Hierarchical Gaussian models for wind field estimation and path planning

by

Antoni M. Musolas Otaño

Eng. Camins, Canals i Ports, UPC (2013)
Eng. Mines, UPC (2013)

Submitted to the Center for Computational Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Computation for Design and Optimization

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2016

Signature redacted

Author .....
Center for Computational Engineering
April 29, 2016

Signature redacted

Certified by .....
Youssef M. Marzouk
Associate Professor of Aeronautics and Astronautics
Thesis Supervisor

Signature redacted

Accepted by .....
Nicolas Hadjiconstantinou
Professor of Mechanical Engineering
Co-Director, Computation for Design and Optimization

Hierarchical Gaussian models for wind field estimation and path planning

by

Antoni M. Musolas Otaño

Submitted to the Center for Computational Engineering
on April 29, 2016, in partial fulfillment of the
requirements for the degree of
Master of Science in Computation for Design and Optimization

Abstract

Improvements in technology, autonomy, and positioning mechanisms have greatly broadened the range of application of unmanned aerial vehicles. These vehicles are now being used in aerial photography, package delivery, infrastructure inspection, and many other areas. Many of these uses demand new techniques for path planning in complex environments—in particular, spatially heterogeneous and time-evolving wind fields [22, 23, 24]. Navigating and planning [26, 25, 28, 12] in wind fields requires reliable and *fast* predictive models that quantify uncertainty in future wind velocities, and benefits strongly from the ability to incorporate onboard and external wind field measurements *in real time*.

To make real-time inference and prediction possible, we construct simple hierarchical Gaussian models of the wind field as follows. Given realizations of the wind field over a domain of interest, obtained from detailed offline measurements or computational fluid dynamic simulations, we extract empirical estimates of the mean and covariance functions. The associated covariance matrices are anisotropic and non-stationary, and capture interactions among the wind vectors at all points in a discretization of the domain. We make the further assumption that, given a particular prevailing wind heading, the local wind velocities are jointly Gaussian. The result is a hierarchical Gaussian model in which the mean and covariance are functions of the prevailing wind conditions. Since these empirical covariances are known only for a few prevailing wind conditions, we close our model by interpolating covariance matrices on the appropriate manifold of positive semi-definite matrices [44], via a computationally efficient construction that takes advantage of low-rank structure.

Finally, assimilation of successive point observations is conducted by embedding a standard Kalman filter within a hierarchical Bayesian inference framework. This representation will then be used for wind field exploitation.

Thesis Supervisor: Youssef M. Marzouk
Title: Associate Professor of Aeronautics and Astronautics

Acknowledgments

This research project would not have been possible without the support of several people.

First and foremost, I would like to thank "la Caixa", my main sponsor, for providing a full scholarship to study in the US. It has constituted an unforgettable experience and will certainly have a wide impact in my future career.

I would like to express my deepest gratitude to my advisor Prof. Youssef Marzouk, who has offered superior guidance and unconditional support. Our discussions in a range of different topics have made me grow not only academically but also as a person.

This thesis was enriched significantly through the comments, answers, and suggestions of Prof. Nicholas Roy and Prof. P.-A. Absil. I would also like to thank deeply to my colleagues in the lab; to name a few, Alessio, Angxiu, Ferran, Jake, Lucio, Mario, Pablo, Patrick, Remi, TC, Zheng...

Last but not least, I wish to express my love to my family and friends; for their understanding and endless support through the duration of my studies. Finally, I would also like to say thanks to every single person who has contributed to increase my passion towards both academia and industry.

To all of them,

Thank you!

Antoni M. Musolas Otaño

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

During the second half of the 20th century, the decision-making process with regards to optimization evolved. First, computational modelling allowed predicting outputs in physical processes and then this approach shifted from a deterministic overview to a probabilistic one. In a second stage, only decisions of designs in which safety or precision were crucial were computed by means of models that account for their uncertainties. Nowadays, most models are being computed stochastically. The combination between computation and stochasticity may allow a better understanding of complex systems before making decisions related to process and design.

Nowadays a wide range of applications require dealing with uncertainty by means of random variables. Most models use Gaussian distributions as underlying variables, both for its simplicity and its capacity to represent experimental data. However, these models usually turn out to be high dimensional, and it is common to be confronted with problems that are 10^6, 10^9 dimensional or even more [32]. Apart from their mean value, Gaussian distributions are defined by covariance or precision matrices. In high dimensional cases, these matrices cannot be even stored. Therefore, it is mandatory to take advantage of their sparsity patterns to deal with them. In particular, usage of iterative methods is in order due to the inexpensive cost per iteration and the small memory requirements as opposed to direct methods that typically include matrix

factorizations, which becomes prohibitively expensive for dimensions greater than 10^5.

One of such applications that combines numerical modeling and sthocasticity is wind field navigation. Improvements in technology, autonomy, and positioning mechanisms are opening the door to more sophisticated and capable unmanned aerial vehicles (UAVs) that fly over uncertain but predictable to some extend wind conditions. As a result of the research on this topic, regulation is evolving and becoming more permissible on UAVs flying in urban environments. At the same time, industry is forecasting huge saving by incorporating these vehicles on their operations. Society is also in the process of accepting UAVs flying in urban environments. These recent developments have greatly broadened the range of application of UAVs. Indeed, these vehicles are now being used in aerial photography, package delivery, infrastructure inspection, and many other areas. But what do the regulators, industry, and society demand? Primarily the aforementioned stakeholders demand the following three characteristics:

1. Safety, any potential risk may become a threat to society.
2. Reliability, the UAV must guarantee arrival to the destination.
3. Efficient paths, need to minimize energy consumption and/or time to become competitive.

In turn, these three depend on the external wind conditions, so there is a clear need for understanding UAV navigation in complex wind fields.

Navigating and planning in wind fields require reliable predictive models that quantify uncertainty in future wind velocities, while enabling inference from onboard and external wind field measurements in real time. These data-informed predictions feed path planning algorithms that aim to minimize energy consumption or flying time. New techniques for path planning in urban environments have to be envisioned, in particular, spatially heterogeneous and time-evolving wind fields. The problem can be cast as path planning under uncertainty, where the urban canopy imparts some predictability on the wind field but the rapid changes of the wind due to its low viscosity makes such prediction to be a challenge.

1.2 Statement of the problem

1.2.1 General problem

The general problem that we aim to solve can be phrased as follows: given a domain (e.g., Figure 1-1a), improve the navigation of a UAV by taking into account the external wind conditions. To solve this problem, we will characterize the wind field first with offline data, and then we will incorporate any source of online measurements using inference techniques.

Before even starting to devise a solution for the problem, we have to bear in mind that the requirements of the proposed procedure shall be:

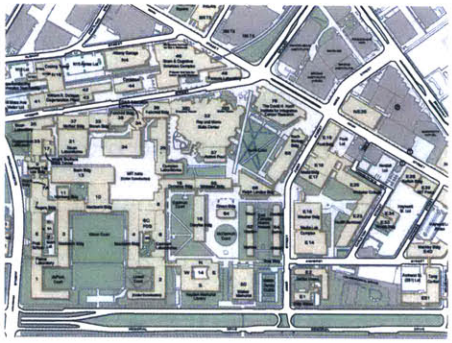
- Methodology must be scalable and robust.
- Solution must be fast to obtain and recomputable online.
- Algorithms must be efficient in terms of data storage.

1.2.2 Problem of interest

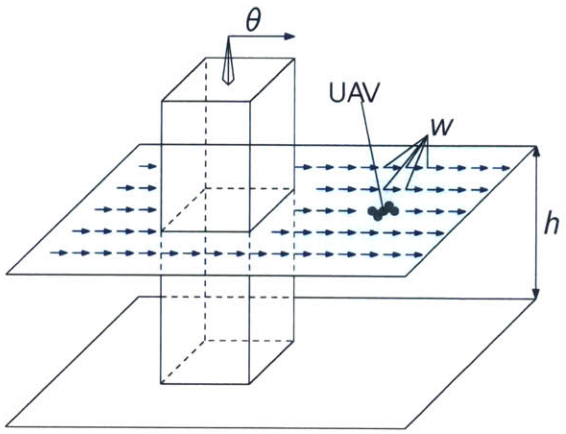
Instead of solving a general domain, we will work on a *toy problem* to focus on the technicalities of modelling rather than in the nuances of problem-specific characteristics. We claim that our solution can be extrapolated to more complex domains.

Geometrically speaking, the domain that will be analyzed is based on a 2D domain of 100×100 meters of side with an inner square building of 10×10 meters. The discretization has dimensionality of 40×40. As will be introduced in section 3.1.2, we will characterize the wind field by building a covariance matrix of the wind velocity components. These matrices will be of around 3200×3200 elements, one row and column for each components of the wind and for each element of the discretization.

Our wind field (Figure 1-1b) will be characterized by its external prevailing wind heading (θ), which is measured in the top of the building. In the 2D representation, we illustrate the heading with a blue arrow in the middle of the obstacle (Figure 1-3a).



(a) Example of the domain for a general problem



(b) Illustration of the problem of interest.

Figure 1-1: Comparison between an example of general domain (MIT main campus) and our problem of interest

In terms of offline data (i.e., precomputations before UAV takes off), we will rely an unsteady computational fluid dynamics (CFD) solver. Online measurements will come from a wind station that measures the external prevailing wind conditions θ. We will measure also the wind velocity in the location of the UAV, that is, the UAV will have an in-built anemometer.

1.3 Previous work

Spatially heterogeneous and time-evolving fields have been broadly studied. Not only the case of *wind* fields (e.g., [22, 23, 24]) is nowadays on the crest of the wave but also similar to some extent is the navigation in the ocean with strong currents [28, 12]. Navigating and planning [26, 25] in fields requires reliable and fast predictive models that quantify uncertainty in future wind velocities, and benefits strongly from the ability to incorporate onboard and external wind field measurements in real time. However, the larger the scale of the domain and the lower the viscosity of the media, the more often we can observe sudden changes in the velocity conditions. In the particular case of wind, the conditions are so volatile that considering only the mean can be not sufficient, and in some cases, even not meaningful. Figure 1-4 illustrates

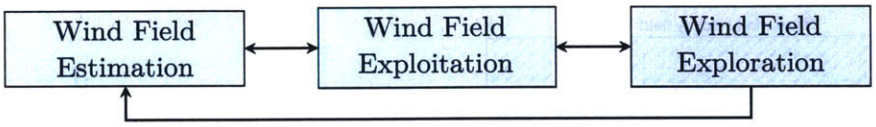


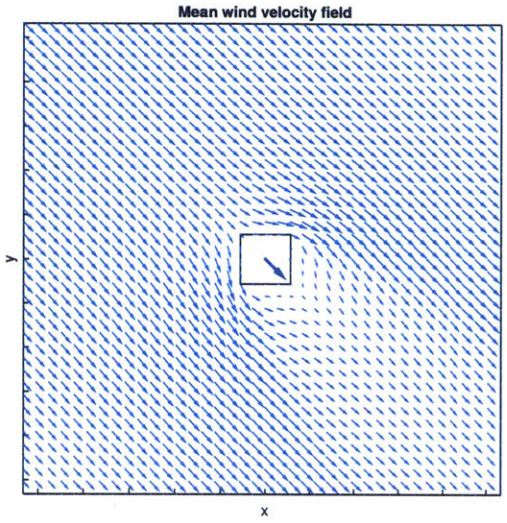
Figure 1-2: Literature division of the wind field navigation problem.

the wind field in different time instances for the exact same external conditions. It is not difficult to notice that these snapshots may differ substantially from the mean (Figure 1-3a).

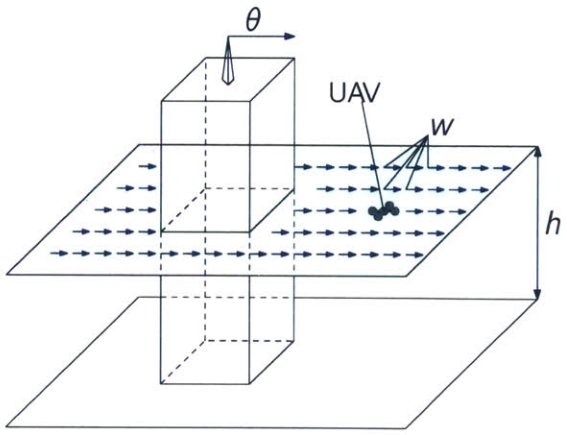
In the literature, the problem of wind field navigation is typically divided (see Figure 1-2) as follows:

- *Estimation* consists of characterizing the wind field with offline data and incorporating any source of online measurements (e.g., [23, 24]).
- *Exploitation* consists of using the wind field to improve planning in terms of efficiency, reliability, or safety (e.g., [25, 28]).
- *Exploration* consists of navigating with the primary goal of learning more features of the wind field online (e.g., [26, 12]).

Although the aforementioned volatility of the wind conditions, most of the solutions in the literature consider only the steady state solution for wind field estimation, and then perform wind field exploitation over it.



(a) 2D view of the problem of interest, mean wind velocity field.



(b) Illustration of the problem of interest (bis).

Figure 1-3: Comparison between a 2D representation of the problem of interest and a 3D view.

In the present study, we investigate the possibility to work with more accurate representations that deal not only with the mean wind velocity field, but also with its spatial covariance. Thus, given a domain and a fixed external conditions, we account for *uncertainty* by estimating the mean and covariance of an unsteady CFD solution.

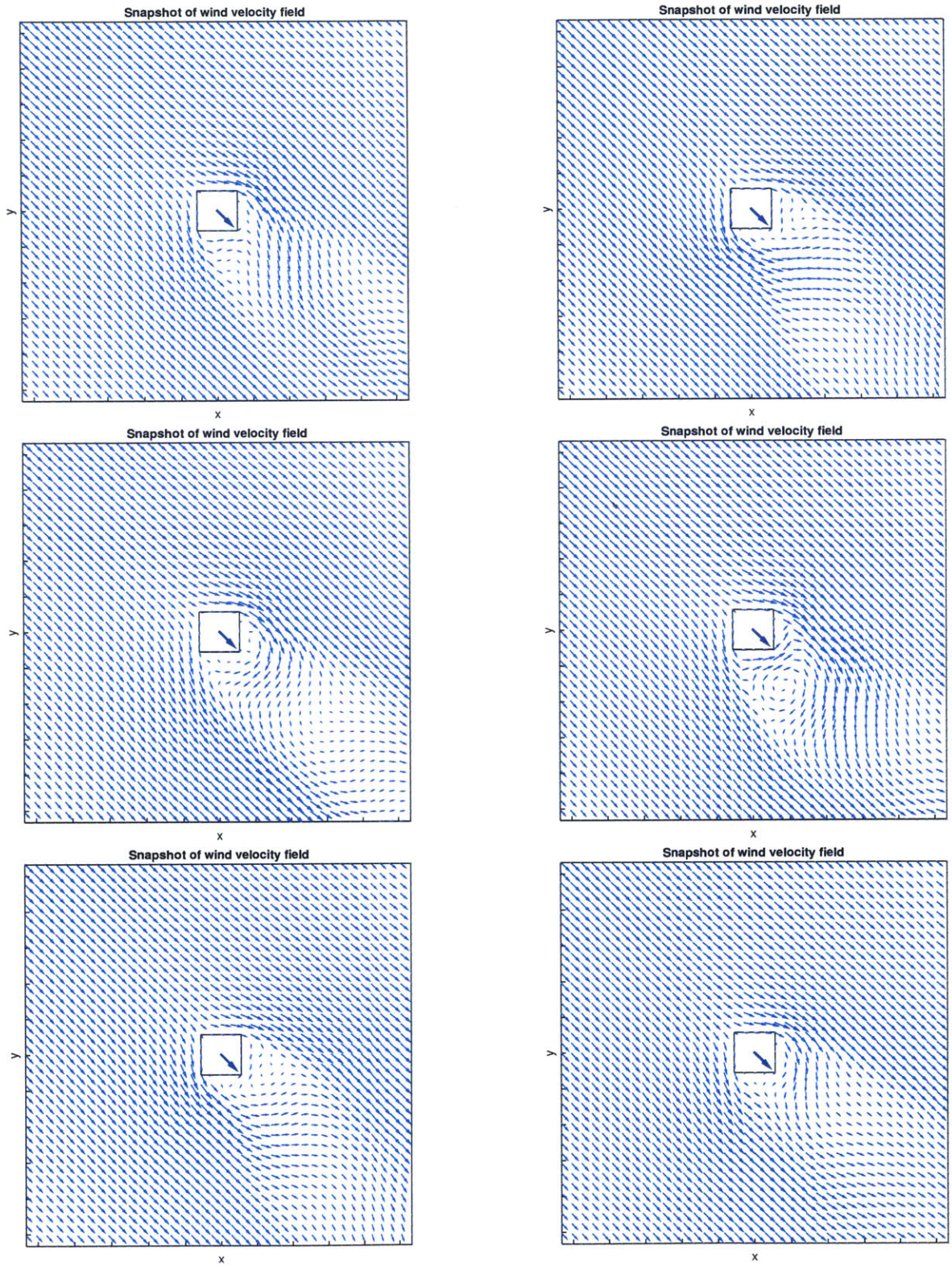
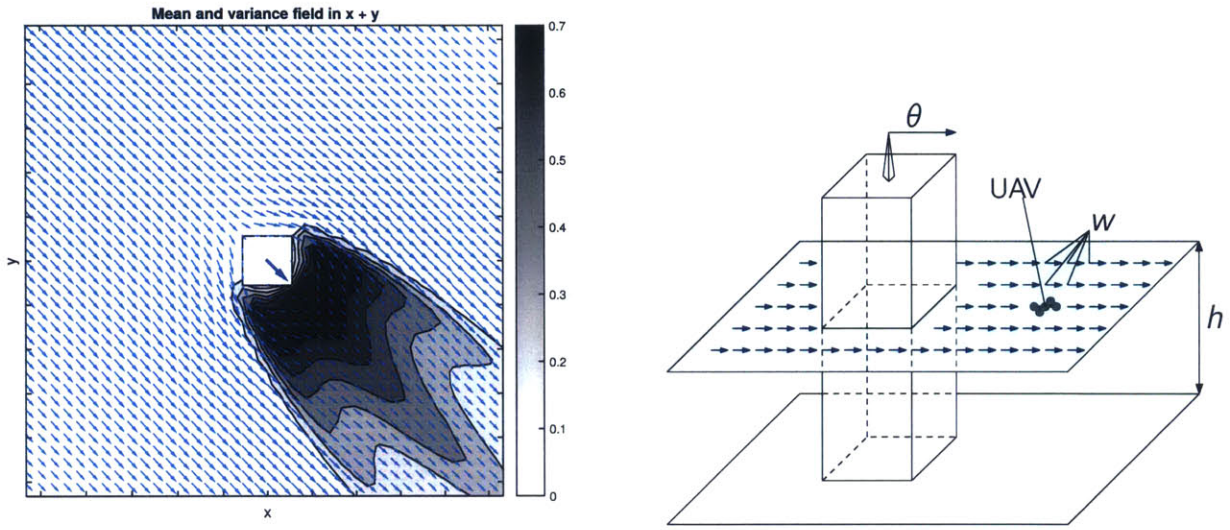


Figure 1-4: Snapshots of an unsteady state solution of the wind velocity field in the problem of interest.



(a) 2D view of the problem of interest, mean and variance velocity field.

(b) Illustration of the problem of interest (bis).

Figure 1-5: Mean and variance field of the wind velocity field in the problem of interest.

1.4 Thesis objectives and organization

The problem of path planning under uncertainty involves two steps: (i) characterizing the uncertainty of the domain of interest and (ii) devising an efficient algorithm that can deal with such uncertainty to find an optimal path. The project focuses on the modeling and inference challenges of wind field estimation by constructing spatiotemporal Gaussian process (GP) representations of complex wind fields, with anisotropic and non-stationary covariances. These representations can then be used for filtering with autoregressive dynamics or direct conditioning. The main goal is to apply our formulation on a simple model navigation problem in an urban wind environment, and evaluate whether planning that takes advantage of wind field modeling and conditioning can outperform more naïve path planning schemes.

First, we make the assumption that, for a given prevailing wind, the domain of interest has a wind field that can be approximated as a Gaussian random field (GRF). Second, since a GRF is continuous over the domain, we discretize it in a regular (or not) grid. Then, we use an unsteady CFD solver to simulate the domain and to obtain

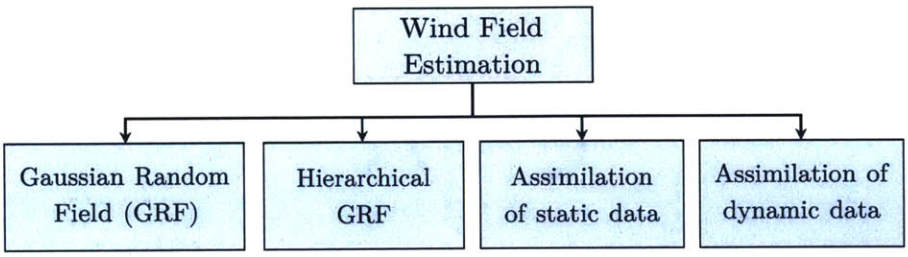


Figure 1-6: Organization of the proposed solution for wind field estimation

a bunch of simulations over time corresponding to a fixed prevailing wind. Forth, we make an ergodicity assumption (i.e., the wind field at a particular time has the same distribution as an ensemble of snapshots acquired over time). Finally, we compute the mean and covariance matrix of the grid points across all the CFD solutions, namely, the *statistics* of the wind field.

To be more specific, the mean refers to the pointwise mean of each component of the velocity. The covariance matrix contains the pointwise variance of each component of the velocity, the covariance of each component of the velocity with the others in the same point in the grid, and the covariance of each component of the velocity with respect to all the other points in the grid.

These statistics will be used later to (i) assimilate static and dynamic data, (ii) obtain approximations of the same statistics corresponding to conditions of prevailing wind for which we do not have data, (iii) update the wind field over time, (iv) perform stochastic path planning.

The project is organized as follows. In Chapter 2, the mathematical building blocks are reviewed. In particular, we review basic concepts of Gaussian random fields and its sampling techniques, as well as Bayesian filtering, and succinct introductions to differential geometry and dynamic programming. In Chapter 3, we describe our wind field estimation approach and how to apply it to wind field exploitation. We describe also the integration of both, wind field estimation and exploitation, in Section 3.5. Numerical results are shown in Chapter 4. After having presented these results, conclusions, limitations, and future work are discussed in Chapter 5.

Chapter 2

Mathematical building blocks

2.1 Gaussian stochastic processes and random fields

2.1.1 Preliminary definitions

A stochastic process [13] is a collection of random variables $\{X_t\}_{t\geq 0}$ defined on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$. That is, for each time $t \geq 0$, $\omega \mapsto X_t(\omega)$ is a random variable.

A stochastic process $X \equiv \{X_t\}_{t\geq 0}$ can be seen as a function that assigns a real number for each real $t \geq 0$ and each $\omega \in \Omega$:

$$\mathbb{R}_+ \times \Omega \to \mathbb{R},$$
$$(t, \omega) \mapsto X_t(\omega).$$

A GP [33] is a stochastic process for which any finite subcollection of $\{X_t\}_{t\geq 0}$ has a joint Gaussian distribution. In terms of notation, we refer to $X_t \sim GP(m, C)$, meaning that X is distributed as a GP with mean function $m(t)$ and covariance kernel $C(t, t')$.

By a joint Gaussian distribution we refer to one that has density defined as follows:

$$f_x(x_1, \ldots, x_k) = \frac{1}{\sqrt{(2\pi)^k |\Sigma|}} \exp\Big(-\frac{1}{2}(x-\mu)^{\mathrm{T}} \Sigma^{-1} (x-\mu)\Big), \tag{2.1}$$

where x and the mean μ are real k-dimensional column vectors and $|\Sigma|$ is the determinant of the covariance matrix Σ. Notice that Σ is a discretization for some values of t of a much more general $C(t, t')$. A random field $\{F_t : t \in T\}$ is a generalization of a stochastic process such that the underlying parameter no longer needs to be a simple real, but can take values that are multidimensional vectors instead. In the same way we defined GP, a GRF is a random field with Gaussian probability density function.

If μ and Σ are functions of certain scalar (or vector) parameter θ, we say that the GP (or GRF) defined above is a *hierarchical* GP (or GRF).

A similar object to the GRF is the Gaussian Markov random field (GMRF). To introduce the notion of GMRF [35] we need first to state the definition of a graph and, in particular, an indirected graph. A graph is an ordered pair $G = (V, E)$ comprising a set V of vertices or nodes together with a set E of edges or links, which are 2-element subsets of V. In other words, an edge is related with two vertices, and the relation is represented as an unordered pair of the vertices with respect to the particular edge.

A GMRF with respect to a graph $G = (V, E)$ is a multivariate normal distribution in which the missing edges correspond to zeros on the precision matrix (Σ^{-1}):

$$X = (X_v)_{v \in V} \sim \mathcal{N}(\mu, \Sigma),$$

where:

$$(\Sigma^{-1})_{uv} = 0 \quad \text{if} \quad \{u, v\} \notin E.$$

2.1.2 Direct-conditioning of Gaussian distributions

We can manipulate the covariance matrix such that we condition the GRF or GMRF to some particular observations in some particular locations. To do that, we introduce the notion of conditional covariance matrix for the multivariate Gaussian Random variable. If μ and Σ are partitioned as follows:

$$\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix},$$

$$\Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix};$$

then, the distribution of x_1 conditional on $x_2 = a$ is multivariate normal $(x_1|x_2 = a) \sim N(\bar{\mu}, \Sigma)$ where:

$$\bar{\mu} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(a - \mu_2),$$

and covariance matrix:

$$\overline{\Sigma} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}.$$

2.1.3 Gaussian mixture model

Given some random variables with distribution f_i and some weights p_i, a mixture is defined as:

$$f(x) = \sum_i p_i f_i(x).$$

A Gaussian mixture corresponds to the case when f_i are of the shape (2.1), i.e., corresponding to Gaussian random variables.

Denoting $\mu^{(k)}$ for the k^{th} moment of f and $\mu_i^{(k)}$ for the k^{th} moment of f_i, it follows immediately from the definition that

$$\mu^{(k)} = \mathbb{E}_f[x^k] = \sum_i p_i \mathbb{E}_{f_i}[x^k] = \sum_i p_i \mu_i^{(k)}.$$

For instance, the mean can be written as:

$$\mathbb{E}_f[x] = \sum p_i \mu_i^{(1)}).$$

Also, the variance reads:

$$\mathrm{Var}(f) = \mu^{(2)} - (\mu^{(1)})^2 = \sum p_i \mu_i^{(2)} - \left(\sum p_i \mu_i^{(1)}\right)^2.$$

2.2 Sampling Gaussian distributions

Sampling becomes really useful for understanding high-level interactions between variables, in particular for the case of multivariate Gaussian random variables representing some properties in a defined space. This is always the case in spatial statistics (see [35]). When data can be represented in a one, two or three dimensional space, observing the interactions of the desired property according to the position may be particularly useful. Examples of the above are wind field models, climate models, weather forecasting or petroleum reservoir predictions that rely on the propagation of samples over time [5]. The entries in the mean value are easily understood since they typically represent mode values of the properties under study. However, the key issue in these models is that entries in either the covariance or precision matrices are difficult to assess. Sampling these fields and analyzing different samples is an excellent way to understand their behavior.

The most common way to sample Gaussian distributions is by means of Cholesky factorization. The method is based basically in a LL^T factorization, which is always seen as an expensive approach. On the other side, since it is a direct method the samples it produces are exact and, once the factorization is done, drawing samples requires only one matrix vector product. Another advantage of Cholesky factorization is that, as it will be presented, it can draw samples from a covariance and precision matrices indistinctly. For large dimensional Gaussian distributions, the conventional technique is Gibbs sampling. This method is powerful thanks to its inexpensive cost per iteration and small memory requirements, especially if the precision matrix is sparse.

In this section we review Cholesky factorization and Gibbs sampling since they are the traditional techniques for drawing samples. After that, we study the sampling

techniques described in [32] and [17] as they are the cutting edge iterative samplers. The first one produces a Gaussian realization using a covariance matrix that is an approximation to the desired covariance in a smaller dimensional Krylov space and finds samples using conjugate gradients. As we will see, the procedure allows working with either precision or covariance matrices. The second sampler uses Chebyshev polynomial acceleration for the stochastic iteration to accelerate convergence in the mean and covariance. Both algorithms do not require explicit storage of covariance or precision, but they need the matrix vector product of it instead. For each technique, we describe its usage, algorithm, and order of operations. It is not the intention of this review to prove the convergence of the sampling algorithms, some references are given to this respect.

2.2.1 Cholesky factorization

Cholesky factorization is the conventional method to sample from Gaussian distributions [42]. The samples it produces are exact and that is what makes this algorithm so attractive. The method can draw samples from either a covariance (Algorithm 1a) or precision matrix (Algorithm 1b). Once the covariance or precision matrix is factorized, sampling from them requires only a triangular matrix vector product or a backsubstitution [34], respectively.

Algorithm 1a: Cholesky factorization sampler from $N(0, A)$

---

input: n x n symmetric positive definite matrix A,

output: c exactly distributed as $N(0, A)$

1. sample $z \sim N(0, I)$;
2. $A = CC^T$; Cholesky factorization
3. $c = Cz$; triangular matrix vector product

Algorithm 1b: Cholesky factorization sampler from $N(0, A^{-1})$

input: n x n symmetric positive definite matrix A,

output: y exactly distributed as $N(0, A^{-1})$

1. sample $z \sim N(0, I)$;
2. $A = CC^T$; Cholesky factorization
3. $z = C^T y$; Triangular system of equations, solved by backsubstitution

Both Algorithm 1a and Algorithm 1b are exact and impressively simple. The only drawback of the method is that factorizing a matrix takes $(1/3)n^3$ operations. If A has bandwidth b, Cholesky factorization requires $\mathcal{O}(b^2 n)$ operations.

Recall that LL^T factorization is only permitted in symmetric positive definite (or semi-definite) matrices, but this is not an issue since covariance and precision matrices satisfy this condition by construction.

2.2.2 Gibbs sampling

For large linear systems, Cholesky factorization may not be feasible, and less demanding techniques, both in terms of operations and memory, should be used. Perhaps the most popular iterative method to sample from a Gaussian distribution is Gibbs sampler (e.g., [11], [18], [21]).

As described in Algorithm 2, the idea behind Gibbs sampler is sampling each component of the multivariate vector conditioning to the value of the other components. By iterating within this procedure, the sample is guaranteed to converge to a sample from $N(0, A^{-1})$.

Algorithm 2: Gibbs sampler from $N(0, A^{-1})$

input: n x n symmetric positive definite matrix A, initial state y^0, and k_{max}

output: $y^{k_{max}}$ approximately distributed as $N(0, A^{-1})$

1. **for** $k = 0, ..., k_{max}$ **do**
    1.1. **for** $i = 1, ..., n$ **do**
        1.1.1. sample $z \sim N(-A_{ii}^{-1} A_{i,-i}(y_{-i}), A_{ii}^{-1})$;
        1.1.2. $y_i = z$;
    1.2. **end**
2. **end**

In Algorithm 2, $A_{i,-i}$ refers to all the elements in row i of the matrix A except the one corresponding to column $-i$. Also, notice that line in 1.1.1. z is an scalar and its variance is A_{ii}^{-1}. The sampling of z is as trivial as sampling a scalar of $N(0, 1)$, multiplying $\sqrt{A_{ii}^{-1}}$ and adding its mean $(-A_{ii}^{-1} A_{i,-i} y_{-i})$.

Although it exists a version of Algorithm 2 that uses the covariance matrix instead, the algorithm is remarkably more efficient for sampling a precision matrix because it takes full advantage of its sparsity.

2.2.3 Other sampling techniques

Sampling in Krylov spaces with conjugate gradients

The idea behind this algorithm is to produce a Gaussian realization using a covariance matrix that is an approximation to the desired covariance in a smaller dimensional Krylov space [32, 39]. As we will see, the procedure allows working with either covariance (Algorithm 3a) or precision (Algorithm 3b) matrices. Not surprisingly, comparing both algorithm we notice that there is no difference apart from line 5.3. Algorithm 3a requires an additional matrix vector product but it is exactly the same than in line 5.4, which therefore is computed in both algorithms only once. A discussion of this algorithm together with its proof of convergence is presented in [32].

The conjugate gradient algorithm to solve systems of equations takes $2n^2$ operations per iteration for a full matrix [31]. Since the conjugate gradient sampler is a slight

modification (adding line 5.3 in Algorithm 3a and 3b), it only requires around $2n$ additional operations [32].

Notice that this algorithm does not require any kind of precomputation. It neither requires building any matrix, it is enough with the matrix vector product of the covariance/precision matrix. As discussed above, the precision matrix tends to be sparse, which makes a vectorization of its matrix vector product feasible. These facts make this algorithm impressively versatile. Moreover, the conjugate gradient algorithm is famous for its extraordinary performance on solving linear systems of equations, and this is a quality to take into consideration when using the conjugate gradient as a sampling technique.

Algorithm 3a: Conjugate gradient sampler from $N(0, A)$

input: Given n x 1 vectors b and x^0, and an n x n
symmetric positive definite matrix A,
specify some stopping tolerance ϵ. and k_{max}

output: $c^{k_{max}}$ approximately distributed as $N(0, A)$

1. $r^0 = b - Ax^0$;
2. $p^0 = r^0$;
3. $d_0 = p^{(0)T} A p^0$;
4. $c^0 = x^0$;
5. **for** $k = 0, ..., k_{max}$ **do**
   - 5.1. $\gamma_{k-1} = \frac{r^{(k-1)T} r^{k-1}}{d_{k-1}}$.
   - 5.2. $x^k = x^{k-1} + \gamma_{k-1} p^{k-1}$.
   - 5.3. *Sample* $z \sim N(0, 1)$, *and set* $c^k = c^{k-1} + \frac{z}{\sqrt{d_{k-1}}} A p^{k-1}$.
   - 5.4. $r^k = -\nabla_x \phi(x^k) = r^{k-1} - \gamma_{k-1} A p^{k-1}$ *is the residual.*
   - 5.5. $\beta_k = -\frac{r^{kT} r^k}{r^{(k-1)T} r^{k-1}}$.
   - 5.6. $p^k = r^k - \beta_k p^{k-1}$ *is the next conjugate search direction.*
   - 5.7. $d_k = p^{(k)T} A p^k$.
   - 5.8. *Break if* $||r^k||_2 < \epsilon$.
6. **end**

Algorithm 3b: Conjugate gradient sampler from $N(0, A^{-1})$

Refer to Algorithm 3a and modify:

- 5.3. *Sample* $z \sim N(0, 1)$, *and set* $y^k = y^{k-1} + \frac{z}{\sqrt{d_{k-1}}} p^{k-1}$.

Chebyshev accelerated SSOR sampler

Finally, we mention without presentation the Chebyshev accelerated SSOR sampler [17]. As its name indicates, the algorithm uses Chebyshev polynomial acceleration [36] for the stochastic iteration to accelerate convergence in the mean and covariance. The

algorithm requires only matrix vector product and the full ensemble of the matrix is not necessary. However, it also requires matrix splitting of successive over-relaxation (SOR) and symmetric successive over-relaxation (SSOR) [29, 1, 19]. The most expensive computations of the algorithm are solving two triangular systems of equations: one backsubstitution and one forward substitution per iteration. For a general matrix A, solving these systems of equations may require around n^2 operations each. Another strong requirement of this algorithm is precomputing the extremum eigenvalues, for which Lanczos algorithm shall be a fast approach.

2.3 Bayesian perspective on filtering

In this section, we introduce a Bayesian perspective on filtering. We use the notation and structure in Chapter 4 of [38]. Filtering consists of recursively updating the estimation of a dynamic system as we acquire more information. As we will see, this technique will then be applied in our wind field estimation framework. Our states will be the components of the velocity of the wind at each point and the new information will be the measurements of the wind that we get from the UAV.

2.3.1 Introduction to filtering

Let $x_k \in \mathbb{R}^n$ be the state of a dynamic system of interest at step $k = 1, 2, \ldots$ and $y_k \in \mathbb{R}^m$ the measurement at k. A probabilistic state space model can be defined as a sequence of conditional probability distributions $x_k \sim p(x_k \mid x_{k-1})$, together with a measurement model of the form: $p(y_k \mid x_k)$. The first equation describes the stochastic dynamics of the system and the second explains the distribution of the measurements conditional to the state.

In order to reduce the computational cost and make the recursion have a close form, we need to assume two Markovian properties. First, the state x_k given x_{k-1} is independent of the previous states and measurements. Secondly, the current measurement y_k given the current state x_k is conditionally independent of the measurement and state histories.

The Bayesian filter operates in the following way. Firstly, the recursion starts from the prior distribution $p(x_0)$. The predictive distribution of x_k, given the dynamic model, is computed as:

$$p(x_k \mid y_{1:k-1}) = \int p(x_k \mid x_{k-1})\, p(x_{k-1} \mid y_{1:k-1})\, \mathrm{d}x_{k-1}.$$

Finally, the posterior distribution of x_k conditioned on y_k is computed as:

$$p(x_k \mid y_{1:k}) = \frac{1}{Z_k} p(y_k \mid x_k)\, p(x_k \mid y_{1:k-1}),$$

where Z_k is the normalization constant.

2.3.2 Kalman filter

The Kalman filter (KF) is the closed form solution to the Bayesian filtering equations for the filtering model where the dynamic and measurement models are linear Gaussian.

Using the notation introduced before, the model reads as:

$$\begin{aligned} x_k &= A_{k-1} x_{k-1} + q_{k-1}, \\ y_k &= H_k x_k + r_k, \end{aligned} \tag{2.2}$$

where $q_{k-1} \sim N(0, R_k)$ is the process noise, and the prior distribution is Gaussian $x_0 \sim N(m_0, P_0)$, A_{k-1} is the transition matrix of the dynamic model, and H_k is the measurement model matrix.

The Equations (2.2) admit a closed form solution and the distributions of interest are Gaussian with the following statistics:

$$\begin{aligned} p(x_k \mid y_{1:k-1}) &= N(x_k \mid m_k^-, P_k^-), \\ p(x_k \mid y_{1:k}) &= N(x_k \mid m_k, P_k), \\ p(y_k \mid y_{1:k-1}) &= N(y_k \mid H_k m_k^-, S_k). \end{aligned}$$

At each step, the parameters of the distributions above are computed using the

following procedure:

$$
\begin{aligned}
m_k^- &= A_{k-1} m_{k-1}, \\
P_k^- &= A_{k-1} P_{k-1} A_{k-1}^{\mathrm{T}} + Q_{k-1}, \\
v_k &= y_k - H_k m_k^-, \\
S_k &= H_k P_k^- H_k^{\mathrm{T}} + R_k, \\
K_k &= P_k^- H_k^{\mathrm{T}} S_k^{-1}, \\
m_k &= m_k^- + K_k v_k, \\
P_k &= P_k^- - K_k S_k K_k^{\mathrm{T}}.
\end{aligned}
$$

Therefore, the statistics of the random variable of interest are:

$$
\begin{aligned}
m_k^- =& A_{k-1} m_{k-1}, \\
P_k^- =& A_{k-1} P_{k-1} A_{k-1}^T + \Sigma_{k-1}, \\
m_k =& m_k^- + P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} (y_k - H_k m_k^-), \\
P_k =& P_k^- - P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} H_k P_k^-.
\end{aligned}
$$

2.4 Introduction to differential geometry

2.4.1 The manifold of symmetric positive definite matrices

Let $S_+(n, r)$ denote the space of symmetric positive (semi-)definite $n \times n$ matrices of rank r, endowed with an appropriate notion of distance d. This manifold has been studied extensively in the literature. Let A_1 and A_2 belong to such manifold. One of the most celebrated results is the existence of a natural metric $d(A_1, A_2)$ that is invariant with respect to matrix inversion:

$$
d(A_1, A_2) = d(A_1^{-1}, A_2^{-1}), \tag{2.3}
$$

and with respect to congruence via any invertible matrix Z:

$$d(A_1, A_2) = d(ZA_1Z^T, ZA_2Z^T). \tag{2.4}$$

Moreover, a parametrization of the geodesic between A_1 and A_2 is given by:

$$\varphi_{A_1 \to A_2}(t) = A_1^{\frac{1}{2}} \exp_m(t \log_m(A_1^{-\frac{1}{2}} A_2 A_1^{-\frac{1}{2}}))A_1^{\frac{1}{2}},$$

where $\varphi_{A_1 \to A_2}(t) \in S_+(n, n)$ for all $t \in \mathbb{R}$, and $A_1^{-\frac{1}{2}}, A_2^{-\frac{1}{2}}$ are symmetric inverse square roots. Clearly, $A_1^{-\frac{1}{2}} A_2 A_1^{-\frac{1}{2}}$ admits an eigenvalue decomposition of the form $U\Lambda U^T$, where Λ is a diagonal matrix containing the generalized eigenvalues of the pencil (A_1, A_2), and U their corresponding eigenvectors. Therefore, $\varphi_{A_1 \to A_2}(t)$ can be expressed as:

$$\varphi_{A_1 \to A_2}(t) = A_1^{\frac{1}{2}} \exp_m(t \log_m(U\Lambda U^T))A_1^{\frac{1}{2}} = A_1^{\frac{1}{2}} U \Lambda^t U^T A_1^{\frac{1}{2}}. \tag{2.5}$$

Notice that we recover the trivial cases $\varphi_{A_1 \to A_2}(t = 0) = A_1$ and $\varphi_{A_1 \to A_2}(t = 1) = A_2$. Apart from the parametrization, the geodesic also admits a close form expression for its distance:

$$d(A_1, A_2) = d(A_1^{-\frac{1}{2}} A_2 A_1^{-\frac{1}{2}}, I) = \| \log_m(A_1^{-\frac{1}{2}} A_2 A_1^{-\frac{1}{2}}) \|_F = \sqrt{\sum_{k=1}^{n} \log^2(\Lambda_{k,k})}.$$

2.4.2 Interpolation on a matrix manifold

Interpolating the covariance matrices with respect to a parameter can be useful as a tool to save storage and computational cost. One could approach this problem by interpolating component-wise the matrices of interest. However, covariance matrices must satisfy some conditions. Indeed, by construction, they are symmetric and positive definite (or at least positive semi-definite). The symmetric property would be preserved in an hypothetical interpolation by components. Also, the positive definiteness would also hold for a convex combination of two matrices. However, in the

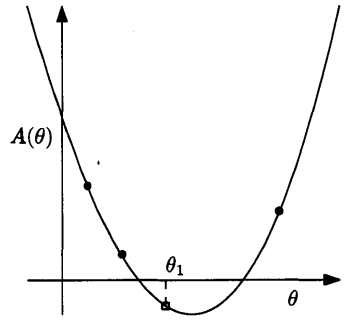


Figure 2-1: Quadratic interpolation among three matrices 1x1, interpolants are positive and interpolated is not.

case we would be interested on higher order interpolation, that may be not guaranteed. To illustrate, Figure 2-1 shows a hypothetical scenario where we are interested on performing quadratic interpolation among three points of a 1×1 matrix $A = A(\theta)$. Notice that even though the three data points are positive, so they belong to the manifold of symmetric positive definite (s.p.d.) matrices, we are not guaranteed that the interpolated $A(\theta)$ is positive.

The intuition of interpolating will be to draw a line in the manifold of s.p.d. matrices of dimension $N \times N$ (denoted as $S_+(N, N)$). In particular, we will use the concept of geodesic line. As defined in differential geometry, a *geodesic* minimizes arc length for points "sufficiently close". In turn, to define a curve of extremal length, we need to define a notion of distance. There is a preferred distance in $S_+(N, N)$ [15, 10], which coincides with the Förstner and Moonen metric introduced in [16] and used in [41].

The geodesic corresponding to this notion of distance defines a line parameterized by s between $A, B \in S_+(N, N)$, [10].

The invariance characteristic is especially relevant for covariance matrices [40] since in that case the $Gl(n)$ action corresponds to a change of basis ($x' = Lx$).

2.5 Introduction to dynamic programming

Dynamic programming was introduced by Richard Bellman in the sixties in its manuscripts [6, 7]. The beauty of this methodology is that it allows incorporating new information as soon as it is obtained (dynamically). Not surprisingly, it turns out that almost all optimization problems can be cast using the dynamic programming framework. Philosophically speaking, the dynamic decision making process is the one that we, as humans, face daily in our life. In the moment a decision has to be made, we do not typically have all the necessary information. Instead, sometimes we need to incorporate information as we go. That is precisely dynamic programming. It is neither surprising that this methodology is used nowadays not only in mathematics and operations research, but also in economics, management science, policy making, and computer science, to name a few.

In the sequel we introduce the infinite horizon problem, which means that the dynamical system will not stop running. We expose the conditions for optimality thereafter.

2.5.1 Infinite horizon problems

As described in [8, 9], the total cost associated to an initial state s_0 and a policy $\pi = \{\mu_0, \mu_1, \dots\}$ for the infinite-stage problem reads:

$$J_\pi(s_0) = \lim_{n\to\infty} \mathop{\mathbb{E}}_{\substack{w_k \\ k=0,1,\dots}} \left\{ \sum_{k=0}^{n-1} \alpha^k g(s_k, \mu_k(x_k), w_k) \right\}, \tag{2.6}$$

where $g(s_k, \mu_k(x_k), w_k)$ is the cost of being in the state s_k in stage k, applying a policy $\mu_k(x_k)$, and having a disturbance w_k. The discount factor of such cost is denoted by α.

If we consider the undiscounted problem ($\alpha = 1$) and we set the cost to $g(s, u, w)$, notice that it is possible to rearrange Equation 2.6 such that we obtain the following

recursion:

$$J_{k+1}(s) = \min_{u \in U(x)} \mathop{\mathbb{E}}_{w} \Big\{ g(s,u,w) + J_k(f(s,u,w)) \Big\}, \; k = 0, 1, \ldots \tag{2.7}$$

Therefore, it is natural to denote the optimal infinite horizon cost as:

$$J^*(s) = \lim_{n \to \infty} J_n(s). \tag{2.8}$$

Combining Equations 2.7 and 2.8, we get the following form:

$$J^*(s) = \min_{u \in U(s)} \mathop{\mathbb{E}}_{w} \Big\{ g(s,u,w) + J^*(f(s,u,w)) \Big\}, \tag{2.9}$$

which is a system of equations. The optimality conditions of such equations are described in the following subsection.

2.5.2 Optimality conditions of the stochastic shortest path problem

As described in [8], we need to satisfy the following two conditions to guarantee the most useful results for the stochastic shortest path (SSP) problem:

- There exists at least one proper policy.
- For every improper policy, the corresponding cost is infinity for at least one state.

Under these two assumptions, we have the following results that will be used later:

- The optimal cost vector (Equation 2.8) is the unique solution of Bellman's equation (Equation 2.9).
- The method of value iteration (and its asynchronous version) converges to the optimal cost vector independently of the initial cost vector.

Chapter 3

Formulation

3.1 The *big* picture

The problem of navigating in complex wind fields requires a multidisciplinary approach. First, we need to understand the wind field behavior over the domain of interest. To do that, given a particular value of the external wind conditions (prevailing wind), we use a Monte Carlo simulation on a proprietary unsteady CFD model to compute an approximation of the pointwise mean and covariance matrix in a predefined regular grid. Using an unsteady CFD model allows taking into account the potential vortices and irregular features as part of the uncertainty described by the mean and covariance. Notice that even in the simplest cases, the covariance is anisotropic and non-stationary. Such sample covariance matrix captures the interactions of the wind behavior among all points in the grid.

Our approach surpass the modeling and inference challenges in this setting by constructing spatiotemporal GP (Section 2.1.1) representations of complex wind fields. We assume that, given a prevailing wind, the wind field over the domain of interest forms a GRF. That is, for any discretization of such domain, the wind components of the points in the grid follow a Gaussian distribution (Equation 2.1).

Once the mean and covariance matrix are computed and the Gaussian assumption is made, the GRF in the grid is well defined for each prevailing wind. This representation can then be used for multiple Bayesian inference techniques. Perhaps the simplest one

would be direct conditioning to particular measurements, which we apply in Section 3.2.4.

Another well-known technique that we use is filtering (e.g., [14]). A lot of work has been done with regards to filtering of Gaussian distributions in both linear and nonlinear dynamics. We focus on the KF (Section 3.3.2), an algorithm that allows updating the state of the system over time by incorporating potential measurements. As described in Section 2.3.2, the algorithm applies only for Gaussian distributions within linear dynamics and it works in a two step fashion. The first step corresponds to updating the system using the linear dynamics. The second step allows incorporating new measurements to reduce the uncertainty of the state. In our particular problem, we can define the state as the components of the wind in all points of the grid. The dynamics of the system will be precisely in charge of updating the wind field such that the covariance in each time discretization matches the one corresponding to the prevailing wind. Measurements acquired online either in some stations of the city or by the UAV itself can also be incorporated in the second step of the algorithm.

3.1.1 Main assumptions of the model

In our framework, we assume that, given a prevailing wind described by its heading θ, the wind field is a GRF of two spatial dimensions. Therefore, given a discretization of the domain, the components of the points in the grid follow a joint Gaussian distribution with mean and covariance matrix given by (3.1) and (3.2), respectively. This assumption is sound since the components of the wind in each point of the grid can take any value in the real line and they tend to be symmetric, as the normal distribution does. Indeed, industry best practices to deal with the uncertainty of wind is to use the Rayleigh distribution for the modulus of the wind in a particular location. Such distribution is obtained precisely by computing the modulus of two Gaussian random variables. Notice also that we are not considering a joint Gaussian distribution for (w, θ) but only for $w(\theta)$ if θ is fixed, which is a more relaxed assumption.

Once the wind field is characterized for a particular time step as a GRF, we make the assumption that it is also an observed linear Markov dynamic model. Essentially,

we need to satisfy the conditions described in 2.3.2 to apply the results of the KF.

As we will see in the sequel, we suppose that the heading of the prevailing wind is observed with certainty. That is, at each time step, we can measure without error the conditions of the external wind conditions. Together with the preceding assumption, we see that we will have a GRF describing the wind field at each time step.

Moreover, given that we obtain the covariance matrices for each prevailing wind from an unsteady CFD solver, we suppose that the wind field at a particular time has the same distribution as an ensemble of snapshots acquired over time. In other words, we assume ergodicity.

Finally, we suppose also that we have certainty in the location of the UAV and, when doing path planning, we do not consider the maneuvers of the vehicle.

3.1.2 Initial data

Suppose we have some snapshots of the wind field corresponding to a certain prevailing wind θ:

$$w^{(1)}(\theta), w^{(2)}(\theta), w^{(3)}(\theta), \ldots, w^{(n)}(\theta), \qquad w^{(i)}(\theta) \in \mathbb{R}^N, \quad i = 1, \ldots, n.$$

Each $w^{(i)}(\theta)$ is a vector containing the components of the velocity vector in each point of a discretization of the domain and corresponds to a particular time step of an unsteady CFD solver. Notice that we are assuming that the dynamical system conformed by all the time steps is *ergodic*. In other words, the system has the same behavior averaged over time as averaged over all the states.

We can compute the sample mean of these snapshots as follows:

$$\bar{w}(\theta) \simeq \frac{1}{n} \sum_{i=1}^{n} w^{(i)}(\theta).$$

Similarly, we can obtain the sample covariance:

$$\widehat{C}_w(\theta) = \frac{1}{n-1} \sum_{i=1}^{n} (w^{(i)}(\theta) - \bar{w}(\theta))(w^{(i)}(\theta) - \bar{w}(\theta))^T.$$

Notice that these two statistics of the sample are parameterized by θ, i.e., we shall have a sample mean and covariance for each value of θ. Suppose we can sample from m different values, i.e., $\theta_1, \ldots, \theta_m$. Thus, we have:

$$\bar{w}(\theta_0) \stackrel{\text{def.}}{=} w_{\theta_0}; \bar{w}(\theta_1) \stackrel{\text{def.}}{=} w_{\theta_1}; \ldots; \bar{w}(\theta_m) \stackrel{\text{def.}}{=} w_{\theta_m}. \tag{3.1}$$

And similarly for the covariance is stored as:

$$\widehat{C}_w(\theta_0) \stackrel{\text{def.}}{=} C_{\theta_0}; \widehat{C}_w(\theta_1) \stackrel{\text{def.}}{=} C_{\theta_1}; \ldots; \widehat{C}_w(\theta_m) \stackrel{\text{def.}}{=} C_{\theta_m}. \tag{3.2}$$

3.2 Hierarchical Gaussian models for wind fields

3.2.1 Preliminaries

Herein we assume that all the matrices in (3.2) are rank deficient. We truncate their rank to r, so they belong to the space of symmetric positive semi-definite $N \times N$ matrices of fixed rank r. We denote this manifold as $S_+(r, N)$ [44]. Notice that even if they were not to have the same rank, it is possible to truncate their singular value decomposition (SVD) (3.3) [43] to a value r for which the least $N-r$ eigenvalues for all the matrices in (3.2) are negligible. In other words, if they have $\text{rank}(\widehat{C}_w(\theta_i)) = r(\theta_i)$, we can still truncate for a $r \leq \min_{1,\ldots m} r(\theta_i)$:

$$\widehat{C}_w(\theta) = \sum_{i=1}^{r(\theta)} \lambda_i(\theta) v_i(\theta) v_i(\theta)^T \simeq \sum_{i=1}^{r} \lambda_i(\theta) v_i(\theta) v_i(\theta)^T. \tag{3.3}$$

Recall that performing an SVD implies defining a notion of orthogonality, we use the L_2 norm.

Since the matrices are rank deficient with $r \ll N$, the number of samples to characterize the sample mean and covariance may be $n \ll N$ and still provide a good approximation.

Notice that we do not ensemble any of the matrices in (3.2). We shall use a randomized SVD approach to obtain the eigenvalue decomposition directly from the

samples. Then, we store the $Y(\theta_i) \stackrel{\text{def.}}{=} Y_{\theta_i}$ that are rectangular real matrices $N \times r$ that satisfy the following relation:

$$\widehat{C}_w(\theta_i) = \underbrace{V_{\theta_i}^r \Lambda_{\theta_i}^{r\,\frac{1}{2}}}_{Y_{\theta_i}} \underbrace{\Lambda_{\theta_i}^{r\,\frac{1}{2}} V_{\theta_i}^{r\,T}}_{Y_{\theta_i}^T} = Y_{\theta_i} {Y_{\theta_i}}^T, \quad i = 1, \ldots, m, \tag{3.4}$$

which can be easily obtained directly from the SVD (3.3) by putting the first r eigenvectors times the square root of the corresponding eigenvalues in columns.

In our particular problem, the sample mean and covariance matrices described above correspond to the components of the wind in a grid, given a condition of prevailing wind θ. A covariance like the one in (3.2) is anisotropic and non-stationary and captures the interactions of the wind behavior among all points in the grid.

3.2.2 Gaussian models for a wind field

As discussed, we assume that, given a prevailing wind described by θ, the wind field is a GRF of two spatial dimensions. However, in our case, the covariance matrix Σ is rank deficient, then the multivariate normal distribution is degenerate and does not have a density. In order to have a generalization of the density for the degenerate case, we choose a different measure. In particular, we can restrict the Lebesgue measure to the r-dimensional affine subspace where the Gaussian distribution is supported. In that case, the distribution has density:

$$f_x(x_1, \ldots, x_k) = \frac{1}{\sqrt{\det{}^*(2\pi\Sigma)}} \exp\Big(-\frac{1}{2}(x-\mu)^{\mathrm{T}} \Sigma^{+} (x-\mu)\Big),$$

where Σ^+ is the generalized inverse and $\det^*$ is the pseudo-determinant.

Nonetheless, as it will be presented below, the rank deficient covariance matrices presented herein will be eventually full rank by the natural addition of white noise.

3.2.3 Relation with the Karhunen-Loève expansion

Once the GRF given θ is defined, the truncation described in (3.3) can be understood as the Karhunen-Loève (KL) expansion [27] of the discretized version of the GRF given θ.

The KL expansion involves the computation of the spectrum of the covariance kernel $C(x, \bar{x})$. The spectrum is related to the covariance kernel via the following second kind integral equation, also known as Fredholm equation:

$$\int_{\Omega} C(x, \bar{x})\phi_i(\bar{x})d\bar{x} = \lambda_i \phi_i(x),$$

where $\phi_i(x)$ are the eigenvectors and λ_i the eigenvalues.

Once the spectrum of the covariance kernel is computed we may proceed to the KL expansion of the GRF Y. In general, a KL expansion may be expressed as:

$$Y(x, \omega) = \mu_Y(x) + \sum_{i \geq 1} \sigma_i(x) z_i(\omega), \qquad \sigma_i(x) = \sqrt{\lambda_i}\phi_i(x),$$

where $\mu_Y(x) = \mu_Y$ and $z_i(\omega) = \xi_i \sim \mathcal{N}(0, 1)$.

Hence, if the eigenvalues are sort from largest to smallest, the latter series can be truncated using only the first r pairs of eigenvalues/eigenfunctions. This approximation is the one that minimizes the mean-squared error.

The covariance matrices in (3.2) correspond to a discretization of the covariance kernel described above. Therefore, some properties of such decomposition can be applied.

As a result, the factorization presented in (3.5) is the r-decomposition that minimizes the total mean square error in the L_2 norm. Also, the total variance of the r-truncated approximation is:

$$\sum_{i=1}^{r} \lambda_k(\theta).$$

Then, the r-truncated expansion explains the following percentage of the total variance

for the prevailing wind condition corresponding to θ:

$$\frac{\sum_{i=1}^{r} \lambda_k(\theta)}{\sum_{i=1}^{r(\theta)} \lambda_k(\theta)}.$$

3.2.4 Direct conditioning of the covariance matrix

Given the Gaussianity of our model, there exists a close form for the conditionals.

First, we rewrite the SVD (3.3) into two pieces where part A contains the unknown information and part B corresponds to some measurements:

$$\begin{aligned}
\widehat{C}_w(\theta) &= \sum_{i=1}^{n \ll N} \lambda_i(\theta) \begin{Bmatrix} v_i^A(\theta) \\ v_i^B(\theta) \end{Bmatrix} \begin{Bmatrix} v_i^A(\theta)^T & v_i^B(\theta)^T \end{Bmatrix} = \\
&= \begin{pmatrix} \sum \lambda_i(\theta) v_i^A(\theta) v_i^A(\theta)^T & \sum \lambda_i(\theta) v_i^A(\theta) v_i^B(\theta)^T \\ \sum \lambda_i(\theta) v_i^B(\theta) v_i^A(\theta)^T & \sum \lambda_i(\theta) v_i^B(\theta) v_i^B(\theta)^T \end{pmatrix} = \\
&= \begin{pmatrix} A(\theta) & C(\theta) \\ C(\theta)^T & B(\theta) \end{pmatrix}.
\end{aligned} \tag{3.5}$$

Notice that it is possible to rearrange the components of any vector and matrix in (3.1) and (3.2) in order to have the components with measurements at the end.

Therefore, the Gaussian random variable that comes from the discretization of the GRF conditioned to some pointwise measurements ($y = Hw$) takes the following form:

$$(W|y = Hw; \theta) \sim N(\bar{w}^A(\theta) + C(\theta)B(\theta)^{-1}(y - \bar{w}^B(\theta)), A(\theta) - C(\theta)B(\theta)^{-1}C(\theta)^T),$$
$$\text{where } \bar{w}(\theta) = \begin{Bmatrix} \bar{w}^A(\theta) & \bar{w}^B(\theta) \end{Bmatrix}.$$

As described in [45], it is trivial to add noise in the measurement by adding an identity matrix times α in the $B(\theta)$ part of (3.5), where α is the additional variance due to noise.

Notice the hierarchical nature of our model, where all random variables depend on their corresponding prevailing wind conditions θ.

3.2.5 Low-rank covariance interpolation

Approximations of the statistics of a random variable are key to characterize uncertainty. Sometimes it is not enough to interpret the mean and then some measure of variability must be quantified. For multidimensional random variables, it is typically sufficient to estimate the covariance matrix to have a sense on variability. An important consideration is that the covariance matrices in such problems may depend upon a set of parameters. In that setting, sometimes interpolation of a covariance matrix may be required for two reasons: (i) impossibility to store the matrices for a continuous value of the explanatory parameter, (ii) having only access to the matrices for a finite number of realizations of the parameter. However, some challenges arise when considering such interpolation. For instance, often the interpolants belong to a particular manifold and it is desirable that the interpolated matrices belong to the same one. The most popular application of interpolation on matrix manifolds is on reduced order models, where interpolation has been proved to be useful (e.g., [3, 2, 4]).

As described in Section 2.4.1, extensive work has been done on characterizing the manifold of s.p.d. matrices (e.g., [15, 40]). The manifold is fully characterized and it has a preferred notion of distance called the natural metric on the symmetric cone [15]. Nevertheless, there are fewer references on the definition of the manifold of symmetric positive semi-definite matrices (s.p.s.d.) [10, 44]. In this former manifold, it does not exist a standard notion of metric or distance because it is not possible to define one with the same characteristics as in s.p.d. matrices. On top of that, even fewer authors talk about interpolation or regression in the aforementioned manifolds [4, 30, 20]. One intuitive approach to perform interpolation is following a geodesic between two interpolants in the manifold of interest. The parameter may be then defined as the *proportion* of distance between the two given data points (matrices).

In order to deal with covariance matrices under a specific manifold, we need to define the concept of distance in the specific space we are working with. However, as shown in [10], there is not a preferred notion of distance in $S_+(r, N)$, $r < N$, that satisfies the invariance characteristics (Equations 2.3 and 2.3). These properties are

especially relevant for covariance matrices. In that case, the invariance under inversion would mean that the distance between covariance matrices is the same than with their corresponding precision matrices. The invariance under congruence transformation corresponds to a change of basis.

Now we would like to build a line similar to the one in (Equation 2.5) but for $S_+(r, N)$. The approach that we will be using going forward [44] consists of interpolating (under some conditions) the square root factors Y of dimension $N \times r$, i.e., we take advantage of the decomposition described in Equation (3.4).

Without loss of generality, let us consider that the two available data points are C_{θ_0} and C_{θ_1} and we can obtain their square roots as follows:

$$C_{\theta_0} = V^r_{\theta_0} \Lambda^r_{\theta_0} V^{r^T}_{\theta_0} = Y_{\theta_0} Y^T_{\theta_0},$$

$$C_{\theta_1} = V^r_{\theta_1} \Lambda^r_{\theta_1} V^{r^T}_{\theta_1} = Y_{\theta_1} Y^T_{\theta_1}.$$

The idea consists of finding a geodesic line that satisfies two conditions. Firstly, we would like the matrix symmetric for all points contained in the line [44]. Secondly, we would like the parametrization to be invariant with respect to the decomposition. Therefore, we need to find $\dot{Y}$ such that:

$$\begin{aligned} &\text{(i) } Y^T_{\theta_0} \dot{Y} = \dot{Y}^T Y_{\theta_0}, \\ &\text{(ii) } Y_{\theta_0} + \dot{Y} = Y_{\theta_1} Q, \text{ for some } Q. \end{aligned} \tag{3.6}$$

Once the matrix Q is obtained, the geodesic line [44] in $S_+(r, N)$ reads as:

$$C_{\theta_s} = (Y_{\theta_0} + s\dot{Y})(Y_{\theta_0} + s\dot{Y})^T.$$

And we recover the C_{θ_s} by multiplying again its factors:

$$C_{\theta_s} = Y_{\theta_s} Y^T_{\theta_s} = \widehat{C}_w(\theta_s).$$

Therefore, it will be enough to obtain a Q of the form in (Equation 3.6). To

do that, we follow the procedure below, first multiplying the second condition in (Equation 3.6) by $Y_{\theta_0}^T$ on the left hand side:

$$Y_{\theta_0}^T Y_{\theta_0} + Y_{\theta_0}^T \dot{Y} = Y_{\theta_0}^T Y_{\theta_1} Q. \tag{3.7}$$

Transposing the equation:

$$Y_{\theta_0}^T Y_{\theta_0} + \dot{Y}^T Y_{\theta_0} = Q^T Y_{\theta_1}^T Y_{\theta_0}. \tag{3.8}$$

Subtracting the Equations (3.7) and (3.8),

$$Y_{\theta_0}^T \dot{Y} - \dot{Y}^T Y_{\theta_0} = Y_{\theta_0} Y_{\theta_1} Q - Q^T Y_{\theta_1}^T Y_{\theta_0}.$$

Constraining using the first condition in Equation (3.6):

$$Y_{\theta_0}^T \dot{Y} = \dot{Y} Y_{\theta_0}.$$

Therefore, it all comes down to satisfying the following equality:

$$Y_{\theta_0}^T Y_{\theta_1} Q = Q^T Y_{\theta_1}^T Y_{\theta_0}. \tag{3.9}$$

Now, we perform a polar decomposition on $Y_{\theta_0}^T Y_{\theta_1}$:

$$Y_{\theta_0}^T Y_{\theta_1} = UP, \text{ where } P = P^T \text{ and } U^T = U^{-1}.$$

Notice that we can do such decomposition by again performing a SVD:

$$Y_{\theta_0}^T Y_{\theta_1} = U_{\theta_0,\theta_1} \Lambda_{\theta_0,\theta_1} V_{\theta_0,\theta_1}^T.$$

And, since V_{θ_0,θ_1} is an orthonormal matrix, we have:

$$Y_{\theta_0}^T Y_{\theta_1} = \underbrace{U_{\theta_0,\theta_1} V_{\theta_0,\theta_1}^T}_{U} \underbrace{V_{\theta_0,\theta_1} \Lambda_{\theta_0,\theta_1} V_{\theta_0,\theta_1}^T}_{P}.$$

So we define U and P as:

$$U = U_{\theta_0,\theta_1} V_{\theta_0,\theta_1}^T,$$
$$P = V_{\theta_0,\theta_1} \Lambda_{\theta_0,\theta_1} V_{\theta_0,\theta_1}^T.$$

Then, Equation (3.9) becomes:

$$UPQ = Q^T P^T U^T.$$

As a conclusion, Q is as simple as:

$$Q = U^T.$$

Therefore,

$$\dot{Y} = Y_{\theta_1} U^T - Y_{\theta_0}.$$

Then we have the curve:

$$s \longmapsto \underbrace{(Y_{\theta_0} + s\dot{Y})}_{Y_{\theta_s}} \underbrace{(Y_{\theta_0} + s\dot{Y})^T}_{Y_{\theta_s}^T}. \tag{3.10}$$

Notice that we have taken advantage of the way we stored the matrices, i.e., $\widehat{C}_w(\theta) = Y_\theta Y_\theta^T$, and we also recover the $\widehat{C}_w(\theta_s)$ again with the same factorization form.

3.2.6 Linear spline through data matrices

Following the previous section, we can also perform piecewise linear interpolation between our data matrices. Therefore, the piecewise interpolation reads as follows:

$$\theta_s \in (\theta_j, \theta_{j+1}) \longmapsto \widehat{C}_w(\theta_s) = \underbrace{(Y_{\theta_j} + s\dot{Y})}_{Y_{\theta_s}} \underbrace{(Y_{\theta_j} + s\dot{Y})^T}_{Y_{\theta_s}^T},$$

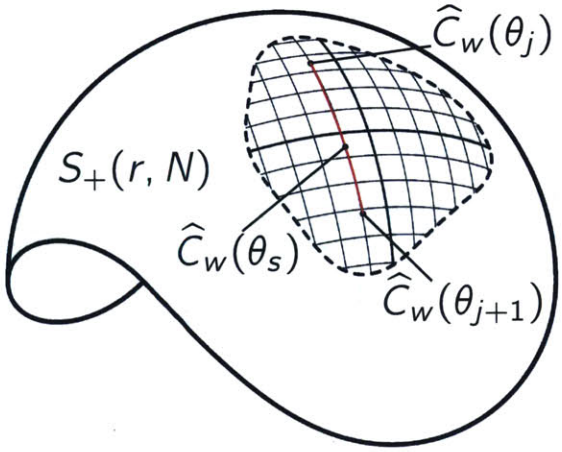


Figure 3-1: Piecewise linear interpolation of covariance matrices.

where s is the following proportion:

$$s = \frac{\theta_s - \theta_j}{\theta_{j+1} - \theta_j}.$$

And $\dot{Y}$ is such that:

$$\text{(i) } Y_{\theta_j}^T \dot{Y} = \dot{Y}^T Y_{\theta_j},$$
$$\text{(ii) } Y_{\theta_j} + \dot{Y} = Y_{\theta_{j+1}} Q,$$
$$\text{for some } Q.$$

Figure 3-1 represents the piecewise interpolation. In particular, we show the step between the covariance matrices corresponding to θ_j and θ_{j+1}. Notice that we take $j = 1, 2, \ldots m$ as described in the initial data in Section 3.1.2.

In the same way, Figure 3-2 shows the goal of the interpolation of in the wind field. Here, we have the covariance matrix corresponding to θ_j for $j = 1, 2, \ldots m$ and we would like to find he one for θ_s.

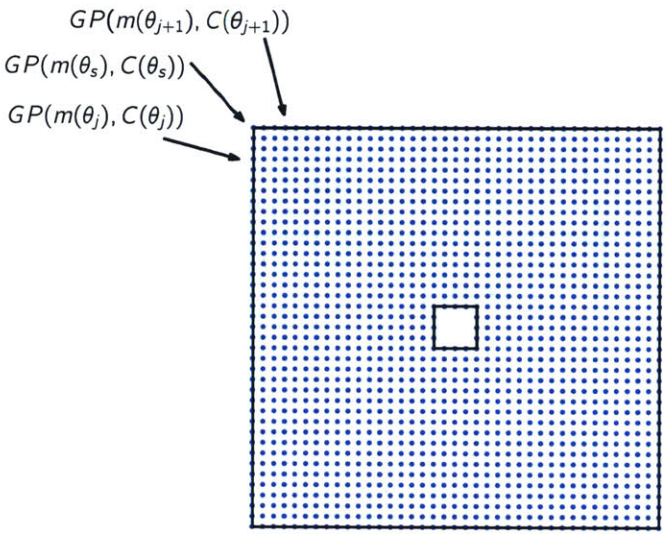


Figure 3-2: Domain of the wind field and representation of the prevailing wind.

3.3 Wind field estimation

As described in Section 1.2.2, the wind field that will be analyzed is based in a 2D domain of 100×100 meters of side with an inner obstacle of 10×10.

As seen previously, given the Gaussianity assumption, we can easily manipulate the covariance matrix such that we condition the samples to some observations obtained in at any locations. To do that, we introduced the notion of conditional covariance matrix for the multivariate Gaussian random variable. We apply this framework in Section 3.3.1.

Apart from direct conditioning, we are interested in creating a dynamic wind field that allows updates as the UAV obtains onboard measurements. To do that, we assume the Markovian property described in Section 2.3.2 and proceed by embedding a KF structure with autoregressive dynamics to the wind field. We formulate this solution in Section 3.3.2.

3.3.1 Static model

As the UAV navigates through the wind field of interest, more information of the pointwise velocity of the wind is obtained. This information can come from:

Figure 3-3: Graphical interpretation of the static model.

- Pointwise measurements obtained from the UAV.
- Measurements from anemometers.

Now we would like to assimilate these observations to make better predictions of the wind field.

The statistical system for the wind field w with prior $w \sim \mathcal{N}(\bar{w}, \widehat{C}_w)$ conditioned on pointwise observations y measured with error covariance Σ reads:

$$\begin{aligned} y &= Hw + q, \\ q &\perp y, \\ q &\sim \mathcal{N}(0, \Sigma). \end{aligned}$$

Graphically, the system can also be interpreted as in Figure 3-3.

After applying the formulation in Section 3.2.4, we get a posterior estimation that captures the information from the observations. Notice that this posterior wind field is also Gaussian and the conditional mean and covariance are computed in closed form. Results are shown in the following Chapter.

3.3.2 Dynamic model

The solution presented above allows assimilating static data, i.e., at one single time step. Now we are interested in assimilating data over a period of time. To do that, we consider a vector that contains all the two components of the wind at all locations w_k, for each time step $k = 1, 2 \ldots$. In the present section, we suppose that the conditions of the prevailing wind θ_k are observed without error for all k. We relax this assumption in Section 3.3.3.

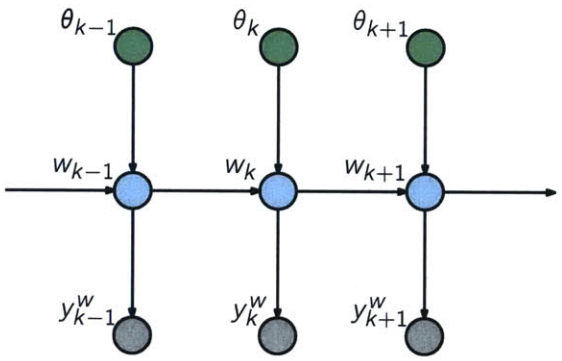


Figure 3-4: Graphical interpretation of the dynamic model.

Now, we define the following filtering framework where the dynamics and measurement models are linear Gaussian:

$$\begin{aligned} w_k &= \alpha w_{k-1} + \sqrt{1-\alpha^2} r_{k-1}, \\ y_k &= H_k w_k + q_k, \\ r_{k-1} &\sim \mathcal{N}\left(\frac{\bar{w}(\theta_k) - \alpha \bar{w}(\theta_{k-1})}{\sqrt{1-\alpha^2}}, \widehat{C}_w(\theta_k) \right). \end{aligned} \tag{3.11}$$

And we have the following conditions:

$$\begin{aligned} &\alpha \in [0,1), \\ &q_k \perp y_k, \\ &q_k \sim \mathcal{N}(0, \Sigma), \\ &w_0 \sim \mathcal{N}(\bar{w}(\theta_0), \widehat{C}_w(\theta_0)). \end{aligned}$$

Graphically, the dynamic system can also be interpreted as in Figure 3-4. Strictly speaking, as per the equations above, the graph should have and edge between θ_{k-1} and w_k, $k = 1, 2 \ldots$. However, notice that we can always augment the state, i.e., considering $\widehat{\theta_k} = (\theta_{k-1}, \theta_k)$ and thus, come back to the presented graph. Therefore, for simplicity and in order to revert to the graph of a typical KF, we decide not to add the mentioned edge.

Clearly, the parameter α describes the dynamics of the system. Three particular details deserve further attention:

- $\alpha \in [0, 1)$ guarantees finite variance for all k.
- Define the increment of time between states as $\Delta t := t_k - t_{k-1}$. If time step is short, then it is better to set $\alpha \approx 1$ and the state estimates do not change a lot over time. Conversely, if Δt is large, then it is preferable to set $\alpha \approx 0$.
- The information from CFD is built into the wind field via the precomputed $\bar{w}(\theta_k)$ and $\widehat{C}_w(\theta_k)$.

Notice that the preceding formulation corresponds to a KF as seen in Section 2.3.2. In our case, we set A_k to be a multiple of the identity for all k, which corresponds to an autoregressive model (e.g., [37]). This filter is useful when we want to recompute the state online relying almost only in new measurements but embedding a notion of memory of previous states.

3.3.3 Hierarchical dynamic model

Using the representation described in Equation 3.10, it is possible to have an approximation of a line that contains the covariance matrices for all possible values of the prevailing wind heading θ. In order to use all the available data points (and so, to get a better approximation), we performed a linear spline among the data to find the unknown matrices.

In Section 3.3.2, we supposed that we observed without error the conditions of the prevailing wind at each time step. Now we relax this assumption and we propose an additional filtering framework (in conjunction with Equation 3.11) for θ_k where the dynamics and measurement models are also linear Gaussian:

$$\begin{aligned} \theta_k &= \beta\theta_{k-1} + \sqrt{1-\beta^2}s_{k-1}, \\ y_k^\theta &= A_k\theta_k + p_k. \end{aligned} \tag{3.12}$$

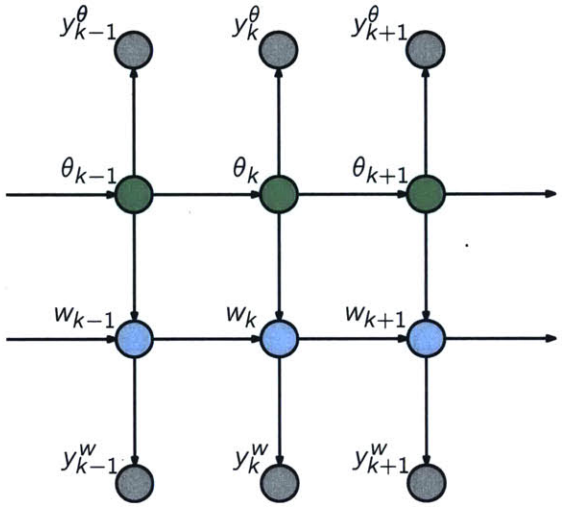


Figure 3-5: Graphical interpretation of the complete model.

The combination of Equations 3.11 and 3.12. This case corresponds to observed but uncertain prevailing wind θ in each step.

Graphically, the dynamic system can also be interpreted as in Figure 3-5. Again, strictly speaking, the same reasoning in the previous section apply and, as per the equations above, the graph should also have and edge between θ_{k-1} and w_k, $k = 1, 2 \ldots$ here. As described above, augmenting the state, i.e., considering $\widehat{\theta_k} = (\theta_{k-1}, \theta_k)$, we recover the graph in Figure 3-5.

Notice that the parameter θ_k enters directly into the statistics of w_k. Therefore, even if θ_k has a normal distribution, the model is non-Gaussian in each time step. For that reason, the solutions of the KF presented in Section 2.3.2 do not apply anymore and more involved and computationally intense algorithms need to be devised.

These algorithms are based in particle filtering, which in our case would consists of, at each step, generating particles and computing first the KF in Equation 3.12, then for each particle solving the KF in Equation 3.11. Finally, computing a Gaussian mixture model of the distributions of all particles to have the estimate of the wind field at the end of each step (flowchart page 59). The details of the application of this algorithm on this context are left as further research. For the sake of simplicity, we use the model in Equation 3.11 in the numerical results, Section 4.1.3.

3.4 Wind field exploitation

Navigating and planning in urban wind fields involves two steps. First, the wind field estimate and its uncertainty within the domain of interest must be characterized. Second, an efficient algorithm must be designed in a way that can deal with such uncertainty to find an optimal path. Quantifying uncertainty in current and future wind fields requires a reliable predictive model and benefits strongly from the ability to incorporate onboard and external wind field measurements in real time. These data-informed predictions in turn inform path planning algorithms that aim to minimize energy consumption or flying time, enabling wind field exploitation. The problem is then typically divided in wind field *estimation* and wind field *exploitation* as defined above; we focus on the former in the present section.

As discussed in Section 3.3, we suppose we have a mean and covariance matrix given a condition of prevailing wind. Such covariance is anisotropic and non-stationary and captures the interactions of the wind behavior among all points in the grid. We assume that, given a prevailing wind, the wind components of the points in the grid follow a Gaussian distribution. Once the mean and covariance matrix are computed and the Gaussian assumption is made, the GRF in the grid is well-defined for a particular prevailing wind. Such problem is now well-posed to be tackled using dynamic programming, in particular as a SSP problem. The goal of the proposed project is comparing a minimum-energy planner aware of the wind field to the naïve shortest distance planning. The end-goal would be to recompute the trajectory online as we obtain more information about the wind field, instead of supposing that the statistics of the behavior of the wind are constant.

In this section we describe the formulation for the path planning algorithm. As described below, this corresponds to a SSP problem. However, as described in 3.1.1, we will consider that the only source of stochasticity is in the costs and, given a policy, we do not have uncertainty in the location of the UAV.

3.4.1 State space

The state (i) is defined as the position of the UAV. As mentioned before, there are $\frac{N}{2}$ nodes in the domain (Figure 3-2). Therefore the state i can take any value from $i = 1, \ldots, \frac{N}{2}$.

3.4.2 Control space

Our space of controls is either moving to any of the 8 neighboring nodes or staying in the present state (Figure 3-6). We increment the cost of taking a diagonal (as opposed to moving vertical or horizontally) by a factor that accounts for the larger distance of such movement. Notice that there are some improper policies. For instance, given the upper left corner as origin and the lower right as destination, the policy of always going down does not terminate. We set to infinity the cost of being in the borders of the wind field and inside of the inner obstacle. Although this is not the case for the toy-problem described herein, the number of states may be so large that we may be obligated to find suboptimal solutions. An alternative would be to optimize only for a particular finite number of n steps (n-step lookahead). Secondly, given a particular location, we only consider a finite number of controls, this approach is also suboptimal in itself and more complex policies should be examined. A common practice is to randomized such transitions and describing the position as the probabilities of staying in the closer nodes. However, this approach increases the computational cost dramatically and it became infeasible to recompute the costs of the wind field online.

Third, notice that we do not consider the dynamics of the UAV. Typically, an aircraft is restricted to the movements allowed by its maneuvers. However, if the distance between nodes is large enough, it is feasible to approximate such dynamics in the way it is formulated in this project.

3.4.3 Cost function

In the formulation presented herein, we are interested in minimizing the consumption of energy.

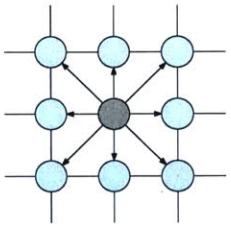


Figure 3-6: Representation of the control space. The UAV can move to any of the 8 neighboring nodes.

First, we can write the recursion for the stochastic shortest path problem [8, 9]:

$$J_k(i) = \min_{u \in U(i)} \left\{ g_k(i,u) + \sum_{j=1}^{N/2} p_{ij}(u) J_{k+1}(j) \right\}, i = 1, \ldots, \frac{N}{2}. \tag{3.13}$$

We suppose that the consumption of energy is proportional (up to a constant α with the appropriate units) to the modulus of the relative velocity, which is the direction defined by our control minus the one defined by the wind. For each i and u, we set $p_{ij}(u) = 1$ only for a particular j. In other words, given a state and a control, the following state is deterministic.

We denote with $V(i)$ as the relative velocity of the UAV ($V_{UAV}(i)$) with respect to the velocity of the wind ($V_w(i)$) in a particular location i. Then, for some constant γ:

$$g_k(i,u) = \gamma \mathbb{E}_w \Big[(V_{UAV} - V_w)^T (V_{UAV} - V_w) \Big].$$

Since $V(i) = [V^{(1)}(i), V^{(2)}(i)]^T$ has 2 components, we will use linearity of the expectation operator and adding twice the well known identity that relates $\mathbb{E}[V^2]$ and $\mathbb{E}[V]^2$:

$$\mathbb{E}[V(i)^T V(i)] = \mathbb{E}[V(i)]^T \mathbb{E}[V(i)] + \sum_{m=1}^{2} \mathrm{Var}[V^{(m)}(i)].$$

Then, for a particular location i, we have the following form for the cost:

$$\begin{aligned}\frac{g_k(i,u)}{\gamma} =& \mathbb{E}_w[(V_{UAV} - V_w)^T(V_{UAV} - V_w)] = \\ =& \mathbb{E}_w[V_{UAV}^T V_{UAV}] - 2\mathbb{E}_w[V_w]^T\mathbb{E}_w[V_{UAV}] + \mathbb{E}_w[V_w^T V_w] = \\ =& V_{UAV}^T V_{UAV} - 2\mathbb{E}_w[V_w]^T V_{UAV} + \mathbb{E}_w[V_w]^T\mathbb{E}_w[V_w] + \sum_{m=1}^{2} \mathrm{Var}[V_w^{(m)}] = \\ =& (V_{UAV} - \mathbb{E}_w[V_w])^T(V_{UAV} - \mathbb{E}_w[V_w]) + \sum_{m=1}^{2} \mathrm{Var}[V_w^{(m)}].\end{aligned} \tag{3.14}$$

As will be described below, we do not consider any randomness in the velocity of the UAV ($V_{UAV}(i)$), so its expectation will equal its value and its variance will be 0.

3.4.4 Solver: asynchronous value iteration

To solve the problem we use asynchronous value iteration, which is iterating Equation 3.13 using always the available information updated for other states in each step. As proved in [8] and summarized in 2.5, if we set to infinity the costs associated to the improper policies and we can guarantee that there exists at least one proper policy, the value iteration algorithm will converge to the optimal solution.

Figure 3-7 describes the convergence of the total cost (sum across all states) as a function of the number of iterations of the asynchronous value iteration algorithm. Notice that after a certain number of iterations, the total cost stays the same. Also, if we start from a vector of costs that has 0 in all the components, we are also guaranteed [8] that after each iteration, the cost for each state will (monotonically but not strictly) increase. Since we do not have any possibility to have negative costs and the costs are increasing for each step, the sum of the costs being constant means that the algorithm has reached convergence. Moreover, the policy corresponding to the optimal cost is an optimal policy.

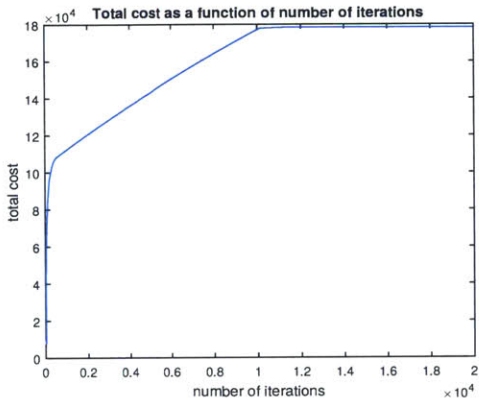


Figure 3-7: Convergence of the total cost (sum across all states).

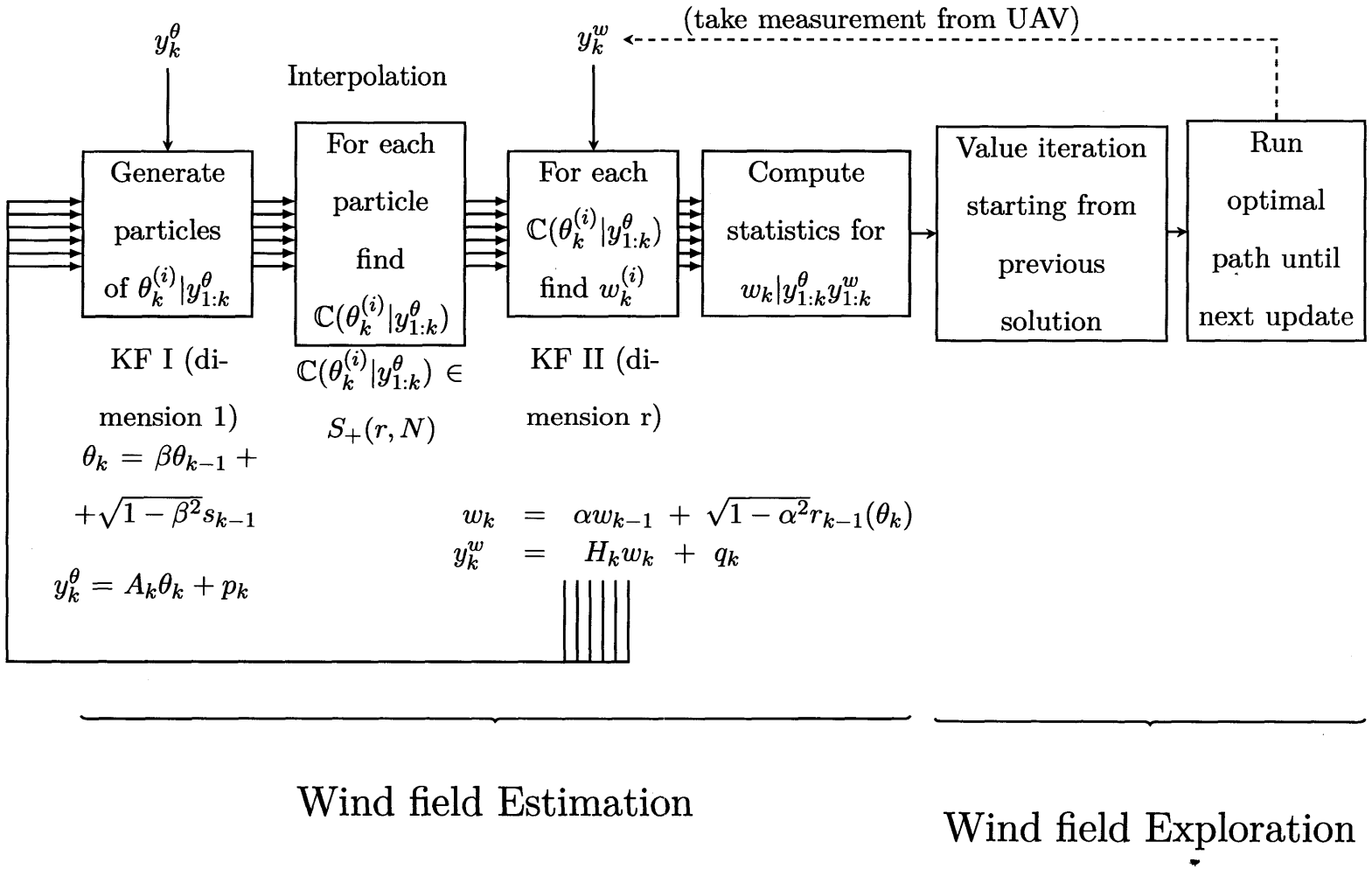
3.5 Integrated estimation and exploitation

Integrating wind field estimation and exploitation consists of using approximations of the wind field to reduce the cost function of a UAV navigating around a domain of interest. In our case, the cost function was the consumption of energy and wind field exploitation reduces to path planning. In this section, we combine Section 3.3 and 3.4 to update our estimate of the wind field while we recalibrate the path simultaneously.

The following flowchart in page 59 represents the necessary steps to pursue wind field integration within the framework defined in Chapter 3. Notice that, as briefly described in Section 3.3.3, the model is no longer Gaussian and it is necessary to use particle filtering.

In our particular example, we found particularly useful to update the path planning using the solution that we had in the previous step as a first approximation for the value iteration algorithm in the subsequent step.

Results of wind field integration are shown in Section 4.3. However, as mentioned previously, in terms of wind field estimation, we use the framework in 3.3.2, which supposes that we observe the prevailing wind without error.


y_k^θ
Interpolation
y_k^w
(take measurement from UAV)
Generate particles of $\theta_k^{(i)}|y_{1:k}^\theta$
For each particle find $\mathbb{C}(\theta_k^{(i)}|y_{1:k}^\theta)$
For each $\mathbb{C}(\theta_k^{(i)}|y_{1:k}^\theta)$ find $w_k^{(i)}$
Compute statistics for $w_k|y_{1:k}^\theta y_{1:k}^w$
Value iteration starting from previous solution
Run optimal path until next update
KF I (dimension 1)
$\theta_k = \beta\theta_{k-1} +$
$+\sqrt{1-\beta^2}s_{k-1}$
$y_k^\theta = A_k\theta_k + p_k$
$\mathbb{C}(\theta_k^{(i)}|y_{1:k}^\theta) \in S_+(r,N)$
KF II (dimension r)
$w_k = \alpha w_{k-1} + \sqrt{1-\alpha^2}r_{k-1}(\theta_k)$
$y_k^w = H_k w_k + q_k$
Wind field Estimation
Wind field Exploration

Chapter 4

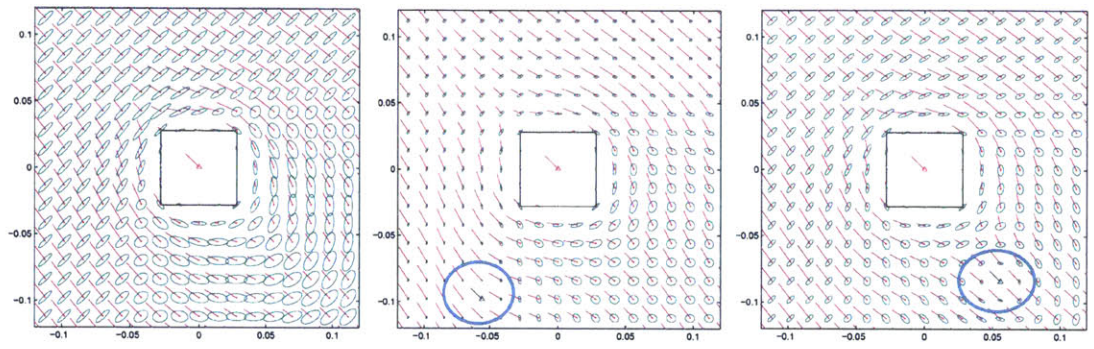
Numerical results

4.1 Wind field estimation

In this section we apply the formulation described in Section 3.3 and we show visually how wind field is updated as we acquire more information. Also, for consistency, throughout this section we use (without description) the notation introduced in Section 3.3.

4.1.1 Assimilation of static data

Assimilation of static data was described mathematically in Section 3.3.1. This approach allows incorporating static wind measurements and extrapolate such information to the whole wind field. A representation of this methodology can be observed in Figure 4-1, where arrows represent pointwise mean and ellipses are pointwise covariance. Notice the increment of variability due to vorticity after the obstacle. Moreover, in this Figure it is possible to ascertain spatial correlation. Indeed, for a particular measurement before the obstacle (Figure 4-1b), the ellipses are reduced considerably in size, implying acquisition of information. However, for a given measurement downstream (Figure 4-1c), there is little gain of information upstream.



(a) Prior estimation (b) Posterior representation (c) Posterior representation

Figure 4-1: Conditioning of the GRF in a simple domain given a measurement. Vectors represent mean and ellipses are pointwise covariance.

4.1.2 Interpolation of the wind variance field

Now we would like to interpolate the covariance matrices as described in Section 3.2.6. We use the data described in 3.1.2. The following sequence of Figures show how the pointwise variance changes as we move from one heading of the prevailing wind conditions to another one. Notice that, even though we interpolate the whole covariance matrices, we only show the variance field (i.e., the diagonal of the matrix). This does not mean that we do not require the other elements of the covariance matrices, we clearly use them when assimilating static and dynamic data. We color the obstacle in green when the variance field corresponds to a data point. Following the notation in Section 3.2.6, we perform piecewise linear interpolation from each covariance data point, where $m = 5$ and so $j = 1, 2, \ldots 5$.

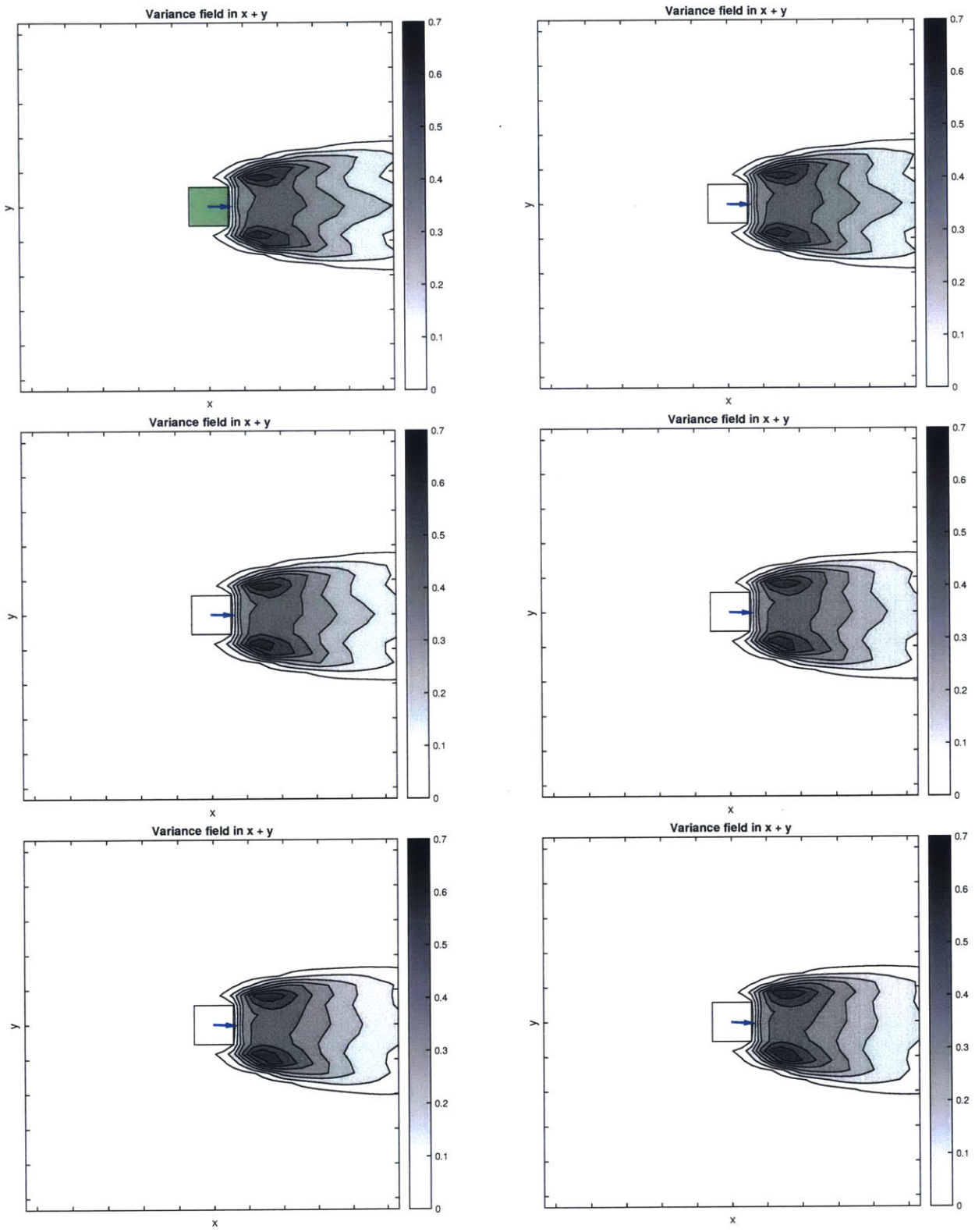


Figure 4-2: Linear interpolation of covariance matrices as a function of θ. The obstacle is green when the plot corresponds to a data point (Part I).

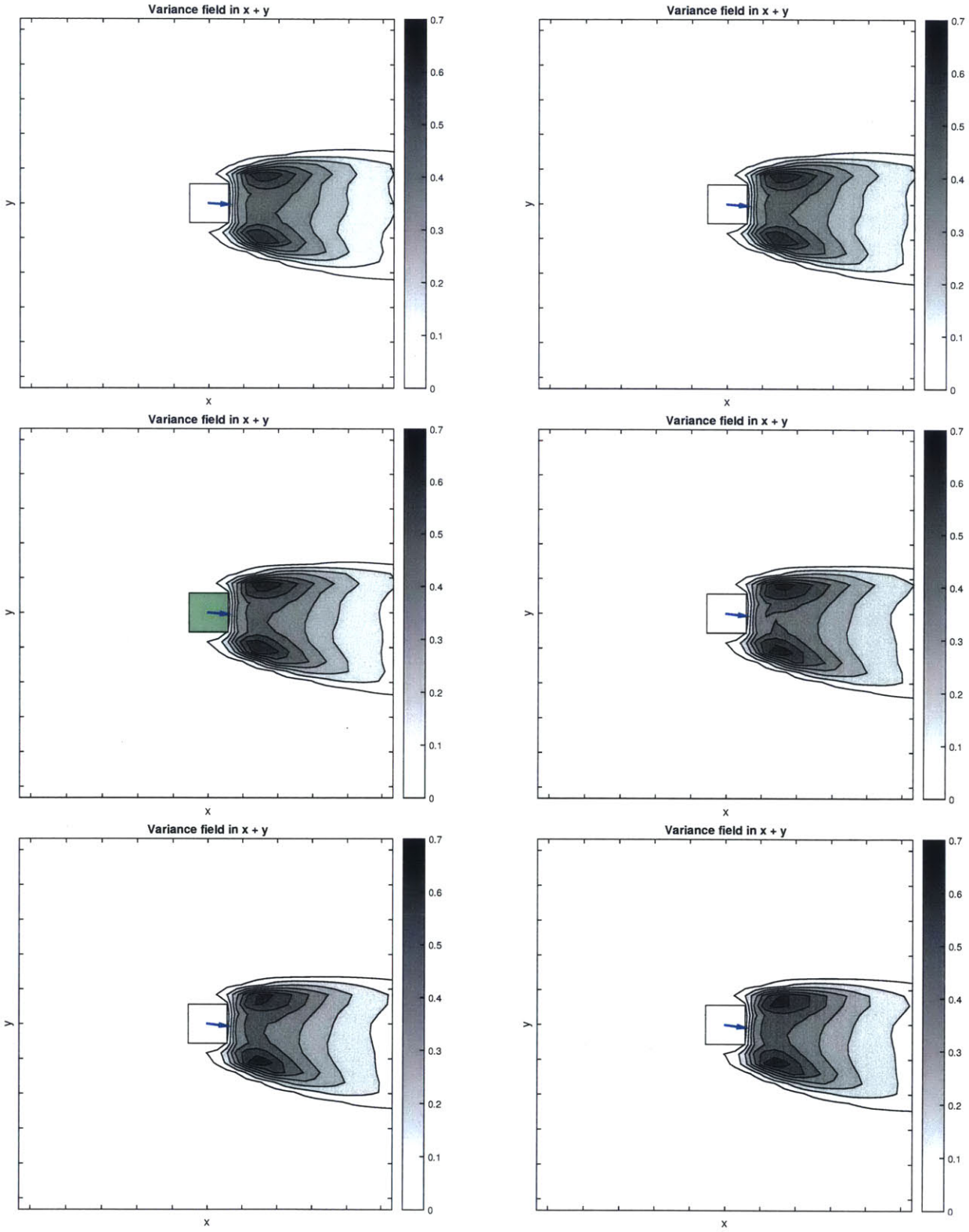


Figure 4-3: Linear interpolation of covariance matrices as a function of θ. The obstacle is green when the plot corresponds to a data point (Part II).

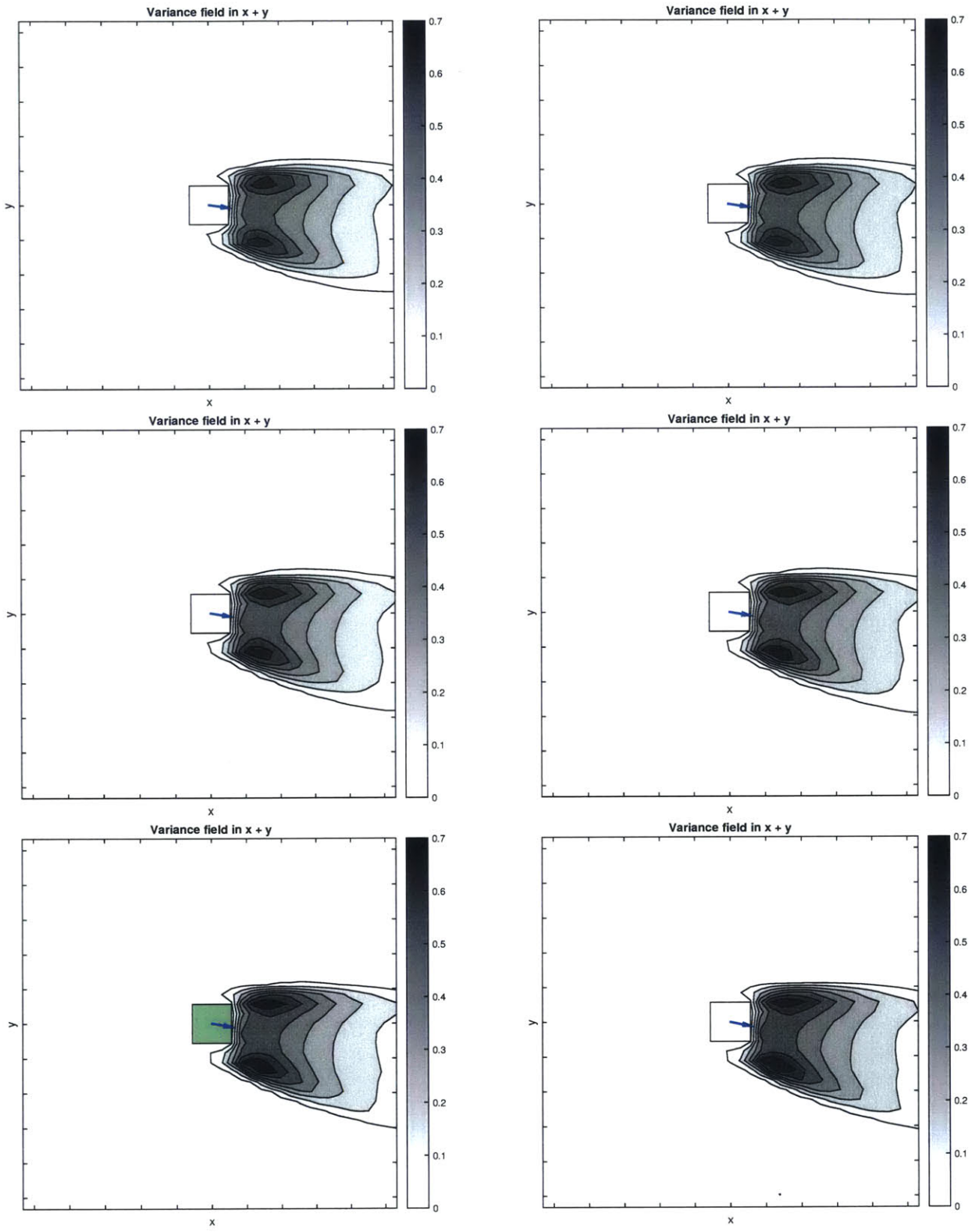


Figure 4-4: Linear interpolation of covariance matrices as a function of θ. The obstacle is green when the plot corresponds to a data point (Part III).

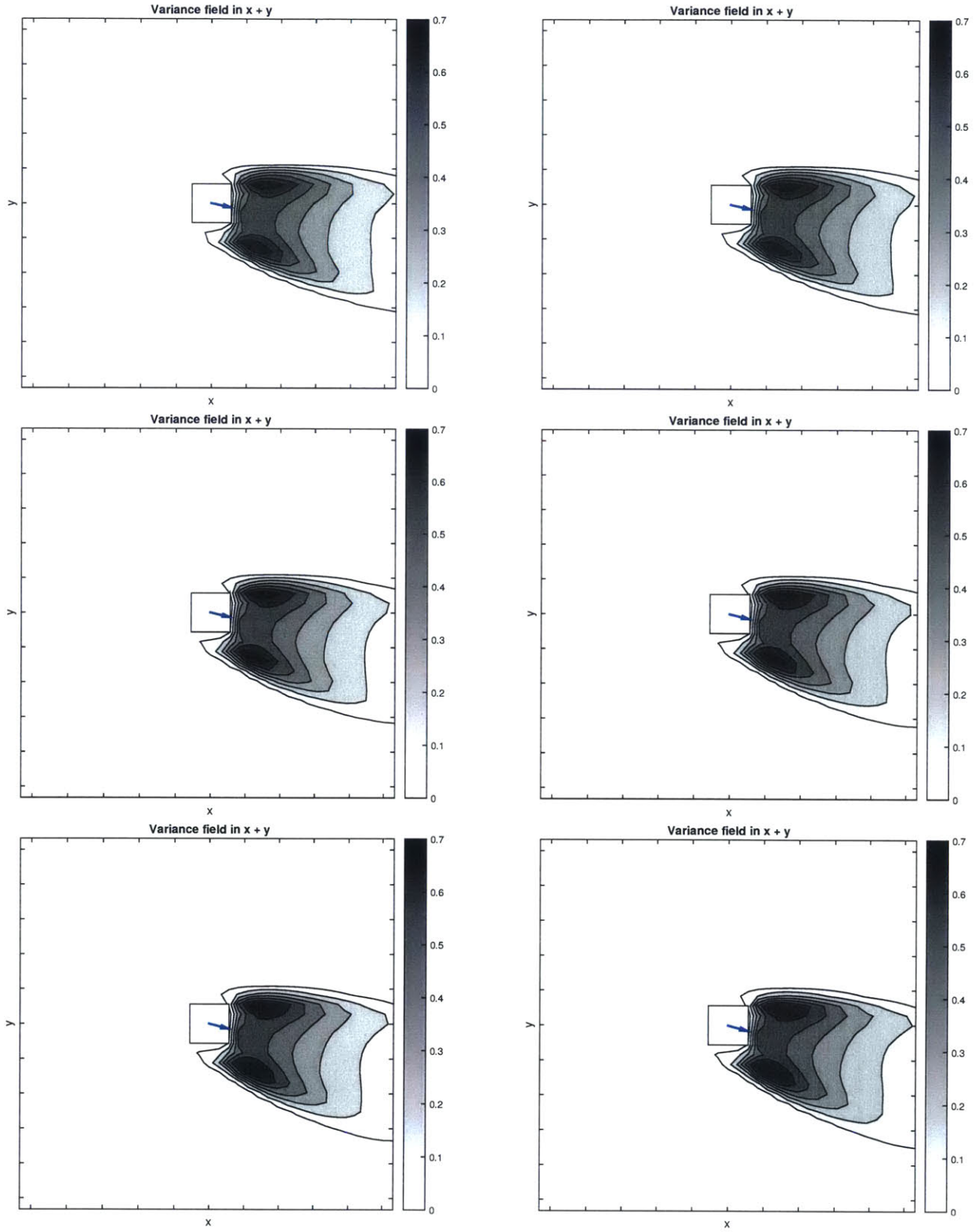


Figure 4-5: Linear interpolation of covariance matrices as a function of θ. The obstacle is green when the plot corresponds to a data point (Part IV).

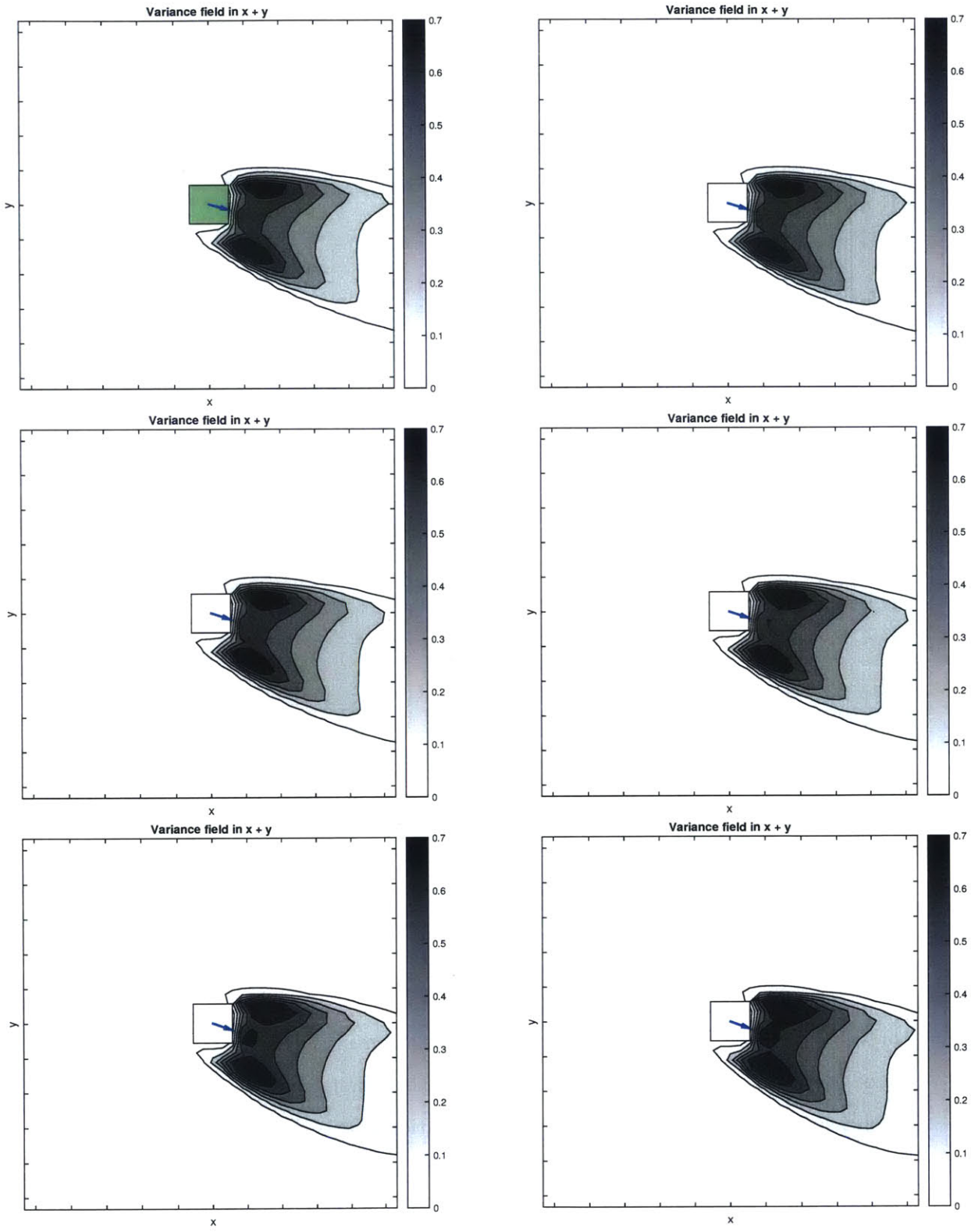


Figure 4-6: Linear interpolation of covariance matrices as a function of θ. The obstacle is green when the plot corresponds to a data point (Part V).

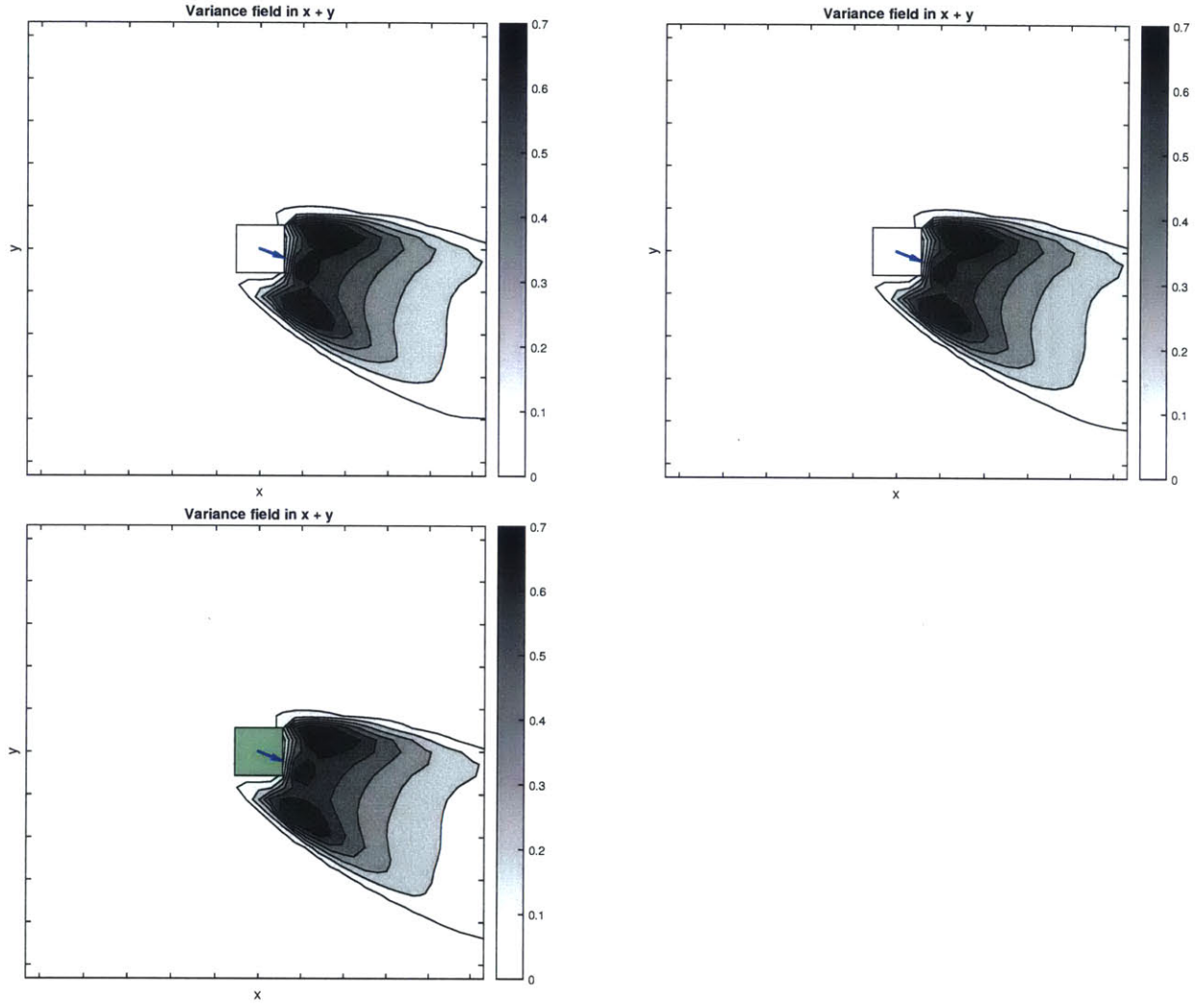


Figure 4-7: Linear interpolation of covariance matrices as a function of θ. The obstacle is green when the plot corresponds to a data point (Part VI).

4.1.3 Assimilation of dynamic data

Next, we show the results from the formulation in Section 3.3.2. The usefulness of the approach presented herein is two-fold. First, the KF allows extrapolating measurements in time. Second, our formulation allows changing the prevailing wind conditions simultaneously. Notice also in the last Figures of the following sequence that, when we do not have measurements, the wind field estimate naturally comes back to the prior wind field.

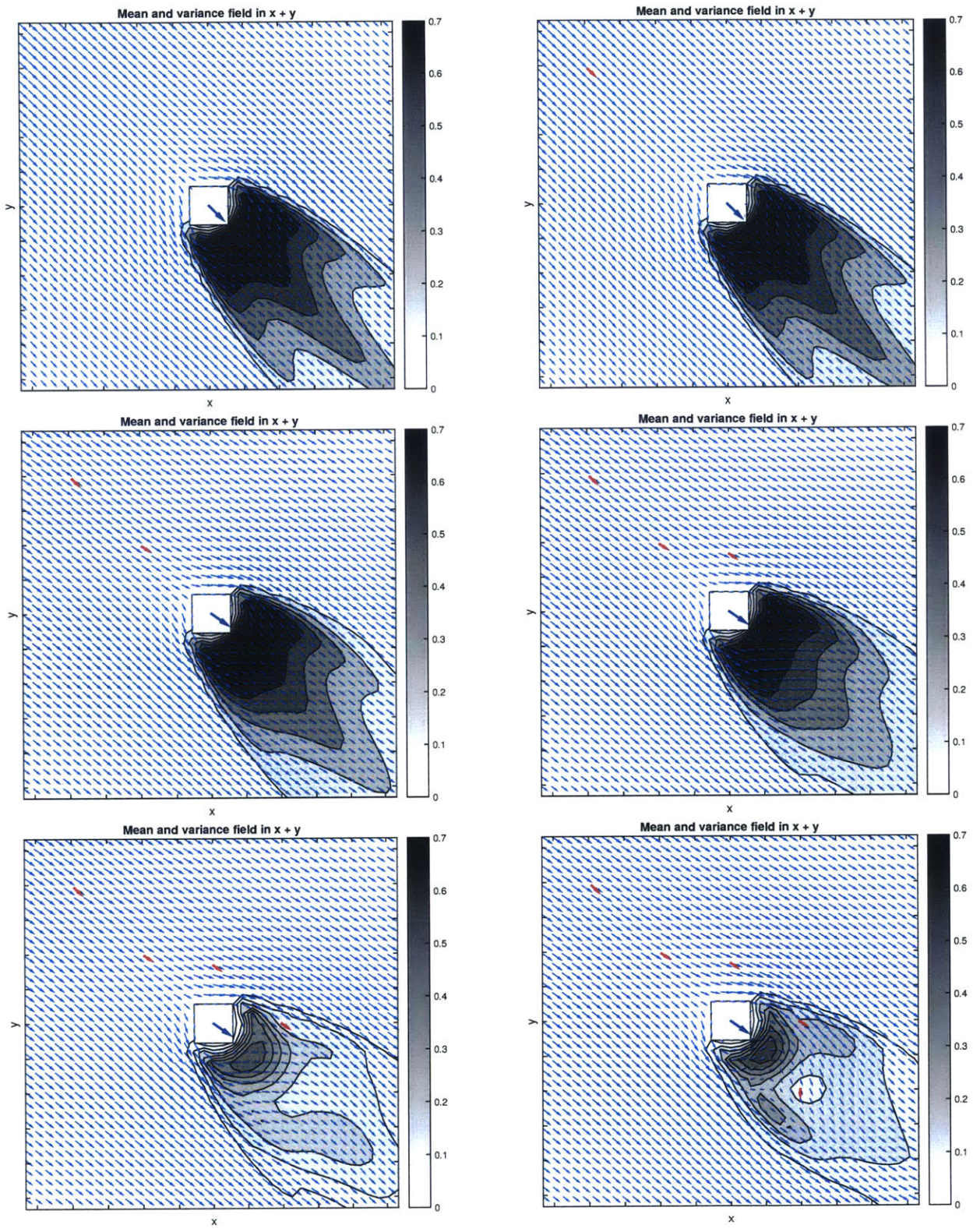


Figure 4-8: The KF enables assimilation of data as the UAV navigates. Not optimal path, just acquiring measurements for wind field estimation (Part I).

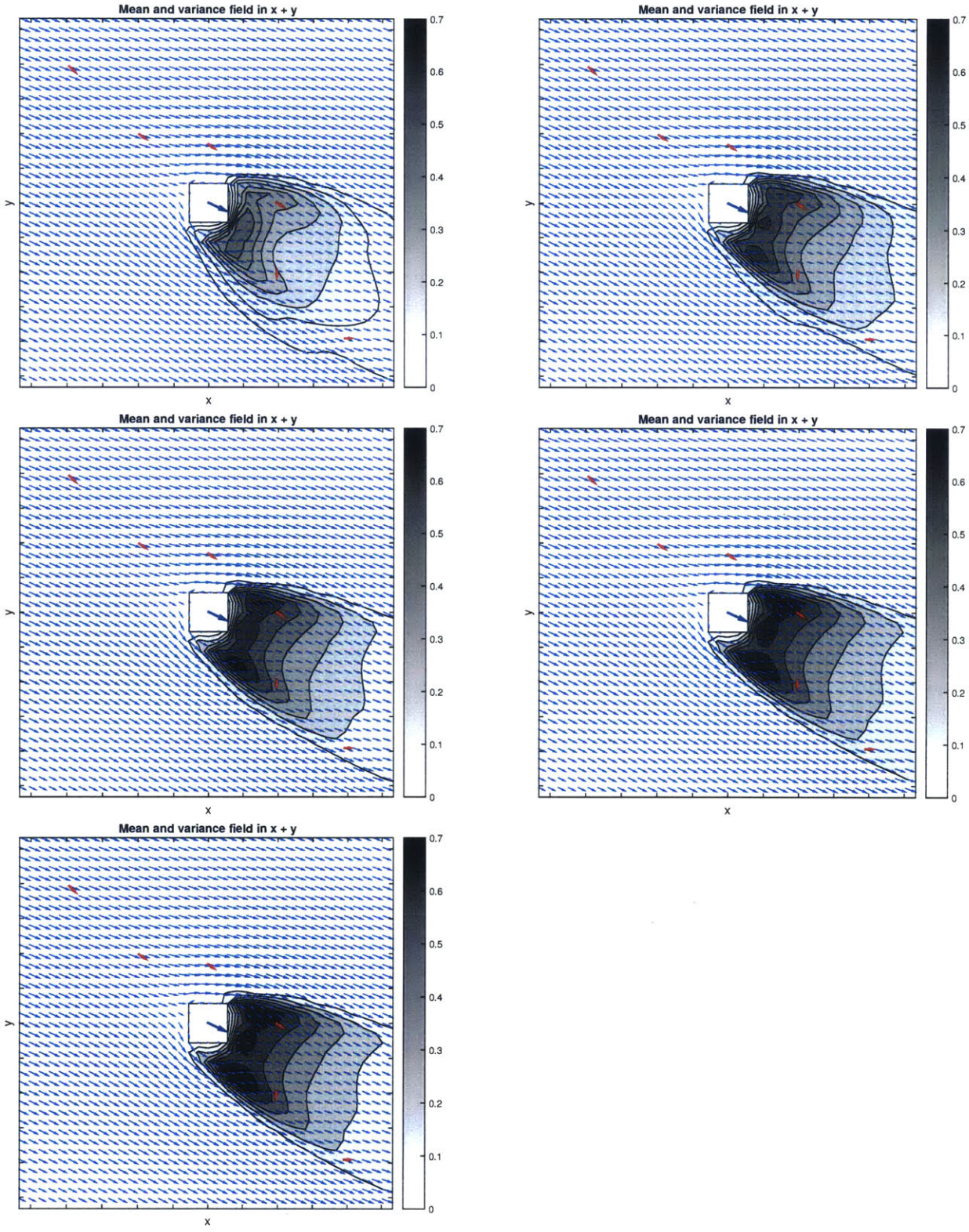


Figure 4-9: The KF enables assimilation of data as the UAV navigates. Not optimal path, just acquiring measurements for wind field estimation (Part II).

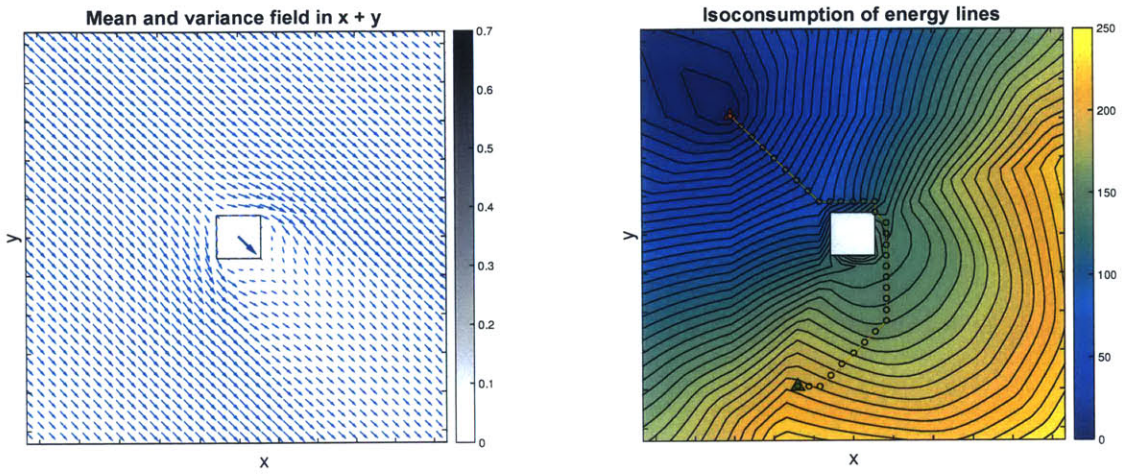


Figure 4-10: Mean of the random variable that defines the wind field (left) and isoconsumption of energy lines (right) for the certainty equivalence solution.

4.2 Wind field exploitation

In this section we analyze our wind field exploitation formulation described in Section 3.4. The solution is presented for three levels of stochasticity by reducing Equation 3.14 to (i) fully deterministic, (ii) be equivalent to the expectation, or (iii) fully stochastic.

4.2.1 Deterministic path planning

For this formulation, we consider the following reduced version of the costs:

$$\frac{g_k(i,u)}{\gamma} = \mathbb{E}_w[(V_{UAV} - V_w)^T(V_{UAV} - V_w)] \approx (V_{UAV} - V_w)^T(V_{UAV} - V_w).$$

4.2.2 Certainty equivalence

For this formulation, we consider the following reduced version of the costs:

$$\frac{g_k(i,u)}{\gamma} = \mathbb{E}_w[(V_{UAV} - V_w)^T(V_{UAV} - V_w)] \approx (V_{UAV} - \mathbb{E}[X_w])^T(X_{UAV} - \mathbb{E}[X_w]).$$

The results are shown in Figure 4-10.

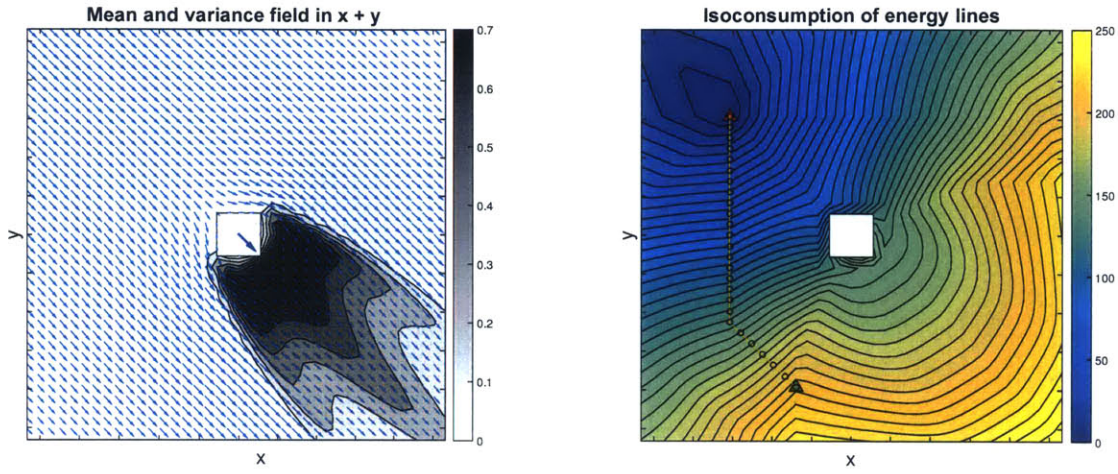


Figure 4-11: Mean and variance field of the random variable that defines the wind field (left) and isoconsumption of energy lines (right) for the stochastic solution.

4.2.3 Stochastic solution

For this formulation, we consider the following complete version of the costs:

$$\frac{g_k(i,u)}{\gamma} = \mathbb{E}_w[(V_{UAV}-V_w)^T(V_{UAV}-V_w)] = (V_{UAV}-\mathbb{E}_w[V_w])^T(V_{UAV}-\mathbb{E}_w[V_w])+\sum_{m=1}^{2}\mathrm{Var}[V_w^{(m)}].$$

The results are shown in Figure 4-11.

Notice that the deterministic solution is the only one that considers the realization of the wind field as opposed to the mean. Also, the stochastic solution is the only that considers the pointwise variance. In the zone right after the obstacle, the mean is zero but the variance is the highest due to vorticity. It is interesting to realize that in the certainty equivalence solution, we are obtaining costs that are lower than in reality (since the mean is close to zero). However, both the deterministic (uses a realization so the velocity after the obstacle is higher) and the stochastic (uses also the variance) would achieve similar results in that region but for completely different reasons.

Finally, Figure 4-12 shows the locus of points where the optimal path changes if we take into account the uncertainty in the wind field velocity.

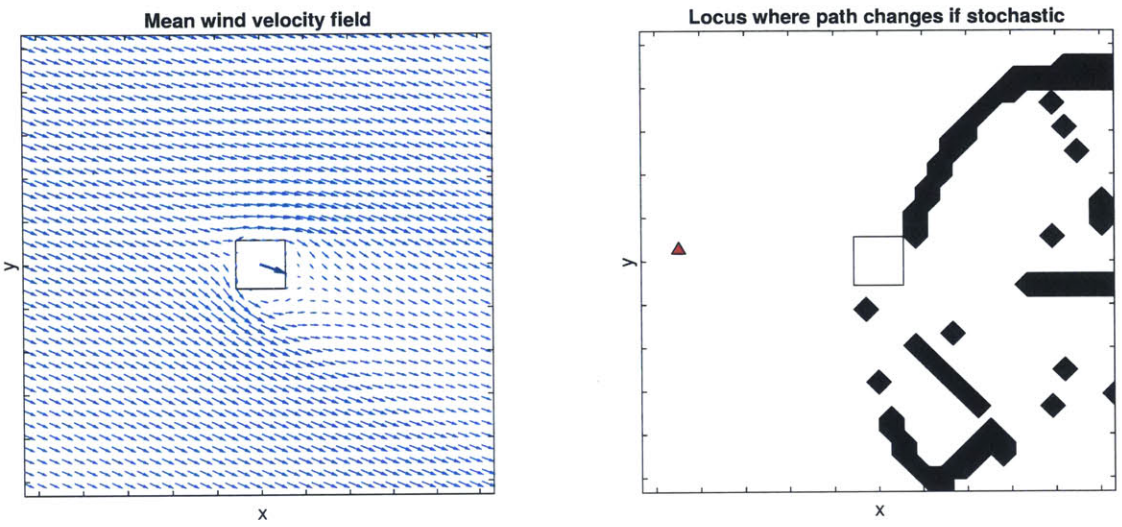


Figure 4-12: Mean wind velocity field (left) for a particular heading of the prevailing wind. Locus of points where optimal path to red triangle would change if we consider stochasticity

4.3 Integrated estimation and exploitation

Finally, as described in Section 3.5, we integrate wind field estimation with path planning. We use the formulation in Section 3.3 for wind field estimation and the formulation in Section 3.4 in its stochastic version for path planning.

In this section, we recompute the asynchronous value iteration for each step of the KF. In other words, the dynamic programming formulation is recomputed whenever we get new information about the wind field. We use as a first approximation for the value iteration algorithm (Section 3.4.4) the optimal policy in the previous step of the KF. Recycling the previous solution as a starting point for value iteration in the subsequent step renders faster convergence.

The solution of the path planning in a dynamic wind field is represented in the following sequence of Figures.

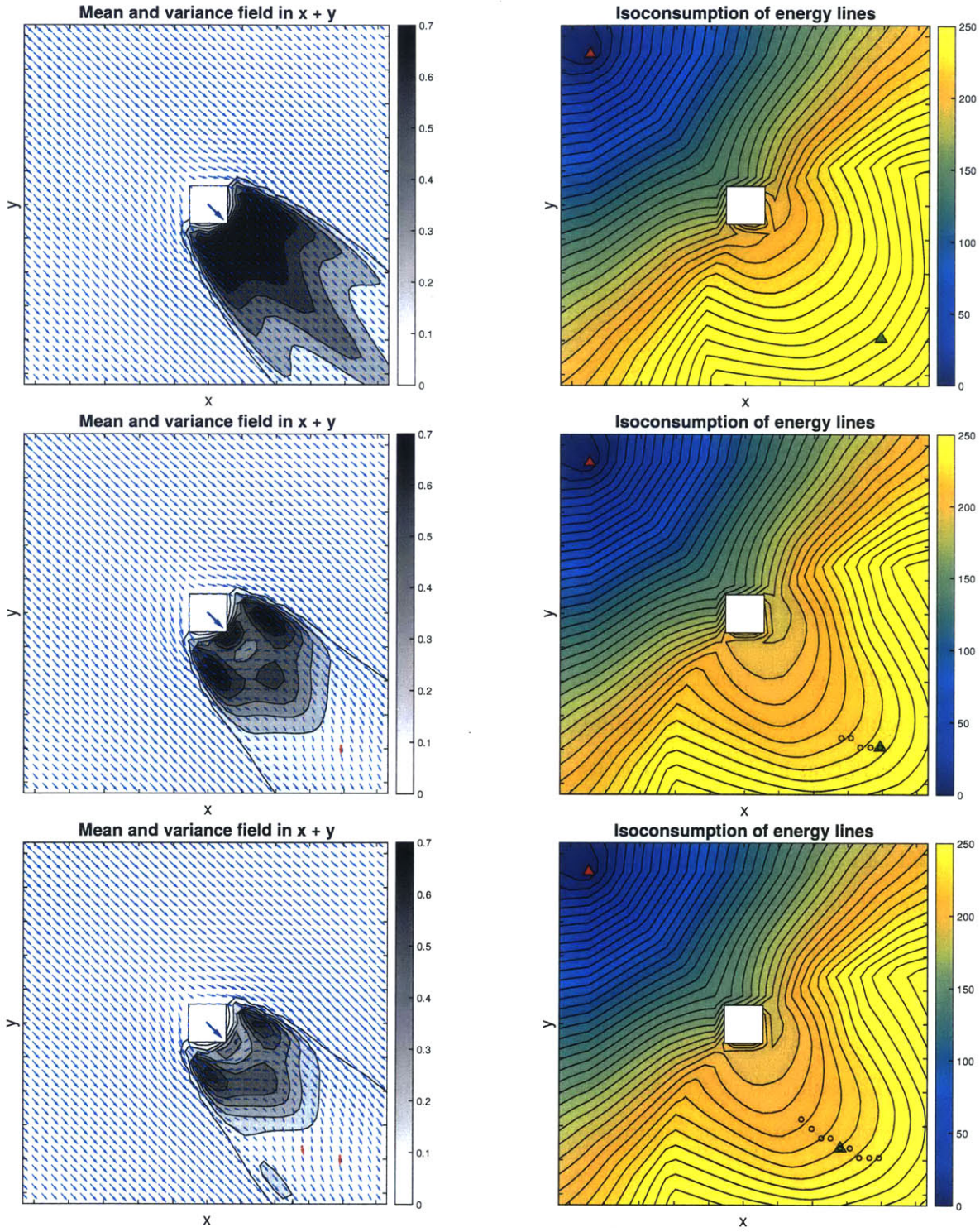


Figure 4-13: Mean and variance field of the random variable that defines the wind field (left). Isoconsumption of energy lines and optimal path (right) for the stochastic solution. The path is updated as the UAV acquires more information (Part I).

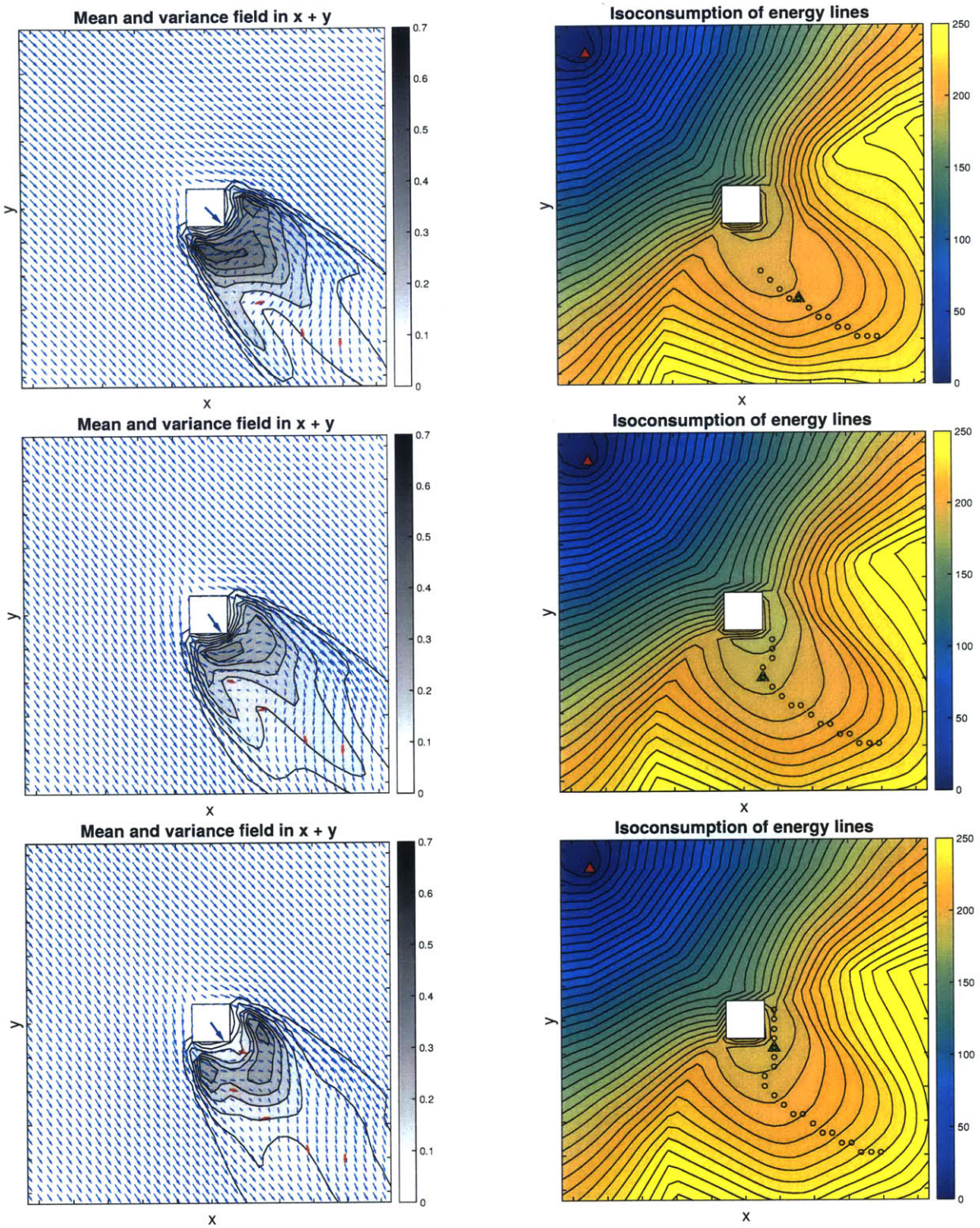


Figure 4-14: Mean and variance field of the random variable that defines the wind field (left). Isoconsumption of energy lines and optimal path (right) for the stochastic solution. The path is updated as the UAV acquires more information (Part II).

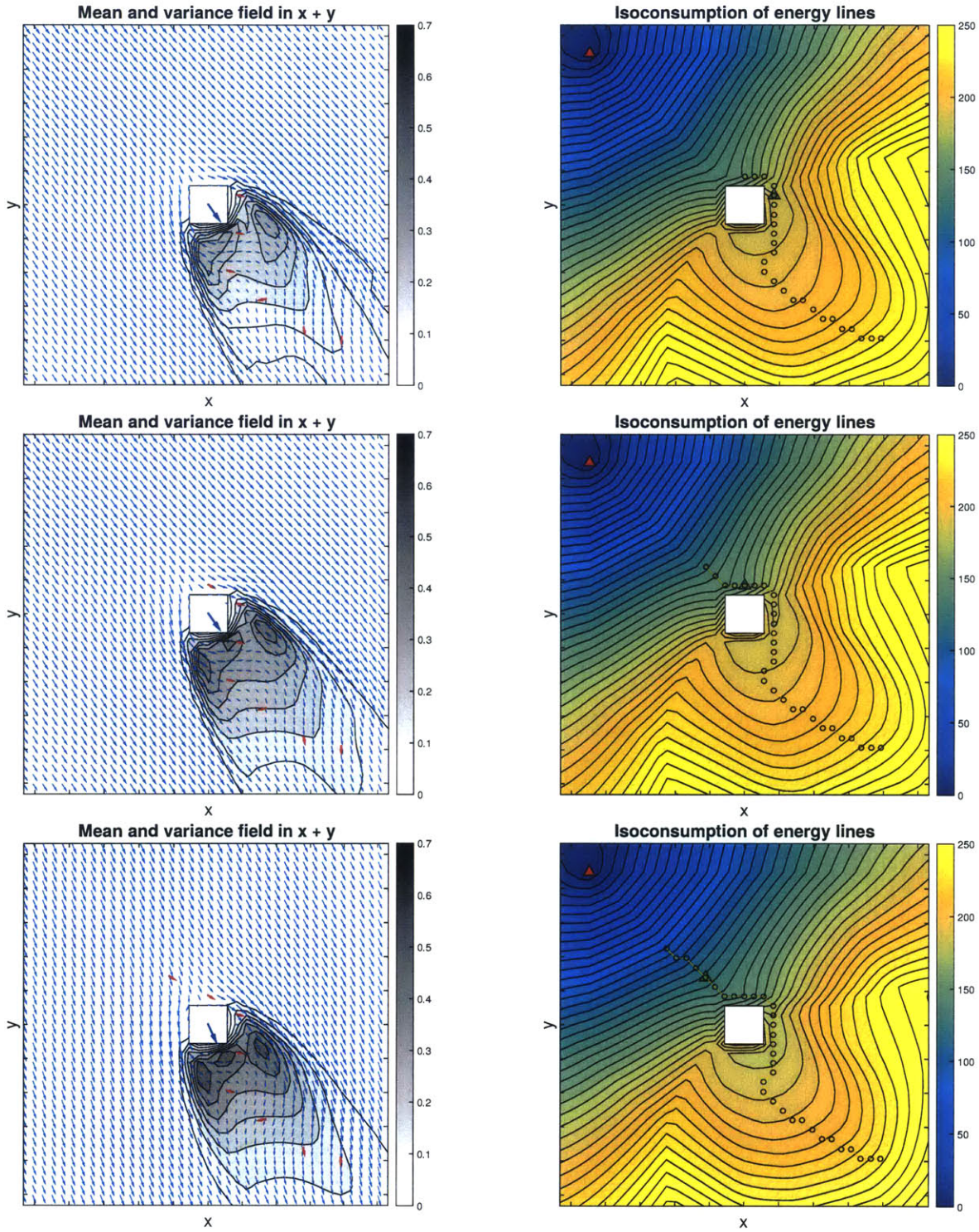


Figure 4-15: Mean and variance field of the random variable that defines the wind field (left). Isoconsumption of energy lines and optimal path (right) for the stochastic solution. The path is updated as the UAV acquires more information (Part III).

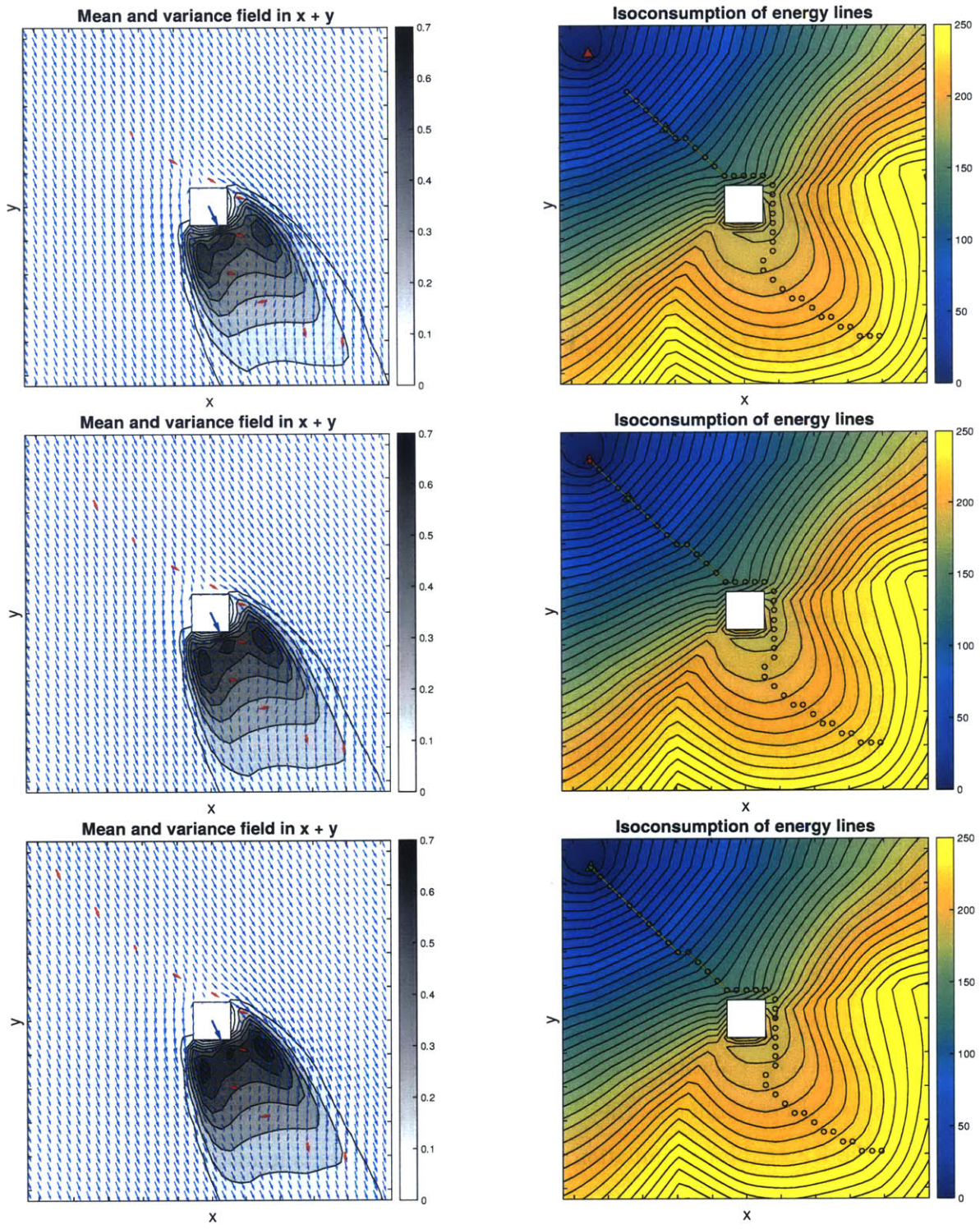


Figure 4-16: Mean and variance field of the random variable that defines the wind field (left). Isoconsumption of energy lines and optimal path (right) for the stochastic solution. The path is updated as the UAV acquires more information (Part IV).

Chapter 5

Conclusions and future work

5.1 Conclusions

In the present thesis, we have proposed a formulation to improve wind field navigation. As a testbed we have used an example that seeks a compromise between approximating as much as possible the reality while keeping simplicity.

In terms of wind field estimation, one of the key steps of the formulation is the interpolation of covariance matrices on matrix manifolds. The main conclusion is that interpolating matrices is feasible as long as it is done in the proper manifold. If we perform a KL expansion of the normal random variables that have our parameterized matrices as covariances and then we interpolate these low rank matrices, this procedure can become an extremely efficient way to store parameterized covariance matrices. Notice the importance of efficiency on the framework we propose, since it has to be used online.

Besides interpolation of low rank covariance matrices, we update our estimation of the wind field online as the UAV follows the optimized path. To do that, we saw that the KF enables assimilation of onboard measurements in an appropriate fashion and, in the absence of data, it recovers the wind field estimation corresponding to the present prevailing wind conditions.

In terms of wind field exploitation, we have considered three levels of stochasticity that correspond to computing the cost function using either a sample, just the

expectation, or fully stochastic. There are also three levels of lookahead when defining the cost function. First, we can treat the problem as stationary, which means that we are not getting any additional information of the wind field while the UAV is flying. Secondly, one can incorporate some onboard measurements to refine the state of the wind field and replan accordingly. The last and most precious would be to have a model that can predict future wind fields and plan taking into account future predictions.

Two main conclusions can drawn from the particular toy problem considered herein. First, it is possible to recompute the value iteration algorithm to solve the shortest path problem online. In particular, the solution in the previous state of the wind field can be used as initial guess for the algorithm. This approach renders fast convergence. Second, taking into account the wind field, the optimal path is no longer just the shortest distance. In other words, as we can see in the previous section, the shortest path is not trivially the least Euclidean distance.

5.2 Limitations

As described above, after generating the wind field library, we extract the mean and covariance matrix in a predefined regular grid. These statistics are then assumed to represent a multivariate Gaussian random variable. As mentioned previously, uncertainty of the modulus of the wind is usually treated with Rayleigh distributions, such distribution is obtained precisely by computing the modulus of two Gaussian random variables corresponding to the value of its components. However, some questions arise in this context. First, although in our context Gaussianity is assumed after conditioning to a given prevailing wind (i.e., creating a hierarchical Gaussian model), does this Gaussian approximation hold for a larger environment? We plan to run validations of the model by acquiring real data. Another question that needs to be addressed is how sparsely can we discretize the prevailing wind conditions. Finally, we plan to learn features to predict the locations for which the Gaussian approximation most closely matches the CFD estimate. Such features may allow obtaining regions

where we are more confident that our results hold.

As described above, one of the principal issues of the approach is the curse of dimensionality. Performing a Monte Carlo simulation requires a lot of samples. That is feasible for small domains but it is a challenge as the problem becomes larger.

Finally, we have to point out that there is not a preferred notion of distance in $S_+(r, N)$. Therefore, the geodesic on which we do the interpolation of covariance matrices must be chosen. This choice may be subjective and adds an undesired degree of freedom to the problem.

5.3 Future work

In terms of wind field estimation, although the methodology presented has been shown to be useful in these initial investigations, significant work remains in addressing the complete problem. The primary challenges will be in generating the necessary set of wind fields from a CFD solver in order to get the inputs for the covariance matrices for complex domains. In that scenario, incorporating the resulting wind field estimates into a trajectory planner could also be challenging in terms of computational cost.

We also plan to benchmark our results against actual in-situ data after defining a suitable success metric that incorporates the result of the planning research. However, local measurements within urban canyon tend to be noisy and updating the entire wind field from an instantaneous measurement might be very inaccurate. To quantify this issue, both direct-conditioning and KF allow incorporating errors in the measurements.

Regarding the potential lines of investigation on interpolation of matrices on manifolds, it would be interesting to include in this analysis higher order interpolation and regression. Moreover, the previous conclusions could be improved if we considered more explanatory variables. Another potential source of further research is to consider the same problem on other manifolds. Finally, we can conclude that this approach could be applied to multiple other fields. It has not been the purpose of the present thesis to discuss to what extent the conclusions that have been drawn might be exported to other areas; rather, we have focused on the specifics of how to interpolate

covariance matrices on matrix manifolds.

As a further research in terms of wind field exploitation, the main interest would be to reduce the sources of suboptimality and limitations of the solution presented herein. Among those, there are:

- Adding the dynamics of the UAV, for instance the Dubins equations.
- Refining control space by using a randomized policy that would be consistent, i.e., one that makes possible to recover the optimal path as we enlarge the grid.
- Adding forecasts of the wind field to be able to solve the complete problem and not only the one step lookahead.
- Applying the same formulation for a complex domain.

Beyond suboptimality, we will evaluate the usefulness of our approach by doing trajectory planning across a significant portion of the MIT campus. Then, we would like to use the resulting wind field estimate and compare the energy consumption of a planner aware of and naïve to the wind conditions.

Finally, as presented in the literature review, there exists the notion of wind field exploration. As described before, exploration consists of navigating with the primary goal of learning more features of the wind field online. We would like to address also this problem by changing the cost function in our wind field exploitation framework to some information theoretical measure, such as mutual information.

Bibliography

[1] Tofigh Allahviranloo. Successive over relaxation iterative method for fuzzy system of linear equations. *Applied Mathematics and Computation*, 162(1):189–196, 2005.

[2] David Amsallem, Julien Cortial, Kevin Carlberg, and Charbel Farhat. A method for interpolating on manifolds structural dynamics reduced-order models. *International Journal for Numerical Methods in Engineering*, 80(9):1241–1258, 2009.

[3] David Amsallem and Charbel Farhat. Interpolation method for adapting reduced-order models and application to aeroelasticity. *AIAA Journal*, 46(7):1803–1813, 2008.

[4] David Amsallem and Charbel Farhat. An online method for interpolating linear parametric Reduced-Order Models. *SIAM Journal on Scientific Computing*, 33:2169, 2011.

[5] Erlend Aune, Jo Eidsvik, and Yvo Pokern. Iterative numerical methods for sampling from high dimensional Gaussian distributions. *Statistics and Computing*, 23(4):501–521, 2013.

[6] Richard Bellman. On the theory of dynamic programming. *Proceedings of the National Academy of Sciences*, 38(8):716–719, 1952.

[7] Richard Bellman. Dynamic programming and lagrange multipliers. *Proceedings of the National Academy of Sciences*, 42(10):767–769, 1956.

[8] Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific Belmont, MA, 1995.

[9] Dimitri P. Bertsekas and John N. Tsitsiklis. An analysis of Stochastic Shortest Path Problems. *Math. Oper. Res.*, 16(3):580–595, August 1991.

[10] Silvere Bonnabel and Rodolphe Sepulchre. Riemannian metric and geometric mean for positive semidefinite matrices of fixed rank. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1055–1070, 2009.

[11] George Casella and Edward I George. Explaining the Gibbs sampler. *The American Statistician*, 46(3):167–174, 1992.

[12] Marco Cococcioni, Beatrice Lazzerini, and Pierre FJ Lermusiaux. Adaptive sampling using fleets of underwater gliders in the presence of fixed buoys using a constrained clustering algorithm. In *In Proc. of OCEANS'15, Genova, Italy, May 18-21*, 2015.

[13] Joseph L Doob. *Stochastic processes*, volume 101. New York Wiley, 1953.

[14] Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and computing*, 10(3):197–208, 2000.

[15] Jacques Faraut and Adam Korányi. *Analysis on symmetric cones*. Oxford university press, 1994.

[16] Wolfgang Förstner and Boudewijn Moonen. A metric for covariance matrices. In *Geodesy-The Challenge of the 3rd Millennium*, pages 299–309. Springer, 2003.

[17] Colin Fox and Albert Parker. Convergence in variance of Chebyshev accelerated Gibbs samplers. *SIAM Journal on Scientific Computing*, 36(1):A124–A147, 2014.

[18] Walter R Gilks and Pascal Wild. Adaptive rejection sampling for Gibbs sampling. *Applied Statistics*, pages 337–348, 1992.

[19] Gene H Golub and Richard S Varga. Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order Richardson iterative methods. *Numerische Mathematik*, 3(1):157–168, 1961.

[20] Sungkyu Jung, Armin Schwartzman, and David Groisser. Scaling-rotation distance and interpolation of symmetric positive-definite matrices. *SIAM Journal on Matrix Analysis and Applications*, 36(3):1180–1201, 2015.

[21] Chang-Jin Kim and Charles R Nelson. State-space models with regime switching: classical and Gibbs-sampling approaches with applications. *MIT Press Books*, 1, 1999.

[22] Jack W Langelaan, Nicholas Alley, and James Neidhoefer. Wind field estimation for small UAVs. *Journal of Guidance, Control, and Dynamics*, 34(4):1016–1030, 2011.

[23] Jack W Langelaan, John Spletzer, Corey Montella, and Joachim Grenestedt. Wind field estimation for autonomous dynamic soaring. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[24] Trenton Larrabee, Haiyang Chao, Matthew Rhudy, Yu Gu, and Marcello R Napolitano. Wind field estimation in uav formation flight. In *American Control Conference (ACC), 2014*, pages 5408–5413. IEEE, 2014.

[25] Nicholas RJ Lawrance and Salah Sukkarieh. Path planning for autonomous soaring flight in dynamic wind fields. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2499–2505. IEEE, 2011.

[26] NR Lawrance and Salah Sukkarieh. Simultaneous exploration and exploitation of a wind field for a small gliding UAV. *AIAA Guidance, Navigation and Control Conference, AIAA Paper*, 8032, 2010.

[27] Olivier P Le Maître and Omar M Knio. Introduction: Uncertainty quantification and propagation. In *Spectral Methods for Uncertainty Quantification*, pages 1–13. Springer Netherlands, 2010.

[28] T Lolla, PJ Haley Jr, and PFJ Lermusiaux. Path planning in multi-scale ocean flows: Coordination and dynamic obstacles. *Ocean Modelling*, 94:46–66, 2015.

[29] MS Lynn. On the equivalence of SOR, SSOR and USSOR as applied to σ1-ordered systems of linear equations. *The Computer Journal*, 7(1):72–75, 1964.

[30] Gilles Meyer, Silvère Bonnabel, and Rodolphe Sepulchre. Regression on fixed-rank positive semidefinite matrices: a riemannian approach. *The Journal of Machine Learning Research*, 12:593–625, 2011.

[31] Martin Fodslette Møller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural networks*, 6(4):525–533, 1993.

[32] Albert Parker and Colin Fox. Sampling Gaussian distributions in Krylov spaces with conjugate gradients. *SIAM Journal on Scientific Computing*, 34(3):B312–B334, 2012.

[33] Carl Edward Rasmussen. Gaussian processes for machine learning. 2006.

[34] Håvard Rue. Fast sampling of Gaussian Markov random fields. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):325–338, 2001.

[35] Havard Rue and Leonhard Held. *Gaussian Markov random fields: theory and applications*. CRC Press, 2005.

[36] Youcef Saad. Chebyshev acceleration techniques for solving nonsymmetric eigenvalue problems. *Mathematics of Computation*, 42(166):567–588, 1984.

[37] Arvind K Saibaba, Eric L Miller, and Peter K Kitanidis. Fast kalman filter using hierarchical matrices and a low-rank perturbative approach. *Inverse Problems*, 31(1):015009, 2015.

[38] Simo Särkkä. *Bayesian filtering and smoothing*, volume 3. Cambridge University Press, 2013.

[39] Daniel P Simpson, I Turner, and A Pettitt. Fast sampling from a Gaussian Markov random field using Krylov subspace approaches. *Scandinavian Journal of Statistics*, 2008.

[40] Steven Thomas Smith. Covariance, subspace, and intrinsic crame r-rao bounds. *Signal Processing, IEEE Transactions on*, 53(5):1610–1630, 2005.

[41] Alessio Spantini, Antti Solonen, Tiangang Cui, James Martin, Luis Tenorio, and Youssef Marzouk. Optimal low-rank approximations of bayesian linear inverse problems. *SIAM Journal on Scientific Computing*, 37(6):A2451–A2487, 2015.

[42] Kunio Tanabe and Masahiko Sagae. An exact Cholesky decomposition and the generalized inverse of the variance-covariance matrix of the multinomial distribution, with applications. *Journal of the Royal Statistical Society: Series B (Methodological)*, pages 211–219, 1992.

[43] Lloyd N Trefethen and David Bau III. *Numerical linear algebra*, volume 50. Siam, 1997.

[44] Bart Vandereycken, P-A Absil, and Stefan Vandewalle. A riemannian geometry with complete geodesics for the set of positive semidefinite matrices of fixed rank. *IMA Journal of Numerical Analysis*, page drs006, 2012.

[45] Christopher KI Williams and Carl Edward Rasmussen. Gaussian processes for machine learning. *the MIT Press*, 2(3):4, 2006.