the ensemble covariance tend to misrepresent and underestimate the actual uncertainty; as the result the EnVO algorithm dial down the control effort in reaction to the smaller ensemble covariance and produces target sites that minimizes the misrepresented uncertainty. Despite this, the EnVO algorithm still demonstrates

Table 4.3: Time averaged result for $t_F = 1$, with ensemble size $N = 8$.

method	estimation error	estimation error covariance	control cost
constant control	54.5e-3	23.3	10.5
stationary	1.6	393.0	0
EnVO	613.3e-3	172.2	413.9e-3

Table 4.4: Time averaged result for $t_F = 5$, with ensemble size $N = 8$.

method	estimation error	estimation error covariance	control cost
constant control	269.7e-6	120.4e-3	2.7
stationary	1.7	395.0	0
EnVO	2.1e-3	731.0e-3	39.8e-3

the ability to balance estimation performance and control effort spent.

Different experiments with decreasing vehicle control penalty $\mathbf{Q_u}$ have also been performed, the results are in Table 4.5. There are several notable features. First, as expected the estimation error and covariance decreases as the control penalty is decrease, since now the vehicles could move more aggressively. Second, as $\mathbf{Q_u}$ is scaled by a certain factor, the control cost is not scaled by the same factor. Take $\mathbf{Q_u} = 0.01$ and $\mathbf{Q_u} = 0.001$ for example, a factor of 10 decrease does not translate to the same decrease in the control cost; there is only about a factor of 5 decrease in control cost. This suggests the control trajectories for $\mathbf{Q_u} = 0.001$ are more aggressive than the ones in $\mathbf{Q_u} = 0.01$. Interestingly, at $\mathbf{Q_u} = 0.0001$ it seems to hit a "sweet spot" where the estimation error and covariance drops significantly; this also highlight one potential operational issue with the EnVO algorithm, that it is sensitive to the penalty weighting selection. However it is expected as the EnVO algorithm is used more often, data can be collected based on past weight selection to build up a data base where one could look up what a "good" weight to choose in different situations. Finally at $\mathbf{Q_u} = 0.00001$ the control cost is equivalent to the constant control case (if this new weight is used to evaluate the control cost for constant control method, it could be 10e-3); however the EnVO algorithm produces better estimation performance given the same control effort.

Table 4.5: EnVO performance as function of $\mathbf{Q_u}$ for time averaged result for $t_F = 1$, with ensemble size $N = 8$.

$\mathbf{Q_u}$	estimation error	estimation variance	control cost
0.01	613.3-3	172.2	413.9e-3
0.005	548.8e-3	146.3	297.2e-3
0.001	339.2e-3	93.4	89.4e-3
0.0005	190.8e-3	49.8	63.0e-3
0.0001	1.9e-3	633.1e-3	23.5e-3
0.00001	431.1e-6	187.5e-3	9.2e-3

4.7 Summary/Discussion

An numerically efficient version of the DAO algorithm in Chapter 3, called EnVO, is derived in this chapter. There are two flavors to the EnVO algorithm, one is the stochastic EnVO algorithm designed to work with the stochastic EnKF, and the other is the deterministic EnVO algorithm for the deterministic ESRF.

The stochastic EnVO algorithm is tested with a 1D Kuramoto-Sivashinksy equation for a short verification time ($t_F = 1$) and a medium verification time ($t_F = 5$), with two different ensemble sizes N=128 and N=8. Two other ad-hoc sampling methods are used to serve as reference to the EnVO performance. Time-averaged statistics are gathered for all three methods and the results suggest EnVO is able to balance reducing estimation error and increase control effort. The results also highlights two potential issues in the EnVO algorithm. The first issue is that because the EnVO algorithm selects future target sites based on the ensemble forecast, it is sensitive to the ensemble size. The average estimation error doubled for the experiment with 8 ensemble members. The second issue is the performance sensitivity of the the EnVO algorithm to the weight selection in the cost; however given the same control effort the EnVO algorithms is able to out-perform the constant control method.

In theory, if the vehicle model is exact, the optimized control trajectory $\mathbf{u}^i(t)^*$ can be sent to the sensor vehicles to be used directly. However vehicle models are typically simplified versions of its real-life counterpart; thus realistically, $\mathbf{u}^i(t)^*$ is applied to (4.2) to produce $\mathbf{q}^i(t)^*$. The target site sequence for the i-th vehicle is extracted from the positions information contained in $\mathbf{q}^i(t)^*$, and is sent to the

vehicle. The main difference between the target sites generated by the EnVO algorithm and existing AO algorithms is that the EnVO algorithm target sites are extracted from vehicle state trajectories that satisfy (conservative approximation of) vehicle dynamic constraints; therefore the probability these target sites are feasible when implemented by the sensor vehicle onboard autopilot is high.

Although the EnVO algorithm is designed to reduce the forecast error effect on the adjoint propagations by propagating an ensemble of adjoints, nevertheless due to modeling errors and insufficient ensemble members, the effect is not completely removed. It is likely that although the beginning of the target site sequences may be valid, the latter ones may not. Therefore, one integral part of the EnVO algorithm is that only the beginning of the target site sequences are used before the algorithm is performed again. This serves two purposes. First, the unused target site sequences can be used as backups should the communication fail between the central computer and the vehicles; and second, the control trajectories for the remaining target site sequences can be used to initialize part of the control trajectories for the next EnVO optimization, which potentially improve optimization speed as part of the optimal solution from previous optimization is carried over. This scheme is called Receding Horizon Model Predictive Control.

Furthermore the EnVO algorithm could be used much like current AO methods. If the underlying system is essentially static compared to the vehicle dynamics, one could instead optimize the sensor positions $\mathbf{q}_k^i$. In this case, since the vehicle dynamic is no longer considered, the EnVO algorithm is reduced to only perturbing the EnKF/ESRF and propagating the state adjoint. Thus by definition, the local gradient of $\nabla_{\mathbf{q}_k^i} J$ is simply $\mathbf{T}_k^i$.

One last EnVO algorithm application is determining the forecast sensitivity to the initial ensemble. The initial ensemble perturbation heavily influence the ensemble forecast; ideally one would choose ensemble perturbations in the principle directions of uncertainty growth. Toth & Kalnay (1993); Houtekamer (1995) discuss methods to generate such perturbation. In (4.16) it is stated $\mathbf{X}_0^{+\prime}$ is zero because the initial ensembles are given; however if $\mathbf{X}_0^{+\prime}$ is allowed to vary, then by definition $\mathbf{S}_0^+$ is the ensemble sensitivity of J to the initial ensembles. Specifically,

$\mathbf{s}_0^{j+}$ is the sensitivity of J to $\mathbf{x}_0^{j+}$ (and similarly $\mathbf{m}_0^{j+}$ is the sensitivity of J to $\mathbf{z}_0^{j+}$ in the deterministic EnVO algorithm). While the idea of computing the sensitivity of J with respect to the initial condition is not new, the EnVO algorithm provides a mean to calculate the sensitivity while taking into account of the future measurement updates; this aspect is not available in current optimal perturbation generation algorithms. Furthermore, the ability to compute the sensitivities to the initial ensemble also open up the possibility to examine which initial ensemble members J is not sensitive to, remove the said ensemble members, and repopulate them in the neighborhood of the sensitive ones. Performing data assimilation with this new initial ensemble with past data may yield a better initial ensemble at current time. However currently it is not certain how to repopulate the ensembles such that the ensemble statistics are preserved.

Chapter 5

Summary

Toward solving the environmental Adaptive Observation (AO) problem considered in Chapter 1, three incremental steps are taken. In the first step, the AO problem is simplified to finding optimal static sensor placement to minimize an infinite-time system estimation uncertainty. The same theory is shown to be able to applied for finding optimal static actuator placements. In the second step, the full AO problem is addressed by considering the the evolution of the system and the vehicle, within a finite time interval. The objective is to find the optimal vehicle waypoints, subjected to vehicle dynamical constraints, to improve the estimation/forecast quality at a later time. The resulting theory is the Dynamic Adaptive Observation (DAO) algorithm. Theoretically, the DAO algorithm is sufficient in solving the environmental AO problem considered in this thesis; however due to real-life computational constraints, many approximation, including some strict assumptions on the system, are required to have the DAO algorithm functioning in a practical manner. To this end, in the third step the DAO algorithm is modified to incorporating the more computationally friendly Ensemble Kalman Filter (EnKF). The modified algorithm is the Ensemble Variational Observation (EnVO) algorithm, and a stochastic and a deterministic version is derived based on the stochastic and deterministic version of the EnKF.

Appendix A

Cramér-Rao Lower Bound

The appropriate interpretation of equation (2.2) is critical to understanding the differences in cost functions (2.3b) and (2.3a). The following theorem clarifies the interpretation and is reprinted here for convenience.

Theorem A.1. [Goodwin & Payne 1977, Theorem 1.3.1] *(The Cramér-Rao Inequality) Let* $\{P_\theta : \theta \in \Theta\}$ *be a family of distributions on a sample space* Ω, $\Theta \subset \mathbb{R}^p$, *and suppose that, for each* Θ, P_θ *is defined by a density* $p_{Y|\Theta}(\cdot|\theta)$. *Then subject to certain regularity conditions, the covariance of any unbiased estimator* $g(Y)$ *of* Θ *satisfies the inequality*

$$\operatorname{cov}(g) \geq \mathbf{I}_F^{-1} \tag{A.1}$$

where

$$\operatorname{cov}(g) \triangleq \mathrm{E}_{Y|\Theta}\{(\mathrm{g}(Y) - \theta)(\mathrm{g}(Y) - \theta)^H\} \tag{A.2}$$

and where $\mathbf{I}_F$, *known as Fisher's Information Matrix (FIM), is defined by*

$$\mathbf{I}_F \triangleq \mathrm{E}_{Y|\Theta}\left\{\left(\frac{\partial \ln \mathrm{p}(Y|\theta)}{\partial \theta}\right)^H \left(\frac{\partial \ln \mathrm{p}(Y|\theta)}{\partial \theta}\right)\right\}. \tag{A.3}$$

Proof. Since $\mathrm{g}(Y)$ is an unbiased estimator of θ, we have

$$\mathrm{E}_{Y|\Theta}\{\mathrm{g}(Y)\} = \theta, \tag{A.4}$$

i.e.,

$$\int_\Omega \mathrm{g}(y)p(y|\theta)\,dy = \theta, \quad \text{so} \quad \frac{\partial}{\partial \theta}\int_\Omega \mathrm{g}(y)p(y|\theta)\,dy = I \tag{A.5}$$

Assuming sufficient regularity to allow differentiation under the integral sign,

$$\int_{\Omega} \mathrm{g}(y) \frac{\partial p(y|\theta)}{\partial \theta}\, dy = I, \quad \text{so} \quad \int_{\Omega} \mathrm{g}(y) \frac{\partial \ln p(y|\theta)}{\partial \theta} p(y|\theta)\, dy = I, \tag{A.6}$$

i.e.,

$$\mathrm{E}_{Y|\Theta}\left\{ \mathrm{g}(Y) \frac{\partial \ln p(Y|\theta)}{\partial \theta} \right\} = I. \tag{A.7}$$

Also

$$\begin{aligned} \mathrm{E}_{Y|\Theta}\left\{ \frac{\partial \ln p(Y|\theta)}{\partial \theta} \right\} &= \int_{\Omega} \frac{\partial \ln p(y|\theta)}{\partial \theta} p(y|\theta)\, dy = \quad \int_{\Omega} \frac{\partial p(y|\theta)}{\partial \theta}\, dy \\ &= \frac{\partial}{\partial \theta} \int_{\Omega} p(y|\theta)\, dy = \frac{\partial}{\partial \theta}(1) = 0. \end{aligned} \tag{A.8}$$

Thus, using equations (A.2), (A.3), (A.4), (A.7), and (A.8), the covariance of $\mathrm{g}(Y)$ and $\partial \ln p(y|\theta)/\partial\theta$ can be written as

$$\mathrm{E}_{Y|\Theta}\left\{ \begin{bmatrix} (\mathrm{g}(Y) - \theta) \\ \left(\dfrac{\partial \ln p(Y|\theta)}{\partial \theta} \right)^H \end{bmatrix} \left[(\mathrm{g}(Y) - \theta)^H \left(\frac{\partial \ln p(Y|\theta)}{\partial \theta} \right) \right] \right\} = \begin{bmatrix} \mathrm{cov}(g) & I \\ I & \mathbf{I}_F \end{bmatrix}. \tag{A.9}$$

By the definition of covariance matrices, (A.9) is positive semi-definite; hence

$$\begin{aligned} \begin{bmatrix} I : & -\mathbf{I}_F^{-1} \end{bmatrix} \begin{bmatrix} \mathrm{cov}(g) & I \\ I & \mathbf{I}_F \end{bmatrix} \begin{bmatrix} I \\ -\mathbf{I}_F^{-1} \end{bmatrix} &\geq 0 \\ \mathrm{cov}(g) - \mathbf{I}_F^{-1} &\geq 0. \end{aligned} \tag{A.10}$$

□

Taylor (1979) took great care to express the FIM for the Kalman Filter with continuous time systems and discrete time measurements. Taylor assumes an invertible state transition matrix satisfying the differential equation

$$\dot{\mathbf{\Phi}}(t, t_0) = \mathbf{A}(t)\mathbf{\Phi}(t, t_0), \tag{A.11a}$$

subject to the initial condition $\mathbf{\Phi}(t, t) = \mathbf{I}$ and $\mathbf{\Phi}_{k+1,k} \triangleq \mathbf{\Phi}(t_{k+1}, t_k)$. Provided this definition, Taylor shows the FIM for the Kalman Filter can be rewritten in a recursive form

$$\mathbf{I}_F(t_k) = \mathbf{\Phi}_{k,k-1}^{-T} \mathbf{I}_F(t_{k-1}) \mathbf{\Phi}_{k,k-1}^{-1} + \mathbf{H}^H \mathbf{R}^{-1} \mathbf{H}, \tag{A.12}$$

where $\mathbf{R}_k$ is the measurement noise covariance. Comparing this results with the recursive propagation of the Information Filter (Not to be confused with the FIM. The information filter is the propagation and update of the information matrix, which is defined as the inverse of the Kalman Filter covariance $\mathbf{P}$; see, Anderson & Moore, 1979),

$$\mathbf{P}_{k|k}^{-1} = \left(\mathbf{\Phi}_{k,k-1}\mathbf{P}_{k-1|k-1}\mathbf{\Phi}_{k,k-1}^{H} + \mathbf{W}\right)^{-1} + \mathbf{H}^{H}\mathbf{R}^{-1}\mathbf{H}, \tag{A.13}$$

where $\mathbf{W}$ is the covariance of the additive noise to the state evolution equation, it is clear that in the limit where the state evolution is deterministic ($\mathbf{W} = 0$), the covariance reaches the bound predicted by the Crámer-Rao inequality. A complete proof can be found in Taylor (1979).

Appendix B

Diagonal $\mathbf{P}_k$ and $\mathbf{S}_k$

During the camera example in Chapter 3 it is mentioned that the DAO algorithm can be carried out by only tracking the diagonals of $\mathbf{P}_k$ and $\mathbf{S}_k$ if $\mathbf{P}_0$ and $\mathbf{R}_k^i$ are diagonal, $\mathbf{H}_k^i = \mathbf{I}$, and $\mathbf{A} = \alpha\mathbf{I}$. The detail of the proof will be shown here.

For each camera sensor, $\mathbf{H}_k^i = \mathbf{I}$ and $\mathbf{R}_k^i$ is diagonal; thus the collective $\mathbf{H}_k$ and $\mathbf{R}_k$ are:

$$\mathbf{H}_k = \begin{bmatrix} \mathbf{I} \\ \vdots \\ \mathbf{I} \end{bmatrix}, \qquad \mathbf{R}_k = \begin{bmatrix} \mathbf{R}_k^1 & & 0 \\ & \ddots & \\ 0 & & \mathbf{R}_k^N \end{bmatrix}. \tag{B.1}$$

With $\mathbf{A} = \alpha\mathbf{I}$, the evolution of P_k^+ is simply

$$\mathbf{P}_{k+1}^+ = \alpha^2\mathbf{P}_k^+ - \alpha^4\mathbf{P}_k^+\mathbf{H}_k^T(\alpha^2\mathbf{H}_k\mathbf{P}_k^+\mathbf{H}_k^T + \mathbf{R}_k)^{-1}\mathbf{H}_k\mathbf{P}_k^+. \tag{B.2}$$

Applying the Matrix Inversion Lemma $(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1} = \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{C}\mathbf{A}^{-1}$ to the inverse, where $\mathbf{A} = \mathbf{R}_k$, $\mathbf{B} = \mathbf{H}_k$, $\mathbf{C} = \mathbf{H}_k^T$, and $\mathbf{D} = -(\alpha^2\mathbf{P}_k^+)^{-1}$, it becomes:

$$\begin{aligned} \mathbf{P}_{k+1}^+ = \alpha^2\mathbf{P}_k^+ &- \alpha^4\mathbf{P}_k^+\mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{H}_k\mathbf{P}_k^+ \\ &- \alpha^4\mathbf{P}_k^+\mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{H}_k\left(-(\alpha^2\mathbf{P}_k^+)^{-1} - \mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{H}_k\right)^{-1}\mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{H}_k\mathbf{P}_k^+. \end{aligned} \tag{B.3}$$

By construction $\mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{H}_k = \sum_{i=1}^M (\mathbf{R}_k^i)^{-1}$ is diagonal; thus if $\mathbf{P}_k^+$ is diagonal then $(-(\alpha^2\mathbf{P}_k^+)^{-1} - \mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{H}_k)^{-1}$ is also diagonal; this guarantees $\mathbf{P}_{k+1}^+$ to be diagonal.

By induction $\mathbf{P}_k$ would remain diagonal as long as $\mathbf{P}_0$ is diagonal. Similarly by using the Matrix Inversion Lemma and taking advantage of the structure in $\mathbf{H}_k$ and $\mathbf{R}_k$, it is easy to show the adjoint variable $\mathbf{S}_k$ remains diagonal during the adjoint propagation if the starting condition $\mathbf{S}_K$ is diagonal.

Appendix C

Symmetric $(\mathbf{Z}_k^-)^T\mathbf{M}_k^+\mathbf{U}_k^T\mathbf{V}_k^{-1}$

In §4.3.2, it is shown for $k = K$, $(\mathbf{Z}_k^-)^T\mathbf{M}_k^+\mathbf{U}_k^T\mathbf{V}_k^{-1}$ is symmetric because when re-expressing $\mathbf{M}_K^+$ to contain $\mathbf{Z}_K^-$,

$$(\mathbf{Z}_K^-)^T\mathbf{M}_K^+\mathbf{U}_K^T\mathbf{V}_K^{-1} = (\mathbf{Z}_K^-)^T\mathbf{A}_{K,F}^T\mathbf{T}\mathbf{A}_{K,F}\mathbf{Z}_K^-.$$

However it is not clear that once $\mathbf{M}_K^+$ is updated to $\mathbf{M}_K^-$ and subsequently propagated to $\mathbf{M}_{K-1}^+$, that $(\mathbf{Z}_{K-1}^-)^T\mathbf{M}_{K-1}^+\mathbf{U}_{K-1}^T\mathbf{V}_{K-1}^{-1}$ is also symmetric and thus ensuring $\mathbf{N}_{K-1}$ exist; in general, it is unclear whether the subsequent $(\mathbf{Z}_k^-)^T\mathbf{M}_k^+\mathbf{U}_k^T\mathbf{V}_k^{-1}$ is symmetric.

Starting from $(\mathbf{Z}_k^-)^T\mathbf{M}_k^+\mathbf{U}_k^T\mathbf{V}_k^{-1}$, one may re-express $\mathbf{M}_k^+$ as the adjoint propagation from $\mathbf{M}_{k+1}^-$:

$$(\mathbf{Z}_k^-)^T\mathbf{A}_{k,k+1}^T\mathbf{M}_{k+1}^-\mathbf{U}_k^T\mathbf{V}_k^{-1}.$$

Using (4.39b), the above equation can in turn expressed as

$$(\mathbf{Z}_k^-)^T\mathbf{A}_{k,k+1}^T\underbrace{(\mathbf{I} - \mathbf{H}_{k+1}^T\mathbf{\Phi}_{k+1}^{-1}\mathbf{\Psi}_{k+1}^T)}_{\mathbf{R}_{k+1}}\mathbf{M}_{k+1}^+\mathbf{U}_{k+1}^T\mathbf{V}_{k+1}^T\mathbf{U}_k^T\mathbf{V}_k^{-1},$$

where $\mathbf{R}_{k+1}$ is defined for convenience. Similarly, one may re-express $\mathbf{M}_{k+1}^+$ as the adjoint propagation from $\mathbf{M}_{k+2}^-$, which through (4.39b) can be written in terms of $\mathbf{M}_{k+2}^+$. If one progresses with the substitutions until $\mathbf{M}_K^+$, one would have

$$\begin{aligned}&(\mathbf{Z}_k^-)^T\mathbf{A}_{k,k+1}^T\mathbf{M}_{k+1}^-\mathbf{U}_k^T\mathbf{V}_k^{-1} = \\ &(\mathbf{Z}_k^-)^T\mathbf{A}_{k,k+1}^T\mathbf{R}_{k+1}\mathbf{A}_{k+1,k+2}^T\mathbf{R}_{k+2}\cdots\mathbf{R}_K\mathbf{M}_K^+\mathbf{U}_K^T\mathbf{V}_K^T\mathbf{U}_{K-1}^T\mathbf{V}_{K-1}^T\cdots\mathbf{U}_{k+1}^T\mathbf{V}_{k+1}^T\mathbf{U}_k^T\mathbf{V}_k^{-1}.\end{aligned} \tag{C.1}$$

From (4.37) it is clear that

$$\mathbf{M}_K^+ = \mathbf{A}_{K,F}^T \mathbf{T} \mathbf{A}_{K,F} \mathbf{Z}_K^- \mathbf{V}_K \mathbf{U}_K,$$

thus (C.1) can be written as

$$(\mathbf{Z}_k^-)^T \mathbf{A}_{k,k+1}^T \mathbf{R}_{k+1} \mathbf{A}_{k+1,k+2}^T \mathbf{R}_{k+2} \cdots \mathbf{R}_K \mathbf{A}_{K,F}^T \mathbf{T} \mathbf{A}_{K,F} \times \mathbf{Z}_K^- \underbrace{\mathbf{V}_K \mathbf{U}_K \mathbf{U}_K^T \mathbf{V}_K^T}_{\mathbf{I} - (\mathbf{Z}_K^-)^T \mathbf{H}_K^T \mathbf{\Phi}_K^{-1} \mathbf{H}_K \mathbf{Z}_K^-} \mathbf{U}_{K-1}^T \mathbf{V}_{K-1}^T \cdots \mathbf{U}_{k+1}^T \mathbf{V}_{k+1}^T \mathbf{U}_k^T \mathbf{V}_k^{-1},$$

which by definition $\mathbf{V}_K \mathbf{U}_K \mathbf{U}_K^T \mathbf{V}_K^T = \mathbf{V}_K \mathbf{V}_K^T = \mathbf{I} - (\mathbf{Z}_K^-)^T \mathbf{H}_K^T \mathbf{\Phi}_K^{-1} \mathbf{H}_K \mathbf{Z}_K^-$. The above equation can be slightly modified by pulling $\mathbf{Z}_K^-$ to the right:

$$(\mathbf{Z}_k^-)^T \mathbf{A}_{k,k+1}^T \mathbf{R}_{k+1} \mathbf{A}_{k+1,k+2}^T \mathbf{R}_{k+2} \cdots \mathbf{R}_K \mathbf{A}_{K,F}^T \mathbf{T} \mathbf{A}_{K,F} \times \underbrace{(\mathbf{I} - \mathbf{\Psi}_K \mathbf{\Phi}_K^{-1} \mathbf{H}_K)}_{\mathbf{R}_K^T} \mathbf{Z}_K^- \mathbf{U}_{K-1}^T \mathbf{V}_{K-1}^T \cdots \mathbf{U}_{k+1}^T \mathbf{V}_{k+1}^T \mathbf{U}_k^T \mathbf{V}_k^{-1},$$

where by earlier definition $\mathbf{R}_K^T = \mathbf{I} - \mathbf{\Psi}_K \mathbf{\Phi}_K^{-1} \mathbf{H}_K$. However by (4.26a) $\mathbf{Z}_K^- = \mathbf{A}_{K-1,K} \mathbf{Z}_{K-1}^+$ and by (4.26b) $\mathbf{Z}_{K-1}^+ = \mathbf{Z}_{K-1}^- \mathbf{V}_{K-1} \mathbf{U}_{K-1}$, thus the above equation is re-expressed as

$$(\mathbf{Z}_k^-)^T \mathbf{A}_{k,k+1}^T \mathbf{R}_{k+1} \mathbf{A}_{k+1,k+2}^T \mathbf{R}_{k+2} \cdots \mathbf{R}_K \mathbf{A}_{K,F}^T \mathbf{T} \mathbf{A}_{K,F} \mathbf{R}_K^T \times \mathbf{A}_{K-1,K} \mathbf{Z}_{K-1}^- \underbrace{\mathbf{V}_{K-1} \mathbf{U}_{K-1} \mathbf{U}_{K-1}^T \mathbf{V}_{K-1}^T}_{\mathbf{I} - (\mathbf{Z}_{K-1}^-)^T \mathbf{H}_{K-1}^T \mathbf{\Phi}_{K-1}^{-1} \mathbf{H}_{K-1} \mathbf{Z}_{K-1}^-} \cdots \mathbf{U}_{k+1}^T \mathbf{V}_{k+1}^T \mathbf{U}_k^T \mathbf{V}_k^{-1},$$

which using the similar method above, $\mathbf{Z}_{K-1}^-$ can be pulled to the right, and rewritten into $\mathbf{A}_{K-2,K-1} \mathbf{Z}_{K-2}^- \mathbf{V}_{K-2} \mathbf{U}_{K-2}$. Repeating these steps until $\mathbf{Z}_{k+1}^-$ is pulled to the right, the above equation becomes

$$(\mathbf{Z}_k^-)^T \mathbf{A}_{k,k+1}^T \mathbf{R}_{k+1} \mathbf{A}_{k+1,k+2}^T \mathbf{R}_{k+2} \cdots \mathbf{R}_K \mathbf{A}_{K,F}^T \mathbf{T} \times \mathbf{A}_{K,F} \mathbf{R}_K^T \cdots \mathbf{R}_{k+2}^T \mathbf{A}_{k+1,k+2} \mathbf{R}_{k+1}^T \mathbf{Z}_{k+1}^- \mathbf{U}_k^T \mathbf{V}_k^{-1}. \quad \text{(C.2)}$$

One final substitution $\mathbf{Z}_{k+1}^- = \mathbf{A}_{k,k+1} \mathbf{Z}_k^- \mathbf{V}_k \mathbf{U}_k$, it is clear $\mathbf{V}_k \mathbf{U}_k \mathbf{U}_k^T \mathbf{V}_k^{-1} = \mathbf{I}$. Equation (C.2) becomes

$$(\mathbf{Z}_k^-)^T \mathbf{A}_{k,k+1}^T \mathbf{M}_{k+1}^- \mathbf{U}_k^T \mathbf{V}_k^{-1} = (\mathbf{Z}_k^-)^T \mathbf{A}_{k,k+1}^T \mathbf{R}_{k+1} \mathbf{A}_{k+1,k+2}^T \mathbf{R}_{k+2} \cdots \mathbf{R}_K \mathbf{A}_{K,F}^T \mathbf{T} \times \mathbf{A}_{K,F} \mathbf{R}_K^T \cdots \mathbf{R}_{k+2}^T \mathbf{A}_{k+1,k+2} \mathbf{R}_{k+1}^T \mathbf{A}_{k,k+1} \mathbf{Z}_k^-,$$

which is symmetric and hence completes the proof.

Appendix D

Schur product operation within the trace

The property $\text{trace}(\mathbf{A}(\rho \bullet \mathbf{B})\mathbf{C}) = \text{trace}(\rho^T \bullet (\mathbf{CA})\mathbf{B})$ is derived here. In index notation,

$$\begin{aligned}
\text{trace}(\mathbf{A}(\rho \bullet \mathbf{B})\mathbf{C}) &= \mathbf{A}_{ij}(\rho \bullet \mathbf{B})_{jk}\mathbf{C}_{ki} \\
&= \mathbf{A}_{ij}\rho_{jk}\mathbf{B}_{jk}\mathbf{C}_{ki} \\
&= \rho_{jk}\mathbf{C}_{ki}\mathbf{A}_{ij}\mathbf{B}_{jk} \\
&= \rho^T_{kj}(\mathbf{CA})_{kj}\mathbf{B}_{jk} \\
&= \text{trace}(\rho^T \bullet (\mathbf{CA})\mathbf{B}).
\end{aligned}$$

Note no assumptions on the shapes of ρ, $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are made except that they are compatible in dimension.

Appendix E

MPDest

MPDest is a Graphical User Interface (GUI) program created by David Zhang at University of California, San Diego; it is written in MATLAB for performing state/parameter estimation. In its core is an adjoint-based algorithm that extract the local gradient of a cost function with respective to the optimization variables. Using the computed local gradient, one may use any gradient-based optimization routines to iteratively optimize the solution. In particular, MPDest uses the Limited-memory Reduced-Hessian for simple Bounds, version Engineer (LRHBvE) optimization algorithm developed by Michael Ferry at University of California, San Diego to perform optimization with inequality linear constraints.

Before moving on to explain in detail how one may use MPDest, it is important to understand the mathematical principles behind the algorithm, so that one may improve upon MPDest and make it more versatile (or at the opposite extreme, optimize it to fit one's unique application). To this end, the general problem MPDest designed to solved is presented in §E.1. The MPDest documentation will start in §E.2, with many examples reflecting various features of MPDest and how different types of problems can be posed in the MPDest framework shown in §E.3.

E.1 Theory

Suppose a nonlinear system is described by

$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = f(\mathbf{x}(t), \mathbf{p}, \mathbf{u}(t)), \tag{E.1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state, $\mathbf{p} \in \mathbb{R}^q$ is some unknown parameters, and $\mathbf{u} \in \mathbb{R}^c$ is the known control input. Furthermore, $\mathbf{x}(t)$ is observed at time t_k through the measurement $\mathbf{y}_k$:

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k) + \mathbf{v}_k, \tag{E.2}$$

where the subscript $(\,)_k$ denotes the time instance t_k, and $\mathbf{v}_k$ is the measurement noise modeled with Gaussian statics of zero mean with covariance $\mathbf{R}_\mathbf{y}$.

In real world situations, (E.1) is not known exactly. Typically, the system parameters $\mathbf{p}$ are estimated based on first principles (e.g. mass, angular inertia, damping coefficient). However these estimates may be different in real life due to assumptions made in order to use first principle and may be in fact not appropriate. For example, when estimating the inertia of oddly shaped components, their geometries are typically simplified in order to use established first principle equations. While such approximation is fairly accurate for a single part, the error begins to compound as more parts are assembled together, resulting an error between computed and actual inertia. The consequence of having the incorrect system parameters varies depends on the robustness of the system and its associated controller. In a robust system incorrect parameters reduce overall system performance, but at least stability is guaranteed; however for highly sensitive systems and/or high performance controllers, incorrect parameter estimates would lead to system instability In real life scenarios system instabilities would lead to consequences ranging from damage machineries all the way to catastrophes.

In order to refine the parameter estimates, typically a time history of measurements within a time interval $[t_0, t_1, \cdots, t_T]$ from the system is taken, and based on the same first principle equations the parameters are adjusted so that the expected measurements from (E.2) matches the actual measurements. In the MPDest

framework, this equates to minimizing a scalar cost J:

$$\min_{\mathbf{x}_0,\mathbf{p}} \underbrace{\frac{1}{2}\left(\sum_{k=0}^{T} ||h(\mathbf{x}_k,\mathbf{p},\mathbf{u}_k)-\mathbf{y}_k||_{\mathbf{Q_y}} + ||\mathbf{p}-\mathbf{p}_b||_{\mathbf{Q_p}} + ||\mathbf{x}_0-\mathbf{x}_b||_{\mathbf{Q_x}}\right)}_{J}, \tag{E.3}$$

where subscript $(\,)_b$ denotes initial estimate and $||\mathbf{a}||_{\mathbf{Q}} \triangleq \mathbf{a}^T\mathbf{Q}\mathbf{a}$.

The first term in (E.3) reflects the mis-match between the actual measurement $\mathbf{y}_k$ and the expected measurement $h(\mathbf{x}_k,\mathbf{p},\mathbf{u}_k)$. The second term in (E.3) penalizes the mis-match between estimated $\mathbf{p}$ with the initial $\mathbf{p}_b$ established through first principle. Although one accept that the initial parameter estimate $\mathbf{p}_b$ is not accurate, one would typically has some confidence in $\mathbf{p}_b$; this confidence can be modeled as the covariance of $\mathbf{p}_b$, $\mathbf{R_p}$. Finally, the third term in (E.3) penalizes the mis-match between the estimated state initial condition $\mathbf{x}_0$ and the state initial condition $\mathbf{x}_b$. Although typically $\mathbf{x}_b$ is known fairly accurate ahead of time (for example, the initial velocity of an system is zero), because the state initial condition heavily influences the outcome of $\mathbf{x}(t)$ and therefore $h(\mathbf{x}_k,\mathbf{p},\mathbf{u}_k)$, it is beneficial to assume $\mathbf{x}_b$ could potentially be inaccurate; the confidence in $\mathbf{x}_b$ is modeled through the covariance $\mathbf{R_x}$.

MPDest computes the local gradient of J with respect to a nominal $\mathbf{x}_0$ and $\mathbf{p}$ value, and use gradient-based iterative optimization method for optimization. The local gradient computation can be derived analytically, which shall be demonstrated in the following.

Suppose one has an initial value for $\mathbf{x}_0$ and $\mathbf{p}$; with this, one could propagate (E.1) to t_T, evaluate all $h(\mathbf{x}_k,\mathbf{p},\mathbf{u}_k)$, and evaluate J. Now imagine perturbations are applied to $\mathbf{x}_0$ and $\mathbf{p}$; such perturbation propagates and affect $\mathbf{x}_k$ and J, with

the first order perturbations being:

$$\begin{gathered}\frac{\mathrm{d}\mathbf{x}'(t)}{\mathrm{d}t} = \mathbf{A}(t)\mathbf{x}'(t) + \mathbf{B}(t)\mathbf{p}', \\ \mathbf{A}(t) \triangleq \frac{\partial f(\mathbf{x}(t), \mathbf{p}, \mathbf{u}(t))}{\partial \mathbf{x}(t)}, \qquad \mathbf{B}(t) \triangleq \frac{\partial f(\mathbf{x}(t), \mathbf{p}, \mathbf{u}(t))}{\partial \mathbf{p}},\end{gathered} \tag{E.4a}$$

$$\begin{aligned}J' = &\sum_{k=0}^{T} (h(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k) - \mathbf{y}_k))^T \mathbf{Q}_\mathbf{y} \mathbf{H}_\mathbf{x}(\mathbf{p}) \mathbf{x}'_k + \sum_{k=0}^{T} (h(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k) - \mathbf{y}_k))^T \mathbf{Q}_\mathbf{y} \mathbf{H}_\mathbf{p}(\mathbf{p}) \mathbf{p}' \\ &+ (\mathbf{x}_0 - \mathbf{x}_b)^T \mathbf{Q}_\mathbf{x} \mathbf{x}'_0 + (\mathbf{p} - \mathbf{p}_b)^T \mathbf{Q}_\mathbf{p} \mathbf{p}',\end{aligned} \tag{E.4b}$$

where $\mathbf{u}'(t) = \mathbf{0}$ because it is not affected by the perturbations. The goal is to express (E.4b) in the form

$$J' = (\nabla_{\mathbf{x}_0} J)^T \mathbf{x}'_0 + (\nabla_\mathbf{p} J)^T \mathbf{p}', \tag{E.5}$$

so that the local gradients are readily extracted. Clearly the majority of (E.4b) except the first term already satisfies the form; adjoint analysis is used perform the conversion for the first term.

For notational simplicity (E.4a) is rewritten by introducing the operator $\mathrm{L}(\mathbf{x}'(t)) \triangleq \mathrm{d}\mathbf{x}'(t)/\mathrm{d}t - \mathbf{A}(t)\mathbf{x}'(t)$, so that now (E.4a) becomes

$$\mathrm{L}(\mathbf{x}'(t)) = \mathbf{B}(t)\mathbf{p}'. \tag{E.6}$$

Now an adjoint variable $\mathbf{r}(t) \in \mathbb{R}^n$ is defined along with the appropriate inner product

$$\langle \mathbf{r}(t), \mathrm{L}(\mathbf{x}'(t)) \rangle_{a,b} = \langle \mathrm{L}^*(\mathbf{r}(t)), \mathbf{x}'(t) \rangle_{a,b} + c, \qquad \langle \mathbf{a}(t), \mathbf{b}(t)) \rangle_{a,b} \triangleq \int_{t_a}^{t_b} \mathbf{a}(t)^T \mathbf{b}(t)\, \mathrm{d}t. \tag{E.7}$$

Using integration by parts, it can be shown that

$$\mathrm{L}^*(\mathbf{r}(t)) = -\frac{\mathrm{d}\mathbf{r}(t)}{\mathrm{d}t} - \mathbf{A}(t)^T \mathbf{r}(t), \tag{E.8a}$$

$$c = \mathbf{r}_b^T \mathbf{x}'_b - \mathbf{r}_a^T \mathbf{x}'_a. \tag{E.8b}$$

Now, if (E.7) is defined over the time interval $(t_{T-1},\, t_T]$, let $\mathrm{L}^*(\mathbf{r}(t)) = \mathbf{0}$ in (E.8a), and let $\mathbf{r}_T = \mathbf{H}_\mathbf{x}(\mathbf{p})^T\mathbf{Q}_\mathbf{y}(h(\mathbf{x}_T, \mathbf{p}, \mathbf{u}_T) - \mathbf{y}_T)$, then (E.4b) can be rewritten into

$$
\begin{aligned}
J' &= \sum_{k=0}^{T-1}(h(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k) - \mathbf{y}_k))^T\mathbf{Q}_\mathbf{y}\mathbf{H}_\mathbf{x}(\mathbf{p})\mathbf{x}'_k + \sum_{k=0}^{T}(h(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k) - \mathbf{y}_k))^T\mathbf{Q}_\mathbf{y}\mathbf{H}_\mathbf{p}(\mathbf{p})\mathbf{p}' \\
&\quad + (\mathbf{x}_0 - \mathbf{x}_b)^T\mathbf{Q}_\mathbf{x}\mathbf{x}'_0 + (\mathbf{p} - \mathbf{p}_b)^T\mathbf{Q}_\mathbf{p}\mathbf{p}' \\
&\quad + \mathbf{r}_T^T\mathbf{x}'_T, \\
&= \sum_{k=0}^{T-1}(h(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k) - \mathbf{y}_k))^T\mathbf{Q}_\mathbf{y}\mathbf{H}_\mathbf{x}(\mathbf{p})\mathbf{x}'_k + \sum_{k=0}^{T}(h(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k) - \mathbf{y}_k))^T\mathbf{Q}_\mathbf{y}\mathbf{H}_\mathbf{p}(\mathbf{p})\mathbf{p}' \\
&\quad + (\mathbf{x}_0 - \mathbf{x}_b)^T\mathbf{Q}_\mathbf{x}\mathbf{x}'_0 + (\mathbf{p} - \mathbf{p}_b)^T\mathbf{Q}_\mathbf{p}\mathbf{p}' \\
&\quad + \langle \mathbf{r}(t), \underbrace{\mathrm{L}(\mathbf{x}'(t))}_{\mathbf{B}(t)\mathbf{p}'}\rangle_{T-1,T} - \langle \underbrace{\mathrm{L}^*(\mathbf{r}(t))}_{\mathbf{0}}, \mathbf{x}'(t)\rangle_{T-1,T} + (\mathbf{r}^+_{T-1})^T\mathbf{x}'_{T-1}, \\
&= \sum_{k=0}^{T-1}(h(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k) - \mathbf{y}_k))^T\mathbf{Q}_\mathbf{y}\mathbf{H}_\mathbf{x}(\mathbf{p})\mathbf{x}'_k + \sum_{k=0}^{T}(h(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k) - \mathbf{y}_k))^T\mathbf{Q}_\mathbf{y}\mathbf{H}_\mathbf{p}(\mathbf{p})\mathbf{p}' \\
&\quad + (\mathbf{x}_0 - \mathbf{x}_b)^T\mathbf{Q}_\mathbf{x}\mathbf{x}'_0 + (\mathbf{p} - \mathbf{p}_b)^T\mathbf{Q}_\mathbf{p}\mathbf{p}' \\
&\quad + \int_{t^+_{T-1}}^{t_T} (\mathbf{B}(t)^T\mathbf{r}(t))^T\mathbf{p}'\,\mathrm{d}t + (\mathbf{r}^+_{T-1})^T\mathbf{x}'_{T-1},
\end{aligned}
\tag{E.9}
$$

where the $(\,)^+$ superscript denotes the time instance immediate to the right of t_{T-1}, since the adjoint identity is defined over the left open interval. Note besides the shifted time index in the first term and the addition of the last two terms in (E.9), it is identical to (E.4b). Therefore, if the same adjoint identity in (E.8) is defined over the time interval $(t_{T-2},\, t_{T-1}]$, let $\mathrm{L}^*(\mathbf{r}(t)) = \mathbf{0}$, and $\mathbf{r}_{T-1} = \mathbf{H}_\mathbf{x}(\mathbf{p})^T\mathbf{Q}_\mathbf{y}(h(\mathbf{x}_{T-1}, \mathbf{p}, \mathbf{u}_{T-1}) - \mathbf{y}_{T-1}) + \mathbf{r}^+_{T-1}$, following the similar analysis

as (E.9) and it can be rewritten again into

$$\begin{aligned} J' = &\sum_{k=0}^{T-2}(h(\mathbf{x}_k,\mathbf{p},\mathbf{u}_k)-\mathbf{y}_k))^T\mathbf{Q}_\mathbf{y}\mathbf{H}_\mathbf{x}(\mathbf{p})\mathbf{x}'_k + \sum_{k=0}^{T}(h(\mathbf{x}_k,\mathbf{p},\mathbf{u}_k)-\mathbf{y}_k))^T\mathbf{Q}_\mathbf{y}\mathbf{H}_\mathbf{p}(\mathbf{p})\mathbf{p}' \\ &+ (\mathbf{x}_0-\mathbf{x}_b)^T\mathbf{Q}_\mathbf{x}\mathbf{x}'_0 + (\mathbf{p}-\mathbf{p}_b)^T\mathbf{Q}_\mathbf{p}\mathbf{p}' \\ &+ \int_{t^+_{T-2}}^{t_T}(\mathbf{B}(t)^T\mathbf{r}(t))^T\mathbf{p}'\,\mathrm{d}t + (\mathbf{r}^+_{T-2})^T\mathbf{x}'_{T-2}, \end{aligned} \tag{E.10}$$

which is just (E.9) with a shifted time index.

In general, if J' is in the form

$$\begin{aligned} J' = &\sum_{k=0}^{K}(h(\mathbf{x}_k,\mathbf{p},\mathbf{u}_k)-\mathbf{y}_k))^T\mathbf{Q}_\mathbf{y}\mathbf{H}_\mathbf{x}(\mathbf{p})\mathbf{x}'_k + \sum_{k=0}^{T}(h(\mathbf{x}_k,\mathbf{p},\mathbf{u}_k)-\mathbf{y}_k))^T\mathbf{Q}_\mathbf{y}\mathbf{H}_\mathbf{p}(\mathbf{p})\mathbf{p}' \\ &+ (\mathbf{x}_0-\mathbf{x}_b)^T\mathbf{Q}_\mathbf{x}\mathbf{x}'_0 + (\mathbf{p}-\mathbf{p}_b)^T\mathbf{Q}_\mathbf{p}\mathbf{p}' \\ &+ \int_{t^+_K}^{t_T}(\mathbf{B}(t)^T\mathbf{r}(t))^T\mathbf{p}'\,\mathrm{d}t + (\mathbf{r}^+_K)^T\mathbf{x}'_K, \end{aligned} \tag{E.11}$$

then by using the adjoint identity defined in (E.8) within the time interval $(t_{K-1},\ t_K]$, and let

$$\mathrm{L}^*(\mathbf{r}(t)) = \mathbf{0}, \tag{E.12a}$$

$$\mathbf{r}_K = \begin{cases} \mathbf{H}_\mathbf{x}(\mathbf{p})^T\mathbf{Q}_\mathbf{y}(h(\mathbf{x}_T,\mathbf{p},\mathbf{u}_T)-\mathbf{y}_T), & \text{if } K=T \\ \mathbf{H}_\mathbf{x}(\mathbf{p})^T\mathbf{Q}_\mathbf{y}(h(\mathbf{x}_K,\mathbf{p},\mathbf{u}_K)-\mathbf{y}_K)+\mathbf{r}^+_K, & \text{otherwise} \end{cases}, \tag{E.12b}$$

the J' equation in (E.11) can be rewritten into

$$\begin{aligned} J' = &\sum_{k=0}^{K-1}(h(\mathbf{x}_k,\mathbf{p},\mathbf{u}_k)-\mathbf{y}_k))^T\mathbf{Q}_\mathbf{y}\mathbf{H}_\mathbf{x}(\mathbf{p})\mathbf{x}'_k + \sum_{k=0}^{T}(h(\mathbf{x}_k,\mathbf{p},\mathbf{u}_k)-\mathbf{y}_k))^T\mathbf{Q}_\mathbf{y}\mathbf{H}_\mathbf{p}(\mathbf{p})\mathbf{p}' \\ &+ (\mathbf{x}_0-\mathbf{x}_b)^T\mathbf{Q}_\mathbf{x}\mathbf{x}'_0 + (\mathbf{p}-\mathbf{p}_b)^T\mathbf{Q}_\mathbf{p}\mathbf{p}' \\ &+ \int_{t^+_{K-1}}^{t_T}(\mathbf{B}(t)^T\mathbf{r}(t))^T\mathbf{p}'\,\mathrm{d}t + (\mathbf{r}^+_{K-1})^T\mathbf{x}'_{K-1}. \end{aligned} \tag{E.13}$$

Repeating the steps from (E.11) to (E.13) to sequentially shift the time index toward t_0^+, eventually one would arrive

$$\begin{aligned} J' = {} & (h(\mathbf{x}_0, \mathbf{p}, \mathbf{u}_0) - \mathbf{y}_0))^T \mathbf{Q}_\mathbf{y} \mathbf{H}_\mathbf{x}(\mathbf{p}) \mathbf{x}_0' + \sum_{k=0}^{T} (h(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k) - \mathbf{y}_k))^T \mathbf{Q}_\mathbf{y} \mathbf{H}_\mathbf{p}(\mathbf{p}) \mathbf{p}' \\ & + (\mathbf{x}_0 - \mathbf{x}_b)^T \mathbf{Q}_\mathbf{x} \mathbf{x}_0' + (\mathbf{p} - \mathbf{p}_b)^T \mathbf{Q}_\mathbf{p} \mathbf{p}' \\ & + \int_{t_0^+}^{t_T} (\mathbf{B}(t)^T \mathbf{r}(t))^T \mathbf{p}' \, \mathrm{d}t + (\mathbf{r}_0^+)^T \mathbf{x}_0'. \end{aligned} \tag{E.14}$$

As a formality the adjoint identity is defined one last time for (E.14) over the time interval $[t_0, t_0^+]$, and it is transformed to

$$\begin{aligned} J' = {} & \sum_{k=0}^{T} (h(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k) - \mathbf{y}_k))^T \mathbf{Q}_\mathbf{y} \mathbf{H}_\mathbf{p}(\mathbf{p}) \mathbf{p}' + (\mathbf{x}_0 - \mathbf{x}_b)^T \mathbf{Q}_\mathbf{x} \mathbf{x}_0' + (\mathbf{p} - \mathbf{p}_b)^T \mathbf{Q}_\mathbf{p} \mathbf{p}' \\ & + \int_{t_0}^{t_T} (\mathbf{B}(t)^T \mathbf{r}(t))^T \mathbf{p}' \, \mathrm{d}t + (\mathbf{r}_0)^T \mathbf{x}_0'. \end{aligned} \tag{E.15}$$

Hence according to (E.5), the local gradient of J with respect to $\mathbf{x}_0$ and $\mathbf{p}$ are

$$\nabla_{\mathbf{x}_0} J = \mathbf{Q}_\mathbf{x}(\mathbf{x}_0 - \mathbf{x}_b) + \mathbf{r}_0, \tag{E.16a}$$

$$\nabla_\mathbf{p} J = \mathbf{Q}_\mathbf{p}(\mathbf{p} - \mathbf{p}_b) + \sum_{k=0}^{T} \mathbf{H}_\mathbf{p}(\mathbf{p})^T \mathbf{Q}_\mathbf{y}(h(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k) - \mathbf{y}_k) + \int_{t_0}^{t_T} \mathbf{B}(t)^T \mathbf{r}(t) \, \mathrm{d}t. \tag{E.16b}$$

Note in (E.15) the $\mathbf{p}'$ term in the integral is moved outside of the integral because $\mathbf{p}$ is not a function of t, thus the integral becomes part of the gradient definition in (E.16b).

The analytically local gradient expression in (E.16) can be used to compute the numerical values of the gradients and readily be used by gradient-based optimization methods.

E.2 Documentation

MPDest is a GUI front-end wraps around the theory established in §E.1, with an gradient-based optimization algorithm developed by Michael Ferry at

UCSD. It is written with MATLAB 2009b and the source code can be checked out using the CVS at `fccr.ucsd.edu:/Users/dzhang/cvsroot` under the project name `MPDest`. There is a version of MPDest for Windows and a version for Mac/Linux; the Mac/Linux version is under the main trunk, and the Window version is under the `windows` branch. The MPDest compatibility with older/newer version of MATLAB has not being tested, but based on the current forward-compatibility trend in MATLAB, MPDest should most likely function within newer versions of MATLAB. In the following the various MPDest menus are explained. Note due to MATLAB's limited GUI control, each menu may looks different on different computer platforms.

An example with a simple spring-damper-mass system is used to illustrate how the system information is entered into MPDest; this full example can be loaded from `./example/spring_damper_mass/spring_damper_mass.mat` (explained in §E.2.9). Other examples can also be loaded within the `example` directory. The spring-damper-mass system is described by a second order ODE:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x1 \\ x2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k & -d \end{bmatrix}\begin{bmatrix} x1 \\ x2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u. \tag{E.17a}$$

One could give the variables more intuitive names, as an illustration from hereon $x1$ and d shall be referred to as *pos* and *damping*. The system is initially at rest (zero position and velocity), and is forced by a sinusoidal input u. Measurement on the position and velocity are taken with measurement noise covariance 0.01:

$$\begin{bmatrix} y1 \\ vel \end{bmatrix} = \begin{bmatrix} pos \\ x2 \end{bmatrix} + \mathbf{v}, \qquad \mathbf{v} \sim N(0, 0.01\mathbf{I}). \tag{E.17b}$$

In this example the spring constant $k = 0.5$ is known accurately whereas the *damping* is not; therefore k will be treated as a constant while *damping* will be treated as a parameter. For simulation purpose, in the truth system *damping* = 0.2, but the initial parameter estimate for *damping* is 0.35 with covariance of 0.25. Also for illustration purposes the initial value of *pos* will be estimated with initial value 0.1 and covariance 0.01.

E.2.1 Main Menu

To start MPDest, the users should first fire up MATLAB, and change the current directory to where MPDest is stored; then one simply type `MPDest` in the command window, which will bring up the main menu as shown in Figure E.1. All the menu items are self explanatory; however if the users are confused about the function of a particular menu item, simply hover the cursor over that item and a tool-tip explaining the menu item will appear.

The first column of text-boxes are where the number of states $\mathbf{x}$, measurements $\mathbf{y}$, constants, parameters $\mathbf{p}$, and forcings $\mathbf{u}$ are entered. The difference between constants and parameters is that parameters are estimated while constants are not. Note the users need to check the `Forced?` check-box if the system is forced; also by default MPDest only estimate the parameter (in other words, the initial condition $\mathbf{x}_0$ is assumed to be exact), however estimation on $\mathbf{x}_0$ can be enabled by checking the `Est. States IC?` check-box. The submenus where the user enter the corresponding system informations are visible only after a nonzero number has been entered into the corresponding text-box; they will be explained in detail in §E.2.2 through §E.2.7.

The `Go` button at the lower right hand side starts the optimization process, and the `Stop` and `Monitor` button will also be explained in §E.2.8 The message box at the lower left hand corner shows the status of the program. Lastly, one can save the entered system information and import it back later, this is done within the `File` submenu which will be explained in §E.2.9. This is also where MPDest program parameters are set. The name of the saved system is displayed at the bottom of the main menu.

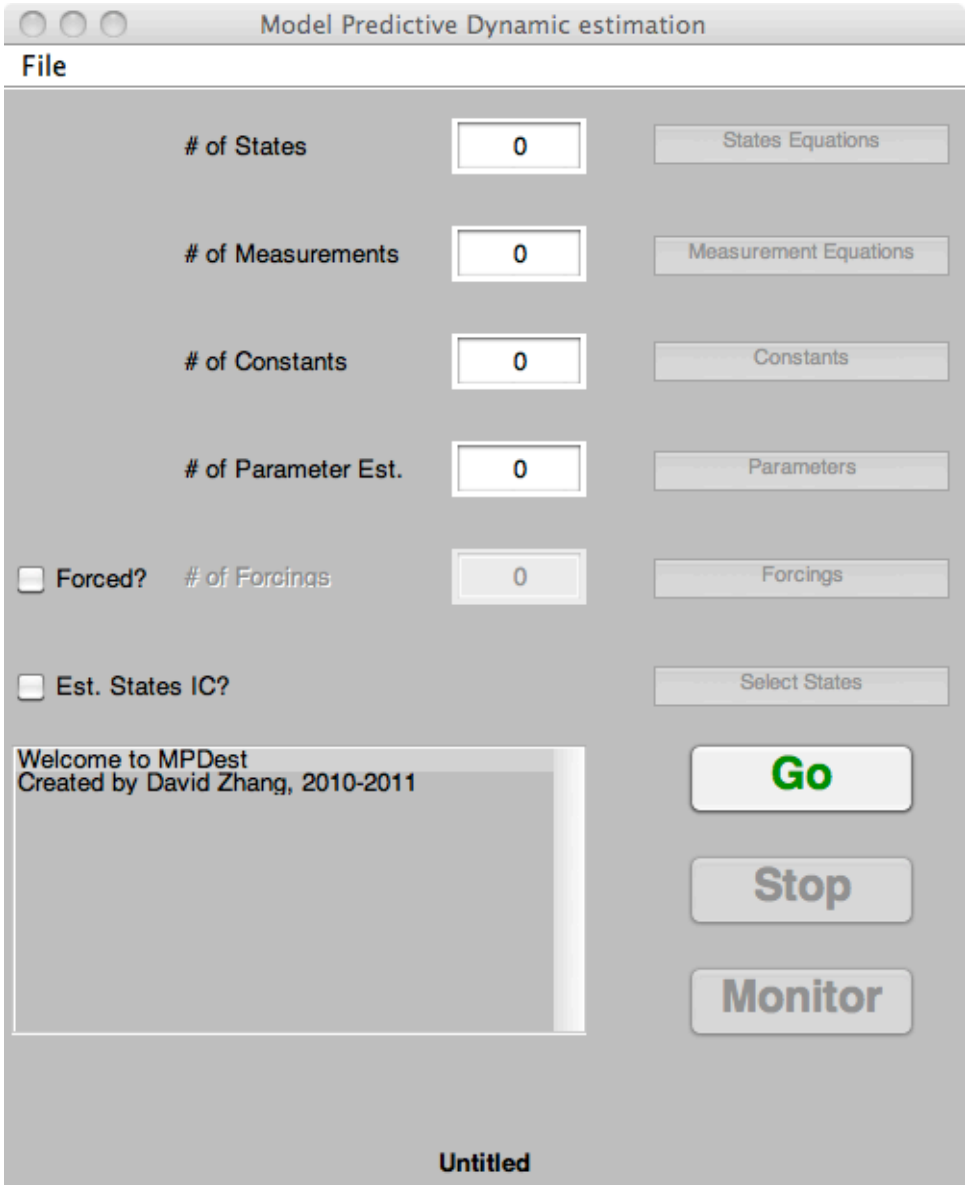


Figure E.1: MPDest Main menu.

E.2.2 State Equations Menu

The State Equation menu is dynamically generated depending on the value entered in the `# of States` text-box. The first column of text-boxes is where the names of the state variables are entered; the variable name could be anything (e.g. x1, velocity, pos). The nonlinear state equations are entered in the second column. One may enter the full state equations including the name of constants, parameters, and forcings; the definition of constants, parameters, and forcings will be handled in other menus. The third column is where the state initial conditions are entered, and any user-specific notes can be optionally entered in the fourth column. Figure E.2 shows an example of a State Equation menu with the system information in (E.17) entered.

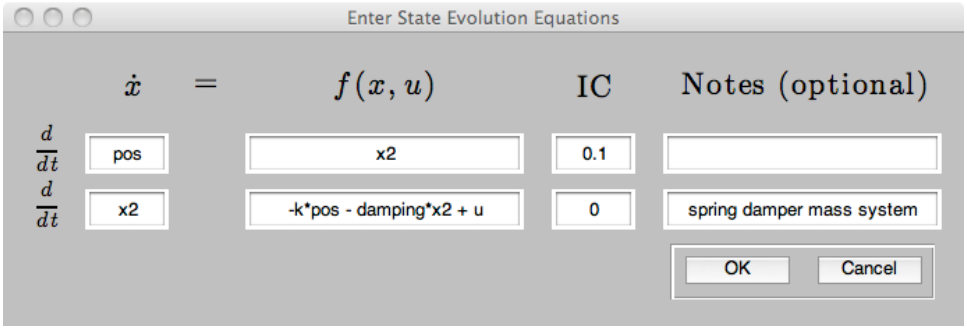


Figure E.2: State Equation menu entered with the spring-damper-mass system information.

E.2.3 Measurement Equations Menu

The Measurement Equation menu is dynamically generated depending on the value entered in the `# of Measurements` text-box. The first column of text-boxes is where the names of the measurements are entered; the variable name could be anything (e.g. x1, velocity, pos). The nonlinear measurement equations are entered in the second column. One may enter the full measurement equations including the name of constants, parameters, and forcings; the definition of constants, parameters, and forcings will be handled in other menus. The third column is where actual measurement and time data files are loaded. Each measurement

data file should only contain a single column of data; the associated time of those data are contained in a single column in the time data file. When data is loaded, the corresponding load button text will turn green, and the button tool-tip will point to the location of the data file. Currently MPDest assumes identical measurements at each time instance (but not necessary regular measurement time intervals), but clearly it could be generalized to accept data sampled at different rate by allowing a time varying $h_k(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k)$. This is one potential improvement that could be done to MPDest. After the data files are loaded, the data can be previewed by clicking the corresponding button in column four. Note the data preview only plots the data sequence without regarding the time, therefore the preview of measurement data with irregular measurement time interval will not accurately represent the actual measurement data. The fifth column is where the measurement noise covariances are entered, and any user-specific notes can be optionally entered in the sixth column. Figure E.3 shows an example of a Measurement Equation menu with the system information in (E.17) entered.

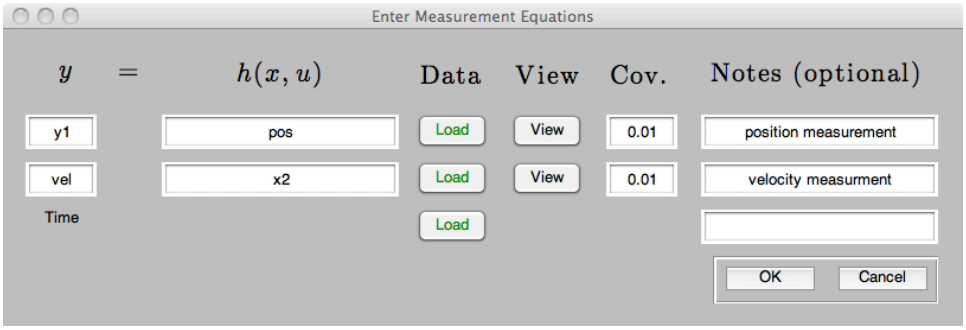


Figure E.3: Measurement Equation menu entered with the spring-damper-mass system information.

E.2.4 Constants Menu

The Constants menu is dynamically generated depending on the value entered in the `# of Constants` text-box. The first column of text-boxes is where the names of the constants are entered; the variable name could be anything (e.g. x1, velocity, pos). The values of the constants are entered in the second column,

with optimal user-specific notes in the third column. Figure E.4 shows an example of a Constants menu with the system information in (E.17) entered.

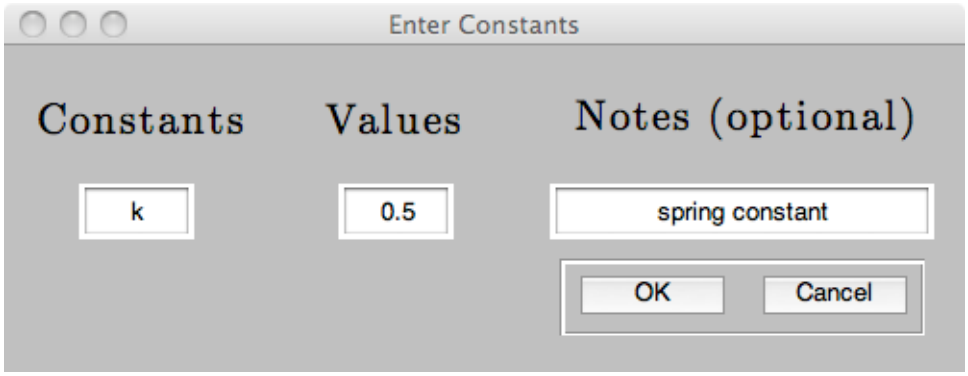


Figure E.4: Constants menu entered with the spring-damper-mass system information.

An unique feature with the Constants menu is that the value input supports expressions; that is, a mathematical formula can be entered in the `Values` text-box instead of numeric values. This may come in handy when some constants are defined asr functions of other constants. For example, one may define three constants M, R, and I, respectively denoting the mass, radius, and moment of inertia of a solid sphere. The moment of inertia for a solid sphere is $I = 2/5MR^2$. In practice one could obtain reasonably accurate measurement on M and R, but not I. However instead of plug in the values of M and R in a calculator and round the result to put in the I field, one may instead input the value of I as a mathematical formula, as illustrated in Figure E.5. Note from Figure E.5 that stand MATLAB functions and constants could also be used. Also note the order which the constant definition appears does not matter, however for clarity it is generally recommended that the users enter constants with numeric values first, then follow by those with mathematical expressions.

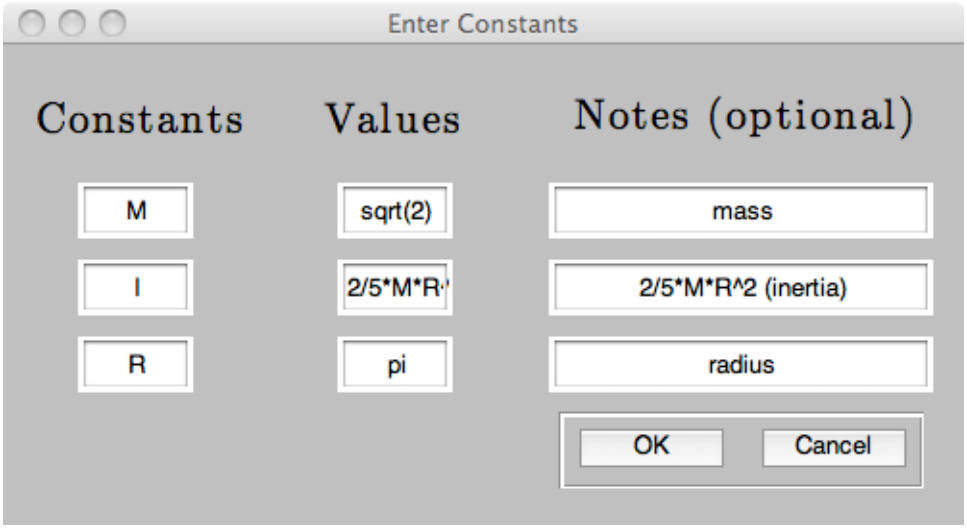


Figure E.5: An example of using mathematical expressions in the Constants menu.

E.2.5 Parameters Menu

The Parameters menu is dynamically generated depending on the value entered in the `# of Parameter Est.` text-box. The first column of text-boxes is where the names of the parameters are entered; the variable name could be anything (e.g. x1, velocity, pos). The estimated values of the parameters are entered in the second column, with the covariance of estimates in the third column. The fourth and fifth column defines the hard constraints which the parameters are valid. This is useful to constraint the optimization to produce physically sensible parameter estimates, so that for example, one would not receive a negative mass parameter estimate as the optimal solution. Note that the value `inf` and `-inf` are valid bound values. The sixth column is where user enter user-specific notes. Figure E.6 shows an example of a Parameters menu with the system information in (E.17) entered.

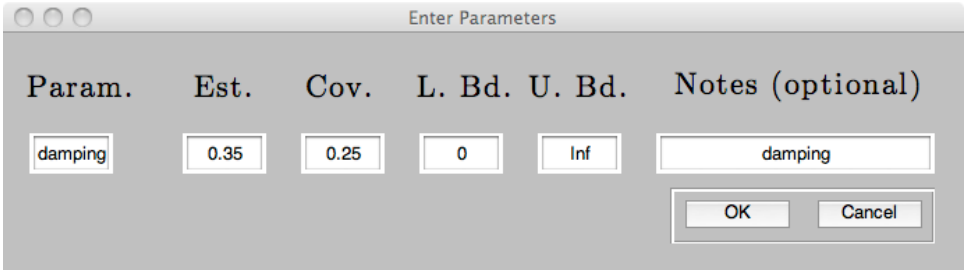


Figure E.6: Parameters menu entered with the spring-damper-mass system information.

E.2.6 Forcings Menu

The Forcings menu is dynamically generated from the value entered in the `# of Forcings` text-box. The first column of text-boxes is where the names of the forcings are entered; the variable name could be anything (e.g. x1, velocity, pos). The second column is where actual forcing data files are loaded; there is no need to load the time data file because currently MPDest assumes all forcings are applied at the same instances as when the measurements are taken, and assumes zero-order-holds between measurement times; but clearly it could be generalized so that the control and measurement times are decoupled. This is another potential improvement that could be done to MPDest. Each forcing data file should only contain a single column of data. When data is loaded, the corresponding load button text will turn green, and the button tool-tip will point to the location of the data file. After the data files are loaded, the data can be viewed by clicking the corresponding button in column three, and any user-specific notes can be optionally entered in the fourth column. Figure E.7 shows an example of a Forcings menu with the system information in (E.17) entered.

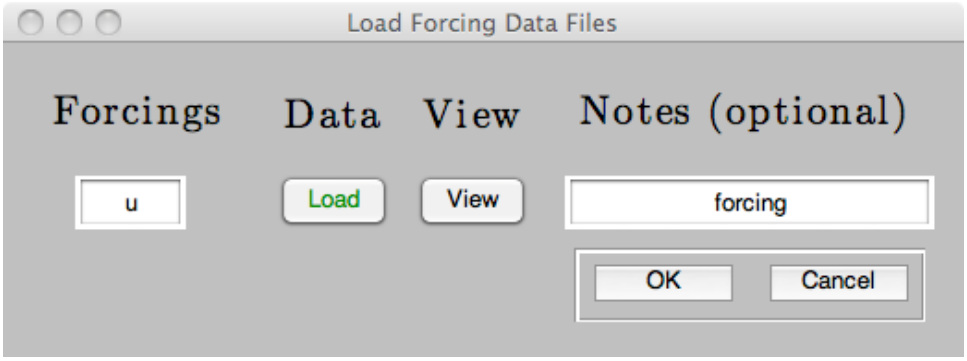


Figure E.7: Forcings menu entered with the spring-damper-mass system information.

E.2.7 Select States Menu

By default the state initial condition $\mathbf{x}_0$ is not estimated, however the user may optionally check the `Est. State IC?` check-box to enable initial state estimate. Once the Select States menu is opened, the first column displays the state names and the third column displays the initial values defined in the State Equation menu. The second column is where the users have the option to select the particular initial state to estimate. Once the check-box is selected, the corresponding covariance, lower bound, upper bound, and notes fields also become available for user input. The covariance field is where the confidence of the initial state estimate is entered. The lower and upper bounds are for the optimization to ensure sensible results. The notes field is for user-specific notes. Figure E.8 shows an example of a Select States menu with the system information in (E.17) entered, note again `inf` and `-inf` are valid bound values.

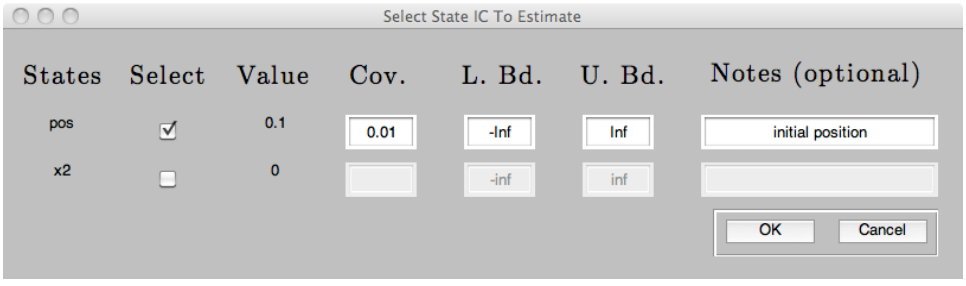


Figure E.8: Select States menu entered with the spring-damper-mass system information.

E.2.8 Begin Optimization

Once all necessary information are entered, the users simply need to press the `Go` button to start the optimization process. At this point, the lower portion of the Main menu will look similar to Figure E.9. Note the users now have access to the `Stop` and `Monitor` buttons. The `Stop` button stops the optimization process and the most current optimization solution can be accessed and exported (to be discussed in more detail in §E.2.9). The `Monitor` button allows the users to view the progress of the optimization by showing the current optimization solution. Figure E.10 and Figure E.11 show an example of the monitor when optimization is performed on the system described in (E.17). In Figure E.10 the red dots represents the actual data from the data files and the blue line represents the expected measurement from the current optimization solution; the last plot shows the history of the cost reduction with logarithmic y-axis. Lastly, the orange progress bar in the Main menu shows the how many optimization iterations have been completed compared to the maximum allowable optimization iteration (default is set to 1000 iterations, which can be changed as discussed in §E.2.9).

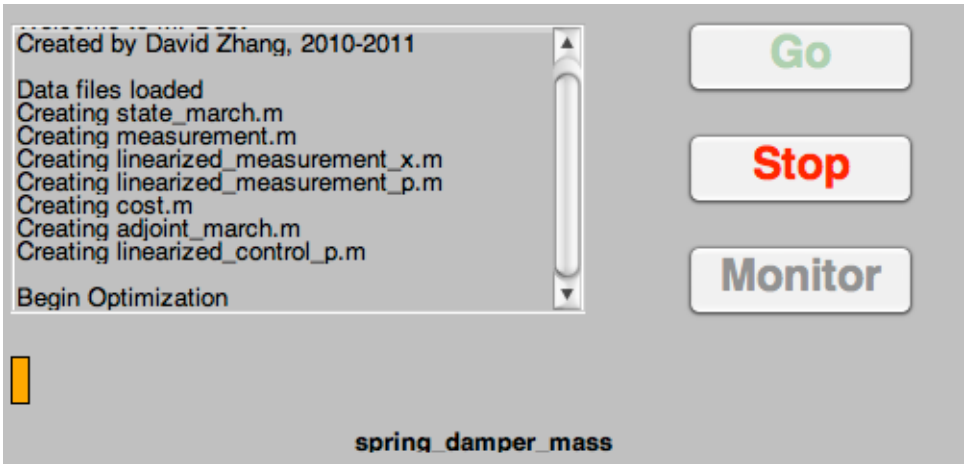


Figure E.9: Lower portion of the Main menu when optimization is in progress.

One thing should be made clear here is that MPDest automatically choose the diagonal of $\mathbf{Q_x}$, $\mathbf{Q_y}$, and $\mathbf{Q_p}$ in §E.1 as the inverse of the covariances entered in the State Equation, Measurement Equation, and Parameters memu. For example

MPDest will choose $\mathbf{Q_y} = \mathbf{R_y}^{-1}$. From maximum likelihood theory it can be shown that this is the optimal choice.

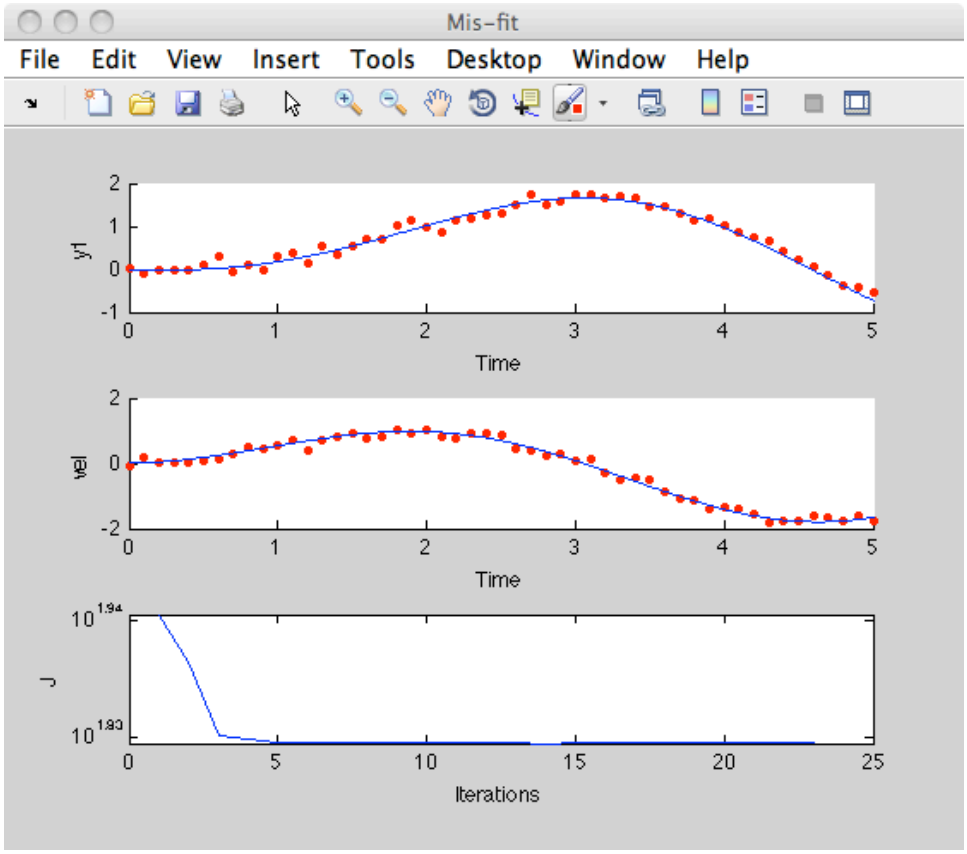


Figure E.10: The Mis-fit monitor.

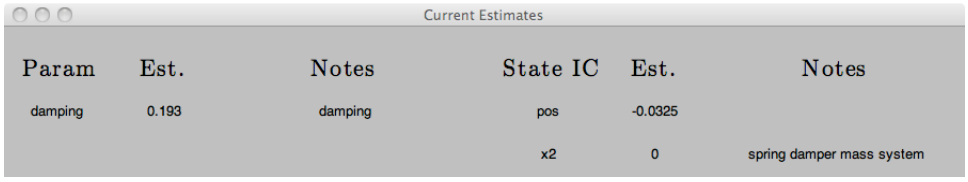


Figure E.11: The Current Estimates monitor.

E.2.9 File Menu

The File menu contains options that allow users to save the entered system, import previously saved system, export optimized solution, and setup some

MPDest-related system parameters, as seen in Figure E.12. Depends on system platforms, the shortcut key for these options will differ.

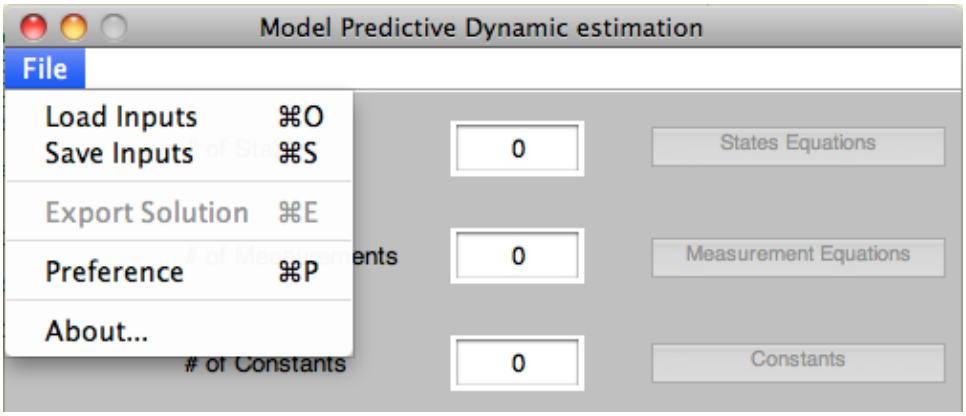


Figure E.12: The File menu.

The `Load Inputs` option allow users to save their system input progress; this can be done at any time, even if the system information has not been fully entered. Similarly, the `Save Inputs` option allows users to load their previously saved system.

The `Export Solution` option only becomes available when an optimization has been completed (whether terminated by the optimization algorithm, maximum optimization iteration reached, or prematurely stopped by the users). Once the solution has been saved, the users can load the solution back into MATLAB to view various aspects of the optimization. There are 4 exported variable within the exported solution: `J_hist`, `solution`, `solution_hist`, and `solution_name`. `J_hist` is the time history of the cost reduction, as shown in the last plot of the `Mis-fit` monitor. `solution` is the optimized solution; each element in `solution` is labeled with the corresponding element in `solution_name`. Lastly, `solution-hist` contains the convergence history of the solutions. For instance, when running the example in (E.17), stopping the optimization prematurely, and exporting the solution, the plot of the extracted history of the cost and the solution from `J_hist`, `solution_hist` and `solution_name` is shown in Figure E.13.

Choosing the `Preference` option brings up the Preference menu as shown in Figure E.14. The first field is the simulation step size used to propagate the

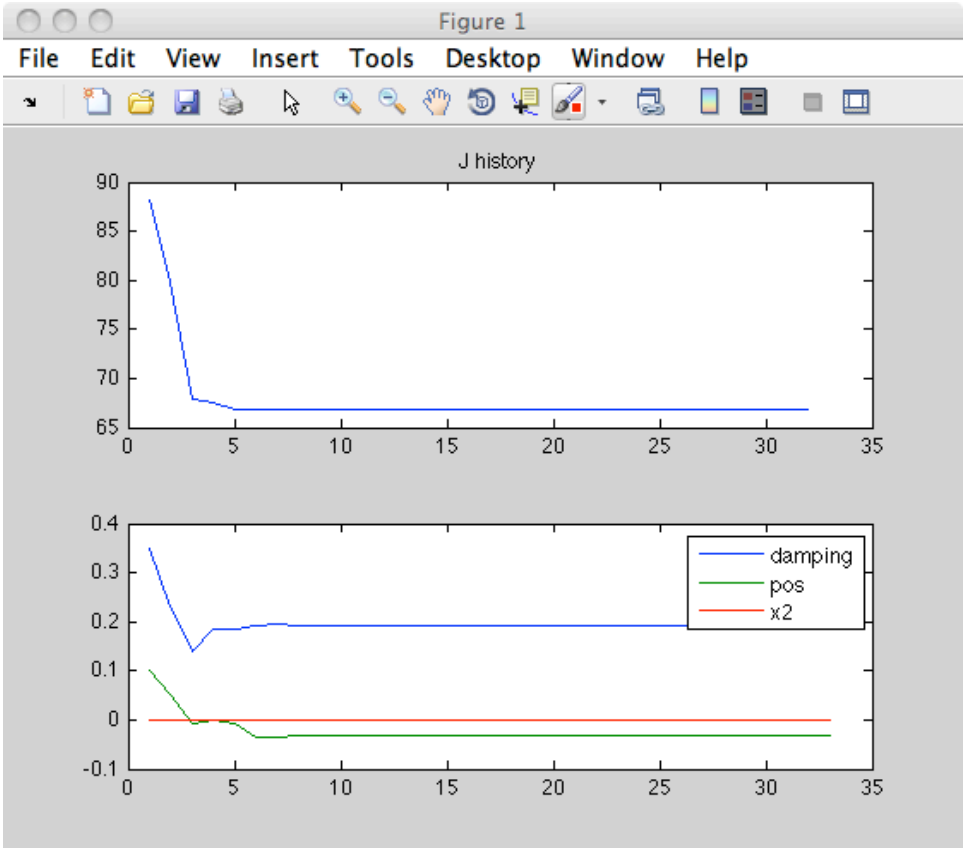


Figure E.13: History of J and the solution for the example system in (E.17)

system forward, where the default is 0.01 time unit. Depending on applications this number may be lengthened (for slow systems) or shortened (for fast systems). The second field sets the maximum optimization iterations, which is defaulted at 1000. The third field sets the number of gradients to be stored for the hessian approximation in the LRHBvE optimization algorithm; it is strongly recommended that the user do not change this number. The fourth field is the monitor refresh period, which is set to 5 seconds per refresh by default. When the refresh rate is high, it is likely the optimization performance will suffer because MATLAB spend some of the computation time in redrawing the `Mis-fit` and `Current Estimate` monitor. Lastly, the `Verbose` check-box enables addition output to the command window for monitoring the optimization progress. This is mostly for program debugging purpose and so the user should not need to enable this option.

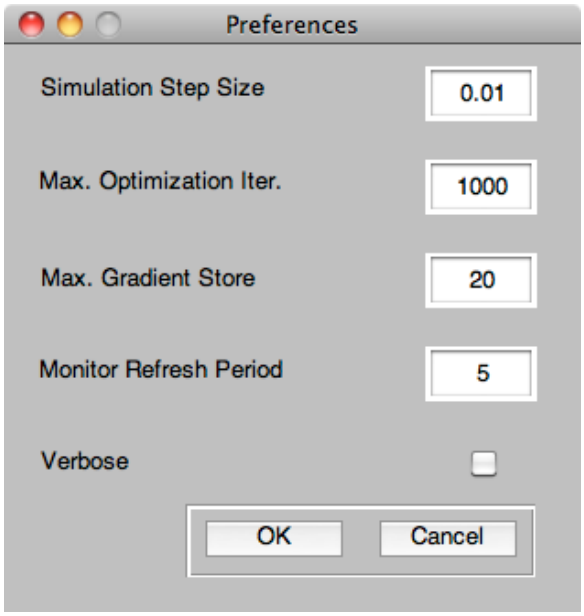


Figure E.14: The Preference menu.

E.3 More Examples

More examples in addition to the example presented in this documentation are available; they can be loaded into MPDest from the `./example` directory. The descriptions of the examples are presented in the following.

E.3.1 Lorenz System

This Lorenz system parameter estimation example is taken from the Master thesis by Sean Summers from UCSD. The Lorenz equation is a chaotic ODE

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= \sigma(y - x) \\ \frac{\mathrm{d}y}{\mathrm{d}t} &= x(\tau - z) - y. \\ \frac{\mathrm{d}z}{\mathrm{d}t} &= xy - \beta z \end{aligned} \tag{E.18}$$

Figure E.15 shows an example of the state trajectory in phase space of a typical Lorenz system. The objective of this example is to estimate the parameters the Prandtl number σ, the Rayleigh number τ, and a physical proportion β given a

time history of measurements made only on the first state x:

$$y = x + v, \qquad v \sim N(0, 1). \tag{E.19}$$

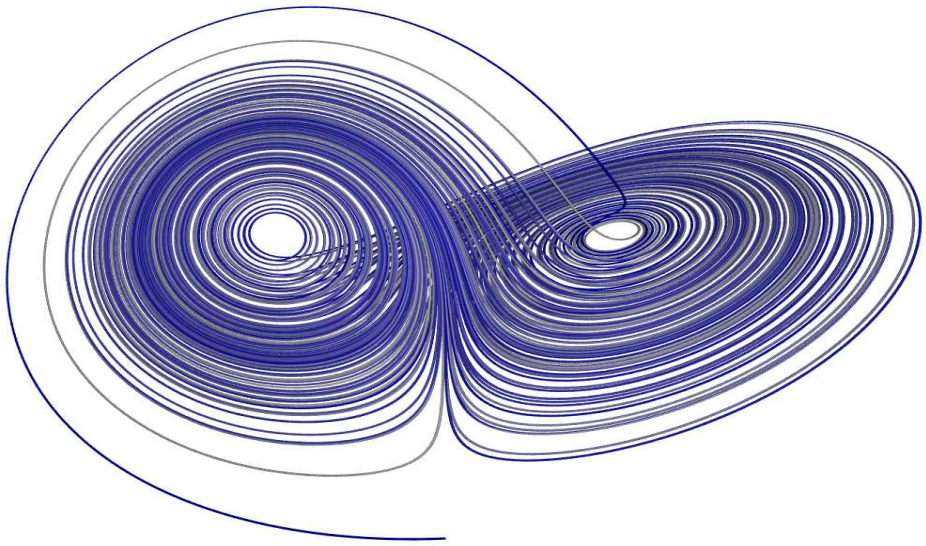


Figure E.15: An example of a Lorenz system in phase space

In this example measurements sampled at 100Hz are taken with a 0.5 time unit interval. The truth system parameter is $\mathbf{p} = \begin{bmatrix} 10 & 28 & 8/3 \end{bmatrix}$, with initial state condition $\mathbf{x} = \begin{bmatrix} 3 & 15 & 1 \end{bmatrix}$. Both the parameters and the state initial condition are estimated; the initial state estimate $\mathbf{x}_b = \begin{bmatrix} 10 & 19 & 2 \end{bmatrix}$ and covariance $\mathbf{R}_\mathbf{x} = \infty\mathbf{I}$, and initial parameter estimate $\mathbf{p}_b = \begin{bmatrix} 3 & 31 & 5/3 \end{bmatrix}$ with covariance $\mathbf{R}_\mathbf{y} = \infty\mathbf{I}$. This example can be loaded from `./example/lorenz_no_constraint`.

A second example for the Lorenz system is to constraint the interval where the solution is valid. Here, while σ and β are allowed to take on any value, β is restricted to be within the interval $\begin{bmatrix} 27.9 & 28.1 \end{bmatrix}$; also, in the z_0 estimate is defined to be valid only within $\begin{bmatrix} 0.9 & 1.1 \end{bmatrix}$. This example can be loaded from `./example/lorenz_w_constraint`.

E.3.2 Segway

This segway example is also taken from the Master thesis by Sean Summers. The segway example originates from designing a multi-functional robotic ground vehicle at the Flow Control and Coordinated Robotics Lab at UCSD. The idea is to build a segway-like robot that has the ability to switch from one configuration to another for different modes of maneuverability. Figure E.16 illustrates the concept design. For a dynamically unstable system such as this, a high performance control law is necessary; however the trade-off would be that the control law is highly sensitive to system parameters such as center-of-mass position and inertia, and the system could easily becomes unstable. Therefore, a good system parameter estimate is essential.

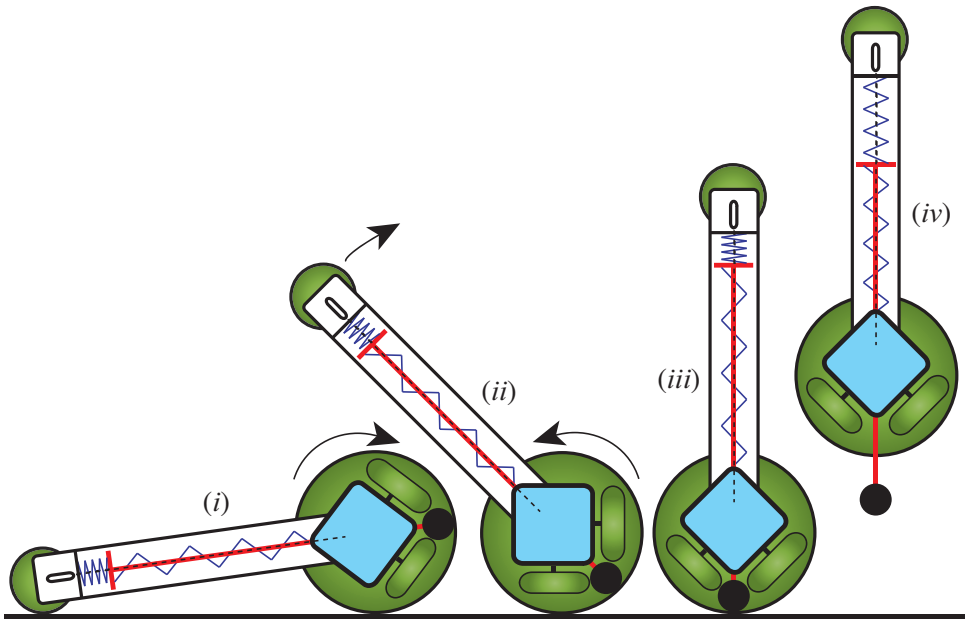


Figure E.16: The multi-functional ground vehicle — iFling.

Dynamical parameters such as inertia of centroid position cannot be measured accurately when the system is static, therefore a dynamical approach is taken. The robot is placed on a track with an open channel in the middle where the arm can pass through, allowing the arm to have a full 360 degrees of motion (see Figure E.17). With the system at rest at the stable equilibrium position, an

external sinusoidal input is applied to the wheels, causing the system to move. The objective is to use the measured data to estimate the uncertain system parameters.

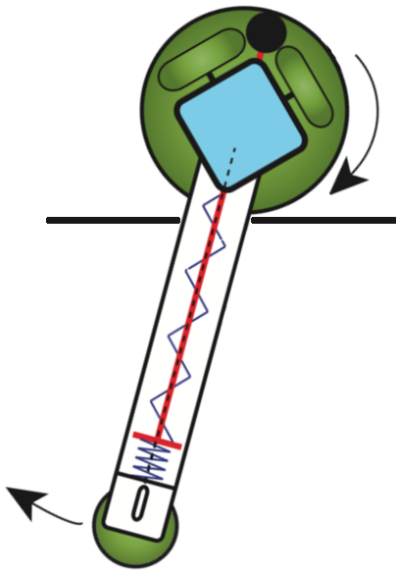


Figure E.17: Experiment setup to collect dynamical data.

Defining the arm angle with respect to the upright vertical θ and the clockwise wheel angle ϕ, the dynamic of Figure E.17 is

$$\begin{aligned}(M_pL^2 + J_p)\ddot{\theta} + M_pr_wL\ddot{\phi}\cos\theta &= M_pgL\sin\theta - \frac{k}{R}\left(\tau + k(\dot{\theta} - \dot{\phi})\right), \\ M_pr_wL\ddot{\theta}\cos\theta + \left(J_w + (M_p + M_w)r_w^2\right)\ddot{\phi} &= M_pr_wL\dot{\theta}^2\sin\theta + \frac{k}{R}\left(\tau + k(\dot{\theta} - \dot{\phi})\right),\end{aligned} \tag{E.20a}$$

where the variable definitions are defined in Table E.1. J_p, J_w, and L will be the parameters to estimate, with the initial estimate values $J_p = 0.02$, $J_w = 0.095$, and $L = 0.045$, respectively; the uncertainties of the initial parameter estimates are unbounded. Assuming a gyroscope is mounted on the arm and optical encoder is mounted on the motor, the angular velocities are measured; therefore,

$$\mathbf{y} = \begin{bmatrix}\dot{\theta} \\ \dot{\phi}\end{bmatrix} + \mathbf{v}, \qquad \mathbf{v} \sim N(\mathbf{0}, 0.1\mathbf{I}). \tag{E.20b}$$

The measurement are taken at 100Hz within a time interval of 2 time units. Within the same time, one sine wave of amplitude of 8 is injected as the motor input voltage

for control. The robot is initially at rest with arm hanging down ($\theta = \pi$), thus the initial state values are easily determined; nevertheless they are estimated, but with covariance 0.001 to reflect that they are known fairly accurate.

Table E.1: List of variables in (E.20a).

M_w	Wheel Mass (`kg`)	1.596
M_p	Pendulum (arm) mass (`kg`)	3.724
r_w	Wheel radius (`m`)	0.1267
k	Motor torque constant	0.5428
R	Motor resistance	2.49
g	Gravity (`m/s^2`)	9.81
J_w	Wheel inertia	0.08
J_p	Pendulum (arm) inertia	0.026
L	Distance from wheel axis to center of mass (`m`)	0.055
τ	Voltage applied to motors (control)	$u(t)$

Equation (E.20a) can be further simplified by introducing additional constants as shown in Table E.2, so that in ODE form (E.20a) now looks like

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C_6L^2 + J_p & 0 & C_2L\cos\theta \\ 0 & 0 & 1 & 0 \\ 0 & C_2L\cos\theta & 0 & C_1 + J_w \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ C_3L\sin\theta + C_4(\dot{\phi} - \dot{\theta}) - C_5\tau \\ \dot{\phi} \\ C_2L\dot{\theta}^2\sin\theta + C_4(\dot{\theta} - \dot{\phi}) + C_5\tau \end{bmatrix}. \quad \text{(E.21)}$$

To apply (E.21) into MPDest one must left multiply the inverse of the left hand side of (E.21) symbolically. This example can be loaded from `./example/segway_w_control`.

Table E.2: Additional constants.

C_1	$(M_p + M_w)r_w^2$
C_2	M_pr_w
C_3	M_pg
C_4	k^2/R
C_5	k/R
C_6	M_p

E.3.3 Advecting Field

This example is designed to so that the unknown parameter resides within the measurement function $h(\mathbf{x}_k, \mathbf{p}, \mathbf{u}_k)$. Imagine a wave propagating through a

1D domain, and measurements of that wave at a specific location are taken. The objective here is to estimate the position where the measurements are taken, along with the physical parameters of the wave.

To this end the wave equation with damping is used:

$$\frac{\partial^2 \phi}{\partial t^2} = \frac{\partial^2 \phi}{\partial x^2} - \mu \frac{\partial \phi}{\partial t}, \tag{E.22a}$$

where $\phi(x,t)$ is the wave, x is the positions, and μ is the damping. To make the measurement function continuous with respect to x, it is modeled as Gaussian masking function:

$$y = e^{-b(x-p)^2}\phi + v, \qquad v \sim N(0, 0.001), \tag{E.22b}$$

where b is the attenuation and p is the center of the Gaussian function. Figure E.18 graphically illustrates the measurement function.

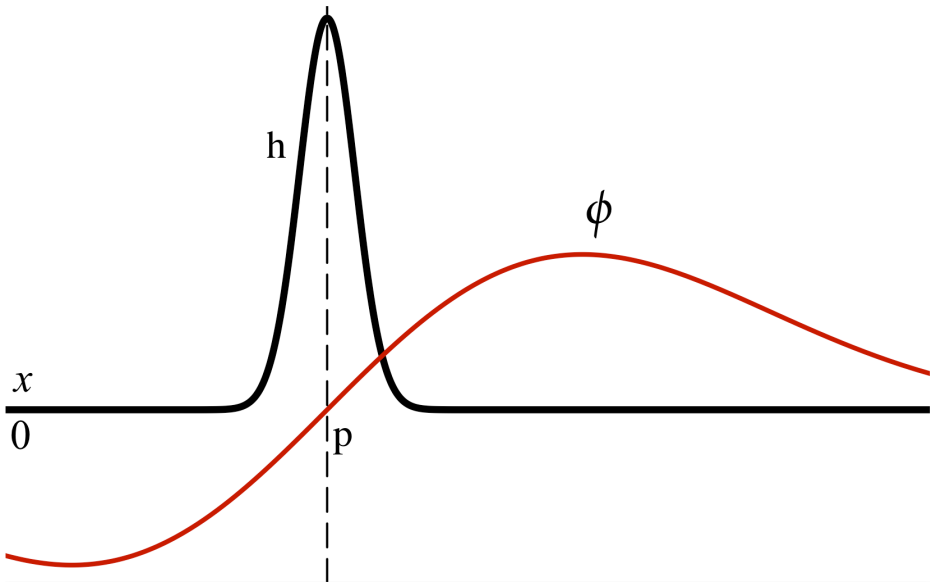


Figure E.18: A graphical illustrate of the measurement function.

In this example (E.22a) is discretized with 1 unit length spacing within a 10 unit length domain; furthermore periodic boundary condition is enforced so that $\phi(0,t) = \phi(10,t)$. Therefore, there are the state size for ϕ is 20 (10 for the position,

10 for the velocity), and the ODE form is

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\boldsymbol{\phi}\\ \dot{\boldsymbol{\phi}}\end{bmatrix} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\\ \mathbf{A} & -\mu\mathbf{I}\end{bmatrix}\begin{bmatrix}\boldsymbol{\phi}\\ \dot{\boldsymbol{\phi}}\end{bmatrix}, \tag{E.23}$$

where $\mathbf{A}$ is a matrix that compute the spacial second derivative using a second order central difference. $\mathbf{A}$ is a tridiagonal circulant Toeplitz matrix with main diagonal of -2 and first upper and lower diagonal of 1. The wave is initially at zero, with unit initial velocity at $x = 2$, and the damping coefficient is $\mu = 0.3$; the initial states are not estimated. A time history of measurements within a 10 time unit interval is sampled at 10Hz with a sensor positioned at $x = 7.5$; the attenuation is $b = 0.5$. In order to simulate real life situations where the measurement covariance is unknown, the measurement covariance is deliberately overestimated to 1. The parameters p, μ, and b are estimated with initial values respectively $p_b = 3$, $\mu_b = 0.5$ and$b_b = 0.7$; their covariances are respectively `inf`, 1, and 1, and their valid value intervals are respectively [0, 10], [0, 1], and [0, 1]. This example can be loaded from `./example/advecting_field`.

Bibliography

Alonso, A.A., Kevrekidis, I.G., Banga, J.R. & Frouzakis, C.E. 2004 Optimal sensor location and reduced order observer design for distributed process systems. *Computers & Chemical Engineering* **28** (1-2), 27–35.

Ancell, B. & Hakim, G. J. 2007 Comparing adjoint- and ensemble-sensitivity analysis with applications to observation targeting. *Monthly Weather Review* **135**, 4117–4134.

Anderson, B.D.O. & Moore, J.B. 1979 *Optimal filtering*. Prentice-Hall Englewood Cliffs, NJ.

Anderson, J. L. 2007 An adaptive covariance inflation error correction algorithm for ensemble filters. *Tellus* **59A** (2), 210–224.

Anderson, J. L. & Anderson, S. L. 1999 A monte carlo implementation of the nonlinear filtering problem to produce ensemble assimilations and forecasts. *Monthly Weather Review* **127**, 2741–2758.

Bagheri, S., Henningson, D.S., Hœpffner, J. & Schmid, P.J. 2009 Input-output analysis and control design applied to a linear model of spatially developing flows. *Applied Mechanics Reviews* **62**, 020803.

Baker, N.L. & Daley, R. 2000 Observation and background adjoint sensitivity in the adaptive observation-targeting problem. *Quarterly Journal of the Royal Meteorological Society* **126** (565), 1431–1454.

Bewley, T.R., Moin, P. & Temam, R. 2001 DNS-based predictive control of turbulence: an optimal benchmark for feedback algorithms. *Journal of Fluid Mechanics* **447**, 179–225.

Bewley, T.R. & Protas, B. 2004 Skin friction and pressure: the "footprints" of turbulence. *Physica D: Nonlinear Phenomena* **196** (1-2), 28–44.

Bishop, C.H. 2000 (submitted) Estimation theory and adaptive observing techniques. *Quarterly Journal of the Royal Meteorological Society* .

Bishop, C.H., Etherton, B.J. & Majumdar, S.J. 2001 Adaptive Sampling with the Ensemble Transform Kalman Filter. Part I: Theoretical Aspects. *Monthly Weather Review* **129** (3), 420–436.

Bucy, R.S. & Joseph, P.D. 1968 *Filtering for stochastic processes with applications to guidance*. John Wiley & Sons.

Buizza, R. & Montani, A. 1999 Targeting Observations Using Singular Vectors. *Journal of the Atmospheric Sciences* **56** (17), 2965–2985.

Bullo, F. & Cortes, J. 2004 Adaptive and distributed coordination algorithms for mobile sensing networks. *Cooperative Control* pp. 431–434.

Burgers, G., Jan van Leeuwen, P. & Evensen, G. 1998 Analysis Scheme in the Ensemble Kalman Filter. *Monthly Weather Review* **126** (6), 1719–1724.

Butala, M.D., Yun, J, Chen, Y, Frazin, R.A. & Kamalabadi, F. 2008 Asymptotic convergence of the ensemble kalman filter. In *ICIP 15th IEEE International Conference on Image Processing*, pp. 825 –828.

Chen, K. & Rowley, C. 2010 Optimal actuator and sensor placement in the linearized complex Ginzburg-Landau system. *Bulletin of the American Physical Society* **55**.

Cortes, J., Martinez, S., Karatas, T. & Bullo, F. 2004 Coverage control for mobile sensing networks. *Robotics and Automation, IEEE Transactions on* **20** (2), 243–255.

Evensen, G. 1994 Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *J. Geophys. Res* **99** (10), 143–10.

Evensen, G. 2003 The Ensemble Kalman Filter: theoretical formulation and practical implementation. *Ocean Dynamics* **53** (4), 343–367.

Faulds, A.L. & King, B.B. 2000 Sensor location in feedback control of partial differential equation systems. In *Control Applications, 2000. Proceedings of the 2000 IEEE International Conference on*, pp. 536–541. IEEE.

Gaspari, G. & Cohn, S.E. 1999 Construction of correlation functions in two and three dimensions. *Quarterly Journal of the Royal Meteorological Society* **125** (554), 723–757.

Giles, M.B. & Pierce, N.A. 2000 An introduction to the adjoint approach to design. *Flow, Turbulence and Combustion* **65** (3), 393–415.

Goodwin, G.C. & Payne, R.L. 1977 *Dynamic system identification: experiment design and data analysis*. Academic Press.

Hamill, T. M., Whitaker, J. S. & Snyder, C. 2001 Distance-dependent filtering of background error covariance estimates in an ensemble kalman filter. *Monthly Weather Review* **129**, 2776–2790.

Horn, R.A. & Johnson, C.R. 1990 *Matrix Analysis*. Cambridge University Press.

Houtekamer, P.L. 1995 The Construction of Optimal Perturbations. *Monthly Weather Review* **123** (9), 2888–2898.

Houtekamer, P.L. & M., Herschel L. 2001 A sequential ensemble kalman filter for atmospheric data assimilation. *Monthly Weather Review* **129**, 123–137.

Jameson, A., Martinelli, L. & Pierce, N.A. 1998 Optimum aerodynamic design using the Navier–Stokes equations. *Theoretical and Computational Fluid Dynamics* **10** (1), 213–237.

Jazwinski, A. H. 1970 *Stochastic processes and filtering theory*. Academic Press.

Kenney, C. & Hewer, G. 1990 The sensitivity of the algebraic and differential Riccati equations. *SIAM Journal on Control and Optimization* **28**, 50.

Khare, S. P. 2004 Observing network design for improved prediction of geophysical fluid flows - analysis of ensemble methods. PhD thesis, Princeton University.

Kwok, A. & Martinez, S. 2010 Unicycle coverage control via hybrid modeling. *Automatic Control, IEEE Transactions on* **55** (2), 528–532.

Langland, R.H. & Rohaly, G.D. 1996 Adjoint-Based Targeting of Observations for FASTEX Cyclones. *Defense Technical Information Center* .

Langland, R.H., Toth, Z., Gelaro, R., Szunyogh, I., Shapiro, M.A., Majumdar, S.J., Morss, R.E., Rohaly, G.D., Velden, C., Bond, N. *et al.* 1999 The North Pacific Experiment (NORPEX-98): Targeted Observations for Improved North American Weather Forecasts. *Bulletin of the American Meteorological Society* **80** (7), 1363–1384.

Laventall, K. & Cortés, J. 2009 Coverage control by multi-robot networks with limited-range anisotropic sensory. *International Journal of Control* **82**, 1113–1121.

Lermusiaux, P.F.J. 2007 Adaptive modeling, adaptive data assimilation and adaptive sampling. *Physica D: Nonlinear Phenomena* **230** (1-2), 172–196.

Livings, D. M., Dance, S. L. & Nichols, N. K. 2008 Unbiased ensemble square root filters. *Physica D: Nonlinear Phenomena* **237** (8), 1021–1028.

Majumdar, S. J., Bishop, Craig H., Etherton, B. J. & Toth, Z. 2002 Adaptive Sampling with the Ensemble Transform Kalman Filter. Part II: Field Program Implementation. *Monthly Weather Review* **130**, 1356–1369.

Martínez, S. & Bullo, F. 2006 Optimal sensor placement and motion coordination for target tracking. *Automatica* **42** (4), 661–668.

Martínez, S., Cortés, J. & Bullo, F. 2007 Motion coordination with distributed information. *IEEE Control Systems Magazine* **27** (4), 75–88.

Petersen, K. B. & Pedersen, M. S. 2008 The matrix cookbook. Technical University of Denmark.

Petrie, R.E. & Dance, S.L. 2010 Ensemble-based data assimilation and the localisation problem. *Weather* **65** (3), 65–69.

Porat, B. & Nehorai, A. 1996 Localizing vapor-emitting sources by moving sensors. *Signal Processing, IEEE Transactions on* **44** (4), 1018 –1021.

Smith, M.W.A. & Robers, A.P. 1978 An exact equivalence between the discrete- and continuous-time formulations of the kalman filter. *Mathematics and Computers in Simulations* **20**, 102–109.

Snyder, C. 1996 Summary of an informal workshop on adaptive observations and FASTEX. *Bull. Amer. Meteor. Soc* **77** (5), 953–961.

Stanković, M. S. & Stipanović, D. M. 2009 Stochastic extremum seeking with applications to mobile sensor networks. In *ACC'09: Proceedings of the 2009 conference on American Control Conference*, pp. 5622–5627. Piscataway, NJ, USA: IEEE Press.

Szunyogh, I., Toth, Z., Morss, R.E., Majumdar, S.J., Etherton, B.J. & Bishop, C.H. 2000 The Effect of Targeted Dropsonde Observations during the 1999 Winter Storm Reconnaissance Program. *Monthly Weather Review* **128** (10), 3520–3537.

Taylor, J. 1979 The Cramér-Rao Estimation Error Lower Bound Computation for Deterministic Nonlinear Systems. *Automatic Control, IEEE Transactions on* **24** (2), 343–344.

Tippett, M. K., Anderson, J. L., Bishop, G. H., Hamill, T. M. & Whitaker, J. S. 2003 Ensemble square root filters. *Monthly Weather Review* **131**, 1485–1490.

Toth, Z. & Kalnay, E. 1993 Ensemble Forecasting at NMC: The Generation of Perturbations. *Bulletin of the American Meteorological Society* **74** (12), 2317–2330.

Uciński, D. 2005 *Optimal Measurement Methods for Distributed Parameter System Identification*. CRC Press.

Wang, X., Bishop, C. H. & Julier, S. J. 2004 Which is better, an ensemble of positive-negative pairs or a centered spherical simplex ensemble? *Monthly Weather Review* **132** (7), 1590–1605.

Whitaker, J.S. & Hamill, T.M. 2002 Ensemble Data Assimilation without Perturbed Observations. *Monthly Weather Review* **130** (7), 1913–1924.

Yilmaz, N. K., Evangelinos, C., Lermusiaux, P. F.J. & Patrikalaks, N. M. 2008 Path planning of autonomous underwater vehicles for adaptive sampling using mixed integer linear programming. *IEEE Journal of Oceanic Engineering* **33** (4), 522–537.

Zhang, D., Colburn, C.H. & Bewley, T.R. 2011 Estimation and Adaptive Observation of Environmental Plumes. In *American Control Conference (ACC), 2011*, pp. 000–000. IEEE.

Zhang, F. & Leonard, N. E. 2010 Cooperative filters and control for cooperative exploration. *IEEE Transactions on Automatic Control* **55** (3), 650–663.