Data Assimilation with a Machine Learned Observation Operator and Application to the Assimilation of Satellite data for Sea Ice Models

Siyang Jing

An undergraduate honors thesis submitted to the faculty of the department of mathematics of University of North Carolina at Chapel Hill.

Chapel Hill
2019

To be approved by:

Prof. Christopher Jones

Prof. David Adalsteinsson

Dr. Christian Sampson

ABSTRACT

Siyang Jing: Data assimilation with a machine learned observation operator
and application to the assimilation of satellite data for sea ice models
In collaboration with Dr. Christian Sampson
Under the direction of Prof. Christopher Jones

Data assimilation embodies a wide variety of techniques used to combine model output and real-world observations in an optimal way to estimate the true state of a system. Important to all data assimilation schemes are the model $\mathcal{M}$ used to evolve physical state variables forward in time, and the observation operator $\mathcal{H}$ used to map those state variables to observed quantities. Ideally, the observed quantities are the state variables themselves in which case $\mathcal{H}$ is simply a projection. However, in practice, this may not be the case. In many cases, the relationship between the physical state variables and the observed quantities can be very complex and highly nonlinear. An example is the case of passive microwave satellite observations of sea ice. Sea ice plays a vital role in the Earth's climate system and is a focus of both remote sensing and modeling efforts in modern times. Passive microwave radiometry provides a daily picture of the ice, despite persistent Arctic cloud cover, at a low resolution of 25km. While one cannot resolve many ice features in this data set, the concentration of ice in a given 25km pixel may be derived from the intensities of observed microwaves at various frequencies. Sea ice is far more emissive in the microwave spectrum than open water and that contrast can be exploited to estimate sea ice concentration. The emissivity of the ice depends on its temperature, bulk salinity, thickness, and its snow cover. Further, the microwaves must pass through the atmosphere producing noise in the observed signal. The map from sea ice state variables to observable microwave intensities is thus very difficult to model.

More empirical methods can obtain concentrations; the NASA TEAM 2 algorithm is an example. Given the frequent observations possible with passive microwave, the long 30-year record sea ice concentration derived from these observations is an enticing data set for assimilation into large-scale sea ice models. In addition, sea ice concentration is a sea ice state variable allowing for a simple projection type observation operator. However, in the summertime, sea ice concentration retrievals

can be inaccurate due to the presence of melt ponds, which are ponds that form atop the ice from melting snow. These ponds block the microwave signature from the ice below them making it appear as though there is less ice leading to an underestimation of sea ice concentration. Assimilation of this data could be detrimental. However, one could avoid the issue by instead using an observation operator which maps the sea ice state, which can include the ponds, directly to the satellite radiances. This way if the model is in line with the radiances themselves we avoid assimilation of incorrect data.

As stated, this relationship is complicated and computationally expensive to model. We propose to that end to machine learn the observation operator that takes ice state to satellite radiances. In this initial study, we use a simplified proxy model of sea ice that mimics ponding behavior as an experimental test bed. Using our model we generate a training data set from the state variables and non-injective functions which produce "observed radiances". We explore the amount of data needed to train the operator to obtain successful assimilations with an Ensemble Kalman Filter scheme. We compare our results to using retrieved concentration values which suffer from the pond masking effect. We find that with sufficient training data our machine learned observation operator leads to better assimilation than a projection operator using incorrectly inverted values.

ACKNOWLEDGEMENTS

This study is conducted in collaboration with Dr. Christian Sampson under the supervision of Prof. Christopher Jones. This study is part of the projects initiated on the AIM work shop "Climate Change and Resilience: Methods of Dynamical Systems and Data Assimilation" supported by NSF Grant #DMS-1722578. Thank the Mathematics and Climate Research Network (MCRN) and American Institute of Mathematics (AIM) for support. Thank Amit Apte for part of the EnKF code. Last but not least, thank Xinrui Ma of Xiamen University, Xiamen, China for her academic advice and her invaluable mental and emotional support. This work would not be possible without her encouragement in the first place.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

DA	Data Assimilation
EnKF	Ensemble Kalman Filter
EW09	The Sea Ice Model Proposed by Eisenman and Wettlaufer in 2009
FNN	Fully Connected Neural Network
i.i.d	Independent and identically distributed
KF	Kalman Filter
ML	Machine Learning
NN	Neural Network
OP	Observation Operator
ODE	Ordinary Differential Equation

LIST OF SYMBOLS

α	Albedo, the percentage of solar radiation reflected back
α_m	Maximum attainable albedo
$C(\mathbb{R}^n, \mathbb{R}^m)$	Continuous function from $\mathbb{R}^n$ to $\mathbb{R}^m$
$\boldsymbol{\epsilon}$	The uncertainty in model
$\boldsymbol{\eta}$	The uncertainty in observation operator
$\mathcal{H}$	Observation operator
$\mathbf{H}$	The linearization or matrix form of $\mathcal{H}$
$\mathcal{H}_{\text{ML}}$	Machine learning observation operator with realistic training data
$\hat{\mathcal{H}}_{\text{ML}}$	Benchmark machine learning observation operator with unrealistically large amount of training data
$\mathcal{H}_{\text{retrieval}}$	Observation operator representing the proxy for satellite retrieval algorithm
$\mathcal{H}_{5\%}$	Machine learning observation operator with 5% data in $E \in [-50, 50]$ for training.
$\mathcal{H}_{30\%}$	Machine learning observation operator with 30% data in $E \in [-50, 50]$ for training.
I_n	The identity mapping in $C(\mathbb{R}^n, \mathbb{R}^n)$
$\mathbf{K}$	The Kalman gain matrix
$\mathcal{M}$	The dynamic model used in data assimilation
$\mathbf{M}$	The linearization or matrix form of $\mathcal{M}$
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$	Normal distribution with mean $\boldsymbol{\mu}$ and error covariance matrix $\boldsymbol{\sigma}$.
$\mathbf{P}$	The covariance matrix representing the uncertainty in state estimate
$\mathbf{Q}$	The covariance matrix representing the uncertainty in model
$\mathbf{R}$	The covariance matrix representing the uncertainty in observation
$\boldsymbol{\xi}$	The error of machine learning observation operator.
$\mathbf{x}$	The state vector, in our study is 2-dimensional containing E and α_m.
$\mathbf{x}^f$	The forecast estimate of the state.
$\mathbf{x}^{f_i}$	The forecast estimate of the i-th ensemble member.
$\mathbf{x}^a$	The analysis estimate of the state.
$\mathbf{y}$	The observation vector, in our study is 5-dimensional representing the satellite radiances.
$\mathbf{y}^{f_i}$	The forecast estimate of the i-th ensemble member.

CHAPTER 1

Introduction

1.1 Data Assimilation

Data assimilation is an analysis technique in which the observed information is accumulated into the model state by taking advantage of consistency constraints with laws of time evolution and physical properties. In this study, we will focus on sequential data assimilation with its discrete time formulation.

We first consider the **model** that evolves the state variables $\mathbf{x}_t \in \mathbb{R}^n$ in time, which is typically a dynamical system defined by the map $\mathcal{M} \in C(\mathbb{R}^n \times \mathbb{N}, \mathbb{R}^n)$

$$\mathbf{x}_{t+1} = \mathcal{M}(\mathbf{x}_t, t) + \boldsymbol{\epsilon}_t, \quad t \in \mathbb{N} \tag{1.1}$$

where $\boldsymbol{\epsilon} = \{\boldsymbol{\epsilon}_t\}_{t\in\mathbb{N}}$ is an i.i.d. sequence. In many applications, models are supplemented by observations of the system as it evolves; this information then changes the probability distribution on the states, typically reducing the uncertainty. To describe such situations, we assume that we are given data, or observations, $\mathbf{y}_t \in \mathbb{R}^m$ defined as follows. At each discrete time instance, we observe a (possibly nonlinear, as in this study) function of the state variables, with additive noise:

$$\mathbf{y}_t = \mathcal{H}(\mathbf{x}_t) + \boldsymbol{\eta}_t, \quad t \in \mathbb{N} \tag{1.2}$$

where $\boldsymbol{\eta} = \{\boldsymbol{\eta}_t\}_{t\in\mathbb{N}}$ is an i.i.d. sequence. The function $\mathcal{H}$ is known as the **observation operator**. The objective of **data assimilation** is to determine information about the states $\mathbf{x}_t$, given data $\mathbf{y}_t$, at each time step t. For example, if we are directly observing all the state variables, then the observation operator is simply the identity mapping I_n. If we are instead observing a subset of all the state

variables, then the observation operator becomes a projection. As later described in Section 2.3, the observation operator in our study is much more complicated and non-linear.

Sequential DA consists of two steps at each discrete time, the prediction step, where the estimate from last time step is evolved using the model to give the **forecast**, and the update step, where the forecast is updated to the **analysis** reflecting the observations.

As described above, the observation operator is very important - it allows us to compare our forecast model states with measurements and obtain a more accurate estimate of the model state. Passive microwave satellite data is an example where the physical state of a system, ocean temperature, for example, is not directly observed, but the intensity of microwave radiation it emits is. In a situation like this, we have two choices for an observation operator.

1. Infer the physical quantities from the light intensities with an inverse scheme. (Often done, but can be inaccurate) and use an observation operator which is a projection on to the inferred state variables.

2. Use an observation operator which maps the model state variable directly to the actually observed quantities.

In cases where the inversion option 1 is not a good choice as incorrect data would be assimilated pushing the model in the wrong direction. In this situation, it is advantageous to use option 2. However, option 2 presents its own difficulties, $\mathcal{H}$ may be highly nonlinear and complex, as is the case for satellite radiance measurements. In this work, we explore methods for discovering $\mathcal{H}$ using data and machine learning with a focus on sea ice modeling.

1.1.1 Kalman Filter

From the beginning of data assimilation, many algorithms that aim to produce the optimal state estimate under various constraints and assumptions have been proposed. Some popular methods include 3D-Var, 4D-Var, particle filter, and Kalman filter along with its variants. In this study, we focus on the Ensemble Kalman Filter. First, let us review the Kalman filter. Named after Rudolf E. Kalman, the Kalman filter first emerged in control theory. It uses a system's dynamic model (e.g., physical laws of motion), known control inputs to that system, and multiple sequential observations, such as from sensors, to form an estimate of the system's state that is better than the estimate obtained

by using only one measurement alone. In terms of data assimilation, the standard Kalman filter (KF) is the optimal sequential data assimilation method for linear dynamics and observations with Gaussian error, i.e., we assume in Equation (1.1) and Equation (1.2) $\mathcal{M}$ and $\mathcal{H}$ are linear mappings and $\boldsymbol{\epsilon}_t$ and $\boldsymbol{\eta}_t$ at each time step are normal distribution with zero mean. Usually, the problem which KF applies to can be formulated as follows:

- $\mathbf{x}^b, \quad \mathbf{B}$, prior guess of initial state vector and error covariance matrix
- $\mathbf{x}_{t+1} = \mathcal{M}(\mathbf{x}_t) + \boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{Q}_t)$, where $\mathcal{M}$ is the dynamic model that evolves the state in time and $\mathbf{Q}_t$ is the error covariance matrix for the model. In addition, we have to know the matrix form (linearization in the case of nonlinear dynamics) of $\mathcal{M}$: $\mathbf{M}$.
- $\mathbf{y}_t = \mathcal{H}(\mathbf{x}_t) + \boldsymbol{\eta}_t, \quad \boldsymbol{\eta}_t \sim \mathcal{N}(0, \mathbf{R}_t)$, where $\mathcal{H}$ is the observation operator that maps system states to observations, and $\mathbf{R}_t$ is the error covariance matrix for the observation operator. Similarly, the matrix form (linearization in the case of nonlinear observation operator) of $\mathcal{H}$: $\mathbf{H}$.

The algorithm goes as follows:

1. Set $\mathbf{x}_0^a = \mathbf{x}^b$ and $\mathbf{P}_0^a = \mathbf{B}$
2. Forecast step:
   - Propagate the prior guess forward in time, $\mathbf{x}_{t+1}^f = \mathcal{M}(\mathbf{x}_t^a)$
   - Propagate the prior guess error covariance matrix forward in time: $\mathbf{P}_{t+1}^f = \mathbf{M}\mathbf{P}_t^a\mathbf{M}^T + \mathbf{Q}$
3. Analysis step:
   - Calculate the Kalman gain matrix: $\mathbf{K} = \mathbf{P}^f\mathbf{H}^T(\mathbf{R} + \mathbf{H}\mathbf{P}^f\mathbf{H}^T)^{-1}$
   - Get the analysis of the state using the observations: $\mathbf{x}^a = \mathbf{x}^f + \mathbf{K}(\mathbf{y} - \mathcal{H}(\mathbf{x}^f)$
   - Calculate the analysis error covariance matrix:$\mathbf{P}^a = (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{P}^f$
4. Repeat steps 2 and 3 until all observations have been assimilated.

1.1.2 Ensemble Kalman Filter

As stated above, the standard Kalman filter (KF) only gives the optimal state estimate for linear dynamics and measurement processes with Gaussian error statistics. For non-linear situations, the extended Kalman filter (EKF) can be used with the linearization of $\mathcal{M}$ and $\mathcal{H}$, although it is well known that EKF is unstable under strong nonlinearities. Both the KF and the EKF explicitly propagate error information with a dynamic equation for the state error covariance matrix. However, the integration of this equation is not computationally feasible for large-scale environmental systems. To overcome these limitations, the ensemble Kalman filter (EnKF) was proposed by (Evensen, 2003). The ensemble Kalman filter is a suboptimal estimator, where the error statistics are predicted by using a Monte Carlo or ensemble integration to solve the Fokker-Planck equation. In this study, we use the formulation of (Gillijns et al., 2006). Like the Kalman filter, EnKF also consists of the forecast step and the analysis step.

First, to represent the error statistics in the forecast step, we assume that at time t, we have an ensemble of q forecasted state estimates with random sample errors. We denote this ensemble as $\mathbf{X}_t^f \in \mathbb{R}^{n\times q}$, where

$$\mathbf{X}_t^f = (\mathbf{x}_t^{f_1}, ..., \mathbf{x}_t^{f_q}) \tag{1.3}$$

and the superscript f_i refers to the i-th forecast ensemble member. Then, the ensemble mean $\bar{\mathbf{x}}_t^f \in \mathbb{R}^n$ is defined by

$$\bar{\mathbf{x}}_t^f = \frac{1}{q}\sum_{i=1}^{q} \mathbf{x}_t^{f_i} \tag{1.4}$$

From the ensemble members, we calculate the forecast output and the forecast output mean

$$\mathbf{y}_t^{f_i} = \mathcal{H}(\mathbf{x}_t^{f_i}) \tag{1.5}$$

$$\bar{\mathbf{y}}_t^f = \frac{1}{q}\sum_{i=1}^{q} \mathbf{y}_t^{f_i} \tag{1.6}$$

We define the ensemble error matrix $\mathbf{E}_t^f \in \mathbb{R}^{n\times q}$ around the ensemble mean by

$$\mathbf{E}_t^f = (\mathbf{x}_t^{f_1} - \bar{\mathbf{x}}_t^f, ..., \mathbf{x}_t^{f_q}) - \bar{\mathbf{x}}_t^f \tag{1.7}$$

and the ensemble of output error $\mathbf{E}_{\mathbf{y}_t}^f \in \mathbb{R}^{n \times q}$

$$\mathbf{E}_{\mathbf{y}_t}^f = (\mathbf{y}_t^{f_1} - \bar{\mathbf{y}}_t^f, ..., \mathbf{y}_t^{f_q}) - \bar{\mathbf{y}}_t^f \tag{1.8}$$

The forecast of the error covariance in the state is estimated with the ensemble error covariance, i.e.

$$\mathbf{P}_t^f = \frac{1}{q-1}\mathbf{E}_t^f(\mathbf{E}_t^f)^T \tag{1.9}$$

We also define

$$\mathbf{P}_{\mathbf{x}\mathbf{y}_t}^f = \frac{1}{q-1}\mathbf{E}_t^f(\mathbf{E}_{\mathbf{y}_t}^f)^T \tag{1.10}$$

$$\mathbf{P}_{\mathbf{y}\mathbf{y}_t}^f = \frac{1}{q-1}\mathbf{E}_{\mathbf{y}_t}^f(\mathbf{E}_{\mathbf{y}_t}^f)^T \tag{1.11}$$

We interpret the forecast ensemble mean $\bar{\mathbf{x}}_t^f$ as the best forecast estimate of the state, and the sample covariance of the ensemble members $\mathbf{P}_t^f$ as the forecast estimate for the uncertainty.

The second step is the analysis step. The Kalman gain matrix is estimated by

$$\mathbf{K}_t = \mathbf{P}_{\mathbf{x}\mathbf{y}_t}^f(\mathbf{P}_{\mathbf{y}\mathbf{y}_t}^f)^{-1} \tag{1.12}$$

To obtain the analysis estimates of the state, the EnKF performs an ensemble of parallel data assimilation cycles, where for $i = 1, ..., q$

$$\mathbf{x}_t^{a_i} = \mathbf{x}_t^{f_i} + \mathbf{K}_t(\mathbf{y}_t^i - \mathcal{H}(\mathbf{x}_t^{f_i}) \tag{1.13}$$

where $\mathbf{y}_t^i$ are perturbed observations given by

$$\mathbf{y}_t^i = \mathbf{y}_t + \boldsymbol{\eta}_t, \quad \boldsymbol{\eta}_t \sim \mathcal{N}(0, \mathbf{R}_t) \tag{1.14}$$

The ensemble mean and covariance are calculated as follows

$$\bar{\mathbf{x}}_t^a = \frac{1}{q}\sum_{i=1}^{q}\mathbf{x}_t^{a_i}) \tag{1.15}$$

$$\mathbf{P}_t^a = \frac{1}{q-1}\mathbf{E}_t^a(\mathbf{E}_t^a)^T \tag{1.16}$$

and are used as the best analysis estimate of the state and uncertainty, respectively. The last step is the prediction of error statistics in the next forecast step:

$$\mathbf{x}_{t+1}^{f_i} = \mathcal{M}(\mathbf{x}_t^{a_i}) + \boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \mathbf{Q}_t) \tag{1.17}$$

Unlike the extended Kalman filter, the evaluation of the filter gain $\mathbf{K}_t$ in the EnKF does not involve an approximation of the nonlinearity $\mathcal{M}$ and $\mathcal{H}$. Hence, the computational burden of evaluating the Jacobians of $\mathcal{M}$ and $\mathcal{H}$ is absent in the EnKF.

As we can see from the formulation of EnKF, it is a sub-optimal estimation derived from Kalman filter to compensate for high computational cost and nonlinearity. Like the Kalman filter, EnKF would still perform better under the same constraints and assumptions of Kalman filter. Typically, the closer the actual system is to linear and Gaussian, the more likely EnKF is useful.

However, when there is a systematic error or bias in our knowledge about the observation operator $\mathcal{H}$, wrong information might be assimilated into the system and the analysis could be even worse than the forecast. If the true observation operator is $\mathcal{H}$, but we think it is instead $\mathcal{H}'$, when calculating the output forecast $\mathbf{y}_t^{f_i}$ in Equation (1.1.2), we would use $\mathcal{H}'$ instead of the correct $\mathcal{H}$. Intuitively, even if the forecast ensemble states $\mathbf{x}_t^{f_i}$ are exactly the same as the truth, forecast output of the ensemble members $\mathbf{y}_t^{f_i} = \mathcal{H}'(\mathbf{x}_t^{f_i})$ would be far from the observation which is generated by $\mathcal{H}$ instead of $\mathcal{H}'$, and the update step based on such difference would likely drive the model farther away from the truth. Since the calculation of the Kalman gain matrix Equation (1.12) is based on the output forecast through Equation (1.1.2) and Equation (1.1.2), the algorithm would likely to give an estimate of the Kalman gain matrix that is far from the truth. Furthermore, in Equation (1.13), each ensemble member is updated with the Kalman gain and the difference between the observation and forecast output, where we again would put the wrong $\mathcal{H}'$ in the equation. In our study, this is exactly what happens in a sea ice concentration retrieval algorithm when melt ponds obscure the microwave signature of the ice underneath.

1.2 Sea Ice Modeling and Data Assimilation

1.2.1 Numerical Sea Ice Models

A complete sea ice model consists of a momentum equation, thermodynamic model, and an equation governing the evolution of the ice thickness distribution. Here we give a brief review of the standard equations used to model sea ice on large scales (Hunke et al., 2017). The typical form of the momentum equation for sea ice is given by,

$$\rho\bar{h}\frac{d\mathbf{v}}{dt} = \nabla\cdot(\bar{h}\sigma) + \tau_{\mathbf{a}} + \tau_{\mathbf{w}} - \mathbf{f_c} - \rho\bar{h}g_a\nabla H. \tag{1.18}$$

Here $\mathbf{v}$ is the ice velocity, σ the stress, $\tau_{\mathbf{a}}$ atmospheric forcing, $\tau_{\mathbf{w}}$ ocean forcing, $\mathbf{f_c}$ the Coriolis force, H a sea surface tilt term, ρ the ice density, $\bar{h}$ the average ice thickness, and g_a gravitational acceleration. In addition to a thermodynamic model, the ice thickness distribution, g, are tracked and evolved according to the ice thickness distribution equation (Thorndike et al., 1975) given by,

$$\frac{dg}{dt} + (\nabla\cdot\mathbf{v})g + \frac{(fg)}{h} = \psi. \tag{1.19}$$

The first two terms in equation 1.19 describe horizontal transport and the changes in ice thicknesses due to flow, $f = dh/dt$ is the rate that thickness changes due to thermodynamic processes and ψ a term accounting for mechanical redistribution due to ridging, where convergence causes ice to pile up. Typically discrete thickness categories are tracked within a grid cell of the model and measures of observable quantities, such as ice concentration, can be calculated by looking at the amount of ice in the zero thickness categories. In this work, we focus on sea ice concentration as a quantity of interest for assimilation into a large scale sea ice model.

1.2.2 Sea Ice Concentration Retrieval

Sea ice concentrations themselves are derived by exploiting the high contrast in the microwave emissivities of sea ice and open water. The satellite itself retrieves the brightness temperature T_B. The brightness temperature can be approximated, in the microwave regime, through the dimensionless emissivity ε_s of the emitting material in the scene (the part of the Earth's surface seen by the satellite)

with the equation

$$T_B(\lambda, p) = \varepsilon_s(\lambda, p) T_s. \tag{1.20}$$

Here λ is the wavelength, p the polarization, and T_s is the physical temperature in degrees Kelvin. The emissivity itself depends on the dielectric properties of the emitting material. For sea ice, this is primarily a function of the salinity and brine volume fraction while for snow it is the grain size and water content. Further, the atmosphere must also be considered since the brightness temperature retrieved by the satellite is a combination of several contributions from the Earth, atmosphere, and space.

Once atmospheric and other effects are accounted for, the retrieved brightness temperature can be thought of as coming from a mix of open water, first year, and multiyear sea ice. The assumption is that the measured temperature brightness is coming from a linear mix of the constituents. That is, with F being the fraction,

$$T_B(\lambda, p) = \varepsilon_s(\lambda, p) T_s = \varepsilon_w(\lambda, p) T_w F_w + \varepsilon_{fy}(\lambda, p) T_{fy} F_{fy} + \varepsilon_{my} T_{my}(\lambda, p) F_{my} \tag{1.21}$$

where s,w,my,fy stand for the scene, water, multiyear and first-year ice respectively.

For many of the established algorithms, retrieval of ice concentration depends on tie points which are observed brightness temperatures for open water T_{Bw}, first-year ice T_{Bfy}, and multiyear ice T_{Bmy}. The tie points themselves incorporate most atmospheric effects and can depend on location, the season, and the sensor itself. The successful retrieval of ice concentration relies heavily on having correct tie points. In the wintertime, these do not vary much. However, in the summer, observed brightness temperatures can vary greatly both spatially and temporally (Willmes et al., 2014).

As the melt season progresses, higher water content in the snow layer atop the ice increases the microwave emissivity of the snow while absorbing most of the emissions from the ice below. The snow itself can have an emissivity close to 1 compared to that of multiyear ice at 0.7-0.8 or first-year ice at 0.9-0.95. This can have the effect of increasing the overall emissivity of a given scene. As the snow melt progresses, water runoff settles into the topographic low points of the surface forming melt ponds. These ponds continue to grow, deepen and eventually connect to form complex geometries throughout the melt season. They can also drain through the porous microstructure of the sea ice

when the ice becomes sufficiently permeable. Melt ponds have a microwave signature very close to that of open water at the relevant frequencies and also absorb most of the emission from the ice below. This has the opposite effect of wet snow, decreasing the overall emissivity. The increased emissivity of the wet snow is likely to push toward a concentration retrieval on the high side, whereas melt ponds tend to push toward lower values (Kern et al., 2016; Kongoli et al., 2011). The extent of these effects on sea ice concentration retrieval depends on the tie points of the algorithm used. Many algorithms are tuned to mitigate the masking effect of melt ponds in the summer to provide concentrations that more closely represent the extent of sea ice. These values are important for heat fluxes as the heat flux of sea ice and the ocean differs significantly.

One might also be interested in albedo. In this case, one would want a concentration that more closely represents the surface fraction of exposed sea ice. This is because the critical factor here is the difference between the albedo of melt ponds and the ice. For this purpose, one may not want to mitigate the masking effect of melt ponds at all. As an example, in (Kern et al., 2016) it was observed that for 100% sea ice concentration with 40% melt pond coverage, many of the standard concentration retrieval algorithms returned concentrations of $\approx 90\%$. While still an underestimation, this value is a far cry from the 60% sea ice surface fraction an algorithm not tuned to account for melt ponds might produce. Further, for an algorithm that is tuned high, the large spatial and temporal variability of surface conditions in summer can lead to concentration retrievals of $> 100\%$. In addition, ponds can rapidly drain through the porous microstructure of the ice when it is warm enough, this means a retrieval algorithm tuned high will be over predicting concentrations in situations where ponds have disappeared.

Attempts to quantify this effect were carried out in (Kern et al., 2016). Linear correlations were found in a comparison of sea ice concentrations obtained using NASA's MODIS sensor, which can resolve melt ponds, and NASA's AMSR-E sensor. The slopes ranged from 0.9 to 1.12 , depending on the algorithm, indicating that some algorithms are underestimating concentration while others are overestimating it. These errors have been found to be as high at 26% when compared to data obtained using NASA's Moderate Resolution Imaging Spectroradiometer (MODIS) sensor (Kern et al., 2016). This can create serious problems when attempting to assimilate concentration data into large scale sea ice models. Many large scale sea ice models also include pond fraction as a state variable, if the model output was suggesting say 100% ice with 30% pond coverage and is close to

the true values, one would not want to assimilate a retrieved concentration of 70% ice, the amount hidden by the ponds.

1.2.3 Assimilating with the satellite radiances

Given that 100% ice and 30% ponds looks the same to a passive microwave satellite as 70% ice and no ponds, in that observed radiances are very similar, one possible way to avoid the assimilation of incorrect data is to change the observation operator and map the state variables of the model to the radiances instead of obtaining an (inaccurate) estimate of state variables from them. This itself is a challenging task and the radiances emitted by a scene of emissivity will depend not only on the fraction of ice, water, and ponds, but on the thickness, salinity, temperature and snow cover of the ice. Perhaps the most difficult parameter to estimate is sea ice emissivity. Sea ice may be viewed as a composite of ice and brine, and the microwave emissivity depends heavily on the volume fraction and configuration of the brine pore space, a problem which relates to classic problems in homogenization theory. Further atmospheric effects need to be taken into consideration as it serves as an intermediate layer between the ice pack and observing satellite.

One method previously studied uses various parameterizations of sea ice emissivities and an atmospheric radiative transfer model to build a complex observation operator which takes sea ice model state variables directly to passive microwave satellite radiances (Scott et al., 2012). It was found that this improved model predictions, especially during the melt season. However, it is sensitive to the parameterization of emissivity chosen and computationally expensive. In this work a 3D-var assimilation scheme was used. However, the Ensemble Kalman Filter (EnKF) has many advantages over 3D-Var but due to the needed atmospheric modeling would be computationally prohibitive. We instead propose to use a machine-learned observation operator which takes inputs of sea ice state variables and rough atmospheric conditions and outputs likely observed satellite radiances. Once learned, the low computational cost of the observation operator would allow for an EnKF scheme to be possible. The primary difficulty here, however, is in building a data set to train on. As a first step, we will use a simple Testbed Model designed to mimic how ponds form and drain during the melt season with proxies for satellite radiance, true ice concentration, pond fraction, and retrieved concentrations to investigate how successful a machine learning observation operator is in improving predictions compared to retrieved concentrations.

1.3 Machine Learning

Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use to effectively perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subfield of artificial intelligence. Machine learning algorithms build a mathematical model of sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task. Machine learning algorithms are used in a wide variety of applications, such as email filtering, and computer vision, where it is infeasible to develop an algorithm of specific instructions for performing the task. Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning.

1.3.1 Neural Networks

Neural networks (NNs) are computing systems vaguely inspired by the biological neural networks that constitute animal brains. The neural network itself is not an algorithm, but rather a framework for many different machine learning algorithms to work together and process complex data inputs. Such systems "learn" to perform tasks by considering examples, generally without being programmed with any task-specific rules.

An ANN is a model based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit information, a "signal", from one artificial neuron to another. An artificial neuron that receives a signal can process it and then signal additional artificial neurons connected to it. In common ANN implementations, the signal at a connection between artificial neurons is a real number, and the output of each artificial neuron is computed by some non-linear function of the sum of its inputs. Artificial neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Typically, artificial neurons are aggregated into layers. Different layers may perform different kinds of transformations on their inputs. Signals travel from the first layer (the input layer) to the last layer (the output layer), possibly after traversing the layers multiple times. In our study, a single layer of

the neural network can be summarized as

$$\boldsymbol{x}_{l+1} = f_l(\boldsymbol{W}_l \boldsymbol{x}_l + \boldsymbol{b}_l) \tag{1.22}$$

where f_l is a element-wise nonlinear function called the "activation function". We may have different activation functions for different purposes in different layers. For example, in our study, the hidden layers all have $\tanh$ as the activation function while the final output layer does not have activation function, i.e., f_{out} is the identity mapping.

1.4 Rationale for Testbed Model

As mentioned in Section 1.2.3, the motivation of this study is to investigate how successful a machine learning observation operator is in improving predictions compared to retrieved concentrations. Typically, sea ice dynamics are simulated with the numerical sea ice models described in Section 1.2.1 e.g., the Community Sea Ice Model (CSIM) (Hunke and Lipscomb, 2010). However, use of such large-scale sea ice models in this study, while describable, would act as a bottleneck preventing timely exploration of the state space and its relationship to satellite radiance values. This is partly because of the computational cost in running the large-scale models, especially in EnKF, where a large ensemble is needed, and the dynamics of every ensemble member needs to be calculated. Also, the state space in a large-scale sea ice model is usually high-dimensional. Here we wish to be able to fully explore the state space, and so we will require a low-dimensional model.

Besides a low-dimensional model, we also need to define proxies for satellite radiances, ice concentration, and pond concentration from the model state variables. While real observations exist, for this initial study, we focus on exploring how machine learning can be used to discover an observation operator. Therefore we desire to have complete control over the state variables, the observations, and the error in each and as such real ice observations would not be appropriate for use at present.

In data assimilation, Lorenz'96 (Lorenz, 1995) is often used as a model problem. It is a 40-dimensional dynamical system formulated by Edward Lorenz in 1996, originally used to study issues in the forecasting of spatially extended chaotic systems such as the atmosphere. We achieved initial

success with Lorenz'96. However, it is not sufficient to make our argument. In this work, we are motivated by the sea ice concentration retrieval problem caused by melt ponds, so we seek a low dimensional model which mimics the specific situation of interest. For example, the formation and drainage of ponds cause errors in retrieved concentrations which would not otherwise be present, and the model should display behaviors representing such phenomenon.

To this end, we use the ODE energy balance model of (Eisenman and Wettlaufer, 2009). This model accounts for energy input and loss in sea ice and as a result, can give us an accounting of how and when a pond should form or drain. However, this model does not explicitly have a variable related to the formation of melt ponds. Therefore, we will introduce a second ODE and variable to account for this. Since we will only have two variables, we will be able to easily explore the whole state space by simply generating model runs of different initial conditions. This will further allow us to create a proxy for satellite radiances, ice concentration, and pond fraction based on a simple input space. The details of this model are presented in Chapter 2. With this model, we are able to focus on the machine learning aspect of our study as we have control over all other variables.

CHAPTER 2

Testbed Model Formulation

2.1 Original Model of Eisenman and Wettlaufer

As mentioned in Section 1.2.3, we build a Testbed Model to mimic the physical process of ponds forming and draining form to investigate how successful a machine learning observation operator is in improving predictions compared to retrieved concentrations. We base our model on a simple sea ice model developed by I. Eisenman and J.S. Wettlaufer in 2009 (EW09). The original ODE of (Eisenman and Wettlaufer, 2009) is given by:

$$\frac{dE}{dt} = [1 - \alpha(E, \alpha_m)]F_s(t) - F_0(t) + F_{co_2} - F_T(t)\frac{E}{c_{ml}H_{ml}} + F_B \tag{2.1}$$

$$\text{where} \quad \alpha(E, \alpha_m) = \frac{\alpha_{ml} + \alpha_m}{2} + \frac{\alpha_{ml} - \alpha_m}{2} \tanh\left(\frac{E}{L_i h_c}\right) \tag{2.2}$$

In Equation (2.1), $F_s(t)$ is the incoming solar radiation, $F_0(t)$ is the amount of longwave radiation(heat) that escapes to space, F_{CO_2} is the amount of longwave radiation reflected back from clouds and CO_2, $F_T(t)$ has to do with heat exchange with lower latitudes, and F_B is the heat input from the ocean below the ice. In the original model, F_{CO_2} and F_B are constants, while $F_s(t)$, $F_0(t)$, and $F_T(t)$ are taken to be step-wise constant functions consisting of the monthly averaged values. In our study, the time-dependent quantities are replaced with continuous functions consisting of 10-term Fourier series fitted to the monthly averaged values. Meanwhile, we are most interested in F_{CO_2}, and we will vary its value to investigate its influence on the system.

In the model, Equation (2.2) represents the albedo (the percent of incoming solar radiation the ice reflects) and depends on the energy of the ice as a whole. In this equation α_{ml} is the albedo of the ocean mixed layer, L_i the latent heat of fusion for ice, and h_c a chosen characteristic ice

thickness which is used to control the smoothness of the parameterization. In the original model, α_m is the maximum attainable albedo of the ice which they take to be 0.68. In reality, it can be as high as 0.8 with snow on top of it in cold conditions. As E increases α goes down, this serves as a way to account for the ice albedo feedback caused by the formation of melt ponds. However, this does not take into account pond drainage. As sea ice warms, it becomes increasingly permeable to fluid flow. Large connected pathways form in the ice called brine channels. When these connect up enough, ponds can drain through the porous microstructure of the ice into the ocean. This happens at temperatures higher than when the ponds initially form. With the dark-colored ponds removed, the albedo of the ice is temporally increased and more solar radiation reflected. In this work, we are interested in having a model which mimics, in some way, ponding on sea ice. To approximate this behavior, we will allow the maximum attainable albedo to vary with E in a way that roughly mimics the effect of pond drainage. We will also use the values of E and α_m to "measure" pond and ice fractions. This is discussed below.

2.2 Proposed Model

The primary argument for making α_m change in time will be the following, in very cold conditions the maximum attainable albedo of the surface should tend toward 0.8 with snow fall and other processes keeping the albedo high. When the ice is in warmer conditions, like melting, the maximum attainable albedo should tend to lower values, e.g. 0.2, the albedo of open water, and there should be a competition of values. Initially, as the ice begins to pond the albedo drives down pretty quickly, however as the ice temperature increases the ice becomes permeable and the melt ponds drain out, which then causes the albedo of the ice to recover quickly. In order to mimic this physical process, we model the rates of change of the maximum attainable albedo α_m with a Filippov system (F. Filippov, 1960) consisting of two parts separated by a smooth boundary $H(E, \alpha_m) = 0$.

$$\frac{dE}{dt} = [1-\alpha(E,\alpha_m)]F_s(t) - F_0(t) + F_{co_2} - F_T(t)\frac{E}{c_{ml}H_{ml}} + F_B$$
$$\frac{d\alpha_m}{dt} = \frac{E^2}{K_1^2}\alpha_m\left(1-\frac{\alpha_m}{0.8}\right) + \frac{K_2^2}{1+E^2}\alpha_m\left(1-\frac{\alpha_m}{0.6}\right)$$
$$\text{in} \quad S_1 = \{(E,\alpha_m) : H(E,\alpha_m) > 0\} \tag{2.3}$$

$$\frac{dE}{dt} = [1-\alpha(E,\alpha_m)]F_s(t) - F_0(t) + F_{co_2} - F_T(t)\frac{E}{c_{ml}H_{ml}} + F_B$$
$$\frac{d\alpha_m}{dt} = \frac{K_3^2}{1+E^2}\alpha_m\left(1-\frac{\alpha_m}{0.6}\right) + \frac{E^2}{K_4^2}\alpha_m\left(1-\frac{\alpha_m}{0.2}\right)$$
$$\text{in} \quad S_2 = \{(E,\alpha_m) : H(E,\alpha_m) < 0\} \tag{2.4}$$

$$\text{where} \quad H(E,\alpha_m) = \alpha(E,\alpha_m) - 0.6. \tag{2.5}$$

In our proposed model, the dynamics for the energy E is the same as the original EW09 model. α_m is modeled by separate equations representing the two different situations. In "cold" conditions, α_m is modeled by competing logistic models with the carrying capacity to be 0.8 and 0.6 Equation (2.3). In "warm" conditions, α_m is similarly modeled by competing models with carrying capacities of 0.2 and 0.6 Equation (2.4). We also let the growth or decay rate in these equations depend on the energy E and some scaling factor K_i's. The two parts of the dynamics are separated by the discontinuity boundary $H = 0$, where H is defined by the smooth function Equation (2.5). Since the boundary is supposed to divide the state space into "cold" and "warm" parts, therefore we choose the boundary $H = 0$ to simply be where the albedo of the system $\alpha(E,\alpha_m)$ crosses the $\alpha = 0.6$ threshold. Figure 2.2 is a plot of a state space trajectory of the system, with initial condition $E = -200$, $\alpha_m = 0.7$ and parameter value $F_{\mathrm{CO}_2} = 60$. It gives an example of how the system behaves.

- In cold conditions, the system is at the left top part of the state space shown in Figure 2.2, where $\alpha(E,\alpha_m) > 0.6$. When the energy is largely negative, the logistic model term with the energy in the numerator dominates in Equation (2.3), and the maximum attainable albedo α_m rapidly approaches the carrying capacity of this term, 0.8, representing the effect of snow

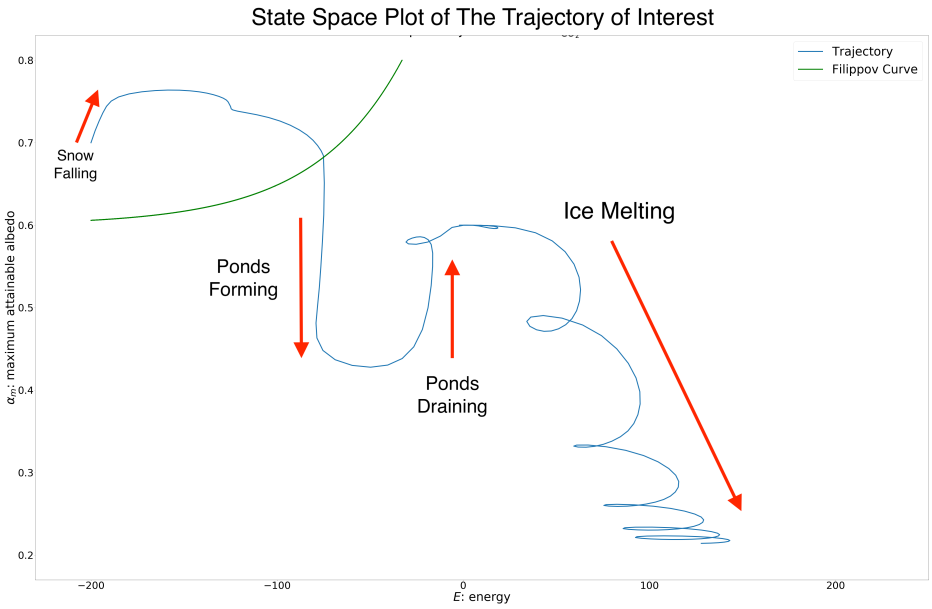

Figure 2.1: The state space trajectory that we use in data assimilation experiments.

falling. The initial rise from 0.7 towards 0.8 in Figure 2.2 illustrates to this process. When the energy is close to zero, the term with the energy in the denominator (with the addition of 1 to avoid singularities at $E = 0$) dominates, and α_m approaches 0.6, which is not apparent from Figure 2.2. As the system's energy keeps increasing, it crosses the boundary between the two parts of dynamics.

- In warm conditions, the system is at the center part of the state space shown in Figure 2.2, where $\alpha(E, \alpha_m) < 0.6$. The rate that α_m approaches 0.2 is faster when the energy is away from 0 whereas the rate to approach 0.6 is faster when the energy is near 0. As a result, we take the energy to be in the denominator for the logistic model with a carrying capacity of 0.6 and the energy in the numerator for the logistic model with a carrying capacity of 0.2 Equation (2.4). In Figure 2.2, we can see that right after the system crosses the boundary, α_m is rapidly driven towards 0.2, representing the process of ponding. When energy is higher, the ice becomes permeable, and melt pond drainage exposes the surface of the ice again, causing α_m to increase back to 0.6 .

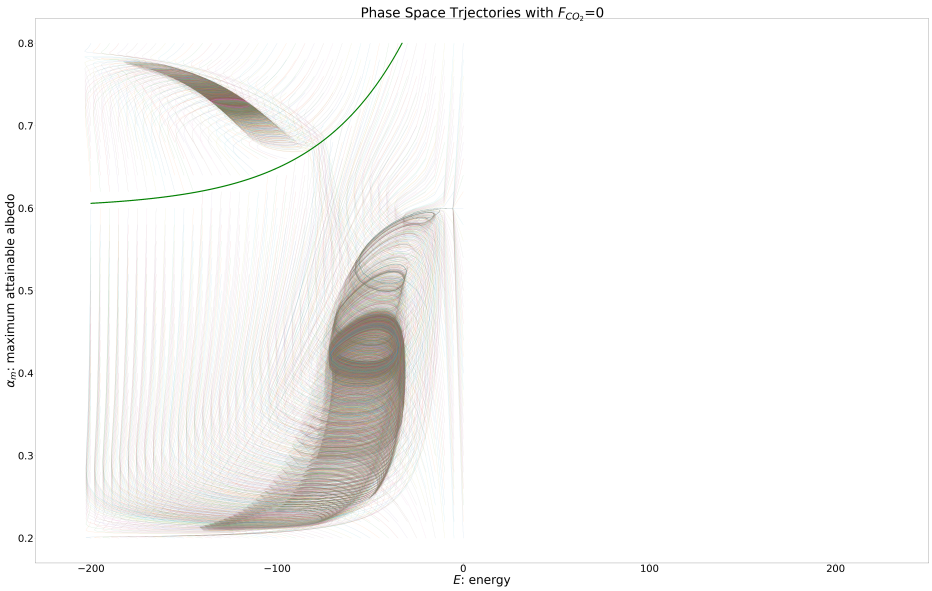

Figure 2.2: Trajectories with various initial points when $F_{\mathrm{CO}_2} = 0$.

Figure 2.2 and Figure 2.2 plot the boundary curve and trajectories from various initial points in the state space with different values of F_{CO_2}, respectively. When the value of F_{CO_2} is small, as in Figure 2.2 where $F_{\mathrm{CO}_2} = 0$, the energy of the system stays negative, whereas when F_{CO_2} is sufficiently large, as in Figure 2.2 where $F_{\mathrm{CO}_2} = 60$, the states that start with negative energy will cross zero and become positive. In our data assimilation experiments, we will focus on one particular trajectory of interest, the one with initial condition $E = -200$, $\alpha_m = 0.7$ and parameter value $F_{\mathrm{CO}_2} = 60$. The aforementioned Figure 2.2 plots this trajectory along with the boundary curve separating the two parts of the dynamical system.

In terms of discrete time sequential data assimilation, the state variable is a two dimensional vector defined by

$$\mathbf{x} = (E, \alpha_m) \tag{2.6}$$

The model $\mathcal{M}$ in Equation (1.1) is therefore defined by integrating the ODE Equation (2.3) or Equation (2.4) in time

$$\mathcal{M}(\mathbf{x}_t, t) = \int_{t\Delta\tau}^{(t+1)\Delta\tau} \frac{d\mathbf{x}}{dt} dt \tag{2.7}$$

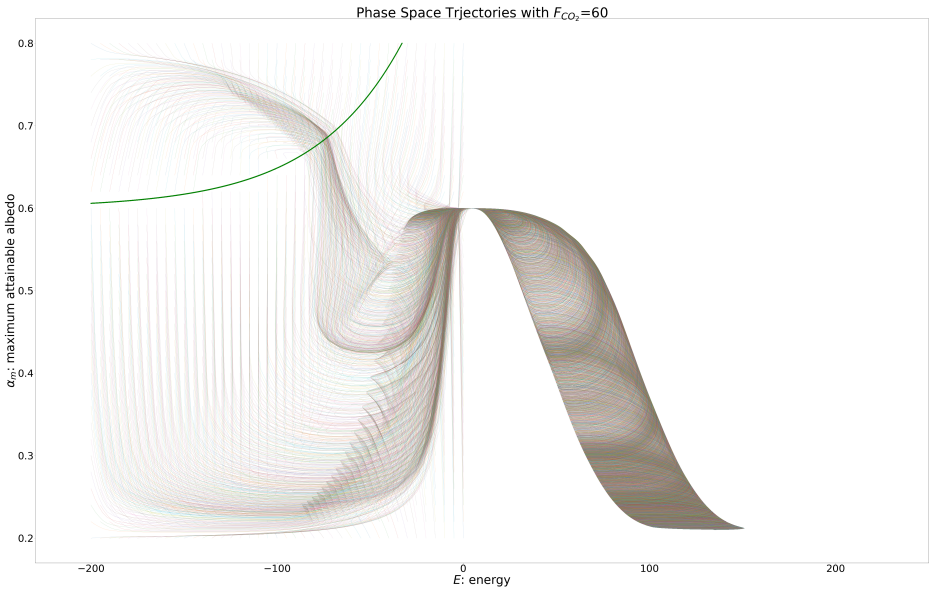

Figure 2.3: Trajectories with various initial points when $F_{\mathrm{CO}_2} = 60$.

where $\frac{d\mathbf{x}}{dt} = (\frac{dE}{dt}, \frac{d\alpha_m}{dt})$, and $\Delta\tau$ is the time step in data assimilation. In our study, all the ODE's are numerically integrated using the Matlab program *disode45* (Calvo et al., 2016) with its default settings (the major algorithm parameters include absolute error tolerance 10^{-6} and relative error tolerance 10^{-4}).

2.3 Proxy for Ice and Pond Concentration and Satellite Radiances

As mentioned in Section 1.2.3, for our particular problem, we need to define from our state variables E and α_m proxies for other physical quantities of interest, including ice and pond concentration values, the satellite radiances, and satellite-retrieved ice concentration. In Equation (2.8) We define the concentration of ice C_i to be the one minus the percent difference between the physically highest attainable albedo of 0.8 and the average of the maximum attainable albedo and the current albedo. The concentration here would approach 1 when the system is in a very cold state and the average is close to 0.8.

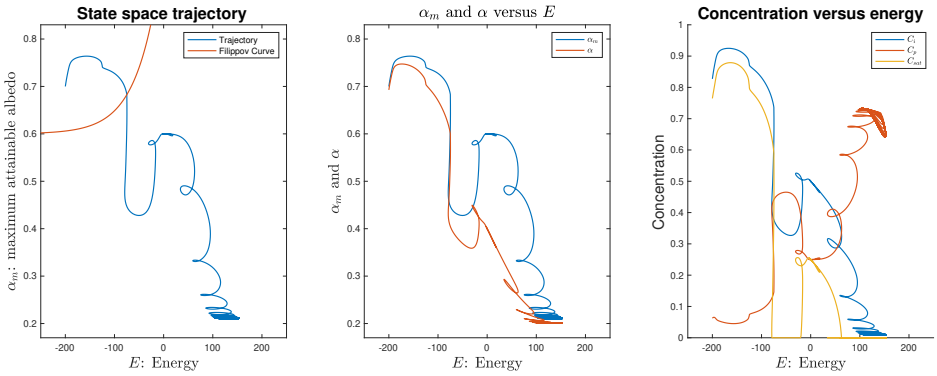

Figure 2.4: Overview of the proposed proxy for the melt pond problem in sea ice concentration retrieval.

$$C_i(E, \alpha_m) = 1 - \left(\frac{0.8 - \frac{1}{2}\left(\alpha_m + \alpha(E, \alpha_m)\right)}{0.6} \right) \tag{2.8}$$

The pond concentration is defined as one minus the ratio of the current albedo to the maximum surface albedo. The idea here is that a low albedo at time t compared to the maximum attainable albedo should mean the surface is covered in ponds.

$$C_p(E, \alpha_m) = (0.5 \tanh(\frac{E + 200}{10}) + 0.5)(1 - \frac{0.2}{\alpha(E, \alpha_m)})^{1000}(1 - \frac{\alpha_m}{0.8}) \tag{2.9}$$

For the satellite retrieved concentration, to model the fact that the melt ponds obscure the microwave signature of the ice and thus melt ponds and open water are indistinguishable in terms of satellite radiances as mentioned in Section 1.2.2, we take the maximum of zero and the difference between ice concentration and pond concentration.

$$C_{\text{sat}}(E, \alpha_m) = \max(0, C_i(E, \alpha_m) - C_p(E, \alpha_m)) \tag{2.10}$$

Figure 2.3 plots the quantities of interest as they change with the state variables on the trajectory shown in Figure 2.2 in the state space. Figure 2.3 gives plots of the time evolution of quantities of interest.

Finally, for the satellite radiances, we define functions of the state variables that give non-unique results around $E = 0$, the energy where ponds form and drain. Figure 2.3 plots the radiances as they

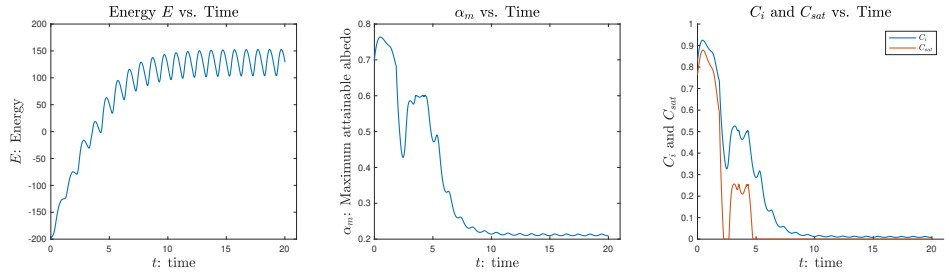

Figure 2.5: Overview of the proposed proxy for the melt pond problem in sea ice concentration retrieval.

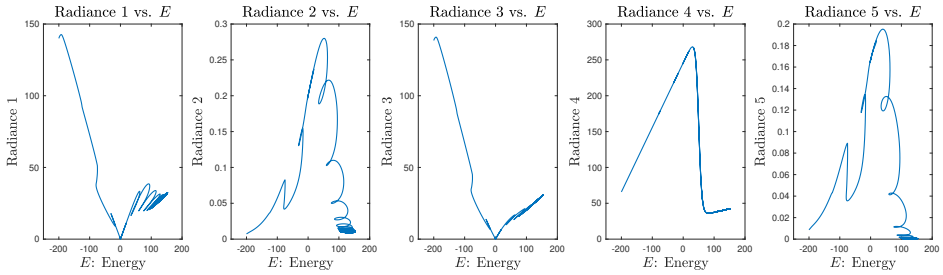

Figure 2.6: The proxy for satellite radiances versus energy.

change with the state variables on the trajectory shown in Figure 2.2 in the state space. Figure 2.3 gives plots of the time evolution of the five radiances.

$$\mathbf{O}(E,\alpha_m) = \begin{bmatrix} |E\alpha_m| \\ \alpha_m - \alpha(E,\alpha_m) \\ \alpha(E,\alpha_m)|E| \\ (0.5 + 0.4\tanh(\frac{50-E}{10}))(E+273.15) \\ C_i(1-\frac{\alpha(E,\alpha_m)}{\alpha_m}) \end{bmatrix} \tag{2.11}$$

2.4 Problem of Using Satellite Retrived Concentration in DA

In terms of data assimilation, if we assimilate with the satellite retrieved concentration, the real observation operator $\mathcal{H}_{\text{sat}}$ generating the observations from state variables in Equation (1.2) would

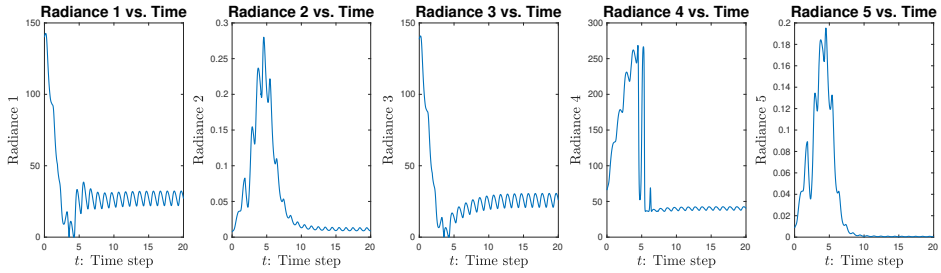

Figure 2.7: The proxy for satellite radiances versus time.

be the actual satellite retrieved ice concentration C_{sat} as a function of E and α_m in Equation (2.10)

$$\mathcal{H}_{\text{sat}}(\mathbf{x}) = \mathcal{H}_{\text{sat}}(E, \alpha_m) = C_{\text{sat}}(E, \alpha_m) = \max(0, C_i(E, \alpha_m) - C_p(E, \alpha_m)) \tag{2.12}$$

However, since we think it is supposed to give the correct ice concentration, the observation operator $\mathcal{H}_{\text{retrieval}}$ that is used to produce forecast output $\mathbf{y}_t^{f_i}$ in Equation (1.1.2) and Equation (1.13) would be the ice concentration $C_i(E, \alpha_m)$ as a function of E and α_m in Equation (2.8)

$$\mathcal{H}_{\text{retrieval}}(\mathbf{x}) = C_i(E, \alpha_m) = 1 - \left(\frac{0.8 - \frac{1}{2}\left(\alpha_m + \alpha(E, \alpha_m)\right)}{0.6} \right) \tag{2.13}$$

In this case, there is a systematic bias between $\mathcal{H}_{\text{sat}}$ and $\mathcal{H}_{\text{retrieval}}$. As discussed in Section 1.1.2, the apparent error in our knowledge about the observation operator $\mathcal{H}$ could create severe problems in EnKF, and thus assimilate wrong information into our state estimate. Therefore, we propose to assimilate directly on satellite radiances with machine learned observation operator. In that case, our observation space is no longer the 1-dimensional space of ice concentration, but the 5-dimensional space of the satellite radiances. The real observation operator $\mathcal{H}$ would be

$$\mathcal{H}(\mathbf{x}) = \mathcal{H}(E, \alpha_m) = \mathbf{O}(E, \alpha_m) = \begin{bmatrix} |E\alpha_m| \\ \alpha_m - \alpha(E, \alpha_m) \\ \alpha(E, \alpha_m)|E| \\ (0.5 + 0.4\tanh(\frac{50-E}{10}))(E + 273.15) \\ C_i(1 - \frac{\alpha(E,\alpha_m)}{\alpha_m}) \end{bmatrix} \tag{2.14}$$

While we do not have information of the specific form of the five functions in Equation (2.14) that make up $\mathcal{H}$, we aim to find with machine learning a function $\mathcal{H}_{\text{ML}} \in C(\mathbb{R}^2, \mathbb{R}^5)$ that closely approximates $\mathcal{H}$, that is

$$\mathcal{H}_{\text{ML}} \approx \mathcal{H} \quad \text{or} \quad \mathcal{H}(\mathbf{x}_t, t) - \mathcal{H}_{\text{ML}}(\mathbf{x}_t, t) = \boldsymbol{\xi}_t \tag{2.15}$$

where $\boldsymbol{\xi}_t$ is ideally an i.i.d sequence with 0 mean and small variance. In this way, although there is still an error between $\mathcal{H}_{\text{ML}}$ and $\mathcal{H}$, the systematic bias present in the case of $\mathcal{H}_{\text{sat}}$ and $\mathcal{H}_{\text{retrieval}}$ is gone. If $\boldsymbol{\xi}_t$ does behave like an i.i.d sequence with 0 mean and small variance, we can incorporate this error as part of the uncertainty $\boldsymbol{\epsilon}_t$ in the forecast model in Equation (1.1) or as part of the uncertainty $\boldsymbol{\eta}_t$ in the observation in Equation (1.2).

- In Equation (1.1.2), $\mathcal{H}_{\text{ML}}$ would be used to calculate $\{\mathbf{y}_t^{f_i}\}_{i=1}^{i=q}$, which are further used to estimate the Kalman gain. Without explicitly quantifying the error in the approximation of $\mathcal{H}_{\text{ML}}$ to $\mathcal{H}$, we could think of this error as coming from the ensemble states $\{\mathbf{x}_t^{f_i}\}_{i=1}^{i=q}$ instead of from $\mathcal{H}_{\text{ML}}$, since both uncertainty in $\{\mathbf{x}_t^{f_i}\}_{i=1}^{i=q}$ and $\mathcal{H}_{\text{ML}}$ could could contribute to the uncertainty in $\{\mathbf{y}_t^{f_i}\}_{i=1}^{i=q}$. We will discuss this perspective in more details in Section 4.3.

- Furthermore, in Equation (1.13), each ensemble member is updated with the Kalman gain and the difference between the observation and forecast output, where $\mathcal{H}_{\text{ML}}$ is used in place of $\mathcal{H}$ in the equation. Here we could consider the uncertainty brought by $\mathcal{H}_{\text{ML}}$ as part of the uncertainty in the observations $\mathbf{y}_t$.

$$\mathbf{y}_t - \mathcal{H}_{\text{ML}}(\mathbf{x}_t^{f_i}) = \mathbf{y}_t - \mathcal{H}(\mathbf{x}_t, t) + \boldsymbol{\xi}_t = \boldsymbol{\eta}_t + \boldsymbol{\xi}_t \tag{2.16}$$

  Originally, the observations are perturbed by $\mathcal{N}(0, \mathbf{R}_t)$ in Equation (1.14) to account for uncertainty. Here we could account for this increased uncertainty from $\mathcal{H}_{\text{ML}}$ by first appropriately estimating $\boldsymbol{\xi}_t$ and further perturbing the observations accordingly. This is discussed in more details in Section 4.3.

CHAPTER 3

Methods

3.1 Data Generation

To generate a data set for the training of machine learning observation operators, we first select points from the state space as $\mathbf{X}$, calculate the corresponding radiances $\mathbf{Y}$, i.e. observations, and finally add error to the observations to get $\tilde{\mathbf{Y}}$.

state space points The first data set we generate is $\boldsymbol{\mathcal{D}}_{\text{traj}}$. To mimic the situation in the real world, we decide to use points on various trajectories in the phase space to form $\mathbf{X}_{\text{traj}}$. From the model's formulation, the maximum attainable albedo α_m is only meaningful between 0.2 and 0.8, and the region of interest for E is between -200 to 0. Therefore, 20×30 uniform grid data points from $S = [-200, 0] \times [0.2, 0.8]$ are chosen as initial conditions. Figure 2.2 and Figure 2.2 are in fact generated with such initial values with $F_{\text{CO}_2} = 0$ and $F_{\text{CO}_2} = 60$, respectively. Note that in our dynamical system Equation (2.3) and Equation (2.4), the forcing parameter F_{CO_2} is also of interest. The real world analogy of F_{CO_2} would be the combined influence of total carbon dioxide emissions, different weather conditions, climate changes, and etc. Thus we should expect various values of F_{CO_2} in the data set. To capture this effect, we generate trajectories with 12 values of F_{CO_2} ranging from -10 to 100. The model is run for 200 time steps with a step length of 0.05 for each initial condition and each value of F_{CO_2}, resulting in 2400 trajectories and 1440000 data points. From the data points, $\mathbf{Y}_{\text{traj}} = \mathcal{H}(\mathbf{X}_{\text{traj}})$ are calculated and noises are added to give $\tilde{\mathbf{Y}}_{\text{traj}}$. $\mathbf{X}_{\text{traj}}$ and $\tilde{\mathbf{Y}}_{\text{traj}}$ form the data set $\boldsymbol{\mathcal{D}}_{\text{traj}} = (\mathbf{X}_{\text{traj}}, \tilde{\mathbf{Y}}_{\text{traj}})$.

grid data points From Figure 2.2 and Figure 2.2, we can see that in places where the limit cycles exist, the trajectories are clustered and the data is dense in that region, whereas in places where the trajectories go through transient states, the data is sparse. As we will later demonstrate in

Section 3.3.1, such imbalance creates a problem for the machine learning algorithm. To fully exploit the potential of the machine learning algorithm to fit the satellite radiances, we also generate a data set with the unrealistically large amount of data filling up the whole state space. 2000×3000 uniform grid data points from $S = [-200, 0] \times [0.2, 0.8]$ are chosen to form $\mathbf{X}_{\text{grid}}$. From the data points, $\mathbf{Y}_{\text{grid}} = \mathcal{H}(\mathbf{X}_{\text{grid}})$ are calculated and noises are added to give $\tilde{\mathbf{Y}}_{\text{grid}}$. $\mathbf{X}_{\text{grid}}$ and $\tilde{\mathbf{Y}}_{\text{grid}}$ form the data set $\boldsymbol{\mathcal{D}}_{\text{grid}} = (\mathbf{X}_{\text{grid}}, \tilde{\mathbf{Y}}_{\text{grid}})$.. We expect that with sufficient data, the machine learning algorithm should perform well everywhere in the state space.

observations and error The observations are calculated for each of the generated data points with Equation (2.11). In the real world, the data is never perfectly clean, so it is important to add error in the observations. Moreover, the machine learning algorithms are expected to average out the white noise error and learn the true pattern behind the noises. Note that the five radiances have different magnitude scales, radiance 1,3,4 are of $\sim 10^1$ while radiance 2,5 are of $\sim 10^{-2}$. We initially did not take such effect into account and add the same error $\mathcal{N}(0, 1)$ to all five radiances. Experiments showed that machine learning algorithms were not able to learn anything about radiance 2,5. We also tried to add error proportional to the value of each radiance at each point. However, this approach created a problem for values near 0, in which case essentially no error is present. Our final approach is adding error proportional to the mean absolute value of each radiance.

$$\bar{y}_j = \sum_{i=1}^{n} |y_{ij}| \tag{3.1}$$

$$\tilde{y}_{ij} = y_{ij} + \eta_{ij} \tag{3.2}$$

$$\eta_{ij} \sim \mathcal{N}(0, \lambda \bar{y}_j) \tag{3.3}$$

where λ is a control parameter for the magnitude of the error. We generate 7 datasets with $\lambda \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$ to investigate how error affects machine learning algorithms. In data assimilation, we also use the same approach to add error to observations.

3.2 Model Training and Selection

Since we construct the function for satellite radiances $\mathcal{H}$ in a highly nonlinear manner, we decide to use the artificial neural network (ANN) as the machine learning algorithm. Other algorithms such as XGBoost and Random Forest are also worth experimenting in the future. We split the data set randomly into 90% for training and validation, and 10% for evaluation. We also exclude from the training set the points on the trajectory that we will use in the DA experiment to avoid overfitting. The training-validation set is further split into the training set and validation set. We choose hyperbolic tangent $\tanh$ as the activation function in hidden layers. We add batch normalization (Ioffe and Szegedy, 2015) before each hidden layer and dropout (Srivastava et al., 2014) after each hidden layer to regularize the network and prevent it from overfitting. All parameters are initialized using Xavier initialization (Glorot and Bengio, 2010). The network is trained with Adam optimizer (Kingma and Ba, 2015). Hyperparameters are chosen based on 9-fold cross-validation. The network is implemented in Python with Keras framework using Tensorflow as backend. After experiments, we choose four layers to get our architecture shown in Figure 3.2. The machine learning model can be written as a composed function in Equation (3.4)

$$\boldsymbol{y} = \boldsymbol{x}_{\text{out}} = \boldsymbol{W}_4(\tanh(\boldsymbol{W}_3(\tanh(\boldsymbol{W}_2(\tanh(\boldsymbol{W}_1\boldsymbol{x}_{\text{in}} + \boldsymbol{b}_1)) + \boldsymbol{b}_2)) + \boldsymbol{b}_3)) + \boldsymbol{b}_4 \tag{3.4}$$

where $\tanh$ is the element-wise hyperbolic tangent function, $\boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{W}_4$ are 5×5 matrices, $\boldsymbol{W}_1$ is a 5×2 matrix and $\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3, \boldsymbol{b}_4$ are 5-dimensional vectors. The total number of parameters is 105. The parameters are determined by the optimization algorithm.

3.3 Evaluation

To examine the accuracy of the prediction model, two evaluation measures are used in this study: Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE). They are defined

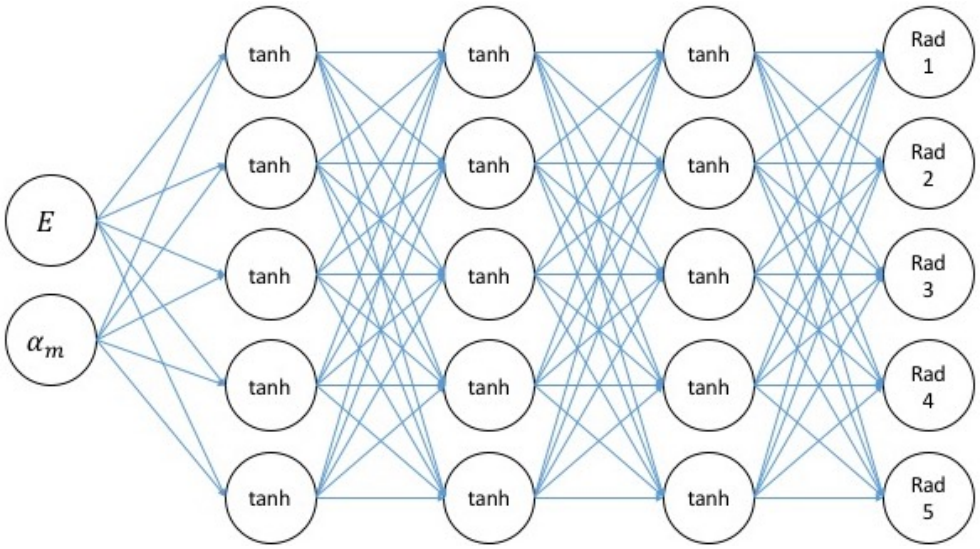

Figure 3.1: Graphical representation of the proposed machine learning model.

as:

$$\text{RMSE} = \sqrt{\frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}(y'_{ij} - y_{ij})^2} \tag{3.5}$$

$$\text{MAPE} = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\left|\frac{y'_{ij} - y_{ij}}{y_{ij}}\right| \tag{3.6}$$

where y_{ij} is the jth feature of the ith sample, y'_{ij} is the predicted value of the corresponding y_{ij}.

Figure 3.3 plots the points on the trajectory that we will use in the DA experiment and the corresponding values of radiance 1,2,4 on those points. Figure 3.3 is a larger illustration of the lower left plot of Figure 3.3. The figures compare the values given by machine learning algorithm $\mathcal{H}_{\text{ML}}(\mathbf{X})$ with the truth $\mathbf{Y}$ and the noisy training data $\tilde{\mathbf{Y}}$ with $\lambda = 0.4$. In most places, the machine learning values overlap with the truth and give reasonable values in general. Particularly, it seems that machine learning is able to average out the noise and discover the real pattern in the data. In the lower middle plot of the 4th radiance value, even though the noise in the training data is disproportionately large, our machine learning model is still able to recover the true values behind the noise. In fact, through experiment, we found out that, with appropriate techniques and sufficient data, machine learning is not severely affected by the amount of noise in the training data. The model trained with $\lambda = 60\%$ is only slightly worse than the model trained with $\lambda = 0\%$. Besides, the performance of the benchmark

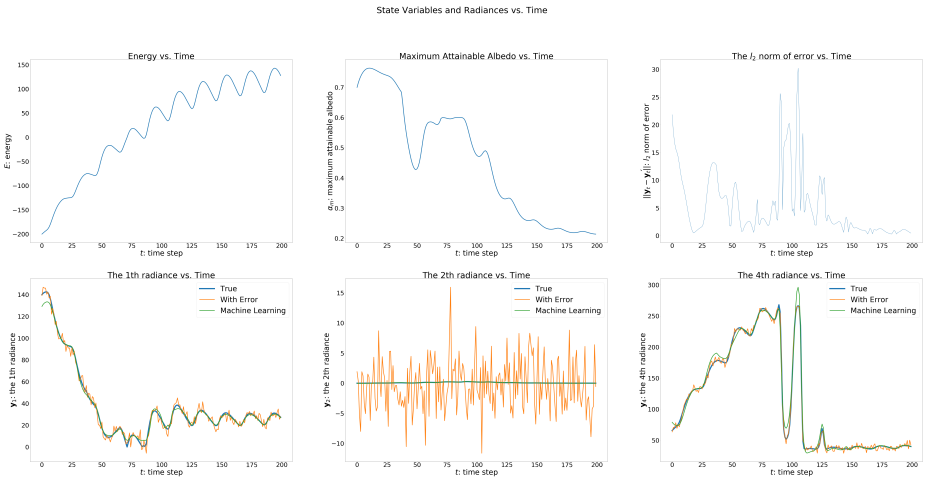

Figure 3.2: The performance of machine learning observation operator evaluated on one trajectory.

machine learning model $\hat{\mathcal{H}}_{\text{ML}}$ trained with a large amount of data is almost perfect, as we expected. The plot is not shown since the machine learning values virtually overlap with the truth everywhere. In some sense, we can safely say that the machine learning algorithm fit the function perfectly and should be identical as and indistinguishable from the true function $\mathcal{H}$. The accuracy of $\mathcal{H}_{\text{ML}}$ and $\hat{\mathcal{H}}_{\text{ML}}$ satisfies our assumption in Equation (2.15) that the error in the machine learned observation operator should be white noise with zero mean and small variance.

3.3.1 Data Density

Although in general machine learning achieves a high level of accuracy, $\mathcal{H}_{\text{ML}}$ displays different levels of error in different parts in the state space. It is well known that the amount of available training data is a significant factor in determining the performance of machine learning algorithms. In our study, since we generate data with trajectories for $\mathcal{H}_{\text{ML}}$, the data points are not uniformly distributed spatially. In regions where the system achieves equilibrium states and where limit cycles exist, the data points are clustered together and we therefore have a large amount of data to train the machine learning algorithm. In regions where the system goes through transient states, the data points are sparse and we have a limited amount of data to train the machine learning algorithm. The experiments show correlation between the accuracy and the amount of available training data, or data density, in different regions in the state space. To demonstrate this correlation and to quantify the

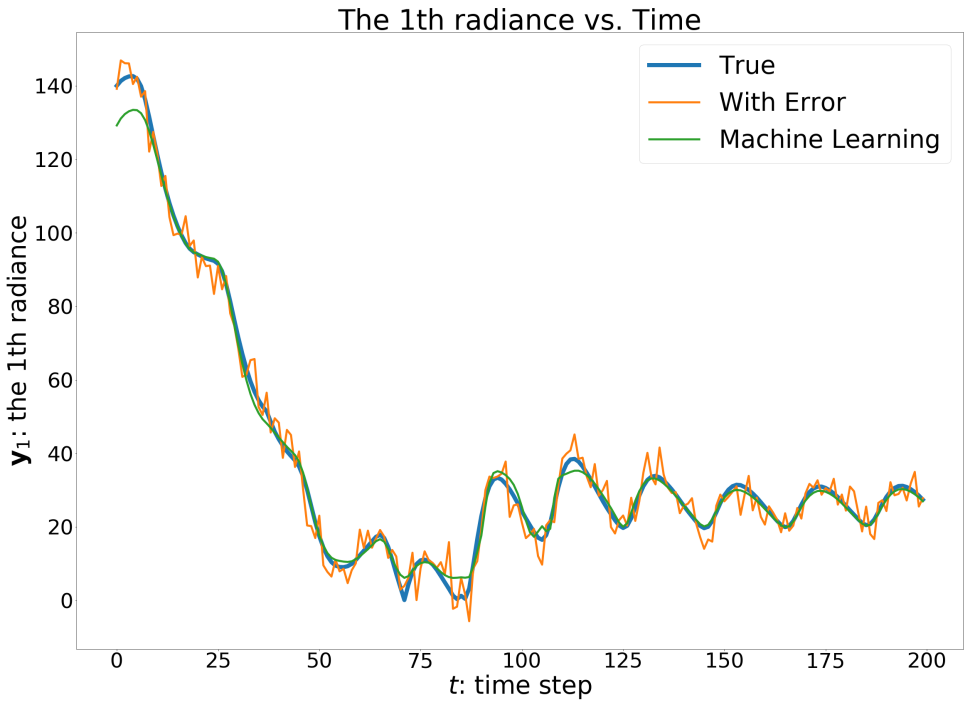

Figure 3.3: Time evaluation of the 1st radiance, true value, noisy value and ML fitted value.

expected error in different regions, we divide the state space into 45×30 uniform grids, with each grid square representing 10 units in energy E and 0.02 unit in maximum attainable albedo α_m. In each grid square, we sum up the number of data points in the training data (and then take logarithm) to represent the data density in this square region. We take the averaged RMSE of all the values in the grid square to serve as the expected error estimation in that region. Figure 3.3.1 is a heat map plot of the density in the grid regions, where the color bar represents the logarithm of the number of available data points in each grid. Figure 3.3.1 is a heat map plot of the expected error in the grid regions, where the color bar represents the averaged RMSE.

From the plots, we can see the correlation between data density and error. The darker blue colors in Figure 3.3.1 roughly correspond to the lighter green-yellow colors in Figure 3.3.1. In particular, for the equilibrium states, where sufficient training data exists, our model tends to perform quite well, whereas for the transient state, where training data is extremely sparse, the values given by machine learning does not make much sense. As previously described in Section 2.4 and Equation (2.15), we expect the difference between $\mathcal{H}_{\text{ML}}$ and $\mathcal{H}$ to be an i.i.d. sequence with 0 mean and small variance. However, as the state variables evolve in the state space, the error $\boldsymbol{\xi}$ also varies. To address this

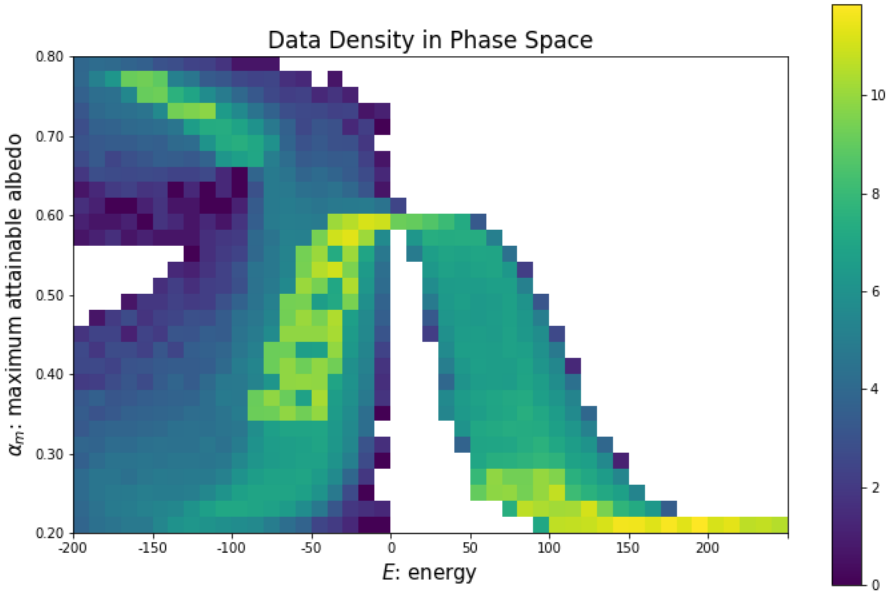

Figure 3.4: Density of available training data for machine learning in the state space.

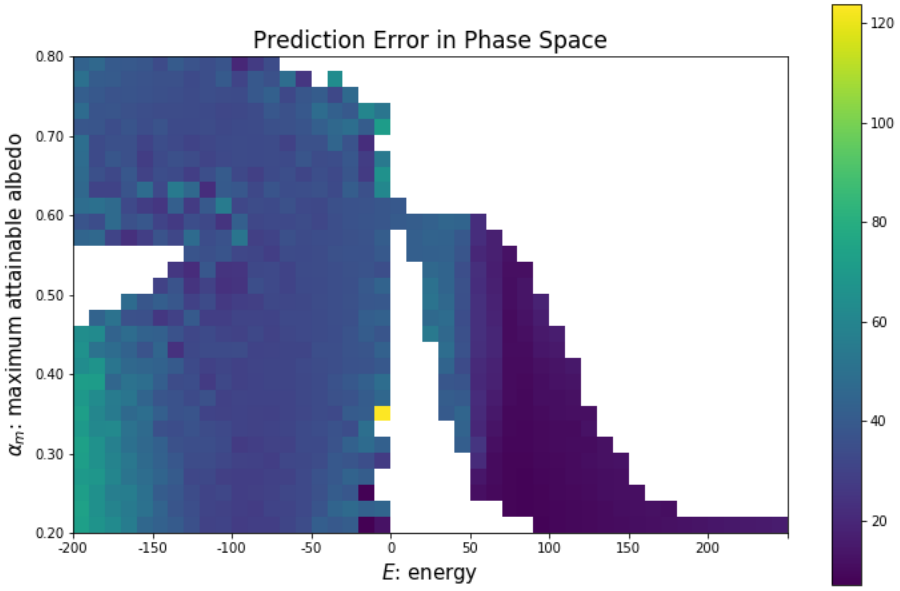

Figure 3.5: Error of machine learning algorithm in state space.

issue in data assimilation, we consider $\boldsymbol{\xi}$ as a function of the state variables E and α_m, and estimate $\boldsymbol{\xi}(E, \alpha_m)$ with the averaged RMSE error and data density in the corresponding grid square. Then we apply appropriate covariance estimation and inflation. The details are discussed later in Section 4.3.

3.3.2 Training with Sparse Data

As discussed above, data density plays a significant role in determining the accuracy of the machine learning algorithm in different regions in the state space. In the real world, different regions of the state space represent different physical configurations of the system. It is completely possible that the current states of the system are rarely seen in the past and thus the machine learning algorithm does not have sufficient historic data to learn the behavior of observation operator in such situation. To further investigate the influence of sparse training data in the region of interest, we generate another two data sets, where we have limited training data in a certain region, and train another two machine learning observation operator with these data sets respectively. We are most interested in the region $S = \{(E, \alpha_m) : E \in [-50, 50]\}$, shown in Figure 3.3.2 as the shaded region, where the energy crosses zero and the interesting dynamics corresponding to pond forming and draining happens. From the data set $\boldsymbol{\mathcal{D}}_{\text{traj}}$, we randomly drop 95% (70%, respectively) of the data from $\mathbf{X}_{\text{traj}}$ in the region S and the corresponding observations from $\tilde{\mathbf{Y}}_{\text{traj}}$ to significantly decrease the amount of available training data in that region. The resulting data sets are $\boldsymbol{\mathcal{D}}_{5\%} = (\mathbf{X}_{5\%}, \tilde{\mathbf{Y}}_{5\%})$ (and $\boldsymbol{\mathcal{D}}_{30\%} = (\mathbf{X}_{30\%}, \tilde{\mathbf{Y}}_{30\%})$, respectively).

After training our neural network with $\boldsymbol{\mathcal{D}}_{5\%}$ and $\boldsymbol{\mathcal{D}}_{30\%}$, we get two machine learning observation operators $\mathcal{H}_{5\%}$ and $\mathcal{H}_{30\%}$. They are again evaluated and analyzed with the same methods described in Section 3.3. Figure 3.3.2 compares the original data density of $\boldsymbol{\mathcal{D}}_{\text{traj}}$ and the data density of $\boldsymbol{\mathcal{D}}_{5\%}$ after dropping 95% data in $E \in [-50, 50]$. Figure 3.3.2 compares the accuracy of $\mathcal{H}_{\text{ML}}$ and $\mathcal{H}_{5\%}$. There is a clear boundary in both figures - inside the region S where data is dropped, data is apparently sparser and the error is significantly larger than outside the region. We can clearly see that with only 5% of the data, our neural network starts to make mistakes.

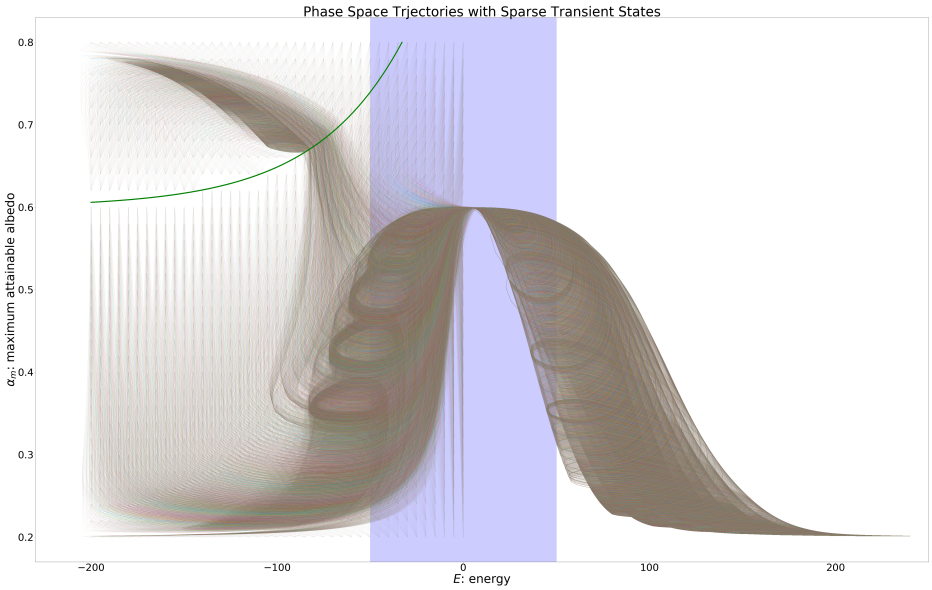

Figure 3.6: Drop out data in the shaded region of interest.

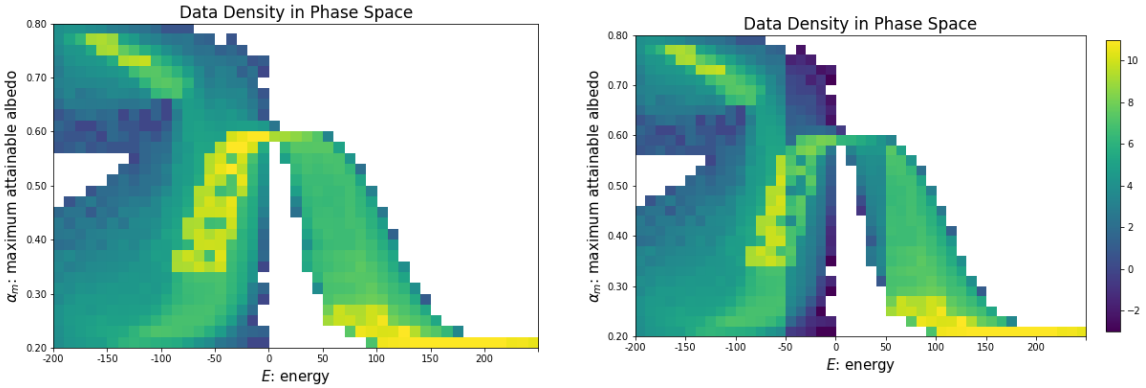

Figure 3.7: Comparison between the original data density (left) and the data density after dropping 95% data in $E \in [-50, 50]$.

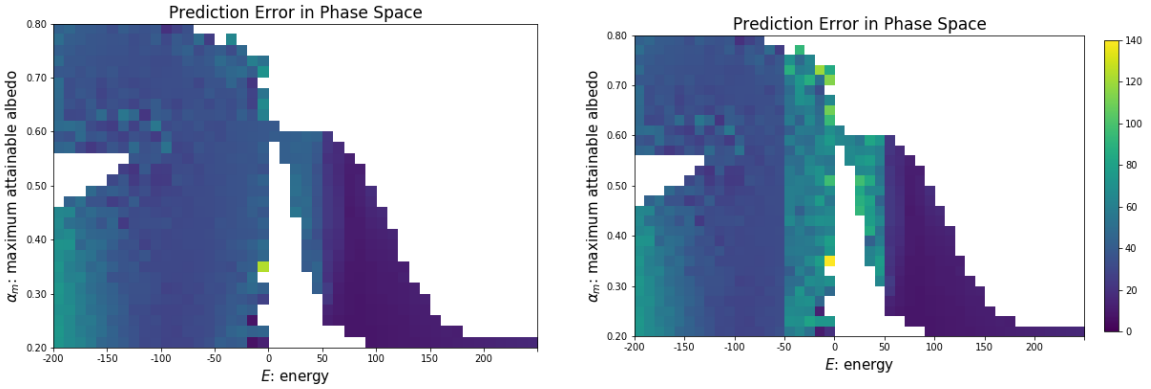

Figure 3.8: Comparison between the error of machine learning algorithm with all data (left) and with 5% data in $E \in [-50, 50]$.

CHAPTER 4

Experiments

4.1 Setup

In this chapter, we test our machine learning observation operator in the data assimilation scheme. The state variable in the experiment is $\mathbf{x} = [E, \alpha_m]$. The model that evolves the states in time $\mathcal{M}$ is defined by Equation (2.3) and Equation (2.4). To focus on the observation operator, we adopt the perfect model assumption in our experiment, i.e. assume we know the true model and exclude any model error from our experiment. Therefore the forecast model dynamics is the same as the true dynamics. The time step is set as 0.05, which roughly corresponds to 18 days in the physical sense, allowing us to see the seasonal effects brought by the seasonally varying terms in the model $F_T(t)$, $F_0(t)$, and $F_s(t)$. The experiment lasts for 200 steps, i.e. 10 years, after which no further interesting behaviors can be observed from the experiments. The true observations are generated by Equation (2.11) in most experiments and by Equation (2.10) in the experiment with $\mathcal{H}_{\text{retrieval}}$. We assume that the observations are available at every time t. We also add noise to the true observations the same way we add error to the training data in Equation (1.1.2), Definition 3.2, and Definition 3.3, with $\lambda = 20\%$. Data assimilations are done at every integration time t_n. The analysis error is defined as the l_2 norm of the difference between the true value and the analysis.

The true initial condition is $\mathbf{x}_0 = [0.7, -200]$, i.e. $E_0 = -200$ and $\alpha_{m_0} = 0.7$, with $F_{\text{CO}_2} = 60$, from which Figure 2.2 is generated. The initial ensemble mean is a perturbed value from the true initial condition:

$$\bar{\mathbf{x}}_0^f = \mathbf{x}_0 + \mathcal{N}(0, \begin{bmatrix} 0.12 & 0 \\ 0 & 40 \end{bmatrix})$$

The initial spread is defined as

$$\mathbf{P}_0^f = \begin{bmatrix} 0.12 & 0 \\ 0 & 40 \end{bmatrix}$$

The size of the ensemble is taken as 100. Through experiments, we found that a larger size of the ensemble does not provide a proportionate improvement in performance. 100 seems a computationally reasonable size for our experiment.

4.2 Choices of Observation operators

We conduct two parts of experiments consisting of several runs of identical twin experiment with ensemble Kalman filter, using different observation operators, in order to show the different performance of EnKF with assimilating the incorrect information from satellite retrieved ice concentration and with assimilating directly the satellite radiances.

In the first part, the observation space is $\mathbb{R}$, and the observations are real numbers representing the satellite retrieved ice concentration generated with $\mathcal{H}_{\text{sat}}$, defined in Equation (2.12), while the forecast output $\mathbf{y}_t^{f_i}$ are generated with $\mathcal{H}_{\text{retrieval}}$, which gives C_i as the supposed satellite retrieved value defined in Equation (2.13).

In the second part, the observation space is $\mathbb{R}^5$, and the observations represent the satellite radiances generated with $\mathcal{H}$ defined in Equation (2.11). We experiment with several machine learning observation operators to generate the forecast output:

- $\mathcal{H}_{\text{ML}}$: Machine learning observation operator trained on $\boldsymbol{\mathcal{D}}_{\text{traj}}$ described in Section 3.1, consisting of various trajectories from different initial points with various values of F_{CO_2}.
- $\hat{\mathcal{H}}_{\text{ML}}$: Machine learning observation operator trained on $\boldsymbol{\mathcal{D}}_{\text{grid}}$ described in Section 3.1, consisting of an unrealistically large amount of data filling up the whole state space.
- $\mathcal{H}_{5\%}$: Machine learning observation operator trained on $\boldsymbol{\mathcal{D}}_{5\%}$ described in Section 3.3.2, with 5% data in $E \in [-50, 50]$ for training
- $\mathcal{H}_{30\%}$: Machine learning observation operator trained on $\boldsymbol{\mathcal{D}}_{30\%}$ described in Section 3.3.2, with 30% data in $E \in [-50, 50]$ for training

4.3 Error Estimation and Inflation

As stated above in Section 1.1.2, the performance of ensemble Kalman filter largely depends on the correct estimation of the error covariance matrix of the model and the observation operator, i.e. **Q** and **R**, respectively. In most data assimilation literatures, **Q** and **R** are assumed given and are usually constant diagonal matrices. However, such simple approach would not work in this case. As stated in Section 3.3.1, the machine learning algorithm displays different levels of accuracy throughout the state space, and we consider the error of $\mathcal{H}_{\text{ML}}$, $\boldsymbol{\xi}$, as a function of $\mathbf{x}$. Therefore, we will use the evaluation error of the machine learning algorithm to estimate **R** in different regions in the state space. Specifically, **R** is a diagonal matrix and is a function of the state variables E and α_m.

$$\mathbf{R}(t) = \mathbf{R}(E(t), \alpha_m(t)) \tag{4.1}$$

$$\mathbf{R}_{i,i}(t) = \boldsymbol{\xi}_i(E(t), \alpha_m(t)) \tag{4.2}$$

To further improve the analyses in our numerical experiments, we employ variance inflation,

$$\mathbf{P}_t^a \rightarrow \mathbf{P}_t^a + \frac{\mu\Lambda}{q}\mathbf{I}_q \tag{4.3}$$

where $\mathbf{P}_t^a$ is the analysis error covariance matrix defined in Equation (1.1.2). μ is an inflation coefficient, Λ is the trace of analysis error covariance matrix $\Lambda = \text{tr}(\mathbf{P}_t^a)$, and q the number of ensemble members. This particular form of variance inflation was proposed in (OTT et al., 2004) where it is referred to as "enhanced variance inflation". Enhanced variance inflation has the effect of enhancing the estimated probability of error in directions that formally show only minimal error probability. The general purpose of employing a variance inflation is to correct for the loss of variance in the ensemble due to nonlinearities and sampling errors. Most importantly, variance inflation can also stabilize the ensemble Kalman filter in the presence of model errors, as it was shown in (BAEK et al., 2006) for Lorenz 96 model. In our study, although model errors are not existent in $\mathcal{M}$, the error in the observation operator can be seen as another form of model error.

Since the machine learning algorithm is largely affected by data density and has very different levels of accuracy in different parts of the state space, we decide to use different values of μ based

on the level of accuracy of the machine learning model. After experiments, we decide to define μ proportional to the logarithm of the error in the respective region of the state space.

4.4 Results

Figure 4.4 plots the performance of data assimilation with the real observation operator, serving as a benchmark for comparison. In the two plots, the red stars represent the true values of E and α_m, respectively. The blue curve and the red curve represent the forecast estimate and the analysis estimate of the state variables, respectively. The shades around the curves represent the ensemble spread, i.e. the estimate for uncertainty in E and α_m. The upper bound of the shade is computed by adding the sample standard deviation of the ensemble to the ensemble mean. Similarly, The lower bound of the shade is computed by subtracting the sample standard deviation of the ensemble from the ensemble mean. The larger the shade is, the more uncertain we are about the estimate and the less confidence we have. In the beginning, the estimates are far from the truth and the variance is large. After $20 - 30$ time steps, or $1 - 1.5$ time units, the error becomes significantly less and the forecast and analysis start to converge to the truth. Eventually, the forecast and analysis curves almost overlap with the truth with a few small deviations caused by the stochasticity in the evolution of ensemble members and the generation of observations. The variance becomes sufficiently smaller, representing the increased confidence in the estimate. Although not apparent from the plots, the variance never goes to zero due to error covariance matrix inflation. We always want some level of uncertainty in our estimate. If the ensemble collapses to a single point, EnKF might not be able to follow the truth once a small deviation caused by stochasticity occurs.

Figure 4.4 plots the performance of data assimilation with $\hat{\mathcal{H}}_{\text{ML}}$, the machine learning observation operator trained with an unrealistically large amount of data. We can barely tell the difference between $\hat{\mathcal{H}}_{\text{ML}}$ and the true observation operator $\mathcal{H}$, except for the different shapes of shade at the first few steps caused by randomly initializing the ensembles. Again, the forecast and analysis estimates converge to the truth after $20 - 30$ time steps and follow along the true trajectory thereafter. Such results demonstrate that the machine learning observation operator can achieve the same performance as the true observation operator, as long as it is provided with sufficient data.

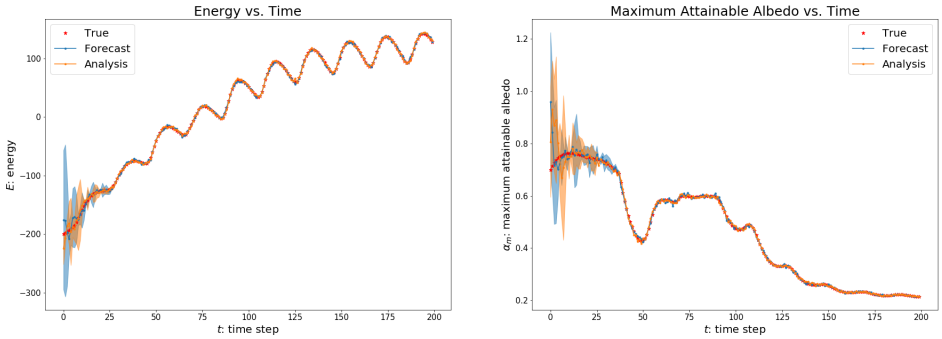

Figure 4.1: DA experiment with $\mathcal{H}$, the true observation operator.

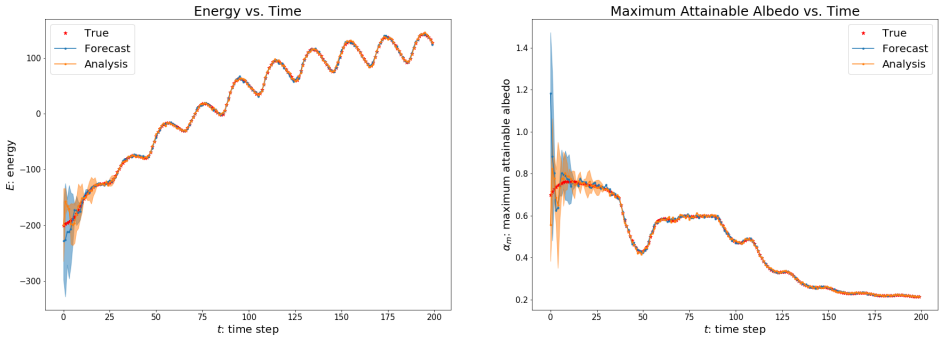

Figure 4.2: DA experiment with $\hat{\mathcal{H}}_{\text{ML}}$, i.e. ML OP with grid training data.

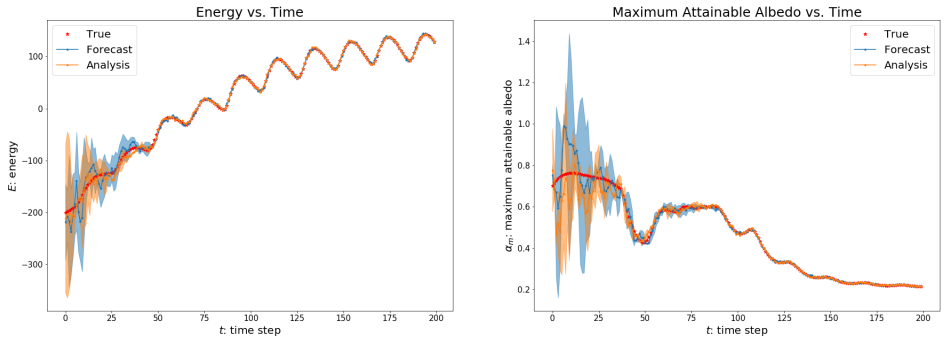

Figure 4.3: DA experiment with $\mathcal{H}_{\text{ML}}$, i.e. ML OP with training data from trajectories.

Figure 4.4 plots the performance of data assimilation with $\mathcal{H}_{\text{ML}}$, the machine learning observation operator trained with data from trajectories. Comparing Figure 4.4 with Figure 4.4, we can see that there is a noticeable difference at the first time steps. This is due to the lack of training data at transient states at the first time steps for $\mathcal{H}_{\text{ML}}$. However, we notice that after around 50 time steps, the forecast and analysis estimates converge to the truth. On the one hand, ensemble Kalman filter generally takes time to converge. On the other hand, once the trajectories enter the area where limit cycles reside and data becomes sufficiently dense, the accuracy of the machine learning observation operator increases, leading to a corresponding improvement of performance in data assimilation. Eventually, like the experiments with $\mathcal{H}$ and $\hat{\mathcal{H}}_{\text{ML}}$, the forecast and analysis values overlap with the truth and the variance becomes visibly smaller. The real world implication would be that even if we do not have enough training representing every possible physical situation or every possible configuration of the sea ice and melt ponds, we can still apply the machine learning observation operator in data assimilation.

Figure 4.4 plots the performance of data assimilation with $\mathcal{H}_{\text{retrieval}}$, the proxy for satellite retrieved ice concentration. Unlike the experiments with $\mathcal{H}$, $\mathcal{H}_{\text{ML}}$, and $\hat{\mathcal{H}}_{\text{ML}}$, the forecast and analysis estimates with $\mathcal{H}_{\text{retrieval}}$ never converge to the truth. Although the curves seem to follow the true dynamics, they are constantly oscillating around the true values and rarely give reasonably accurate predictions. This closely resembles the real world situation, where in the summertime, as the melt ponds start to form, the sea ice models assimilating the inaccurate sea ice concentration retrieved from satellite radiances begin to fail.

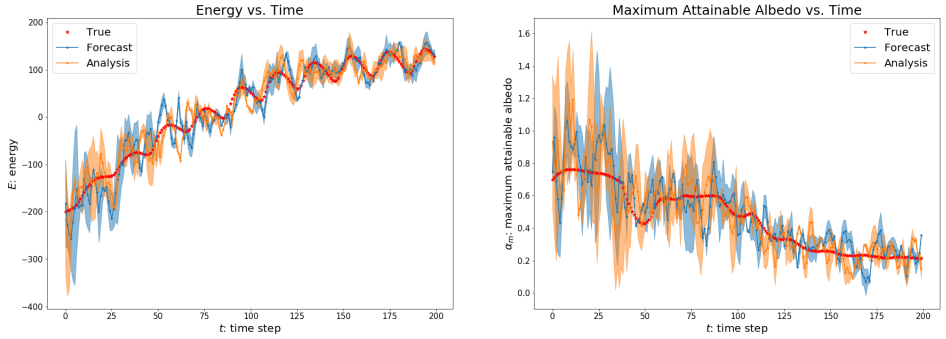

Figure 4.4: DA experiment with $\mathcal{H}_{\text{retrieval}}$, the observation operator representing the proxy for satellite retrieval algorithm.

4.4.1 Results with Sparse Data

Figure 4.4.1 and Figure 4.4.1 show the performance of data assimilation with $\mathcal{H}_{5\%}$ and $\mathcal{H}_{30\%}$, the machine learning observation operator trained with only 5% or 30% data in the region of interest. Comparing Figure 4.4.1 and Figure 4.4.1 with Figure 4.4 and Figure 4.4, we can see that the performance is obviously worse in general. In particular, the estimates with $\mathcal{H}_{5\%}$ does not converge to the truth after 200 steps. This shows that the severe lack of training data can cause the failure of data assimilation, especially in the regions of interest where the most important dynamics take place. However, we notice that for $\mathcal{H}_{30\%}$ after around 150 time steps, when the trajectory begins to deviate from the region of insufficient data, i.e. when $E > 50$, the forecast and analysis estimates still converge to the truth. Most importantly, $\mathcal{H}_{30\%}$ clearly outperforms $\mathcal{H}_{\text{retrieval}}$, suggesting that machine learning is still a desirable option even in situations where the data set is incomplete.

4.4.2 Review and Comparison

Figure 4.4.2 and Figure 4.4.2 compare the performance of different observation operators in terms of forecast and analysis absolute error in E and α_m at each time step. Obviously, data assimilation with the retrieved concentration performs far worse than the others.

Figure 4.4.2 and Figure 4.4.2, after removing $\mathcal{H}_{\text{retrieval}}$, provide a clearer illustration of how assimilating using the machine learning observation operator performs. We can see that the behavior in E and α_m are consistent. Around the first 30 time steps, $\mathcal{H}_{\text{ML}}$ does not perform very well. This

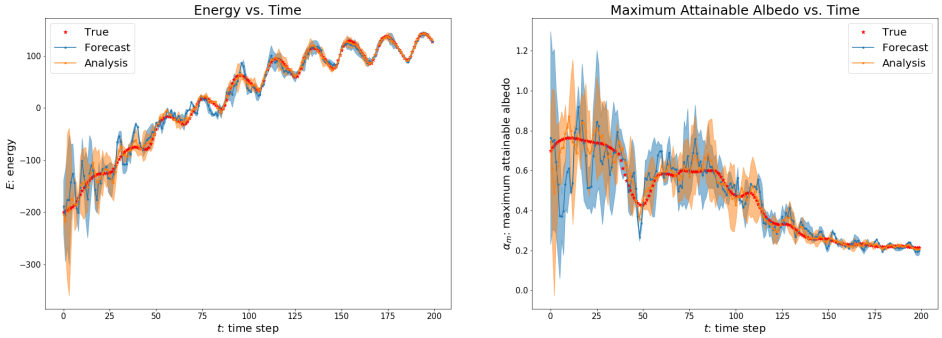

Figure 4.5: DA experiment with $\mathcal{H}_{30\%}$, i.e. ML OP trained with 30% data in $E \in [-50, 50]$.

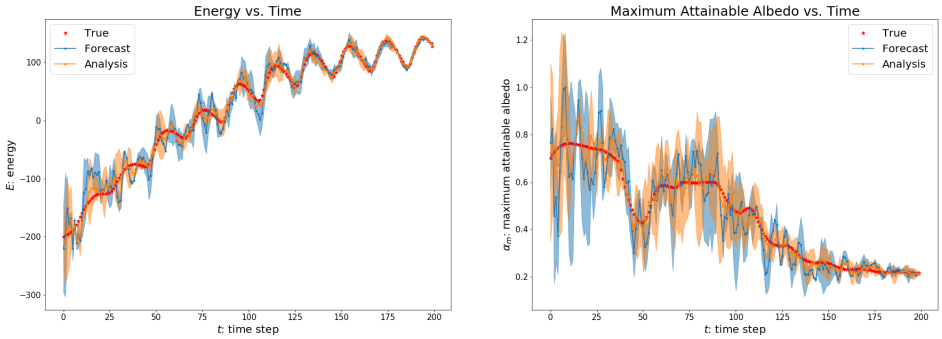

Figure 4.6: DA experiment with $\mathcal{H}_{5\%}$, i.e. ML OP trained with 5% data in $E \in [-50, 50]$.

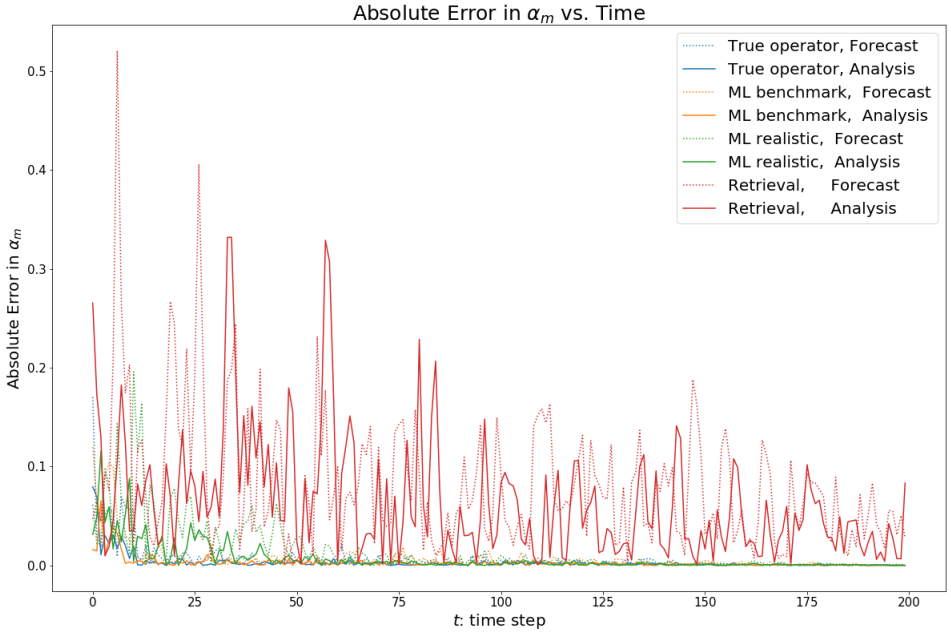

Figure 4.7: Time evolution of absolute error of α_m with $\mathcal{H}$, $\hat{\mathcal{H}}_{\text{ML}}$, $\mathcal{H}_{\text{ML}}$, and $\mathcal{H}_{\text{retrieval}}$, respectively

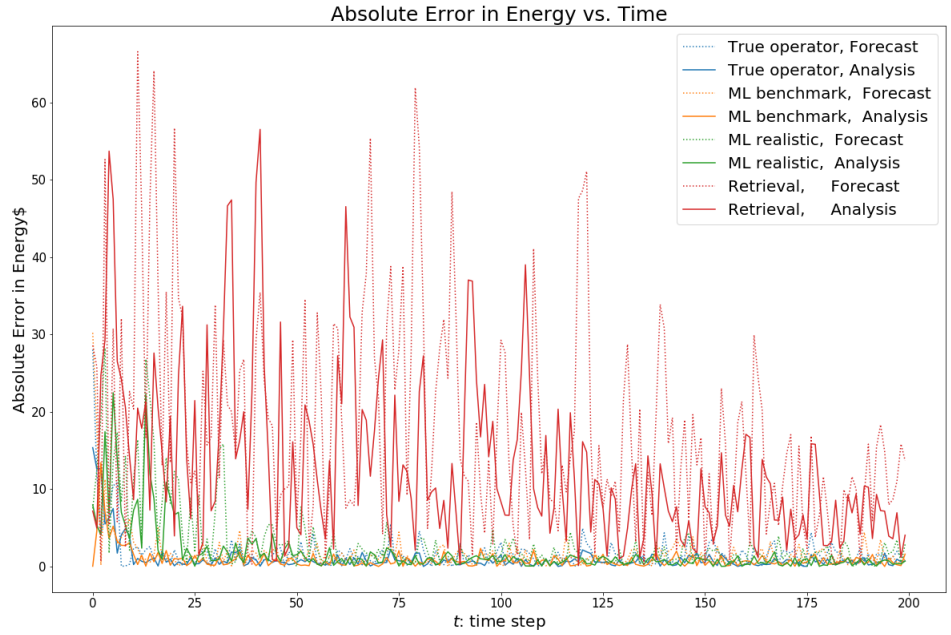

Figure 4.8: Time evolution of absolute error of E with $\mathcal{H}$, $\hat{\mathcal{H}}_{\text{ML}}$, $\mathcal{H}_{\text{ML}}$, and $\mathcal{H}_{\text{retrieval}}$, respectively

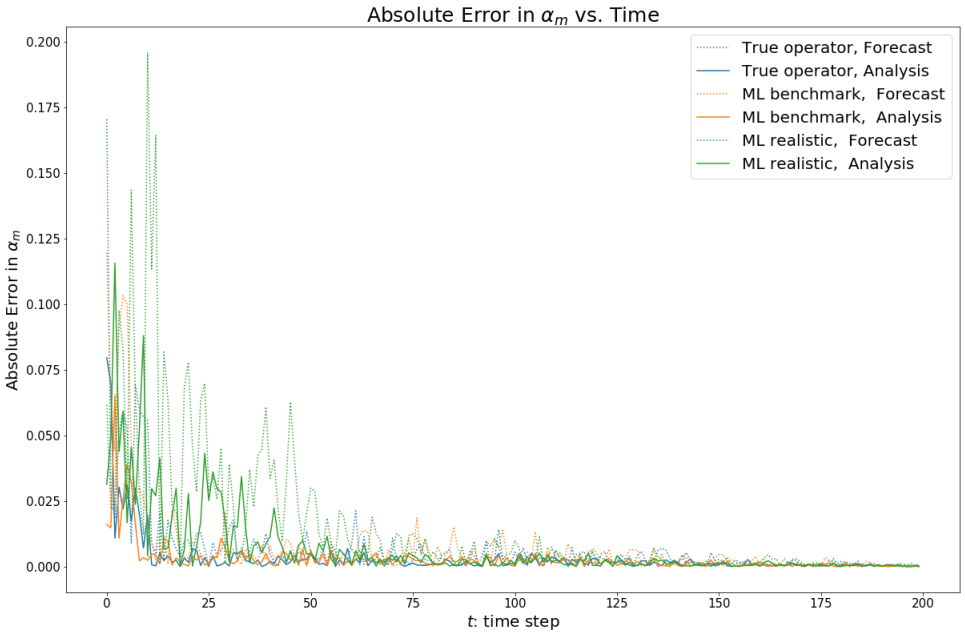

Figure 4.9: Time evolution of absolute error of α_m with $\mathcal{H}$, $\hat{\mathcal{H}}_{\text{ML}}$, and $\mathcal{H}_{\text{ML}}$, respectively

is expected since, initially, the system is in transient states where the machine learning algorithm does not have sufficient data to learn the true observation operator. After a while, the error of $\mathcal{H}_{\text{ML}}$ decreases to roughly the same as $\mathcal{H}$ and $\hat{\mathcal{H}}_{\text{ML}}$.

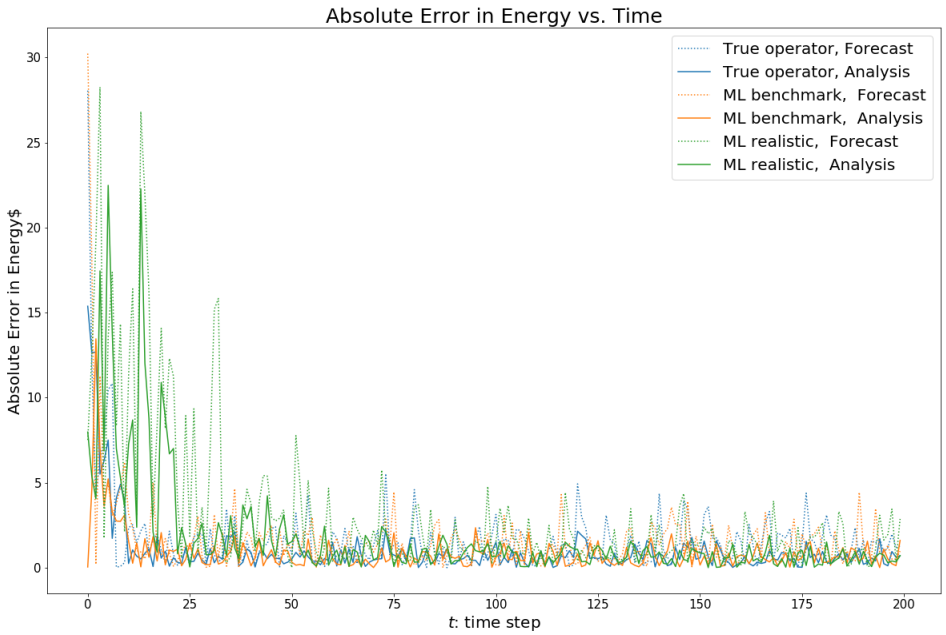

Figure 4.10: Time evolution of absolute error of E with $\mathcal{H}$, $\hat{\mathcal{H}}_{\mathrm{ML}}$, and $\mathcal{H}_{\mathrm{ML}}$, respectively

CHAPTER 5

Discussion

5.1 Conclusions

In this study, we built a test bed model proxy in Chapter 2 for the melt pond problem in sea ice concentration retrieval, along with proxies for ice and pond concentration and satellite radiances. Our model roughly mimics the actual physical processes and give us interesting behaviors to investigate.

We generated several data sets to train machine learning observation operators for different purposes in Section 3.1. The machine learning algorithm approximates the true observation operator reasonably well. With a sufficient amount of data, it is able to achieve roughly the same performance as the true observation operator. While its performance severely affected by the amount of data available, where in transient states, without sufficient data, the machine learning observation operator could make large errors, we were able to find a way of quantifying the accuracy with data density and averaged test error in the grided state space Section 3.3.1.

The DA experiments in Chapter 4 demonstrate that in our testbed model, assimilating directly on the proxy for satellite radiances with machine learning observation operator works well and far outperforms assimilating on the proxy for inaccurately retrieved sea ice concentration. Even for models trained with sparse data, the performance is still better than assimilating with wrong concentration. We can use the performance on the test set and data density to estimate the error covariance matrix of machine learning observation operator and apply covariance inflation accordingly at corresponding locations in the phase space.

In conclusion, for data assimilation in sea ice concentration with melt ponds, it is a desirable option to use historic data to construct a machine learning observation operator taking the state variables directly to the satellite radiances. Even if only a small amount of historic data is available

for training or the data is highly noisy, machine learning is still likely to offer a better solution than satellite retrieval algorithm.

5.2 Future Direction

The scope of the study is limited by the time we have, so we only tested with one machine learning algorithm, that is the neural network, and one data assimilation scheme, the ensemble Kalman filter. To fully explore the potential advantages and challenges in our idea, we will try using other machine learning algorithms like XGBoost and Random Forest. Furthermore, since our system is low dimensional and highly non-linear, the particle filter is a reasonable choice to do data assimilation with.

In this study, we demonstrated our idea on a simple testbed model. The first step is successful, and starting from here, we will test our idea on larger scale models that are actually used in the reality, e.g. The Community Sea Ice Model (CSIM) (Hunke and Lipscomb, 2010) along with an atmospheric radiative transfer model that computes the actual satellite radiances from state variables. The major challenges would be first, the dimension of the real systems is significantly larger, second the computational cost would be much higher, and third, it is in question whether machine learning would still be able to approximate the complicated atmospheric model.

The eventual goal of our study is to build a machine learning model that takes sea ice states to the satellite radiances. The first step is to gather data of matching instances where we know pond and open water fraction with passive microwave measurements we could generate a training set. MOSAiC - The Multidisciplinary Drifting Observatory for the Study of Arctic Climate is a mission which may provide some data like this.

BIBLIOGRAPHY

BAEK, S.-J., HUNT, B. R., KALNAY, E., OTT, E., and SZUNYOGH, I. (2006). Local ensemble kalman filtering in the presence of model bias. *Tellus A*, 58(3):293–306.

Calvo, M., Montijano, J. I., and Rández, L. (2016). Algorithm 968: Disode45: A matlab runge-kutta solver for piecewise smooth ivps of filippov type. *ACM Trans. Math. Softw.*, 43(3):25:1–25:14.

Eisenman, I. and Wettlaufer, J. S. (2009). Nonlinear threshold behavior during the loss of arctic sea ice. *Proceedings of the National Academy of Sciences*, 106(1):28–32.

Evensen, G. (2003). The ensemble kalman filter: theoretical formulation and practical implementation. *Ocean Dynamics*, 53(4):343–367.

F. Filippov, A. (1960). *Differential Equations with Discontinuous Right-Hand Side*, volume 2.

Gillijns, S., Mendoza, O. B., Chandrasekar, J., De Moor, B. L. R., Bernstein, D. S., and Ridley, A. (2006). What is the ensemble kalman filter and how well does it work? In *2006 American Control Conference*, pages 6 pp.–.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10). Society for Artificial Intelligence and Statistics*.

Hunke, E., Lipscomb, W., Jones, P., Turner, A., Jeffery, N., and Elliott, S. (2017). Cice, the los alamos sea ice model, version 00.

Hunke, E. C. and Lipscomb, W. H. (2010). CICE: The Los Alamos Sea Ice Model, Documentation and Software User's Manual, Version 4.1. Technical Report LA-CC- 06-012, Los Alamos National Laboratory, Los Alamos, New Mexico. available at http://oceans11.lanl.gov/trac/CICE (last access: 6 February 2014).

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167.

Kern, S., Rosel, A., Pedersen, L. T., Ivanova, N., Saldo, R., and Tonboe, R. T. (2016). The impact of melt ponds on summertime microwave brightness temperatures and sea-ice concentrations. *The Cryosphere*, 10(5):2217–2239.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *ICLR*.

Kongoli, C., Boukabara, S.-A., Yan, B., Weng, F., and Ferraro, R. (2011). A new sea-ice concentration algorithm based on microwave surface emissivities application to AMSU measurements. *IEEE Transactions on Geoscience and Remote Sensing*, 49(1):175–189.

Lorenz, E. (1995). Predictability: a problem partly solved. In *Seminar on Predictability, 4-8 September 1995*, volume 1, pages 1–18, Shinfield Park, Reading. ECMWF, ECMWF.

OTT, E., HUNT, B. R., SZUNYOGH, I., ZIMIN, A. V., KOSTELICH, E. J., CORAZZA, M., KALNAY, E., PATIL, D. J., and YORKE, J. A. (2004). A local ensemble kalman filter for atmospheric data assimilation. *Tellus A*, 56(5):415–428.

Scott, K., Buehner, M., Caya, A., and Carrieres, T. (2012). Direct assimilation of amsr-e brightness temperatures for estimating sea ice concentration. *Monthly Weather Review*, 140:997–1013.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Thorndike, A. S., Rothrock, D. A., Maykut, G. A., and Colony, R. (1975). The thickness distribution of sea ice. *Journal of Geophysical Research*, 80(33):4501–4513.

Willmes, S., Nicolaus, M., and Haas, C. (2014). The microwave emissivity variability of snow covered first-year sea ice from late winter to early summer: a model study. *The Cryosphere*, 8(3):891–904.