Title
Data Assimilation in Large-scale Networks of Open Channels

Author
Rafiee Jahromi, Mohammad

Publication Date
2012

Peer reviewed|Thesis/dissertation

Data Assimilation in Large-scale Networks of Open Channels

by

Mohammad Rafiee Jahromi

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

Doctor of Philosophy

in

Engineering-Mechanical Engineering
and the Designated Emphasis
in
Communication, Computation and Statistics

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in charge:

Professor Alexandre M. Bayen, Co-Chair
Professor Andrew K. Packard, Co-Chair
Professor Ronald W. Yeung
Professor Laurent El Ghaoui

Fall 2012

Data Assimilation in Large-scale Networks of Open Channels

by

Mohammad Rafiee Jahromi

Abstract

Data Assimilation in Large-scale Networks of Open Channels

by

Mohammad Rafiee Jahromi

Doctor of Philosophy in Engineering-Mechanical Engineering
and the Designated Emphasis
in
Communication, Computation and Statistics

University of California, Berkeley

Professor Alexandre M. Bayen, Co-Chair
Professor Andrew K. Packard, Co-Chair

This dissertation is mainly focused on assimilation of data into hydrodynamic models of water flow in open channel networks, which is motivated by the need for accurate flow models in various applications such as emergency response and flood monitoring systems, automated gate systems and hydrological studies. We investigate application of different data assimilation techniques in different scenarios to incorporate the available flow measurements obtained from sensors into flow models to improve their accuracy.

Water flow in open channels is an instance of the so-called distributed parameters systems, in which the dynamics of the system is described by a set of partial differential equations. As the flow model, the Saint-Venant equations, also known as shallow water equations, which are a set of first-order hyperbolic nonlinear partial differential equations are used. Different practical scenarios are considered. In a case in which streaming measurements of the flow are available and real-time estimation of the flow state is desired, we present how standard state estimation techniques such as the Kalman filter, the Extended Kalman filter and the Unscented Kalman filter can be applied to integrate the available measurements into the shallow water equations. It is also shown how these techniques can be adapted to a case in which some of the model parameters are unknown to estimate the unknown parameters along with the state of the system.

For data assimilation in large-scale networks which lead to high dimensional models, application of two sequential Monte Carlo methods, the optimal sampling importance resampling and the implicit particle filters, are considered. The computational cost of propagating each particle is higher in implicit particle filters, however, they provide more accurate results with smaller number of particles by choosing the particles in a way that they belong to the high probability regions of the posterior density function. We also propose a maximum-a-posteriori-based method to perform the state estimation, which is shown to perform better in terms of both accuracy and computational cost for the application of interest.

For flow estimation in tidally influenced channels, an efficient estimation method that takes advantage of spectral decomposition of the state is proposed. The estimation problem is formulated as a least squares regression with an l_1-norm regularization, known as the LASSO, and a homotopy-based algorithm is implemented to solve the resulting optimization problem recursively as new measurements become available.

Finally, we consider the problem of optimal topology design in multi-agent systems for efficient average consensus. The network design problem is posed in two different ways. (1) Assuming that the maximum communication cost, i.e. the maximum number of communication links, is known, the goal is to find the network topology which results in the fastest convergence to the consensus (in presence of communication time delays on the links). (2) If a minimum performance of the protocol is required, the design problem is posed as finding the network with lowest possible communication cost which fulfills the required performance. The design problem is formulated as an optimization problem which is finally transformed to a mixed integer semidefinite program.

To my parents...

Contents

Chapter 1

Introduction

1.1 Central topic of the dissertation

Data assimilation is the process of integrating observations or measurements into a mathematical model of a physical system, in order to estimate some quantities of interest. Recently, data assimilation has provided rapid advances in geosciences such as meteorology, oceanography and hydrology [1, 3, 7, 69, 20, 24, 48, 81, 90]. Different methods for assimilating data include variational data assimilation [8, 67, 35, 74, 17], filtering-based methods [50, 73, 86, 117, 115, 118, 16, 89], optimal statistical interpolation [85], or the Newtonian relaxation [65, 95].

Water flow in open channels is an example of the so-called *distributed parameters systems* in which the dynamics of the system can be modelled by a set of *partial differential equations* (PDEs). These systems are also called infinite dimensional systems in that the state of the system is a function of both time and a continuum of space. Among PDEs, a specific class, called conservation laws [41, 88], are most commonly used to describe physical systems. As the name suggests, these models are obtained from conservation laws governing the physics of the system. For modelling the water flow in rivers and open channels, the Saint-Venant equations, which are a set of coupled first-order hyperbolic nonlinear PDEs are commonly used [40, 2, 77]. These PDEs are obtained from conservation of mass and conservation of momentum and are usually derived for two cases of one-dimensional and two-dimensional models. In the one-dimensional case, the state of the system is the discharge (flow) and the stage (water depth) or equivalently the average flow velocity and the stage. In other words, a one-dimensional model describes the time evolution of discharge and stage everywhere along a channel. A two-dimensional model has the two components of flow velocity vector on the water surface and the stage as its state. One-dimensional and two-dimensional models are both used extensively in hydraulics. In this thesis, we focus on data assimilation and estimation of one-dimensional models.

Analytical solutions to Saint-Venant equations do not exist except in some particular cases due to the strong nonlinearities in the momentum equation [77, 36]. These equations are usually solved numerically using different discretization schemes. Solving the PDEs requires a knowledge of the initial and boundary conditions. An accurate knowledge of the initial conditions, the values of the state everywhere along the channel at the initial time, is not typically possible. However, the inaccuracies in the assumed initial conditions only affect the initial portion of the solution and if the PDEs are solved for a long enough period of time, the effect of these inaccuracies are washed away as the time evolves. The boundary conditions, which are the time series of the values of the flow variables at the open boundaries of the domain of interest, are usually obtained from sensors measuring the corresponding flow variables at the boundaries.

Due to different sources of error, the solution obtained from solving the PDEs numerically does not exactly match the true state of the system. The error in the solution arises from different sources such as the modelling assumptions and simplifications, inaccurate knowledge of the model parameters and bathymetry, errors introduced by numerical methods for solving the PDEs and the measurement noise in the measurements of the boundary conditions. When additional observations (measurements) of the system are available, it is desirable to take advantage of these measurements to improve the model. It is important to note that these measurements typically contain noise and uncertainty as well and are not to be assumed as perfectly accurate. However, they still provide additional information about the state of the system which can be used to improve the model results. In this dissertation, we consider different scenarios in which some measurements of the flow in an open channel system are available and we propose different data assimilation techniques to incorporate these measurements into a flow model to reduce the mismatch between the values computed by the model and the actual system throughout the whole domain of interest.

As for the sensors measuring the flow quantities, in addition to traditional static sensors, we consider cases in which Lagrangian measurements of the flow are available. *Lagrangian measurements* are measurements of the flow properties at a point moving with the flow along the streamline whereas *Eulerian measurements* are measurements of the flow properties at a fixed location. Lagrangian sensors which move with the flow and report their location and possibly other local quantities of interest (temperature, salinity, etc.) are commonly used in oceanography [9, 51, 105] (usually referred to as *drifters*) and in river hydraulics [4, 112, 106, 107, 16, 17]. Lower production and maintenance cost, as well as flexibility in deployment, are the main advantages of the drifters over the traditional static sensors. These drifters are equipped with GPS receivers and report their position, velocity and other measurements at every time step. The bathymetry at the corresponding cross-section can be used to estimate the discharge from the drifter velocity which is assumed to be equal to the local flow velocity.

The data assimilation methods proposed are implemented on different channels and networks of open channels in Sacramento-San Joaquin Delta in northern California and an artificial channel located on the eastern border of Carl Blackwell Lake, in Stillwater, Oklahoma. The Sacramento-San Joaquin Delta in northern California is a complex network of over 1150 km of tidally influenced channels and sloughs covering 738,000 acres of land and is of great significance in California as it is the main source of drinking water in the

state. It is also the source of irrigation of most of California's farmland. One of the main challenges that the *California Department of Water Resources* (DWR) faces in the Delta today is maintaining a sound levee system to protect the islands, towns and farms in the Delta and the freshwater supply streaming through the Delta. Since 1980, 18 Delta islands have been partially or completely flooded and recent studies have found that Delta levees are deteriorating. Water salinity is another major issue in the Delta which affects the potability of drinking water supplies, the quality of farmland irrigation supplies, and the aquatic ecosystems in the Delta. During the dry periods of the year, salty water propagates from the ocean to the Delta which makes the water impotable leading to the purchase of drinking water from other sources at a much higher cost. Finally, another issue is the Delta's ecosystem. In the fall of 2004, fish surveys reported rapid declines in several pelagic species such as striped bass, green and white sturgeon, perch, Chinook salmon, threadfin shad and delta smelt [84]. In dealing with all of these issues and other issues that arise in open channels and irrigation systems, having access to an accurate flow model is very beneficial. For instance, a real-time model of the flow in a network can help to respond to emergency conditions, such as a gate malfunction or a flood, properly by monitoring the state of the system and taking preemptive measures against possible damages.

In chapter 7, we shift gears to a problem in consensus theory. The motivation is the application of distributed and consensus-based algorithms [10], [15], [33] to fuse sensor data which can potentially be utilized in experiments with a large number of sensors (drifters) deployed. More precisely, the problem considered in the chapter is how to optimally design the communication network topology in multi-agent systems with a decentralized communication architecture to achieve consensus among the agents as efficiently as possible. More precisely, noting that the communication network can be represented by an algebraic graph in which the nodes represent the agents, the edges represent the communication links and the weights on the edges represent the weights used in the consensus algorithm, the goal is to determine a *Laplacian matrix* which corresponds to a network which results in converging to the consensus most efficiently. We define the efficiency in two ways: (1) Assuming that the maximum communication cost, i.e. the maximum number of communication links, is known, the design objective is to find the network topology which results in the fastest convergence to the consensus (in presence of communication time delays on the links). (2) If a minimum performance of the protocol is required, the design problem can be posed as finding the network with lowest possible communication cost which fulfills the required performance. In both approaches, we formulate the problem of finding the optimal communication graph among a class of directed graphs, *strongly balanced* digraphs, as a *Mixed Integer Semidefinite Program* (MISDP). By solving this MISDP, the optimal graph and the weights on communication links are obtained.

1.2 Contributions of the dissertation

The contributions of this dissertation are as follows:

- Construction of state-space models for networks of open channels using shallow water equations and consistency constraint at the junctions.
- Application of standard filtering techniques, the *Kalman filter*, the *extended Kalman filter*, the *unscented Kalman filter* to perform state and parameter estimation for open channel flow using drifter (Lagrangian sensor) data.
- Application of sequential Monte Carlo methods, the *optimal sampling importance resampling* and recently-developed *implicit particle filters* to perform state estimation in large-scale open channel networks and comparison of their performance and development of a *maximum a posteriori* approximate method to perform the state estimation with continuous and intermittent measurements.
- Development of an efficient estimator for real-time flow estimation in tidal channels using the recursive *LASSO*.
- Formulating the optimal communication network topology design for efficient average-consensus in multi-agent system as a *Mixed Integer Semidefinite Program* (MISDP).

1.3 Organization of the dissertation

In chapter 2, we present the one-dimensional shallow water equations, also known as the Saint-Venant equations. It is shown how these equations can be discretized and state-space models can be constructed to describe the flow in networks of open channels. In chapter 3, we show how different standard filtering-based state estimation methods, such as the Kalman filter, the Extended Kalman filter and the Unscented Kalman filter can be used to estimate the flow state using the available measurements. In chapter 4, we present an experiment which was performed on an artificial channel adjacent to Carl Blackwell Lake, in Stillwater, Oklahoma on November 10, 2009 in which a number of drifters were deployed to measure flow velocities in the case of simulating a levee break. The experiment data were used to investigate how the state estimation methods can be adapted to estimate unknown model parameters along with the state of the system in real time using the available flow measurements. In chapter 5, we investigate application of Monte Carlo-based techniques to assimilate data into models of large-scale networks of open channels. To perform data assimilation for such high dimensional systems, it is important to use techniques that are computationally tractable so that they can be implemented in real time. In particular, we implement a particle filter, the optimal sampling importance resampling filter, to assimilate data into the shallow water equations. We show how a recently-developed Monte Carlo method, the implicit particle filter, can be applied to the problem of interest to produce better results. In this chapter, we also present a few heuristic methods to perform the data assimilation with better accuracy and lower computational cost in the case of application of interest. Chapter 6 is concerned with a case in which estimates of flow variables are desired at a specific location in a tidal channel when observations of flow are available at other locations along the channel. After deriving a z-domain transfer function representation of Saint-Venant

equations, the estimation problem is posed as a parametric input estimation problem. We formulate the resulting estimation problem as a least squares with an l_1-norm regularization, also known as the LASSO, which we solve recursively using a homotopy-based algorithm. In chapter 7, the problem of optimal network topology design in in multi-agent systems for efficient average consensus is considered. The design problem as an optimization problem and after algebraic manipulation of the constraints and application of duality theory and optimality conditions, the optimization problem is formulated as a *Mixed Integer Semidefinite Program* (MISDP) which can be solved using over-the-shelf optimization packages such as YALMIP and SEDUMI. Finally, the dissertation is summarized in the concluding chapter 8. The results presented in chapters 3 and 4 were published in conference papers [96, 100] and a journal article [111]. The research on the application of Monte Carlo methods and the implicit particle filters were published in a conference paper [97] and a journal article under review [98]. Finally, the results of chapter 6 is part of the submitted journal article [62]. Finally, the results presented in chapter 7 were published in a conference paper [99].

Chapter 2

One-Dimensional Model of Flow in Open Channels

Open channel flow can be modelled by so-called *distributed parameters systems*. A physical system which is modelled by a set of *partial differential equations* (PDE) is called a distributed parameters system. In this chapter, we present the flow model used in the applications that will be presented in later chapters. For most of the applications considered in later chapters, a discrete-time state space model of the flow in a channel or a network of channels is needed. Here, we also explain how the governing partial differential equations can be discretized and consistency constrains at the junction can be used to construct a discrete-time state space model of the flow for a network of open channels. For the case of state estimation using the Kalman filter, presented in chapter 3, a linear state-space model desired. In the current chapter, we focus on construction of a nonlinear state-space model and we postpone the construction of the linear state space model to chapter 3 where it is used.

2.1 Saint-Venant Model

In unsteady flow, velocities and stage (depth) at any location in a channel vary with time. In one dimensional flow models, discharge and stage are functions of space and time and therefore they are considered as the dependent variables. This means that two PDEs are required to fully described the flow. These equations can be derived from conservation of mass and conservation of momentum. This yields two coupled first-order hyperbolic PDEs which are called the *Saint-Venant equations*. The Saint-Venant model is one of the most common models used for modeling the flow in open channels and irrigation systems [40], [2], [108].

The one dimensional Saint-Venant equations are derived with the following assumptions:

- The flow is one dimensional, i.e. the velocities are uniform and the water level across the cross sections is horizontal.
- The vertical accelerations are negligible and the pressure is hydrostatic.
- The channel bed slope is small so that the cosine of the angel that the bottom of the channel makes with the horizon is almost equal to unity and the sine of the angel can be approximated with the tan.
- The frictional bed resistance is the same as that of steady flow so that the Manning or Chezy equations can be used to approximate the mean boundary shear stress.
- The channel bed is stable and the bed elevations don't change with time.

While in a general case, lateral inflow and outflows can be considered, we assume there is no lateral inflow or outflow.

For prismatic channels, the Saint-Venant equations can be written as follows [108]:

$$T\frac{\partial H}{\partial t} + \frac{\partial Q}{\partial x} = 0 \tag{2.1}$$

$$\frac{\partial Q}{\partial t} + \frac{\partial}{\partial x}\left(\frac{Q^2}{A}\right) + \frac{\partial}{\partial x}(gh_cA) = gA(S_0 - S_f) \tag{2.2}$$

for $(x,t) \in (0,L) \times \Re^+$, where

- L is the river reach (m).
- $Q(x,t)$ is the discharge or flow ($^{m^3}/_{\mathrm{s}}$) across cross section $A(x,t) = T(x)H(x,t)$.
- $H(x,t)$ is the stage or water-depth (m).
- $T(x)$ is the free surface width (m).
- $D = A/T$ is the hydraulic depth (m).
- $S_f(x,t)$ is the friction slope ($^{\mathrm{m}}/_{\mathrm{m}}$).
- S_b is the bed slope ($^{\mathrm{m}}/_{\mathrm{m}}$).
- h_c is the distance of the centroid of the cross section from the free surface (m), see Fig. 2.1.
- g is the gravitational acceleration ($^{\mathrm{m}}/_{s^2}$).

The friction slope is empirically modelled by the Manning-Strickler's formula [77]:

$$S_f = \frac{m^2 Q^2 P^{4/3}}{A^{10/3}} \tag{2.3}$$

with $Q(x,t) = V(x,t)A(x,t)$ the discharge across cross-section $A(x,t)$, P the wetted perimeter, i.e. the perimeter of the wetted portion of the cross-section (see Fig. 2.1), and m the Manning's roughness coefficient ($sm^{-1/3}$).

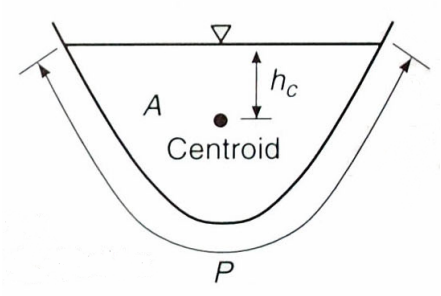


Figure 2.1: Illustration of the wetted perimeter P and h_c.

In the case of sub-critical flow, the boundary conditions are taken to be upstream flow $Q(0,t)$ and downstream stage $H(L,t)$ or vice versa [77].

For channels with non-rectangular cross-sections, three correction parameters, α, η and γ can be introduced through the following equations, $A = \alpha TH$, $P = \eta(2T + H)$ and $h_c = \gamma H$. These parameters are calculated based on the average stage.

2.2 Discretization

We use the Lax diffusive scheme [37], [108] which is a first-order explicit scheme to discretize the equations at internal grid points. Using f to represent the state variables, Q and H, the derivatives become

$$\frac{\partial f}{\partial t} = \frac{f_i^{k+1} - \frac{1}{2}(f_{i+1}^k + f_{i-1}^k)}{\Delta t} \tag{2.4}$$

$$\frac{\partial f}{\partial x} = \frac{(f_{i+1}^k - f_{i-1}^k)}{2\Delta x} \tag{2.5}$$

using traditional finite difference discretization notation, with subscript i for space and superscript k for time.

Applying this scheme to equations (2.1) and (2.2), we obtain the following set of finite difference equations,

$$A_i^{k+1} = \frac{1}{2}(A_{i-1}^k + A_{i+1}^k) - \frac{\triangle t}{2\triangle x}(Q_{i+1}^k - Q_{i-1}^k) \tag{2.6}$$

$$Q_i^{k+1} = \frac{1}{2}(Q_{i-1}^k + Q_{i+1}^k)$$

$$- \frac{\triangle t}{2\triangle x}\left[\left(\frac{Q^2}{A} + gAh_c\right)_{i+1}^k - \left(\frac{Q^2}{A} + gAh_c\right)_{i-1}^k\right] \tag{2.7}$$

$$+ \triangle t\left(\frac{\phi_{i+1}^k + \phi_{i-1}^k}{2}\right) \tag{2.8}$$

where

$$\phi = gA(S_b - S_f) \tag{2.9}$$

This scheme is stable provided that the *Courant-Friedrich-Lewy* (CFL) condition holds, i.e.

$$\frac{\Delta t}{\Delta x}|V + C| \leq 1 \tag{2.10}$$

where $C = \sqrt{gD}$ is the *wave celerity* and V is the average velocity.

However, the equations above may only be used for interior grid points. At the boundaries, these equations cannot be applied since there is no grid point outside the domain. Therefore, another method needs to be used to compute the unknown variables at the boundaries. Here, we use the *method of specified time intervals* to compute these variables [37]. In this method, after computing the characteristics, the boundary grid point is projected backward to the previous time step along its corresponding characteristic curve. After computing the variables at the projected point, which is usually done by using linear interpolation, the characteristic equations are used to compute the unknown variable at the boundary grid point at the next time step.

Denoting the projected points corresponding to the upstream and downstream boundary conditions by subscripts L and R, respectively, the resulted update equations for upstream stage and downstream velocity read as follows [37]

$$H_1^{k+1} = H_L^k + \frac{C_L^k}{g}(V_1^{k+1} - V_L^k) + C_L^k \Delta t(S_{f_L}^k - S_{b_1}) \tag{2.11}$$

$$V_N^{k+1} = V_R^k + g\frac{H_R^k - H_N^{k+1}}{C_R^k} - g\Delta t(S_{f_R}^k - S_{b_N}) \tag{2.12}$$

where the velocity and wave celerity at projected points can be calculated as

$$V_L^k = \frac{V_1^k + \beta(C_1^k V_2^k - C_2^k V_1^k)}{1 + \beta(-V_1^k + V_2^k + C_1^k - C_2^k)} \tag{2.13}$$

$$C_L^k = \frac{C_1^k + \beta V_L^k (C_1^k - C_2^k)}{1 + \beta(C_1^k - C_2^k)} \tag{2.14}$$

$$V_R^k = \frac{V_N^k + \beta(C_N^k V_{N-1}^k - C_{N-1}^k V_N^k)}{1 + \beta(-V_N^k + V_{N-1}^k + C_N^k - C_{N-1}^k)} \tag{2.15}$$

$$C_R^k = \frac{C_N^k + \beta V_R^k (C_{N-1}^k - C_N^k)}{1 + \beta(C_N^k - C_{N-1}^k)} \tag{2.16}$$

where $\beta = \dfrac{\Delta t}{\Delta x}$, C_i^k is the wave celerity at cell i at time k, and N is the number of cells.

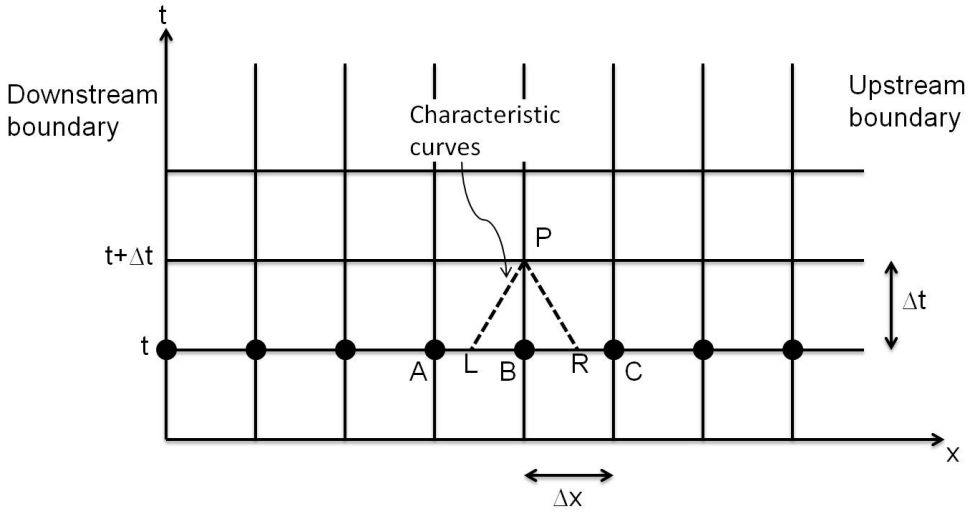


Figure 2.2: Illustration of method of specified time intervals.

2.2.1 Internal Conditions for Confluence in Channel Network

In order to apply this model to a channel network, it is necessary to impose networked internal boundary conditions at every confluence in the channel network. Here, the internal boundary conditions constraints are briefly described for a simple confluence as in Figure 2.3 which comprises three channels.

The constraints corresponding to the internal boundary conditions of stage and discharge are as follows:

$$H_1 = H_2 = H_3$$
$$Q_1 = Q_2 + Q_3$$

where H_1, H_2, and H_3 represent the stage in the cross sections 1, 2, and 3, respectively, and Q_1, Q_2, and Q_3 are the discharge at the three cross sections. The first equation is simply consistency of stage in all channels at the junction and the second equation is just the conservation of mass at the junction.

The discretized equations obtained from the Lax scheme and the method of specified time intervals along with the internal boundary constraints assembled to obtain a state-space model for the entire network of interest, written in a compact form as follows

$$x_{k+1} = f(x_k, u_k) \tag{2.17}$$

where x_k is the state vector at time k which consists of discharge and stage at all cells throughout the whole network excluding the external boundary condition variables, and the input vector u_k contains the external boundary conditions.

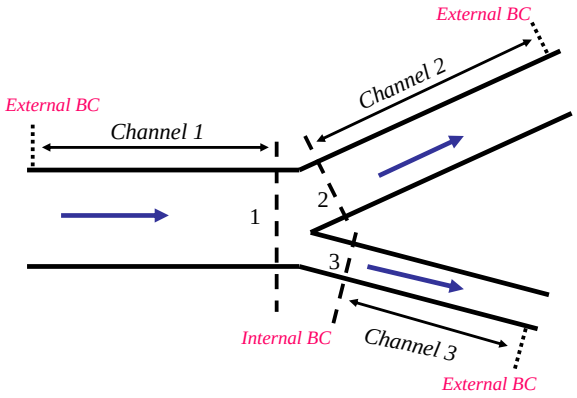


Figure 2.3: Illustration of internal and external boundary conditions for a channel network. Image: courtesy of Qingfang Wu [119].

2.2.2 Stochastic State-space Model

The effect of modelling uncertainties, as well as inaccuracies in measurements of the inputs, are commonly considered as an additive noise term in the state equations (2.17) to obtain a stochastic equation

$$x_{k+1} = f(x_k, u_k) + v_k \tag{2.18}$$

The noise v_k is usually assumed to be zero-mean white Gaussian and

$$E[v_k v_l^T] = Q_k \delta_{kl} \tag{2.19}$$

$x_0 \in \mathbb{R}^m$ is the initial state which is also assumed to be Gaussian and

$$x_0 = \mathcal{N}(\bar{x}_0, P_0) \tag{2.20}$$

where $\bar{x}_0$ and P_0 are the initial guesses for state and error covariance.

Similarly, the errors and uncertainties in the measurements can be taken into account by adding a noise term to the measurement model to obtain

$$z_k = g(x_k, k) + e_k \tag{2.21}$$

where g is the function that relates the measurements to the state vector and e_k is the measurement noise of the sensors which is assumed to be zero-mean white Gaussian and

$$E[e_k e_l^T] = R_k \delta_{kl} \tag{2.22}$$

We also assume that the process and measurement noises and the initial conditions are all uncorrelated.

2.2.3 Summary

In this chapter, we showed how a stochastic state-space model describing the flow in a network of open channels can be constructed. We discretized the one-dimensional Saint-Venant equations and used the consistency constraints at the junctions to build a state-space model in which the state vector consists of the discharge and stage at all grid points in the network excluding the boundary conditions and the input vector contains the boundary conditions.

Chapter 3

State Estimation in Open Channels using the Kalman filter and its extensions

In the previous chapter, we obtained a discrete state-space model describing the dynamics of the flow in open channel networks. Simplifications and numerical approximations made to obtain such a model result in discrepancies between the model and the actual system. In many applications including automated gate control systems, emergency response and flood monitoring systems and real-time hydrologic studies, estimating the flow state in real time is needed. When measurements of the flow, other than the boundary conditions, are available, it is desired to integrate the additional measurements into the flow model to improve the model accuracy. In this chapter, we specifically focus in a case in which Lagrangian measurements of the flow are available. *Lagrangian measurements* are measurements of the flow properties at a point moving with the flow along the streamline whereas *Eulerian measurements* are measurements of the flow properties at a fixed location. Lagrangian sensors which move with the flow and report their location and possibly other local quantities of interest (temperature, salinity, etc.) are commonly used in oceanography [9, 51, 105] (usually referred to as *drifters*) and in river hydraulics [4]. Lower production and maintenance cost, as well as flexibility in deployment are the main advantages of the drifters over the traditional static sensors. These drifters are equipped with GPS receivers and report their position and other measurements at every time step. The position of the drifters at every time step can be used to approximate the velocity of the flow at the corresponding location and time step. Then the goal is to estimate the average velocity of the flow throughout the whole domain of interest, i.e. at all discretized cells, using the local velocity measurements of the flow obtained from a number of drifters.

In this chapter, we apply a few standard state estimation techniques such as the *Kalman Filter* (KF), the *Extended Kalman Filter* (EKF) and the *Unscented Kalman Filter* (UKF). Application of any of these filters require a state-space model of the system. In the previous chapter, we showed how a nonlinear state-space model can be obtained from the shallow water equations. Nonetheless, for application of the Kalman filter a linear state-space model of the system is required. In this chapter, we first linearize the Saint-Venant equations and construct a linear state-space model of the flow in an open channel. We then review the KF, EKF and UKF and apply the methods on a section of Sacramento River above Georgiana Slough in Sacramento-San Joaquin Delta in northern California and present the numerical results.

3.1 Linear State-space model of the flow

To obtain a linear state-space model of the flow, the Saint-Venant equations first need to be linearized around a steady state, often called the backwater curve. In what follows, we explain how the backwater curve can be calculated and how the equations can be linearized around the calculated steady state. The linearized equations are then discretized to obtain a linear state-space model describing the flow in an open channel.

In this chapter, we use a form of the Saint-Venant equations which the average velocity V and stage H instead of discharge and stage as its state variables. The Saint-Venant equations with V and H as state variables have the following form

$$T\frac{\partial H}{\partial t} + \frac{\partial (THV)}{\partial x} = 0 \tag{3.1}$$

$$\frac{\partial V}{\partial t} + V\frac{\partial V}{\partial x} + g\frac{\partial H}{\partial x} + g(S_f - S_b) = 0 \tag{3.2}$$

with the Manning-Strickler's formula

$$S_f = \frac{m^2 V|V|P^{4/3}}{(TH)^{4/3}} \tag{3.3}$$

3.1.1 Steady flow: Backwater curve

Backwater curve is the longitudinal profile of the surface of the water in a non-uniform flow in an open channel when the water surface is naturally or artificially raised above its normal level. Denoting the variables corresponding to the steady state by adding a bar, $\bar{\cdot}$,

the steady state equations can be written as:

$$\frac{d\bar{V}(x)}{dx} = -\frac{\bar{V}(x)}{\bar{H}(x)}\frac{d\bar{H}(x)}{dx} - \frac{\bar{V}(x)}{T(x)}\frac{dT(x)}{dx} \tag{3.4}$$

$$\frac{d\bar{H}(x)}{dx} = \frac{S_b - \bar{S}_f}{1 - \bar{F}(x)^2} \tag{3.5}$$

with $\bar{C} = \sqrt{g\bar{H}}$ the wave celerity, $\bar{F} = \bar{V}/\bar{C}$ the Froude number. Throughout this dissertation, we assume the flow to be *sub-critical*, i.e., $\bar{F} < 1$.

Remark 1 *In the case of uniform flow, the steady velocity, $\bar{V}(x) = \bar{V}$, and the normal depth, $\bar{H}(x) = H_n$, can be calculated by solving the normal depth equation, $\bar{S}_f = S_b$.*

3.1.2 Linearized Saint-Venant Model

The Saint-Venant equations are nonlinear in the flow variables V and H. It is a common practice to linearize the equations when a linear model of the system is desired [5], [6]. Each term $f(V, H)$ in the Saint-Venant model can be expanded in a Taylor series around the steady state flow variables $\bar{V}(x)$ and $\bar{H}(x)$. Considering only the first order perturbations, $f(V, H) \approx f(\bar{V}, \bar{H}) + (f_V)\,|_{(\bar{V},\bar{H})} v(x,t) + (f_H)\,|_{(\bar{V},\bar{H})} h(x,t)$. The first order perturbations in velocity (resp. stage) is given by $v(x,t) = V(x,t) - \bar{V}(x)$ (resp. $h(x,t) = H(x,t) - \bar{H}(x)$).

After substituting the expressions of H and V with $\bar{H} + h$ and $\bar{V} + v$ in equations (3.1) and (3.2) and some manipulation of terms, the linearized Saint-Venant model for the perturbed flow variables v and h can be written in the following form

$$h_t + \bar{H}(x)v_x + \bar{V}(x)h_x + \alpha(x)v + \beta(x)h = 0 \tag{3.6}$$

$$v_t + \bar{V}(x)v_x + gh_x + \gamma(x)v + \eta(x)h = 0 \tag{3.7}$$

with $\alpha(x)$, $\beta(x)$, $\gamma(x)$ and $\eta(x)$ given by

$$\alpha(x) = \frac{d\bar{H}}{dx} + \frac{\bar{H}}{T}\frac{d\bar{T}}{dx} \tag{3.8}$$

$$\beta(x) = -\frac{\bar{V}}{\bar{H}}\frac{d\bar{H}}{dx} - \frac{\bar{V}(x)}{T(x)}\frac{dT(x)}{dx} \tag{3.9}$$

$$\gamma(x) = 2gm^2\frac{|\bar{V}|}{\bar{H}^{\frac{4}{3}}} - \frac{\bar{V}}{\bar{H}}\frac{d\bar{H}}{dx} - \frac{\bar{V}(x)}{T(x)}\frac{dT(x)}{dx} \tag{3.10}$$

$$\eta(x) = -\frac{4}{3}gm^2\frac{\bar{V}|\bar{V}|}{\bar{H}^{\frac{7}{3}}} \tag{3.11}$$

3.1.3 Discretization: Lax Diffusive Scheme

We use the Lax diffusive scheme [37], [108] which is a first-order explicit scheme to discretize the equations. Using f to represent the dependent variables, v and h, the derivatives become

$$\frac{\partial f}{\partial t} = \frac{f_i^{k+1} - \frac{1}{2}(f_{i+1}^k + f_{i-1}^k)}{\Delta t} \tag{3.12}$$

$$\frac{\partial f}{\partial x} = \frac{(f_{i+1}^k - f_{i-1}^k)}{2\Delta x} \tag{3.13}$$

Applying this scheme to equations (3.6) and (3.7), we get:

$$\begin{aligned} h_i^{k+1} = &\frac{1}{2}(h_{i+1}^k + h_{i-1}^k) \\ &- \frac{\Delta t}{4\Delta x}(\bar{H}_{i+1} + \bar{H}_{i-1})(v_{i+1}^k - v_{i-1}^k) \\ &- \frac{\Delta t}{4\Delta x}(\bar{V}_{i+1} + \bar{V}_{i-1})(h_{i+1}^k - h_{i-1}^k) \\ &- \frac{\Delta t}{2}(\alpha_{i+1}v_{i+1}^k + \alpha_{i-1}v_{i-1}^k) \\ &- \frac{\Delta t}{2}(\beta_{i+1}h_{i+1}^k + \beta_{i-1}h_{i-1}^k) \end{aligned} \tag{3.14}$$

$$\begin{aligned} v_i^{k+1} = &\frac{1}{2}(v_{i+1}^k + v_{i-1}^k) \\ &- \frac{\Delta t}{4\Delta x}(\bar{V}_{i+1} + \bar{V}_{i-1})(v_{i+1}^k - v_{i-1}^k) \\ &- \frac{g\Delta t}{2\Delta x}(h_{i+1}^k - h_{i-1}^k) \\ &- \frac{\Delta t}{2}(\gamma_{i+1}v_{i+1}^k + \gamma_{i-1}v_{i-1}^k) \\ &- \frac{\Delta t}{2}(\eta_{i+1}h_{i+1}^k + \eta_{i-1}h_{i-1}^k) \end{aligned} \tag{3.15}$$

This scheme is stable provided that the Courant-Friedrich-Lewy (CFL) condition holds, i.e.

$$\frac{\Delta t}{\Delta x}|V + C| \leq 1 \tag{3.16}$$

3.1.4 Discrete State-Space Model

Using the discretization of the constitutive equations, we can form a linear state-space model as follows

$$x_{k+1} = Ax_k + Bu_k \tag{3.17}$$

where

$$x_k = (v_2^k, \cdots, v_I^k, h_1^k, \cdots, h_{I-1}^k)^T \tag{3.18}$$

and u_k is the boundary conditions, i.e. the upstream velocity perturbation and downstream stage perturbation,

$$u_k = (v_1^k, h_I^k)^T \tag{3.19}$$

v_i^k and h_i^k are velocity and stage perturbations at cell i at time $k\Delta t$, respectively, and I is the number of cells used for the discretization of the channel.

3.2 State Estimation Framework

3.2.1 Process Model

Modeling the uncertainties by adding a noise term w_k to the state-space equation (3.17) leads to

$$x_{k+1} = Ax_k + Bu_k + w_k \tag{3.20}$$

The process noise is assumed to be white Gaussian noise and

$$E[w_k w_l^T] = Q_k \delta_{kl} \tag{3.21}$$

$z_0 \in \mathbb{R}^m$ is the initial state and it is assumed that

$$x_0 = \mathcal{N}(\bar{x}_0, P_0) \tag{3.22}$$

3.2.2 Measurement Model

The information of the position of the drifters equipped with GPS can be used to obtain Lagrangian measurements of the flow velocity. Each drifter reports its current position at every time step k which is used to calculate the speed of the drifter at every time step. Since our estimation method is based on a one-dimensional model of the flow, we have the drifter released at the center line of the channel and we assume it stays on the center line as it moves along the channel. This is a realistic assumption as long as the drifter is moving on the same channel since the lateral components of the flow velocity are usually negligible.

Denoting the collection of average velocities obtained from the drifters at time step k by z_k, the measurement model can be written as

$$z_k = H_k x_k + e_k \tag{3.23}$$

where $e(k)$ is the measurement noise on the sensors which is assumed to be white Gaussian noise and

$$E[e_k e_l^T] = R_k \delta_{kl} \tag{3.24}$$

We also assume that the process and measurement noises and the initial conditions are all independent.

Note that the observation operator H_k is time-varying since the drifters are moving with the flow and therefore the cells at which the flow velocity is measured are changing over time.

Defining the mean and the covariance of the estimations with the following notations

$$\hat{x}_k = E[x_k|z_0, \cdots, z_k] \tag{3.25}$$
$$\hat{x}_k^- = E[x_k|z_0, \cdots, z_{k-1}] \tag{3.26}$$
$$P_k^- = \Sigma_{k|k-1} \tag{3.27}$$
$$P_k = \Sigma_{k|k} \tag{3.28}$$

the iterations of the Kalman filter can be written as follows [18, 79]
Time update:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k \tag{3.29}$$
$$P_k^- = AP_{k-1}A^T + Q \tag{3.30}$$

Measurement update:

$$K_k = P_k^- C_k^T (H_k P_k^- H_k^T + R)^{-1} \tag{3.31}$$
$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H_k\hat{x}_k^-) \tag{3.32}$$
$$P_k = (I - K_k H_k)P_k^- \tag{3.33}$$

3.2.3 Extended Kalman Filter

In the *Extended Kalman Filter* (EKF), the states of the system are approximated by a Gaussian random variable and are propagated through a linearized approximation of the state equations. The *prior* mean of the state is fed into the state equations to yield the prediction of the state. The *posterior* covariance matrices are calculated for a linear model which is obtained from linearizing the state equations around the current estimate [18].

With the stochastic state-space model given in the previous section, the iterations of the EKF can be summarized as follows

Time update:

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_{k-1}, 0) \tag{3.34}$$
$$P_{k|k-1} = \Phi_{k-1}P_{k-1|k-1}\Phi_{k-1}^T + B_{k-1}Q_{k-1}B_{k-1}^T \tag{3.35}$$

Measurement update:

$$K_k = P_{k|k-1}G_k^T(G_kP_{k|k-1}G_k^T + D_kR_kD_k^T)^{-1} \tag{3.36}$$
$$\hat{y}_k = G_k\hat{x}_{k|k-1} \tag{3.37}$$
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - \hat{y}_k) \tag{3.38}$$
$$P_{k|k} = (I - K_kG_k)P_{k|k-1} \tag{3.39}$$

where

$$\Phi_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k|k-1},u_{k-1}}, B_{k-1} = \left.\frac{\partial f}{\partial w}\right|_{\hat{x}_{k|k-1},u_{k-1}} \tag{3.40}$$

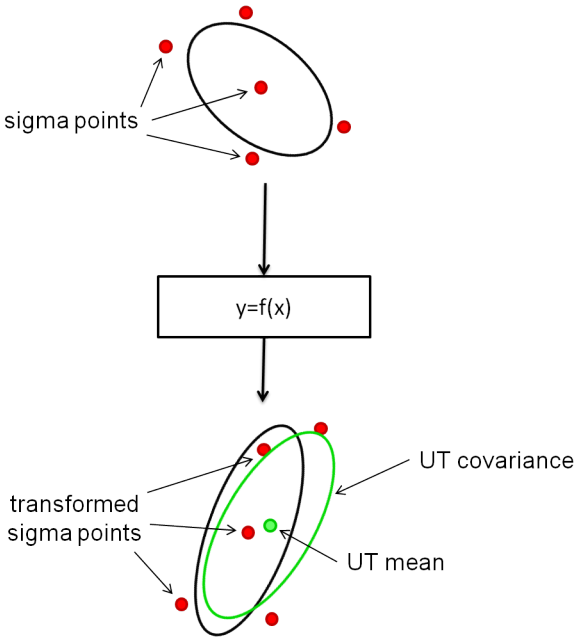


Figure 3.1: Schematic of the unscented transformation.

3.2.4 Unscented Kalman Filter

In the *Unscented Kalman Filter* (UKF), the statistical properties of the prior distribution are used to choose a set of weighted *sigma points* deterministically. After propagating each sigma point through the nonlinear model, the statistical properties of the posterior distribution are calculated [66], [116], [61].

Defining the *augmented state* as $x_k^a = [x_k^T, w_k^T, e_k^T]^T$, for the stochastic model given in the previous section, the UKF algorithm can be summarized as follows

Initialize with:

$$\hat{x}_{0|0} = E[x_0] = \bar{x}_0 \tag{3.41}$$

$$P_{0|0} = E[(x_0 - \bar{x}_0)(x_0 - \bar{x}_0)^T] = P_0 \tag{3.42}$$

$$\hat{x}_0^a = E[x^a] = [\hat{x}_0^T \, \mathbf{0} \, \mathbf{0}]^T \tag{3.43}$$

$$P_0^a = E[(x_0^a - \bar{x}_0^a)(x_0^a - \bar{x}_0^a)^T] = \begin{bmatrix} P_0 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & Q_0 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & R_0 \end{bmatrix} \tag{3.44}$$

Calculate the weights of sigma points:

$$W_i^{(m)} = \begin{cases} \dfrac{\lambda}{n^a + \lambda} & \text{if } i = 1 \\ \dfrac{1}{2(n^a + \lambda)} & \text{otherwise} \end{cases} \tag{3.45}$$

$$W_i^{(c)} = \begin{cases} \dfrac{\lambda}{n^a + \lambda} + (1 - \alpha^2 + \beta) & \text{if } i = 1 \\ \dfrac{1}{2(n^a + \lambda)} & \text{otherwise} \end{cases} \tag{3.46}$$

where n^a is the dimension of the augmented state, λ is defined as $\lambda = \alpha^2(n^a + \kappa) - n^a$ where α, β and κ are scaling parameters. α determines the spread of the sigma points and is usually taken as a small positive number. The parameter κ is usually taken as 0 and β is chosen based on the prior distribution which is set to 2 for the case of Gaussian distribution.

In each iteration,

Calculate sigma points:

$$\begin{aligned} \mathcal{X}_{k-1|k-1}^a = [\, & \hat{x}_{k-1|k-1}^a, \quad \hat{x}_{k-1|k-1}^a + \sqrt{n^a + \lambda}\tilde{P}_{k-1|k-1}^a, \\ & \hat{x}_{k-1|k-1}^a - \sqrt{n^a + \lambda}\tilde{P}_{k-1|k-1}^a \,] \end{aligned} \tag{3.47}$$

where $\tilde{P}_{k-1}^a$ is the Cholesky factor of P_{k-1}^a.

Time update:

$$\mathcal{X}_{k|k-1}^x = f(\mathcal{X}_{k-1|k-1}^x, u_{k-1}, 0) \tag{3.48}$$

$$\hat{x}_{k|k-1} = \sum_{i=1}^{2n^a+1} W_i^{(m)} \mathcal{X}_{i,k|k-1}^x \tag{3.49}$$

$$\begin{aligned} P_{k|k-1} = & \sum_{i=1}^{2n^a+1} W_i^{(c)} (\mathcal{X}_{i,k|k-1}^x - \hat{x}_{k|k-1}) \\ & \times (\mathcal{X}_{i,k|k-1}^x - \hat{x}_{k|k-1})^T \end{aligned} \tag{3.50}$$

$$\mathcal{Y}_{k|k-1} = G_k \mathcal{X}^x_{k-1|k-1} \tag{3.51}$$

$$\hat{y}_{k|k-1} = \sum_{i=1}^{2n^a+1} W_i^{(m)} \mathcal{Y}_{i,k|k-1} \tag{3.52}$$

where $\mathcal{X}^x_{k-1|k-1}$ is the first n rows of $\mathcal{X}_{k-1|k-1}$ and $\mathcal{X}^x_{i,k-1|k-1}$ denotes the i^{th} column of $\mathcal{X}^x_{k-1|k-1}$.

Measurement update:

$$P_{y_k} = \sum_{i=1}^{2n^a+1} W_i^{(c)} (\mathcal{Y}_{i,k|k-1} - \hat{y}_{k|k-1}) \times (\mathcal{Y}_{i,k|k-1} - \hat{y}_{k|k-1})^T \tag{3.53}$$

$$P_{x_k y_k} = \sum_{i=1}^{2n^a+1} W_i^{(c)} (\mathcal{X}^x_{i,k|k-1} - \hat{x}_{k|k-1}) \times (\mathcal{Y}_{i,k|k-1} - \hat{y}_{k|k-1})^T \tag{3.54}$$

$$\mathcal{K}_k = P_{x_k y_k} P_{y_k}^{-1} \tag{3.55}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + \mathcal{K}_k (y_k - \hat{y}_k) \tag{3.56}$$

$$P_{k|k} = P_{k|k-1} - \mathcal{K}_k P_{y_k} \mathcal{K}_k^T \tag{3.57}$$

3.3 Implementation

3.3.1 Experiment Set-up

The method is implemented on a part of the Sacramento River, upstream of the intersection with the Georgiana slough. The Sacramento River is a part of the Sacramento-San Joaquin Delta in California which is an integral part of California's water system. The bathymetry of the channel, shown in Figure 3.2, is provided by the *United States Geological Survey* (USGS). The section of interest is of 900m length and is 85m wide in the narrowest part and 190m wide in the widest part. For discretization, we divide the channel to 30 cells with each cell being 30m long. This results in a state-space model with 58 states as described in section 3.1.4.

The so-called *forward simulation* is performed in a commercial hydrodynamic software TELEMAC 2D [103] to generate the *true state* as well as the drifter position data. TELEMAC uses a streamline upwind Petrov-Gelerkin based finite element solver for hydrodynamic equations. The mesh used for the simulation has 1939 nodes and 3525 triangular

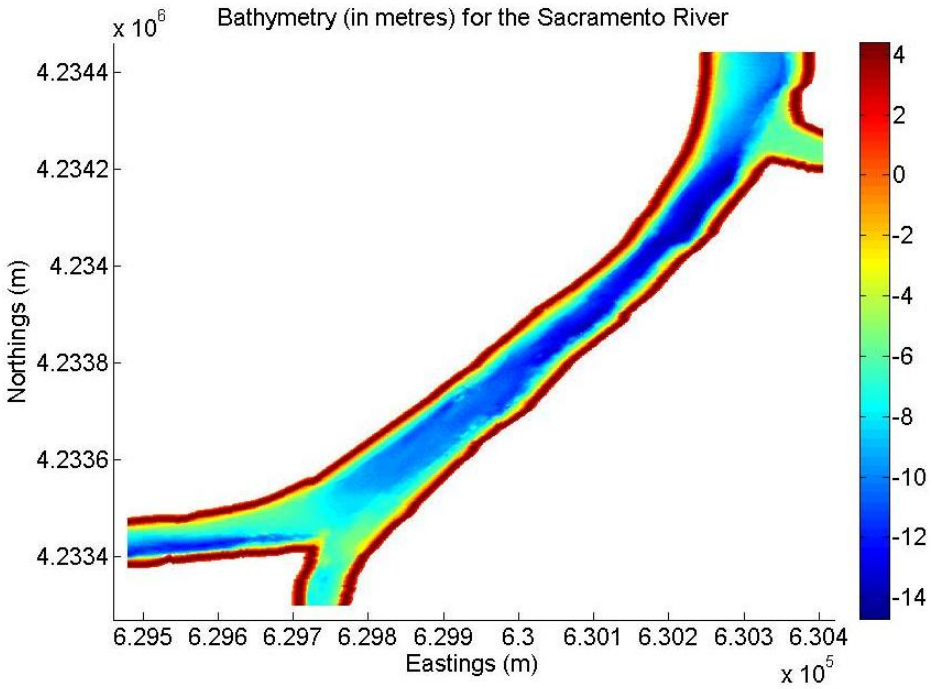


Figure 3.2: Bathymetry in Sacramento River. The bathymetry on this section of the Sacramento River varies from -14m in the deepest part to +2 on the river banks. Image: courtesy of Qingfang Wu [119].

elements. The boundary conditions, shown in Figure 3.4, are computed using the *Delta Simulation Model II* (DSM2) [29]. DSM2 is a model of the San Francisco Bay and Sacramento Delta that provides discharge and surface elevation at various locations every one hour. The flow diagram of the implementation procedure is shown in Figure 3.3. Note that the true state generated by the 2D simulation in TELEMAC must be converted to 1D data to be compared with the estimation results which are based on a 1D model of the flow.

The experiment starts at 3:40PM on March 16th 2007. The simulation runs for two and a half hours before the experiment so that the effects of the initial conditions are washed away and the model is stabilized. The time frame of the experiment is chosen such that the flow variations are as noticeable as possible. Also, as it can be seen in Figure 3.4, there is an abrupt change in the boundary conditions at 4PM which is a result of linear interpolation of the DSM2 data. This enables us to evaluate the performance of the Kalman filter in estimating the flow states in case of sudden changes in the condition of flow.

At 3:40PM, a single drifter is released at the upstream end of the river. The drifter moves with the current and its position is recorded at every time step. The trajectory of the drifter is illustrated in Figure 3.5. The data assimilation starts as soon as the drifter is released and it ends when the drifter reaches the downstream end of the river, at 4:18PM. At each time step, the velocity of the drifter is approximated by the difference between its current and previous position divided by the time step, Δt, which is chosen to be 3 seconds in this experiment. This gives us a measurement of the velocity of the current at the position of the drifter at every time step.

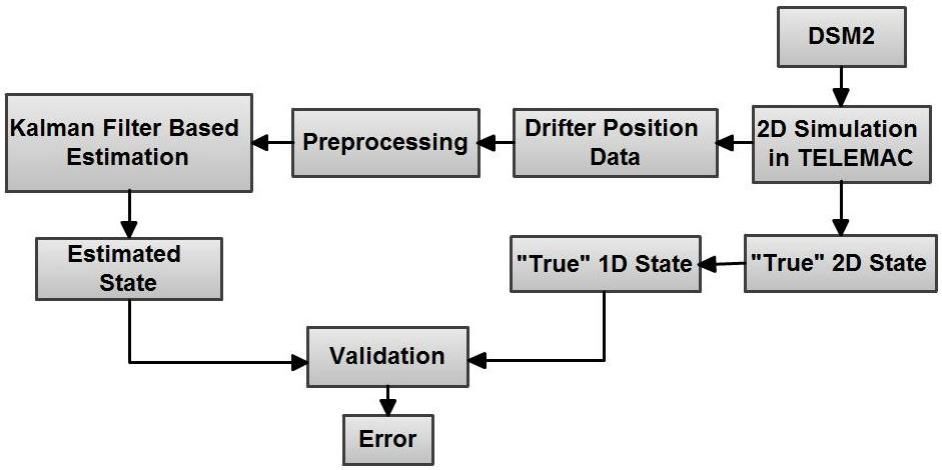


Figure 3.3: Flow diagram of the implementation procedure.

3.3.2 Numerical Results

Figure 3.7 shows the estimated velocity and the *true velocity* at four different cells. Figure 3.8 shows the time evolution of the relative error of the estimated velocity which is calculated using the following formula

$$\text{error}(k) = \sqrt{\frac{\sum_{i=1}^{N_{\text{cell}}} (u_i^k - \hat{u}_i^k)^2}{\sum_{i=1}^{N_{\text{cell}}} (u_i^k)^2}} \tag{3.58}$$

where u_i^k and $\hat{u}_i^k$ are the true and estimated values of the velocity at cell i and time step k.

As can be seen in Figure 3.8, the relative error decreases quickly and reaches below 4% at time step 50 and remains below 4% until time step 400. After time step 400, ignoring the fluctuations, the error increases relatively rapidly. Note that time step 400 corresponds to 4pm which is the time when an abrupt change in the boundary conditions occurs. In fact, the increase in the error after time step 400 is due to the fact that the deviation of the state of the system from the steady state around which the system has been linearized gets too large. Furthermore, as it can be seen in Figure 3.7, after time step 400, there are some noticeable fluctuations in the system which are essentially a result of the fluctuations in the velocity boundary conditions which can be seen in Figure 3.4. In spite of the abrupt change in the boundary conditions and also the oscillations in the forcing function of the system, the relative error stays below 15% until the end of the experiment. The relative error in case of using the linear model without using any measurements is also shown in Figure 3.8. This in fact proves that incorporating the measurements obtained from the drifter into the model significantly improves the estimation results.

The computational cost of the method is very reasonable. In the above experiment with a state-space model of 58 states, each iteration of the Kalman filter takes less than 1 millisecond on a 2.4GHz Pentium dual core processor.

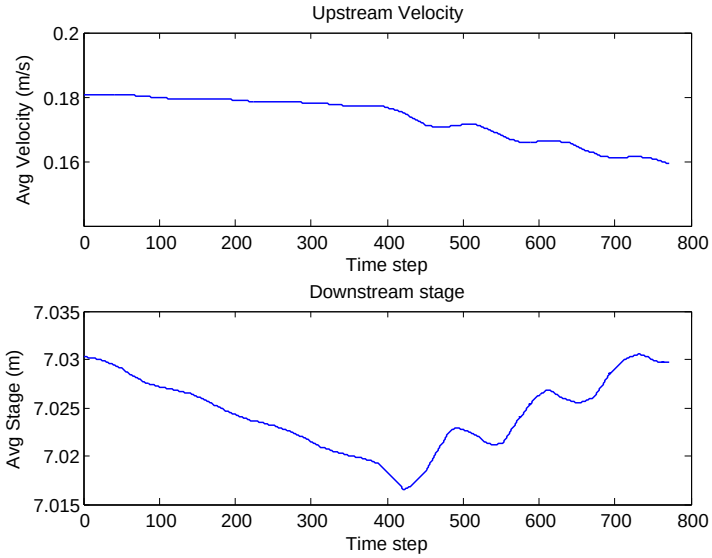


Figure 3.4: The boundary conditions.

To implement the EKF and UKF, we perform a longer experiment. The experiment starts at 3:00PM and ends at 5:00PM on March 16th 2007. At 3:00PM, a single drifter is released at the upstream end of the domain. The drifter moves with the current and its position is recorded at every time step. The trajectory of the drifter is illustrated in Figure 3.5. When the drifter reaches the downstream end of the domain, another drifter is released at the upstream ends. This procedure is repeated until the end of the experiment so that there is one drifter in the channel at all times during the experiment.

Figure 3.7 shows the estimated velocity and the *true velocity* at two different cells corresponding to both the EKF and the UKF algorithms. Figure 3.8 shows the time evolution of the relative error of the estimated velocity for both cases of EKF and UKF.

As can be seen in Figure 3.8, after time step 200, the error reduces to below 10% and it stays around 10% most of the time until the end of the experiment. It is also interesting to note the relative error corresponding to both EKF and UKF are very similar. Figure 3.8 also shows the relative error corresponding to the one dimensional Saint-Venant model without using Lagrangian measurements. It can be seen that utilizing the Lagrangian measurements via both filters reduces the relative error by about 6% most of the time during the experiment. Also, comparing the results with the case of using the Kalman filter based on a linearized model of the flow, the performance of the Kalman filter is slightly better initially for about 800 time steps. But, both nonlinear filters outperform the Kalman filter significantly after this period of time. It can be seen that the relative error in case of the KF starts to increase rapidly after time step 1200. However, in case of the nonlinear filters, the error stays around 10% until the end of the 2-hour experiment. This can be explained according the fact that as the system's state deviation from the steady state around which the model has been linearized increases, the linear model diverges from the underlying dynamics of the system.

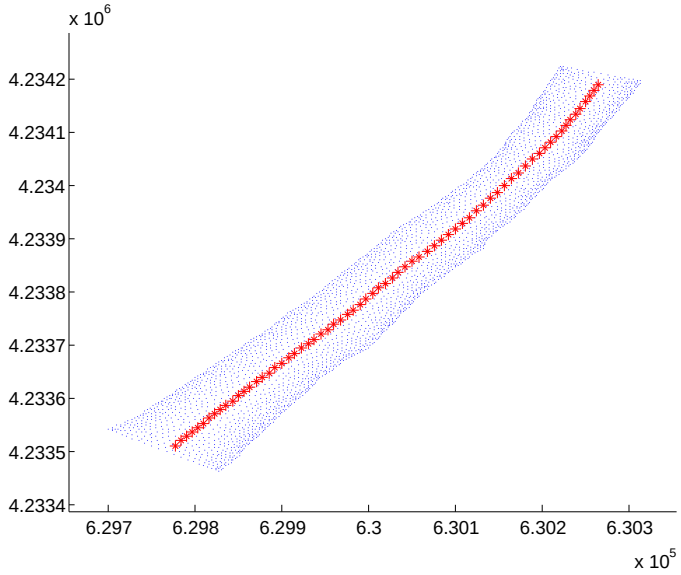


Figure 3.5: Trajectory of the drifter.

3.3.3 Implementation with experiment data in Sacramento River

In this section, we present an experiment performed with drifters in Sacramento River at the junction of Georgiana Slough, between points A, B and C as shown in Figure 3.9. The experiment was done on Sep 10, 2011 at 9:30 AM for a period of three hours. 12 drifters were released at location A in the map of Figure 3.9 and as they reached points C or E, they were retrieved by the team members and were released at the upstream point A again. The lengths of sections A-D, D-B and D-C are 2,928 ft., 2,658 ft. and 1,446 ft., respectively. The upstream stage at point A, the stage at point B and the discharge at point C are used as the boundary conditions and they are obtained from USGS sensors. Figure 3.10 shows the boundary conditions for the duration of the experiment.

The information of the position of the drifters equipped with GPS can be used to obtain Lagrangian measurements of the flow velocity. Each drifter reports its current position at every time step which is used to calculate the speed of the drifter at every time step. In order to derive the relation between the drifter velocity and the flow at the corresponding cross-section, we assume a quartic velocity profile on the surface and a logarithmic profile along the depth [25]. For a given particle moving at a distance y from the center line and z from the surface, the particle's velocity $v_p(y,z)$ is related to the flow Q with the following equations:

$$v_p(y,z) = F_T(y)F_V(z)\frac{Q}{A} \tag{3.59}$$

with

$$F_T(y) = A_q + B_q\left(\frac{2y}{w}\right)^2 + C_q\left(\frac{2y}{w}\right)^4 \tag{3.60}$$

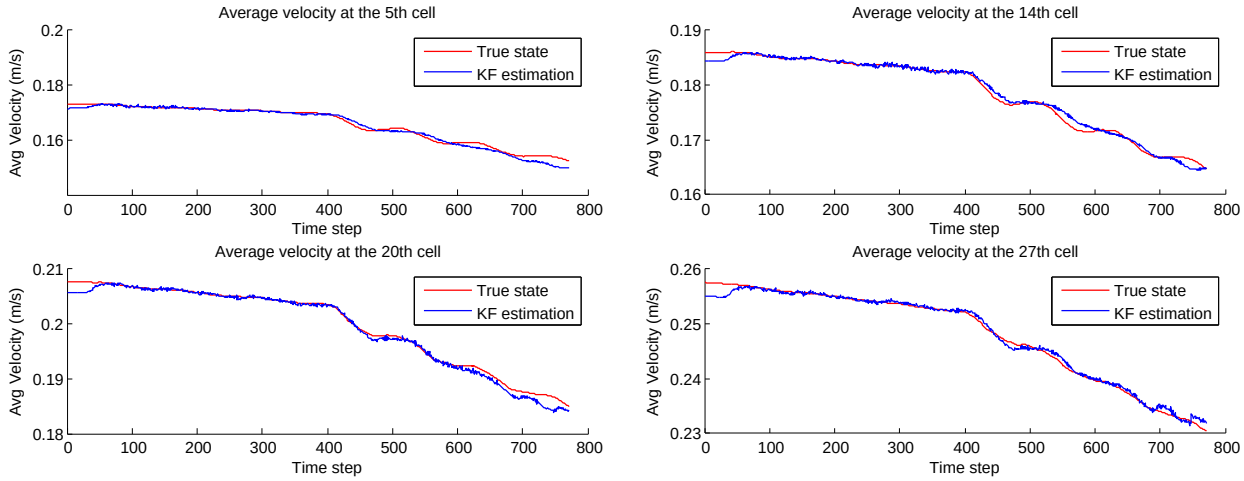


Figure 3.6: The time evolution of the estimated velocity and the true velocity at four cells.

$$A_q + B_q + C_q = 0 \tag{3.61}$$

$$A_q + \frac{B_q}{3} + \frac{C_q}{5} = 1 \tag{3.62}$$

$$F_V(z) = 1 + \left(\frac{0.1}{\kappa}\right)\left(1 + log\left(\frac{z}{d}\right)\right) \tag{3.63}$$

where w is the channel width, d is the water depth, and A_q, B_q and C_q are constants and $\kappa = 0.4$. A_q is commonly calculated experimentally and equations (4.6) and (4.7) are used to compute B_q and C_q.

Denoting the collection of velocity measurements obtained from the drifters at time step k by y_k, the measurement model can be written as

$$y_k = g(x_k, k) \tag{3.64}$$

Note that the observation operator g is time-varying since the drifters are moving with the flow. Therefore, the cells at which the flow velocity is measured are changing over time.

Before using the drifter data, a data pre-processing is done. In particular, based on the drifter velocity, it is determined if the drifter is drifted towards the bank or is being carried in a boat in which case the data is thrown away.

With the above measurement model, we apply the extended Kalman filter to perform state estimation. Figure 3.11, shows the downstream discharge at point B computed by the forward simulation, the EKF and the true value obtained from USGS sensor. As can be seen in this figure, assimilation of drifter data into the model improves the estimated discharge at this location.

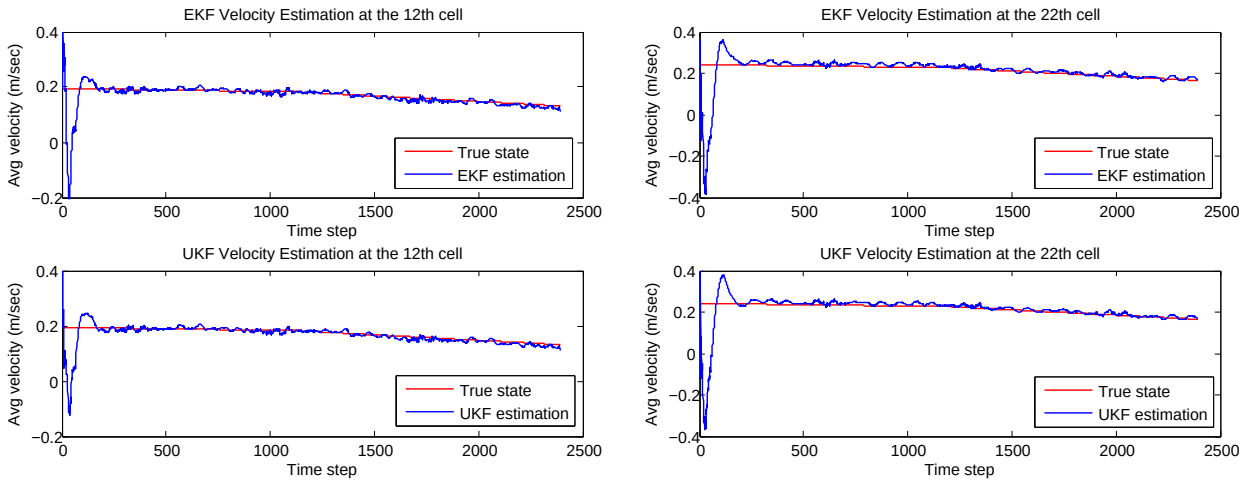


Figure 3.7: The time evolution of the estimated velocity and the true velocity corresponding to EKF algorithm (top) and UKF algorithm (bottom) at the 12^{th} and the 22^{nd} cells.

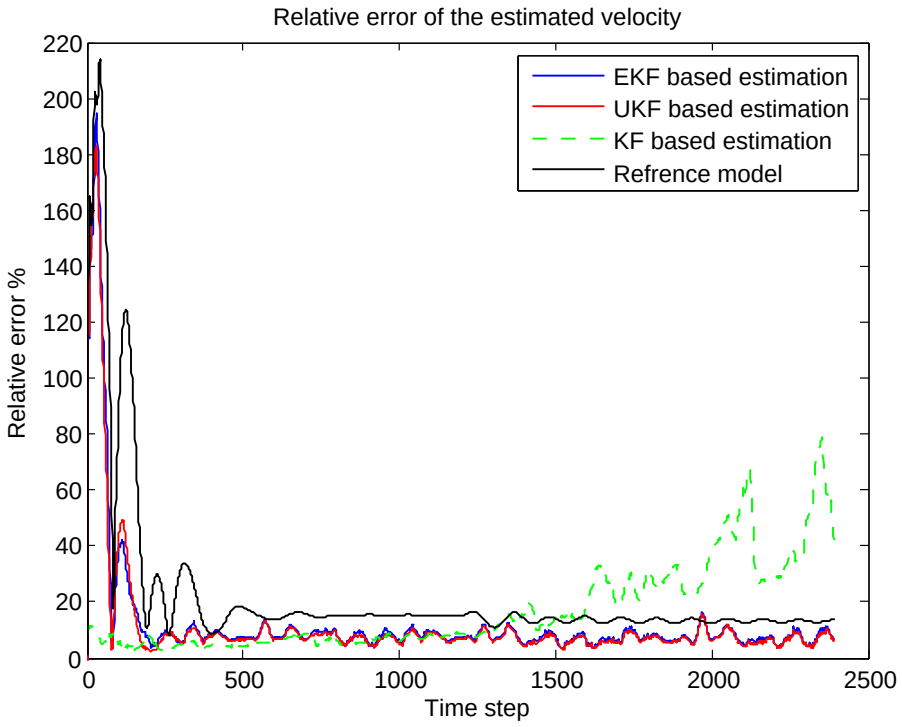


Figure 3.8: Relative error of the estimated velocity in four cases of EKF, UKF and KF algorithms and 1D Saint-Venant model.

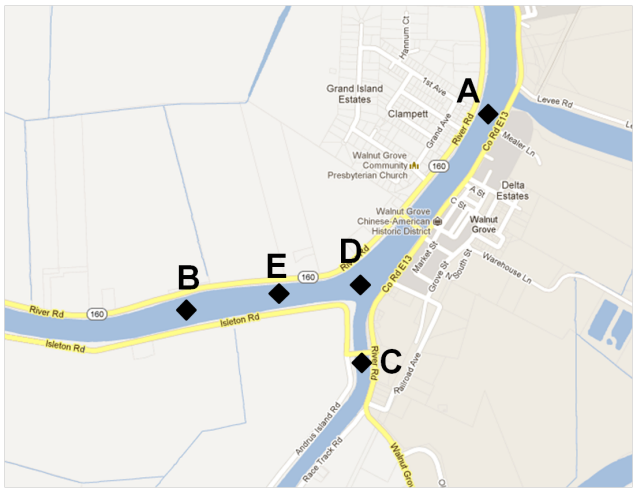


Figure 3.9: The map of area of the experiment at the junction of Sacramento River and Georgiana Slough near the city of Walnut Grove, CA.

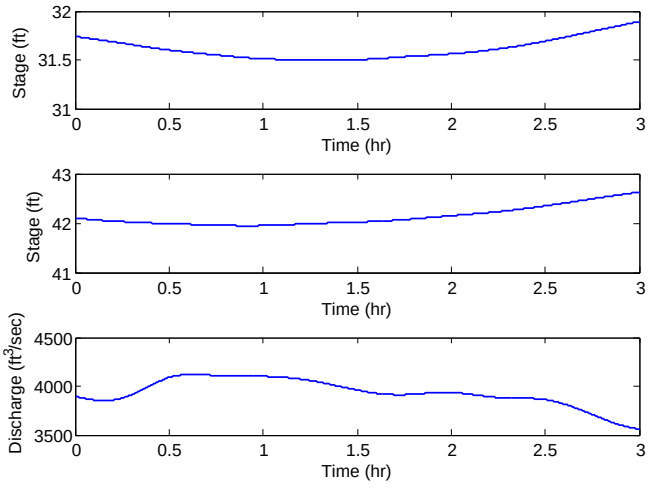


Figure 3.10: The boundary conditions, upstream stage and downstream discharges for the duration of the experiment.

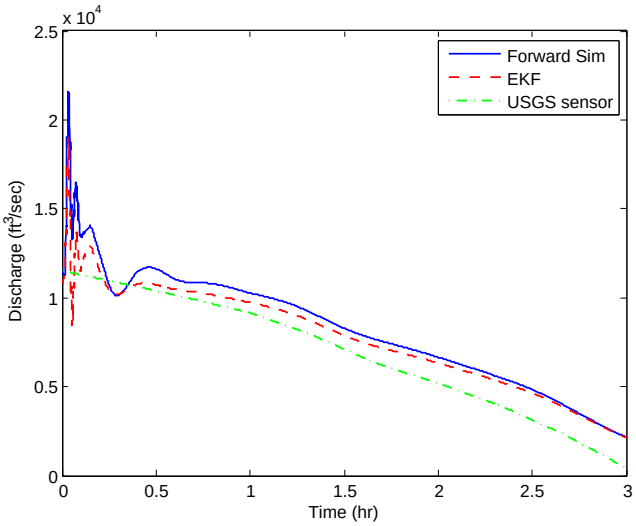


Figure 3.11: The downstream discharge at point B computed by the forward simulation, the EKF and the true value obtained from USGS sensor.

Chapter 4

Combined state-parameter Estimation using the extended Kalman filter

Following the previous chapter in which we presented how standard filtering-based estimation methods can be used to perform real-time state estimation of flow in open channels, in the current chapter, we consider a case in which real-time estimation of flow in an open channel is desired while some of the model parameters are unknown. In practice, it is sometimes not possible to obtain an accurate approximation of the parameters of the model because of lack of proper equipment, time constraints, costs, etc. As a matter of fact, one of the motivations of using drifters to obtain measurements as opposed to traditional static sensors is their applicability in new areas where no infrastructure is available. For instance, in case of an emergency (e.g. a levee break, gate malfunction), measurements of the flow can be obtained by releasing a group of drifters in the region of interest and use these measurements to model the flow in real time. For such applications, there will not be enough time to design an experiment to identify the model parameters, or even if all the parameters have been identified before, in case of an emergency, the flow conditions (e.g. the channel geometry) may change significantly such that the former values of parameters are no longer accurate enough. One possible approach is to assume a rough approximation of the parameter and perform the data assimilation method. However, depending on the sensitivity of the model to the unknown parameters, the error introduced by these approximations may be large. In this chapter, we propose a method to estimate the unknown model parameters along with the state in real time by augmenting the unknown parameters to the state vector and applying the extended Kalman filter to perform state estimation on the augmented state-space model. It is clear considering parameters as unknown as opposed to having fixed values adds to the degrees of freedom of the model and hence may improve the estimation results.

We evaluate the performance of the method using data collected from an experiment

performed at the *USDA-ARS Hydraulic Engineering Research Unit* (HERU) in Stillwater, Oklahoma in November 2009. Since the bottom elevation of the channel is not available, the bed slope of the channel is assumed as an unknown parameter. Compared to the case of performing state estimation assuming a zero bed slope, it is shown that considering the bed slope as an unknown parameter and using the measurements to estimate it improves the model prediction.

4.1 State-Space Model

The discretized equations obtained in section 2.2 can be used to obtain a state-space model

$$x_{k+1} = f(x_k, u_k) \tag{4.1}$$

where x_k is the state vector at time k

$$x_k = (Q_2^k, \cdots, Q_N^k, H_1^k, \cdots, H_{N-1}^k)^T \tag{4.2}$$

and the input u_k contains the boundary conditions, i.e. the upstream flow and downstream stage,

$$u_k = (Q_1^k, H_N^k)^T \tag{4.3}$$

Q_i^k and H_i^k are the flow and stage at cell i at time $k\Delta t$, respectively, and N is the number of cells used for the discretization of the channel.

Assuming that all model parameters are known, when measurements of the flow other than the boundary conditions are available, these measurements can be incorporated into the state-space model using one of the standard nonlinear filters, e.g. the extended Kalman filter. However, in practice, it is sometimes impossible or expensive to obtain accurate values for one or more of these parameters. For instance, it is usually a difficult task to obtain an accurate value for the bed slope of a channel. As will be shown in section 4.3.3, the results of the model are very sensitive to the value of the bed slope. In such cases, proper experiments can be designed to obtain measurements of the system and these measurements may be used later to *identify* the unknown parameters. Nevertheless, it is sometimes not possible to carry out this kind of experiments beforehand due to time constraints, lack of proper equipment, high costs, etc.

In order to obtain estimates of the unknown parameters in real time, we augment a vector of unknown parameters v_k to the state vector and consider $v_{k+1} = v_k$ as the time evolution of the parameters. A nonlinear filter can then be applied to the augmented state-space model to simultaneously estimate the parameters and the actual state of the system.

4.1.1 Measurement Model

The information of the position of the drifters equipped with GPS can be used to obtain Lagrangian measurements of the flow velocity. Each drifter reports its current position at

every time step which is used to calculate the speed of the drifter at every time step. In order to derive the relation between the drifter velocity and the flow at the corresponding cross-section, we assume a quartic velocity profile on the surface and a logarithmic profile along the depth [25]. For a given particle moving at a distance y from the center line and z from the surface, the particle's velocity $v_p(y, z)$ is related to the flow Q with the following equations:

$$v_p(y, z) = F_T(y)F_V(z)\frac{Q}{A} \tag{4.4}$$

with

$$F_T(y) = A_q + B_q\left(\frac{2y}{w}\right)^2 + C_q\left(\frac{2y}{w}\right)^4 \tag{4.5}$$

$$A_q + B_q + C_q = 0 \tag{4.6}$$

$$A_q + \frac{B_q}{3} + \frac{C_q}{5} = 1 \tag{4.7}$$

$$F_V(z) = 1 + \left(\frac{0.1}{\kappa}\right)\left(1 + log\left(\frac{z}{d}\right)\right) \tag{4.8}$$

where w is the channel width, d is the water depth, and A_q, B_q and C_q are constants and $\kappa = 0.4$. A_q is commonly calculated experimentally and equations (4.6) and (4.7) are used to compute B_q and C_q.

Denoting the collection of velocity measurements obtained from the drifters at time step k by y_k, the measurement model can be written as

$$y_k = g(x_k, k) \tag{4.9}$$

Note that the observation operator g is time-varying since the drifters are moving with the flow. Therefore, the cells at which the flow velocity is measured are changing over time.

4.1.2 Stochastic State-space Model

The effect of modelling uncertainties, as well as inaccuracies in measurements of the inputs, are commonly considered as an additive noise term in the state equations (4.1) to obtain a stochastic equation

$$x_{k+1} = f(x_k, u_k, w_k) \tag{4.10}$$

The noise w_k is usually assumed to be zero-mean white Gaussian and

$$E[w_k w_l^T] = Q_k\delta_{kl} \tag{4.11}$$

$x_0 \in \mathbb{R}^m$ is the initial state which is also assumed to be Gaussian and

$$x_0 = \mathcal{N}(\bar{x}_0, P_0) \tag{4.12}$$

where $\bar{x}_0$ and P_0 are the initial guesses for state and error covariance.

Similarly, the errors and uncertainties in the measurements can be taken into account by adding a noise term to the measurement model (4.13) to obtain

$$y_k = g(x_k, e_k, k) \tag{4.13}$$

where e_k is the measurement noise of the sensors which is assumed to be zero-mean white Gaussian and

$$E[e_k e_l^T] = R_k \delta_{kl} \tag{4.14}$$

We also assume that the process and measurement noises and the initial conditions are all independent.

4.2 Extended Kalman Filter

In the *Extended Kalman Filter* (EKF), the states of the system are approximated by a Gaussian random variable and are propagated through a linearized approximation of the state equations. The *prior* mean of the state is fed into the state equations to yield the prediction of the state. The *posterior* covariance matrices are calculated for a linear model which is obtained from linearizing the state equations around the current estimate [18].

With the stochastic state-space model given in the previous section and the following notations

$$\hat{x}_{k|k-1} = \mathbb{E}[x_k | y_0, y_1, \cdots, y_{k-1}] \tag{4.15}$$
$$\hat{x}_{k|k} = \mathbb{E}[x_k | y_0, y_1, \cdots, y_k] \tag{4.16}$$
$$P_{k|k-1} = \mathbb{E}[(x_k - \hat{x}_{k|k-1})(x_k - \hat{x}_{k|k-1})^T | y_0, y_1, \cdots, y_{k-1}] \tag{4.17}$$
$$P_{k|k} = \mathbb{E}[(x_k - \hat{x}_{k|k})(x_k - \hat{x}_{k|k})^T | y_0, y_1, \cdots, y_k] \tag{4.18}$$

the iterations of the EKF can be summarized as follows

Time update:

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_{k-1}, 0) \tag{4.19}$$
$$P_{k|k-1} = \Phi_{k-1} P_{k-1|k-1} \Phi_{k-1}^T + B_{k-1} Q_{k-1} B_{k-1}^T \tag{4.20}$$

Measurement update:

$$K_k = P_{k|k-1} G_k^T (G_k P_{k|k-1} G_k^T + D_k R_k D_k^T)^{-1} \tag{4.21}$$
$$\hat{y}_k = G_k \hat{x}_{k|k-1} \tag{4.22}$$
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k (y_k - \hat{y}_k) \tag{4.23}$$
$$P_{k|k} = (I - K_k G_k) P_{k|k-1} \tag{4.24}$$

where

$$\Phi_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k|k-1},u_{k-1}}, B_{k-1} = \left.\frac{\partial f}{\partial w}\right|_{\hat{x}_{k|k-1},u_{k-1}} \tag{4.25}$$

4.3 Implementation

4.3.1 Sensor Hardware

The Floating Sensor Network project at UC Berkeley (http://float.berkeley.edu) designs and builds drifters for riverine and estuarine environments. Six second-generation drifters were used in this experiment.

Figure 4.1: Overview of the drifter hull manufactured by the floating sensor network project. Left: closed. Right: open.

The hull is manufactured at UC Berkeley using low-cost, small-run manufacturing techniques. The drifter has a vertical cylinder configuration in order to present a uniform profile to surface currents while also supporting the antennas a small distance above the waterline. The hull consists of four major components, shown in Figure 4.1: a hand-cast fiberglass lower hull (A), machined aluminum parts for the watertight seal (B), a commercially available fiberglass pipe for the upper hull (C), and a vacuum-formed polycarbonate top cap (D). The lower hull is flooded so that water quality sensors mounted in the bulkhead may contact the water but also be mechanically protected. In order to keep the center of mass below the

center of buoyancy (a necessary condition for stability), 800 g of ballast must be located at the bulkhead between the upper and lower hull. The battery that powers the electronics is part of this ballast. Our standard configuration is to use a 200 g battery and a 600 g lead weight. The battery and water quality sensor are labelled (E) in Figure 4.1.

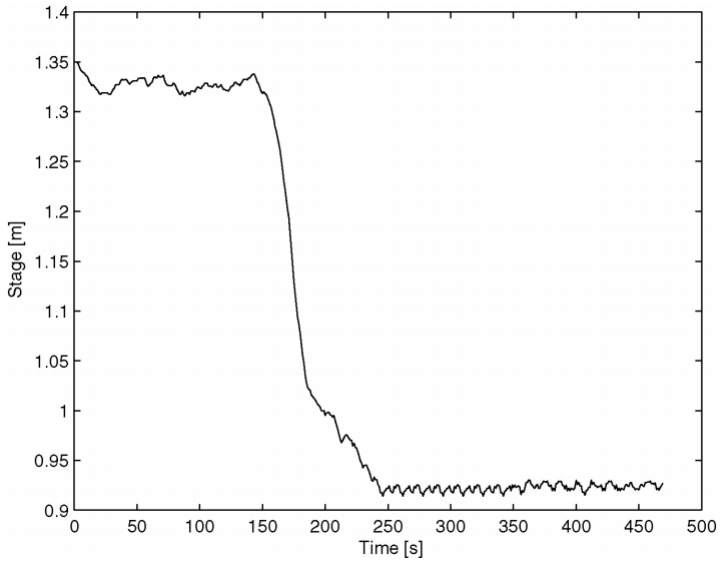


Figure 4.2: The downstream stage (m).

4.3.2 Mission Description

In November 2009, an experiment was performed at the USDA-ARS Hydraulic Engineering Research Unit (HERU) in Stillwater, Oklahoma. The HERU facility, located adjacent to Lake Carl Blackwell, has a gravity-fed supply canal which can have a controlled rate of up to 4.25 m^3/s (150 ft^3/s). The supply canal feeds a number of experimental units which are normally used for investigations into levee reliability, reservoir safety, and spillway design [28]. For our experiment, we deployed drifters into the supply canal itself. The upstream boundary condition was the supply canal flow control, set to 1.42 m^3/s (50 ft^3/s); the downstream boundary condition was a gate that could be raised or lowered to restrict the flow out of the experimental region. In this experiment, the downstream gate was opened as soon as the final drifter was released. The water stage was captured at the downstream boundary with a video camera. Figure 4.2 shows the stage at the downstream end of the channel. As can be seen in this figure, the downstream stage is initially 1.33 m and it starts to decrease as the downstream gate is opened until it becomes 0.92m.

Drifters were released at approximately 30 s intervals near the upstream boundary, at point A in Figure 4.3. After travelling through the canal for approximately 400 s, they were individually retrieved at point B. Point C marks the location of the downstream control gate.

Figure 4.4 shows the cross section of the prismatic channel over most of its extent.

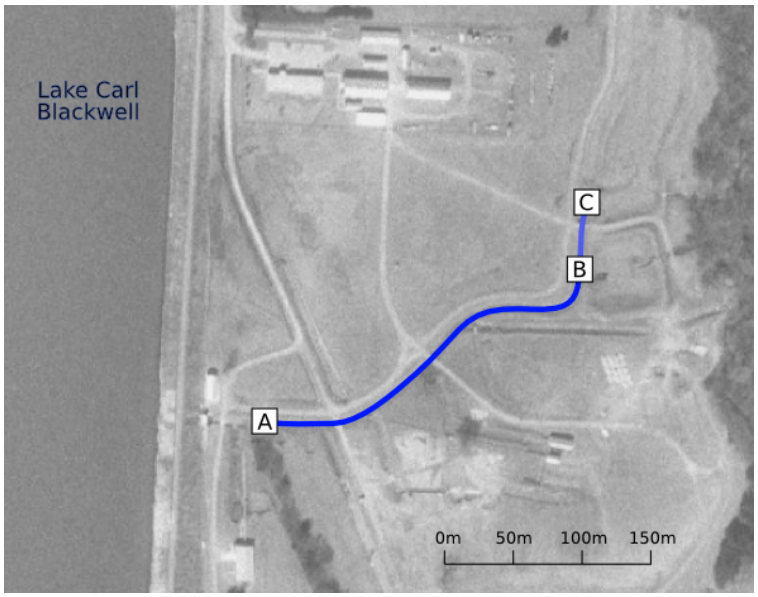


Figure 4.3: HERU facility, with experimental channel annotated. Image courtesy of USGS.

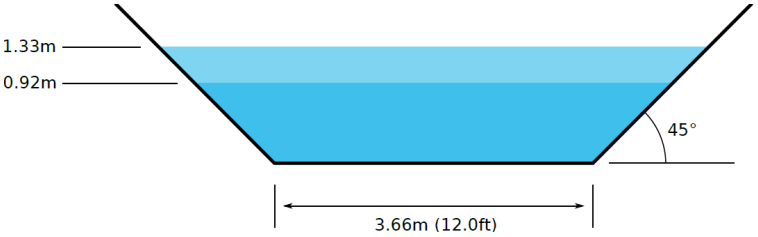


Figure 4.4: Channel profile, including minimum and maximum water height.

4.3.3 Numerical Results

The discretization is done by dividing the channel to 60 cells and the temporal step size is chosen as 1 s. Since we do not have any data about the bottom elevation of the channel, we cannot calculate the bed slope of the channel. In order to determine the sensitivity of the model with the given boundary conditions to the value of the bed slope, we run the forward simulation with three different values of bed slope. In each case, the initial condition is chosen to be the backwater curve (steady state) which is computed using the following equations:

$$\frac{\partial Q}{\partial x} = 0 \tag{4.26}$$

$$\frac{\partial H}{\partial x} = \frac{gA(S_0 - S_f)}{-Q^2 \frac{T_b+2H}{H^2(T_b+H)^2} + g(T_b H + H^2)} \tag{4.27}$$

where T_b is the bottom width.

Figure 4.5: Two drifters in the HERU facility supply canal.

Figure 4.6 shows the flow and stage at the 10^{th} cell for the three values of bed slope. It is not surprising to see that the results of forward simulation varies significantly with different values of the bed slope.

To implement the data assimilation method, we use the measurements obtained from five drifters. We then estimate the velocity of the 6^{th} drifter using the estimated flow which we compare with its actual value obtained from the 6^{th} drifter. We implement the extended Kalman filter with and without estimating the bed slope. Figure 4.7 shows the flow and stage at a few different cells predicted by the forward simulation (i.e. state-space model) assuming the bed slope is zero, estimated flow and stage by performing the data assimilation method while the bed slope is assumed to be zero, and estimated flow and stage by performing the data assimilation method and estimating the bed slope as an unknown parameter. As can be seen in Figure 4.2, the downstream stage starts to decrease at around time step 150 due to the gate opening. As can be seen in Figure 4.7, the flow increases as a result of opening the gate. It can be seen in Figure 4.7 that the stage reduction caused by opening the gate propagates backward through the channel. However, in case of assuming the bed slope as an unknown parameter, this reduction is stage is more moderate. In particular, at cell 10, no decrease in the stage is seen. This is due to the fact that for a nonzero bed slope, the backwater curve (steady state) is not uniform. Since the initial estimate of the bed slope is taken to be equal to zero, the extended Kalman filter is initialized by a uniform steady state corresponding to a zero bed slope. However, as the estimated bed slope deviates from zero, the steady state of the system deviates from uniform steady state accordingly. While the values of flow and stage estimated by the data assimilation methods seem physically more reasonable, it is not possible to formally evaluate the performance of the method by looking at these figures. In order to obtain a more quantifiable assessment of the method, we calculate the velocity of the 6^{th} drifter using the estimated flow at the corresponding cell. We use the same velocity profiles on the surface and along the depth as described in section

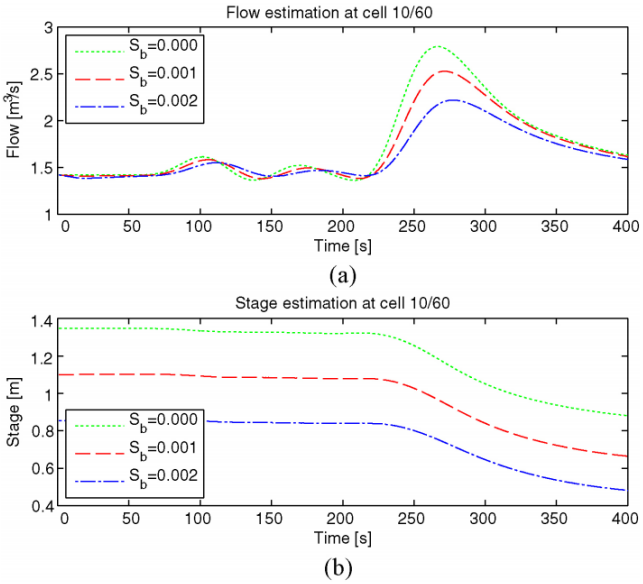


Figure 4.6: The flow (top) and stage (bottom) at the 10^{th} cell for $S_b = 0$ (green), $S_b = 0.001$ (red), $S_b = 0.002$ (blue).

4.1.1 to calculate the drifter velocity from the estimated flow. Figure 4.8 shows the velocity of the 6^{th} drifter predicted by the forward simulation and both data assimilation methods as well as its actual value. As can be seen in this figure, the data assimilation methods significantly improve the estimation results. Also, it can be seen that considering the bed slope as an unknown parameter and using the measurements to estimate it improves the estimation results further.

4.3.4 Summary

In this chapter, we considered a situation in which real-time estimates of flow state were desired while some of the model parameters were unknown. The method proposed was to augment the unknown parameters to the state vector and consider them as virtual state with identity time evolution factor and apply the extended Kalman Filter to perform the state and parameter estimation simultaneously.

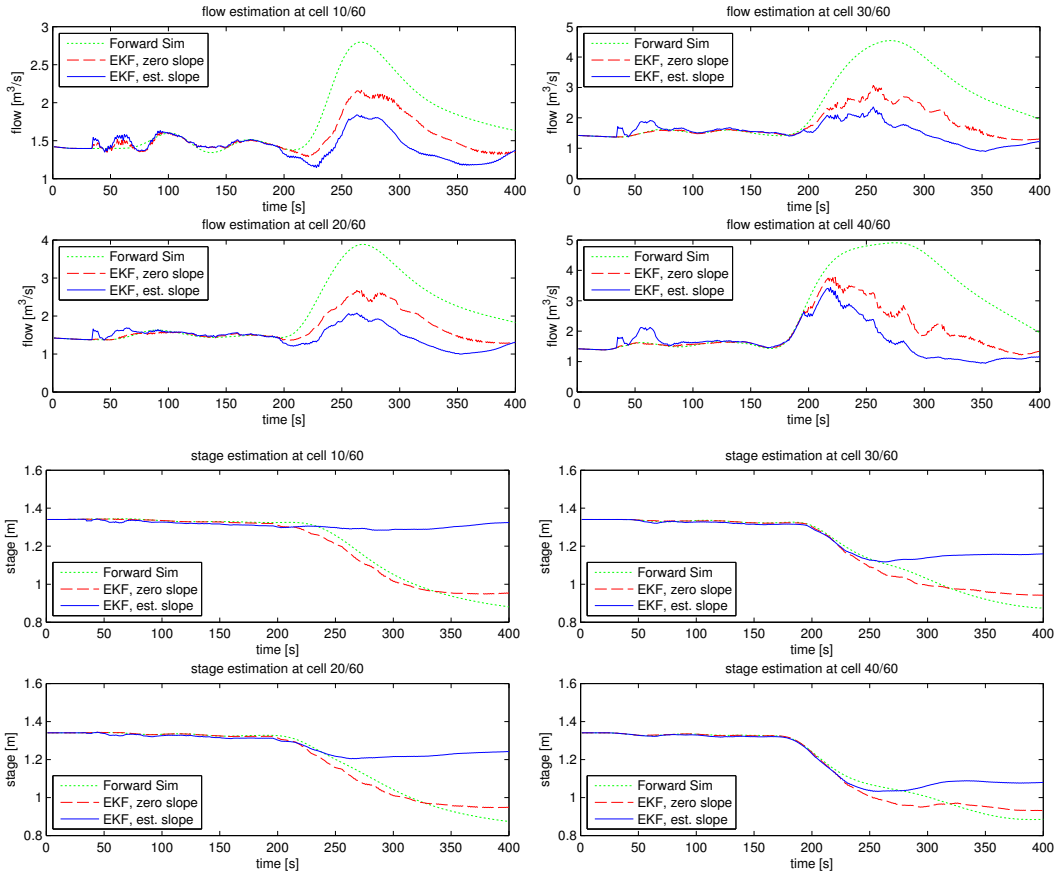


Figure 4.7: The flow (m^3/sec) (top) and stage (m) (bottom) at the 10^{th}, 20^{th}, 30^{th}, 40^{th} cells, forward simulation (green), EKF with zero bed slope (red), EKF with estimating bed slope (blue).

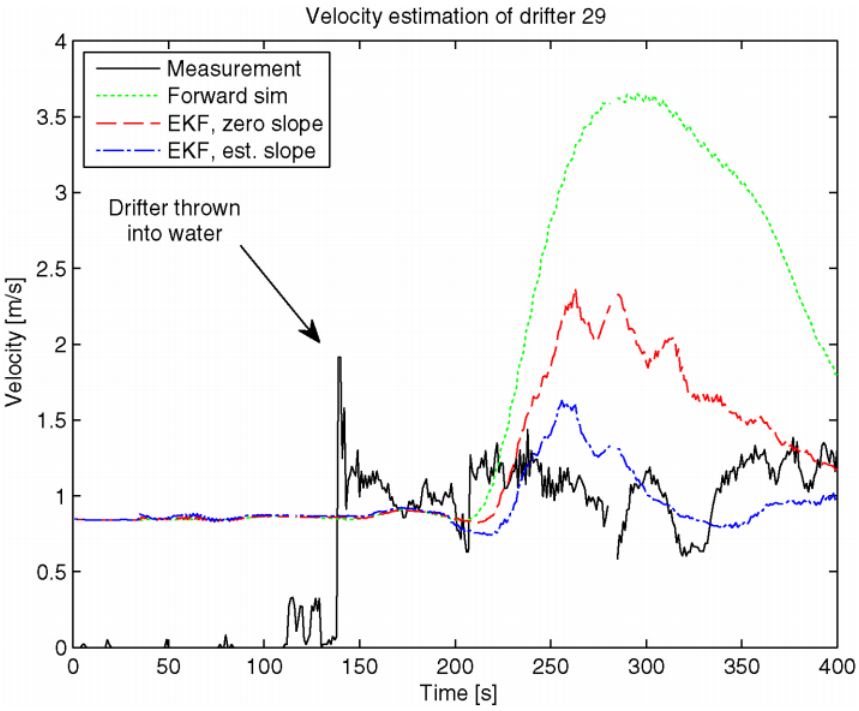


Figure 4.8: the velocity of the 6th drifter, forward simulation (green), EKF with zero bed slope (red), EKF with estimating bed slope (blue) and the actual drifter measurements (black).

Chapter 5

State estimation in large-scale networks of open channels using sequential Monte Carlo

In the previous chapters, we focused on estimation of flow in one open channel or a small network of open channels and we applied filtering-based techniques to perform real-time state and parameter estimation. For large networks, the model will have high dimensions and computationally efficient and scalable methods are needed to make the real-time estimation tractable.

In the last decade, *sequential Monte Carlo methods*, also known as *Particle Filters* have attracted a lot of attention among researchers and practitioners [59, 78, 21, 45, 46]. Particle filters and their variations have been extensively applied to estimation problems in geosciences, such as meteorology and hydrology, due to their generality and scalability. More specifically, particle filters can be applied to nonlinear systems and they do not require Gaussianity assumptions on the noises. In [57, 83], particle filters have been used to assimilate water stage measurements of rivers obtained from the *Synthetic Aperture Radar* (SAR) into hydraulic models. In [86], the authors use the particle filter to estimate parameters as well as the state in a hydrologic model. In particle filters, the posterior probability density function is approximated by a number of particles with their corresponding weights. These particles are propagated forward and their weights are updated at every time step. A larger number of particles results in more accurate results while it increases the computational cost of the method. Nonetheless, particle filters have shown to encounter different problems when implemented on various systems. The most critical issue observed in implementations of particle filters is the degeneracy problem [46]. When degeneracy happens, almost all of the particle weights vanish after a number of iterations meaning that most of the samples

get too far from the actual state of the system and consequently they no longer contribute to approximating the posterior density function. Different methods have been developed to deal with the degeneracy problem among which *sampling Importance Resampling (SIR)* filter is a most commonly used approach [72]. In the SIR filter, after each time step, the density function is resampled so that the samples with small weights are discarded and more probable samples are duplicated according to their weights.

A more subtle issue which usually arises with particle filters is that even with resampling, a lot of particles end up having small weights meaning that they correspond to low-probability regions. Hence, the number of particle contributing to the approximation of the posterior density function is usually smaller than the actual number of particles and most of the computational effort is wasted on unlikely particles. To overcome this problem, *Implicit Particle Filter* is introduced in [39, 38, 87]. Implicit particle filtering is a method to obtain high-probability samples from the density function. Implicit sampling requires solving an underdetermined equation for each particle at every time step when a measurement becomes available. While the cost of sampling can be higher in implicit filters, more accurate results may be obtained with a smaller number of particles as the particles belong to the high-probability region of the density function.

In the current chapter, our goal is to evaluate the performance of different types of Monte Carlo methods for real-time state estimation of water flow in complex networks of open channels. In particular, we are interested to see how implicit particle filters, developed recently, perform in a practical application compared to particle filters in terms of both accuracy and computational complexity. After constructing a state-space model for the network under consideration from the Saint-Venant equations, we apply a number of state estimation methods to incorporate some available measurements into the model in two situations: when measurements of the system are available at every time step and when the measurements become available intermittently. Given the observation model is linear and we assume the noises are Gaussian, we apply the optimal SIR filter to the system in which case $p(x_k|x_{k-1}, z_k)$, which is chosen as the importance density, happens to be a Gaussian density function. We also consider a case in which measurements of the system become available intermittently, i.e. we do not have measurements at every time steps, and we apply the implicit particle filter to incorporate these measurements in order to obtain estimates. We use the random map method [87] to solve the implicit sampling equation. In this method, the samples are parametrized via a random map and samples are obtained by solving an algebraic equation of one variable, for each sample, which is obtained from substituting the random map in the implicit sampling equation. To calculate the minimum of the function F_j, which is required for the sampling equation, and for a matrix used in the random map, the Jacobian and Hessian matrices of this function need to be calculated for each particle and each step of the minimization algorithm. As will be seen, although, this function is algebraically very complex for our system, we are able to calculate most entries of the Jacobian and Hessian matrices of this function analytically. This significantly reduces the computational cost of the implicit sampling.

We also consider a few different heuristics to improve the estimation results and to reduce the computational cost of the methods. We consider a case in which observations are

available at every time step. But, we apply the implicit particle filter and do the sampling over time intervals of a desired length. We change the implicit filter equations slightly to incorporate all the available measurements during the interval. As will be seen, this heuristic method is very effective in the case of the system under consideration. We believe this block-sampling implementation of implicit particle filter when measurements are available at every time step can be beneficial in the case of dynamic systems with *band-diagonal* structure, i.e. systems where the value of the state at each cell is determined by the value of the state at the neighboring cells at the previous time step. Note that most physical system lie in this category since in such systems, information propagates in space continuously with time. We will elaborate more on this in section 5.3.1.

As a second heuristics, we propose a *maximum a posteriori (MAP)* estimation method to calculate the state trajectory over the time interval between two measurements at time steps $k+1$ and $k+r$, which maximizes the posterior density function in this interval. This requires the knowledge of the conditional probability density function $p(x_k|z_{1:k})$ which we approximate by a Gaussian density function, mean and covariance of which are calculated from the estimation over the previous interval. As will be seen, this amounts to a probabilistic version of weak constraint 4D-Var [113, 114] which is a variational data assimilation method used in meteorology. In cases in which the posterior density is symmetric or is almost symmetric and the approximation of conditional density $p(x_k|z_{1:k})$ with a Gaussian density is not too inaccurate, the MAP method may provide more accurate estimates while having a much lower computational cost than the implicit filter. In fact, after implementation and comparison of the performance of this method with the other estimation schemes, it will be seen that this method outperforms the implicit particle filter in terms of both accuracy and computational cost for the application of interest.

5.1 Optimal sampling Importance Resampling Filter

In Bayesian estimation, the goal is to recursively calculate the conditional *probability density function* (pdf) $p(x_k|z_{1:k})$, where x_k is the state vector at time k and $z_{1:k}$ is the set of measurements obtained up to time step k. Assuming the initial state pdf $p(x_0)$ is known, the pdf $p(x_k|z_{1:k})$ may be calculated recursively in two steps, *prediction step* and *update step*. The prediction step uses the state-space model to propagate the conditional pdf forward in time. In other words, it calculates $p(x_k|z_{1:k-1})$ given $p(x_{k-1}|z_{1:k-1})$ via the Chapman-Kolmogorov equation

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1} \tag{5.1}$$

In the update step, when the measurements z_k become available, the conditional pdf is updated using the Bayes' rule

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})} \tag{5.2}$$

where

$$p(z_k|z_{1:k-1}) = \int p(z_k|x_k)p(x_k|z_{1:k-1})dx_k \tag{5.3}$$

While the above set of equations theoretically solve the Bayesian estimation problem, analytic solutions are tractable only in certain simplified cases, e.g. Kalman filter for linear systems with Gaussian noise [68]. For more general cases, different approximate solutions have been devised. Extended Kalman filters [18], approximate grid-based filters, unscented Kalman filters [66, 116, 61] and particle filters [45] are examples of these approximate methods.

Particle filtering is a sequential Monte Carlo method which calculates approximate solutions to the above equations for a general case of nonlinear systems with arbitrary process and measurement noises. The basic idea behind particle filters is that the posterior pdf $p(x_{0:k}|z_{1:k})$, where $x_{0:k} = \{x_j, j = 0, \cdots, k\}$ is the set of all state vectors up to time k, is approximated using a number of particles or random samples with their corresponding weights (probabilities). In other words

$$p(x_{0:k}|z_{1:k}) \approx \sum_{i=1}^{N_s} w_k^i \delta(x_{0:k} - x_{0:k}^i) \tag{5.4}$$

where $x_{0:k}^i$ is the i^{th} sample and w_k^i is its corresponding weight (the weights are normalized so they sum to one) and N_s is the number of samples. The estimates are computed using the particles and their associated weights and the weights are chosen using the principle of importance sampling [23].

A common problem with the sequential importance sampling particle filter is the degeneracy problem. However, a good choice of importance density can reduce the degeneracy of the particles. In this work, we choose the importance density $q(x_k|x_{k-1}^i, z_k)$ to be

$$\begin{aligned} q(x_k|x_{k-1}^i, z_k) &= p(x_k|x_{k-1}^i, z_k) \\ &= \frac{p(z_k|x_k, x_{k-1}^i)p(x_k|x_{k-1}^i)}{p(z_k|x_{k-1}^i)} \end{aligned} \tag{5.5}$$

It has been shown [46] that this choice of importance density minimizes the variance of *true weights*, w_k^{*i} defined as $w_k^{*i} = p(x_k^i|z_{1:k})/q(x_k^i|x_{k-1}^i, z_k)$ which in turn maximizes the effective sample size defined as $N_{\text{eff}} = \dfrac{N_s}{1 + \text{Var}(w_k^{*i})}$. We assume the process and measurement noises are mutually independent and *independent identically distributed* (i.i.d.) Gaussian and

$$v_{k-1} \sim \mathcal{N}(\mathbf{0}, Q_{k-1}) \tag{5.6}$$
$$e_{k-1} \sim \mathcal{N}(\mathbf{0}, R_{k-1}) \tag{5.7}$$

With this assumption, for systems with nonlinear dynamics and linear measurement model

$$x_k = f_k(x_{k-1}) + v_{k-1} \tag{5.8}$$
$$z_k = H_k x_k + e_k \tag{5.9}$$

it has been shown that $p(x_k|x_{k-1}^i, z_k)$ is Gaussian [46, 43] and

$$p(x_k|x_{k-1}, z_k) = \mathcal{N}(m_k, \Sigma_k) \tag{5.10}$$
$$p(z_k|x_{k-1}) = \mathcal{N}(H_k f_k(x_{k-1}), Q_{k-1} + H_k R_k H_k^T) \tag{5.11}$$

with

$$\Sigma_k^{-1} = Q_{k-1}^{-1} + H_k^T R_k^{-1} H_k \tag{5.12}$$
$$m_k = \Sigma_k(Q_{k-1}^{-1} f_k(x_{k-1}) + H_k^T R_k^{-1} z_k) \tag{5.13}$$

With this choice of importance density, the weights update equation simplifies to [21]

$$w_k^i \propto w_{k-1}^i p(z_k|x_{k-1}^i) \tag{5.14}$$
$$= w_{k-1}^i \int p(z_k|x_k')p(x_k'|x_{k-1}^i)dx_k' \tag{5.15}$$

When the measurements are obtained, (5.11) can be used to calculate $p(z_k|x_{k-1}^i)$ to update the weights using equation (5.14).

In cases in which the degeneracy occurs even with this choice of importance density, resampling can be done whenever the effective sample size becomes too small [34]. In this resampling method, each particle generates a number of duplicates proportional to its weight and particles with small weights are discarded. A pseudo-code description of the resampling algorithm is provided in Algorithm 1 below [21]. The algorithm generates a new set of particles $\{x_k^{j*}\}_{j=1}^{N_s}$ with equal weights $\frac{1}{N_s}$.

Algorithm 1: Resampling Algorithm

$[\{x_k^{j*}, w_k^j, i^j\}_{j=1}^{N_s}] = \text{RESAMPLE}[\{x_k^i, w_k^i\}_{i=1}^{N_s}]$
Initialize the CDF: $c_1 = 0$
for $i = 2$ to N_s **do**
  Construct CDF: $c_i = c_{i-1} + w_k^i$
end for
Start at the bottom of the CDF: $i = 1$
Draw a starting point: $u_1 \sim \mathbb{U}[0, N_s^{-1}]$
for $j = 1$ to N_s **do**
  Move along the CDF: $u_j = u_1 + \dfrac{j-1}{N_s}$
  while $u_j > c_j$ **do**
    $i = i + 1$
  end while
  Assign sample: $x_k^{j*} = x_k^i$
  Assign weight: $w_k^j = \dfrac{1}{N_s}$
  Assign parent: $i^j = i$
end for

Algorithm 2 below illustrates the particle filter with resampling

---

Algorithm 2: Particle filter with resampling

$[\{x_k^{j*}, w_k^j\}_{j=1}^{N_s}] = \text{PF}[\{x_{k-1}^i, w_{k-1}^i\}_{i=1}^{N_s}, z_k]$
for $i = 1$ to N_s **do**
  Draw $x_k^i \sim q(x_k|x_{k-1}^i, z_k) = p(x_k|x_{k-1}^i, z_k)$ from (5.10)
  Assign the particle a weight, w_k^i, using (5.15).
end for
Calculate total weight: $t = \sum_{i=1}^{N_s} w_k^i$
for $i = 1$ to N_s **do**
  Normalize: $w_k^i = \dfrac{w_k^i}{t}$
end for
Calculate $\hat{N}_{\text{eff}} = \dfrac{1}{\sum_{i=1}^{N_s}(w_k^i)^2}$
if $\hat{N}_{\text{eff}} < N_T$ **then**
  Resample using algorithm 1:
  $[\{x_k^i, w_k^i, -\}_{i=1}^{N_s}] = \text{RESAMPLE}[\{x_k^i, w_k^i\}_{i=1}^{N_s}]$
end if

5.2 Implicit Particle Filter

As discussed in the previous section, the main issue with the particle filters is the fact that many of the particles end up in low-probability regions and consequently, a considerable amount of computational effort is wasted in propagating these particles forward while they do not contribute to the approximation of the posterior density. Implicit particle filters have been developed to solve this problem [39, 38, 87]. The basic idea behind implicit particle filters is to create samples with high probability. Although, the computational cost per particle is in general higher in implicit filters, a better accuracy may be obtained by using a smaller number of particles than traditional particle filters.

We consider a case in which intermittent observations are available. Suppose that we have a collection of N_s particles, $\{x_k^j, \text{for } j = 1, \cdots, N_s\}$ and we have a set of observations $z_{0:k}$ up to time step k. We consider a case in which the next observation of the system, z_{k+r} becomes available at time $k+r$. We can write the posterior as

$$\begin{aligned} P(x_{0:k+r}^j | z_{0:k}, z_{k+r}) &= P(x_{0:k}^j | z_{0:k}) \\ &\times P(x_{k+1}^j | x_k) \cdots P(x_{k+r-1}^j | x_{k+r-2}) \\ &\times P(x_{k+r}^j | x_{k+r-1}^j) P(z_{k+r} | x_{k+r}^j) / Z_k \end{aligned} \tag{5.16}$$

where Z_k is a normalization factor. The goal of implicit sampling is to obtain high probability samples from $P(x_{0:k+r}^j | z_{0:k}, z_{k+r})$. This is done by defining a function $F_j(x_{k+1:k+r})$ via [87]

$$\begin{aligned} \exp(-F_j(x_{k+1:k+r}^j)) &= P(x_{k+1}^j | x_k) \cdots P(x_{k+r-1}^j | x_{k+r-2}^j) \\ &\times P(x_{k+r}^j | x_{k+r-1}^j) P(z_{k+r} | x_{k+r}^j) / Z_k \end{aligned} \tag{5.17}$$

A high-probability sample can be obtained by solving the following algebraic equation for $x_{k+1:k+r}^j$

$$F_j(x_{k+1:k+r}^j) - \phi_j = \frac{1}{2} \xi_j^T \xi_j \tag{5.18}$$

where ξ_j is a sample from a zero-mean Gaussian distribution, $P_\xi = \mathcal{N}(0, I)$ and $\phi_j = \min F_j(X)$. It is clear that with this sampling method, we obtain a sample with high probability.

The weights corresponding to the particles are calculated as follows

$$w_{k+1}^j \propto w_{k+1}^j \exp(-\phi_j) J \tag{5.19}$$

where $J = \left| \det \frac{\partial x_{k+1:k+r}^j}{\partial \xi_j} \right|$ is the Jacobian of the map between ξ_j and the particle trajectory $x_{k+1:k+r}^j$.

In the special case of the system considered in this article in which the observation model is linear and the process noise and measurement noise are assumed to be Gaussian,

the function F_j will have the following form

$$
\begin{aligned}
F_j(x^j_{k+1:k+r}) = \log\Bigg(& \frac{1}{(2\pi)^{(nr+m)/2}} \prod_{i=k}^{k+r} \frac{1}{\sqrt{\det Q_i}} \frac{1}{\sqrt{\det R_{k+r}}} \Bigg) \\
& + \frac{1}{2} \sum_{i=k}^{k+r-1} (x_{i+1} - f(x_i, u_i))^T Q_k^{-1} (x_{i+1} - f(x_i, u_i)) \\
& + \frac{1}{2} (z_{k+r} - H_{k+r} x_{k+r})^T R_{k+r}^{-1} (z_{k+r} - H_{k+r} x_{k+r})
\end{aligned} \tag{5.20}
$$

To implement the implicit particle filter for the system considered, the main tasks at each time step are calculating $\min F_j(X)$, the Jacobian of the implicit map J, and solving equation (5.18) for each particle to obtain a sample. In what follows, we elaborate on how we handle each one of these problems for our application.

5.2.1 Calculating the minimum of implicit sampling functions

To calculate $\min F_j(X)$, we use the Newton's method for which we need to calculate the Jacobian and Hessian of F_j. Note that the function F_j is in general a different function for each particle. Therefore, the Jacobian and Hessian must be calculated for each particle individually. This may make the implicit sampling procedure computationally expensive in cases in which the Jacobian and Hessian of F_j cannot be calculated analytically. Fortunately, in our case we can calculate the partial derivatives for the interior grid points analytically. And we use a numerical method to calculate the partial derivatives at boundary points. This reduces the computational cost of the implicit sampling significantly.

We can write the Jacobian, J^j, of F_j as

$$J^j(x^j_{k+1:k+r}) = (A_{k+1}, \cdots, A_{k+r}) \tag{5.21}$$

$$A_{k+1} = (X_{k+1} - f(x_k))^T Q_k^{-1} \tag{5.22}$$

$$
\begin{aligned}
A_i = & (X_{i+1} - f(x_i))^T Q_i^{-1} - (X_{k+1} - f(x_k))^T Q_i^{-1} J_f(x_i) \\
& \text{for } i = k+2, \cdots, k+r-1
\end{aligned} \tag{5.23}
$$

$$
\begin{aligned}
A_{k+r} = & (X_{k+r+1} - f(x_{k+r}))^T Q_{k+r-1}^{-1} \\
& + (Hx_{k+r} - z) R_{k+r-1}^{-1} H
\end{aligned} \tag{5.24}
$$

where $J_f(x^j_k)$ is the Jacobian matrix of the system's state-space function f which we calculate analytically for interior grip points and numerically at the boundary points in each channel. The analytical partial derivatives of f for interior grid points are provided in Appendix **??**.

The Hessian of F_j can be written as a band block-diagonal sparse matrix where the diagonal blocks are as follows

$$H_F = \begin{pmatrix} D_1 & C_1 & 0 & \cdots & 0 \\ C_1^T & D_2 & C_2 & \ddots & \vdots \\ 0 & C_2^T & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & D_{r-1} & C_{r-1} \\ 0 & \cdots & 0 & C_{r-1}^T & D_r \end{pmatrix}$$

With

$$\begin{aligned} C_i &= -J_f(x_{i+1})^T Q_{i+1}^{-1} \\ D_r &= Q_r^{-1} + H_{k+r}^T R^{-1} H_{k+r} \\ D_i &= Q_i^{-1} + J_f(x_i)^T Q_{i+1}^{-1} J_f(x_i) + L_i^l, \ i < r \\ L_i^l &= (x_{i+1} - f(x_i))^T Q_{i+1}^{-1} H_f^l(x_i) \end{aligned}$$

where $\mathcal{H}_f^l(x_i) = \left[\frac{\partial^2 f_s}{\partial x_l \partial x_t}\right]_{s,t}$ is a slice of the Hessian matrix of the system's state-space function f which we calculate analytically for interior grid points and numerically at the boundary points in each channel. The analytical partial derivatives of f for interior grid points are provided in appendix **??**. Note that f_s denotes the s^{th} component of f.

To obtain the initial seed for the Newton method in computation of $\min F_j(X)$, we run the forward simulation with the mean of particles as initial condition and we run a few steps of Newton method to polish the initial guess of the minimizing solution. We use a line-search method to find a proper step size for the Newton method.

5.2.2 Solving the sampling equation via a random map

In order to solve equation (5.18), we use the random map method introduced in [87]. Assuming that the level-sets of the function F_j are closed, the following map is considered as the map between ξ_j and $x_{k+1:k+r}^j$

$$x_{k+1:k+r}^j = \mu_j + \lambda_j L_j^T \eta_j \tag{5.25}$$

where $\eta_j = \xi_j / \sqrt{\xi_j^T \xi_j}$ is the direction of the sample ξ_j of the Gaussian reference variable $\xi_j \sim \mathcal{N}(0, I)$, μ_j is the location of the minimum of F_j, i.e. $F_j(\mu_j) = \phi_j$, and $L_j \in \mathbb{R}^{nr \times nr}$ is an invertible matrix chosen deterministically. It has been observed that taking L_j as the Cholesky factorization of the inverse of the Hessian, i.e. $H_j^{-1} = L_j L_j^T$ is a good choice for L_j specially if F_j is nearly quadratic.

Note that in equation (5.25), only λ_j is unknown after the reference Gaussian variable is sampled. By substituting equation (5.25) in equation (5.18), one obtains an algebraic equation for the single scalar λ_j which can be easily solved. This will yield a sample for the particle trajectory $x_{k+1:k+r}^j$. Assuming the level-sets of function F_j are closed, solving

equation (5.25) for λ_j corresponds to a solution of equation (5.18) in a specific direction, $L_j^T \eta_j$.

5.2.3 Calculating the Jacobian of the implicit sampling function

By differentiating equation (5.25), we obtain [87]

$$\frac{\partial x_{k+1:k+r}^j}{\partial \xi_j} = L_j^T \left(\eta_j \frac{\partial \lambda_j}{\partial \xi_j} \right) + \lambda L_j^T \frac{\partial \eta_j}{\partial \xi_j} \tag{5.26}$$

After some manipulations, the Jacobian of the implicit map can be written as

$$J = 2 \left| \det L_j \right| \rho_j^{1-nr/2} |\lambda_j^{rn-1} \frac{\partial \lambda_j}{\partial \rho_j}| \tag{5.27}$$

where $\rho_j = \xi_j^T \xi_j$ and $\dfrac{\partial \lambda_j}{\partial \rho_j}$ can be calculated numerically.

A pseudo-code description of the implicit particle filter is presented in Algorithm 3.

Algorithm 3: Implicit Particle filter with resampling

$[\{x_k^j, w_k^j\}_{j=1}^{N_s}] = \text{Imp-PF}[\{x_{k-1}^j, w_{k-1}^j\}_{j=1}^{N_s}, z_k]$
for $j = 1$ to N_s **do**
  Calculate $\phi_j = \min F_j(X)$.
  Draw a sample $\xi_j \sim \mathcal{N}(0, I)$
  Calculate the Hessian of F_j, H_j and set $L_j = \text{chol}(H_j^{-1})$
  Substitute the random map, equation (5.25) in the sampling equation (5.18) and solve for λ_j to obtain a sample trajectory $x_{k+1:k+r}^j$.
  Calculate the Jacobian of the implicit map using (5.27).
  Update the particle weight, w_k^j, using equation (5.19).
end for
Calculate total weight: $t = \sum_{j=1}^{N_s} w_k^j$
for $j = 1$ to N_s **do**
  Normalize: $w_k^j = \dfrac{w_k^j}{t}$
end for
Calculate $\hat{N}_{\text{eff}} = \dfrac{1}{\sum_{j=1}^{N_s} (w_k^j)^2}$
if $\hat{N}_{\text{eff}} < N_T$ **then**
  Resample using algorithm 1:
  $[\{x_k^j, w_k^j, -\}_{j=1}^{N_s}] = \text{RESAMPLE}[\{x_k^j, w_k^j\}_{j=1}^{N_s}]$
end if

5.3 Heuristics

5.3.1 Implicit Particle Filter with Block Sampling

In this section, we consider a case in which observations are available at every time step. It is known that in the case of systems with linear observation model and Gaussian noises when measurements are available at every time step, the implicit particle filter becomes equivalent to the optimal SIR filter [87]. Nonetheless, as a heuristic method, we use the implicit particle filter to do the sampling every r time steps while using the measurements at all time steps instead of sampling at every time step. We believe that for systems with *band-diagonal structure*, i.e. systems in which the value of each state at the next time step is determined by the current values of its neighboring states, this heuristic method may improve the estimation results. The shallow water model used in the current article is an example of such a system. As can be seen in the discretized Saint-Venant equations in section 2.2, the value of flow or stage at each cell is determined by values of flow and stage at its neighboring cells at the previous time step. In such systems, the information is propagated in space one cell every time step. Therefore, a measurement provides information about the value of the state at a cell which is r cells far from the location of the measurement at r time steps before. This suggests that using all the measurements obtained over a block of time steps and performing the sampling over the block may improve the results. To implement this method, the implicit function needs to be slightly modified as follows so that the filter uses all the available measurements

$$\begin{aligned} P(x^j_{0:k+r}|z_{0:k+r}) &= P(x^j_{0:k}|z_{0:k}) \\ &\times P(x^j_{k+1}|x_k)\cdots P(x^j_{k+r-1}|x^j_{k+r-2})P(x^j_{k+r}|x^j_{k+r-1}) \\ &\times P(z_{k+1}|x^j_{k+1})\cdots P(z_{k+r}|x^j_{k+r})/Z_k \end{aligned} \tag{5.28}$$

$$\begin{aligned} F_j(x^j_{k+1:k+r}) = \log(&\frac{1}{(2\pi)^{(nr+m)/2}}\prod_{i=k}^{k+r}\frac{1}{\sqrt{\det Q_i}}\frac{1}{\sqrt{\det R_{k+r}}}) \\ &+\frac{1}{2}\sum_{i=k}^{k+r-1}(x_{i+1}-f(x_i,u_i))^T Q_k^{-1}(x_{i+1}-f(x_i,u_i)) \\ &+\frac{1}{2}\sum_{i=k+1}^{k+r}(z_i-H_{k+r}x_{k+r})^T R_{k+r}^{-1}(z_i-H_{k+r}x_{k+r}) \end{aligned} \tag{5.29}$$

And the Jacobian and Hessian must be changed accordingly.

5.3.2 Maximum a posteriori (MAP) estimation

The idea investigated in this section is to propagate the maximum-probability trajectory obtained in the minimization step of the implicit particle filter forward and use that as the

estimates. This significantly reduces the computational cost of the method as there will be no need to create particles. Moreover, the minimization problem needs to be solved only once instead of once for each particle. The simplest thing to do would be to calculate $\max_{x_{k+1:k+r}} P(x_{k+1}, \cdots, x_{k+r}|x_k, z_{k+r})$ with using x_k as a deterministic vector obtained from the maximization over the previous time interval. However, this does not take into account the uncertainty in the value of x_k. In order to account for the uncertainty in x_k, one can consider x_k as a random variable. Considering x_k as a random variable, we aim to find the *maximum a posteriori (MAP)* estimate of the state trajectory over the interval between time steps $k+1$ and $k+r$, i.e. the solution of the following optimization problem

$$\hat{x}_{k:k+r} = \arg\max_{x_{k:k+r}} P(x_k, \cdots, x_{k+r}|z_{1:k+r}) \tag{5.30}$$

The cost function in equation (5.30) can be factored as follows

$$\begin{aligned} P(x_{k:k+r}|z_{1:k+r}) &= \frac{P(x_{k:k+r}|z_{1:k})P(z_{k+r}|z_{1:k}, x_{k:k+r})}{P(z_{k+r}|z_{1:k})} \\ &= \frac{P(x_k|z_{1:k})P(x_{k+1:k+r}|x_k)P(z_{k+r}|x_{k+r})}{P(z_{k+r}|z_{1:k})} \end{aligned} \tag{5.31}$$

Defining the function $\tilde{F}$ as

$$\exp(-\tilde{F}(x_{k:k+r})) = P(x_k|z_{1:k})P(x_{k+1:k+r}|x_k)P(z_{k+r}|x_{k+r}) \tag{5.32}$$

we have

$$\hat{x}_{k:k+r} = \arg\min_{x_{k:k+r}} \tilde{F}(x_{k:k+r}) \tag{5.33}$$

Now, we explain how we solve the above optimization problem. It can be readily seen that

$$\tilde{F}(x_{k:k+r}) = F(x_{k+1:k+r}) - \log(P(x_k|z_{1:k})) \tag{5.34}$$

where F is the function defined in equation (5.17). The posterior density function $P(x_k|z_{1:k})$ is not generally available, but we approximate it with a Gaussian density, $\mathcal{N}(\hat{x}_k, K_k)$. The mean, $\hat{x}_k$, is the estimate of x_k obtained from solving the MAP estimation over the previous interval. To find the covariance, we approximate $P(x_{k-r:k}|z_{1:k})$ with a Gaussian density with covariance being equal to the inverse of the Hessian of $P(x_{k-r:k}|z_{1:k})$ which is already available to us from solving the MAP estimation over the previous time interval. We then obtain the covariance of the marginal density $P(x_k|z_{1:k})$ which is the desired covariance matrix K_k.

For the case of the system under consideration, we can write

$$\begin{aligned} \tilde{F}(x_{k:k+r}) = &\log\left(\frac{1}{(2\pi)^{(nr+m)/2}}\prod_{i=k}^{k+r}\frac{1}{\sqrt{\det Q_i}}\frac{1}{\sqrt{\det R_{k+r}}}\right) \\ &+ \frac{1}{2}\sum_{i=k}^{k+r-1}(x_{i+1} - f(x_i, u_i))^T Q_k^{-1}(x_{i+1} - f(x_i, u_i)) \\ &+ \frac{1}{2}(z_{k+r} - H_{k+r}x_{k+r})^T R_{k+r}^{-1}(z_{k+r} - H_{k+r}x_{k+r}) \\ &+ \frac{1}{2}(x_k - \hat{x}_k)^T K_k^{-1}(x_k - \hat{x}_k) \end{aligned} \tag{5.35}$$

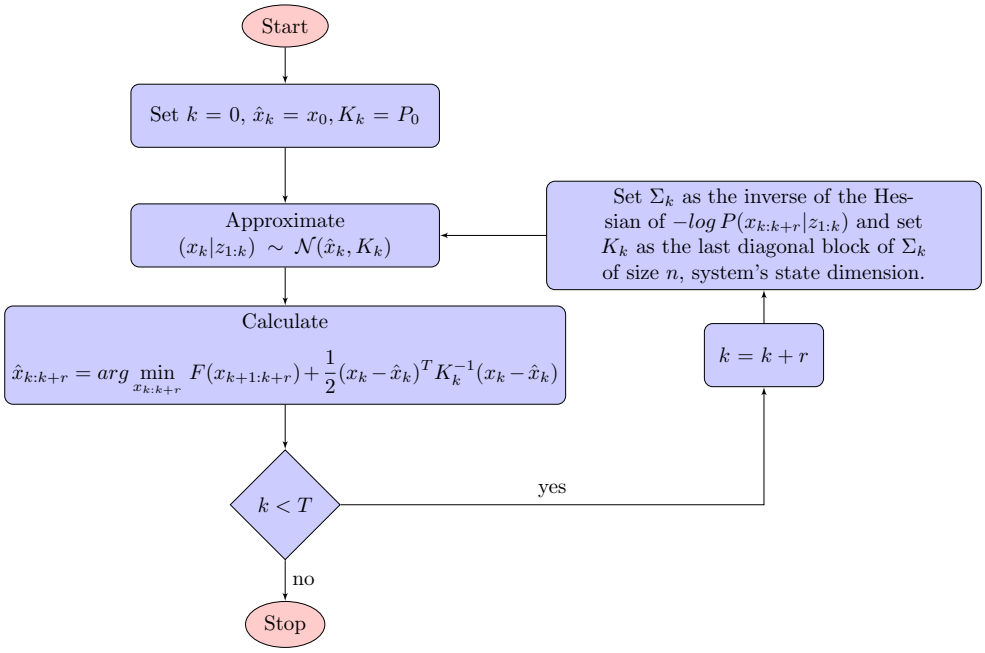


Figure 5.1: Flow chart of the MAP estimation heuristic method.

The optimization problem (5.33) can be solved using the Newton method with the forward simulation predicted state trajectory as the initial seed to obtain an optimal solution. Note that $\tilde{F}(x_{k:k+r})$ has one additional quadratic term than $F(x_{k+1:k+r})$ and the calculated Jacobian and Hessian of F need to be slightly changed to obtain those of $\tilde{F}(x_{k:k+r})$.

It is not hard to see that the above approach is similar to the weak constraint 4D-Var method where a cost function consisting of quadratic terms accounting for model error, measurement error and error in the initial conditions is minimized [113, 114].

Figure 5.1 shows a flow chart of the MAP estimation algorithm presented in this section.

Note that in cases in which the posterior density $P(x_{k+1}, \cdots, x_{k+r}|x_k, z_{k+r})$ is symmetric, the MAP estimates will be more accurate than the estimates obtained from the weighted average of the particles and as the number of particles converges to infinity, the weighted average estimate converges to the MAP estimate which is the trajectory which maximizes the posterior density. In cases in which the posterior density $P(x_{k+1}, \cdots, x_{k+r}|x_k, z_{k+r})$ is close to symmetric and the approximation of $P(x_k|z_{1:k})$ with a Gaussian density is reasonable, we expect that the MAP method introduced above to be a very appealing estimation method as it may provide more accurate estimates with a much lower computational cost than the optimal SIR and the implicit particle filter. As we apply the method for the system under consideration, we will see in section 5.4 that this is the case for this system. In other words, we obtain more accurate results with a smaller computational cost by using the MAP method.

Table 5.1: The names and geometry information of the subchannels in the open channel network in Sacramento-San Joaquin Delta in California used for the implementations.

Channel	River	Length	Avg. width	Avg. depth
1-2	Italian Slough	14198	234.0	14.0
2-3	Italian Slough	2723	203.4	16.4
2-5	Italian Slough	3227	437.5	10.3
3-4	Old River	4754	351.3	21.8
3-5	Old River	5022	351.2	22.0
5-6	Old River	4313	238.4	13.0
5-11	West Canal	10041	253.0	28.0
6-7	Victoria Canal	8760	386.5	18.3
6-8	Old River	2722	276.3	15.2
8-9	Old River	2793	109.0	11.7
Channel	River	Length	Avg. width	Avg. depth
8a9	Old River	5347	157.1	9.0
9-10	Old River	2456	109.0	11.7
9a10	Old River	5062	157.1	9.0
10-11	Old River	7744	198.4	12.4
11-13	Old River	2609	266.0	19.0
13-14	Old River	3857	245.0	17.8
14-16	Old River	12089	176.0	10.0
14-15	Mendota Canal	12500	196.0	18.0
13-17	Grant Line Canal	15831	404.0	16.0

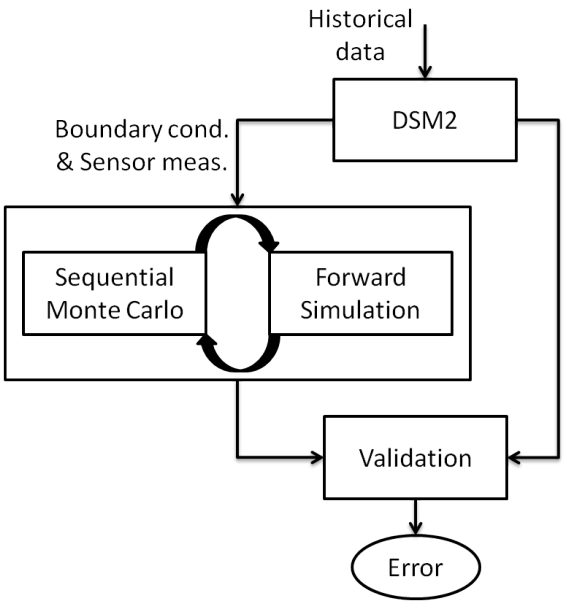


Figure 5.2: A flow diagram of the experiment.

5.4 Implementation

We consider a network of 19 subchannels and one reservoir in the southern part of Sacramento-San Joaquin to implement the data assimilation methods. The Sacramento-San Joaquin Delta, in northern California, is the hub of California's water system. This complex network covers 738,000 acres interlaced with over 1150 km of tidally influenced channels and sloughs and approximately 50 percent of California's average annual streamflow flows to the Delta. Figure 5.3 shows a map of the Delta.

The network considered for implementation consists of the Clifton Court Forebay and its surrounding channels which are parts of the Old River, the Italian Slough, the Mendota Canal, the West Canal, the Victoria Canal and the Grant Line Canal and is located on the northern side of Tracy. Figure 5.4 shows a satellite picture of the area and a map of the network with the channel configuration of the network, considered in building a one-dimensional model of the flow. As can be seen in this figure, this network consists of one reservoir, 19 subchannels and 10 junctions. The total length of the channels in the network is 38,420 m. Figure 6.4 shows a cross section of some of the channels. Note that the channels are not uniform and the presented cross sections correspond to a particular location along the channels. Table 5.1 presents the name and some geometry information about the channels. The channels are represented by their parent nodes from figure 5.4.

Figure 5.2 shows a flow diagram of the experiment. We use the *Delta Simulation Model II* (DSM2) to obtain measurements for boundary conditions and data assimilation as well as to evaluate our results. DSM2 is a one dimensional mathematical model, developed in the California *Department of Water Resources* (DWR), for dynamic simulation of hydrodynamics, water quality and particle tracking in the Delta. DSM2 can calculate stages, flows,

velocities, transport of individual particles, and mass transport processes for conservative and non-conservative constituents, including salt, water temperature, dissolved oxygen, and dissolved organic carbon. We use DSM2 since there are not enough USGS sensor stations in the area of interest to obtain boundary conditions and additional flow measurements needed for the data assimilation. DSM2 is one of the reference models used by the DWR for operations and will be considered in this work as the ground truth. Figure 5.6 compares the discharge and stage at the upstream of Old River (node 4 in Figure 5.4) generated by DSM2 with the USGS sensor measurements for June 10-12, 2006 and illustrates the applicability of DSM2 as a modeling tool to compute discharge and stage at locations where we need them for boundary conditions or as virtual sensor measurements.

5.4.1 Numerical results

We perform an experiment for a period of 25 hours using historical data corresponding to June 12, 2006. We obtain the boundary conditions and measurements used for data assimilation from DSM2. As boundary conditions, discharge is imposed at nodes 1, 7, 15, 16 and 17 and stage is imposed at nodes 4 and 12. Figure 5.7 shows the boundary conditions. The Tracy pumping plant is located at node 15 and as can be seen in Figure 5.7 (b), there is a constant outflow soon after the experiment starts and there is no outflow at node 1 for the period of the experiment. In order to model the measurement noise, we add a zero-mean Gaussian noise with variance of 5 ${}^{ft^6}/_{s^2}$ and 0.05 ft^2 to the boundary conditions for discharge and stage measurements, respectively.

The number of cells in each subchannel is chosen in a way that the spatial step size in the subchannel is close to and not smaller than 900 ft and the temporal step size is chosen to be 15 sec. This choice of spatial step size results in 204 cells for the full network. Since at each internal cell, there are two states, discharge and stage, and there is one state at the boundary cells, we will have a 401 dimensional system. We run DSM2 with spatial step size of 5,000 ft and temporal step size of 15 min. We run DSM2 starting one day earlier so that the effects of inaccurate initial conditions are washed away and the DSM2 results are close to reality from the beginning of the experiment.

Figure 5.8 shows the stage at two different locations computed by our model, the *forward simulation*, compared to the stage at the corresponding locations obtained from DSM2. As can be seen in this figure, the stage obtained from the forward simulation and DSM2 match closely. This is true at all locations throughout the network. However, there is a discrepancy between the discharge computed by the forward sim and DSM2. This is due to the fact that the bathymetry used for the forward simulation is different than that of DSM2. DSM2 computes the water surface (relative free surface elevation with respect to a datum) and we use the same bathymetry used for the forward sim to convert the water surface to stage. This decreases the sensitivity of stage to bathymetry which explains the lower discrepancy in stage computed by DSM2 and the forward sim.

To perform data assimilation, we only focus on assimilating discharge to improve the forward simulation results. We use six discharge measurements at the middle of channels

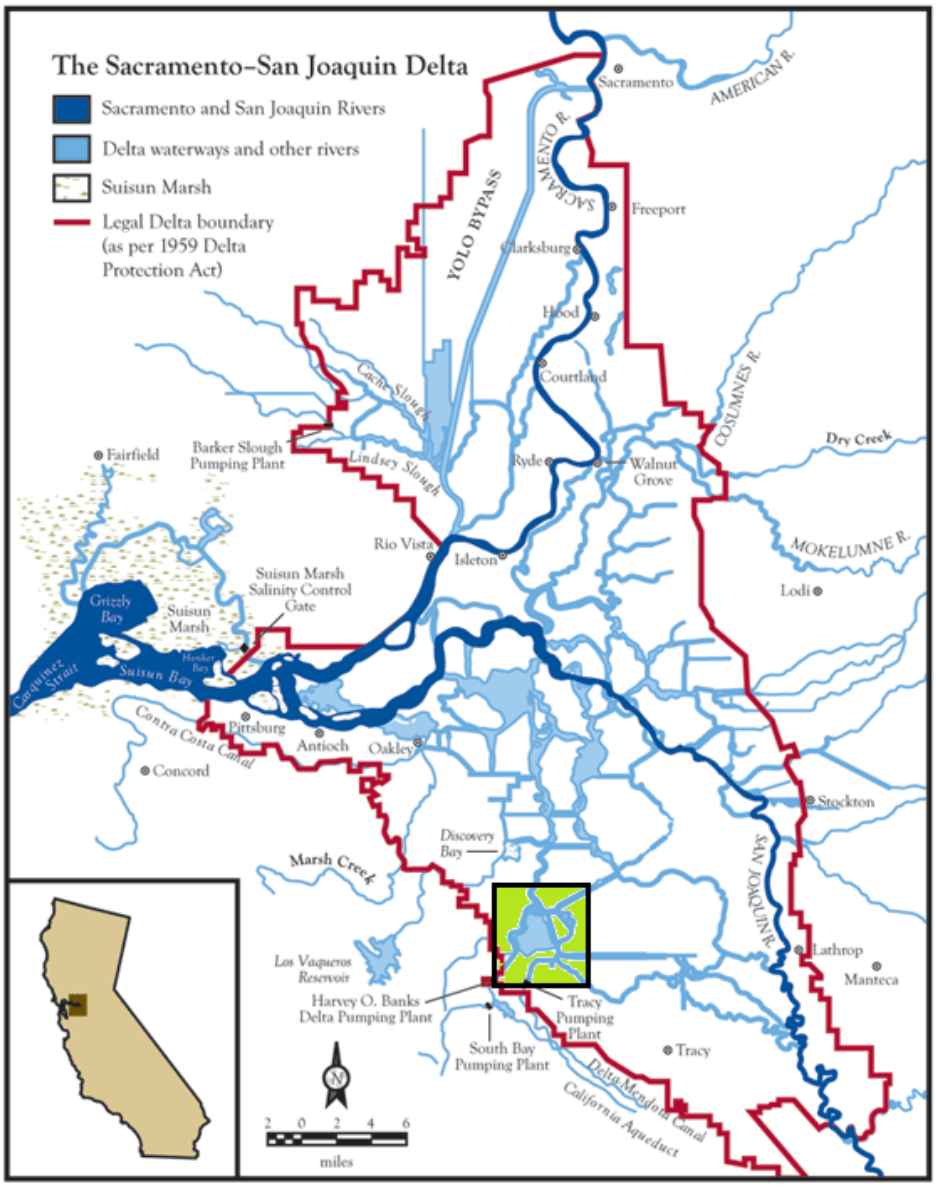
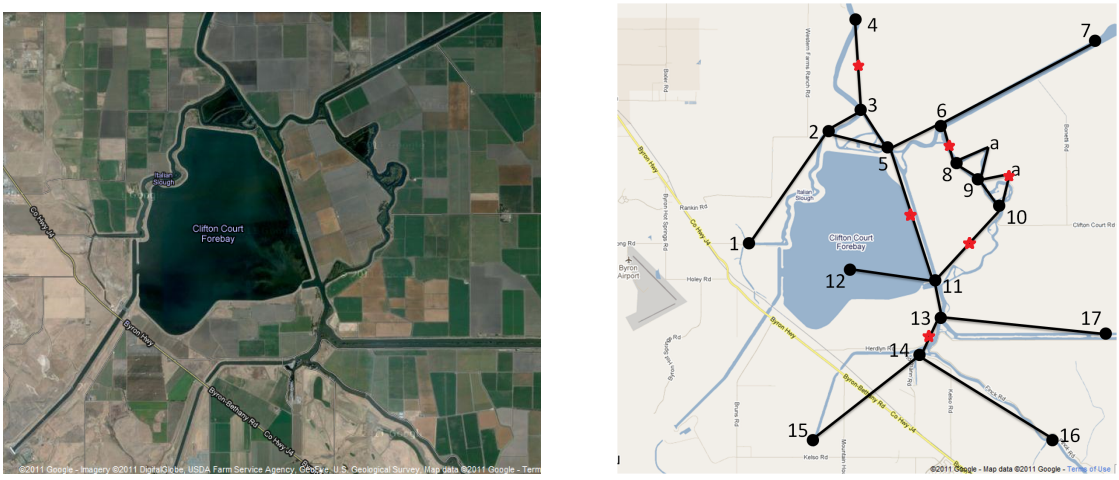


Figure 5.3: The Sacramento-San Joaquin Delta, image adapted from [80]. The small box on the southern part of the Delta is the network considered for implementation of the methods proposed in the current article.



(a) (b)

Figure 5.4: (a) Satellite image of the channel network around the Clifton Court Forebay used for the experiment. (b) The network connectivity of the channel network used for the experiment consisting of 19 subchannels, 10 junctions and one reservoir. As boundary conditions, discharge is imposed at nodes 1, 7, 15, 16 and 17 and stage is imposed at nodes 4 and 12 where node 12 represents the reservoir. The red stars show the locations from which flow measurement are obtained for data assimilation.

3-4, 6-8, 9a10, 10-11, 5-11, 13-14, the locations of which are shown by red stars in Figure 5.4 (b). The process and measurement noises are assumed to be zero-mean white Gaussian noise. We assume that the noise on different measurements are uncorrelated. At each cell, the process noise on discharge is assumed to be correlated with the discharge at its four neighboring cells from each side. The variance on discharge at each cell is taken to be 25 ft^6/s^2 and the correlations are taken to be 20 ft^6/s^2, 12 ft^6/s^2, 8 ft^6/s^2, 4 ft^6/s^2 with the discharge at the neighboring cells, respectively. These approximations of the noise variance were chosen based on sample variances of the process noise at a few locations in the network and other variances and correlations were approximated accordingly. The sensor measurements are obtained from DSM2 and a zero-mean Gaussian noise with a variance of 50 ft^6/s^2 is added to these measurements to simulate the uncertainty in the measurements.

Using these flow measurements, the optimal SIR filter is applied. Figure 5.9 shows the results of the forward simulation and the SIR filter with 1000 particles compared to the corresponding results obtained from DSM2. The discharge at six locations in the network are illustrated for the period of the experiment. As can be seen in this figure, the SIR filter improves the model results significantly. In order to quantify the performance of the methods rigorously, we calculate the relative error throughout the whole domain at each time step using the following formula

$$E(k) = \sqrt{\frac{\sum_{i=1}^{N_{\text{cells}}} (\hat{Q}_i^k - Q_i^k)^2}{\sum_{i=1}^{N_{\text{cells}}} (Q_i^k)^2}} \tag{5.36}$$

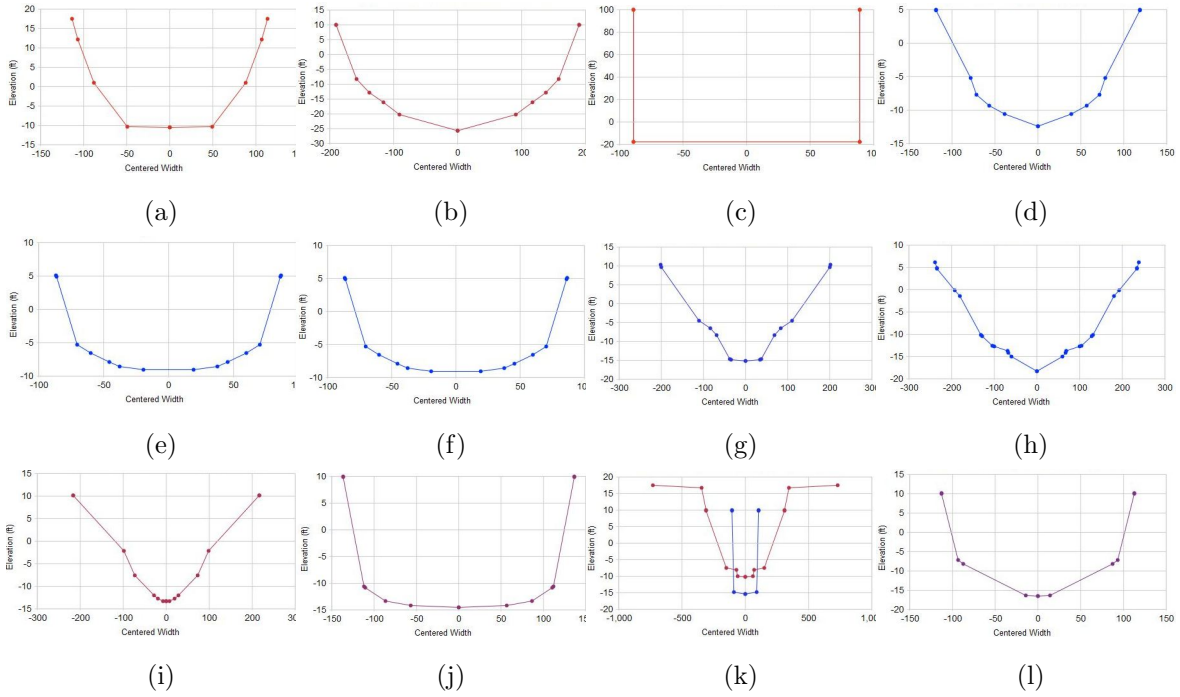


Figure 5.5: The cross-section of some of subchannels in the network used for the experiment, channels (a) 14-16 (b) 3-4 (c) 14-15 (d) 10-11 (e) 9-a-10 (f) 8-a-9 (g) 6-8 (h) 6-7 (i) 5-6 (j) 1-2 (k) 2-5 (l) 2-3.

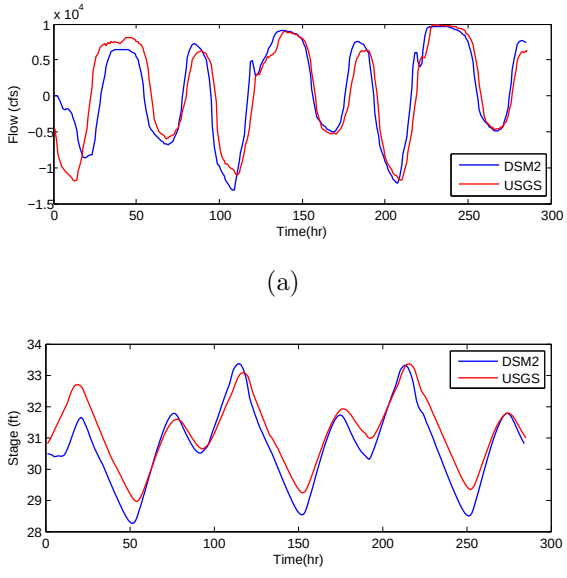


Figure 5.6: A comparison of the DSM2 prediction of flow and stage at node 4 in Figure 5.4 and USGS sensor measurements, corresponding to June 10-12, 2006.

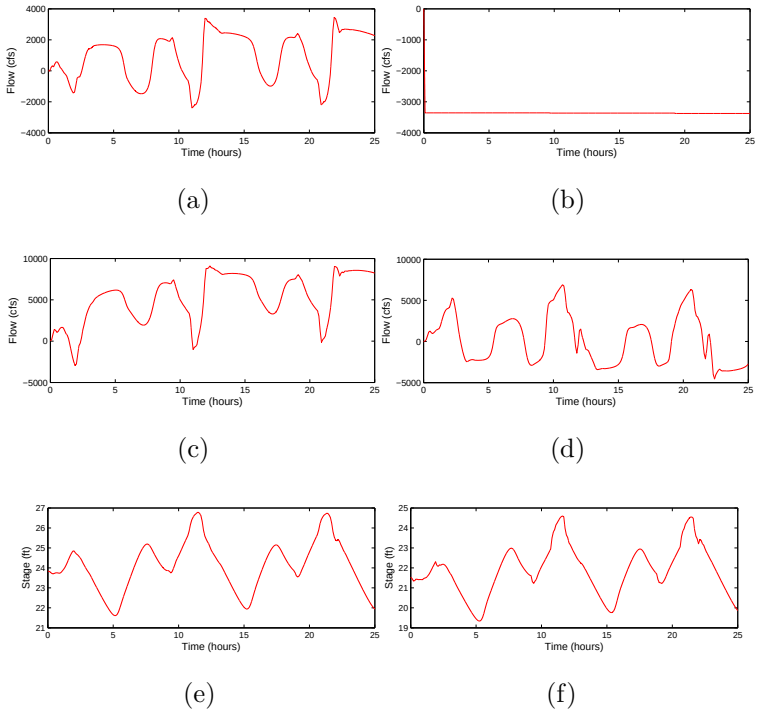


Figure 5.7: The boundary conditions, discharge at nodes (a) 16, (b) 15, (c) 17, (d) 7, and stage at nodes (e) 4 , and (f) 12.

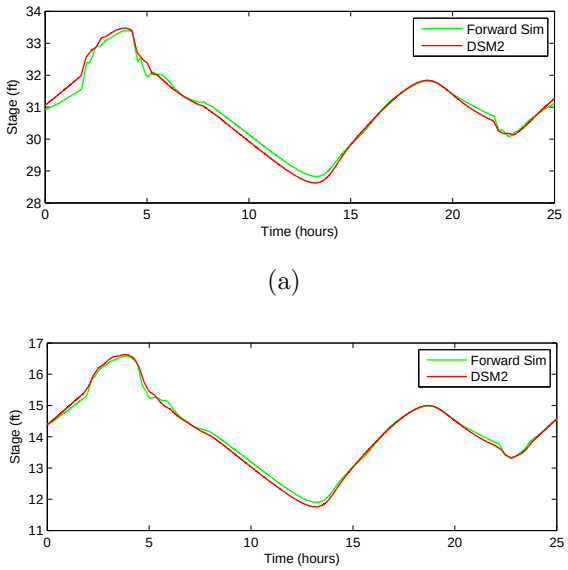


Figure 5.8: The stage computed by the forward simulation compared to DSM2 results, at channels (a) 5-11, and (b) 8-9.

where Q_i^k and $\hat{Q}_i^k$ are the true value of the flow and the estimated flow at cell i and time step k, respectively.

The implicit particle filter is also applied for two cases, a case with sparse measurements and as case in which measurements are available every time step, but we apply the first heuristic method to perform the sampling over intervals of 10 time steps while using all the available measurements. Figure 5.10 shows the time evolution of the relative error corresponding to the forward simulation, the SIR filter with 1000 particles and the implicit particle filter with 50 particles while using measurements at every time step and performing block-sampling. It can be seen in this figure that the relative error of the forward simulation has large peaks corresponding to more than 60 % error. The SIR filter reduces the relative error to below 20 % almost at all times. It can also be seen that using the implicit particle filter with the block-sampling heuristic method discussed in section 5.3.1 improves the results further. The relative error corresponding to the implicit particle filter with block-sampling remains below 10 % most of the time.

The MAP estimation method proposed in section 5.3.2 is also implemented for both cases of sparse measurements and measurements available at every time step. Figure 5.10 compares the time evolution of the relative error of the implicit particle filter with block-sampling with 50 particle and the MAP method for the case where measurements are available at every time step. As can be seen in this figure, the MAP method improves the relative error compared to the implicit particle filter almost at all times.

We implement different estimation methods discussed in previous sections for different scenarios and in order to compare the performance of different methods in different cases, we calculate the average of the relative error per time step over the period of the experiment as follows

$$\bar{E} = \sqrt{\frac{\sum_{k=1}^{T}\sum_{i=1}^{N_{\text{cells}}}(\hat{Q}_i^k - Q_i^k)^2}{\sum_{k=1}^{T}\sum_{i=1}^{N_{\text{cells}}}(Q_i^k)^2}} \tag{5.37}$$

In the cases of SIR filter and the implicit particle filter, the relative error is a random variable due to the randomness in propagation of the particles. Therefore, we perform the assimilation 10 times for each scenario in these cases to calculate the mean and variance of the relative error.

In Table 5.2, the average relative error is provided for the forward simulation, the optimal SIR and implicit particle filters with block-sampling for a few different number of particles, as well as, the MAP method. As can be seen in this table, performing data assimilation using the optimal SIR filter reduces the average error of the model from about 23% to around 10% which is a significant improvement. More interestingly, using the implicit particle filter with block-sampling reduces the average error to about 8%. As discussed before, this is due to the fact that using more measurements before sampling, results in more accurate samples, especially in cases where information propagates in space continuously with time. Also, note that increasing the number of particles improves the results slightly. Unfortunately, we were not able to run the methods for larger number of particles due to the required computational time. For instance, implementing the implicit particle filter with 50 particles for the period

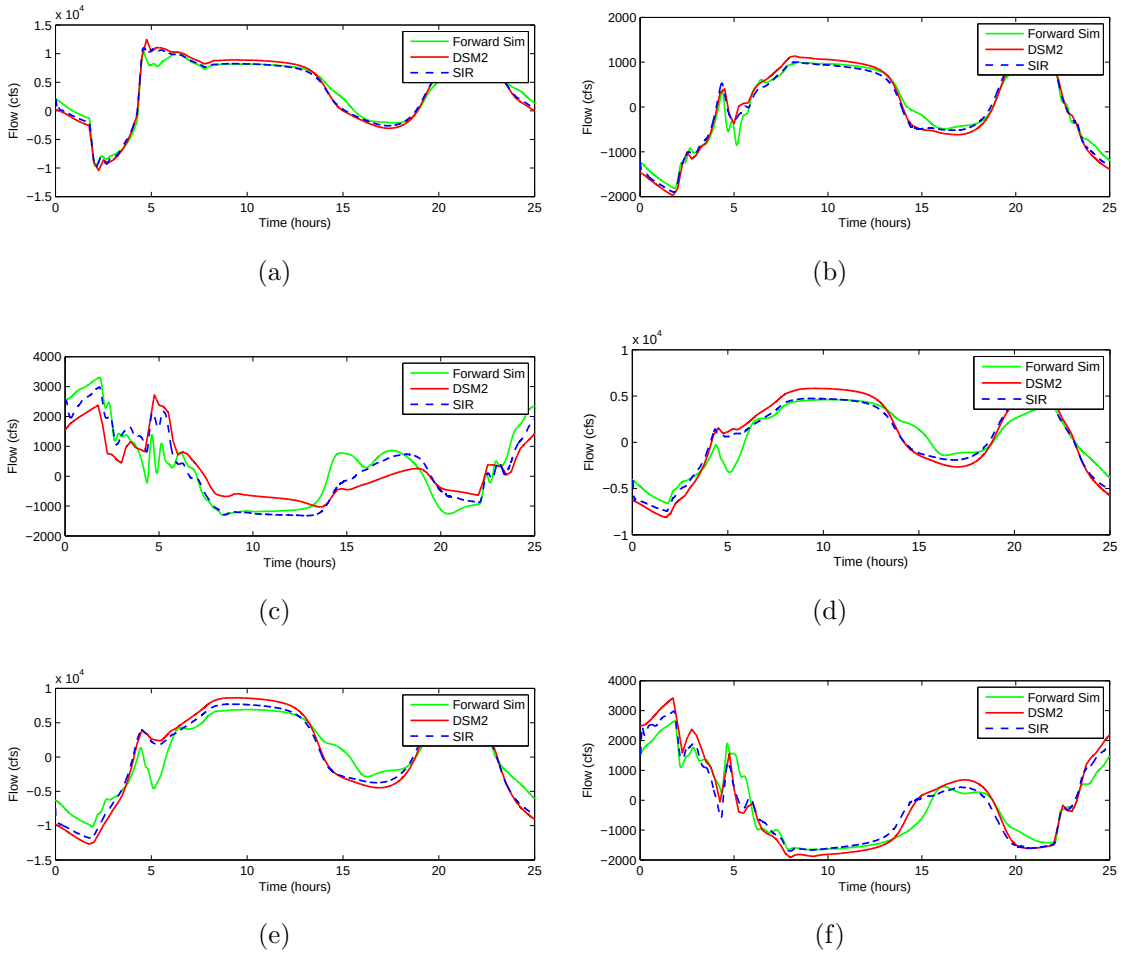
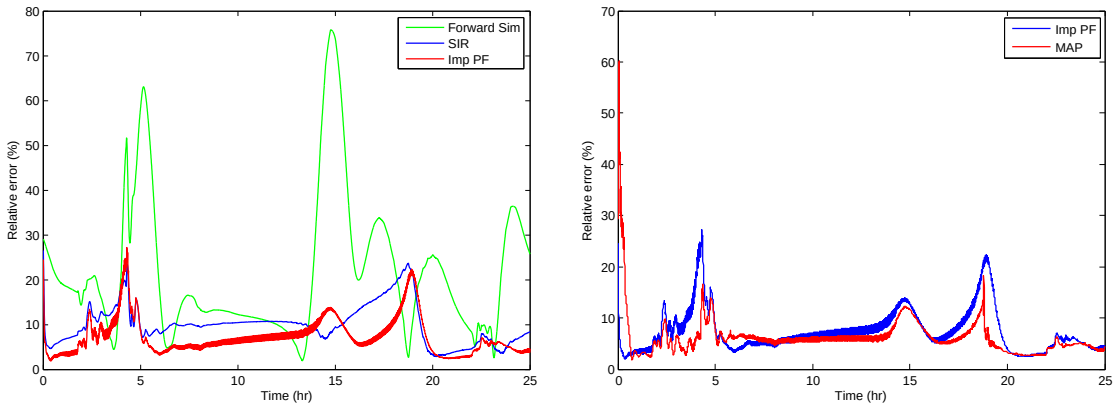


Figure 5.9: Time evolution of discharge at six channels in the network, (a) channel 11-13 (b) channel 8-9 (c) channel 5-6 (d) channel 3-5 (e) channel 3-4 (f) channel 2-5, obtained from the forward simulation, the optimal SIR filter compared with the ground truth, i.e. DSM2 results.



(a) (b)

Figure 5.10: Time evolution of relative error for (a) forward simulation, optimal SIR and implicit particle filter with block-sampling, (b) the implicit particle filter with block-sampling and the MAP method.

of the experiment on a 2.6 GHz Dual Core desktop takes a few days. Finally, note that the MAP method improves the results further and reduces the average relative error to about 7%. It is remarkable that the MAP method provides the best estimate considering the fact that the computational cost of the MAP method is also less than other methods. In the MAP method the minimization is solved only once at every time interval in contrast to once for every particle in the case of the implicit particle filter and there is also no need to solve the sampling equation.

Note that the particle filters even with one particle decrease the model error significantly by taking advantage of the six additional discharge measurements used for the data assimilation.

Table 5.2: Average Relative error corresponding to the forward simulation, the optimal SIR, the implicit particle filters with block-sampling (IMP PF with BS) for a few different number of particles, and the MAP method, for a case where measurements are available at every time step.

Method	Number of particles	Error(%): mean/variance
Forward sim.	-	23.36
SIR	1	10.87 / 0.63
SIR	10	10.41 / 0.48
SIR	100	10.28 / 0.39
SIR	1000	10.15 / 0.30
Imp PF with BS	1	8.03 / 0.41
Imp PF with BS	10	7.74 / 0.26
Imp PF with BS	50	7.55 / 0.21
MAP	-	6.92

We also consider a case in which sparse measurements are available. We assume that measurements become available every 10 time steps and we implement the implicit particle filter and the MAP method to incorporate the sparse measurements into the model. Note that it is not necessary to assume the time interval between the measurements is fixed, although, the implementation is done with this assumption. Table 5.3 shows the relative error in the case of sparse measurements resulted from using the implicit particle filter and the MAP method. As can be seen in this table, the MAP method provides a smaller average error than the implicit filter.

Table 5.3: Average Relative error corresponding to the implicit particle filters (IMP PF) for 10 and 50 particles, and the MAP method, for a case in which measurements are available at every 10 time steps.

Method	Number of particles	Error(%): mean/variance
Imp PF	1	14.31 / 0.47
Imp PF	10	13.88 / 0.32
Imp PF	50	13.68 / 0.27
MAP	-	11.72

Finally, in the case in which measurements are available at every time step, we investigate the effect of r, i.e. the length of the sampling time interval, on the performance of the methods for the case of the implicit particle filter with block-sampling and the MAP method. Table 5.4 provides the relative error corresponding to four different values of r. Note that when r is taken to be 1, the implicit particle filter has the same error as the optimal SIR filter. This is consistent with the fact that in the case of a linear observation model and Gaussian noise, the implicit particle filter recovers the optimal SIR filter. It is also interesting to see that $r = 5$ provides the least error in both cases. We think that this is for two reasons. First, the dimension of the problem increases with r. More precisely, the dimension of the state trajectory over the interval increases linearly with r. Therefore, larger inaccuracies will result from numerical processes, e.g. solving the minimization problem, as the dimension of the problem increases. Second, as r becomes large, some of the measurements used for the estimation of the state trajectory may become irrelevant, i.e. the measurements obtained at the end of the interval may not contain relevant information about the state at the beginning of the interval if the length of the interval, r, is too large.

Finally, in order to compare the computational complexity of each of the methods discussed, we provide the average computational time per time step for each case, while implemented on a 2.6 GHz Dual Core desktop computer, in Table 5.5. It should be emphasized that the programs are not currently written in an efficient way specially in the case of the implicit particle filter. As can be seen in this table, the computational time of the implicit filter with 10 particles is almost the same as that of the optimal SIR with 500 particles. Nonetheless, as seen in table 5.2, the implicit filter with 10 particles produces an average error of about 8% while the optimal SIR produces an average error of about 10%. Also, note that the MAP method has a lower average computational time than the implicit filter. In fact, it can be seen that the computational time of the MAP method is almost that of the

Table 5.4: Average Relative error corresponding to the implicit particle filters with block-sampling (IMP PF with BS) for 10 particles, and the MAP method, for a case in which measurements are available at every time step, for four different values of r.

r	Imp PF with BS	MAP
1	10.89 / 0.48	10.38
5	7.52 / 0.22	6.20
10	7.74 / 0.26	6.92
20	9.83 / 0.35	8.96

implicit filter divided by the number of particles. This is because in the MAP method the minimization, which is the most costly step of the implicit filtering, is done once as opposed to once for every particle in the case of implicit filters.

Table 5.5: Average computational time of the methods per time step, for the optimal SIR, the implicit particle filter with block-sampling (IMP PF with BS) and the MAP method.

Method	Number of particles	Avg. comp. time/time step (sec)
SIR	10	0.14
SIR	50	0.54
SIR	500	5.16
Imp PF with BS	10	5.24
Imp PF with BS	50	25.57
MAP	-	0.55

5.4.2 Summary

Application of a few Monte Carlo methods to estimate flow in open channel networks was the subject of this chapter. Considering a case in which measurements are available at every time step, we implemented the optimal sampling importance resampling filter. We also considered a case in which sparse measurements are available, i.e. measurements become available every r time steps, where r could be a random integer in general (although, we considered a fixed r in this work). We implemented a random map implementation of the implicit particle filter implemented to incorporate the sparse measurements into the model. A heuristic *maximum a posteriori*-based method was also proposed in which the goal is to maximize the posterior density of the state trajectory over the interval given all measurements. By approximating the conditional density of the state at the beginning of the interval with a Gaussian density, we were able to solve the corresponding optimization problem with the same computational cost as the optimization problem in the case of implicit filters. It was observed that the MAP method provides more accurate estimates than the implicit particle filter for the current application while having a lower computational cost.

In the following chapter, we investigate an optimization-based estimation technique for flow estimation in tidally forced channels.

Chapter 6

Flow estimation in tidal channels using the LASSO

In previous chapters, real-time state and parameter estimation of flow in open channels was investigated and different situations were considered. In some applications, estimates of flow state at all grid points may not be desired. In this chapter, we consider a case in which streaming measurements of flow in a tidal channel are available and the goal is to efficiently estimate flow variables in a specific location along the channel in real time. Using the linearized Saint-Venant equations, we derive a z-domain transfer function representation of the flow in the channel which relates the flow variables that are measured to the ones that are desired such that the desired flow variables are the inputs and the measured variables are the outputs of this *multi input-multi output (MIMO)* system. Hence, the estimation problem boils down to an input estimation problem. After parametrizing the inputs using the dominant tidal modes, we cast the parameter estimation problem as an optimization problem which minimizes the l_2-norm of the error (difference between the measurements and the model predictions) with an l_1-norm regularization.

This optimization problem which is the least squares problem with an l_1-norm regularization is called the LASSO in the literature [110] which has generated a lot of interest in the statistics [110, 44], signal processing [22, 30] and machine learning [60, 91] communities, in particular for estimation problems. It has been shown that the solution of the LASSO is sparse in the expression considered in the l_1-norm penalty [110] which is a desirable property in numerous problems in order to achieve model selection [31], data compression [32], or for obtaining interpretable results.

The LASSO can easily be transformed to a *quadratic program* by introducing new slack variables for each component of the l_1-norm which can be solved efficiently using standard over-the-shelf optimization packages [64]. A specialized interior-point method for large-scale problems was introduced in [70]. Other methods to solve the LASSO include iterative

thresholding algorithms [42, 55, 104], feature-sign search [75], bound optimization methods [53] and gradient projection algorithms [54]. Homotopy methods have also been applied to the LASSO to compute the full regularization path when the regularizaion parameter μ_n varies [94, 49, 82]. They are particularly efficient when the solution is very sparse [47]. When the training samples are obtained sequentially, Garrigues et al. [56] introduce a homotopy algorithm to compute the solution of the LASSO recursively. We use this algorithm to solve the LASSO that will be resulted from the estimation approach we consider in this section. The l_1-norm penalty we consider is the difference between the decision variables (unknown parameters) and their estimates from the previous time step. Since the solution to the LASSO is sparse, only a few of estimates of the tidal amplitudes get updated at every time step. In other words, as new measurements of the flow become available, the LASSO is solved recursively to update the estimates of the most significant tidal modes.

6.1 The LASSO problem

The LASSO (6.1) was first introduced in [110]. The LASSO is essentially the least square regression problem with an l_1-norm regularization. In other words, it is a convex optimization problem in which the cost function consists of the l_2-norm of an affine function of the decision variables and an l_1-norm of the decision variables. In mathematical terms, it can be written as the following optimization problem

$$x = \arg\min_{x\in\mathbb{R}^m} \frac{1}{2}\sum_{i=1}^{n}(a_i^T x - y_i)^2 + \mu_n||x||_1 \tag{6.1}$$

For large enough regularization parameters, the solution to the LASSO is sparse. In order to see this fact, note that the LASSO can be written equivalently in its dual form as follows:

$$x = \arg\min_{x\in\mathbb{R}^m} \frac{1}{2}\sum_{i=1}^{n}(a_i^T x - y_i)^2 \qquad : \qquad ||x||_1 \leq \eta \tag{6.2}$$

Figure 6.1 shows the level sets of the cost function which are circles and it can be seen that the optimal solution lies on a vertex of the polytope $||x||_1 = \eta$ which results in a sparse solution.

In this section, we summarize previous work which used the optimality conditions to solve this optimization problem [49, 56]. The objective function of (6.1) is convex and non-smooth since the l_1-norm is not differentiable when there exists an index i such that the i^{th} element of x (denoted x_i) equals zero. There is a global minimum at x if and only if the subdifferential of the objective function at x contains the 0-vector. The subdifferential of the l_1-norm at x is the following set

$$\partial\|x\|_1 = \left\{ v \in \mathbb{R}^m \ : \ \begin{array}{rcll} v_i & = & \text{sgn}(x_i) & \text{if} \quad |x_i| > 0 \\ v_i & \in & [-1,1] & \text{if} \quad x_i = 0 \end{array} \right\}$$

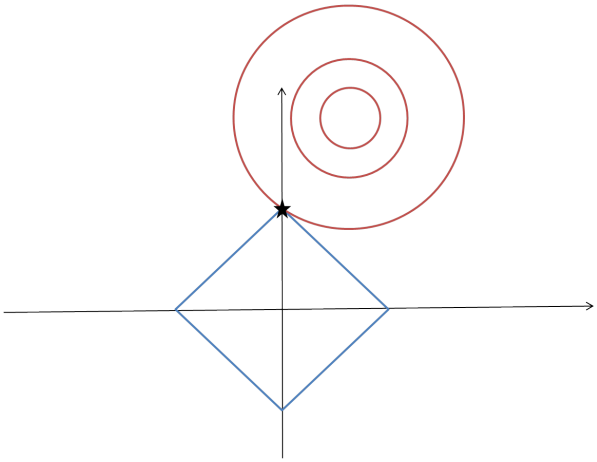


Figure 6.1: illustration of the solution to the LASSO.

where $\text{sgn}(\cdot)$ is the sign function.

Let $A \in \mathbb{R}^{|I_n| \times m}$ be the matrix whose i^{th} row is equal to a_i^T, and let $y = (y_i)_{i \in I_n}^T$ be the vector of response variables.

The optimality conditions for the LASSO problem (6.1) are given by $A^T(Ax - y) + \lambda v = 0$, $v \in \partial \|x\|_1$.

We define as the *active set* (resp. *non active set*), the indices representing non-zero elements (resp. zero elements) of x. The active and non-active sets are referenced by the subscripts a and na respectively.

For example A_a is the matrix representing the columns of A in the active set. The vector x_a represents the non-zero coordinates of x and x_{na} is the 0-vector. The index a_i (resp. na_i) references the i^{th} coordinate of the active (resp. non active set). Since $v \in \partial \|x\|_1$, the i^{th} coordinate of v_a is $v_{a_i} = \text{sgn}(x_{a_i})$, and the i^{th} coordinate of v_{na} is $v_{na_i} \in [-1, 1]$. If the solution is unique, it can be shown that $A_a^T A_a$ is invertible, and we can rewrite the optimality conditions as

$$\begin{aligned} x_a &= (A_a^T A_a)^{-1}(A_a^T y - \mu_n v_a) \\ -\mu_n v_{na} &= A_{na}^T (A_a x_a - y) \end{aligned}$$

Note that if we know the active set and the signs of the coefficients of the solution, thus the vector v_a, we can compute it in closed form. When observations come sequentially, it is possible to solve the LASSO problem recursively using a homotopy algorithm [56]. We present the details of this algorithm in the next section.

6.2 Recursive LASSO

The homotopy-based algorithm which we use to solve the LASSO recursively as new measurements become available considers the following problem:

$$x(t,\mu) = \arg\min_{x\in\mathbb{R}^m} \frac{1}{2}\left\| \begin{pmatrix} A \\ ta_{n+1}^T \end{pmatrix} x - \begin{pmatrix} y \\ ty_{n+1} \end{pmatrix} \right\|_2^2 + \mu \left\| x \right\|_1 .$$

Adding (resp. removing) a point is equivalent to computing the path from the solution at $t = 0$ to the solution at $t = 1$ (resp. from $t = 1$ to $t = 0$). Varying the regularization parameter is equivalent to computing the path from the solution at $\mu = \mu_n$ to the solution at $\mu = \mu_{n+1}$, in other words, we have $x^n = x(0, \mu_n)$ and $x^{n+1} = x(1, \mu_{n+1})$ where x^n and x^{n+1} represent the solution to the LASSO when the n^{th} and $(n+1)^{\text{th}}$ batches of measurements become available, respectively.

The homotopy-based algorithm computes the path from x^n to x^{n+1} in two steps:

Step 1: Vary the regularization parameter from μ_n to μ_{n+1} with t = 0. It is shown that the regularization path between μ_n and μ_{n+1} is piecewise linear [110]. Since in the application presented in this work we have taken the regularization parameter to be constant, we do not review this step here and we refer the reader to [110] for more details.

Step 2: Vary the parameter t from 0 to 1 with $\mu = \mu_{n+1}$. In what follows we explain how this step can be implemented.

It has been proven [56] that $x(t, \mu)$ is a piecewise smooth function of t and the active set and the signs of the coefficients changes only at a finite set of transition points. We know that if the active set and the signs of the parameters are known, the solution to the LASSO can be calculated analytically. Therefore if one can calculate the transition points and changes in the active set and the parameter signs at these points, the solution at the previous time step $x(0, \mu)$ can be used to obtain the new solution $x(1, \mu)$. We now explain how the transition points can be calculated:

Defining the notation $w(t)^T = (v_1^T, w_2(t)^T) \in \partial \parallel x(t) \parallel_1$, a transition point is reached when either an entry of $x_1(t)$ becomes zero or one of the entries of $w_2(t)$ reaches one in absolute value.

We first calculate the points at which an entry of $x_1(t)$ becomes zero. Let $\tilde{A} = \begin{pmatrix} A \\ a_{n+1}^T \end{pmatrix}$ and $\tilde{y} = \begin{pmatrix} y \\ y_{n+1}^T \end{pmatrix}$. Partitioning $\tilde{A} = \begin{pmatrix} \tilde{A}_1 \tilde{A}_2 \end{pmatrix}$ and $x_{n+1}^T = (x_{n+1,1}^T, x_{n+1,2}^T)$ according to the active set, we have

$$x_1(t) = \tilde{x}_1 - \frac{(t^2-1)\bar{e}}{1+\alpha(t^2-1)} u \tag{6.3}$$

where

$$\tilde{x}_1 = (\tilde{A}_1^T \tilde{A}_1)^{-1}(\tilde{A}_1^T \tilde{y} - \mu\nu_1) \tag{6.4}$$

$$\bar{e} = a_{n+1,1}^T - \tilde{x}_1 - y_{n+1} \tag{6.5}$$

$$\alpha = a_{n+1,1}^T (\tilde{A}_1^T \tilde{A}_1)^{-1} a_{n+1,1} \tag{6.6}$$

$$u = (\tilde{A}_1^T \tilde{A}_1)^{-1} a_{n+1,1} \tag{6.7}$$

Defining t_{1i} as the value of t such that $x_{1i}(t) = 0$, we have

$$t_{1i} = \left(1 + \left(\frac{\bar{e}u_i}{\tilde{x}_{1i}} - \alpha\right)^{-1}\right)^{\frac{1}{2}} \tag{6.8}$$

We now calculate the points at which an entry of $w_2(t)$ reaches one in absolute value. Denoting the values of t such that $w_{2j}(t) = 1$ and $w_{2j}(t) = -1$ by t_{2j}^+ and t_{2j}^-, respectively, we have

$$t_{2j}^+ = \left(1 + \left(\frac{\bar{e}(x^{(j)} - c_j^T \tilde{A}_1 u)}{-\mu - c_j^T \tilde{e}} - \alpha\right)^{-1}\right)^{\frac{1}{2}} \tag{6.9}$$

$$t_{2j}^- = \left(1 + \left(\frac{\bar{e}(x^{(j)} - c_j^T \tilde{A}_1 u)}{-\mu - c_j^T \tilde{e}} - \alpha\right)^{-1}\right)^{\frac{1}{2}} \tag{6.10}$$

$$\tag{6.11}$$

Therefore, the transition point t^* will be equal to $\min \min_i t_{1i}, \min_j t_{2j}^+, \min_j t_{2j}^-$.

Putting everything together, the homotopy algorithm can be summarized as below

---

Algorithm 1: Homotopy algorithm for recursive LASSO

1: Compute the path from $x(0, \mu_n)$ to $x(0, \mu_{n+1})$.
2: Initialize the active set as the non-zero entries of $x(0, \mu_{n+1})$ and let v_1 and $a_{n+1,1}$ be the subvectors of $v = \text{sign}(x(0, \mu_{n+1}))$ and a_{n+1} corresponding to the active set and $\tilde{A}_1$ the submatrix of $\tilde{A}$ whose columns correspond to the active set. Initialize $t^* = 0$ and $\tilde{x}_1$ using equation 6.4.
3: Calculate the next transition point t^{**}.
while $t^{**} < t^*$ or $t^{**} \geq 1$ **do**
  if The entry of $x_1(t^{**})$ corresponding to the i^{th} decision variable becomes zero **then**
    Remove i from the active set and update v accordingly.
  end if

if The entry of $w_2(t^{**})$ corresponding to the j^{th} decision variable reaches one in absolute value **then**

Add j to the active set and set v_j equal to 1 or -1 accordingly.

end if

4: Update v_1, $\tilde{A}_1$, $a_{n+1,1}$ and $\tilde{x}_1$ according to the updated active set.

Set $t^* = t^{**}$ and calculate the next transition point t^{**}.

end while

5: Calculate the final value at $t = 1$, where the non-zero entries of the decision vector are given by $\tilde{x}_1$.

The above recursive algorithm may be initialized with one data point (a, y) for which case the active set will have at most one element and defining $i_0 = \arg\max_i |a^(i)|$ and $v = \text{sign}(ya^{(i_0)})$, we have

$$x^{(1)} = \begin{cases} \frac{1}{(a^{(i_0)})^2}(ya^{(i_0)} - \mu_1 v)e_{i0} & if \qquad |ya^{(i_0)}| > \mu_1 \\ 0 \qquad otherwise \end{cases} \tag{6.12}$$

In the next section, we will apply LASSO to perform water flow estimation in tidal channels. In this application, at each time step, as new set of data become available, the $l_1 -$ norm penalty we consider, consists of the difference between the decision variables and the solution at the previous time step. In other words, the recursive estimation problem has the following form

$$x^{n+1} = \arg\min_{x \in \mathbb{R}^m} \frac{1}{2} \sum_{i=1}^{n} (a_i^T x - y_i)^2 + \mu ||x - x^n||_1 \tag{6.13}$$

where x^n represents the solution to the LASSO before the $(n+1)^{\text{th}}$ batch of data arrived. Given the solution to the LASSO results in a solution which is sparse in the value of the $l_1 -$ norm regularization, the above recursive estimator updates only the most significant parameters as new measurements become available. Note that the above optimization problem can be transformed to the standard LASSO form by a change of variable as $\bar{x} = x - x^n$ and all the machinery discussed in this section can be applied to solve this optimization problem recursively.

6.2.1 Water Flow Estimation in Tidal Channels

In this section, we present an application of the LASSO to estimate water flow (discharge) in tidal channels. More precisely, having measurements of the flow at a given location in a channel with tidal forcing, the goal is to estimate the flow in any desired location in the channel. We first derive a transfer function representation of flow in z domain using the linearized Saint-Venant equations. The derived transfer function corresponds to a *multi-input multi-output* (MIMO) system whose outputs are the available measurements and the

inputs are the flow and stage at the location where an estimate is desired. The estimation problem is then posed as an input estimation problem. After parametrizing the inputs using the dominant tidal modes, we use recursive LASSO to estimate the unknown parameters, i.e. the mode amplitudes, recursively. The l_1-norm penalty that we consider in LASSO is the difference between the decision variables and the optimal solution at the previous time step. LASSO enforces a sparse variation in the estimated parameters and it essentially updates the most dominant modes at every time step.

While more traditional state estimation methods such as Kalman filtering could be used to estimate the flow everywhere throughout the channel, the proposed method is particularly useful when estimates of the flow are desired only at a specific location along the channel.

Transfer Function Representation of Shallow Water Equations

To obtain the *open-loop transfer matrix* of the system, we follow the same method as introduced in [5]. Applying z-transform to equations (3.7) and (3.6) and rearranging terms, we obtain the following ordinary differential equation

$$\frac{d}{dl}\begin{pmatrix} q_z(l) \\ h_z(l) \end{pmatrix} = \mathcal{A}_z(l)\begin{pmatrix} q_z(l) \\ h_z(l) \end{pmatrix} \tag{6.14}$$

with

$$\mathcal{A}_z(l) = \begin{pmatrix} 0 & -T_0(l)(\frac{z-1}{\Delta tz}) \\ \dfrac{-(\frac{z-1}{\Delta tz}) + \beta_0(l)}{T_0(l)(C_0(l)^2 - V_0(l)^2)} & \dfrac{2V_0(l)T_0(l)(\frac{z-1}{\Delta tz}) + \gamma_0(l)}{T_0(l)(C_0(l)^2 - V_0(l)^2)} \end{pmatrix} \tag{6.15}$$

Defining $\zeta(l) = (q_z(l), h_z(l))^T$, the differential equation (6.14) has a solution of the form

$$\zeta(l) = \Gamma_z(l, 0)\zeta_0 \tag{6.16}$$

For the case of uniform flow, $\mathcal{A}_z$ does not depend on l and consequently the solution to the differential equation can be calculated analytically and we will have

$$\Gamma_z(l, 0) = e^{\mathcal{A}_z l} \tag{6.17}$$

To solve the differential equation for a general case, the method introduced in [5] can be used. Using this method, the interval $[0, l]$ is divided to smaller intervals $0 = l_0 < l_1 < \cdots < l_n = l, \quad l_{k+1} = l_k + L_k$ over which the flow can be approximated by uniform flow and after solving the differential equation over the small intervals, the overall transfer matrix is obtained by multiplying the individual transfer matrices, i.e. we can write

$$\Gamma_z(l, 0) = \prod_{k=1}^{n} e^{\mathcal{A}_z l_k} \tag{6.18}$$

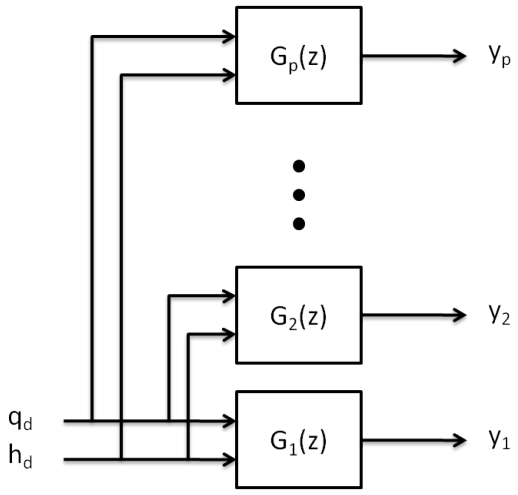


Figure 6.2: The block diagram of system transfer functions.

Approximating the matrix exponentials with the first m terms, we have

$$e^{\mathcal{A}_z l_k} = I + (\mathcal{A}_z l_k) + \frac{1}{2!}(\mathcal{A}_z l_k)^2 + \cdots + \frac{1}{m!}(\mathcal{A}_z l_k)^m \tag{6.19}$$

This will result in a transfer matrix $\Gamma_z(l, 0)$ whose entries are polynomials of degree nm in z. This transfer matrix relates the upstream discharge and stage with the discharge and stage at any location along the channel.

With the location at which estimates of discharge is desired and the locations of the available measurements fixed, we can carry out the same procedure as above to obtain transfer matrices between the desired discharge, q_d, and each measurement, y_i. Denoting the transfer matrix corresponding to each y_i by $G_i(z) = \begin{bmatrix} G_i^q(z) & G_i^h(z) \end{bmatrix}$, we can construct a block diagram as shown in Figure 6.2.

Estimation set-up

In channels with tidal forcing, the flow and stage can be considered as the superposition of the dominant modes of the tides and accordingly q_d and h_d can be parametrized as follows

$$q_d(k) = a_0 + \sum_{i=1}^{N_{\text{modes}}} a_i \cos(w_i k) + b_i \sin(w_i k) \tag{6.20}$$

$$h_d(k) = c_0 + \sum_{i=1}^{N_{\text{modes}}} c_i \cos(w_i k) + d_i \sin(w_i k) \tag{6.21}$$

where N_{modes} is the number of dominant modes considered.

With the above parametrization of Q_d and H_d, the estimation problem boils down to estimation of the coefficients a_i, b_i, c_i, d_i for $i = 1, \cdots, N_{\text{modes}}$.

Let us define

$$u(k) = (q_d(k), h_d(k))^T \tag{6.22}$$

$$x_q = (a_0, a_1, \cdots, a_{\text{Nmodes}}, b_1, \cdots, b_{\text{Nmodes}})^T \tag{6.23}$$

$$x_h = (c_0, c_1, \cdots, c_{\text{Nmodes}}, d_1, \cdots, d_{\text{Nmodes}})^T \tag{6.24}$$

$$C(k) = (1, \cos(w_1 k), \cdots, \cos(w_{\text{Nmodes}} k), \sin(w_1 k), \cdots, \sin(w_{\text{Nmodes}} k)) \tag{6.25}$$

and let $y(k) = (y_1(k), \cdots, y_p(k))^T$ be the vector of deviation of p available measurements from their corresponding steady state at time step k. We can now write

$$\begin{aligned} y_i(k) =& q_d * g_i^q + h_d * g_i^h + e(k) \\ =& \sum_{j=1}^{mn} C(k-j) x_q g_i^q(j) + C(k-j) x_h g_i^h(j) + e(k) \end{aligned} \tag{6.26}$$

where $\{g_i^q(j)\}_{j=1}^{mn}$ and $\{g_i^h(j)\}_{j=1}^{mn}$ are the impulse responses of $G_i^q(z)$ and $G_i^h(z)$, respectively, and $e(k)$ represents the error and $*$ represents convolution.

We can write equation (6.26) in compact form as follows

$$y_i(k) = A_i(k)x + e(k) \tag{6.27}$$

where

$$A_i(k) = \sum_{j=1}^{mn} \begin{bmatrix} C(k-j) g_i^q(j) & C(k-j) g_i^h(j) \end{bmatrix} \tag{6.28}$$

We formulate the estimation problem as the following optimization problem

$$\hat{x}^K = \arg\min_x \parallel Ax - y \parallel_2^2 + \mu_K \parallel x - \hat{x}^{K-1} \parallel_1 \tag{6.29}$$

where

$$A = [A_1(1)^T, \cdots, A_p(1)^T, \cdots, A_1(K)^T, \cdots, A_p(K)^T]^T \tag{6.30}$$

$$y = (y_1(1), \cdots, y_p(1), \cdots, y_1(K), \cdots, y_p(K))^T \tag{6.31}$$

The l_1-norm penalty enforces the variations of the estimates to be sparse. In other words, at each time step, the amplitudes corresponding to the more significant modes are updated.

Implementation

We implement the method on a 23.4 km long channel in Sacramento-San Joaquin Delta in northern California which is a complex network of over 1150 km of tidally influenced

channels and sloughs which cover 738,000 acres of land. The Delta is of great significance in the state of California as it is the main source of drinking water for more than 20 million Californians and it is the source of irrigation of most of California's farmland. The channel chosen for the implementation is located on the southern side of Sacramento as shown in Figure 6.3. Figure 6.4 shows the channel cross section at a few different locations along the section used for the experiment. As can be seen in this figure, the free surface width of the channel is about 440 ft. and the average depth of the channel is about 35 ft..

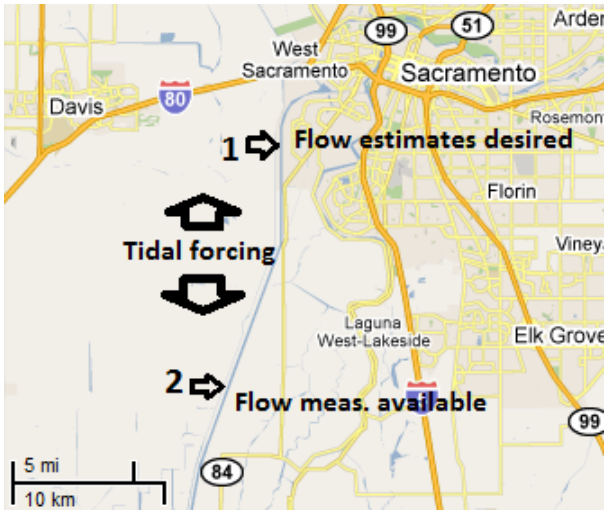


Figure 6.3: The map of the channel used for implementation.

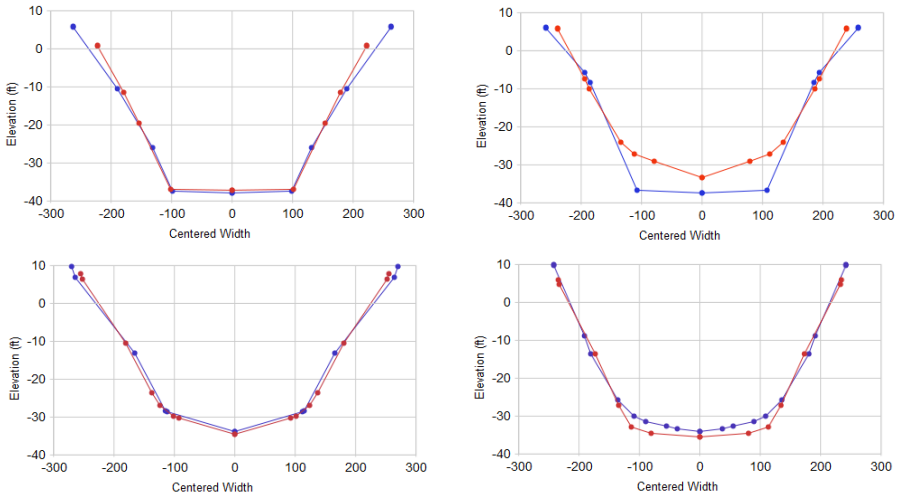


Figure 6.4: The cross-section profile of the channel at eight different locations along the channel.

The *Delta Simulation Model II* (DSM2) is used as the flow model to obtain the measurements of the flow used for performing the estimation and also to evaluate the quality of the estimates. DSM2 is a one-dimensional mathematical model of the flow in Sacramento-San

Joaquin Delta which has been developed in the *California Department of Water Resources* (DWR). DSM2 uses measurements from USGS sensors as boundary conditions and provides discharge and stage at any location within the Delta. More detailed information about DSM2 can be found in [19].

To perform the estimation, we run DSM2 based on historical data starting August 10, 2006 until August 12, 2006. We consider a case in which estimations of the discharge at location 1 is desired when the flow measurements at location 2 are available. To obtain a parametrization of discharge and stage, we perform a spectral analysis of the downstream flow and we use the first eight dominant modes to parametrize the discharge. Figure 6.5 shows the power spectrum of the downstream discharge for the month of July 2006. We perform the estimation for 200 time steps with temporal step size of 15 minutes. Figure 6.6 shows the estimated flow at location 1 and the ground truth, i.e. the flow obtained from DSM2.

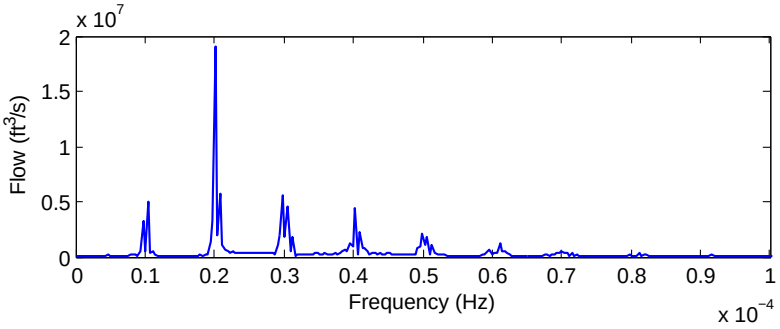


Figure 6.5: The power spectrum of the downstream discharge for the month of August 2006.

Summary

We applied the LASSO to estimate flow in a tidal channel. Given flow measurements at a location in the channel, we used the linearized Saint-Venant equations to obtain the transfer matrix between the flow at any desired location and the measurements. After parametrizing the flow considering the dominant tidal modes, we applied LASSO to estimate the unknown parameters. By considering an L1-norm penalty which is taken as the difference between the decision variables and their estimates obtained at the previous time step, the variations in the estimated parameters are imposed to be sparse, hence, using the available data to only update the most significant parameters at every time step. In cases in which estimates of the flow are desired at a specific location along the channel, this method can be used to perform real-time estimation efficiently.

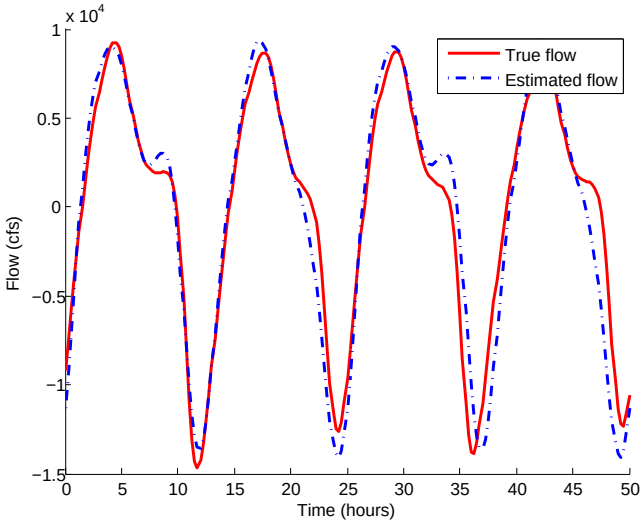


Figure 6.6: The estimated and true flow at location 1.

Chapter 7

Optimal Network Topology Design in Multi-Agent Systems for Efficient Average Consensus

In the previous chapters, we focused on applying and developing data assimilation techniques to incorporate sensor measurements into flow models. In the implementations and experiments presented, two types of sensors, drifters and static sensors, were used. In applications in which a large number of sensors (drifters) are deployed to obtain flow measurements in large networks of open channels, it may be appropriate to implement decentralized communication architecture among the sensors. In such cases, distributed algorithms and consensus-based algorithms must be utilized to fuse the sensor data [10], [15], [33]. In this chapter, we consider a topic in consensus theory which is how to optimally design the communication network topology in multi-agent systems with a decentralized communication architecture to achieve consensus among the agents as efficiently as possible.

Distributed consensus algorithms have received a lot of attention among researchers in the last few years. This is mainly because of their application in various multi-agent systems, including formation control in robotic systems [52], [101] and flocking [92], cooperation in networked multi-agent systems [93], [102], distributed sensor fusion and estimation [10], [15], [33] among other examples. The average-consensus algorithm mainly developed in [14] is the most popular consensus algorithm. Authors in [14] have studied the convergence and performance of the average-consensus algorithm under different conditions. It is shown that the convergence speed of the average-consensus protocol is determined by the second smallest eigenvalue of the *Laplacian matrix* of the *mirror graph* of the network which is itself determined by the topology of the network. A natural question that arises is how to choose the weights on communication links, the entries of the *adjacency matrix* of the

graph, in order to achieve the best performance. In most cases, the network designer can also determine which agents communicate with each other. Therefore, one can go even further and investigate the best communication topology, i.e. the communication graph as well as the weights on the links, so that the performance of the consensus algorithm is optimized.

Since the development of the average-consensus algorithm, extensive efforts have been made to improve the performance of the algorithm. Authors in [121] consider the problem of fast distributed linear averaging in discrete time. They define the *asymptotic convergence factor* which they maximize by taking the weights on the links as decision variables. They formulate this maximization problem for networks with undirected graphs as a semidefinite program. Authors in [11] show that the convergence speed of the average-consensus protocol may be increased dramatically by adding a few long-range communication links. Authors in [120] use *genetic algorithm* (GA) methods to optimize the long-range link configuration to obtain a small-world network with a faster consensus. In [71] assuming that the weights for a link between two nodes is a function of the distance between the nodes, the authors find the best positional configuration of the nodes in order to maximize the convergence speed of the average-consensus protocol.

In this chapter, we consider the network design problem for fast consensus in a general setting. It is assumed that a uniformly constant time delay exists on all communication links. We introduce two approaches to design an efficient network. In the first approach, with a given number of agents, the goal is to find the network topology such that the *communication cost* of the network is less than a given value and the average-consensus protocol converges as fast as possible in presence of communication time-delays on the links. In the second approach, a minimum performance is required for the protocol and the goal is to find the most efficient communication topology, which is a topology with the lowest communication cost, which fulfills the required performance condition. Here, by communication topology, we mean the configuration of the communication graph as well as the weights on the communication links. We formulate both forms of the problem as a *Mixed Integer Semidefinite Program* (MISDP). The resulted MISDP can be used as a powerful network design tool in other ways as well. For instance, consider a case where a network has already been designed. The method presented here can be used to investigate how much the performance of the network can be improved if a number of communication links are added. Depending on the extent of the improvement, the designer may consider slight increase of the number of communication links. Similarly, as will be seen in examples in the last section of this chapter, in some cases, eliminating some of the communication links does not affect the convergence speed of the protocol. In such cases, the communication cost of the network can be reduced without degrading the performance of the network.

The more general setting of directed graphs is considered. In fact, the case of undirected graphs will be included as a special case. Also, a more interesting case which is the case of networks with undirected graphs but non-symmetric weights is included in this setting. To this end, we first generalize a few theorems that are previously proven for undirected graphs to the case of directed graphs in section 7.2; to our knowledge the extension of these results for undirected graphs have not previously been proven in the literature.

7.1 Background

We consider a network of n agents with underlying communication graph G. Let $G = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ be a weighted directed graph with $\mathcal{V} = \{v_1, \cdots, v_n\}$ the set of n nodes of the graph, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ the set of edges of the graph, and $\mathcal{A} = [a_{ij}]$ the *weighted adjacency matrix* of the graph. If agent i does not communicate with agent j, i.e. $e_{ij} \notin \mathcal{E}$, $a_{ij} = 0$, otherwise a_{ij} is a positive number. An edge of G can be denoted by $e_{ij} = (v_i, v_j)$. The set of indices of the nodes is denoted by $\mathcal{I} = \{1, 2, \cdots, n\}$. The set of *neighbors* of a node v_i is defined as $N_i = \{v_j \in \mathcal{V} | (v_i, v_j) \in \mathcal{E}\}$.

Let $x_i \in \mathbb{R}$ be the state of node v_i which might represent a physical quantity, e.g. position, velocity or the heading angle of agents. A *network* is defined as $G_x = (G, x)$ with $x = [x_1, \cdots, x_n]^T$ where G is called the *topology* of the network and x is called the state (value) of the network. The nodes v_i and v_j are said to *agree* if $x_i = x_j$. We say that a *consensus* has been reached among the nodes of a network if $x_i = x_j$ for all $i, j \in \mathcal{I}, i \neq j$ in which case the common value of all nodes is called the *group decision value*.

A *dynamic network* is a dynamical system with state (G, x) where the value of x evolves in time according to the network dynamics $\dot{x} = F(x, u) = [f(x_1, u_1), \cdots, f(x_1, u_1)]^T$. The *average-consensus problem* is the problem of calculating $\frac{1}{n}\sum_{i=1}^{n} x_i(0)$ in a distributed way, meaning that the input of each node u_i only depends on the states of the node and its neighbors. The state feedback $u_i = k_i(x_{j_1}, \cdots, x_{j_{m_i}})$ is called a *protocol* if we have $\{v_{j_1}, \cdots, v_{j_{m_i}}\} \subseteq \{v_i\} \cup N_i$. A protocol is said to asymptotically solve the average-consensus problem if $\frac{1}{n}\sum_{i=1}^{n} x_i(0)\mathbf{1}$ with $\mathbf{1}$ a 1-by-n vector of ones is an asymptotically stable equilibrium of $\dot{x} = F(x, k(x))$.

For a weighted digraph, the in-degree and the out-degree of node v_i is defined as follows:

$$\deg_{\text{in}}(v_i) = \sum_{j=1}^{n} a_{ji}, \qquad \deg_{\text{out}}(v_i) = \sum_{j=1}^{n} a_{ij} \tag{7.1}$$

The *degree matrix* of a graph is defined as $\Delta = \text{diag}(\mathcal{A}\mathbf{1})$ or equivalently as a diagonal matrix with the i^{th} diagonal entry being equal to the out-degree of the i^{th} node v_i.

A few definitions and previously-proven theorems that are used in the subsequent sections are stated here. We refer the reader to [14], [12] and [13], for further details.

Definition 1 *A directed graph (digraph) is said to be strongly connected if there exists a path between any two distinct nodes of the graph.*

Definition 2 *A digraph is said to be balanced if the in-degree of each node is equal to its out-degree.*

Consider the following protocol

$$\dot{x}(t) = -Lx(t) \tag{7.2}$$

where L is the *Laplacian* of the graph G which is defined as $L = \Delta - \mathcal{A}$, or equivalently

$$\dot{x}(t) = u(t) \tag{7.3}$$

with

$$u_i(t) = \sum_{v_j \in N_i} a_{ij}(x_j(t) - x_i(t)) \tag{7.4}$$

Theorem 7.1.1 *Let G be a digraph with Laplacian matrix L. Denoting the maximum node out-degree of G by $d_{max}(G)$, all eigenvalues of L lie in the following disk in the complex plane:*

$$\mathcal{D} = \{z \in \mathbb{C} \,|\, |z - d_{max}(G)| \leq d_{max}(G)\} \tag{7.5}$$

Theorem 7.1.2 *In a network with a directed graph, the above protocol globally asymptotically solves a consensus problem if the graph is strongly connected. With this assumption, this protocol globally asymptotically solves the average-consensus problem if and only if the graph is balanced.*

Definition 3 *Let $G = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ be a weighted digraph. The mirror of G denoted by $\hat{G}$ is the underlying undirected graph of G with the adjacency matrix $\hat{\mathcal{A}} = [\hat{a}_{ij}]$ where*

$$\hat{a}_{ij} = \hat{a}_{ji} = \frac{a_{ij} + a_{ji}}{2} \tag{7.6}$$

Theorem 7.1.3 *Let G be a digraph with Laplacian L. Then $L_s = Sym(L) = (L + L^T)/2$ is the Laplacian matrix of the mirror of G, $\hat{G}$, if and only if G is balanced.*

Theorem 7.1.4 *In a network of integrators with a balanced strongly connected digraph, the protocol* (7.2) *solves the average-consensus problem globally asymptotically with a speed equal to $\lambda_2(\hat{G})$, the Fiedler eigenvalue of the mirror graph of G. $\lambda_2(\hat{G})$ is also called the algebraic connectivity of $\hat{G}$.*

Assuming that there is time-delays on communication links, protocol (7.4) changes to

$$u_i(t) = \sum_{v_j \in N_i} a_{ij}(x_j(t - \tau_{ij}) - x_i(t - \tau_{ij})) \tag{7.7}$$

where τ_{ij} is the delay on the communication link between v_i and v_j.

Theorem 7.1.5 *In a network of integrators with a fixed, undirected and connected graph and equal time-delay $\tau > 0$ on all links, protocol (7.7) globally asymptotically solves the average-consensus problem if and only if $\tau < \pi/2\lambda_{max}(L)$, where $\lambda_{max}(L)$ is the maximum eigenvalue of L.*

Theorem 7.1.6 *If a directed graph G is strongly connected, then rank$(L) = n - 1$ where L is the Laplacian of G and n is the number of nodes of the graph. For the case of undirected graphs, this is a necessary and sufficient condition, i.e. an undirected graph G is connected if and only if rank$(L) = n - 1$.*

Definition 4 *The communication cost of a network is defined as the number of its communication links, i.e.*

$$C = \sum_{i,j=1}^{n} sgn(a_{ij}) \tag{7.8}$$

sgn denotes the sign function which takes negative numbers to -1, 0 to 0 and positive numbers to 1.

7.2 Preliminary results

The *in-valency* and *out-valency* of a node in a directed graph is defined as the number of edges ending on the node and starting from the node, respectively. A *weak path* in a directed graph is a sequence of distinct nodes $v_{i_1}, \cdots, v_{i_m}$ such that either (v_{i_k}, v_{i_k+1}) or (v_{i_k+1}, v_{i_k}) belong to the set of edges of the graph for $i = 1, \cdots, m$. A digraph is called *weakly connected* if any two distinct nodes of the graph can be connected by a weak path.

Definition 5 *We call a weighted directed graph geometrically balanced if the graph is balanced and the in-valency of each node of the graph is equal to its out-valency.*

The following theorem states that strongly-connectedness is equivalent to weakly-connectedness for the case of geometrically balanced graphs. Note that it is necessary that the in-valency and out-valency of each node are equal for the following result to hold. This condition is not satisfied for any balanced graph; in balanced graphs the in-degree and out-degree of each node are equal.

Theorem 7.2.1 *Let G be a digraph such that the in-valency of each node is equal to its out-valency. Then G is strongly connected if and only if it is weakly connected.*

Proof 1 *We refer the reader to any book on algebraic graph theory, e.g. [58], for a proof.*

The following theorem generalizes Theorem 7.1.6 to the case of directed graphs.

Theorem 7.2.2 *Let G be a geometrically balanced digraph. Then G is strongly connected if and only if rank$(\hat{L}) = n - 1$ where $\hat{L}$ is the Laplacian of the mirror graph of G and n is the number of nodes of G.*

Proof 2 *This can be proven directly from Theorems 7.1.6 and 7.2.1.*

The following theorem provides a sufficient condition for convergence of protocol (7.7) for the case of directed graphs. This theorem can be considered as a weak extension of Theorem 7.1.5 to the case of networks with directed graphs.

Theorem 7.2.3 *In a network of integrators with a fixed, strongly connected and balanced digraph and equal time-delay $\tau > 0$ on all links, protocol (7.7) globally asymptotically solves the average-consensus problem if $\tau \leq \dfrac{1}{2d_{max}(G)}$ where $d_{max}(G)$ is the maximum degree of the graph G.*

Proof 3 *The first part of the proof follows directly from the proof of Theorem 7.1.5 [14]. Since the time delay on all links are assumed to be equal, we have $\sum_{i=1}^{n} u_i = 0$ which implies that Ave(x) is invariant under protocol (7.7). Therefore, we just need to prove that (7.7) is stable. Since $X(s) = (sI_n + e^{-\tau s}L)^{-1}x(0)$, it suffices to show that all zeros of $Z_\tau(s) = (sI_n + e^{-\tau s}L)$, lie in the left half plan or at the origin for a graph with the prescribed properties.*

First, note that any eigenvector v of $Z_\tau(s)$ is an eigenvector of L and vice versa. Since the graph is assumed to be strongly connected, L has a simple eigenvalue at the origin. In fact, $s = 0$ in the direction of v_0 is a zero of $Z_\tau(s)$ where v_0 is an eigenvector corresponding to the eigenvalue of L at the origin. Let us denote any nonzero eigenvalue of L by λ and its corresponding eigenvector by v. For $s \neq 0$ being a zero of $Z_\tau(s)$ in the direction of v, i.e. $Z_\tau(s)v = 0$, we must have

$$\frac{1}{\lambda} + \frac{e^{-\tau s}}{s} = 0 \tag{7.9}$$

Thus, using the Nyquist stability criterion, protocol (7.7) is stable if the net encirclement of the Nyquist plot of $\Omega(s) = \dfrac{e^{-\tau s}}{s}$ around $-\dfrac{1}{\lambda}$ is zero.

We have

$$\Omega(jv) = \frac{e^{-jv\tau}}{jv} = -\frac{\sin(v\tau)}{v} - j\frac{\cos(v\tau)}{v} \tag{7.10}$$

Note that $Re(\Omega(jv)) \geq -\tau$ meaning that the Nyquist plot of $\Omega(s)$ is entirely on the right side of $-\tau$.

According to Theorem 7.1.1, we know that all eigenvalues of L lie on or inside the disk $\mathcal{D} = \{z = z_r + jz_{im} \in \mathbb{C} \,|\, (z_r - d_{max}(G))^2 + z_{im}^2 \leq d_{max}(G)^2\}$ where $d_{max}(G)$ is the maximum out-degree of the graph G. For any $z = z_r + jz_{im} \in \mathcal{D} \setminus \{0\}$, we have

$$(z_r - d_{max}(G))^2 + z_{im}^2 \leq d_{max}(G)^2 \tag{7.11}$$

$$\iff \quad z_r^2 + z_{im}^2 \leq 2z_r d_{max}(G) \tag{7.12}$$

$$\iff \quad \frac{-z_r}{z_r^2 + z_{im}^2} \leq \frac{-1}{2d_{max}(G)} \tag{7.13}$$

This implies that the map $f(z) = -\dfrac{1}{z}$ transforms $\mathcal{D} \setminus \{0 + j0\}$ into the half space $\mathcal{H} = \{z = z_r + jz_{im} \in \mathbb{C} \,|\, z_r \leq \dfrac{-1}{2d_{max}(G)}\}$. Therefore, $Re\left(-\dfrac{1}{\lambda}\right)$ lies in $\mathcal{H}$ for all nonzero eigenvalues λ of L. As a result, if $\tau \leq \dfrac{1}{2d_{max}(G)}$, the net encirclement of the Nyquist plot of $\Omega(s) = \dfrac{e^{-\tau s}}{s}$ around $-\dfrac{1}{\lambda}$ is zero and protocol (7.7) is stable.

7.3 Optimal network design

7.3.1 Network design for fast consensus

Our goal is to design a communication topology for a network with a given number of nodes such that the convergence speed of protocol (7.7) is maximized while the communication cost of the network is below a predefined value $C_{\max}$ and an equal non-zero time-delay which is less than $\tau_{\max}$ exists on all communication links.

According to Theorems 7.1.4, 7.2.2 and 7.2.3, the network design problem can be formulated as the following optimization program:

$$\max_{L\in\mathbb{R}^{n\times n}} \ \lambda_2(\hat{L}) \tag{7.14}$$

$$\text{s.t.} \quad \hat{L} = \frac{L+L^T}{2} \tag{7.15}$$

$$\hat{L} \succeq 0 \tag{7.16}$$

$$l_{ij} \leq 0 \quad \forall i,j \in \mathcal{I}, i \neq j \tag{7.17}$$

$$L\mathbf{1} = 0 \tag{7.18}$$

$$\mathbf{1}^T L = 0 \tag{7.19}$$

$$\text{The underlying graph of } L \text{ is strongly connected.} \tag{7.20}$$

$$d_{\max}(G) \leq \frac{1}{2\tau_{\max}} \tag{7.21}$$

$$C = \sum_{i,j=1, i\neq j}^{n} \text{sgn}(-l_{ij}) \leq C_{\max} \tag{7.22}$$

where l_{ij} is the entry of L located at the i^{th} row and j^{th} column of L.

In the above optimization program, equation (7.15) expresses $\hat{L}$ in terms of L, constraints (7.17) and (7.18) ensure that L is a legitimate Laplacian matrix. Constraint (7.19) is imposed to make L correspond to a balanced graph. According to Theorem 7.2.3, constraints (7.19), (7.20) and (7.21) are needed to ensure the convergence of the average-consensus protocol. Finally, constraint (7.22) imposes the bound on the communication cost of the network.

Our goal is to transform the above optimization program to a standard convex optimization program which can be solved efficiently using the available solvers. As will be seen later, we transform the above problem to a *Mixed Integer Semidefinite Program* (MISDP).

We start with the constraints. We shall transform all constraints to affine equalities or *Linear Matrix Inequalities* (LMI). In fact, constraints (7.15), (7.17), (7.18) and (7.19)

are already in the desired form. We deal with constraint (7.20) after constraints (7.21) and (7.22). Constraint (7.21) may be formulated as

$$l_{ii} \leq \frac{1}{2\tau_{\max}} \quad \forall i \in \mathcal{I}. \tag{7.23}$$

In order to transform constraint (7.22), we prove the following theorem

Theorem 7.3.1 *For each $l_{ij} \leq 0, i, j = 1 \cdots, n, i \neq j$ introduce a binary variable $\gamma_{ij} \in \{0, 1\}$. Assuming an arbitrarily small lower bound $M < 0$ on l_{ij} for $i, j = 1 \cdots, n, i \neq j$, the following two sets of expressions are equivalent:*

1.

$$l_{ij} \leq 0 \quad i, j = 1, \cdots, n, i \neq j \tag{7.24}$$

$$\sum_{i,j=1,i\neq j}^{n} sgn(-l_{ij}) \leq C_{max} \tag{7.25}$$

2.

$$\gamma_{ij} \in \{0, 1\} \quad i, j = 1, \cdots, n, i \neq j \tag{7.26}$$

$$l_{ij} \leq 0 \quad i, j = 1, \cdots, n, i \neq j \tag{7.27}$$

$$l_{ij} < 1 - \gamma_{ij} \quad i, j = 1, \cdots, n, i \neq j \tag{7.28}$$

$$l_{ij} \geq \gamma_{ij} M \quad i, j = 1, \cdots, n, i \neq j \tag{7.29}$$

$$\sum_{i,j=1,i\neq j}^{n} \gamma_{ij} \leq C_{max} \tag{7.30}$$

Proof 4 *We first show that equations (7.26), (7.27), (7.28) and (7.29) state that $l_{ij} = 0$ if and only if $\gamma_{ij} = 0$ for $i, j = 1, \cdots, n, i \neq j$. Suppose $l_{ij} = 0$, then equation (7.29) implies that $\gamma_{ij} = 0$, and all other inequalities remain valid. If $\gamma_{ij} = 0$, then equations (7.27) and (7.29) imply that $l_{ij} = 0$ and other inequalities are valid. Equivalently, we have $l_{ij} < 0 \Leftrightarrow \gamma_{ij} = 1$ for $i, j = 1, \cdots, n, i \neq j$ and this completes the proof.*

If we restrict the graph to strongly balanced digraphs, we can write constraint (7.20) in the desired form. We use the boolean variables defined in the above theorem to enforce the condition that the in-valency and out-valency of all nodes of the underlying graph of L are equal. This can be written as

$$\Gamma \mathbf{1} = \Gamma^T \mathbf{1} \tag{7.31}$$

where $\Gamma \in \{0,1\}^{n \times n}$ is a binary matrix with $\Gamma_{ij} = \gamma_{ij}$ for $i, j = 1, \cdots, n, i \neq j$ and $\Gamma_{ii} = 0$ for $i = 1, \cdots, n$.

By using Theorem 7.2.2, we can replace constraint (7.20) by the following condition

$$\text{rank}(\hat{L}) = n - 1 \tag{7.32}$$

Since $\hat{L}$ is a positive semidefinite matrix with one eigenvalue equal to zero, we have the following equivalence [63]:

$$\text{rank}(\hat{L}) = n - 1 \iff \lambda_2(\hat{L}) > 0 \tag{7.33}$$

Consequently, constraints (7.20) and (7.16) can be replaced by conditions (7.31) and (7.33).

Performing all transformations so far, the optimization program (7.14) can be written as the following mixed integer program:

$$\max_{L \in \mathbb{R}^{n \times n}, \Gamma \in \{0,1\}^{n \times n}} \lambda_2(\hat{L}) \tag{7.34}$$

$$\text{s.t.} \quad \hat{L} = \frac{L + L^T}{2} \tag{7.35}$$

$$l_{ij} \leq 0 \quad \forall i, j \in \mathcal{I}, i \neq j \tag{7.36}$$

$$L\mathbf{1} = 0 \tag{7.37}$$

$$\mathbf{1}^T L = 0 \tag{7.38}$$

$$l_{ii} \leq \frac{1}{2\tau_{\max}} \quad \forall i \in \mathcal{I} \tag{7.39}$$

$$\gamma_{ij} \in \{0,1\} \; i, j = 1, \cdots, n, i \neq j \tag{7.40}$$

$$\gamma_{ii} = 0 \; i = 1, \cdots, n \tag{7.41}$$

$$l_{ij} < 1 - \gamma_{ij} \; i, j = 1, \cdots, n, i \neq j \tag{7.42}$$

$$l_{ij} \geq \gamma_{ij} M \; i, j = 1, \cdots, n, i \neq j \tag{7.43}$$

$$\sum_{i,j=1, i \neq j}^{n} \gamma_{ij} \leq C_{\max} \tag{7.44}$$

$$\lambda_2(\hat{L}) > 0 \tag{7.45}$$

$$\Gamma \mathbf{1} = \Gamma^T \mathbf{1} \tag{7.46}$$

We are now left with the cost function and constraint (7.45) which need to be transformed to LMIs.

For the second smallest eigenvalue of the Laplacian of a graph, we can write [58]

$$\lambda_2(\hat{L}) = \min_{z \neq 0} \frac{z^T \hat{L} z}{\|z\|^2} \quad : \mathbf{1}^T z = 0 \tag{7.47}$$

By defining $W = zz^T$, we can write

$$\lambda_2(\hat{L}) = \min_{W \in S^n} \langle W, \hat{L} \rangle \tag{7.48}$$

$$s.t. \quad W \succeq 0 \tag{7.49}$$

$$\mathbf{Tr} W = 1 \tag{7.50}$$

$$W\mathbf{1} = [0]_{n \times 1} \tag{7.51}$$

$$\text{rank}(W) = 1 \tag{7.52}$$

where the inner product between two matrices, W and $\hat{L}$ is defined as $\langle W, \hat{L} \rangle = \text{Tr}(\hat{L}W)$. We first relax the rank constraint and form the dual. Then we show that the rank relaxation is exact. After eliminating the rank constraint, i.e. constraint (7.52), the dual problem can be written as follows

$$\begin{aligned} \mathcal{L}(W, V, \nu_1, \nu_2) =& \langle W, \hat{L} \rangle - \langle W, V \rangle \\ &+ \nu_1(1 - \mathbf{Tr}\, W) + \nu_2^T W \mathbf{1} \end{aligned} \tag{7.53}$$

where $V \in S_+^n, \nu_1 \in \mathbb{R}, \nu_2 \in \mathbb{R}^n$.

$$g(V, \nu_1, \nu_2) = \min_{W \in S^n} \langle W, \hat{L} - V - \nu_1 I + \nu_2 \mathbf{1}^T \rangle + \nu_1 \tag{7.54}$$

$$= \begin{cases} \nu_1 & if \quad \hat{L} - V - \nu_1 I + \nu_2 \mathbf{1}^T = 0 \\ -\infty & otherwise \end{cases} \tag{7.55}$$

Therefore,

$$d^* = \max_{V \in S_+^n, \nu_1 \in \mathbb{R}, \nu_2 \in \mathbb{R}^n} g(V, \nu_1, \nu_2) \tag{7.56}$$

$$= \max_{\nu_1 \in \mathbb{R}, \nu_2 \in \mathbb{R}^n} \quad \nu_1 \tag{7.57}$$

$$s.t. \quad (\nu_2)_1 = \cdots = (\nu_2)_n = \alpha \tag{7.58}$$

$$\nu_2 \mathbf{1}^T - \nu_1 I + \hat{L} \succeq 0 \tag{7.59}$$

$$= \max_{\nu_1, \alpha \in \mathbb{R}} \quad \nu_1 \tag{7.60}$$

$$s.t. \quad \alpha \mathbf{1}\mathbf{1}^T - \nu_1 I + \hat{L} \succeq 0 \tag{7.61}$$

We now show that the rank relaxation, i.e. relaxation of constraint (7.52), is exact. In other words, eliminating constraint (7.52) does not change the optimal value.

The primal and the dual problems are both strictly feasible. Hence, according to the Slater's theorem, strong duality holds and the primal and dual problems are both attained by some primal-dual triplet (W^*, ν_1^*, α^*). Therefore, the *Karush-Kuhn-Tucker* (KKT) conditions are necessary and sufficient for the optimal solutions [27, 26]. These conditions are as follows

- Primal feasibility: $W^* \succeq 0,\ \mathbf{Tr}\, W^* = 1,\ W^*\mathbf{1} = [0]_{n\times 1}$
- Dual feasibility: $\alpha^*\mathbf{1}\mathbf{1}^T - \nu_1^* I + \hat{L} \succeq 0$
- Complementary slackness: $(\alpha^*\mathbf{1}\mathbf{1}^T - \nu_1^* I + \hat{L})W^* = 0$

Suppose W^* is a primal optimal solution. The last KKT condition proves that $(\alpha^*\mathbf{1}\mathbf{1}^T - \nu_1^* I + \hat{L})W^* = 0$, therefore for any non-zero column w^* of W^* we have: $(\alpha^*\mathbf{1}\mathbf{1}^T - \nu_1^* I + \hat{L})w^* = 0$. After normalizing w^* we have: $w^*(w^*)^T \succeq 0,\ \mathbf{Tr}\, w^*(w^*)^T = 1,\ w^*(w^*)^T\mathbf{1} = [0]_{n\times 1}$. Hence, $w^*(w^*)^T$ is a primal optimal solution whose rank is equal to 1. This proves the exactness of the rank relaxation.

Finally, the optimization program (7.34) can be formulated as the following MISDP

Optimization Program.I:

$$\max_{L\in\mathbb{R}^{n\times n},\Gamma\in\{0,1\}^{n\times n},\nu,\alpha\in\mathbb{R}} \quad \nu \tag{7.62}$$

$$\text{s.t.} \quad \hat{L} = \frac{L + L^T}{2} \tag{7.63}$$

$$l_{ij} \le 0 \quad \forall i,j \in \mathcal{I}, i \ne j \tag{7.64}$$

$$L\mathbf{1} = 0 \tag{7.65}$$

$$\mathbf{1}^T L = 0 \tag{7.66}$$

$$\nu > 0 \tag{7.67}$$

$$\alpha\mathbf{1}\mathbf{1}^T - \nu I + \hat{L} \succeq 0 \tag{7.68}$$

$$l_{ii} \le \frac{1}{2\tau_{\max}} \quad \forall i \in \mathcal{I} \tag{7.69}$$

$$\gamma_{ii} = 0\ i = 1, \cdots, n \tag{7.70}$$

$$l_{ij} < 1 - \gamma_{ij}\ i,j = 1, \cdots, n, i \ne j \tag{7.71}$$

$$l_{ij} \ge \gamma_{ij} M\ i,j = 1, \cdots, n, i \ne j \tag{7.72}$$

$$\Gamma\mathbf{1} = \Gamma^T\mathbf{1} \tag{7.73}$$

$$\mathbf{1}^T\Gamma\mathbf{1} \le C_{\max} \tag{7.74}$$

Note that constraint (7.67) is imposed as a replacement for constraint (7.45).

A few remarks are in order:

Remark 2 *For cases in which some constraints exist on communications between agents due to geometric configuration, hardware capabilities, etc., appropriate constraints may be imposed in the above program. For instance, if agent i may not communicate with agent j because of long distance between them, the constraint $l_{ij} = 0$ can be added to the constraints of the above program to incorporate this condition.*

Remark 3 *If the communication topology of the network is fixed a priori, the weights on the links may be chosen optimally by solving a SDP obtained from Optimization Program I after omitting constraints* (7.70), (7.71), (7.73) *and* (7.88).

Remark 4 *Note that the problem of designing a network with undirected communication graph is a special case of the above problem. However, we can use Theorem 7.1.5 instead of Theorem 7.2.3 to ensure the convergence of protocol* (7.7). *Therefore, we should impose $\lambda_{max}(L) < \frac{\pi}{2\tau_{max}}$. To transform this constraint to a LMI, using the following formulation of the maximum eigenvalue of a matrix*

$$\lambda_{max}(L) = \max_{y \neq 0} \frac{y^T \hat{L} y}{\|y\|^2} \tag{7.75}$$

we have

$$\max_{y \neq 0} \frac{y^T L y}{\|y\|^2} < \frac{\pi}{2\tau_{max}} \Longleftrightarrow \frac{y^T L y}{\|y\|^2} < \frac{\pi}{2\tau_{max}} \; \forall y \neq 0 \tag{7.76}$$

$$\Longleftrightarrow \frac{\pi}{2\tau_{max}} I - L \succ 0 \tag{7.77}$$

The Optimization Program I reduces to the following for the case of networks with undirected graphs

Optimization Program.II:

$$\max_{L\in S^n,\Gamma\in\{0,1\}^{n\times n},\nu,\alpha\in\mathbb{R}} \quad \nu \tag{7.78}$$

$$s.t. \quad l_{ij} \leq 0 \quad \forall i,j \in \mathcal{I}, i<j \tag{7.79}$$

$$L\mathbf{1} = 0 \tag{7.80}$$

$$\nu > 0 \tag{7.81}$$

$$\alpha \mathbf{1}\mathbf{1}^T - \nu I + L \succeq 0 \tag{7.82}$$

$$\frac{\pi}{2\tau_{max}} I - L \succ 0 \tag{7.83}$$

$$\gamma_{ij} = \gamma_{ji}\ i,j = 1,\cdots,n, i<j \tag{7.84}$$

$$\gamma_{ii} = 0\ i = 1,\cdots,n \tag{7.85}$$

$$l_{ij} < 1-\gamma_{ij}\ i,j = 1,\cdots,n, i<j \tag{7.86}$$

$$l_{ij} \geq \gamma_{ij} M\ i,j = 1,\cdots,n, i<j \tag{7.87}$$

$$\frac{1}{2}\mathbf{1}^T\Gamma\mathbf{1} \leq C_{max} \tag{7.88}$$

7.3.2 Network design for low communication cost

In this section, we adapt the previous results to another situation, in which a minimum performance in terms of the convergence speed of the consensus protocol (7.7) is desired and the goal is to design a network with the lowest possible communication cost which fulfills the performance requirement.

We assume that the desired performance is given in terms of the Fiedler eigenvalue of the mirror graph, i.e. a network is desired such that $\lambda_2(\hat{L}) \geq \lambda_2^{\min}$ where $\hat{L}$ is the Laplacian of the mirror graph. Note that this assumption is valid since $\lambda_2(\hat{L})$ determines the speed of convergence of the group disagreement.

According to the results of the previous section, this design problem can be formulated as the following mixed integer semidefinite program:

Optimization Program.III:

$$\min_{L\in\mathbb{R}^{n\times n},\Gamma\in\{0,1\}^{n\times n},\nu,\alpha\in\mathbb{R}} \quad \mathbf{1}^T\Gamma\mathbf{1} \tag{7.89}$$

$$\text{s.t.} \quad \hat{L} = \frac{L+L^T}{2} \tag{7.90}$$

$$l_{ij} \leq 0 \quad \forall i,j \in \mathcal{I}, i \neq j \tag{7.91}$$

$$L\mathbf{1} = 0 \tag{7.92}$$

$$\mathbf{1}^T L = 0 \tag{7.93}$$

$$\nu > \lambda_2^{\min} \tag{7.94}$$

$$\alpha\mathbf{1}\mathbf{1}^T - \nu I + \hat{L} \succeq 0 \tag{7.95}$$

$$l_{ii} \leq \frac{1}{2\tau_{\max}} \quad \forall i \in \mathcal{I} \tag{7.96}$$

$$\gamma_{ii} = 0 \; i = 1, \cdots, n \tag{7.97}$$

$$l_{ij} < 1 - \gamma_{ij} \; i,j = 1, \cdots, n, i \neq j \tag{7.98}$$

$$l_{ij} \geq \gamma_{ij} M \; i,j = 1, \cdots, n, i \neq j \tag{7.99}$$

$$\Gamma\mathbf{1} = \Gamma^T\mathbf{1} \tag{7.100}$$

Note that constraints (7.68) and (7.94) guarantee that $\lambda_2(\hat{L}) > \lambda_2^{\min}$.

7.4 Numerical examples and simulation results

In this section, we provide some numerical examples and simulation results. The optimization programs are solved using YALMIP [76] with SeDuMi [109] as the SDP solver. In all of the following examples, the communication time-delay upper bound is taken as .1 second, i.e. $\tau_{\max} = 0.1$.

Example 1 *In this example, we implement Optimization Program I for two cases, a network of 5 agents with $C_{max} = 14$, and a network of 9 agents with $C_{max} = 20$. Figure 7.1, shows the optimal graphs in both cases. The eigenvalue distribution of the Laplacian matrices of the optimal mirror graphs are shown in Tables 7.1 and 7.2. As can be seen in Table 7.1, for the first network, $n = 5$, all eigenvalues of the Laplacian of the optimal mirror graph, except the smallest one which has to be 0, are equal to 6.2500. Furthermore, by solving Optimization Program I for larger C_{max}, we realize that increasing the maximum allowable communication cost does not change the optimal solution. This means that this communication topology pro-*

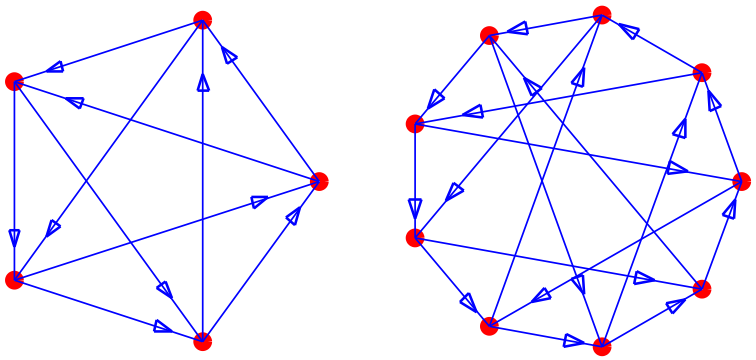


Figure 7.1: Optimal graphs, $n = 5$ (left), $n = 9$ (right), corresponding to Example 1, obtained by solving Program.I.

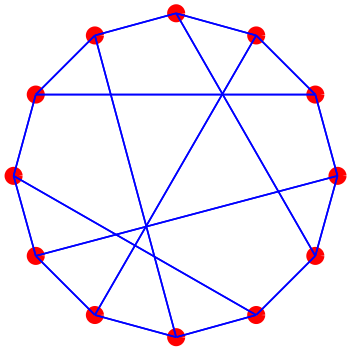


Figure 7.2: Optimal undirected graph, corresponding to Example 2, obtained by solving Program.II.

vides the best possible performance that one could possibly achieve for the average-consensus algorithm in terms of the convergence speed. On the other hand, note that the optimal graph has only 10 links. This means that adding any number of communication links to the network will not have any effect on the convergence speed of the average-consensus protocol.

In the second network, it can be seen in Figure 7.1 that the optimal graph has only 18 links although C_{max} was set to 20. This is due to the fact that we are restricting the feasible set to the case of strongly-balanced graphs for which the in-valency of each node must be equal to its out-valency.

Table 7.1: Eigenvalues of the Laplacian of the optimal mirror graph for the first network in Example 1.

$\lambda_1(\hat{L})$	$\lambda_2(\hat{L})$	$\lambda_3(\hat{L})$	$\lambda_4(\hat{L})$	$\lambda_5(\hat{L})$
0.0000	*6.2500*	*6.2500*	*6.2500*	*6.2500*

Table 7.2: Eigenvalues of the Laplacian of the optimal mirror graph for the second network in Example 1.

$\lambda_1(\hat{L})$	$\lambda_2(\hat{L})$	$\lambda_3(\hat{L})$	$\lambda_4(\hat{L})$	$\lambda_5(\hat{L})$
0.0000	*4.0672*	*4.0672*	*4.0672*	*4.0672*
$\lambda_6(\hat{L})$	$\lambda_7(\hat{L})$	$\lambda_8(\hat{L})$	$\lambda_9(\hat{L})$	
5.6734	*5.6734*	*8.6922*	*8.6922*	

Example 2 *In this example, we implement Optimization Program II for designing an undirected network of 12 agents with* $C_{max} = 18$. *Figure 7.2 and Table 7.3 show the optimal graph and the eigenvalue distribution of the optimal Laplacian matrix, respectively. The optimal graph has 18 links.*

Table 7.3: Eigenvalues of the Laplacian of the optimal graph for the network in Example 2.

$\lambda_1(\hat{L})$	$\lambda_2(\hat{L})$	$\lambda_3(\hat{L})$	$\lambda_4(\hat{L})$	$\lambda_5(\hat{L})$	$\lambda_6(\hat{L})$
0.0000	*4.3669*	*4.3669*	*4.6505*	*4.6505*	*7.4023*
$\lambda_7(\hat{L})$	$\lambda_8(\hat{L})$	$\lambda_9(\hat{L})$	$\lambda_{10}(\hat{L})$	$\lambda_{11}(\hat{L})$	$\lambda_{12}(\hat{L})$
7.4023	*12.6726*	*13.1070*	*13.1070*	*15.7080*	*15.7080*

Example 3 *In this example, we design the cheapest (in the sense of communication cost) network for a network of 7 agents for different values of* λ_2^{min} *which is a user defined parameter that determines the minimum required convergence speed of the protocol. Optimization Program III is solved for all cases. Figure 7.3 shows the optimal graphs for four values of* λ_2^{min}*, 2.5, 4, 4.75, 5. As can be seen in Figure 7.3, the communication cost of the optimal network is 12, 14, 18 and 21 for these four cases, respectively. The eigenvalue distribution of the Laplacian matrices of the optimal mirror graphs for different values of* λ_2^{min} *is shown in Table 7.4. Figure 7.4 shows the communication cost of the optimal network for different values of* λ_2^{min}*. This plot can be very useful in designing network topologies. It can be seen in Figure 7.4 that improving the network topology for a faster consensus is very costly in*

terms of communication cost when $\lambda_2^{min} = 4.5$. *In fact, changing* $\lambda_2^{min} = 4.5$ *to* $\lambda_2^{min} = 5$ *increases the communication cost by 50%. However, the network topology can be improved from* $\lambda_2^{min} = 3$ *to* $\lambda_2^{min} = 4.5$ *without increasing the communication cost of the network.*

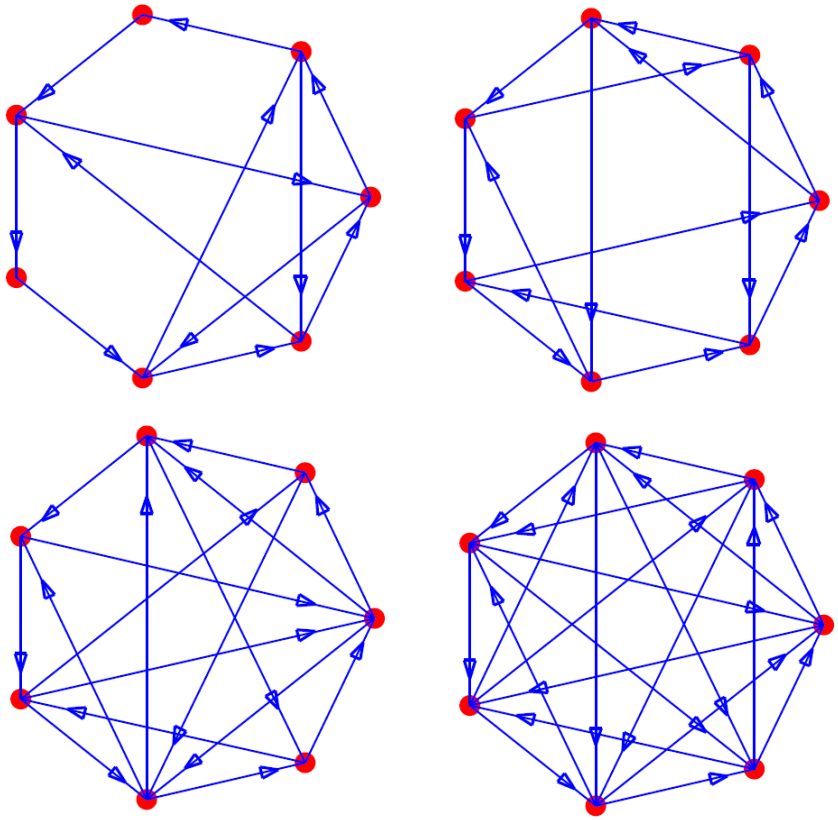


Figure 7.3: $\lambda_2(\hat{L}) \geq 2.5$ (top, left), $\lambda_2(\hat{L}) \geq 4$ (top, right), $\lambda_2(\hat{L}) \geq 4.75$ (bottom, left), $\lambda_2(\hat{L}) \geq 5$ (bottom, right), corresponding to Example 3, obtained by solving Program.III.

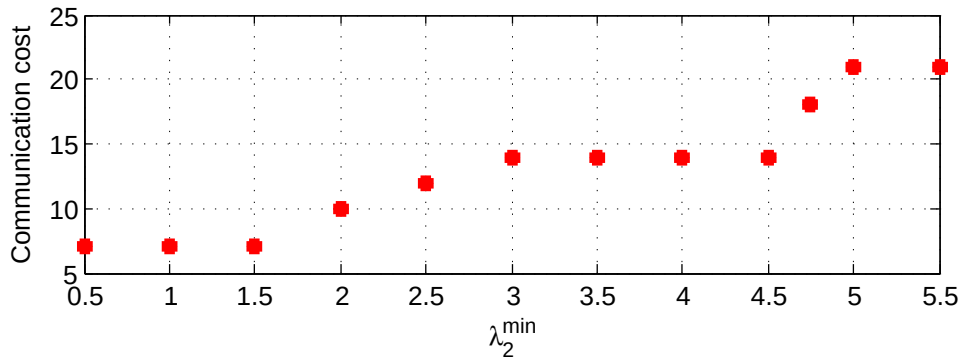


Figure 7.4: Communication cost of the optimal graph for different values of $\lambda_2^{\min}$ for the network in Example 3.

Table 7.4: Eigenvalues of the Laplacian of the optimal mirror graph for the network in Example 3.

$\lambda_2^{\min}$	$\lambda_2(\hat{L})$	$\lambda_3(\hat{L})$	$\lambda_4(\hat{L})$	$\lambda_5(\hat{L})$	$\lambda_6(\hat{L})$	$\lambda_7(\hat{L})$
1	1.1206	1.1206	3.6387	3.6387	5.6580	5.6580
2	2.0000	2.0000	4.9593	6.6657	7.2069	8.9154
3	3.4497	3.5003	4.6388	4.7704	6.5879	6.8518
4	4.0182	4.0299	4.6207	5.7061	7.6199	7.9180
5	5.4412	5.4473	5.4533	5.4579	5.4616	5.4690
5.7	5.7650	5.7658	5.7674	5.7679	5.7684	5.7703

Chapter 8

Conclusions and future work

8.1 Summary

This dissertation presented different methods for assimilation of data into mathematical models of flow in open channels and networks of open channels. First, the Saint-Venant model which is a set of nonlinear partial differential equations was presented. It was shown how these equations can be discretized using explicit discretization schemes and a characteristic-based discretization method to obtain a state-space model of the flow in a network of open channels. Standard state estimation techniques such as the Kalman filter, the Extended Kalman filter and the Unscented Kalman filter, were applied to estimate the state of the system in real time. In a case in which an accurate knowledge of some of the model parameters is not available, it was shown how these state estimation methods can be used to simultaneously estimate the unknown parameters and the state of the system given the available measurements of the flow. It was also shown how Lagrangian sensors, also called drifters, can be used to obtain measurements of the flow. These drifters which are equipped with GPS receivers and communication modules move with the flow and transmit their positions and velocities at every time step. Knowing the bathymetry of the channel at the corresponding cross section, these local flow velocity measurements can be used to calculate the flow or discharge at the corresponding cross section. In comparison to traditional static sensors, drifters have much lower production and maintenance cost. They can be deployed in any area which is of interest and be retrieved at the end. In particular, they can be utilized in cases of emergency response and in places where an appropriate infrastructure is not available. An example of such deployments was presented in an experiment performed to simulate a levee break in an artificial channel located on the eastern border of Carl Blackwell Lake, in Stillwater, Oklahoma.

For data assimilation in large-scale networks of open channels which lead to high dimensional models, more sophisticated methods were used to tractably perform state estimation

in real time. The optimal sampling importance resampling filter which is an instance of particle filters, also known as sequential Monte Carlo, was applied to perform state estimation for such nonlinear and high dimensional models. Also, we presented the application of a recently developed Monte Carlo method, the implicit particle filter, to the flow estimation problem. We investigated the performance of this method in terms of accuracy of estimation results and the computational cost. It was shown that although the computational cost of propagating each particle is higher in the implicit particle filter, better estimation results can be achieved with smaller number of particles. This is because in implicit particle filter, the particles are all chosen such that they belong to a high probability region of the posterior probability density function. Moreover, it was seen that the implicit particle filter could be easily applied in the case of intermittent observations. More precisely, in a practical situation in which measurements of the flow are not available at every time step, the implicit particle filter was applied to incorporate the measurements as they become available to the model. Two heuristic approaches were also studied. In particular, a maximum a posteriori based method was proposed to perform the data assimilation with a lower computational cost.

Another practical case was considered in which estimation of flow variables at a specific location in a channel with tidal forcing was desired while measurements of the flow at other locations were available. The estimation problem was posed as an input estimation problem after obtaining a transfer matrix representation of the Saint-Venant equations. The inputs were parametrized using the dominant tidal modes and the LASSO, least squares with l_1-norm regularization was used to estimate the unknown parameters. A homotopy-based algorithm was used to solve the LASSO recursively to update the most significant unknown parameters, tidal amplitudes, as new measurements would become available.

Finally, we considered the problem of optimal network topology design in multi-agent systems for efficient average consensus. The communication network topology can be represented using algebraic graphs. After generalizing some results proven for undirected graphs to the case of directed graphs, the problem of designing a network with optimal communication graph for efficient average-consensus protocol was considered. We posed the problem in two different ways. One approach was to find the topology which results in the fastest possible average-consensus while the communication cost, i.e. the number of communication links, is less than a given value and the network tolerates communications time delays of smaller than a given bound. In the second approach, under the same conditions, a minimum convergence speed is desired and the design problem is posed as finding the communication topology with the lowest communication cost that provides the desired speed performance. We formulated both design problems as a mixed integer semidefinite program. The problem was solved for the case of directed graphs and the case of undirected graphs is considered as a special class. Moreover, in applications where an undirected communication topology is desired, the convergence speed of the protocol can be improved by considering non-symmetric weights on the communication links, i.e. an undirected graph with a non-symmetric adjacency matrix.

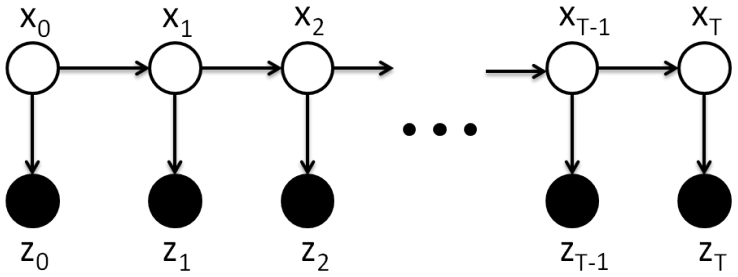


Figure 8.1: Hidden Markov Model.

8.2 Topics of future research

Research can be done in a few different directions to continue the work presented in this dissertation. One direction of interest, would be to use historical measurements of the flow in the network of interest to estimate the covariance matrices of the process and measurement noises. In the rest of this chapter, we explain this can be done using the *expectation maximization* (EM) method.

8.2.1 Noise identification via expectation maximization (EM)

We can consider the state-space model constructed in section 3.1.4, as a *Hidden Markov Model* (HMM), as shown in Figure 8.1. With the Gaussianity assumptions we made on the process and measurement noises, the transition and emission probabilities are Gaussian and we have

$$x_0 \sim \mathcal{N}(\bar{x}_0, P_0) \tag{8.1}$$
$$x_k|x_{k-1} \sim \mathcal{N}(f(x_{k-1}), Q) \tag{8.2}$$
$$z_k|x_k \sim \mathcal{N}(Hx_k, R) \tag{8.3}$$

With these, the hidden Markov model is fully characterized and Q and R are the parameters we are aiming to estimate. The hidden Markov model is a partially observed graphical model where z_k's are the observed variables and x_k's are the latent variables. In such models, the *expectation maximization* (EM) algorithm is the standard method to estimate the unknown parameters, $\theta = Q, R$.

To apply the EM algorithm, the complete log likelihood needs to be calculated:

$$
\begin{aligned}
l_c(D|\theta) =& \log p(z, x|\theta) = \log \left(p(x_0) \prod_{k=0}^{T-1} p(x_{k+1}|x_k, Q) \prod_{k=0}^{T} p(z_k|x_k, R) \right) \\
=& \log p(x_0) + \sum_{k=0}^{T-1} \log p(x_{k+1}|x_k, Q) + \sum_{k=0}^{T} \log p(z_k|x_k, R) \\
=& -\log \left((2\pi)^{\frac{1}{2}(n+m)(T+1)} (\det P_0)^{\frac{1}{2}} (\det Q)^{\frac{T}{2}} (\det R)^{\frac{T+1}{2}} \right) \\
& + (x - \bar{x}_0)^T P_0^{-1} (x - \bar{x}_0) + \sum_{k=0}^{T-1} (x_{k+1} - f(x_k))^T Q^{-1} (x_{k+1} - f(x_k)) \\
& + \sum_{k=0}^{T} (z_k - H x_k)^T R^{-1} (z_k - H x_k)
\end{aligned} \tag{8.4}
$$

Taking derivatives of $l_c(D|\theta)$ with respect to Q and R and setting them equal to zero, we obtain the following estimates for Q and R:

$$
\hat{Q} = \frac{1}{T} \sum_{k=0}^{T-1} (x_{k+1} - f(x_k))(x_{k+1} - f(x_k))^T \tag{8.5}
$$

$$
\hat{R} = \frac{1}{T+1} \sum_{k=0}^{T} (z_k - H x_k)(z_k - H x_k)^T \tag{8.6}
$$

Hence, $\Phi_Q(x_{1:T}) = \sum_{k=0}^{T-1} (x_{k+1} - f(x_k))(x_{k+1} - f(x_k))^T$ and $\Phi_R(x_{1:T}, z_{1:T}) = \sum_{k=0}^{T} (z_k - Hx_k)(z_k - Hx_k)^T$ are the sufficient statistics for Q and R, respectively. In the E step of the EM algorithm, we need to calculate the conditional expectations of the sufficient statistics, i.e. $\mathbb{E}[\Phi_Q(x_{1:T})|z_{0:T}, Q^t]$ and $\mathbb{E}[\Phi_R(x_{1:T}, z_{1:T})|z_{0:T}, R^t]$. We calculate the expected sufficient statistics via Monte Carlo (forward-filtering, backward smoothing), which is explained in the next section:

$$
\mathbb{E}[\Phi_Q(x_{1:T})|z_{0:T}, Q^t] \approx \frac{1}{N_s} \sum_{i=1}^{N_s} \Phi_Q(x_{1:T}^i)
$$

$$
\mathbb{E}[\Phi_R(x_{1:T}, z_{1:T})|z_{0:T}, R^t] \approx \frac{1}{N_s} \sum_{i=1}^{N_s} \Phi_R(x_{1:T}^i, z_{1:T})
$$

where $\{x_{1:T}^i\}$ is the collection of samples obtained from the particle smoother.

The M step consists of the following recursive equations:

$$
\hat{Q}^{t+1} = \frac{1}{T} \Phi_Q^t \tag{8.7}
$$

$$
\hat{R}^{t+1} = \frac{1}{T+1} \Phi_R^t \tag{8.8}
$$

Backward smoothing: For the backward smoothing procedure, a sequential Monte Carlo approximation is presented below [45]

$$p(x_k|z_{1:T}) \approx \sum_{i=1}^{N_s} w_{k|T}^i \delta(x_k - x_k^i) = \sum_{i=1}^{N_s} w_k^i \left[\sum_{j=1}^{N_s} w_{k+1|T}^i \frac{p(x_{k+1}^j|x_k^i)}{\sum_{l=1}^{N_s} w_k^l p(x_{k+1}^j|x_k^l)} \right] \delta(x_k - x_k^i) \quad (8.9)$$

Having obtained the forward weights $\{w_k^i | i = 1, \cdots, N_s, k = 1, \cdots, T\}$, equation (8.9) can be used to calculate the smoothed weights $\left\{ w_{k|T}^i | i = 1, \cdots, N_s, k = 1, \cdots, T \right\}$, recursively.

Bibliography

[1] P. Brasseur and J.C.J. Nihoul. Data assimilation: Tools for modelling the ocean in a global change perspective. *In NATO ASI Series, Series I: Global Environmental Change*, 19(239), 1994.

[2] J.A Cunge, F.M. Holly, and A. Verwey. *Practical aspects of computational river hydraulics*. Pitman, London, UK, 1980.

[3] D.B. Haidvogel and A.R. Robinson . Special issue on data assimilation. *Dyn. Atmos. Oceans*, 13:171–517, 1989.

[4] M. Honnorat, J. Monnier, and F.X.L. Dimet. Lagrangian data assimilation for river hydraulics simulations. In *European Conference on Computational Fluid Dynamics*, Dec 2006.

[5] X. Litrico and V. Fromion. Frequency modeling of open channel flow. *ASCE Journal of Journal of Hydraulic Engineering*, 130(8):806–815, 2004.

[6] X. Litrico and V. Fromion. Boundary control of linearized saint-venant equations oscillating modes. *Automatica*, 42(6):967–972, 2006.

[7] P. Malanotte-Rizzoli. Modern approaches to data assimilation in ocean modeling. *Elsevier Oceanography Series*, 1996.

[8] I.M. Navon. Practical and theoretical aspects of adjoint parameter estimation and identifiability in meteorology and oceanography. *Dyn. Atmos. Oceans*, (27), 1997.

[9] M. Nodet. Variational assimilation of lagrangian data in oceanography. *Inverse Problems*, 22:245–263, 2006.

[10] R. Olfati-Saber. Distributed Kalman filter with embedded consensus filters. In *Proceedings of the 44th IEEE Conference on Decision and Control*, Seville, Spain, Dec 2005.

[11] R. Olfati-Saber. Ultrafast consensus in small-world networks. In *Proceedings of the American Control Conference*, June 2005.

[12] R. Olfati-Saber and R.M. Murray. Consensus protocols for undirected networks of dynamic agents with communication time-delays. *Technical Report CIT-CDS 03 007*, 2002.

[13] R. Olfati-Saber and R.M. Murray. Agreement problems in networks with directed graphs and switching topology. *Technical Report CIT-CDS 03 005*, 2003.

[14] R. Olfati-Saber and R.M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520–1533, 2004.

[15] R. Olfati-Saber and J.S. Shamma. Consensus filters for sensor networks and distributed sensor fusion. In *Proceedings of the 44th IEEE Conference on Decision and Control*, Dec 2005.

[16] O.-P. Tossavainen, J. Percelay, A. Tinka, Q. Wu, and A. Bayen. Ensemble Kalman filter based state estimation in 2d shallow water equations using lagrangian sensing and state augmentation. In *Proceedings of the 46th IEEE Conference on Decision and Control*, Dec 2008.

[17] Q. Wu, M. Rafiee, A. Tinka, I. Strub, and A. Bayen. Inverse estimation of open boundary conditions in channel network via quadratic programming. In *Proceedings of the 48th IEEE Conference on Decision and Control*, Shanghai, China, Dec 2009.

[18] B.D.O. Anderson and J.B. Moore. *Optimal Filtering*. Prentice-Hall, 1979.

[19] J. Anderson and M. Mierzwa. An introduction to the Delta Simulation Model II (DSM2) for simulation of hydrodynamics and water quality of the sacramento- san joaquin delta. *Delta Modeling Section, Office of State Water Project Planning, California Department of Water Resources*, 2002.

[20] K. M. Andreadis, E. A. Clark, D. P. Lettenmaier, and D. E. Alsdorf. Prospects for river discharge and depth estimation through assimilation of swath-altimetry into a raster-based hydrodynamics model. *Geophys. Res. Lett.*, 34, 2007. doi:10.1029/2007GL029721.

[21] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Baysian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[22] R.G. Baraniuk. Compressive sensing. *IEEE Signal Processing Magazine*, 24(4):118, 2007.

[23] N. Bergman. *Recursive Baysian Estimation: Navigation and Tracking applications*. PhD Dissertation, Linkoping University, Linkoping, Sweden, 1999.

[24] S. Biancamaria, M. Durand, K.M. Andreadis, P.D. Bates, A. Booneg, N.M. Mognard, E. Rodriguez, D.E. Alsdorf, D.P. Lettenmaier, and E.A. Clark. Assimilation of virtual wide swath altimetry to improve arctic river modelling. *Remote Sens. of Environ.*, 115(2):373–381, 2010. doi:10.1016/j.rse.2010.09.008.

[25] Gilbert V. Bogle. Stream velocity profiles and longitudinal dispersion. *Journal of Hydraulic Engineering*, 123(9):816–820, 1997.

[26] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*. SIAM studies in applied mathematics: 15, Philadelphia, PA, 1994.

[27] S.P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.

[28] S.L. Britton, G.J. Hanson, and D.M. Temple. A historic look at the USDA-ARS hydraulic engineering research unit. In Glenn O. Brown, Jurgen D. Garbrecht, and Will H. Hager, editors, *Henry P.G. Darcy and other pioneers in hydraulics: contributions in celebration of the 200th birthday of Henry Philibert Gaspard Darcy*, pages 263–276. American Society of Civil Engineers, 2003.

[29] California Department of Water Resources., http://modeling.water.ca.gov/delta/models/dsm2/index.html. *Enhanced Calibration and Validation of DSM2 HYDRO and QUAL*, Nov 2001.

[30] E.J. Candès. Compressive sampling. In *Proceedings of the International Congress of Mathematicians*, volume 3, pages 1433–1452, 2006.

[31] E.J. Candès and Y. Plan. Near-ideal model selection by L1 minimization. *The Annals of Statistics*, 37(5A):2145–2177, 2009.

[32] E.J. Candès and M.B. Wakin. An introduction to compressive sampling. *IEEE Signal Processing Magazine*, 25(2):21–30, 2008.

[33] R. Carli, A. Chiuso, L. Schenato, and S. Zampieri. Distributed Kalman filtering based on consensus strategies. *IEEE Journal on Selected Areas in Communications*, 26(4):622–633, 2008.

[34] J. Carpenter, P. Clifford, and P. Fearnhead. Improved particle filter for nonlinear problems. *Proc. Inst. Elec. Eng., Radar, Sonar, Navig.*, 1999.

[35] W. Castaings, D. Dartus, M. Honnorat, F.X. Le Dimet, Y. Loukili, and J. Monnier. Automatic differentiation: a tool for variational data assimilation and adjoint sensitivity analysis for flood modeling. *Springer Lecture Notes in Computational Science and Engineering*, 50:249–262, 2006.

[36] M. Chalfen and A. Niemiec. Analytical and numerical solution of Saint-Venant equations. *Journal of Hydrology*, 86:1–13, 1986.

[37] M.H. Chaudhry. *Open-Channel Flow*. Springer, 2008.

[38] A.J. Chorin, M. Morzfeld, and X. Tu. Implicit particle filters for data assimilation. *Comm. Appl. Math. Comp.*, 5(2):221–240, 2010.

[39] A.J. Chorin and X. Tu. Implicit sampling for particle filters. *Proceedings of the National Academy of Sciences*, 106(41):17249–17254, 2009.

[40] V. Chow. *Open-channel Hydraulics*. McGraw-Hill Book Company, New York, NY, 1988.

[41] C.M. Dafermos. *Hyperbolic Conservation Laws in Continuum Physics*. Springer-Verlag, 2010.

[42] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57(11):1413–1457, 2004.

[43] P. Del Moral. Measure valued processes and interacting particle systems. application to nonlinear filtering problems. *Ann. Appl. Probab.*, 1998.

[44] Y. Dodge. *Statistical data analysis based on the L1-norm and related methods*. Birkhauser, Basel, Switzerland, 2002.

[45] A. Doucet, N. de Freitas, and editors N. Gordon. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, 2001.

[46] A. Doucet, S. Godsil, and C. Andrieu. On sequential Monte Carlo methods for Baysian filtering. *Statistics and Computing*, 2000.

[47] I. Drori and D.L. Donoho. Solution of L1 Minimization Problems by LARS/Homotopy Methods. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2006*, volume 3, 2006.

[48] M. Durand, K.M. Andreadis, D.E. Alsdorf, D.P. Lettenmaier, D. Moller, and M.D. Wilson. Estimation of bathymetric depth and slope from data assimilation of swath altimetry into a hydrodynamic model. *Geophys. Res. Lett.*, 35, 2008. doi:10.1029/2008GL034150.

[49] B Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407–451, 2004.

[50] G. Evensen. *Data Assimilation: The Ensemble Kalman Filter*. Springer-Verlag, 2007.

[51] S. Fan, L.Y. Oey, and P. Hamilton. Assimilation of drifter and satelite data in a model of the northeastern gulf of mexico. *Continental Self Research*, 24:1001–1013, 2004.

[52] A. Fax and R.M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control*, 49(9):1465–1476, 2004.

[53] M.A.T. Figueiredo and R.D. Nowak. A bound optimization approach to wavelet-based image deconvolution. In *IEEE International Conference on Image Processing, ICIP 2005.*, volume 2, 2005.

[54] M.A.T. Figueiredo, R.D. Nowak, and S.J. Wright. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):586–597, 2008.

[55] J. Friedman, T. Hastie, H. Höfling, and R. Tibshirani. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007.

[56] P.J. Garrigues and L. El Ghaoui. An homotopy algorithm for the Lasso with online observations. In *Neural Information Processing Systems (NIPS)*, volume 21, 2008.

[57] L. Giustarini, P. Matgen, R. Hostache, M. Montanari, D. Plaza, V.R.N. Pauwels, G.J.M De Lannoy, R. De Keyser, L. Pfister, L. Hoffmann, and H.H.G. Savenije. Assimilating sar-derived water level data into a flood model: a case study. *Hydrology and Earth System Sciences*, 15:2349–2365, 2011.

[58] C. Godsil and G. Royle. *Algebraic Graph Theory*, volume 207 of *Graduate Texts in Mathematics.* volume 207 of Graduate Texts in Mathematics. Springer, 2001.

[59] N. J. Gordon, D. J Salmond, A. F. M. Smith, and T. Clapp. Novel approach to nonlinear/non-Gaussian Baysian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107–113, 1993.

[60] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[61] S.S. Haykin. *Kalman filtering and neural networks.* John Wiley & Sons Inc., 2001.

[62] A. Hofleitner, T. Rabbani, M. Rafiee, S. Rosat, L. El Ghaoui, and A. Bayen. Online homotopy algorithm for a generalization of the lasso with applications to online learning. *IEEE Transactions on Automatic Control*, 2012. Under review.

[63] R.A. Horn and C.R. Johnson. *Matrix Analysis.* Cambridge University Press, Cambridge, UK, 1987.

[64] Ilog. *ILOG AMPL CPLEX.* System Version 11.0 User Guide, 2008.

[65] Y. Ishikawa, T. Awaji, K. Akitomo, and B. Qiu. Successive correction of the mean sea surface height by the simultaneous assimilation of drifting buoy and altimetric data. *J. Phys. Oceanogr.*, (26):2381–2397, 1996.

[66] S.J. Julier and J.K. Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *Proc. of AeroSense: The 11th Int. Symp. on Aerospace/Defense Sensing, Simulations and Controls*, 1997.

[67] J. Kaipio and E. Somersalo. *Statistical and computational inverse problems.* Springer-Verlag, New York, NY, 2004.

[68] R.E. Kalman. A new approach to linear filtering and prediction problems1. *Journal of Basic Engineering*, 1960.

[69] E. Kalnay. *Atmospheric Modeling, Data Assimilation and Predictability*. Cambridge University Press, 2003.

[70] S.J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An Interior-Point Method for Large-Scale l1-Regularized Least Squares. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):606–617, 2008.

[71] Y. Kim and M. Mesbahi. On maximizing the second smallest eigenvalue of a state-dependent graph Laplacian. *IEEE Transactions on Automatic Control*, 51:116–120, 2006.

[72] G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian non-linear state space models. *Journal of Comput. Graph. Statist.*, 5(1):1–25, 1996.

[73] L. Kuznetsov, K. Ide, and C.K.R.T. Jones. A method for assimilation of lagrangian data. *Mon. Wea. Rev.*, 131(10):2247–2260, 2003.

[74] F.X. Le Dimet and O. Talagrand. Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects. *Dynamic Meteorology and Oceanography*, 38:97–110, 1986.

[75] H. Lee, A. Battle, R. Raina, and A.Y. Ng. Efficient sparse coding algorithms. *Advances in Neural Information Processing Systems*, 19:801, 2007.

[76] J. Lfberg. Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[77] X. Litrico and V. Fromion. *Modeling and Control of Hydrosystems*. Springer, 2009.

[78] S. Liu and R. Chen. Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93(443):1032–1044, 1998.

[79] L. Ljung. *System identification – theory for the user*. Prentice Hall, 1999.

[80] J. Lund, E. Hanak, W. Fleenor, R. Howitt, J. Mount, and P. Moyle. *Envisioning futures for the Sacramento-San Joaquin Delta*. 2007.

[81] H. Madsen and C. Skotner. Adaptive state updating in real-time river flow forecasting - a combined filtering and error forecasting procedure. *J. Hydrol.*, 308:302–312, 2005.

[82] D.M. Malioutov, M. Cetin, and A.S. Willsky. Homotopy continuation for sparse signal representation. In *Proceedings. (ICASSP '05). IEEE International Conference on Acoustics, Speech, and Signal Processing.*, pages 733–736, Philadelphia, Pennsylvania, USA, 2005.

[83] P. Matgen, M. Montanari, R. Hostache, L. Pfister, L. Hoffmann, D. Plaza, V.R.N. Pauwels, G.J.M. De Lannoy, R. De Keyser, and H.H.G Savenije. Towards the sequential assimilation of sar-derived water stages into hydraulic models using the particle filter: proof of concept. *Hydrol. Earth Syst. Sci.*, 14:1773–1785, 2010.

[84] S. McClurg. *Laypersons guide to the Delta*. 2000.

[85] A. Molcard, L.I. Piterbarg, A. Griffa, T. Ozgokmen, and A. Mariano. Assimilation of drifter observations for the reconstruction of the eulerian circulation field. *J. Geophys. Res.*, 108(C3), 2003.

[86] H. Moradkhani, K.L. Hsu, H. Gupta, and S. Sorooshian. Uncertainty assessment of hydrologic model states and parameters: Sequential data assimilation using the particle filter. *Water Resources Research*, 41, 2005.

[87] M. Morzfeld, X. Tu, E. Atkins, and A.J. Chorin. A random map implementation of implicit filters. *Journal of Computational Physics*, 231:2049–2066, 2012.

[88] B.R. Munson, D.F. Young, and T.H. Okiishi. *Fundamentals of Fluid Mechanics*. John Wiley, 2005.

[89] J.C. Neal, P.M. Atkinson, and C.W. Hutton. Flood inundation model updating using an ensemble kalman filter and spatially distributed measurements. *J. Hydrol.*, 336:401–415, 2007.

[90] J.C. Neal, G. Schumann, P.D. Bates, W. Buytaert, P. Matgen, and F. Pappenberger. A data assimilation approach to discharge estimation from space. *Hydrol. Process.*, 23:3641–3649, 2009.

[91] A.Y. Ng. Feature selection, L1 vs. L2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78. ACM, 2004.

[92] R. Olfati-Saber. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3):401–420, 2006.

[93] R. Olfati-Saber, J.A. Fax, and R.M. Murray. Consensus and cooperation in networked multi-agent systems. In *Proceedings of the 46th IEEE Conference on Decision and Control*, New Orleans, Louisiana USA, 2007.

[94] M.R. Osborne, B. Presnell, and B.A. Turlach. A new approach to variable selection in least squares problems. *IMA journal of numerical analysis*, 20(3):389, 2000.

[95] C. Paniconi, M. Marrocu, M. Putti, and M. Verbunt. Newtonian nudging for a Richards equation-based distributed hydrological model. *Adv. Water. Resour.*, 26(2):161–178, 1996.

[96] M. Rafiee, Q. Wu, and A. Bayen. Kalman filter based estimation of flow states in open channels using lagrangian sensing. In *Proceedings of the 48th IEEE Conference on Decision and Control*, Shanghai, China, Dec 2009.

[97] M. Rafiee, A. Barrau, and A. Bayen. State estimation in networks of open channels using particle filters. In *American Control Conference*, Montreal, QC, June 2012.

[98] M. Rafiee, A. Barrau, and A.M. Bayen. Data assimilation in large scale networks of open channels using monte carlo methods: Sequential importance resampling and implicit particle filters. *Water Resources Research Journal*. Under review.

[99] M. Rafiee and A.M. Bayen. Optimal network topology design in multi-agent systems for efficient average consensus. In *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, GA, Dec 2010.

[100] M. Rafiee, A. Tinka, J. Thai, and A. Bayen. Combined state-parameter estimation for shallow water equations. In *American Control Conference*, San Francisco, CA, June 2011.

[101] W. Ran. Consensus based formation control strategies for multi-vehicle systems. In *American Control Conference*, 2006.

[102] W. Ren, R.W. Beard, and E.M. Atkins. A survey of consensus problems in multi-agent coordination. In *Proceedings of American Control Conference*, 2005.

[103] Report EDF. *TELEMAC 2D. Version 5.2 – Principle note*, 2002.

[104] C. Rozell, D. Johnson, R. Baraniuk, and B. Olshausen. Locally competitive algorithms for sparse approximation. In *IEEE International Conference on Image Processing, ICIP 2007*, volume 4, 2007.

[105] H. Salman, L. Kuznetsov, and C. Jones. A method for assimilating lagrangian data into a shallow-water-equation ocean model. *Monthly Weather Review*, 134:1081–1101, 2006.

[106] I. Strub, J. Percelay, M. Stacey, and A. Bayen. Inverse estimation of open boundary conditions in tidal channels. *Ocean Modelling*, 29(1):85–93, 2009.

[107] I. Strub, J. Percelay, O.-P. Tossavainen, and A. Bayen. Inverse estimation of open boundary conditions in tidal channels. *Networks and Heterogeneous Media*, 4(2):409–430, 2009.

[108] T.W. Strum. *Open Channel Hydraulics*. McGraw-Hill, 2001.

[109] J.F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999. Version 1.05 available from `http://fewcal.kub.nl/sturm`.

[110] R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[111] A. Tinka, M. Rafiee, and A.M. Bayen. Floating sensor networks for river studies. *IEEE Systems Journal*, 2012. To appear.

[112] A. Tinka, I. Strub, Q. Wu, and A. Bayen. Quadratic programming based data assimilation with passive drifting sensors for shallow water flows. In *Proceedings of the 48th IEEE Conference on Decision and Control*, Dec 2009.

[113] Y. Tremolet. Accounting for an imperfect model in 4D-Var. *Q. J. R. Meteorol. Soc.*, 102:2483–2504, 2006.

[114] Y. Tremolet. Model error estimation in 4D-Var. *Q. J. R. Meteorol. Soc.*, 133:1267–1280, 2007.

[115] P. van Leeuwen. Nonlinear data assimilation in geosciences: An extremely efficient particle filter. *Quart. J. Roy. Meteo. Soc.*, 136:1991–1999, 2010.

[116] E.A. Wan and R. van der Merwe. The unscented Kalman filter for nonlinear estimation. In *Proc. of the IEEE Symp. on Adaptive Systems for Signal Processing, Communication and Control (AS-SPCC)*, Lake Louise, Alberta, Canada, 2000.

[117] D. Wang, Y. Chen, and X. Cai. State and parameter estimation of hydrologic models using the constrained ensemble Kalman filter. *Water Resources Research*, 45, 2009.

[118] J. Weare. Particle filtering with path sampling and an application to a bimodal ocean current model. *J. Comp. Phys.*, 228:4312–4331, 2009.

[119] Q. Wu, X. Litrico, and A. Bayen. Open channel flow estimation and data reconciliation using modal decomposition. *Advances in Water Resources*, 32:193–204, 2009.

[120] Z. Wu and R. Wang. The consensus in multi-agent system with speed-optimized network. *International Journal of Modern Physics*, 23(10):2339–2348, 2009.

[121] L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. *Systems & Control Letters*, 53:65–78, 2004.