Estimation of Sea Ice Drift Velocity

Anders Hetland

Master of Science in Engineering Cybernetics
Submission date: June 2013
Supervisor: Lars Imsland, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

NTNU

Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Engineering Cybernetics

PROJECT DESCRIPTION SHEET

Name of the candidate:	Anders Hetland
Thesis title (Norwegian):	
Thesis title (English):	Estimation of sea ice drift velocity

Background

As offshore oil- and gas production enters arctic seas, the presence of ice becomes a substantial challenge in station-keeping operations. An important part of such operations in the Arctic, is a system for doing *ice management*, that is, gather information about, and physically control, the ice environment. One of the most important ice properties are the drift speed of incoming sea ice. In this project, the candidate will model ice by distributed advection equations and develop estimators for drift velocity based on different sensor configurations.

Work description

1) Give a brief overview of ice management challenges, and the role of ice estimation systems.
2) Give a very brief description of modeling and simulation of drift of sea ice.
3) Discretize, implement and simulate a distributed model of ice concentration based on the advection equation.
4) Implement a moving horizon estimation algorithm for estimation of drift velocity, for a 1D ice advection model. Consider drift velocity a parameter to be estimated. Consider different sensor information:
   a) Measure the complete ice concentration
   b) A few (two?) stationary sensors
   c) Mobile sensors ("UAVs")
5) Extend to consider a 2D ice model.
6) Make illustrative simulations, and discuss.

Start date: 14. January, 2013 **Due date:** XX June, 2013

Supervisor: Lars Imsland
Co-advisor(s): Joakim Haugen

Trondheim, __24. January 2013__________

___[illegible]____

Lars Imsland
Supervisor

Abstract

The Arctic is an important region containing large amount of unused potential. It is expected that a large amount of the worlds undiscovered oil and gas resources are located here, and the shortest route between northern Europe and eastern Asia called the Northern Sea Route (NSR) goes through the area. However, the presence of large amount of ice in the Arctic makes it a harsh and difficult environment to operate in.

In order to operate in the areas where ice is present, some sort of ice management system is required. Ice management can be described as the objective to minimize the effect of ice features on floating vessels. Operations including oil and gas exploration and exploitation is complex and require predictability with respect to drifting ice in order to be able to disconnect in time. This has lead to the need of ice intelligence which is able to predict the ice movement.

This study has look upon the problem of estimating ice drift velocity which is a critical parameter in an ice intelligence system. The estimation scheme used in this paper is based on ice concentration measurements and uses the measurements in combination with a numerical model to estimate the ice drift velocity.

The results from the study shows that it is possible to estimate both ice drift velocity and ice concentration based on a limited amount of measurements.

Preface

This is my master's thesis, the final work in my master's degree in Engineering Cybernetics at the Norwegian University of Science and Technology (NTNU). The thesis has been written for the Department of Engineering Cybernetics at NTNU, and was given to me by Professor Lars Imsland.

I hope and believe that my contribution in this study is of interest to the department, and that it will be of interest to future students and researchers in the field.

My motivation for choosing a thesis about the estimation of sea ice drift velocity is two-sided. The main reason is an interest in utilizing mathematical models in order to estimate parameters and states, which are not directly measurable. The other reason is an interest in harsh environments where complex systems are needed in order to operate.

I would like to thank my supervisor Professor Lars Imsland, for giving me useful insight and never giving up on discussions throughout the last semester. I would also like to thank Joakim Haugen, my co-advisor, for discussions and support.

Finally, I would like to thank my family for always supporting me, and my girlfriend Elise for her love, support and encouragements.

Contents

List of Figures

List of Tables

Acronyms

AD	automatic differentiation.
CAS	computer algebra system.
DAE	differential algebraic equation.
DP	dynamic positioning.
EKF	extended Kalman filter.
ERK	explicit Runge-Kutta.
IVP	initial value problem.
KF	Kalman filter.
MHE	moving horizon estimation.
NLP	nonlinear programming.
NSR	Northern Sea Route.
NTNU	the Norwegian University of Science and Technology.
ODE	ordinary differential equation.
PDE	partial differential equation.
QP	quadratic programming.
UAV	unmanned aerial vehicle.
UUV	unmanned underwater vehicle.

1 Introduction

In the recent years it has been discovered that the Arctic region contains large potential for the oil and gas industry, tourism, mining and transportation. As the climate changes, the environment in the Arctic changes and the area which is seasonally ice-free grows (Berkman et al., 2009).

The United States Geological Survey has suggested that about 30% of the worlds undiscovered gas and 13% of the worlds undiscovered oil is located north of the Arctic Circle (Gautier et al., 2009). This has led to increased oil and gas related activities in the Arctic (Borch et al., 2012).

It is a harsh environment in the Arctic with the presence of sea ice, ice ridges and icebergs. This makes the operations more complex (Borch et al., 2012). One example of an operation which becomes more complex in the offshore Arctic environment, is the station-keeping operation, which is a key operation for oil and gas exploration and exploitation. Station-keeping can be performed by mooring or dynamic positioning (DP), where the term DP is used when the vessel maintains its position exclusively by the use of thrusters. Both methods have problems maintaining the position when exposed to sea ice (Hamilton, 2011).

In shipping, the Northern Sea Route (NSR) is expected to become more important for the industry in the near future. It is the the shortest sea route between northern Europe and eastern Asia, and the distance between Yokohama and Hamburg is almost half of the distance compared to the route through the Suez Canal (Johannessen et al., 2007). The route is covered by ice in the winter and it is not completely ice free in the summer either. The ships which uses it is thus able to break ice, or they are escorted by icebreakers.

The Arctic environment is very vulnerable and it has been proposed that oil in Arctic sea water is more serious than in warmer water (Dunbar, 1973). Crude oil spills in the Arctic is complex due to the oil behaviour together with snow and ice with respect to absorption, transportation and spreading (Fingas and Hollebone, 2003). This leads to the demands of even greater focus on safety and regulations (Jensen, 2010).

In the ice infested areas where open water seasons without drift ice are small to non-existent, some sort of ice management is needed in order to perform safe operations (Hamilton et al., 2011). It has been suggested by Eik (2008) that the success of ice management systems are considered to represent the main factors for operating successfully in waters covered with ice.

1.1 Background

Ice management has been defined by Eik (2008):

> *Ice management is the sum of all activities where the objective is to reduce or avoid actions from any kind of ice features. This will include but is not limited to:*
>
> - *Detection, tracking and forecasting of sea ice, ice ridges and icebergs*
> - *Threat evaluation*
> - *Physical ice management such as ice breaking and iceberg towing*
> - *Procedures for disconnection of offshore structures applied in search for or production of hydrocarbons*

In order to perform detection, tracking and forecasting of sea ice, ice ridges and icebergs, some sort of ice intelligence is needed. Ice intelligence is the process of collecting and analysing relevant information about the environment in an area of interest (Haugen et al., 2011). The current and past methods for obtaining the necessary intelligence includes reconnaissance aircraft equipped with radar systems, satellite imagery, shipboard sensors, drift buoys and visual observations (Timco et al., 2005; Eik and Løset, 2009). In addition to these methods, the use of unmanned aerial vehicles (UAVs) and unmanned underwater vehicle (UUV), in order to gather ice information, has been suggested in the literature (Haugen et al., 2011; Eik and Løset, 2009).

Figure 1 shows an ice management system where an UAV is used for collecting ice intelligence. The picture shows how the different icebreakers are located upstream from the operating vessel. It shows the importance of ice forecast, and especially the importance of ice drift velocity and directing. This is crucial information for the ice management system which uses the information in order to determine the position of the ice breakers (Hamilton, 2011).

The sensors currently in use may all be limited by weather, and none of them are able to supply enough ice intelligence on their own (Eik, 2008; Eik and Løset, 2009). It is therefore necessary to combine methods in order to achieve the desired intelligence. This might be both overwhelming and time consuming for a human to do and it has been suggested in the literature to include this in an ice observer system (Haugen et al., 2011).

An ice observer system gathers information of the environment though measurements and process the information in order to analyse it. The measurements can be gathered by any of the previously mentioned methods, and the analysis will be a process utilizing measurements and models. The output is the

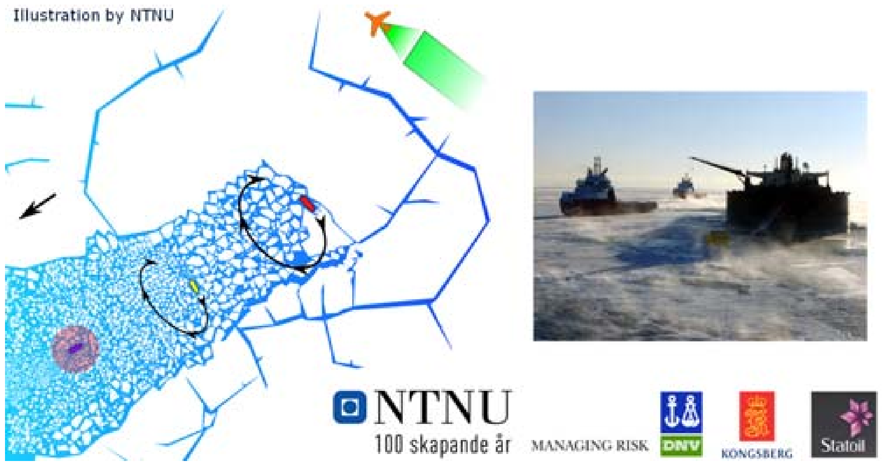


Figure 1: Ice management system for assisting DP operations as pictured by the *KMB Arctic DP* (2013) project.

analysed data which can be estimates of unmeasured information, forecasts, noise free measurements, visualisations, or other products of the processed data (Haugen et al., 2011).

However, there is a need for numerical models combined with estimation techniques in the forecasting of the environment Haugen et al. (2011).

1.2 Aim of the study

The aim of this study is to use a numerical model combined with estimation techniques in order to estimate the drift velocity of sea ice. This will be achieved by using a spatial discretized advection model to describe the drifting ice, and then use the model in cooperation with measurements of ice concentration in order to estimate the velocity.

The spatial discretized model describes ice drifting freely in an area of interest, where ice leaving and entering the area are described through boundary values. The boundary can either be estimated or measured.

The study will focus on the possibility to estimate both unknown states and boundary values together with the velocity estimate. This will include scenarios in both one and two dimensions, where the scenarios in one dimension is well suited to illustrate principles and comparisons.

The estimation schemes used in this study will be moving horizon estimation (MHE) and a variant of the extended Kalman filter (EKF).

1.3 Outline

The next section will cover the method used to model the ice concentration, and includes the necessary discretizations for numerical evaluation. Section 3 will cover how the discretized model can be used to estimate both unmeasured states and unknown parameters. Both hybrid EKF and MHE will be covered as estimation schemes. The tools used to implement the methods will be described in section 4. Section 5 outlines the problem setup including scenarios and tuning of the methods. Results from the simulations will be shown in section 5, and then be discussed in section 6. Conclusions drawn from the discussion will be presented in section 7 together with suggestions for further work.

2 Model description and discretization

This section will look into the description of the mathematical model chosen to describe the sea ice drift. The model will be spatially discretized, and two methods for discretization in time, for use in a nonlinear programming (NLP) formulation, will be presented.

2.1 Sea ice drift modelling

In order to simulate the drift of ice, a mathematical model describing the involved dynamics is necessary. The model will be implemented and simulated on a computer, and hence a model which can be discretized and solved numerically is to be desired. Hibler III (1979) developed a model which most present drift ice dynamic models are based on (Leppäranta, 2011). The model describes both ice thickness and circulation, and is meant to be used for simulation of sea ice over a large area. In the case of ice management, the area of interest is normally small and contains only open water and ice. Which means that the ice will not collide and change shape. Due to this, many of the effects related to ice colliding and hitting obstacles will be neglected and hence the model will be simplified to a continuity equation. Since the timespan of interest is relatively small, the melting and freezing of ice will also be neglected.

2.2 The continuity equation

The mathematical model which will be used to describe the ice concentration is the continuity equation. The equation is a partial differential equation (PDE) describing the transport of conserved quantity, but may also have a source/sink term if the quantity changes. The continuity equation has previously been used by Haugen et al. (2012) to describe the ice thickness, and the formulation in this report will be the same with the exception that ice concentration has been used instead of ice thickness. The source/sink therm will be the freezing and melting of ice respectively.

In order to describe the area of interest, the open set $\Omega \subset \mathbb{R}^2$ with the closure $\partial\Omega \subset \mathbb{R}^2$ has been introduced and represents the area of interest and its boundary respectively. Together they form the closed set $\bar{\Omega} := \Omega \cup \partial\Omega$. The continuity equation describing the ice concentration can then be stated as

$$\frac{\partial c}{\partial t} + \nabla \cdot (uc) = S_c(t, p, c), \tag{2.1}$$

where $c(p,t) \in \mathbb{R}$ is the ice concentration, $t \in \mathbb{R}$ is time, $p = [x,y]^T \in \bar{\Omega}$ is the position, and $u(p,t) : \bar{\Omega} \times \mathbb{R} \to \mathbb{R}^2$ is the velocity flux field. The term $S_c(t,p,c)$ is the source/sink term, and ∇ is defined as

$$\nabla := \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right]. \tag{2.2}$$

Since the model will be used to simulate and estimate ice concentration drifting over a relatively short period of time, the source/sink term can be neglected, $S_c(t,p,c) = 0$. Thus the continuity equation (2.1) can be stated as the following advection equation,

$$\frac{\partial c}{\partial t} + \nabla \cdot (uc) = 0. \tag{2.3}$$

In addition to the simplifications mentioned above, the velocity flux field has been simplified to a uniform velocity. Which mean that $u(t) : \mathbb{R} \to \mathbb{R}^2$ is only dependent of time and not space. Uniform velocity has been assumed since the area of interest is relatively small, which implies that the changes in velocity due to location will be small.

The model is continuous in both space and time. Since the model will be evaluated numerical, a discretization scheme is needed in order to end up with a finite number of states and variables.

2.3 Spatial discretization

Due to the model being continuous in space, a spatial discretization scheme is needed. The drifting ice concentration which is modelled by (2.3) can have a response which looks like a step (e.g ice drifting into open water). This is a difficult response and introduce the need of a discretization scheme which is able to mimic the response.

A third order Kurganov and Tadmor (2000) finite volume discretization method has been chosen. This is a method which avoids the non physical oscillations which can arise around step responses in lower order methods. The model has been discretized as described in Kurganov and Levy (2000) by Joakim Haugen [1]. The model has been spatially discretized in order to become an ordinary differential equation (ODE), where the states are ice concentration in the discretized areas. Equation (2.3) has been discretized with $r_x \in \mathbb{N}$ points in the

[1]Joakim Haugen, Department of Engineering Cybernetics, Norwegian University of Science and Technology, O. S. Bragstads plass 2D, 7491 Trondheim, Norway. joakim.haugen at itk.ntnu.no

x direction and $r_y \in \mathbb{N}$ points in the y direction, and the area of interest becomes $[0, (r_x + 1)\, d_x] \times [0, (r_y + 1)\, d_y]$ where d_x and d_y is the distance between two points in the x and y direction respectively. The ice concentration in a given discretized point $(m, n) \in \mathbb{Z} \times \mathbb{Z}$ will be defined as

$$c_{m,n}(t) \coloneqq c\left([m \cdot d_x, n \cdot d_y]^T, t\right). \tag{2.4}$$

The discretized area of interest will be defined as

$$\bar{\Omega} \coloneqq \{(m, n) \in \mathbb{Z} \times \mathbb{Z} : m \in [0, r_x + 1], n \in [0, r_y + 1]\}, \tag{2.5}$$
$$\Omega \coloneqq \{(m, n) \in \mathbb{Z} \times \mathbb{Z} : m \in [1, r_x], n \in [1, r_y]\}, \tag{2.6}$$
$$\partial\Omega \coloneqq \bar{\Omega} \setminus \Omega. \tag{2.7}$$

The ODE can then be stated as

$$\dot{\bar{c}} = f\left(\bar{c}, c_{\partial\Omega}, u\right) \tag{2.8}$$

where $\bar{c}(t) \in \mathbb{R}^{(r_x r_y)}$ is a vector containing the ice concentration in the discretized points, $u(t) \in \mathbb{R}^2$ is the velocity vector and $c_{\partial\Omega} \in \mathbb{R}^{2r_x+2r_y}$ is the vector containing the ice concentration on the boundary. $f\left(\bar{c}, c_{\partial\Omega}, u\right) : \mathbb{R}^{(r_x r_y)} \times \mathbb{R}^{2r_x+2r_y} \times \mathbb{R}^2 \to \mathbb{R}^{(r_x r_y)}$ is a nonlinear function describing $\dot{\bar{c}}(t)$ (for details, see Kurganov and Levy (2000) and the implementation shown in Appendix A).

The interior states $c_{m,n} : (m, n) \in \Omega$ are sorted in $\bar{c}$ by the following ordering

$$\bar{c} \coloneqq \left[c_{1,1}, \ldots, c_{1,r_y}, c_{2,1}, \ldots, c_{2,r_y}, \ldots, c_{r_x,1}, \ldots, c_{r_x,r_y}\right]^T, \tag{2.9}$$

and the boundary values $c_{m,n} : (m, n) \in \partial\Omega$ are sorted in $c_{\partial\Omega}$ as

$$\begin{aligned} c_{\partial\Omega} \coloneqq [&c_{1,0}, \ldots, c_{r_x,0}, c_{1,r_y+1}, \ldots, c_{r_x,r_y+1}, \\ &c_{0,1}, \ldots, c_{0,r_y}, \ldots, c_{r_x+1,1}, \ldots, c_{r_x+1,r_y}]^T. \end{aligned} \tag{2.10}$$

2.4 Discretization in time

The model (2.8) will be used in a NLP problem for estimation. This requires that the model can be evaluated numerically. Since the model is continuous in time, it will need some sort of discretization to be evaluated numerically. Where direct single shooting, direct multiple shooting and direct collocation are the most common methods for use with optimization. Solving ODEs with an initial condition is called initial value problems (IVPs), and is often solved by integration. The time is discretized in smaller finite elements.

In order to discretize time in finite elements, an interval containing the time of interest will be described as $t \in [t_0, t_f]$, where t_0 is the time at the beginning and t_f is the time at the end. We now wish to partition the interval into a finite number $(N+1) \in \mathbb{N}$. We thus introduce the time step h defined as $h := \frac{t_f - t_s}{N}$. One time element can thus be expressed as $t_i = t_{i-1} + h$ where $i \in \{1, \dots, N\}$ and $t_N = t_f$.

2.4.1 Direct single and multiple shooting

Both direct single shooting and direct multiple shooting relies on the ODE first being discretized in time e.g. by a Runge-Kutta solver, and then numerically solved for the necessary time steps.

Direct single shooting formulates the NLP problem such that the ODE is solved only one time. The formulation uses an initial value for the start, and solves the problem as an IVP in order to get the final value.

In direct multiple shooting, the ODE is solved for every element. That is, a new IVP problem is formulated and solved for every time element, where the initial value is the previous result, and the final value is the value at the next step.

Multiple shooting relies on an ODE solver, and in this report a fourth order explicit Runge-Kutta (ERK) method will be used. The fourth order ERK for (2.8) can be expressed as follows (Egeland and Gravdahl, 2002)

$$
\begin{aligned}
k_1 &= f\left(\bar{c}_n, u\right), && \text{(2.11a)}\\
k_2 &= f\left(\bar{c}_n + h\left(a_{21}k_1\right), c_{\partial\Omega}, u\right), && \text{(2.11b)}\\
k_3 &= f\left(\bar{c}_n + h\left(a_{31}k_1 + a_{32}k_2\right), c_{\partial\Omega}, u\right), && \text{(2.11c)}\\
k_4 &= f\left(\bar{c}_n + h\left(a_{41}k_1 + a_{42}k_2 + a_{43}k_3\right), c_{\partial\Omega}, u\right), && \text{(2.11d)}\\
\bar{c}_{n+1} &= c_n + h\left(b_1k_1 + b_2k_2 + b_3k_3 + b_4k_4\right), && \text{(2.11e)}
\end{aligned}
$$

where h is the length of the time step, $\bar{c}_0$ represent the initial value and $\bar{c}_n$ the discretized value at time $t_n = n \cdot h$. It is assumed that u and $c_{\partial\Omega}$ are constant during one time step h. The constants a_{ij} and b_j are given by Table 2.

The approximated solution to (2.8) can thus be expressed as

$$\bar{c}(t_n) \approx \bar{c}_n, \tag{2.12}$$

where $\bar{c}_0 = \bar{c}(t_0)$ is given as the initial value. We can also write it as

$$\bar{c}(t_n) \approx F\left(\bar{c}_0, c_{\partial\Omega}, u, t_n, t_0\right), \tag{2.13}$$

since $\bar{c}_0, c_{\partial\Omega}$ and u are the only variables needed in order to calculate $\bar{c}(t_n)$. It is assumed that $c_{\partial\Omega}$ and u are constant from t_0 to t_n, and F is a function which

0				
$c_2 = \frac{1}{2}$	$a_{21} = \frac{1}{2}$			
$c_3 = \frac{1}{2}$	$a_{31} = 0$	$a_{32} = \frac{1}{2}$		
$c_4 = 1$	$a_{41} = 0$	$a_{42} = 0$	$a_{43} = 1$	
	$b_1 = \frac{1}{6}$	$b_2 = \frac{2}{6}$	$b_3 = \frac{2}{6}$	$b_4 = \frac{1}{6}$

Table 2: The Butcher array for the fourth order ERK method.

calls (2.11e) n times. The function F is thus an approximation of the integral

$$\bar{c}(t_n) = \int_{t_0}^{t_n} \left(f\left(\bar{c}, c_{\partial\Omega}, u\right)\right) dt. \tag{2.14}$$

Note that the time step h, is usually lower in ERK method than the time step h used in multiple shooting steps.

2.4.2 Direct collocation

Direct collocation utilize polynomials in order to discretize and solve the ODE. The collocation equations become a part of the NLP formulation and is solved through the optimization (Biegler, 2010).

The method relies on approximations made by polynomials. The time is partitioned in N time steps, and one time step h is partitioned in smaller elements τ_j, where $j \in \{0, \ldots, K\}$, as shown in Figure 2. This is an effective approximation where the partitioned time elements indicate the interpolation points of the polynomial, which will be of degree K.

The use of collocation hence introduces a larger system, but with more sparsity and structure. This can be utilized by many large scale NLP solvers and it can therefore be solved very effectively (Biegler, 2010). We introduce the notation $\bar{c}_{i,j}$ where $i \in \{0, \ldots, N\}$ and $j \in \{0, \ldots, K+1\}$. The parameter $\bar{c}_{i,0}$ will be the approximation of $\bar{c}(t_i)$.

The system (2.8) will be approximated with a polynomial $\bar{c}^K(t)$ with degree K in order to approximate the states $\bar{c}(t)$, where Lagrange interpolation will be used to find its coefficients. The finite element is then divided in $K+1$ interpolation points $\tau_{ik}, \quad k \in \{0, \ldots, K\}$ as shown in Figure 2. The polynomial for a finite element i can then be expressed as

$$\left.\begin{array}{r} t = t_{i-1} + h\tau, \\ \bar{c}^K(t) = \sum_{j=0}^{K} \ell_j(\tau)\, \bar{c}_{ij}, \end{array}\right\} t \in [t_{i-1}, t_i], \tau \in [0,1], i \in \{1, N\}, \tag{2.15}$$

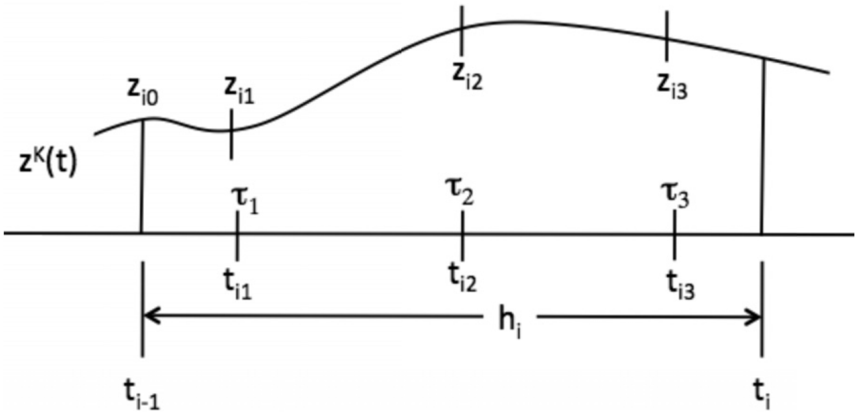


Figure 2: Polynomial approximation for state profile across a finite element i. Courtesy of Biegler (2010)

where the time h represent the time of one finite element i from time t_{i-1} to t_i. The element $\ell_j(\tau)$ is called the cardinal polynomial and given by

$$\ell_j(\tau) = \prod_{k=0,\neq j}^{K} \frac{(\tau - \tau_k)}{(\tau_j - \tau_k)}, \tag{2.16}$$

where $\tau_0 = 0$, $\tau_j < \tau_{j+1}$, $j \in \{0, \dots, K-1\}$. The polynomial $\bar{c}^K$ now has the property that $\bar{c}^K(t_{ij}) = \bar{c}_{ij}$ with $t_{ij} = t_{i-1} + \tau_j h$. Inserting $\bar{c}^K$ in (2.8) and assuming constant parameters gives us the equation

$$\frac{d\bar{c}^K}{dt}(t_{ik}) = f\left(\bar{c}^K(t_{ik}), c_{\partial\Omega}, u\right), \quad k = 1, \dots, K, \tag{2.17}$$

where we get the collocation equation by applying $\frac{d\bar{c}^K}{d\tau} = h\frac{d\bar{c}^K}{dt}$

$$\sum_{j=0}^{K} \bar{c}_{ij} \frac{d\ell_j(\tau_k)}{d\tau} = hf(\bar{c}_{ik}, c_{\partial\Omega}, u), \quad k = 1, \dots, K. \tag{2.18}$$

In order to assure continuity between the finite elements, the following equality restrictions between finite elements are added

$$\begin{aligned} \bar{c}_{i+1,0} &= \sum_{j=0}^{K} \ell_j(1)\,\bar{c}_{ij}, \quad i = 1, \dots, N-1, \\ \bar{c}_f &= \sum_{j=0}^{K} \ell_j(1)\,\bar{c}_{Nj}, \quad \bar{c}_{1,0} = \bar{c}_0, \end{aligned} \tag{2.19}$$

Since our problem will be a MHE problem, the number of finite time steps N will be the same as the horizon length.

The interpolation nodes τ has been chosen from the shifted Gauss-Legendre and Radau roots from Biegler (2010) as shown in Table 3. This is in order to achieve a better interpolation than with equal spacing.

The implicit collocation equations will be solved by the NLP solver as part of the estimation problem.

Table 3: Shifted Gauss-Legendre and Radau roots as collocation points

Degree K	Legendre Roots	Radu Roots
1	0.500000	1.000000
2	0.211325	0.333333
	0.788675	1.000000
3	0.112702	0.155051
	0.500000	0.644949
	0.887298	1.000000
4	0.069432	0.088588
	0.330009	0.409467
	0.669991	0.787659
	0.930568	1.000000
5	0.046910	0.057104
	0.230765	0.276843
	0.500000	0.583590
	0.769235	0.860240
	0.953090	1.000000

3 State and parameter estimation

The goal of this study is to know every parameter and state in (2.8) based on a few measurements. In order to accomplish this, some sort of estimation scheme is needed. The most used estimation scheme for nonlinear systems is probably the EKF (Julier and Uhlmann, 2004). However, there are some drawbacks with the EKF, and an alternative to EKF is the MHE which has been successful in estimating states in cases where EKF fails (Tenny and Rawlings, 2002; Haseltine and Rawlings, 2005).

This section will include a description of the MHE and the hybrid EKF. The focus has been placed on the MHE which is a relatively new method to use compared with the EKF.

The continuous time spatial discretized system (2.8) will be subject to measurements done in discrete time. That is, a measurement y_k at time t_k expressed as

$$y_k = h_k\left(\bar{c}_k\right) + v_k, \tag{3.1}$$

where $v_k \in \mathbb{R}^{n_y}$ is measurement noise, $\bar{c}_k = \bar{c}(t_k)$ and $h_k(\bar{c_k}) : \mathbb{R}^{r_x r_y} \to \mathbb{R}^{n_y}$. The number of measurements is n_y.

3.1 Observability

Observability is a property that says if a system is observable, then it is possible to uniquely determine the unknown initial states from the output and input of the system over the time span $(0, t)$ (Chen, 1999). For linear systems the observability can be determined from the matrix relating states to outputs, and the matrix relating old states to new states. The observability does not include noise and model uncertainties, and can therefore fail due to noise or disturbances.

It is very difficult to find the observability for nonlinear systems, and in this report an approximation will be used. The approximation will be a linearised observability matrix around the current state and parameter estimate. It will be hard to conclude on the observability of the system based on the linearisation, but it will provide a good indication on the observability.

Observability is usually a requirement in order to achieve proper estimates. Due to the linearisation, the system may still be observable even if the linearised system is not observable. Observability may not even be a requirement for good estimates in this case, since all the states is not necessary of interest.

The linearised observability matrix for t_k can be expressed as

$$O_k = \begin{bmatrix} C_k \\ C_k A_k \\ \vdots \\ C_k A_k^{n-1} \end{bmatrix}, \tag{3.2}$$

where

$$\begin{aligned} C_k &= \begin{bmatrix} \frac{\partial h}{\partial \bar{c}}\Big|_{\bar{c}_k, p_k} & \frac{\partial h}{\partial p}\Big|_{\bar{c}_k, p_k} \end{bmatrix}, \\ A_k &= \begin{bmatrix} \frac{\partial f}{\partial \bar{c}}\Big|_{\bar{c}_k, p_k} & \frac{\partial f}{\partial p}\Big|_{\bar{c}, p_k} \\ 0_{n_p \times n_{\bar{c}}} & 0_{n_p \times n_p} \end{bmatrix}. \end{aligned} \tag{3.3}$$

Here, p is a vector containing the parameters which are to be estimated, and n is the dimension of A_k.

3.2 Hybrid extended Kalman filter

In this study, the hybrid EKF will be used for state and parameter estimation. The estimates from the EKF will additionally be used in the arrival cost in the MHE problem. The hybrid EKF is a continuous time EKF with discrete time updates (Simon, 2006).

From (2.8) and (3.1), the system can be written as

$$\dot{\bar{c}} = f\left(\bar{c}, c_{\partial\Omega}, u\right) + w, \tag{3.4a}$$
$$y_k = h_k\left(\bar{c}_k\right) + v_k, \tag{3.4b}$$
$$w\left(t\right) \sim \left(0, Q\right), \tag{3.4c}$$
$$v_k \sim \left(0, R_k\right), \tag{3.4d}$$

where $w\left(t\right) \in \mathbb{R}^{r_x r_y}$ and v_k are white noise with covariance Q and R_k respectively. The time update equation for the hybrid EKF will then be

$$\begin{aligned} \dot{\hat{\bar{c}}} &= f\left(\hat{\bar{c}}, c_{\partial\Omega}, u\right), \\ \dot{P} &= AP + PA^T + LQL^T. \end{aligned} \tag{3.5}$$

The equation propagates the state estimate $\hat{\bar{c}}$ from $\hat{\bar{c}}_{k-1}^{+}$ to $\hat{\bar{c}}_k^{-}$, and P from P_{k-1}^{+} to P_k^{-}. Where A and L is given by:

$$\begin{aligned} A &= \frac{\partial f}{\partial \bar{c}}\bigg|_{\hat{\bar{c}}}, \\ L &= \frac{\partial f}{\partial w}\bigg|_{\hat{\bar{c}}}. \end{aligned} \tag{3.6}$$

The updates which needs to be calculated at each step k is given by

$$\begin{aligned} K_k &= P_k^- H_k^T \left(H_k P_k^- H_k^T + M_k R_k M_k^T\right)^{-1} \\ \hat{\bar{c}}_k^+ &= \hat{\bar{c}}_k^- + K_k \left[y_k - h_k\left(\hat{\bar{c}}_k^-\right)\right] \\ P_k^+ &= \left(I - K_k H_k\right) P_k^- \left(I - K_k H_k\right)^T + K_k M_k R_k M_k^T K_k^T. \end{aligned} \tag{3.7}$$

H_k and M_k are the partial derivatives of (3.4b) with respect to $\bar{c}_k$ and v_k respectively. They are evaluated at $\hat{\bar{c}}_k^-$. The relationship between Q and R_k determine how much the measurements are trusted compared to the model, where high covariance in the process noise compared to the covariance in the measurements will lead to more trust in the measurements.

In order to perform parameter estimation with the hybrid EKF, the parameters need to be augmented into the state variables. This can be achieved by introducing the augmented state variable Z as

$$Z := \begin{bmatrix} \bar{c} \\ p \end{bmatrix}, \tag{3.8}$$

where p is a vector containing the unknown states. The covariance matrix Q will need to be augmented in order to contain covariances for the parameters, and H_k will be the partial derivative of (3.4b) with respect to Z. The augmented version of (3.6) becomes

$$\begin{aligned} A &= \begin{bmatrix} \frac{\partial f}{\partial \bar{c}}\Big|_{\hat{\bar{c}},\hat{p}} & \frac{\partial f}{\partial p}\Big|_{\hat{\bar{c}},\hat{p}} \\ 0_{n_p \times n_{\bar{c}}} & 0_{n_p \times n_p} \end{bmatrix}, \\ L &= \frac{\partial f}{\partial w}\Big|_{\hat{\bar{c}},\hat{p}}. \end{aligned} \tag{3.9}$$

Since the parameters are assumed to be constant, (3.5) need to be extended with

$$\begin{aligned} \dot{c}_{\partial\Omega} &= 0, \\ \dot{u} &= 0. \end{aligned} \tag{3.10}$$

The relationship $h_k\left(\bar{c}_k\right)$ between the measurements and the states will in this case be a linear relationship. With the augmented states, (3.4b) can be written as

$$y_k = H_k Z_k + M_k v_k, \tag{3.11}$$

where $H_k \in \mathbb{R}^{(n_y)\times(r_y r_x + n_p)}$, $M_k \in \mathbb{R}^{n_y \times n_y}$ and $n_p = 2(r_x + r_y) + 2$ in the case of full parameter estimation.

3.3 Moving horizon estimation

Another approach for estimating states and parameter is to use a MHE scheme, where the estimation problem becomes a dynamic optimization problem. This is a method which require more computation speed than EKF and thus is becoming more and more used as new and better computer hardware is introduced. MHE can be seen as an extension of the EKF (Rao and Rawlings, 2002).

The objective is to minimize an estimation error with respect to the model dynamics as a constraint together with minimum and maximum values on the states and parameters.

3.3.1 Problem formulation

In order to formulate the MHE problem, some sort of cost function needs to be formulated. In this case, the estimation error squared will be used as the cost function together with process noise and arrival cost. The quadratic cost makes large deviations more expensive than small, and is well suited for estimation problems.

The state estimation error can be formulated as

$$e_j = y_j - h_j\left(\tilde{\tilde{c}}_j\right), \tag{3.12}$$

where $\tilde{\tilde{c}}_j = \tilde{\tilde{c}}(t_j)$ represent the estimate of the state at time t_j. The quadratic cost function $\mathcal{L}_c\left(e_j\right)$ then becomes

$$\mathcal{L}_c\left(e_j\right) = e_j^T R^{-1} e_j, \tag{3.13}$$

where the index j indicate the time sample in the horizon. The weighting matrix $R \in \mathbb{R}^{(n_y)\times(n_y)}$ is a diagonal covariance matrix of the measurement noise, and n_y is the number of states which is being measured. The process noise cost term can be formulated as

$$\mathcal{L}_w(\tilde{w}_j) = \tilde{w}_j^T Q^{-1} \tilde{w}_j, \tag{3.14}$$

where $\tilde{w}_j = \tilde{w}(t_j)$ is the estimated process noise and Q is a diagonal covariance matrix describing the process noise.

Equation (3.4) can then be expressed as a MHE problem with the horizon length N

$$\min_{\substack{\tilde{\tilde{c}}_{k-N},\dots,\tilde{\tilde{c}}_{k-1},\tilde{\tilde{c}}_k \\ \tilde{p} \\ \tilde{w}_{k-N},\dots,\tilde{w}_{k-1}}} \Gamma\left(\tilde{\tilde{c}}_{k-M}, c_{\partial\Omega}, u\right) + \sum_{j=k-N}^{k} \mathcal{L}_c\left(e_j\right) + \sum_{j=k-N}^{k-1} \mathcal{L}_w(\tilde{w}_j) \tag{3.15}$$

$$\begin{aligned} s.t. \quad & \dot{\tilde{\tilde{c}}} = f\left(\tilde{\tilde{c}}, c_{\partial\Omega}, u\right) + w(t), && t \in [t_{k-N}, t_k] \\ & \tilde{\tilde{c}}_{min} \leq \tilde{\tilde{c}}(t) \leq \tilde{\tilde{c}}_{max} && t \in [t_{k-N}, t_k] \\ & \tilde{p}_{min} \leq \tilde{p} \leq \tilde{p}_{max} \\ & \tilde{w}_{min} \leq \tilde{w}(t) \leq \tilde{w}_{max} && t \in [t_{k-N}, t_k] \end{aligned} \tag{3.16}$$

Where the present time is t_k and p is a vector containing the parameters which need to be estimated. The measurements used are thus $y_{k-N}, \ldots, y_{k-1}, y_k$.

The result is a semi discrete dynamic optimization problem where the cost function is discrete and the constraints are continuous in time. If the cost function had been continuous in time, it would have been an integral.

3.3.2 Arrival cost

In the optimal case, the horizon used in the MHE formulation should be infinitely large in order to cover all the information available. This is not so practical since the problem would get infinitely large and hence require too much computational power. In order to cope with the limited horizon, a term $\Gamma\left(\tilde{\tilde{c}}_{k-M}, c_{\partial\Omega}, u\right)$ is introduced in order to representing the information from t_0 to the start of the estimation horizon $t_l = t_{k-N}$. This term is known as the arrival cost, and will in many cases be calculated with a variant of the Kalman filter (KF) (Qu and Hahn, 2009). Without the arrival cost, horizons containing small to none changes in measurements might fail on estimating states and parameters.

The arrival cost will in this case be the quadratic error between the estimation variables at time t_l and the estimations from the hybrid EKF. We introduce the variable V defined as

$$V_j := \begin{bmatrix} \tilde{\tilde{c}}_j \\ p \end{bmatrix}, \tag{3.17}$$

where V_j indicates the variables at time t_j.

The arrival cost function can then be defined as

$$\Gamma\left(\tilde{\tilde{c}}_{k-M}, c_{\partial\Omega}, u\right) := \left(V_{k-M} - \hat{Z}^+_{k-M}\right)^T \left(P^+_{k-M}\right)^{-1} \left(V_{k-M} - \hat{Z}^+_{k-M}\right), \tag{3.18}$$

where $\hat{Z}^+_{k-M}$ and P^+_{k-M} are given by (3.7).

3.3.3 Dynamic optimization

A dynamic optimization problem is an optimization problem which is subject to dynamic equations as constraints. Hence a differential algebraic equation (DAE) or ODE solver usually need to be embedded within the NLP problem.

The MHE problem formulated in (3.15) and (3.16) needs to be solved as a dynamic optimization problem, and needs some discretization in order to make it a finite element problem.

It has been suggested in the literature to use direct collocation for the discretization of the ODE (Biegler, 2010) in the NLP problem. With the use of direct collocation as described by (2.18) and (2.19), the NLP problem (3.15) and (3.16) becomes

$$\min_{\substack{z \\ \tilde{p} \\ \tilde{w}_{k-N},\dots,\tilde{w}_{k-1}}} \quad \Gamma\left(\tilde{\bar{c}}_{k-M}, c_{\partial\Omega}, u\right) + \sum_{j=k-N}^{k} \mathcal{L}_c\left(e_j\right) + \sum_{j=k-N}^{k-1} \mathcal{L}_w(\tilde{w}_j) \tag{3.19}$$

$$\begin{array}{lll} s.t. & \sum_{j=0}^{K} \dot{\ell}_j(\tau_k)\tilde{\bar{c}}_{i,j} - hf\left(\tilde{\bar{c}}_{ik}, c_{\partial\Omega}, u\right) = 0 & i = k-N,\dots,k-1,k \\ & z_{min} \leq z \leq z_{max} & \\ & p_{min} \leq p \leq p_{max} & \\ & \tilde{\bar{c}}_{i+1,0} = \sum_{j=0}^{K} \ell_j(1)\tilde{\bar{c}}_{ij} & i = k-N,\dots,k-1 \\ & \tilde{\bar{c}}_f = \sum_{j=0}^{K} \ell_j(1)\tilde{\bar{c}}_{kj} & \tilde{\bar{c}}_{1,0} = \tilde{\bar{c}}(t_0) \end{array} \tag{3.20}$$

where z is a vector containing the collocated states

$$z := \left[\tilde{\bar{c}}_{k-N,0}, \dots, \tilde{\bar{c}}_{k-N,K}, \tilde{\bar{c}}_{k+1-N,0}, \dots, \tilde{\bar{c}}_{k+1-N,K}, \dots, \tilde{\bar{c}}_{k,K}\right]^T, \tag{3.21}$$

and $\tilde{\bar{c}}_i = \tilde{\bar{c}}_{i,0}$.

The alternative to direct collocation is direct multiple shooting, and the problem would then become

$$\min_{\substack{\tilde{\bar{c}}_{k-N},\dots,\tilde{\bar{c}}_{k-1},\tilde{\bar{c}}_k \\ \tilde{p} \\ \tilde{w}_{k-N},\dots,\tilde{w}_{k-1}}} \Gamma\left(\tilde{\bar{c}}_{k-M}, c_{\partial\Omega}, u\right) + \sum_{j=k-N}^{k} \mathcal{L}_c\left(e_j\right) + \sum_{j=k-N}^{k-1} \mathcal{L}_w(\tilde{w}_j) \tag{3.22}$$

$$\begin{array}{lll} s.t. & \tilde{\bar{c}}_{i+1} - F\left(\tilde{\bar{c}}_i, c_{\partial\Omega}, u, t_{i+1}, t_i\right) = 0 & i = k-N,\dots,k-1 \\ & \tilde{\bar{c}}_{j,min} \leq \tilde{\bar{c}}_j \leq \tilde{\bar{c}}_{j,max} & j = k-N,\dots,k-1,k, \\ & p_{min} \leq p \leq p_{max} & \\ & \tilde{\bar{c}}_f = F\left(\tilde{\bar{c}}_k, c_{\partial\Omega}, u, t_{k+1} - t_k\right) & \tilde{\bar{c}}_0 = \tilde{\bar{c}}(t_0) \end{array} \tag{3.23}$$

4 Tools

The implementation is written in Python with the use of SciPy, Numpy, Matplotlib, CasADi, IPOPT and CVODES. The implemented code can be seen in Appendix A.

4.1 Python

Python is a cross-platform programming language, and is very powerful when considering flexibility, syntax, style and extendability (Bressert, 2012). It support interactive mode and it is easy to combine with compiled languages like Fortran, C and C++ (Langtangen, 2012). This makes it a fast language to work with, and together with the packages SciPy, Matplotlib and Numpy it offers much of the same functionality as Matlab (Jones et al., 2001–; Hunter, 2007; Oliphant, 2007).

4.2 CasADi

CasADi is a free, open source software written in self containing code(Andersson et al., 2012*a*). The software can be used with Python, Octave and C++, and it is intended for use with nonlinear optimization problems with focus on dynamic optimization problems.

It can be described as a minimalistic computer algebra system (CAS) and implements automatic differentiation (AD). AD is an algorithmic derivative method where the calculation of the derivative is done by the chain rule. For more information on the subject, see Griewank and Walther (2008).

In order to solve the optimization problems, CasADi supplies interfaces to several numerical NLP solvers, ODE/DAE integrators, quadratic programming (QP) and linear systems solvers (Andersson et al., 2012*a*,*b*).

In this report, CasADi has been used through Python to implement the model symbolically, formulating the MHE problem, setting up the hybrid EKF, and interfacing CVODES and IPOPT.

4.3 CVODES

CVODES is an ODE solver for IVPs from SUNDIALS (Hindmarsh et al., 2005). It is a stiff and nonstiff solver with sensitivity analysis capabilities and solves the ODE by either backward differentiation formula or Adams-Moulton Serban and Hindmarsh (2005). When CVODES is used through the CasADi interface, the necessary Jacobian and the forward and adjoint sensitivity equations are automatically calculated by CasADi.

CVODES has been used used in this report to simulate the model, and to create datasets for the MHE and EKF to work on.

4.4 IPOPT

IPOPT is a NLP solver based on Interior Point OPTimization. It has been interfaced through CasADi, which means that CasADi will supply it with the Jacobian of the constraints and the Hessian of the Lagrangian function. The mathematics behind IPOPT can be read about in (Wächter and Biegler, 2006).

IPOPT is an interior point method which makes it a very efficient solver for large and sparse problems. When using MHE together with direct collocation the resulting NLP problem becomes large and sparse (Zavala and Biegler, 2009; Biegler, 2010).

The MHE problem (3.19) and (3.20) has been solved with IPOPT using the MA27 linear solver from HSL (2013). MA27 solves sparse symmetric systems.

5 Problem setup

This section will cover the setup of the problem. That is, which scenarios that has been chosen, tuning of the different algorithms and general settings for the simulations.

5.1 Scenarios

There will be simulations of several scenarios. Some of the scenarios will be to illustrate the performance of the system, and some will be more focused on realistic similarities.

In the scenarios, it will be assumed that some of the states are known through measurements. The measurements represent the ice concentration in a state given in the range of $[0, 1]$, where 0 is open water, 1 is completely covered by ice, and the values between indicates a combination. With infinite spatial resolution, the measurement would be either 0 or 1. In this setup the resolution will be limited to $r_x = 15$ for the one dimensional case, and $r_x = 5$ and $r_y = 5$ when operating with two dimensions. The values are a trade-off between simulation time and accuracy.

The simulation will be over an area of $750\mathrm{m} \times 750\mathrm{m}$, which in the one dimensional case will be a $750\mathrm{m}$ wide area. This is a relatively small area for ice surveillance, but the effects seen in the simulations will be similar to those one would expect to see in a larger case. Ice drifting into the area will be constant in some of the simulations and time dependent in others. The simulations with constant boundary are included in order to study the performance of constant variable estimation.

The time horizon will be $900\mathrm{s}$ (15 minutes) discretized in $t_n = 51$ steps.

There will be simulations with both constant and time dependent velocity, and the velocity will be in the range of $[-1, 1]\mathrm{m/s}$. This is a realistic range according to Hamilton (2011).

The initial values for for both estimation schemes has been set to zero for all parameters and states. Both process and measurement noise has been left out of the simulation.

5.2 Tuning of the hybrid EKF

The tuning in the EKF is done through the covariance matrices R and Q. The preferred values was found by trial and error, and validated through estimation tests. The result was the following diagonal weighting matrices

$$Q = \begin{bmatrix} q_{1,1} & 0 & \dots & 0 \\ 0 & q_{2,2} & \iddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & q_{n_z,n_z} \end{bmatrix}, \tag{5.1}$$

$$R = \begin{bmatrix} r_{1,1} & 0 & \dots & 0 \\ 0 & r_{2,2} & \iddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & r_{n_y,n_y} \end{bmatrix}, \tag{5.2}$$

where n_z is sum of states and unknown parameters and n_y is the number of measurements. An element $q_{i,i}$ in the matrix corresponds to the covariance associated with Z_i, where Z_i is the i-th element in the augmented state vector (3.8). The following weights was found by trail and error

$$q_{i,i} = \begin{cases} 0.1 & \text{if } Z_i \text{ is a measured state} \\ 1.5 & \text{if } Z_i \text{ is a parameter estimate} \\ 1.1 & \text{if } Z_i \text{ is a state estimate} \end{cases}, \tag{5.3}$$

and the elements $r_{j,j} = 0.1$ for $i = \{0, \dots, n_z\}$ and $j = \{0, \dots, n_y\}$.

5.3 Tuning of the MHE

There are a couple of parameters which can be tuned in the MHE formulation, including the weighting matrices, boundaries, the horizon length, time step for the ERK method for multiple shooting and degree of the polynomial approximation in the collocation method.

5.3.1 Weighting matrices

The model used is fully known, and it was hence decided to drop the process noise term. That is $Q = 0$. This reduces the number of variables in the NLP problem, but it makes the problem strict since there is no room for model inaccuracies if the process noise is removed. With no Q matrix, the weights in the weighting matrix R becomes less important and it has been set to the identity matrix.

5.3.2 Boundaries

The ability to put boundaries on the variables is one of the strengths in MHE. The states in this study represent ice concentration, which only gives meaning in the interval $[0, 1]$, and the minimum value for the states has hence been set to 0 and the max to 1. The same goes for the boundary, while the velocity has a minimum value of -3 and maximum of 3 which should be more than enough.

5.3.3 Time step h for the fourth order ERK method

The performance of the fourth order ERK solver used in the multiple shooting formulation has been tested against CVODES, a variable step ODE solver in order to test its accuracy. The results is included in Appendix B. The time step h was chosen to $h = 1$ as a compromise between speed and accuracy.

5.3.4 Horizon length

Since the parameters are constant in the estimation scheme, a short horizon will make it more adaptable to slowly varying parameters. The velocity might be slowly varying, but the boundary will be a fast changing parameter in most cases. The optimal horizon found by trial and error was $nk = 5$ and has been used throughout the simulations. Longer horizon length introduces a larger NLP problem, which increases the need for computational power and time.

5.3.5 Tuning of the collocation algorithm

The direct collocation algorithm is based on a polynomial interpolation where the interpolation points can be tuned. Table 3 shows the optimal interpolation points for a polynomial up to a degree of five. A comparison between multiple shooting and direct collocation with a three and fifth order polynomial can be seen in Appendix B. The interpolation points used in this study was chosen as the fifth order Radu roots.

5.4 Choosing MHE method

A comparison between the direct collocation and multiple shooting method was performed and is included in Appendix B. The comparison was performed in order to decide which method to use. The results showed that the time used by multiple shooting was many times as high as the one with collocation, and the differences in the estimates was low in most of the tests. This lead to the conclusion to use direct collocation as the discretization method in the MHE formulation.

5.5 Error calculation

A function to calculate the error from the different methods has been implemented. The error is calculated with quadratic cost, that is

$$\begin{aligned} Error_{states} &= \sum_{i=0}^{i=N} \left[c_{est,i} - c_{real,i}\right]^T \left[c_{est,i} - c_{real,i}\right], \\ Error_{parameters} &= \sum_{i=0}^{i=N} \left[p_{est,i} - p_{real,i}\right]^T \left[p_{est,i} - p_{real,i}\right], \\ Error_{total} &= Error_{states} + Error_{parameters}, \end{aligned} \tag{5.4}$$

where N is the number of estimates calculated.

6 Results

The simulations which has been run shows different scenarios with combinations of state and parameter estimations. The grid on the state estimation plots corresponds with the spatial discretization.

6.1 Simulations in one dimension

Most of the simulations has been performed in one dimension. The results are easier to represent graphical in one dimension, and it is easier to compare plots. Most of the results from the simulation in one dimension will be applicable for the case with two dimensions.

6.1.1 Velocity estimation

The first simulation is velocity estimation where every state and the boundary is known. The spatial resolution is $r_x = 15$, and the boundary values are $c_{\partial\Omega} = [1, 0]^T$ which means that ice is drifting in from the left.

Table 4 sums up data about the estimation.

Table 4: Data from the velocity estimation

	EKF	MHE
Time used:	2.02s	226.95s
Total error:	$1.033 \cdot 10^{-2}$	$3.392 \cdot 10^{-4}$
– State error:	$1.502 \cdot 10^{-9}$	$1.445 \cdot 10^{-4}$
– Parameter error:	$1.033 \cdot 10^{-2}$	$1.947 \cdot 10^{-4}$
Max rank on linearised observability matrix:	16	
Min rank on linearised observability matrix:	16	

A plot showing the results can be seen in Figure 13 in Appendix C. The figure shows some snapshots of the states from $t_0 = 0$ up to $t_f = 900$, and the estimated velocity.

6.1.2 Velocity and state estimation (two states are measured)

The next simulation shows the performance when the velocity is estimated from only two states, c_0 and c_5. The boundary values are known and are the same as in the previous simulation. The results can be seen in Figure 3 and Table 5.

Table 5: Data from the velocity and state estimation (two states are measured)

	EKF	MHE
Time used:	2.20s	520.18s
Total error:	0.0610	23.258
– State error:	0.0370	8.2939
– Parameter error:	0.0240	14.964
Max rank on linearised observability matrix:	16	
Min rank on linearised observability matrix:	10	

6.1.3 Parameter and state estimation (four states are measured)

Figure 4 shows a simulation where both the velocity and boundary concentration are estimated. The measured states are c_0, c_4, c_5 and c_9, which means that the 11 other states also are estimated. Table 6 shows the error from the estimates.

Table 6: Data from the parameter and state estimation (four states are measured)

	EKF	MHE
Time used:	2.90s	442.17s
Total error:	29.891	38.878
– State error:	17.406	7.5239
– Parameter error:	12.484	31.354
Max rank on linearised observability matrix:	18	
Min rank on linearised observability matrix:	14	

6.1.4 Parameter and state estimation (three states are measured)

The result from removing one of the measurements from the previous simulation is shown in Figure 5. Figure 5b and Figure 5d shows that the performance with only three measurements are poor which can be verified by the errors shown in Table 7. It is only c_0, c_5 and c_9 which are measured.

6.1.5 Parameter and state estimation, time dependent boundary

The simulation with time varying boundary is shown in Figure 6 and Table 8, where the simulation illustrate what it might look like if an ice floe came drifting into the area.

Table 7: Data from the parameter and state estimation (three states are measured)

	EKF	MHE
Time used:	3.08s	1323.94s
Total error:	167.37	79.014
– State error:	133.44	45.118
– Parameter error:	33.928	33.895
Max rank on linearised observability matrix:	18	
Min rank on linearised observability matrix:	9	

Table 8: Data from the parameter and state estimation, time dependent boundary

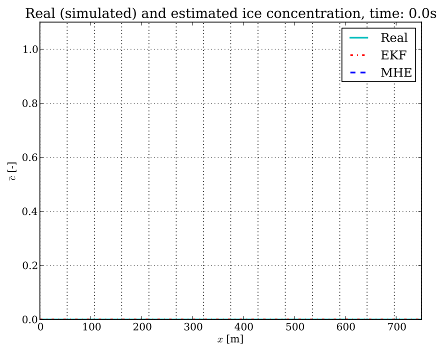
	EKF	MHE
Time used:	2.91s	1147.94s
Total error:	40.711	25.697
– State error:	23.267	5.1373
– Parameter error:	17.444	20.560
Max rank on linearised observability matrix:	18	
Min rank on linearised observability matrix:	14	

6.1.6 Parameter and state estimation, time dependent boundary and velocity

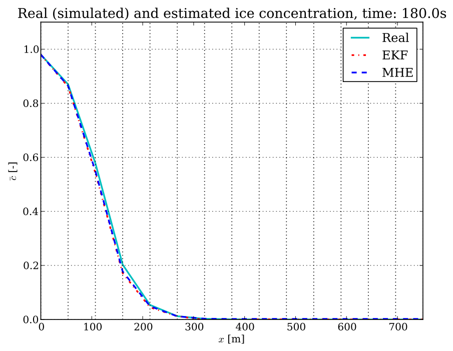
In a real life scenario, both boundary and velocity would be dependent of time. This has been simulated in Figure 7. Table 9 shows that the MHE does a better estimate than the EKF.

Table 9: Data from the parameter and state estimation, time dependent boundary and velocity

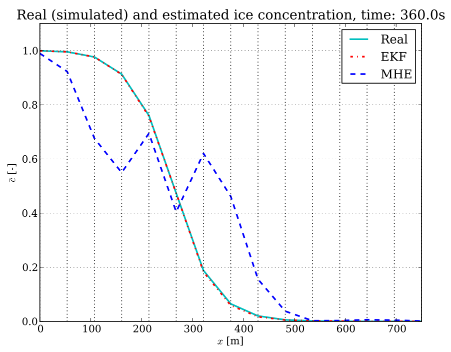
	EKF	MHE
Time used:	2.85s	849.45s
Total error:	63.274	28.621
– State error:	42.862	14.339
– Parameter error:	20.411	14.282
Max rank on linearised observability matrix:	18	
Min rank on linearised observability matrix:	15	



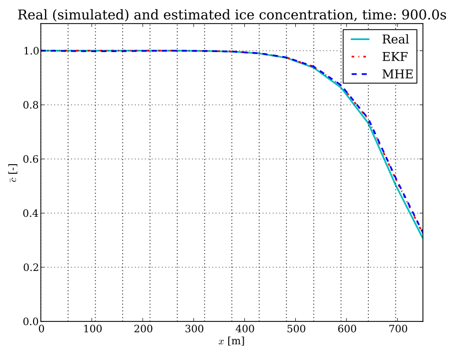
(a) Initial values



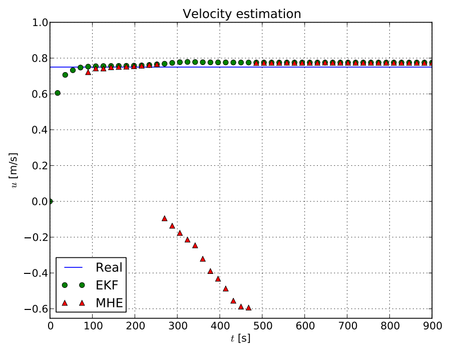
(b) The states after 180s



(c) The states after 360s

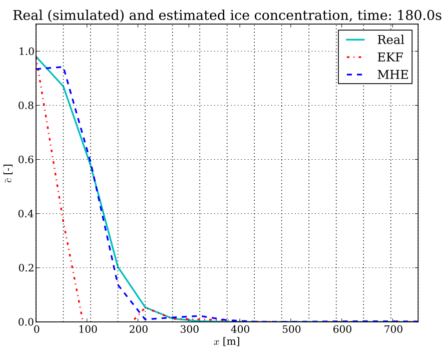


(d) The states at the end

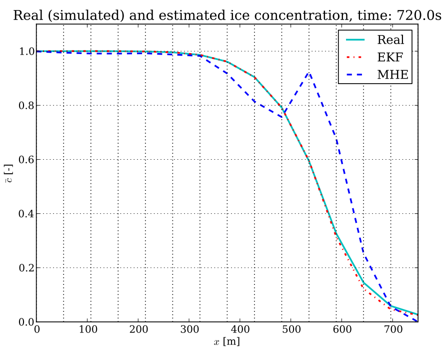


(e) The estimated velocity

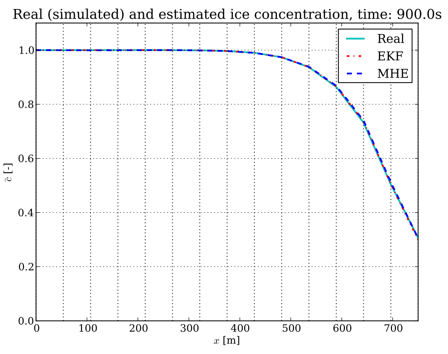
Figure 3: Simulation of state and parameter (velocity) estimation. Only the states c_0 and c_5 are measured, and the boundary is assumed to be known.



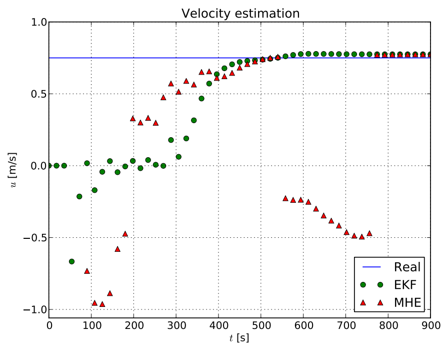
(a) The states after 180s



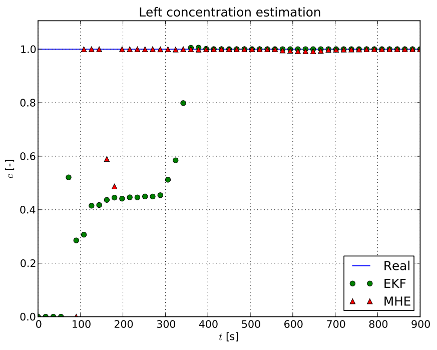
(b) The states after 720s



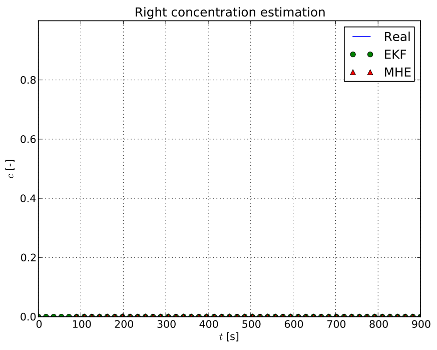
(c) The states at the end



(d) The estimated velocity

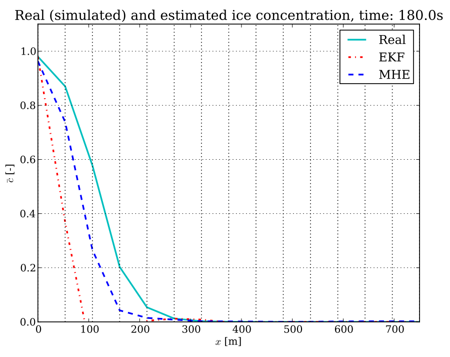


(e) The estimated left boundary

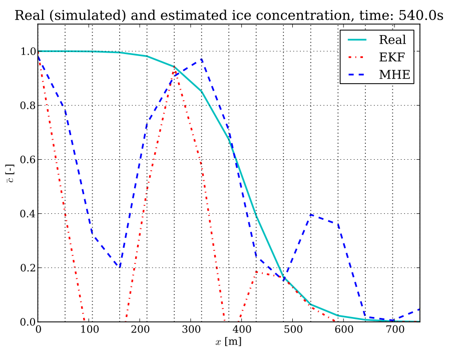


(f) The estimated right boundary

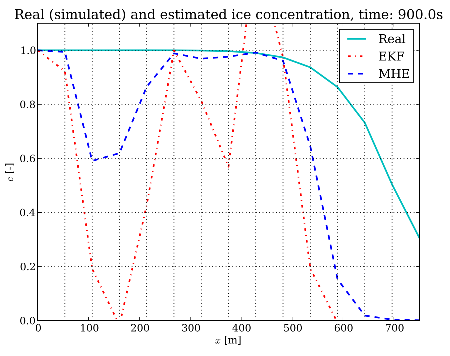
Figure 4: Simulation of state and parameter estimation. The states c_0, c_4, c_5 and c_9 are measured. The rest of the states together with all the parameters are estimated.



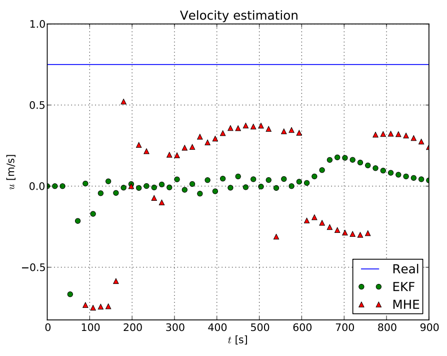
(a) The states after 180s



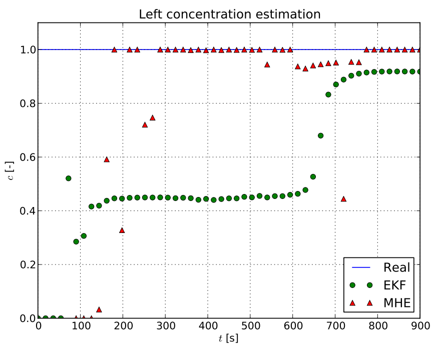
(b) The states after 540s



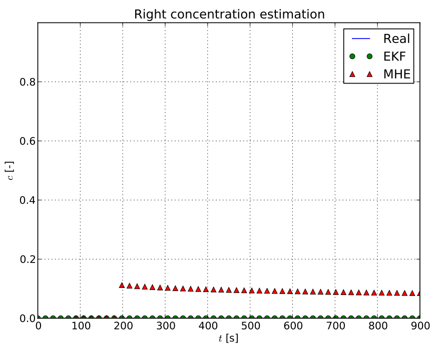
(c) The states at the end



(d) The estimated velocity

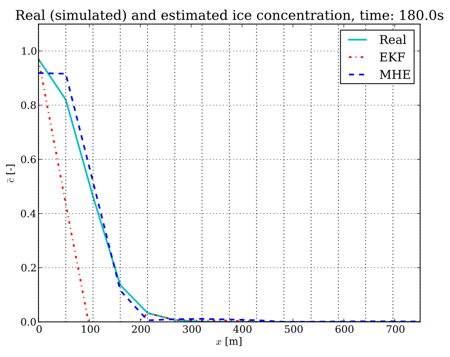


(e) The estimated left boundary

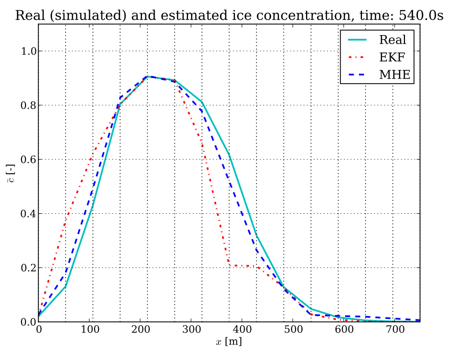


(f) The estimated right boundary

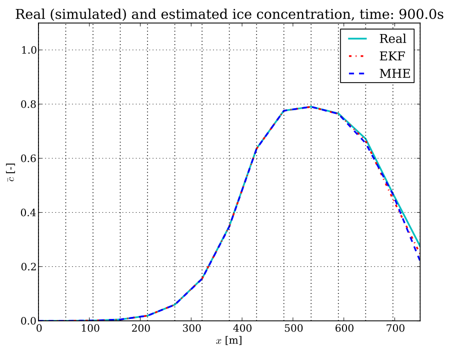
Figure 5: Simulation of state and parameter estimation. The states c_0, c_5 and c_9 are measured. The rest of the states together with all the parameters are estimated.



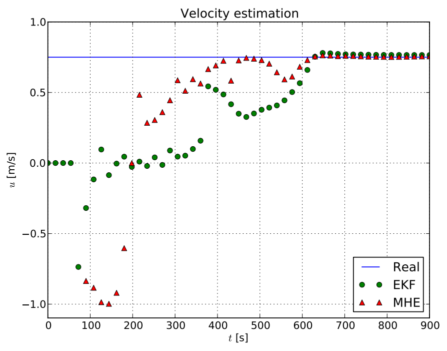
(a) The states after 180s



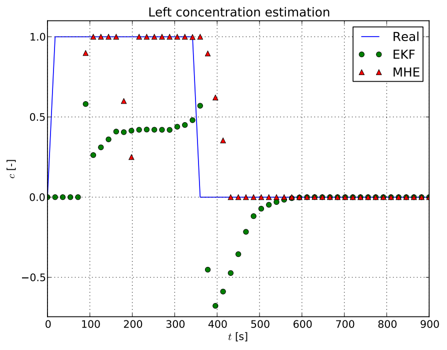
(b) The states after 540s



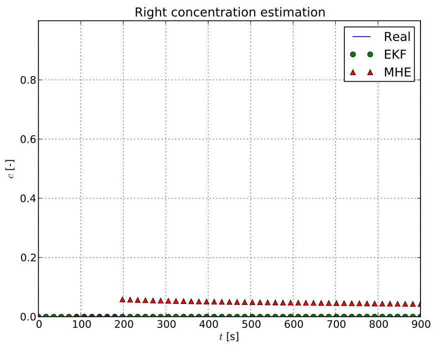
(c) The states at the end



(d) The estimated velocity

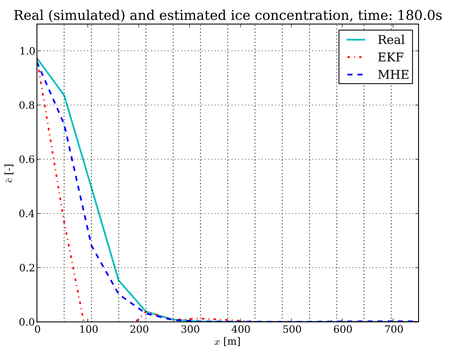


(e) The estimated left boundary

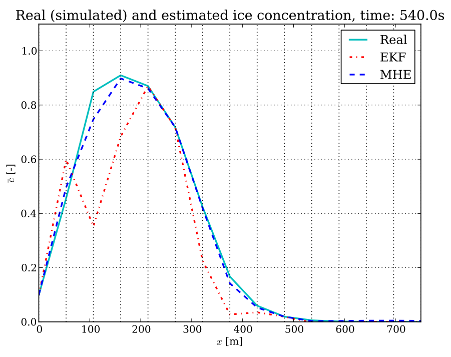


(f) The estimated right boundary

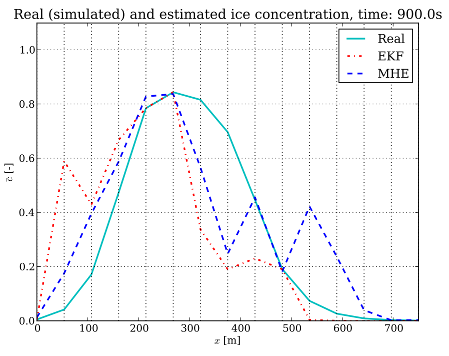
Figure 6: Simulation of state and parameter estimation. The states c_0, c_4, c_5 and c_9 are measured. The rest of the states together with all the parameters are estimated. The left boundary varies with time.



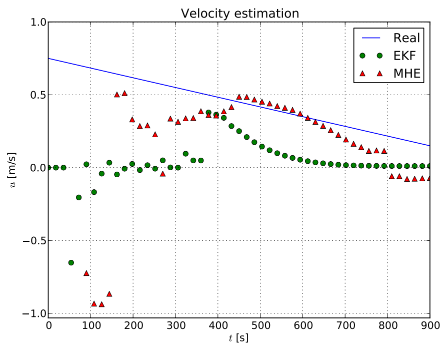
(a) The states after 180s



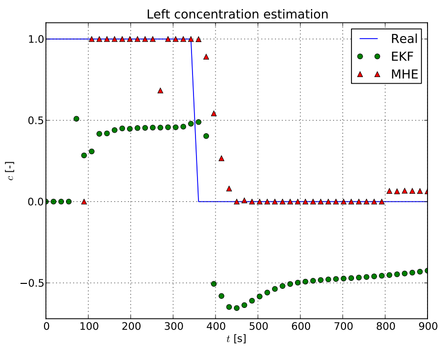
(b) The states after 540s



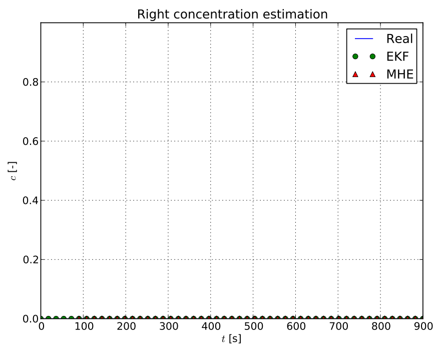
(c) The states at the end



(d) The estimated velocity



(e) The estimated left boundary



(f) The estimated right boundary

Figure 7: Simulation of state and parameter estimation. The states c_0, c_4, c_5 and c_9 are measured. The rest of the states together with all the parameters are estimated. The left boundary and the velocity varies with time.

6.2 Simulations in two dimensions

The more realist approach is to use two dimensions. The model used to simulate the ice concentration when operating in two dimensions is more complex, and it is only the EKF which manage to produce estimates within a reasonable amount of time when the spatial resolution is $r_x = 5$ and $r_y = 5$.

6.2.1 Parameter estimation in two dimensions

Figure 8 shows a simulation where all the parameters are unknown, and all the states are measured. Figure 8e and Figure 8f shows that the estimated velocity converge to the real velocity, but it can be seen from Table 10 that there are some errors in the estimate of the boundary.

Table 10: Data from the parameter estimation in two dimensions

	EKF
Time used:	376.41s
Total error:	87.132
– State error:	0.0010
– Parameter error:	87.131
Max rank on linearised observability matrix:	44
Min rank on linearised observability matrix:	30

Note that all the parameters was estimated. The number n_z of variables is therefore $n_z = n_x + n_p$, where $n_x = r_x \cdot r_y = 25$, and $n_p = 2 + 2r_x + 2r_y = 22$. Thus the number of variables to estimate is $n_z = 47$.

6.2.2 Velocity and state estimation in two dimensions

Figure 9 shows a simulation with both state and velocity estimation. Only the states $c_{0,1}$, $c_{1,0}$ and $c_{2,3}$ are measured. The boundary values are known. Figure 9e and Figure 9f shows that the velocity estimations are close to the real value. Table 11 confirms that the estimation error is very small.

The number n_z of variables is therefore $n_z = n_x + n_p$, where $n_x = r_x \cdot r_y = 25$, and $n_p = 2$. This makes it a total of $n_z = 27$ variables to estimate.

Table 11: Data from the velocity and state estimation in two dimensions

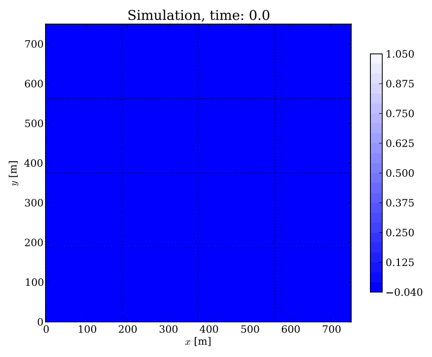
	EKF
Time used:	22.77s
Total error:	3.1784
– State error:	1.2075
– Parameter error:	1.9709
Max rank on linearised observability matrix:	27
Min rank on linearised observability matrix:	19

6.2.3 Parameter and state estimation in two dimensions

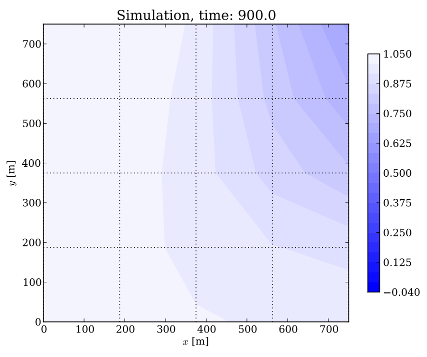
Figure 10 shows a simulation with both state and parameter estimation. Only the states $c_{0,1}$, $c_{1,0}$, $c_{4,0}$, $c_{3,1}$, $c_{2,2}$, $c_{1,3}$, $c_{0,4}$, $c_{2,1}$ and $c_{1,2}$ are measured. Figure 10e and Figure 10f shows that only the velocity u in x direction is estimated close to the real value. Table 12 shows that the error in the estimated states are very low compared to the estimated parameters.

Table 12: Data from the parameter and state estimation in two dimensions

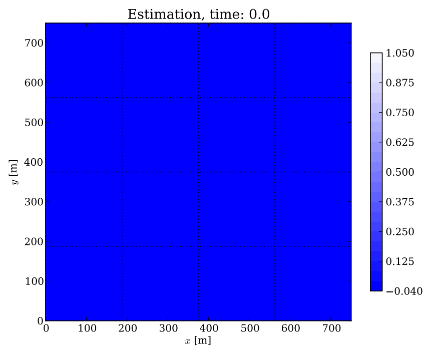
	EKF
Time used:	507.49s
Total error:	459.40
– State error:	24.460
– Parameter error:	434.94
Max rank on linearised observability matrix:	34
Min rank on linearised observability matrix:	32



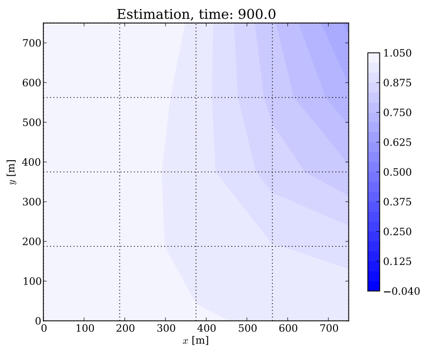
(a) The simulations initial condition



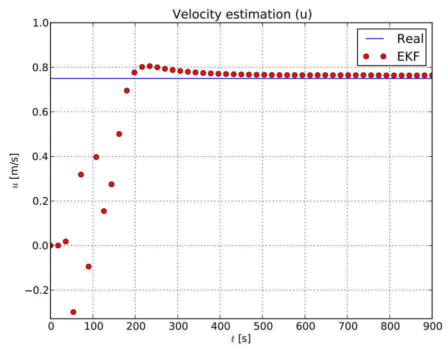
(b) The estimation initial condition



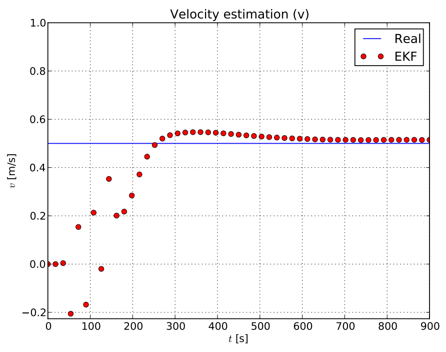
(c) The simulations at the end



(d) The estimation at the end

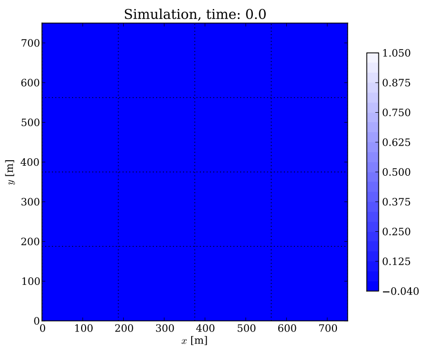


(e) The estimated velocity u

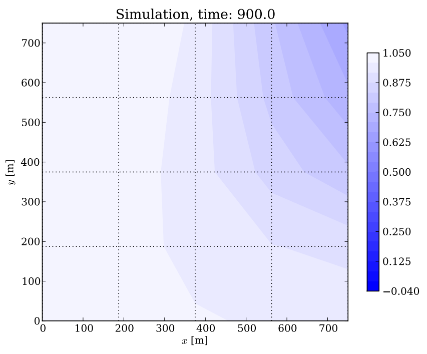


(f) The estimated velocity v

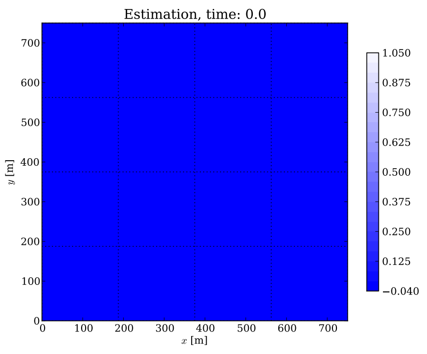
Figure 8: Simulation in two dimensions. Every state is known, but the velocity and boundary are unknown. The blue area indicates water, while white indicates ice.



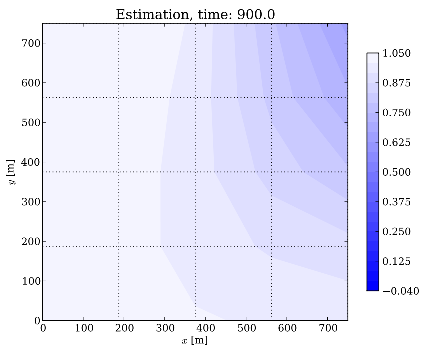
(a) The simulations initial condition



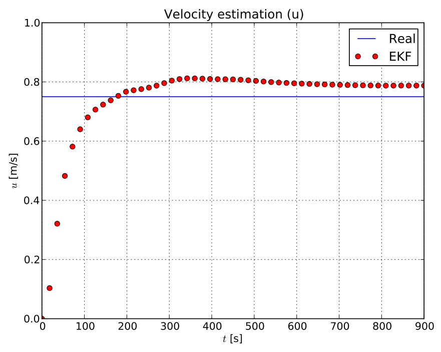
(b) The estimation initial condition



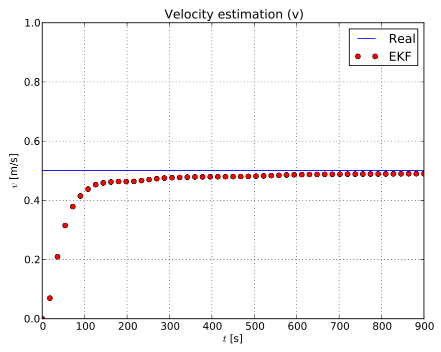
(c) The simulations at the end



(d) The estimation at the end

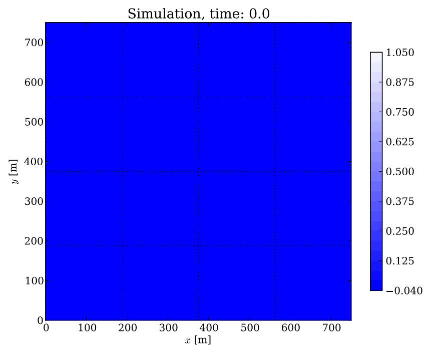


(e) The estimated velocity u

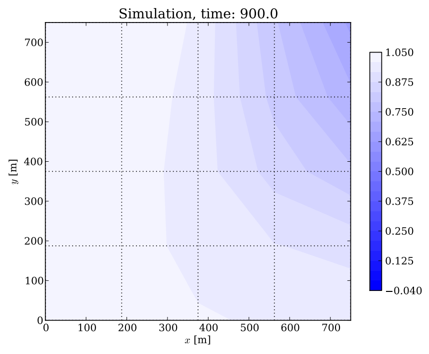


(f) The estimated velocity v

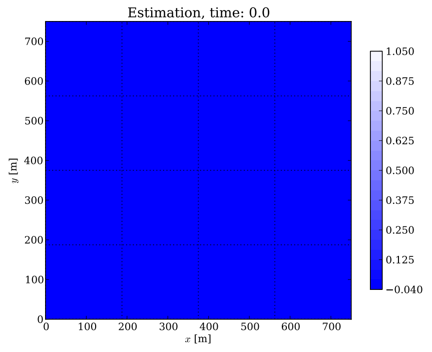
Figure 9: Simulation in two dimensions. The states $c_{0,1}$, $c_{1,0}$ and $c_{2,3}$ together with the boundary are known. The blue area indicates water, while white indicates ice.



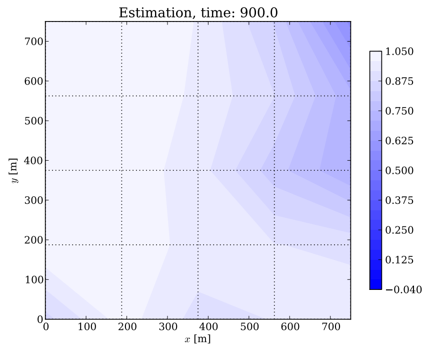
(a) The simulations initial condition



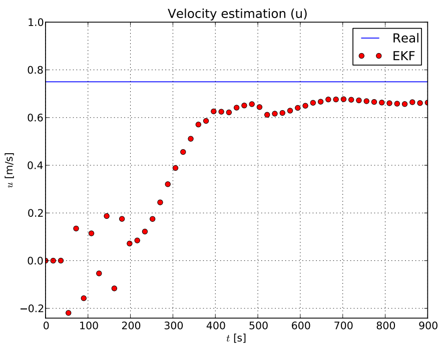
(b) The estimation initial condition



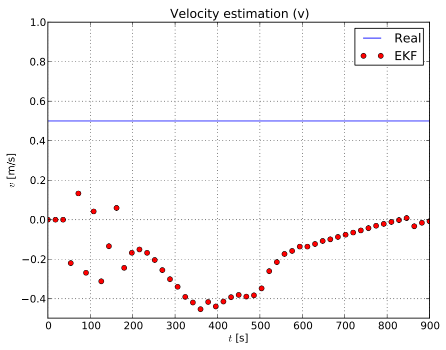
(c) The simulations at the end



(d) The estimation at the end



(e) The estimated velocity u



(f) The estimated velocity v

Figure 10: Simulation in two dimensions. The states $c_{0,1}$, $c_{1,0}$, $c_{4,0}$, $c_{3,1}$, $c_{2,2}$, $c_{1,3}$, $c_{0,4}$, $c_{2,1}$, and $c_{1,2}$ is known. The other states together with the velocity and boundary are unknown. The blue area indicates water, while white indicates ice.

7 Discussion

It is clear from the results that it is possible to estimate the ice drift velocity. The first simulation shown in Figure 13 demonstrates how accurate the estimation is when only the ice drift velocity is unknown. The result is as expected since there is only one unknown. The linearised observability matrix has full rank throughout the estimation, and hence support that the problem is solvable.

The next simulation extended the problem to include both states and parameters. Only two states were measured, and the rest of the states together with the velocity was estimated. The boundary was constant and known. Figure 3 shows that the estimator manages to estimate the unknown states and velocity. From Figure 3c and Figure 3e it can be seen that the MHE algorithm has some problems. Figure 3e shows that estimated velocity is completely off from $270\mathrm{s}$ to $470\mathrm{s}$. It might be that the NLP problem is ill conditioned in the interval. IPOPT failed to converge to a solution during some of the iterations, and that explains the drastic increase in time used to solve the problem.

Extending the problem to include full parameter estimation, based on a few state measurements, can be seen in Figure 4 and Figure 5. In the first case with four measurements located at c_0, c_4, c_5 and c_9, all the parameters converge to their correct values. In the other simulation, the measurement of c_4 was left out. The result with one less measurement was that both the EKF and MHE failed to estimate both states and parameters. The MHE formulation still has problems with the velocity estimation as shown in Figure 5d. It was noted from running the simulations that IPOPT failed to find the minimum several times during the iterations.

Figure 6 and Figure 7 contains simulations with time varying parameters. The first case contains a time varying boundary while both boundary and velocity are time varying in the second case. The MHE makes better estimates with the time varying parameters compared to the simulation with constant parameters, while the EKF makes slightly worse estimates with variable parameters. The reason for the improvements in the MHE is probably due to the higher variations in the measured states. The result shows that the MHE manage to estimate time varying parameters even when the parameters are constants in the MHE problem. This is related to the horizon length $n_k = 5$, where a short horizon makes the periods with constant parameters shorter.

The simulations in two dimensions was only performed with the EKF and constant parameters. The MHE was too computationally expensive. The reason might be that the implemented model is too complex. By printing the symbolic equation for $\dot{c}_0$, it is observed that the print contains around 11000 symbols (including brackets and variable names such as $c_{i,j}$).

The simulation shown in Figure 8, where all the parameters are estimated, shows that it is possible to estimate all the variables based on measurements of all the states. It is only the velocities estimation which are shown since those are the ones of interest. Note that the linearised observability matrix is not observable, and the errors shown in section 6.2.3 indicates that it does not manage to estimate all the boundary values. Figure 9 shows that it is possible to estimate most of the states and the velocity if the boundary values are known.

A simulation where only some of the states are measured has been performed, and the results are shown in Figure 10. It shows that it almost manages to estimate the velocity u in the x direction. The linearised observability matrix reaches at most a rank of 34 whereas the number of parameters in the estimation problem is 47. This indicate that it might not be possible to estimate the parameters based on the current configuration. A problem with the simulation is the low resolution, which makes the number of parameters very high compared with the number of states.

The effect seen in some of the plots, both states and parameters "jumps" out of position and that IPOPT fails to find the minimum, might be related to the objective function in cooperation with constraints. It might be related to small differences in the dynamic constraints and the real model, that is both collocation and multiple shooting with the ERK solver relies on simplifications compared to the CVODES solver used for simulation. The noise term q in the implemented MHE was dropped due to the simulations being noise free. This might have been an oversimplification since there might be small differences in the mentioned ODE solver, which the process noise term would have taken care of.

8 Conclusion and future work

The results show that the problem with estimating sea ice drift velocity is solvable. The important question is only how much data is available. It has been shown to be an easy problem when both states and boundaries are measured or known. This is not usually the case, but it might be applicable to aerial sensors or satellites. The fact that all states are measured simultaneously usually implies a small area with normal resolution or a large area with low resolution.

The case where only some of the states are measured showed that it was possible to estimate both unknown states and parameters. It showed that around four states were needed in one dimension, and around 10 states were needed in the two dimensional case. The two dimensional estimation problem would probably be easier to solve with higher resolution, given the same ratio between measured and unmeasured states, due to the high amount of boundary values. This will most likely be the case in real world applications.

The simulations showed that there were only a few cases where MHE provided better estimates than EKF, but it used several times the amount of time as the EKF. The problem with MHE was mainly related to ill conditioned problems which lead to IPOPT reaching its maximum number of iterations. The performance of MHE with respect to estimation error was superior in the cases with time varying parameters, which are the realistic scenarios. It should be noted that the objective function in the EKF and the MHE was different since the MHE did not include process noise.

It is noted that with the current implementation, measurements of many of the states are needed in order to achieve good estimates of the sea ice drift velocity. In cases where enough measurements are available, the estimates of the sea ice drift velocity in combination with ice coverage estimates might be very useful for offshore activities.

Good ice management and ice surveillance in the combination with a warmer Arctic might make the NSR more accessible. Estimates of the ice drift velocity in cooperation with ice concentration estimates could make it possible to do path planning in order to avoid ice.

In the case of the oil and gas industry, where the main challenge is the desire to do reliable station-keeping operations, better ice velocity estimates together with ice concentration estimates of the area might make it possible to extend the time of safe station-keeping.

8.1 Future work

During the study, some areas were discovered which would be interesting to study further. This includes the possibility of changing the NLP problem in cases where the data is not exciting, as proposed in Sui and Johansen (2011).

Another improvement might be to combine the estimation schemes with the use of smart mobile sensors, which would make it possible to get state corrections for a large area. The use of mobile sensors might help on the exciting of the data, and thus make the estimation problem easier to solve.

In order to make MHE faster, a simpler model for use in the MHE formulation should be investigated, including a simpler spatial discretization scheme. The inaccuracy caused by a simpler model might not be a problem since other inaccuracies between the model and reality will be present.

The system proposed in this study is general in the sense that the model used describes the transport of conserved quantities. It might be of interest in areas other than ice management, and could maybe be used to estimate oil spills.

Appendix A CD

The implemented code has been included in a CD attached to the report. A short summary of the included material is:

Folder 1: Source code. Contains the source files in order to run the simulations.

RunEstimationOfDriftIceVelocity.py The main file used to run the simulation. Most settings will be defined in the file.

iceConcentrationModel1D.py and *iceConcentrationModel2D.py* contains the spatial discretized model for the one and two dimensional cases respectively.

iceConcentrationSimulation1D.py runs the simulation and generates the dataset for estimation. The datasets are saved to txt files.

iceConcentrationMHE.py is the MHE algorithm done with direct collocation together with the hybrid EKF. It outputs the measurements as txt files.

iceConcentrationMHEmultipleShooting_RK4.py is the MHE algorithm done with multiple shooting.

iceConcentrationPlotting1D.py and *iceConcentrationPlotting2D.py* are used to plot in one and two dimensions respectively.

iceConcentrationParameterPlotting.py is used to plot and compare the parameters.

RK4.py includes the implementation of the fourth order ERK method.

iceConcentrationHEKF.py is the implemented hybrid EKF.

estimationErrorFcn.py includes a function for calculating quadratic errors.

Appendix B Comparison plots

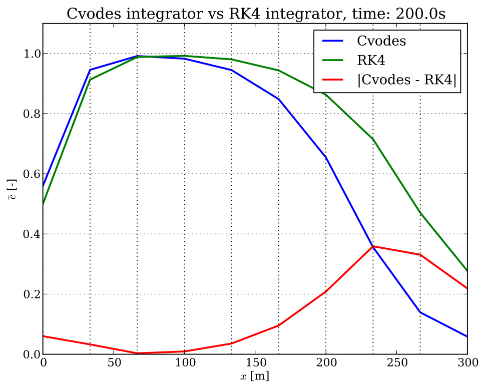
Comparison between CVODES and a fourth order ERK method

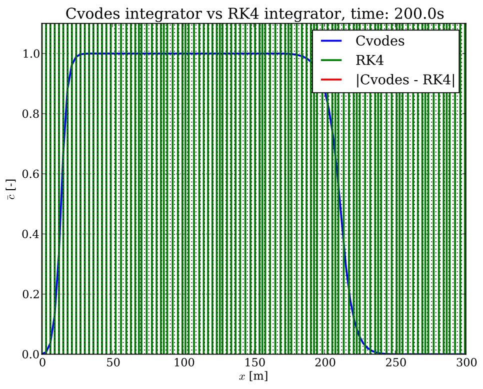
In order to determine what time step h should be used in the fourth order ERK method (2.11e), a series of simulations comparing the variable step solver CVODES against the ERK method was performed. The velocity of the drifting ice in the simulation is set to $1.5\mathrm{m/s}$.

The tests were performed with different spatial resolutions and time steps in order to determine how large the time step could be before the solver got unstable. Figure 11b shows that the ERK method is unstable for $h = 2$ when the spatial resolution is $r_x = 100$, and Figure 11a shows that the error is large even for $r_x = 10$. The simulation time with the ERK method is $3.68\mathrm{s}$ for $r_x = 100$ and $0.35\mathrm{s}$ for $r_x = 10$.

Figure 11c and Figure 11d shows the difference when $h = 1$, and it is stable for both $r_x = 10$ and $r_x = 100$. The time used to run the simulation is $6.18\mathrm{s}$ and $1.58\mathrm{s}$ for $r_x = 100$ and $r_x = 10$ respectively.

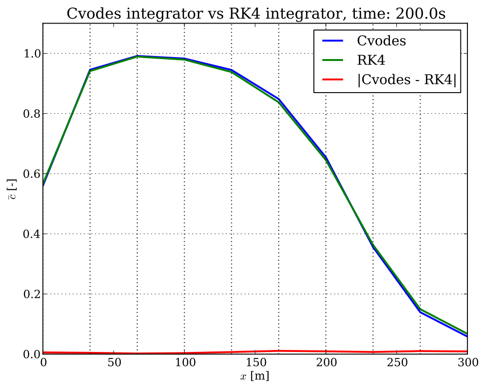
A simulation with $h = 0.5$ can be seen in Figure 11e and Figure 11d. The figures shows that the error decrease when h decreases. The simulation took $12.25\mathrm{s}$ for $r_x = 100$ and $1.27\mathrm{s}$ when $r_x = 10$.

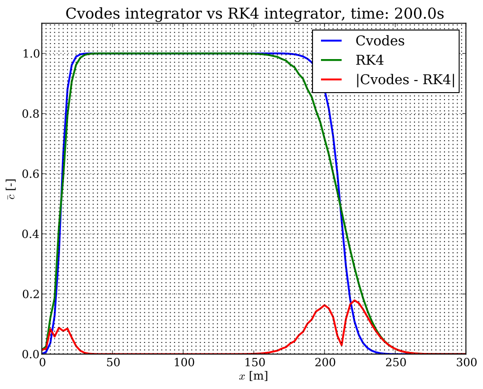




(a) Spatial discretized with 10 points, time step $h = 2$

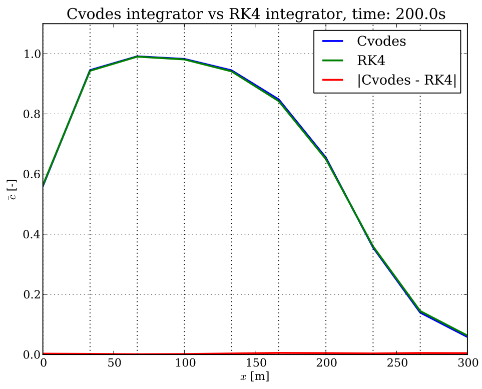
(b) Spatial discretized with 100 points, time step $h = 2$

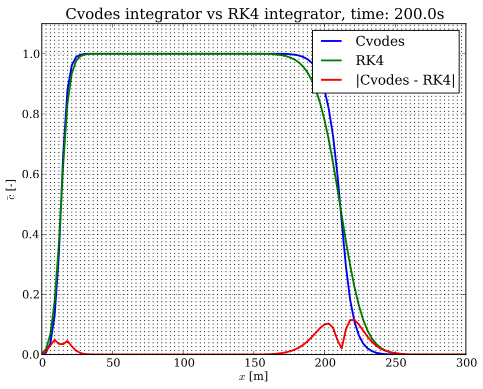




(c) Spatial discretized with 10 points, time step $h = 1$

(d) Spatial discretized with 100 points, time step $h = 1$





(e) Spatial discretized with 10 points, time step $h = 0.5$

(f) Spatial discretized with 100 points, time step $h = 0.5$

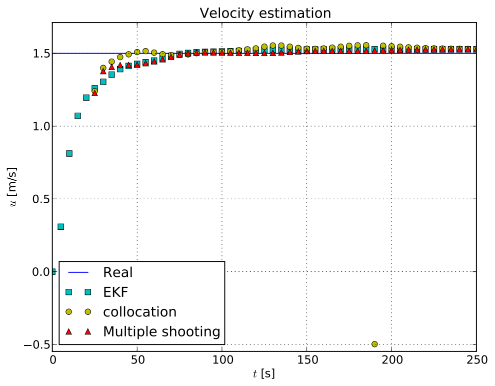
Figure 11: A comparison between the variable step solver CVODES and the fixed step fourth order ERK solver. Simulated with different time steps and spatial resolution.

Comparison between multiple shooting, direct collocation and EKF

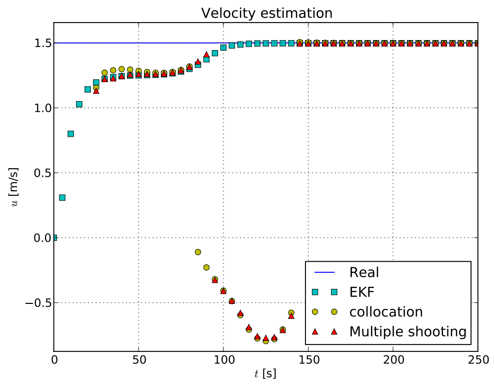
The performance of MHE implemented with multiple shooting has been tested against the MHE implemented with direct collocation. The result can be summed up by studying the parameter estimation shown in Figure 12.

The performance with respect to speed can be analysed by considering the number of variables. The total number of variables in the case of multiple shooting is $n_x \cdot (nk+1) + n_p = 61$ when $n_x = r_x = 10$ and only one parameter is estimated. It can be seen from the IPOPT analysis that the number of nonzeros in equality constraint Jacobian is 600, and the number of nonzeros in Lagrangian Hessian is 331. The sparsity of the Jacobian is therefore $1 - \frac{600}{61^2} \approx 84\%$. In the case of direct collocation where the number of variables is 211 and the number of nonzeros in the Jacobian is 1360, we end up with a sparsity of 97%.

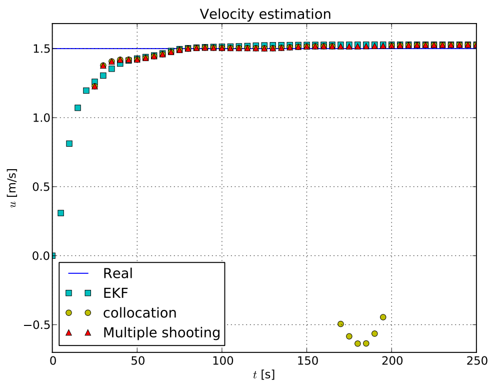
The time used to solve the MHE problem with multiple shooting was $503\mathrm{s}$ while the implementation with direct collocation took $87\mathrm{s}$. The simulations which were compared were the ones where five of the states are measured, and the direct collocation polynomial was of order three.



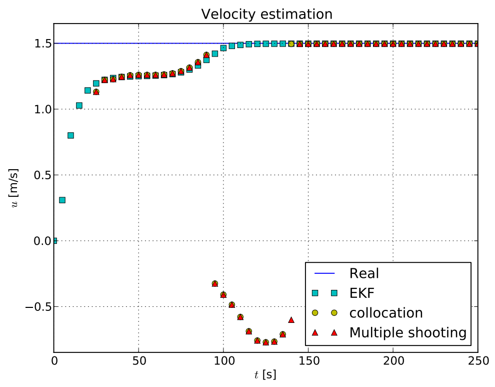
(a) Direct collocation with 1. order polynomial, five of the ten states are measured.



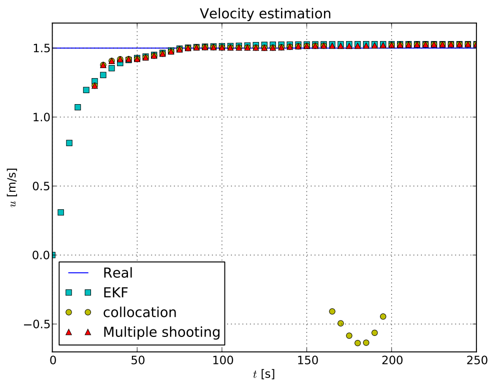
(b) Direct collocation with 1. order polynomial, two of the ten states are measured.



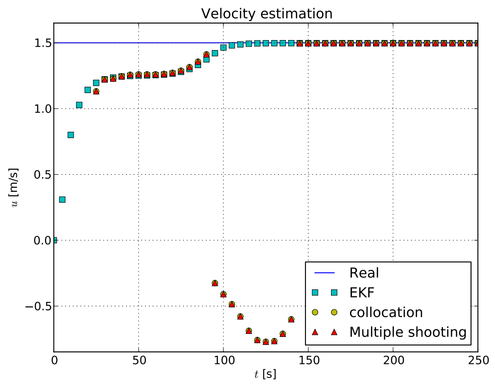
(c) Direct collocation with 3. order polynomial, five of the ten states are measured.



(d) Direct collocation with 3. order polynomial, two of the ten states are measured.



(e) Direct collocation with 5. order polynomial, five of the ten states are measured.

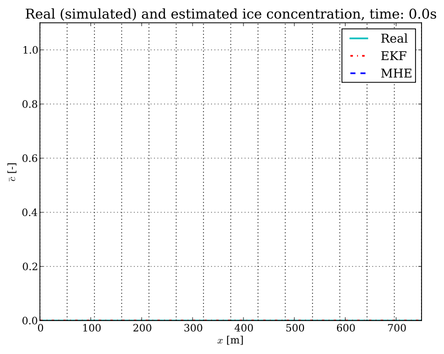


(f) Direct collocation with 5. order polynomial, two of the ten states are measured.

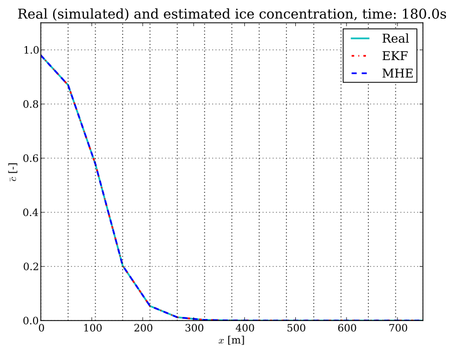
Figure 12: Comparison of the parameter estimation from MHE with direct collocation and multiple shooting, and the EKF.

Appendix C Additional plots

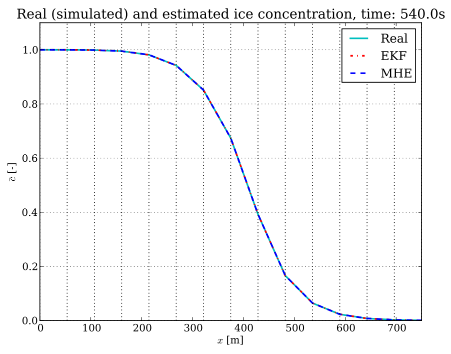
Some plots where left out from the results and put in the appendix in order to keep the result section easy to follow.



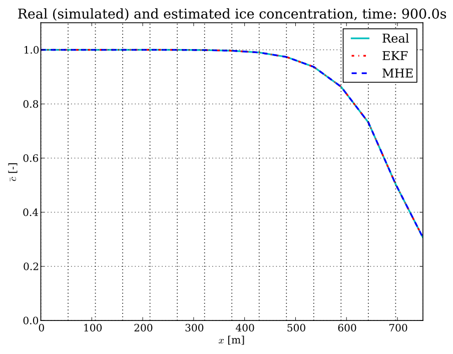
(a) Initial conditions



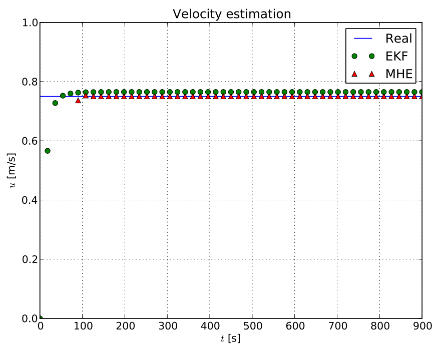
(b) The states after 180s



(c) The states after 540s



(d) The states at the end



(e) The estimated velocity

Figure 13: Simulation of parameter estimation where only velocity is estimated.

Bibliography

Andersson, J., Åkesson, J. and Diehl, M. (2012*a*), CasADi – A symbolic package for automatic differentiation and optimal control, *in* S. Forth, P. Hovland, E. Phipps, J. Utke and A. Walther, eds, 'Recent Advances in Algorithmic Differentiation', Vol. 87 of *Lecture Notes in Computational Science and Engineering*, Springer Berlin Heidelberg, pp. 297–307.

Andersson, J., Åkesson, J. and Diehl, M. (2012*b*), Dynamic optimization with casadi, *in* 'Decision and Control (CDC), 2012 IEEE 51st Annual Conference on', pp. 681–686.

Berkman, P. A., Young, O. R. et al. (2009), 'Governance and environmental change in the arctic ocean', *Science* **324**(5925), 339–340.

Biegler, L. T. (2010), *Nonlinear programming: concepts, algorithms, and applications to chemical processes*, Vol. 10, Society for Industrial and Applied Mathematics.

Borch, O., Westvik, M., Ehlers, S. and Berg, T. (2012), Sustainable arctic field and maritime operation, *in* 'OTC Arctic Technology Conference'.

Bressert, E. (2012), *SciPy and NumPy*, Oreilly and Associate Series, Oreilly & Associates Incorporated.

Chen, C. (1999), *Linear system theory and design*, The Oxford Series In Electrical And Computer Engineering, OXFORD University Press.

Dunbar, M. (1973), 'Commentary: Stability and fragility in arctic ecosystems', *Arctic* **26**(3), 179–185.

Egeland, O. and Gravdahl, J. T. (2002), *Modeling and simulation for automatic control*, Marine Cybernetics.

Eik, K. (2008), 'Review of experiences within ice and iceberg management', *Journal of Navigation* **61**(04), 557–572.

Eik, K. and Løset, S. (2009), Specifications for a subsurface ice intelligence system, *in* 'ASME 2009 28th International Conference on Ocean, Offshore and Arctic Engineering', ASME, pp. 103–109.

Fingas, M. and Hollebone, B. (2003), 'Review of behaviour of oil in freezing environments', *Marine Pollution Bulletin* **47**(9), 333–340.

Gautier, D. L., Bird, K. J., Charpentier, R. R., Grantz, A., Houseknecht, D. W., Klett, T. R., Moore, T. E., Pitman, J. K., Schenk, C. J., Schuenemeyer, J. H.

et al. (2009), 'Assessment of undiscovered oil and gas in the arctic', *Science* **324**(5931), 1175–1179.

Griewank, A. and Walther, A. (2008), *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, Other titles in applied mathematics, Society for Industrial and Applied Mathematics.

Hamilton, J., Holub, C., Blunt, J., Mitchell, D. and Kokkinis, T. (2011), Ice management for support of arctic floating operations, *in* 'OTC Arctic Technology Conference'.

Hamilton, J. M. (2011), 'The challenges of deep-water arctic development', *International Journal of Offshore and Polar Engineering* **21**(4), 241–247.

Haseltine, E. L. and Rawlings, J. B. (2005), 'Critical evaluation of extended kalman filtering and moving-horizon estimation', *Industrial & Engineering Chemistry Research* **44**(8), 2451–2460.

Haugen, J., Grøtli, E. I. and Imsland, L. (2012), State estimation of ice-thickness using mobile sensors, *in* 'Proc. of the IEEE Multi-Conference on Systems and Control', Dubrovnik, Croatia.

Haugen, J., Imsland, L., Løset, S. and Skjetne, R. (2011), Ice observer system for ice management operations, *in* 'Proc. 21st Int. Offshore (Ocean) and Polar Eng. Conf., June 19-24, 2011, Maui, Hawaii, USA', pp. 1120–1127.

Hibler III, W. (1979), 'A dynamic thermodynamic sea ice model', *Journal of Physical Oceanography* **9**(4), 815–846.

Hindmarsh, A. C., Brown, P. N., Grant, K. E., Lee, S. L., Serban, R., Shumaker, D. E. and Woodward, C. S. (2005), 'Sundials: Suite of nonlinear and differential/algebraic equation solvers', *ACM Trans. Math. Softw.* **31**(3), 363–396.

HSL (2013), *A collection of Fortran codes for large scale scientific computation*.
URL: *http://www.hsl.rl.ac.uk*

Hunter, J. D. (2007), 'Matplotlib: A 2D graphics environment', *Computing In Science & Engineering* **9**(3), 90–95.

Jensen, Ø. (2010), 'The imo guidelines for ships operating in arctic ice-covered waters: From voluntary to mandatory tool for navigation safety and environmental protection?'.

Johannessen, O., Alexandrov, V., Alexandrov, V., Frolov, I. and Bobylev, L. (2007), *Remote Sensing of Sea Ice in the Northern Sea Route: Studies and Applications*, Springer Praxis Books / Geophysical Sciences, Springer London, Limited.

Jones, E., Oliphant, T., Peterson, P. et al. (2001–), 'SciPy: Open source scientific tools for Python'.
URL: *http://www.scipy.org/*

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

KMB Arctic DP (2013).
URL: *http://www.marin.ntnu.no/arctic-dp*

Kurganov, A. and Levy, D. (2000), 'A third-order semidiscrete central scheme for conservation laws and convection-diffusion equations', *SIAM Journal on Scientific Computing* **22**(4), 1461–1488.

Kurganov, A. and Tadmor, E. (2000), 'New high-resolution central schemes for nonlinear conservation laws and convection-diffusion equations', *Journal of Computational Physics* **160**(1), 241 – 282.

Langtangen, H. (2012), *A primer on scientific programming with Python*, Texts in computational science and engineering, Springer.

Leppäranta, M. (2011), *The drift of sea ice*, Springerverlag Berlin Heidelberg.

Oliphant, T. E. (2007), 'Python for scientific computing', *Computing in Science Engineering* **9**(3), 10–20.

Qu, C. C. and Hahn, J. (2009), 'Computation of arrival cost for moving horizon estimation via unscented kalman filtering', *Journal of Process Control* **19**(2), 358–363.

Rao, C. V. and Rawlings, J. B. (2002), 'Constrained process monitoring: Moving-horizon approach', *AIChE Journal* **48**(1), 97–109.

Serban, R. and Hindmarsh, A. C. (2005), CVODES, the sensitivity-enabled ODE solver in SUNDIALS, *in* 'Proceedings of the 5th International Conference on Multibody Systems, Nonlinear Dynamics and Control, Long Beach, CA'.

Simon, D. (2006), *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*, Wiley.

Sui, D. and Johansen, T. (2011), Exponential stability of regularized moving horizon observer for n-detectable nonlinear systems, *in* 'Control and Automation (ICCA), 2011 9th IEEE International Conference on', pp. 806–811.

Tenny, M. and Rawlings, J. (2002), Efficient moving horizon estimation and nonlinear model predictive control, *in* 'American Control Conference, 2002. Proceedings of the 2002', Vol. 6, pp. 4475–4480 vol.6.

Timco, G., Gorman, B., Falkingham, J. and O'Connell, B. (2005), Scoping study: Ice information requirements for marine transportation of natural gas from the high arctic, Technical report, Canadian Hydraulics Centre, Ottawa, Ont., Canada.

Wächter, A. and Biegler, L. T. (2006), 'On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming', *Mathematical Programming* **106**(1), 25–57.

Zavala, V. and Biegler, L. (2009), Nonlinear programming strategies for state estimation and model predictive control, *in* L. Magni, D. Raimondo and F. Allgöwer, eds, 'Nonlinear Model Predictive Control', Vol. 384 of *Lecture Notes in Control and Information Sciences*, Springer Berlin Heidelberg, pp. 419–432.