A Kalman Filter Model with Sparse Matrices in Spatial-Temporal Prediction

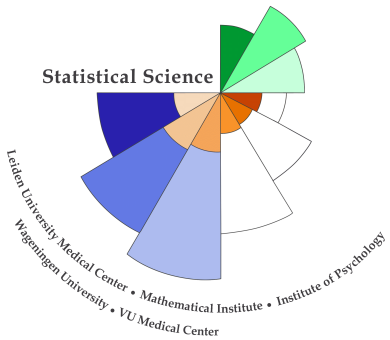
Bingjing Gu

Thesis advisor: Dr. Tim van Erven

MASTER THESIS

Defended on May 31st, 2017

STATISTICAL SCIENCE
FOR THE LIFE AND BEHAVIOURAL SCIENCES



Abstract

The Kalman filter has numerous applications in spatial-temporal prediction. A common application is for guidance, navigation, and control of vehicles, particularly aircraft and spacecraft. [1] In this thesis, we focus on one typical spatial-temporal data type of discrete time and discrete space. We consider a rectangular grid for the space domain. We make a first order Markov property assumption in both time and space to reduce complexity. In addition, several input control features are introduced into the Kalman filter. In other words, the distribution of future states depends only on the current states and input control features in their own area and their neighboring areas.

Under our Markov assumption, it is natural for the transition matrix in the Kalman filter to be sparse for spatial-temporal data where sparse transition matrices with constrained structure are designed to interpret the spatial correlation among all the areas. We will derive the equations for inference in this particular spatial system, namely the Kalman filter and Kalman smoother. Using the results for the Kalman filter and Kalman smoother, we further consider the determination of the parameters of the Kalman filter model through a modified Expectation–Maximization(EM) algorithm that estimates sparse transition matrices. This stands in contrast with the standard EM algorithm, which usually produces a dense estimate for the matrices. To respect the spatial pre-constrained sparsity structure, we specify greedy EM updates that work on rows of the transition matrix.

We study the properties of our new method in simulations and apply the method to a real data set on aviation safety where the goal is to predict which areas at Schiphol airport are at risk of having a large density of birds in the near future.

Acknowledgements

I would first like to thank my thesis supervisor Tim van Erven. He consistently spent all the dozens of hours on our discussions and gave me valuable painstaking suggestions and feedbacks with patience. I really appreciate the inspiring way during the discussions which not only refines my knowledge but also expands my thinking space.

I would also like to thank Data Innovation Lab in Schiphol Airport, Diederik, Michel, Stijn and Rok, for giving me internship opportunity and allowing me to further use the bird data in my thesis project. Their passion and professional competence stimulated my curiosity and promoted my learning on Machine Learning area.

Stackoverflow is always accompanying me during the whole thesis project.

Finally, I must express my very profound gratitude to my parents for providing me with unfailing support and continuous encouragement throughout my years of this Master Program. Thank you for a big amount of financial sponsorship. This accomplishment would not have been possible without them.

Contents

Chapter 1

Introduction

1.1 Sparse Kalman Filter

1.1.1 Simple Kalman Filter

A sequence of spatial temporal data could be modeled in a Kalman filter model. Such an approach would exploit the sequential patterns in the data, such as correlations between true values that are consecutive in the sequence. The true spatial temporal state at time t and location s denoted by $y(t, s)$ is related to the previous true state through a linear relationship. Given that the true system is unknown and considering that the dynamical field is observed at every time t and locations s referring to $z(t, s)$, the simplest model of a time-varying state is

$$y(t, s) = A_s y(t-1, s) + w_{t-1,s} \tag{1.1}$$

$$z(t, s) = y(t, s) + v_{t,s} \tag{1.2}$$

where $w_{t-1,s} \sim N(0, Q_s)$ and $v_{t,s} \sim N(0, R_s)$ are the white noise of the system. Since the model describes the relationship in each location, A_s, Q_s and R_s are all scalars.

This model is the simplest Kalman filter model containing temporal relation, but without input control variable and any spatial assumptions. As all geographic phenomena evolve over time, both spatiality and temporality are central to the understanding of geographic process and events. [8] A joint analysis of spatial and temporal data is preferable to combine the temporal correlation and the spatial correlation to the largest extent in one stage. Therefore, we investigate an integration field of a spatial-temporal process in both domains at the same time.

1.1.2 Spatial-temporal Data Assumption

In order to introduce the spatial influence into the Kalman Filter model, we first give the spatial temporal data assumption.

Here we consider one of the applications from a practical perspective in areal data type. The areal data refers to a finite number of areal units with well defined boundaries

partitioned by an entire given space area $\mathcal{L}$, for instance: counties, provinces, cities. [18] To be more precise, the given spatial area shall be rectangle so that we could design a grid segmentation to cover all the given space area $\mathcal{L} \subset \mathbb{R}^2$. Given a finite $l \times m$ grid $s = (i, j)$ where $i = 1, 2, \ldots, l$ and $j = 1, 2, \ldots, m$ represent each cell in this grid, we determine a set of neighbourhood cells $N((i, j))$ as

$$N((i,j)) = \{(i',j') : |i - i'| \leq 1, |j - j'| \leq 1; i' \in 1, 2, \ldots, l; j' \in 1, 2, \ldots, m\} \tag{1.3}$$

In accordance with this definition, one complete set of neighbourhood cells of a given cell in the grid contains nine cells in total which include the eight horizontal, vertical and diagonal adjacent cells and the cell itself. After that, we define a core research region which is an area with particular importance or at risk to be researched. Briefly, it could be treated as a crucial research area in the experiment. We shall always pick up cells which are not in the edges of the entire grid to make up the core region $\mathcal{S}$ so that we could make sure each cell in the core region $\mathcal{S}$ would contain the most complete nine cells in its neighbourhood cells set and there is no missing neighbourhood information in it. Namely, the core region is the subset of a set of cells with nine neighbourhood cells in the given space area $\mathcal{L}$. In general, we ought to determine the core region $\mathcal{S}$ during the first stage and then extend the area boundaries until it meets predetermined requirements of the given area $\mathcal{L}$. Therefore, the whole given area $\mathcal{L}$ is modeled in Kalman filter model, but we only care about the predictions in core region because the cells in the edge of the whole given area $\mathcal{L}$ lack the necessary complete neighborhood information but still give the neighborhood information of the cells in the core region.

Tobler's first law of geography suggests that everything is related to everything else, but near things are more related than distant things. [7] In addition, Markov property referring to the memory loss property of a stochastic process could also be taken into account since the future states of a process which is conditional on both past and present states depends only upon the present state, not on the sequence of events that preceded it. [9] In terms of theses two basic concepts, we give the assumptions in the relationship of the spatial temporal data that the observations in cell $s = (i, j)$ at time $t + 1$ is only related to the set of its neighbourhood cells $N((i, j))$ one time step before, namely time t in a given $m \times n$ grid area $\mathcal{L}$ and its core region $\mathcal{S}$

An intuitive example is being drawn in Figure 1.1. Suppose that the interval time is 1, Figure 1.1 shows how the connection among the observations in a given cell at time $t+1$ is set up by those features from the neighbourhood cells set of the given cell at time t. The hollow circle in the middle cell represents the observation z in the middle cell at time $t + 1$ and the nine solid circles at time t stand for those features vectors which include the observation, the spatial features and non-spatial features of each element in the neighbourhood cells set at time t. The spatial features are spatially varying at each time while non-spatial features share the same value in a certain spatial area at same time. By using these nine features vectors at time t, we could assume that the future value of the observation at location s depends not only on the previous values

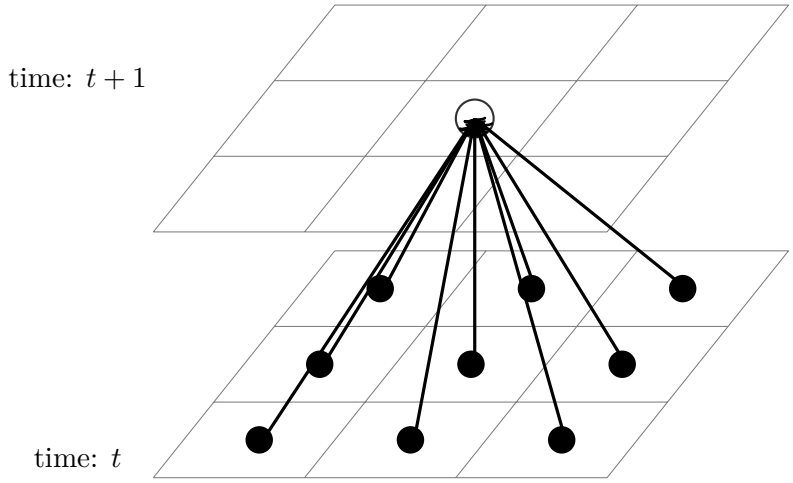


Figure 1.1: Example of spatial-temporal neighborhood relationship assumption for a given cell

of the feature variables at the same location, but also on the past values of adjacent neighborhood locations. Iteratively applying the rule, we would have a spatial temporal dynamical process.

1.1.3 General Sparse Kalman Filter

Now we would upgrade the simplest Kalman filter model on top of this spatial temporal assumption. At first, we put all the locations into one vector and shift the notation of true states and observations into $\boldsymbol{y}(t)$ and $\boldsymbol{z}(t)$.

$$\boldsymbol{y}(t) = \begin{bmatrix} \boldsymbol{y}(t,(1,1)) \\ \vdots \\ \boldsymbol{y}(t,(l,m)) \end{bmatrix}$$

$$\boldsymbol{z}(t) = \begin{bmatrix} \boldsymbol{z}(t,(1,1)) \\ \vdots \\ \boldsymbol{z}(t,(l,m)) \end{bmatrix}$$

where the vector contains all the separated cells after grid segmentation in the given space area $\mathcal{L}$.

In the meanwhile, scalar A_s is extended to a transition matrix A which represents certain spatial neighborhood correlations among all the cells. Then we consider the external effect and put all spatial features and non-spatial features into input control features $\boldsymbol{u}(t)$. The control matrix B with certain spatial neighborhood correlations is to describe the external effect. We would design the matrix A and matrix B into

a particular structured sparse matrix to follow the rules of spatial relationship where A and B are parameter matrices for transition and newly introduced input control features respectively. According to this settings, we could make sure that the true states at current time are only related to the set of its neighbourhood states one time step previously even if we do not get enough data and information to estimate this spatial relation pattern. Therefore, we have a complete sparse Kalman filter model in all locations:

$$\boldsymbol{y}(t) = A\boldsymbol{y}(t-1) + B\boldsymbol{u}(t-1) + w_{t-1} \tag{1.4}$$

$$\boldsymbol{z}(t) = \boldsymbol{y}(t) + v_t \tag{1.5}$$

$$w_{t-1} \sim N(0, Q) \tag{1.6}$$

$$v_t \sim N(0, R) \tag{1.7}$$

The details of sparse Kalman Filter model will be discussed in Chapter Two.

After giving the inference of sparse Kalman filter, we would utilize the expectation-maximization algorithm which is an elegant and powerful method for finding maximum likelihood parameters A, B, Q, R for models with latent variables [4]. However a general EM algorithm would give a non-sparse estimate for matrix A and matrix B, which would violate the spatial relationship assumption. Thus, in this thesis we propose a modified Expectation–Maximization algorithm to learn the parameters of the system by combining quantities from Kalman filter and Kalman smoother to respect the given sparsity structure. All code in this project are performed in the R-software environment and the R code has been provided in Github (https://github.com/sangaj/Thesis), along with some mathematical derivation details in Appendix.

1.2 Literature Review

This section covers literature review in the spatial temporal prediction model. As mentioned before, there are many data types in space and time domain. We only focus on the application on discrete types of data both in time and space dimension. A category of the method to deal with it is introducing Markov system in the sequential data since it is impractical to consider future dependence on all previous observations in time variant sequence.

J.F.Mari et al [9] propose a high order hidden Markov model to mine temporal and spatial data. In their agricultural landscape case, they propose a Markov Random Fields modelling the neighbourhood dependencies of land use categories on space domain. The land use categories at time t depend upon the former land use categories at previous times. They do not explicitly state the reciprocal dynamical influence on the space as time went by. They implement a hierarchical second order Hidden Markov Model where a temporal analysis is performed in order to identify temporal regularities before locating these regularities in the landscape by the concept of Markov Random Fields. Their hierarchical Hidden Markov Model is used to segment the landscape into patches,

each of them being characterized by a temporal second order Hidden Markov Model. Although they give a hierarchical spatial temporal model, they do not take the space and time dimension into account simultaneously. Nevertheless, they show that an ordered sequence with spatial property could be adequately modelled by a Markov process.

Luis Sanchez et al [19] present an ensemble Kalman Filter in spatial temporal dynamic hierarchical model. They define the spatial relation by a convolution integral difference linear equation of first order so that the whole area could be extended regarding those observation locations which make the dynamics in discrete time and continuous space. The process is governed by a kernel mixture which makes it an attractive representation in the spatio-temporal context since it simultaneously interpolates spatially and predicts temporarily permitting the consideration of unknown fields in unobserved points of interest. The researchers implement a Monte Carlo approximation of the Kalman filter called Ensemble Kalman filter to solve the system. They begin to create n initial ensemble members and generate the predicted value of n members of the ensemble by the process. After that, they use synthetic observations and Bayes linear fit to update the tied n members of ensemble. This exclusive methodology on spatial temporal data allows a large database to be split into smaller ones for separate evaluation and eventual combination of the individual results and leads to reduce the computational cost of the covariance matrix.

1.3 Structure of the Thesis

The current Chapter gives the framework in one of the spatial temporal data I confront in this project. The literature review section gives some efficient methods to tackle this similar data structure. We also introduce our modified sparse Kalman filter approach in terms of the inspiration from those papers.

Chapter Two will introduce the general framework of our sparse Kalman filter model with corresponding Kalman filter and Kalman smoother on spatial temporal data in detail. The sparsity is from the spatial relation with Markov property.

In Chapter Three, the modified learning method on sparse Kalman filter dynamics by Expectation–Maximization algorithm will be introduced. It will show those equations and derivations step by step that maintains a pre-specified sparse structure in parameter matrices.

Chapter Four covers studying the property of the new modified EM algorithm for learning sparse Kalman filter and its comparison with the standard EM algorithm on simulated data.

Chapter Five will present the results in the application to Schiphol bird data and its discussion. All R code written for this thesis is available on Github.

The last Chapter, Chapter Six concludes some thoughts and discussion on the methods.

Chapter 2

Spatial Linear Dynamical System

In this chapter, we will introduce the general architecture of spatial linear dynamical system based on the type of spatial temporal data. The modified sparse Kalman filter and Kalman smoother will also be described in this chapter. A summary of the notations used in next several Chapters is in Appendix A.

2.1 System Model

The spatial linear dynamical system relates to two steps; One is the measurement, the other is the process. From a hierarchical model point of view, there are noise errors in both steps. Considering that the spatial temporal dynamical process satisfies the first-order Markov property both on spatial and temporal domain, we need to improve the estimates by reducing random noise error in one stage at the same time. In particular, we consider a linear-Gaussian state space model so that the continuous true state in all locations $\boldsymbol{y}(t)$, as well as the observed measurement variables $\boldsymbol{z}(t)$ have multivariate Gaussian distributions. Each pair of nodes $\boldsymbol{y}(t), \boldsymbol{z}(t)$ represents a linear-Gaussian true variable model for that particular observation. However, the true states $\boldsymbol{y}(t)$ are no longer treated as independent but now form a first-order Markov chain in both time and space dimension. [4] That is to say, given the true value $y(t-1,(i,j))$ from all locations $\boldsymbol{s} = (1,1),(2,1),\ldots,(i,j),\ldots$ where $\boldsymbol{s} \in \mathcal{L}$, $t \in \mathcal{T}$, the value of current state $y(t,(i,j))$ is independent of all the states in the area $\mathcal{L}$ that are not adjacent to the location (i,j) and prior to $t-1$.

Now, we rephrase the spatial temporal dynamical sequence by a linear stochastic difference functions of the following form:

$$\boldsymbol{y}(t) = A\boldsymbol{y}(t-1) + B\boldsymbol{u}(t-1) + w_{t-1} \tag{2.1}$$

$$\boldsymbol{z}(t) = \boldsymbol{y}(t) + v_t \tag{2.2}$$

$$w_{t-1} \sim N(0, Q) \tag{2.3}$$

$$v_t \sim N(0, R) \tag{2.4}$$

where $\boldsymbol{y}(t)$ is the true value (hidden state) vector and $\boldsymbol{z}(t)$ is the observation value vector in all locations. The entries in these two vectors represent the value in all labelled cells after grid segmentation in the given space area $\mathcal{L}$, denoted by $\boldsymbol{s} = \{(1,1), \ldots, (i,j), \ldots (l,m)\}$ where $l \times m$ is the size of the grid segmentation. Thus the vector $\boldsymbol{y}(t)$ and vector $\boldsymbol{z}(t)$ refer to all corresponding values in the given space area $\mathcal{L}$ after grid segmentation at time t respectively. The w and v represent the process noise and measurement noise which is zero-mean normally-distributed random variables with covariance matrices Q and R respectively. $\boldsymbol{u}(t)$ including spatial features and non-spatial features is the matrix with all n features at locations $\boldsymbol{s}$ at time t,

$$\boldsymbol{u}(t) = \boldsymbol{u}(t, \boldsymbol{s}) = \begin{bmatrix} \boldsymbol{u}_1(t) \\ \vdots \\ \boldsymbol{u}_n(t) \end{bmatrix}$$

where each entry in the matrix $\boldsymbol{u}_i(t)$ with $i = 1, \ldots, n$ is the vector with all locations at time t. The length of the row vector n is determined by the number of features.

The initial value $\boldsymbol{y}(1)$ is normally distributed with mean vector π_1 and variance matrix V_1. We assume $\boldsymbol{y}(1)$, w_1, w_2, $\ldots$, v_1, v_2, $\ldots$ are jointly Gaussian and independent. Since $\boldsymbol{y}(t)$ and $\boldsymbol{z}(t)$ are linear functions of $\boldsymbol{y}(1)$, w_1, w_2, $\ldots$, w_t, v_1, v_2, $\ldots$, v_t, we conclude that the joint distribution over all variables, as well as all marginal distributions and conditional distributions are all Gaussian.

The state transition matrix A in the difference equation (2.1) relates the value at the previous time step $t-1$ to the value at the current time step, which also represents a quantification of the spatial relationships between the cells. We design this matrix A as a structured sparse matrix where the zero value entries mean that they do not have any spatial correlation between two locations. The control input matrix B relates the optional control input to the true state. The size of the matrix B is determined by the number of the control input and the number of cells after segmentation of the given area $\mathcal{L}$. The zero value entries in matrix B have the same meaning as those in matrix A which refer to no spatial relationship on the input features. Strictly speaking, matrix B is the broadwise concatenation by several matrices with totally same structure as matrix A. Given the matrix $\mathcal{B}$ with totally same structure as matrix A, the augmented matrix B is equal to $(\mathcal{B}|\mathcal{B}|\ldots)$.

2.1.1 Motivating Schiphol Case in System

A practical motivating example is a bird mass prediction project for Schiphol Airport. Collisions between birds and aircraft, defined as bird strikes, which is on a take off or landing roll could have catastrophic losses to the aviation industry and airport in terms of damage and delays every year. [12] Schiphol Airport is responsible for the implementation of the bird control programme, including both habitat management and active bird control. [17] A predictive model may help them reinforce the effectiveness and purposiveness of the implementation in bird control.

Figure 2.1: Grid Segmentation Example

The runway Polderbaan in Schiphol Airport and its vicinity area regarded as the given area $\mathcal{L}$ could be divided into 6×4 grid. In the meantime, a 4×2 grid in the middle of the original given grid area $\mathcal{L}$ is considered as the core region at risk $\mathcal{S}$ to prevent the bird strikes. Each cell $s = (i, j)$ in the given area $\mathcal{L}$ is labelled by $s = (i, j), i = 1, 2, \ldots, 6, j = 1, 2, \ldots, 4$. The core region at risk $\mathcal{S}$ only contains $s = (i, j), i = 2, \ldots, 5, j = 2, 3$. The Figure 2.1 shows how the grid segmentation is determined in this surrounding area. The light grey area with light blue edge is the given space area $\mathcal{L}$ and the dark grey region with orange edge is the core region at risk $\mathcal{S}$.

To accomplish the goal of preventing and reducing the bird strikes in Schiphol Airport, we would like to predict the average bird mass of each cell in the core region at risk $\mathcal{S}$ at time $t + 1$ in location $s \in \mathcal{S}$ where $s = (i, j), i = 2, \ldots, 5, j = 2, 3$. The spatial features vector contains velocity, airspeed, heading, position, peak mass and mass correction. These features and the observed mean mass are all detected and generated by 3D Flex System provided by a bird detection system vendor RobinRadar. [13] The non-spatial features vector consists of visibility, cloud layer for coverage, average forecasted wind direction, average forecasted wind speed, average forecasted wind speed including gust, forecasted showers of rain, snow or thunderstorms, hours and time index (morning, afternoon, evening and night). These features are provided by Air Traffic Control the Netherlands which is the agency in charge of air traffic control in the airspace of the Netherlands. [14] The average bird mass in a given cell $s = (i, j) \in \mathcal{S}$ at time $t + 1$ is

predicted by the average bird mass, spatial features and non-spatial features in neighborhood cell set $N(s)$ at one time step advance t. The spatial features and non-spatial features together refer to the input variable $u_{t,s}$ in the system. Only those information in neighborhood cells set at one time step advance could affect the observed mean mass at current time.

Just to be clear, we assign each cell a numerical label starting from 01 to 24 from top left cell of the grid (24 cells in total in a 6×4 grid) in the designed Schiphol grid instead of i and j.

01	02	03	04
05	06	07	08
09	10	11	12
13	14	15	16
17	18	19	20
21	22	23	24

Figure 2.2: Example indicator for each cell in Schiphol case

Thus, the matrix A is made up of 24 vectors $\vec{a_{01}}$, $\vec{a_{02}}$, ..., $\vec{a_{24}}$. And each entry in the 24 vectors represents the relationships between the cells in the grid in accordance with the spatial Markov assumption. That is to say, the observations in one cell are only related to its neighbourhood cells in the given area $\mathcal{L}$ and its own cell one time step advance. For instance: the vector $\vec{a_{03}}$ is composed of eighteen zero entries which are the number of the non-adjacent cells and six non-zero entries which are the number of the adjacent cells. We would not take into account three of the potential adjacent cells which are not in the given area; The vector $\vec{a_{14}}$ is composed of all nine adjacent cells and other fifteen zero entries. Zero value in entries means there is no significant connection between two cells.

$$\vec{a_{03}} = (0, a_{0302}, a_{0303}, a_{0304}, 0, a_{0306}, a_{0307}, a_{0308}, \underbrace{0, \ldots, 0}_{16})$$

$$\vec{a_{14}} = (\underbrace{0, \ldots, 0}_{8}, a_{1409}, a_{1410}, a_{1411}, 0, a_{1413}, a_{1414}, a_{1415}, 0, a_{1417}, a_{1418}, a_{1419}, \underbrace{0, \ldots, 0}_{5})$$

Suppose that there are eight input variables in a 6×4 grid, each entry of the matrix B represent the parameters between the cells and input variables.

$$B = \begin{bmatrix} \mathcal{B}_1 & \mathcal{B}_2 & \ldots & \mathcal{B}_8 \end{bmatrix}$$

where $\mathcal{B}_i$, $i = 1, 2, \ldots, 8$ has the same matrix structure with A which means it is a sparse matrix following the spatial Markov assumptions. The eight same structured

sparse matrices are concatenated into the parameter matrix B so that the dimension of the matrix B in this example is 24×192.

In practice, the parameter matrices A and B, the process noise covariance matrix Q and measurement noise covariance matrix R might change with each time step or measurement, however here we assume they are constant. [2] The parameters $\theta = \{A, B, Q, R\}$ in the model can be determined by maximum likelihood through the Expectation–Maximization algorithm which will be discussed in Chapter 3. Before we apply the Expectation–Maximization algorithm to learn the parameters, we wish to solve the inference problem of making predictions of the next state and of the next observation by current and past observations in the next section which gives rise to the Kalman filter and Kalman smoother equations. Since the conditional distributions are normally-distributed for the system described by equations (2.1) to (2.4), so it suffices to find the mean and variance of the distribution. [3]

2.2 Kalman Filter & Kalman Smoothing

In the following, we will describe the general time-invariant Kalman Filter and Kalman Smoother with input controlled variables in standard vectors and matrices notations. Meanwhile, we would give those equations respectively.

In the beginning, we assume that the parameters of the system are fixed so that we could derive the Kalman Filter equations and Kalman Smoother equations step by step inspired and followed by [3], [6] and [20]. The Kalman Filter equations are generated under the current time and previous time, whilst we smooth the state by the Kalman Smoother equations if the future timeframe is available. In short, if we would like to estimate a true state given data sequence, then we recursively apply the Kalman filter equations until reaching some timestamp at first. After that, we execute a backward recursion by applying the Kalman smoother equations and quantities calculated from Kalman filter until reaching the time we plan to estimate. [6] The quantity on hidden state is the genuinely optimal estimate after traversing these two passes. Therefore, it is quite obvious that Kalman filter and Kalman smoother is one type of forward and backward algorithm. These two steps will be discussed in the next two sections.

In the next chapter, we will show how to employ the Kalman Filter and Kalman Smoother equations to efficiently estimate the parameters of the model from training data. Since the model has latent variables, this could be addressed using the Expectation–Maximization algorithm. [6]

2.2.1 Kalman Filter

The Kalman filter could be governed by the system from 2.1 to 2.4 which contains all the elements in a complete linear dynamical system. We will begin this section with addressing some notation to simplify the conditional expectation and conditional variance and show how the whole procedure works exactly in Figure 2.3

$$\{\boldsymbol{y}\}_{t_0:t_1} = \{\boldsymbol{y}(t_0), \boldsymbol{y}(t_0+1), \boldsymbol{y}(t_0+2), \ldots, \boldsymbol{y}(t_1)\} \tag{2.5}$$

$$\{\boldsymbol{z}\}_{t_0:t_1} = \{\boldsymbol{z}(t_0), \boldsymbol{z}(t_0+1), \boldsymbol{z}(t_0+2), \ldots, \boldsymbol{z}(t_1)\} \tag{2.6}$$

$$\{\boldsymbol{u}\}_{t_0:t_1} = \{\boldsymbol{u}(t_0), \boldsymbol{u}(t_0+1), \boldsymbol{u}(t_0+2), \ldots, \boldsymbol{u}(t_1)\} \tag{2.7}$$

$$P_t^{t^*} = E(\boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:t^*}, \{\boldsymbol{u}\}_{1:t^*}) \tag{2.8}$$

$$V_t^{t^*} = Var(\boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:t^*}, \{\boldsymbol{u}\}_{1:t^*}) \tag{2.9}$$

where $\{\boldsymbol{z}\}_{t_0:t_1}$ and $\{\boldsymbol{u}\}_{t_0:t_1}$ are arrays of the observations vector and input variables in the whole given area $\mathcal{L}$ from time t_0 to t_1 respectively. $\boldsymbol{y}(t)$ simplified by $\boldsymbol{y}(t, \boldsymbol{s})$ is the hidden state vector at time t in the area $\mathcal{L}$ and $P_t^{t^*}$ is the expected state vector at time t in the same area $\mathcal{L}$ given the observations vector and input variables from time one to time t^*. $V_t^{t^*}$ refers to the state covariance matrix at time t in the area $\mathcal{L}$ given the observations vector and input variables from time one to time t^*.

The Kalman filter recursion with initial value mean π_1 and variance V_1 involves two parts. In the process part, the time updated equations are responsible for projecting forward the current state in time and covariance estimates to obtain the new estimates of the state variables along with their uncertainties for the next time step. [2] This is the predicting equations based on the current and past observations. Once the outcome of the next measurement is observed, the measurement updated equations are responsible for the feedback incorporating a new measurement into the estimates to adjust them which is more accurate than simply using the time update equations. [2] This is the estimating equations based on the current and past observations after the measurement becoming available. The crucial point of this step is Kalman gain K_t which tells how much we would like to tune the estimates by a given measurement to correct the estimates. This quantity is calculated from the covariance between estimates and measurements in order to generate a better estimated uncertainty than either alone. Now we could go on and feed it back into another round of predicting and updating as many times as we like. This set of powerful mathematical equations could still work even when the precise state of the system is unknown. [2]

In the following two parts, we would elaborate those Kalman filter equations.

Time Updated Equations Before we derive the main interest state P_t^t and variance V_t^t given the time $1:t$ during the Kalman filter recursion, we firstly seek to calculate the predicted states P_t^{t-1} and its covariance V_t^{t-1}, expressed in terms of the previous state estimates P_{t-1}^{t-1} and covariance V_{t-1}^{t-1}. Apparently, this predict phase could be produced by the sequential nature of linear dynamical system described by equations (2.1) to (2.4).

$$\begin{aligned} P_t^{t-1} &= E(\boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:t-1}, \{\boldsymbol{u}\}_{1:t-1}) \\ &= E(A\boldsymbol{y}(t-1)|\{\boldsymbol{z}\}_{1:t-1}, \{\boldsymbol{u}\}_{1:t-1}) + E(B\boldsymbol{u}(t-1)|\{\boldsymbol{z}\}_{1:t-1}, \{\boldsymbol{u}\}_{1:t-1}) \\ &= AP_{t-1}^{t-1} + B\boldsymbol{u}(t-1) \end{aligned} \tag{2.10}$$

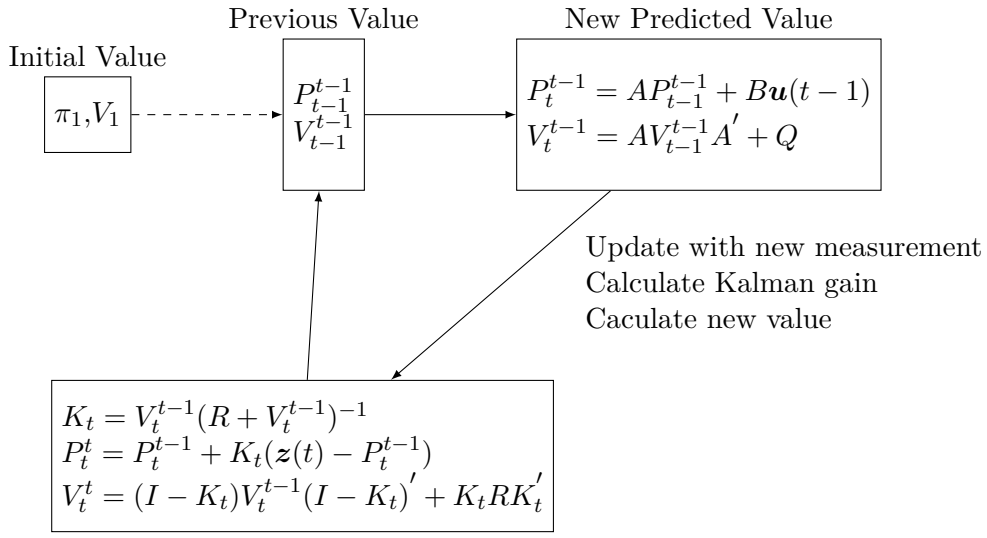


Figure 2.3: Flow chart of Kalman Filter

To find the time update equation for the variance, we use the fact that $A\boldsymbol{y}(t)$, $B\boldsymbol{u}(t)$ and w_t are independent conditional on $\{\boldsymbol{z}\}_{1:t-1}$.

$$\begin{aligned} V_t^{t-1} &= Var(A\boldsymbol{y}(t-1)|\{\boldsymbol{z}\}_{1:t-1}) + Var(B\boldsymbol{u}(t-1)|\{\boldsymbol{z}\}_{1:t-1}) + Var(w_{t-1}|\{\boldsymbol{z}\}_{1:t-1}) \\ &= AV_{t-1}^{t-1}A^{'} + Q \end{aligned} \tag{2.11}$$

These two equations form the time updated equations of the Kalman filter and provide one-step-ahead prediction on state estimates and estimated covariance.

Measurement Updated Equations When the new measurements arrive, we spontaneously combine the estimated state from the knowledge of prediction and the new measurements so as to update a more reliable quantity. The relative weight given to the measurement and current state estimate is Kalman gain K_t. With this Kalman gain, we get our main interest of each step's best estimated state P_t^t and variance V_t^t during the whole Kalman filter recursion. Therefore, the measurement updated equations are given as follows: [20]

$$\begin{aligned} K_t &= V_t^{t-1}(R + V_t^{t-1})^{-1} \\ P_t^t &= P_t^{t-1} + K_t(\boldsymbol{z}(t) - P_t^{t-1}) \\ V_t^t &= V_t^{t-1} - K_tV_t^{t-1} \end{aligned} \tag{2.12}$$

The interested reader could find the derivation of these equations from [20] or Appendix A. Equations (2.12) are providing the estimates correction after introducing the measurement quantities. However, the variance updated equation is a difference between

two positive definite matrices which may result in numerical instability and not being guaranteed to a positive definite matrix. [6] We would replace it with sum of two positive definite matrices:

$$V_t^t = (I - K_t)V_t^{t-1}(I - K_t)^{'} + K_t R K_t^{'} \tag{2.13}$$

[6]

Since the Kalman gain K_t is the ratio of the error in the estimates divided by the sum of error in estimates and error in the measurements, the determinate of Kalman gain K_t is between 0 and 1 (including 0 and 1). If the Kalman gain K_t is large, the new updated value puts more weights on the recent measurements, which means the measurements are more accurate because the measurement now contains less noise. On the other hand, the filter would follow the predictions more closely and ignore the measurement information because the measurement contains more noise if the gain is small.

2.2.2 Kalman Smoother

Next, we would like to give the backward pass with Kalman smoother equations. Since the filtered distributions are only conditional on the measurements obtained current and before, the estimates of the sequence hidden states are suboptimal. This implies that we are going to include all available measurements $\{\boldsymbol{z}\}_{1:T}$ where $T > t$ to improve the estimates of the hidden states so that the results is getting smoother and less noisy. [6] Like the derivation on seeking the state and variance of filtered distribution $P(\boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:t}, \{\boldsymbol{u}\}_{1:t})$, we would like to look for the state P_t^T and variance V_t^T of the smoothed distribution $P(\boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:T}, \{\boldsymbol{u}\}_{1:T})$ expressed in terms of the later state P_{t+1}^T and variance V_{t+1}^T. We are also interested in the covariance of the joint posterior distribution $P(\boldsymbol{y}(t+1), \boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:T}, \{\boldsymbol{u}\}_{1:T})$, denoted $V_{t+1,t}^T$. [3]

The Kalman Smoother recursion involves variance updated equations and mean updated equations. It normally proceeds from the last value we obtained from the Kalman filter by evaluating the influence of future measurements on the states in the past. Similarly, we make use of a smoother gain weight analogous to the Kalman gain K_T calculated by updated variance and predicted variance to refine the true states and variance. The flow chart of Kalman smoother in Figure 2.4 illustrates the whole process. The cross variance of the joint posterior distribution $V_{t+1,t}^T$ could also be generated during this recursion and this is a quite useful quantity in the parameter learning.

The derivation of equations in Figure 2.4 could be found in Appendix B or [20]. By now the input control features do not appear in those update estimate equations in Kalman smoother recursion so that those features will not change anything conceptually in the Kalman smoother procedure.

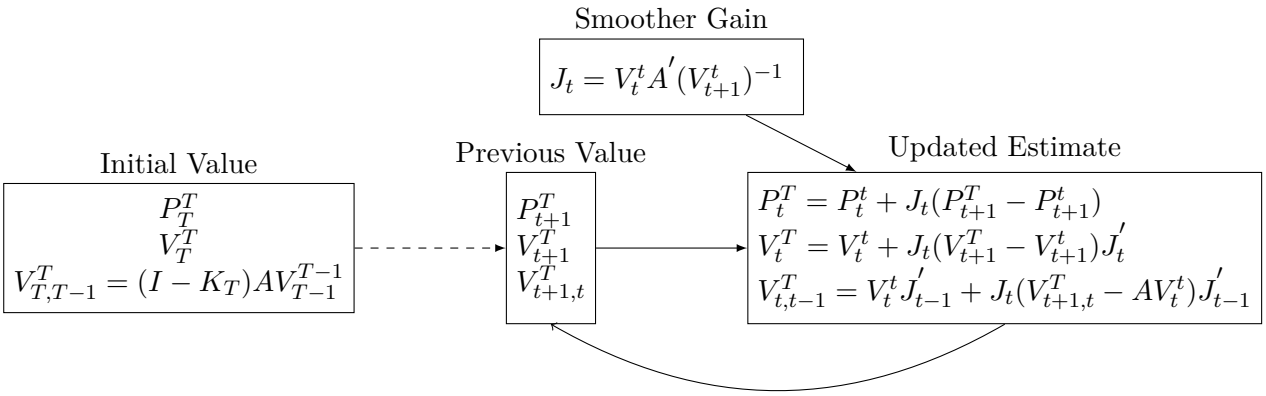


Figure 2.4: Flow chart of Kalman Smoother

2.3 Sparse Parameters in Equations

In standard Kalman filter model, many of the parameter matrices will in practice be determined by the linear relationship of physics phenomena such as location, speed and acceleration. In our particular case, we substitute geographic relationship matrix for physical transition matrix so that the spatial correlation could be introduced into the dynamics. The modified parameters matrix is constrained by the designed sparse spatial relationship and the matrix will be a sparse matrix to some extent. The layout of matrix structure could guarantee to keep the spatial relation in each time step during the whole evolution. This regulation is also valid to input control matrices that map control effect onto state vector. The sparse structure in parameters matrix do not display on the matrix notation so that they do not make any difference in the form of Kalman filter and Kalman smoother equations and still give the same expression as we described in the last two sections. However, this spatial property would influence parameter estimation of the Kalman filter model in the next chapter.

Chapter 3

Sparse Parameter Estimation by a Modified Greedy EM Algorithm

So far, we derived inference equations in the spatial linear dynamical system by knowing the parameters of the system. However, we need to turn to learn these parameters of the system in view of machine learning task. Given the observations $\{\boldsymbol{z}\}_{1:t}$, we would like to optimize the parameters $\theta = \{A, B, Q, R, \pi_1, V_1\}$. Because the model has latent variables, this can be addressed using the EM algorithm. [4] This involves finding the posterior distribution of the latent variables $P(\{\boldsymbol{y}\}_{1:t}|\{\boldsymbol{z}\}_{1:t}, \theta)$ and maximizing the likelihood of observed data under local posterior marginals. [15] Aside from it, we are modifying the standard EM algorithm to estimate the sparse matrices with certain pre-specified structure by incorporating greedy algorithm.

The standard Expectation Maximization algorithm is an iterative procedure for finding parameters that maximize the likelihood of observed data $P(\{\boldsymbol{z}\}_{1:T}|\theta)$ in the presence of latent states $\{\boldsymbol{y}\}_{1:T}$. [15] The crucial part of each iteration consists of two steps, the expectation step and the maximization step. The expectation step, abbreviated as E-step, calculates the expected value of the log likelihood function with respect to the posterior conditional distribution of latent variables given observations under the current estimate of the parameters. The maximization step, abbreviated as M-step, maximizes the formula in E-step with respect to the parameters to obtain a new setting of the θ. [4] The newly obtained parameters of the system are estimated by taking the corresponding partial derivative of the expected log-likelihood, setting to zero and solving for the maximizing parameter. Then we repeat the E-step and M-step until convergence. Since log function is a monotone function and the gradient of it is generally more well-scaled, we take the logarithm of likelihood instead. We define some tolerance parameter so that we could declare convergence if the increase in likelihood between successive iterations is smaller than the tolerance parameter.

Therefore, the whole procedure for learning parameters by Expectation–Maximization algorithm contains these four steps:

Step 1: Give initial value A,B,Q,R,π_1,V_1 to start the algorithm

Step 2: Implement the Kalman recursion given A,B,Q,R,π_1,V_1 and $\{\boldsymbol{z}\}_{1:T}$ to calculate the required quantities $E[\boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:T}]$, $E[\boldsymbol{y}(t)\boldsymbol{y}(t-1)'|\{\boldsymbol{z}\}_{1:T}]$ and $E[\boldsymbol{y}(t)\boldsymbol{y}(t)'|\{\boldsymbol{z}\}_{1:T}]$.

Step 3: Maximize the expected log likelihood Φ to find the updated new value of A,B,Q,R,π_1,V_1.

Step 4: Check the convergence criteria of the likelihood Φ, repeat the step (2) and step (3) until it has converged.

However, the particular sparse structure of the parameter matrices A and B is not put into consideration during the standard Expectation Maximization(EM) algorithm. If the matrices are optimized by the standard EM algorithm as is usually done, then non-sparse matrices will be found. In the following, we will address this issue by introducing a modification of EM algorithm to enforce the restriction that certain pre-specified elements in matrices A and B are zero. We substitute estimating each row of matrices for estimating the whole matrices by incorporating indicator matrices which are representing the spatial relationship to meet the sparse matrices assumption.

3.1 Expected Log-likelihood for Dynamical System

We start to consider the joint log likelihood of $\{\boldsymbol{z}\}_{1:T}$ and $\{\boldsymbol{y}\}_{1:T}$ on all available data. Given the observed value sequence $\{\boldsymbol{z}\}_{1:T}$ and the true value state sequence $\{\boldsymbol{y}\}_{1:T}$, we have the joint probability by Markov property:

$$P(\{\boldsymbol{z}\}_{1:T},\{\boldsymbol{y}\}_{1:T}) = P(\boldsymbol{y}(1))\prod_{t=2}^{T} P(\boldsymbol{y}(t)|\boldsymbol{y}(t-1))\prod_{t=1}^{T} P(\boldsymbol{z}(t)|\boldsymbol{y}(t)) \tag{3.1}$$

From the assumption of the spatial linear dynamical system, namely equations (2.1) to (2.4) and initialized value, each parts of the equation (3.1) are :

$$P(\boldsymbol{y}(1)) = \exp\left\{-\frac{1}{2}(\boldsymbol{y}(1)-\pi_1)'{V_1}^{-1}(\boldsymbol{y}(1)-\pi_1)\right\}(2\pi|V_1|)^{-\frac{1}{2}} \tag{3.2}$$

$$P(\boldsymbol{y}(t)|\boldsymbol{y}(t-1)) = \exp\left\{-\frac{1}{2}(\boldsymbol{y}(t)-A\boldsymbol{y}(t-1)-B\boldsymbol{u}(t-1))'Q^{-1}(\boldsymbol{y}(t)-A\boldsymbol{y}(t-1)-B\boldsymbol{u}(t-1))\right\} (2\pi|Q|)^{-\frac{1}{2}} \tag{3.3}$$

$$P(\boldsymbol{z}(t)|\boldsymbol{y}(t)) = \exp\left\{-\frac{1}{2}(\boldsymbol{z}(t)-\boldsymbol{y}(t))'R^{-1}(\boldsymbol{z}(t)-\boldsymbol{y}(t))\right\}(2\pi|R|)^{-\frac{1}{2}} \tag{3.4}$$

After taking the logarithm of (3.1) to get the log likelihood of joint distribution, we now take the quantities required for the M step which are the expected values,

with respect to the posterior distribution $P(\{\boldsymbol{y}\}_{1:t}|\{\boldsymbol{z}\}_{1:t},\theta)$, of the analogous sufficient statistics required for maximum likelihood estimation in the complete data case. [29] Then for $\theta = \{A, B, Q, R, \pi_1, V_1\}$ we have the expectation:

$$E[\log P(\{\boldsymbol{y}\}_{1:T}, \{\boldsymbol{z}\}_{1:T}|\theta, \{\boldsymbol{z}\}_{1:T})] \tag{3.5}$$

We could write out this expectation by notation Φ in a matrix form and drop the conditional part in expression for simplicity:

$$\begin{aligned}
\Phi &= E[\log P(\{\boldsymbol{y}\}_{1:T}, \{\boldsymbol{z}\}_{1:T}|\theta, \{\boldsymbol{z}\}_{1:T})] \\
&= -\frac{1}{2}(E[\boldsymbol{y}(1)^{'}V_1^{-1}\boldsymbol{y}(1)] - E[\boldsymbol{y}(1)^{'}V_1^{-1}\pi_1] - E[\pi_1^{'}V_1^{-1}\boldsymbol{y}(1)] + E[\pi_1^{'}V_1^{-1}\pi_1]) \\
&\quad - \frac{1}{2}\sum_{t=2}^{T}(E[\boldsymbol{y}(t)^{'}Q^{-1}\boldsymbol{y}(t)] - E[\boldsymbol{y}(t)^{'}Q^{-1}A\boldsymbol{y}(t-1)] - E[\boldsymbol{y}(t)^{'}Q^{-1}B\boldsymbol{u}(t-1)] \\
&\quad - E[(A\boldsymbol{y}(t-1))^{'}Q^{-1}\boldsymbol{y}(t)] + E[(A\boldsymbol{y}(t-1))^{'}Q^{-1}(A\boldsymbol{y}(t-1))] \\
&\quad + E[(A\boldsymbol{y}(t-1))^{'}Q^{-1}(B\boldsymbol{u}(t-1))] - E[(B\boldsymbol{u}(t-1))^{'}Q^{-1}\boldsymbol{y}(t)] \\
&\quad + E[(B\boldsymbol{u}(t-1))^{'}Q^{-1}(A\boldsymbol{y}(t-1))] + E[(B\boldsymbol{u}(t-1))^{'}Q^{-1}(B\boldsymbol{u}(t-1))]) \\
&\quad - \frac{1}{2}\sum_{t=1}^{T}(E[\boldsymbol{z}(t)^{'}R^{-1}\boldsymbol{z}(t)] - E[\boldsymbol{z}(t)^{'}R^{-1}\boldsymbol{y}(t)] - E[\boldsymbol{y}(t)^{'}R^{-1}\boldsymbol{z}(t)] \\
&\quad + E[\boldsymbol{z}(t)^{'}R^{-1}\boldsymbol{z}(t)]) \\
&\quad - \frac{1}{2}\log|V_1| - \frac{T}{2}\log|R| - \frac{T-1}{2}\log|Q| - T\log 2\pi
\end{aligned} \tag{3.6}$$

3.2 Maximize the Log-likelihood

The M step is to maximize (3.6) with respect to the parameters of the joint density. The parameters are estimated by taking the corresponding partial derivative of the expected log-likelihood, setting to zero and solving it. Before we take the partial derivative of Φ with respect to each parameter, we give some key identities on matrix derivatives with appropriate dimensions: [16]

$$\frac{\partial a^{'}c}{\partial a} = \frac{\partial c^{'}a}{\partial a} = c^{'} \tag{3.7}$$

$$\frac{\partial a^{'}Db}{\partial D} = \frac{\partial b^{'}D^{'}a}{\partial D} = ba^{'} \tag{3.8}$$

$$\frac{\partial \log|C|}{\partial C} = -\frac{\partial \log|C^{-1}|}{\partial C} = (C^{'})^{-1} = (C^{-1})^{'} \tag{3.9}$$

$$\frac{\partial a^{'}Ca}{\partial a} = \frac{\partial aC^{'}a^{'}}{\partial a} = 2a^{'}C \tag{3.10}$$

$$\frac{\partial a^{'} C^{-1} c}{\partial C} = -C^{-1} a c^{'} C^{-1} \tag{3.11}$$

$$\frac{\partial b^{'} D^{'} C D d}{\partial D} = d b^{'} D^{'} C + b d^{'} D^{'} C^{'} \tag{3.12}$$

3.2.1 The Equation for B

We first derive the new B from the equation (3.6) by taking derivatives with respect to B and equating to zero. We could drop out all terms which are not involving B since they could be absorbed into the additive constant and get zero in terms of the differentiation rule. To maintain the spatial assumption in the linear dynamical system, we ought to ensure that the B matrix is a sparse matrix under certain spatial relationship in each step. Generally, standard maximum likelihood estimates may not fulfill the spatial property in their results in each step. The approach we propose here is greedy algorithm by substituting estimating each row of matrix for estimating whole matrix to promise spatial condition with certainty.

We introduce diagonal indicator matrix $\boldsymbol{J}_s$ and $\boldsymbol{K}_s$ for each row of matrix A and B where $s = (i, j)$ represents one cell in the grid. Each indicator matrix follows the rule of the spatial assumption of corresponding cells.

Therefore, the matrix A and B could be transformed into this notation:

$$A = \begin{bmatrix} a_{\vec{(1,1)}} \boldsymbol{J}_{(1,1)} \\ \vdots \\ a_{\vec{(i,j)}} \boldsymbol{J}_{(i,j)} \\ \vdots \\ a_{\vec{(l,m)}} \boldsymbol{J}_{(l,m)} \end{bmatrix} \tag{3.13}$$

$$B = \begin{bmatrix} b_{\vec{(1,1)}} \boldsymbol{K}_{(1,1)} \\ \vdots \\ b_{\vec{(i,j)}} \boldsymbol{K}_{(i,j)} \\ \vdots \\ b_{\vec{(l,m)}} \boldsymbol{K}_{(l,m)} \end{bmatrix} \tag{3.14}$$

where $a_{\vec{(i,j)}}$ is a row vector whose length is the same as the number of cells and $b_{\vec{(i,j)}}$ is also a row vector whose length is the same as the number of cells times the number of input control features. The diagonal of each indicator matrix $\boldsymbol{J}_s$ is the vectors $a_{\vec{01}}$, $a_{\vec{02}}$, ... that we mentioned in Chapter 2. Those vectors also form the diagonal submatrices of $\boldsymbol{K}_s$ and those diagonal submatrices further generate the $\boldsymbol{K}_s$ in terms of the number of features. Instead of estimating matrix A and B, the row vectors $a_{\vec{(i,j)}}$ and $b_{\vec{(i,j)}}$ ought to be estimated by maximum likelihood. Because we multiply some components in vectors by 0 in the matrices $\boldsymbol{J}_s$ and $\boldsymbol{K}_s$, the maximum likelihood estimations are non-identifiable and have multiple solutions. We pick up one of them by using Moore–Penrose

pseudoinverse since we do not care which one we pick for the likelihood. The sparse indicator matrices would help us force some entries into zero no matter what we get. The Moore–Penrose pseudoinverse of arbitrary matrix O satisfying all of the following four criteria: $OO^+O = O$, $O^+OO^+ = O^+$, $(OO^+)^* = OO^+$, $(O^+O)^* = O^+O$. [30]

The derivation for matrix B is now decomposed into two steps. One is to take the partial derivative in terms of $\vec{b_s}$ where $s = (i, j)$ and then consolidate the row-wise results into a matrix B. The matrix derivatives identities and the fact $Q^{-1} = (Q^{-1})^{'}$ are applied to the procedure.

Lemma 3.2.1. *The update row vectors in matrix B are:*

$$\vec{b_s} = \sum_{t=2}^{T}(E[y(t,s)](\boldsymbol{K}_s\boldsymbol{u}(t-1))^{'} - \vec{a_s}\boldsymbol{J}_sE[\mathbf{y}(t-1)](\boldsymbol{K}_s\boldsymbol{u}(t-1))^{'})(\boldsymbol{K}_s\sum_{t=2}^{T}(\boldsymbol{u}(t-1)\boldsymbol{u}(t-1)^{'})\boldsymbol{K}_s^{'})^{+}$$

Proof.

$$\begin{aligned}
\frac{\partial\Phi}{\partial\vec{b_s}} &= -\frac{1}{2}\sum_{t=2}^{T}(-E[\frac{\partial(y(t,s)^{'}Q^{-1}\vec{b_s}\boldsymbol{K}_s\boldsymbol{u}(t-1))}{\partial\vec{b_s}}] + E[\frac{\partial((\vec{a_s}\boldsymbol{J}_s\boldsymbol{y}(t-1))^{'}Q^{-1}(\vec{b_s}\boldsymbol{K}_s\boldsymbol{u}(t-1)))}{\partial\vec{b_s}}] \\
&\quad - E[\frac{\partial((\vec{b_s}\boldsymbol{K}_s\boldsymbol{u}(t-1))^{'}Q^{-1}y(t,s))}{\partial\vec{b_s}}] + E[\frac{\partial((\vec{b_s}\boldsymbol{K}_s\boldsymbol{u}(t-1))^{'}Q^{-1}(\vec{a_s}\boldsymbol{J}_s\boldsymbol{y}(t-1)))}{\partial\vec{b_s}}] \\
&\quad + E[\frac{\partial((\vec{b_s}\boldsymbol{K}_s\boldsymbol{u}(t-1))^{'}Q^{-1}(\vec{b_s}\boldsymbol{K}_s\boldsymbol{u}(t-1))}{\partial\vec{b_s}}]) \\
&= -\frac{1}{2}\sum_{t=2}^{T}(-2\boldsymbol{K}_s\boldsymbol{u}(t-1)E[y(t,s)^{'}]Q^{-1} + 2\boldsymbol{K}_s\boldsymbol{u}(t-1)E[\boldsymbol{y}(t-1)^{'}](\vec{a_s}\boldsymbol{J}_s)^{'}Q^{-1} \\
&\quad + 2\boldsymbol{K}_s\boldsymbol{u}(t-1)(\boldsymbol{K}_s\boldsymbol{u}(t-1))^{'}\vec{b_s}^{'}Q^{-1})
\end{aligned} \tag{3.15}$$

Set equation (3.15) to zero, cancel out Q^{-1} by multiplying by Q, get rid of the $-\frac{1}{2}$ and transpose the whole equation, then multiply by the Moore–Penrose pseudoinverse of $\boldsymbol{K}_s\sum_{t=2}^{T}(\boldsymbol{u}(t-1)^{'}\boldsymbol{u}(t-1))\boldsymbol{K}_s^{'}$ to give the new $\vec{b_s}$ that maximizes the expectation Φ:

$$\sum_{t=2}^{T}(E[y(t,s)](\boldsymbol{K}_s\boldsymbol{u}(t-1))^{'} - (\vec{a_s}\boldsymbol{J}_s)E[\boldsymbol{y}(t-1)](\boldsymbol{K}_s\boldsymbol{u}(t-1))^{'} - \vec{b_s}\boldsymbol{K}_s\boldsymbol{u}(t-1)\boldsymbol{u}(t-1)^{'}\boldsymbol{K}_s^{'}) = 0$$

$$\vec{b_s} = \sum_{t=2}^{T}(E[y(t,s)](\boldsymbol{K}_s\boldsymbol{u}(t-1))^{'} - \vec{a_s}\boldsymbol{J}_sE[\boldsymbol{y}(t-1)](\boldsymbol{K}_s\boldsymbol{u}(t-1))^{'})(\boldsymbol{K}_s\sum_{t=2}^{T}(\boldsymbol{u}(t-1)\boldsymbol{u}(t-1)^{'})\boldsymbol{K}_s^{'})^{+} \tag{3.16}$$

□

The greedy algorithm gives the optimal solution for each row in a matrix, namely each cell. The matrix B is composed of these row wise optimal choices on top of equation (3.16). However, row wise association may not end up a local optimum for the whole matrix because this is a greedy procedure and optimizing the rows separately is not equivalent to optimizing all non-zero elements of matrix B together.

3.2.2 The Equation for A

We use the same strategy to derive the new A from the equation (3.6) by taking derivatives with respect to $\vec{a_s}$ and equating to zero. The matrix A are made up of row-wise estimations. We can drop out all terms which are not involving A since they can be absorbed into the additive constant and get 0 in terms of the differentiation rule. The matrix derivatives identities and the fact $Q^{-1} = (Q^{-1})'$ are applied to the procedure.

Lemma 3.2.2. *The update row vectors in matrix A are:*

$$\vec{a_s} = \sum_{t=2}^{T}(E[y(t,s)\mathbf{y}(t-1)'] - \vec{b_s}\boldsymbol{K}_s\boldsymbol{u}(t-1)E[\mathbf{y}(t-1)'])(\boldsymbol{J}_s\sum_{t=2}^{T}E[\mathbf{y}(t-1)\mathbf{y}(t-1)'])^{+}$$

Proof.

$$\begin{aligned}
\frac{\partial\Phi}{\partial\vec{a_s}} &= -\frac{1}{2}\sum_{t=2}^{T}(-E[\frac{\partial(y(t,s)'Q^{-1}\vec{a_s}\boldsymbol{J}_s\boldsymbol{y}(t-1))}{\partial\vec{a_s}}] - E[\frac{\partial((\vec{a_s}\boldsymbol{J}_s\boldsymbol{y}(t-1))'Q^{-1}y(t,s))}{\partial\vec{a_s}}] \\
&\quad + E[\frac{\partial((\vec{a_s}\boldsymbol{J}_s\boldsymbol{y}(t-1))'Q^{-1}(\vec{a_s}\boldsymbol{J}_s\boldsymbol{y}(t-1)))}{\partial\vec{a_s}}] + E[\frac{\partial((\vec{a_s}\boldsymbol{J}_s\boldsymbol{y}(t-1))'Q^{-1}(\vec{b_s}\boldsymbol{K}_s\boldsymbol{u}(t-1)))}{\partial\vec{a_s}}] \\
&\quad + E[\frac{\partial((\vec{b_s}\boldsymbol{K}_s\boldsymbol{u}(t-1))'Q^{-1}(\vec{a_s}\boldsymbol{J}_s\boldsymbol{y}(t-1)))}{\partial\vec{a_s}}]) \\
&= -\frac{1}{2}\sum_{t=2}^{T}(-2\boldsymbol{J}_sE[\boldsymbol{y}(t-1)y(t,s)']Q^{-1} + 2\boldsymbol{J}_sE[\boldsymbol{y}(t-1)'\boldsymbol{y}(t-1)](\vec{a_s}\boldsymbol{J}_s)'Q^{-1} \\
&\quad + 2\boldsymbol{J}_sE[\boldsymbol{y}(t-1)](\vec{b_s}\boldsymbol{K}_s\boldsymbol{u}(t-1))')Q^{-1}
\end{aligned} \tag{3.17}$$

Set equation (3.17) to zero, cancel out Q^{-1} and $\boldsymbol{J}_s$ by multiplying by Q and $\boldsymbol{J}_s^{-1}$ respectively, get rid of the $-\frac{1}{2}$ and transpose the whole equation, then multiply the right inverse of $\boldsymbol{J}_s\sum_{t=2}^{T}E[\boldsymbol{y}(t-1)\boldsymbol{y}(t-1)']$ to give the new $\vec{a_s}$ that maximizes the expectation Φ

$$\begin{aligned}
&\sum_{t=2}^{T}(E[y(t,s)\boldsymbol{y}(t-1)'] - \vec{a_s}J_sE[\boldsymbol{y}(t-1)\boldsymbol{y}(t-1)'] - \vec{b_s}K_s\boldsymbol{u}(t-1)E[\boldsymbol{y}(t-1)']) = 0 \\
&\vec{a_s} = \sum_{t=2}^{T}(E[y(t,s)\boldsymbol{y}(t-1)'] - \vec{b_s}K_s\boldsymbol{u}(t-1)E[\boldsymbol{y}(t-1)'])(J_s\sum_{t=2}^{T}E[\boldsymbol{y}(t-1)\boldsymbol{y}(t-1)'])^{+}
\end{aligned} \tag{3.18}$$

□

We use same strategy as updating matrix B to construct matrix A. Matrix A and B do not have particular order to estimate. The experiment shows that order reversal do not change the results a lot.

3.2.3 The Equation for Q

After we get the update value of matrix A and B, we can determine Q step by step through each of the likelihood terms and take the derivative of Φ with respect to Q. We drop out all terms which are not involving Q since they can be absorbed into the additive constant and get 0 in terms of the differentiation rule. It is not necessary to decompose matrix Q into its rows since it does not require a specific matrix structure. The matrix derivatives identities and the fact $Q^{-1} = (Q^{-1})^{'}$ are applied to the procedure

Lemma 3.2.3. *The update equation for matrix Q is:*

$$\begin{aligned}
Q = (1-T)^{-1}\sum_{t=2}^{T}(&-E[\mathbf{y}(t)\mathbf{y}(t)^{'}] + E[\mathbf{y}(t)\mathbf{y}(t-1)^{'}]A^{'} + E[\mathbf{y}(t)]\boldsymbol{u}(t-1)^{'}B^{'} \\
&+ AE[\mathbf{y}(t-1)\mathbf{y}(t)^{'}] - AE[\mathbf{y}(t-1)\mathbf{y}(t-1)^{'}]A^{'} - AE[\mathbf{y}(t-1)]\boldsymbol{u}(t-1)^{'}B^{'} \\
&+ B\boldsymbol{u}(t-1)E[\mathbf{y}(t)^{'}] - B\boldsymbol{u}(t-1)E[\mathbf{y}(t-1)^{'}]A^{'} \\
&- B\boldsymbol{u}(t-1)\boldsymbol{u}(t-1)^{'}B^{'})
\end{aligned}$$

Proof.

$$\begin{aligned}
\frac{\partial\Phi}{\partial Q} = -\frac{1}{2}\sum_{t=2}^{T}(&E[\frac{\partial(\boldsymbol{y}(t)^{'}Q^{-1}\boldsymbol{y}(t))}{\partial Q}] - E[\frac{\partial(\boldsymbol{y}(t)^{'}Q^{-1}A\boldsymbol{y}(t-1))}{\partial Q}] \\
&- E[\frac{\partial(\boldsymbol{y}(t)^{'}Q^{-1}B\boldsymbol{u}(t-1))}{\partial Q}] - E[\frac{\partial((A\boldsymbol{y}(t-1))^{'}Q^{-1}\boldsymbol{y}(t))}{\partial Q}] \\
&+ E[\frac{\partial((A\boldsymbol{y}(t-1))^{'}Q^{-1}A\boldsymbol{y}(t-1))}{\partial Q}] + E[\frac{\partial((A\boldsymbol{y}(t-1))^{'}Q^{-1}B\boldsymbol{u}(t-1))}{\partial Q}] \\
&- E[\frac{\partial((B\boldsymbol{u}(t-1))^{'}Q^{-1}\boldsymbol{y}(t))}{\partial Q}] + E[\frac{\partial((B\boldsymbol{u}(t-1))^{'}Q^{-1}(A\boldsymbol{y}(t-1)))}{\partial Q}] \\
&+ E[\frac{\partial((B\boldsymbol{u}(t-1))^{'}Q^{-1}(B\boldsymbol{u}(t-1)))}{\partial Q}]) - \frac{\partial(\frac{T-1}{2}\log|Q|)}{\partial Q} \\
= -\frac{1}{2}\sum_{t=2}^{T}&Q^{-1}(-E[\boldsymbol{y}(t)\boldsymbol{y}(t)^{'}] + E[\boldsymbol{y}(t)\boldsymbol{y}(t-1)^{'}]A^{'} + E[\boldsymbol{y}(t)]\boldsymbol{u}(t-1)^{'}B^{'} + \\
&AE[\boldsymbol{y}(t-1)\boldsymbol{y}(t)^{'}] - AE[\boldsymbol{y}(t-1)\boldsymbol{y}(t-1)^{'}]A^{'} - AE[\boldsymbol{y}(t-1)]\boldsymbol{u}(t-1)^{'}B^{'} + \\
&B\boldsymbol{u}(t-1)E[\boldsymbol{y}(t)^{'}] - B\boldsymbol{u}(t-1)E[\boldsymbol{y}(t-1)^{'}]A^{'} - B\boldsymbol{u}(t-1)\boldsymbol{u}(t-1)^{'}B^{'})Q^{-1} - \frac{T-1}{2}Q^{-1}
\end{aligned} \tag{3.19}$$

Set equation (3.19) to zero, cancel out Q^{-1} by multiplying by Q twice, get rid of the $\frac{1}{2}$ to give the new Q that maximizes the expectation Φ:

$$
\begin{aligned}
Q = (1-T)^{-1} \sum_{t=2}^{T} (&-E[\boldsymbol{y}(t)\boldsymbol{y}(t)^{'}] + E[\boldsymbol{y}(t)\boldsymbol{y}(t-1)^{'}]A^{'} + E[\boldsymbol{y}(t)]\boldsymbol{u}(t-1)^{'}B^{'} \\
&+ AE[\boldsymbol{y}(t-1)\boldsymbol{y}(t)^{'}] - AE[\boldsymbol{y}(t-1)\boldsymbol{y}(t-1)^{'}]A^{'} - AE[\boldsymbol{y}(t-1)]\boldsymbol{u}(t-1)^{'}B^{'} \\
&+ B\boldsymbol{u}(t-1)E[\boldsymbol{y}(t)^{'}] - B\boldsymbol{u}(t-1)E[\boldsymbol{y}(t-1)^{'}]A^{'} \\
&- B\boldsymbol{u}(t-1)\boldsymbol{u}(t-1)^{'}B^{'}) \qquad (3.20)
\end{aligned}
$$

□

3.2.4 The Equation for R

We take the derivative of Φ with respect to R. We could drop out all terms which are not involving R since they could be absorbed into the additive constant and get 0 in terms of the differentiation rule. The matrix derivative identities and the fact $R^{-1} = (R^{-1})^{'}$ are applied to the procedure.

Lemma 3.2.4. *The update equation for matrix R is:*

$$R = T^{-1} \sum_{t=1}^{T} (E[\mathbf{z}(t)\mathbf{z}(t)^{'}] - E[\mathbf{z}(t)\mathbf{y}(t)^{'}] - E[\mathbf{y}(t)\mathbf{z}(t)^{'}] + E[\mathbf{y}(t)\mathbf{y}(t)^{'}])$$

Proof.

$$
\begin{aligned}
\frac{\partial \Phi}{\partial R} &= -\frac{1}{2} \sum_{t=1}^{T} (E[\frac{\partial(\boldsymbol{z}(t)^{'}R^{-1}\boldsymbol{z}(t))}{\partial R}] - E[\frac{\partial(\boldsymbol{z}(t)^{'}R^{-1}\boldsymbol{y}(t))}{\partial R}] - E[\frac{\partial(\boldsymbol{y}(t)^{'}R^{-1}\boldsymbol{z}(t))}{\partial R}] \\
&\quad + E[\frac{\partial(\boldsymbol{y}(t)^{'}R^{-1}\boldsymbol{y}(t))}{\partial R}]) - \frac{\partial \frac{T}{2}\log|R|}{\partial R} \\
&= \frac{1}{2} \sum_{t=1}^{T} R^{-1}(E[\boldsymbol{z}(t)\boldsymbol{z}(t)^{'}] - E[\boldsymbol{z}(t)\boldsymbol{y}(t)^{'}] - E[\boldsymbol{y}(t)\boldsymbol{z}(t)^{'}] + E[\boldsymbol{y}(t)\boldsymbol{y}(t)^{'}])R^{-1} - \frac{T}{2}R^{-1}
\end{aligned} \qquad (3.21)
$$

Set equation (3.21) to zero, cancel out R^{-1} by multiplying by R twice, get rid of the $\frac{1}{2}$ to give the new R that maximizes the expectation Φ:

$$R = T^{-1} \sum_{t=1}^{T} (E[\boldsymbol{z}(t)\boldsymbol{z}(t)^{'}] - E[\boldsymbol{z}(t)\boldsymbol{y}(t)^{'}] - E[\boldsymbol{y}(t)\boldsymbol{z}(t)^{'}] + E[\boldsymbol{y}(t)\boldsymbol{y}(t)^{'}]) \qquad (3.22)$$

□

3.2.5 The Equation for π_1 and V_1

For completeness, we continue the derivation as before and take the derivative of Φ with respect to π_1 and V_1 respectively which means we treat the initial states in the case of following unknown mean π_1 and variance V_1. The matrix derivatives identities and the fact $V^{-1} = (V^{-1})'$ are applied into the procedure.

Lemma 3.2.5. *The update equations for π_1 and V_1 are:*

$$\pi_1 = E[\mathbf{y}(1)]$$
$$V_1 = Var(\mathbf{y}(1))$$

Proof.

$$\begin{aligned}\frac{\partial \Phi}{\partial \pi_1} &= -\frac{1}{2}(-E[\frac{\partial \boldsymbol{y}(1)' V_1^{-1}\pi_1}{\partial \pi_1}] - E[\frac{\partial \pi_1' V_1^{-1}\boldsymbol{y}(1)}{\partial \pi_1} + E[\frac{\partial \pi_1' V_1^{-1}\pi_1}{\partial \pi_1}) \\ &= -\frac{1}{2}(-2E[\boldsymbol{y}(1)' V_1^{-1}] + 2\pi_1' V_1^{-1}) \end{aligned} \tag{3.23}$$

$$\begin{aligned}\frac{\partial \Phi}{\partial V_1} &= -\frac{1}{2}(E[\frac{\partial \boldsymbol{y}(1)' V_1^{-1}\boldsymbol{y}(1)}{\partial V_1}] - E[\frac{\partial y(1)' V_1^{-1}\pi_1}{\partial V_1}] - E[\frac{\partial \pi_1' V_1^{-1}\boldsymbol{y}(1)}{\partial V_1} + E[\frac{\partial \pi_1' V_1^{-1}\pi_1}{\partial V_1}) \\ &\quad - \frac{\partial \frac{1}{2}\log|V_1|}{\partial V_1} \\ &= \frac{1}{2}V_1^{-1}(E[\boldsymbol{y}(1)\boldsymbol{y}(1)'] - E[\boldsymbol{y}(1)\pi_1'] - E[\pi_1 \boldsymbol{y}(1)'] + E[\pi_1\pi_1'])V_1^{-1} - \frac{1}{2}V_1^{-1}\end{aligned} \tag{3.24}$$

Set equation (3.23) and (3.24) to zero, cancel out V_1^{-1} by multiplying by V_1 once for equation (3.23) and twice for equation (3.24), take the transpose for equation(3.23) and get rid of the $\frac{1}{2}$ to give the new π_1 and V_1 that maximizes the expectation Φ:

$$\pi_1 = E[\boldsymbol{y}(1)] \tag{3.25}$$

$$\begin{aligned}V_1 &= E[\boldsymbol{y}(1)\boldsymbol{y}(1)'] - E[\boldsymbol{y}(1)\pi_1'] - E[\pi_1 \boldsymbol{y}(1)'] + E[\pi_1\pi_1'] \\ &= Var(\boldsymbol{y}(1))\end{aligned} \tag{3.26}$$

□

The new π_1 is the expected value of $\boldsymbol{y}(1)$ given the data from time 1 to T while the new V_1 is the variance of $\boldsymbol{y}(1)$ conditional on the data from time 1 to T. Both of these values are the initial values in the Kalman recursions in Chapter 2.

The equations of new update value sets θ in (3.16), (3.18), (3.20), (3.22) reveals that the only quantities need to be prepared for calculation are $E[\boldsymbol{y}(t)]$, $E[\boldsymbol{y}(t)\boldsymbol{y}(t-1)']$ and $E[\boldsymbol{y}(t)\boldsymbol{y}(t)']$. These three quantities could be derived from the definition of expectation,

variance and covariance, and the exact value of these quantities are computed from the results of Kalman smoother recursions in Chapter 2.

$$E[\boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:T}] = P_t^T \tag{3.27}$$

$$E[\boldsymbol{y}(t)\boldsymbol{y}(t-1)'|\{\boldsymbol{z}\}_{1:T}] = V_{t,t-1}^T + P_t^T {P_{t-1}^T}' \tag{3.28}$$

$$E[\boldsymbol{y}(t)\boldsymbol{y}(t)'|\{\boldsymbol{z}\}_{1:T}] = V_t^T + P_t^T {P_t^T}' \tag{3.29}$$

where the rows of the matrix value $E[\boldsymbol{y}(t)]$ and $E[\boldsymbol{y}(t)\boldsymbol{y}(t-1)']$ are exactly the quantity of $E[y(t,s)]$ and $E[y(t,s)\boldsymbol{y}(t-1)']$ since each row in the matrix represents the corresponding spatial position.

Chapter 4

Properties of Sparse Kalman Model

This Chapter will give some properties of our modified sparse Kalman Model. The modification still retains some properties of Kalman Model while some others can not be equal. We will discuss numerical stability, initialization sensitivity and likelihood non-monotonicity in this chapter. We will test properties on simulated data without input control features and compare them with the standard Kalman filter.

4.1 Initialization Sensitivity

Initialization is a necessary computational step in both Kalman filter and the Expectation–Maximization algorithm. The modified sparse Kalman Filter model is not an exemption. The initial estimates and covariance error are sensitive and affect how quickly the estimation error will converge in Kalman filter and determine how close it is to the global optimization in Expectation–Maximization algorithm. If we have an accurate initial estimate and error covariance, the filter will quickly converge. Otherwise, it will take time. Reasonable initial values will also help to find the best local maximum in the Expectation–Maximization algorithm.

Since either some computational or experimental results are available, these quantities can be set as initial value for π_1.[21] The average or median value of all measurements is one of the choices to be taken as the initial value for it. In addition, running the Expectation–Maximization algorithm several times with different chosen starting points is thus recommended to avoid using a poor randomly generated approximation to the true maximum.

In principle, the initial error covariance matrix V_1 is arbitrary. However, if we have high confidence on choice for the initial estimates and set it equal to zero, then the filter ignores and learns nothing from the measurements. This is almost never the case. And if we set a very large value and give pessimistic on the initial estimates, then the filter believes the measurements much more which will ignore the state value and lead to large fluctuations in the state and huge uncertainty in the parameter estimates. We prefer to

utilize an identity matrix rather than the zero matrix if we do not have a full grasp of where to start. If we take the average value of all measurements as the initial value, a diagonal matrix with the covariance of all measurements is the natural choice of initial value to avoid the extreme situations mentioned above.

The process noise matrix Q corresponds to the uncertainty that we expect in our state equations. This could include modelling errors or other uncertainties in the equations themselves. The value in matrix Q should be small enough to retain the learning potential from the measurement and to avoid increasing the uncertainty. [21]

If we could get enough prior information on how measurement is carried out, the information on calibration and reliability of the measurement could be regarded as a good initial estimate for R. Basically, we choose to give optimistic confidence on it otherwise the filter estimates become useless and we need to investigate the reliability of the data source. We assume an initial value for R in the filter which could be close to zero and later adaptively update and estimate it. [21]

We recommend using a zero-one sparse matrix with specific structure for initialized parameter matrix A and B since we do not have any prior information on these except for the spatial correlation.

Diagonal Entries	0.01	0.1	0.2	0.5	1
LogLikelihood	-2×10^6	-8×10^3	-1×10^2	-2×10^3	-8×10^2

Table 4.1: Maximum loglikelihood comparison between different diagonal entries of the matrix R during the 50 iterations

Suppose we verify the loglikelihood from different initial noise matrix R by fixing other initialized quantities. We choose the average value of all measurements as initial state and a diagonal matrix with the covariance of all measurements as the initial error covariance matrix. The process noise matrix Q is generated by an identity matrix. Table 4.1 illustrates how different the maximum log likelihood would be during the 50 iterations under different diagonal entries of the noise matrix R. The initials in modified sparse Kalman model would not change the sensitivity to likelihood. Decent initials would easily generate a better likelihood and the difference among power in variates of initials is huge so that those worse initials are likely to get stuck in local maximum which is far away from the global maximum.

4.2 Numerical Instability

When carrying out the Expectation–Maximization algorithm in Kalman filter, the numerical instability may lead to some issue and generate a non-positive definite matrix for covariance matrix. We already mentioned one situation during the derivation of Kalman filter equations. Unfortunately, this is not the only case. Besides, the error matrices can usually be non-positive definite matrices from rounding and basic arithmetic in matrices.

The measures we take here is to choose the lower triangular matrix of the results and duplicate it to its upper triangular position so as to keep the symmetricity of matrix first. After that, we implement a James-Stein-type shrinkage estimator for the covariance matrix, with separate shrinkage for variances and correlations. [23] It always returns a positive definite and well-conditioned covariance matrix. [22] Each covariance matrix shall go through this manipulation in each update step of Expectation–Maximization algorithm to make sure it is an invertible matrix before we continue to calculate their inverse matrix.

A Moore–Penrose pseudo-inverse to compute a best fit solution is applied for those parameters lacking a unique solution.[25] As explained in Chapter 3, it does not matter which solution we pick. The derivation of the Moore-Penrose generalized inverse of a matrix ensures that those estimated parameter matrices always have their solutions. It is widely used during learning the parameter matrix B and matrix A since both of them have sparse structure with zero values on the main diagonal of the matrix.

On the contrary, the updating equation for matrix A is $(\sum_{t=2}^{T} E[\boldsymbol{y}(t)\boldsymbol{y}(t-1)^{'}])$ $(\sum_{t=2}^{T} E[\boldsymbol{y}(t-1)\boldsymbol{y}(t-1)^{'}])^{-1}$ conditional on the parameters in the standard Kalman filter without input control features. [4] The equation in (3.27) suggests that the matrix $E[\boldsymbol{y}(t-1)\boldsymbol{y}(t-1)^{'}]$ is invertible since the sum of positive definite matrices and a positive semi-positive matrices is a positive definite and it is invertible.

4.3 Maximum Likelihood non-Monotonicity

The standard Expectation–Maximization algorithm increases its lower bound continuously so that the maximum likelihood estimate can be guaranteed to never get worse, but the monotonicity alone cannot necessarily guarantee the convergence of the sequence to the global maximum. If the likelihood function has multiple peaks, expectation–maximization algorithm will not necessarily find the global maximum of the likelihood.

In practice, it is common that the log-likelihood is not concave and one simple approach is to start Expectation–Maximization algorithm from multiple random initial guesses, and choose the one with the largest likelihood as the final guess. [28]

To accommodate the spatial structure in modified sparse Kalman model, we break in greedy algorithm for estimating the locally optimal choice at each row of matrix in order to approximate the local optimal matrix, instead of searching the local optimal matrix directly. It leads to the sparsity of matrix due to the spatial correlation. Nevertheless, this approximated estimates can not guarantee monotonicity since the estimated matrix by rows is not the genuine maximization for the expectation of the joint density.

In the same simulated data, it is obvious that the log likelihood generated by the standard Expectation Maximization algorithm converges rapidly and increases monotonically in the picture above in Figure 4.1. In contrast, the plot below clearly shows that the iterations of greedy algorithm have one deep concave and one shallow concave which means that it violates the property of monotonicity in standard Expectation–Maximization

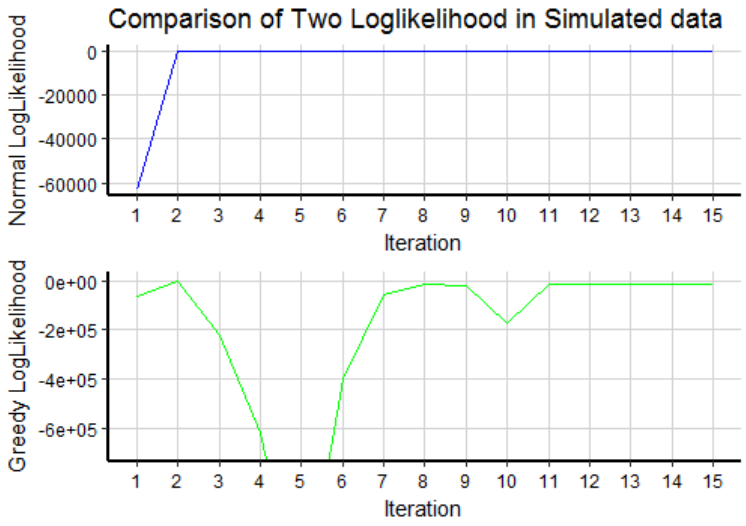


Figure 4.1: Comparison between greedy algorithm and normal algorithm

algorithm. The zigzag green line also suggests that the best log likelihood in limited iterations through greedy algorithm could be anywhere and do not guarantee to stay one of the peak points all along.

Chapter 5

Application to Schiphol Bird Case

This chapter will cover the implementation of a practical case at Schiphol Airport, which involves four steps. At first, we reduce the number of variables by principal component analysis. Afterwards, we train a spatial linear dynamical system on the basis of Kalman filter and smoother equations and modified EM algorithm. Then, we investigate the performance of the method we proposed. In the end, we train a simple penalized regression as our base performance to compare.

5.1 Dataset

The Schiphol Bird dataset contains two parts. RobinRadar has a moving objects detection radar system which provides birds behavior related data. [13] Meanwhile, Air Traffic Control the Netherlands provides the weather data in Schiphol area. [14] Data rights are reserved by Data innovation Lab Schiphol. If readers are interested in this data set, they could take part in the Kaggle competition which is still in preparation or ask Data innovation Lab Schiphol for the rights.

The data set from Robinradar contains the following variables: timestamp, position with latitude and longitude, velocity, airspeed, heading, heading vertical, peak mass, mass , mass correction, distance travelled and score. The weather data contains visibility, cloud layer for coverage, average forecasted wind direction, average forecasted wind speed, average forecasted wind speed including gust, forecasted showers of rain, snow or thunderstorms. According to the request from the bird control team in Schiphol Airport, we are interested in predicting the mass in some particular areas in order to instruct bird controllers to take measures efficiently. Therefore, the observations $\boldsymbol{z}(t)$ in the spatial dynamical system is mass from Robinradar data set while the input control features $\boldsymbol{u}(t)$ are position with latitude and longitude, velocity, airspeed, heading, heading vertical, peak mass , mass correction, distance travelled, score, visibility, cloud layer for coverage, average forecasted wind direction, average forecasted wind speed, average forecasted wind speed including gust and forecasted showers of rain, snow or thunderstorms.

Here we choose two days' data from one of the birds active time periods in one day and only focus on runway Polderbaan and its vicinity which is our core region of interest

$\mathcal{S}$ to ensure that the given space area $\mathcal{L}$ is large enough for most of bird to fly into runway Polderbaan in five minutes. That is to say, only those birds in area $\mathcal{L}$ may fly into runway Polderbaan in five minutes. The given area $\mathcal{L}$ is divided into 6×4 grid and the core region takes over the middle 4×2 grid. These can be exactly found in Figure 2.1. We aggregate the data by one minute as well and take its mean value.

The final data set starts from 10:00 to 12:00 on 7th April 2016 and 8th April 2016 which means we have 120 minutes and each minute contains 24 cells data information including 8 core region cells. Since it is time-correlated data from two separate days, the best option for validation is training a first day's data and testing on the next day. Here we suppose these two days have a similar environmental condition.

5.2 Features Reduction

When we cope with this particular practical challenge, we have already filtered relevant features by hand from data vendors through understanding the research questions. The remaining 18 features mentioned above may still be correlated, so the first step of data pretreatment is feature reduction. For instance, from the physics point of view, the distance travelled, velocity and airspeed are correlated. We prefer to search for the representative principal components to achieve the purpose of feature selection.

The sample data set we choose is so small that some of the features only contain one value in the train set. They are considered to have no predictive power in the model than those features with all levels. After dropping those features in both train and test set, only spatial features remain in the data set, which means the weather condition could be treated as stable during that period on that day since there are not any more weather features in both train and test set. The assumption of similar environmental condition we made before is met. A principal component analysis is implemented here to find fewer features to represent data.

Principal component analysis is used to extract the important information from a multivariate data table. The information in a given data set corresponds to the total variation it contains. The number of dimensions we choose should capture more than 90% of the total inertia where the inertia is the total variance of data set. We have to scale the remaining features in order to give the same influence to each one since the features are not measured in the same units. FactoMineR and Factoextra packages in R are used for computing principal component analysis and for visualization respectively. [26]

The graph in Figure 5.1 shows the loading plot of PCA to give the correlations among all features and variance explained by the first two components. Features are colored according to the values of the squared loadings which are used to give the amount of explained variation and estimate the quality of the representation. Because mass is the feature we need to predict, we do not want it to participate in the principal component analysis and set it as a supplementary feature. The supplementary feature will not be

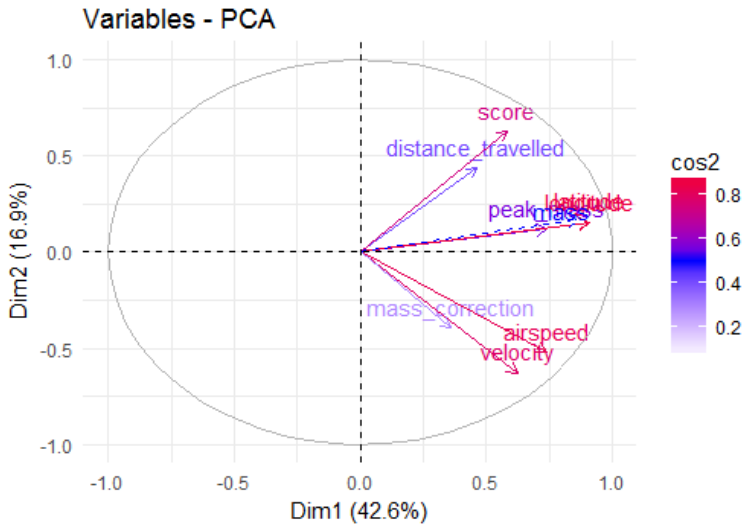


Figure 5.1: Variable Factor Map

active in the analysis but to bring supplementary information and help to interpret the dimensions of variability.[26] The closer a feature is to the circle of correlations which is the grey circle in Figure 5.1, the more important it is to interpret these two components because the squared length of the vector is the portion of that variance explained by the two components. On the other hand, features that are closed to the center of the plot are less important for the components. Thus, we could detect those important features in first two dimensions such as: latitude, longitude, airspeed,velocity,etc. But the first two dimensions only capture no more than 60% of the total inertia and this is not enough to represent most information from data set.

Therefore, we inspect more components to achieve more than 90% of the total variance of data set. Based on these principal components, we could explore which features contribute more to the principal components. For a given component, a feature with a contribution larger than the expected average contribution which is the red dashed line in Figure 5.2 could be considered as important in contributing to the component. Thus, longitude, latitude, heading, mass correction and velocity are crucial features. This is consistent with common sense as well. The bird movements are strongly correlated to their location, direction and speed.

5.3 Algorithm Application

We implement Expectation Maximization algorithm to estimate the structured parameters in sparse linear spatial dynamical system. The details of system structure are in the Section 2.1. However, the Expectation–Maximization algorithm could be trapped

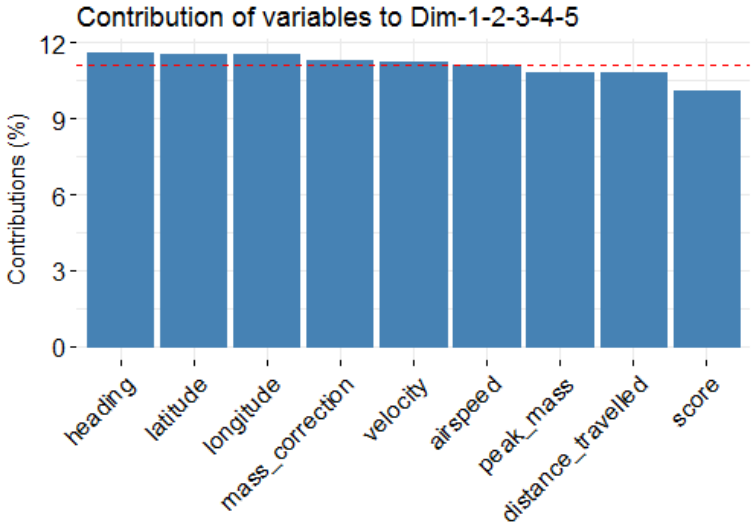


Figure 5.2: Contribution of Features on 5 Dimensions

at local maxima and consequently fails to reach global maxima. [27] Since we also implement the greedy algorithm during the search process of matrix A and B, it does not appear to guarantee to reach the local maximum and it can just approximate the local maximum as we discussed in the last Chapter. In addition, the best local maximum is highly depending on the start values. Therefore, we are focusing on searching the best initial error matrix Q and R in terms of the prediction ability in this section. We start from the same diagonal matrix Q and R and then we try to reduce the measurement error matrix R since we are supposed to be confident on the measurement equipment which is the RobinRadar detect system.. In the end, we select two metrics, best log-likelihood and best root mean square error, to compare the better one on the test data set.

We initialize state value and covariance matrix by the global mean and global variance from the training data set. We begin to set the measurement and process error matrices in the diagonal matrices with same entries value on 0.01, 0.05, 0.1 ,0.2 and 0.5 under 50 iterations and 0.001 tolerance.

Apart from the likelihood in EM algorithm, we also use the root mean square error and the mean absolute error as alternative metrics to assist the performance review and model selection based on those initial values. We will report the performance in test set from the best likelihood and best root mean square error we can find from the train set by different initial values.

It is inevitable to normalize all features before performing the algorithm. The correct way to apply standardization is only using the mean and standard deviation of training data set on both training and validation data set. Essentially, if we apply the normalization into the whole data set or training and test data set respectively, it would leak information either from the future or from test set into the training or evaluation of the

model which could cause considerable optimism bias in the model evaluation. The main point is that we can not employ the features in the validation and test set for anything except comparing to the performance of predicted values.

Initial Value	0.01	0.05	0.1	0.2	0.5
LogLikelihood	-0.0038	**-0.0017**	-0.0021	-0.0026	-0.0029
RMSE	**0.9649**	0.9653	0.9653	0.9651	0.9652
MAE	0.5820	0.5825	0.5821	0.5808	**0.5786**

Table 5.1: Performance metrics in training set by using the largest maximum likelihood estimates among all 50 iterations.

From Table 5.1, we can easily find the best loglikelihood do not generate the best root mean square error and mean absolute error. Generally speaking, minimizing the root mean squared error is equivalent to maximizing the log likelihood of the data. However, this is inappropriate to our case since we apply a greedy EM algorithm. Thus, we will find the models from best loglikelihood and best root mean square error respectively and set mean absolute error as side metric. Owing to expectation–maximization algorithm finding the local maxima, the conclusion we can draw is that initial entries 0.01 may be the closest one to the global maxima among all these settings if we set root mean square error as our main metrics. In the meantime, we will choose initial entries 0.05 if we put loglikelihood as main metrics. On the other side, the measurement equipment is not completely trustworthy if we use initial entries 0.05 or 0.01 on the measurement error matrix since we expect the error would be very close to zero, otherwise it is meaningless to collect and apply the data from unreliable radar system. Then, we attempt to find another local maxima so as to discover a better performance with reliable initial certainty of measurement. We test diagonal entries of 0.0001, 0.0002, 0.0005, 0.001 and 0.002 as the new entries of the measurement error matrix R under the best initial process error matrix we can find above which is the diagonal matrix Q with entries 0.01 or 0.05.

Initial Value	0.0001	0.0002	0.0005	0.001	0.002
LogLikelihood	-0.0066	-0.0098	-0.0089	**-0.0048**	-0.0136
RMSE	0.9842	0.9881	**0.9805**	0.9873	0.9870
MAE	0.6143	0.6201	**0.6128**	0.6195	0.6174

Table 5.2: Performance metrics in training set by using the largest maximum likelihood estimates among all 50 iterations under the initial entries of 0.01 in matrix Q.

If we choose the best root mean square error as our initial settings on matrix Q, we can find that the best root mean square error among the settings in matrix R is the initial entries value 0.0005 which it quite makes sense we shall trust more on our measurement

equipment. However, the prediction ability does not improve after we choose to trust more on the measurement. The best prediction performance is still the initial diagonal entries of 0.01 on both measurement error and process error matrices. In addition, it does not give the best loglikelihood among these settings and it generates much worse loglikelihood than the best one we find in last settings group. It is logical that those settings seem to go far away from one of the local maximums. The details can be found in Table 5.2.

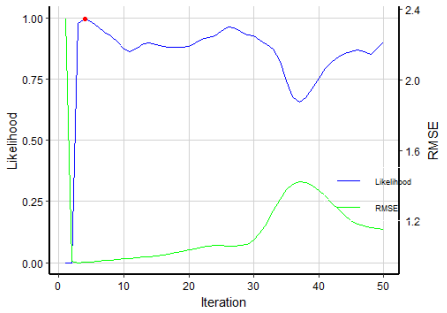
Initial Value	0.0001	0.0002	0.0005	0.001	0.002
LogLikelihood	**-0.00018**	-0.00038	-0.00156	-0.0066	-0.0081
RMSE	**0.9783**	0.9793	0.9827	0.9889	1.0625
MAE	**0.6051**	0.6069	0.6120	0.6202	0.6996

Table 5.3: Performance metrics in training set by using the largest maximum likelihood estimates among all 50 iterations under the initial entries of 0.05 in matrix Q.

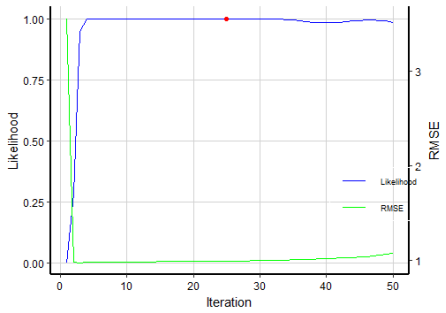
We test five smaller possibilities of measurement error matrices (0.0001,0.0002,0.0005, 0.001 and 0.002) and the results are shown in Table 5.3 by only using the best maximum likelihood estimates among all 50 iterations. The best loglikelihood can simultaneously generate the best root mean square error in those settings. In addition, the best loglikelihood in these settings is also the best one among all setting we have tried and it confirms that the measurement equipment is relatively trustworthy from this result. Nevertheless, the root mean square error generated by the best loglikelihood is not the model with best prediction ability among all settings we have selected. Comparing the results from both model selection criteria, the measurement equipment from best root mean square error is not as trustworthy as the one from best likelihood. The initial measurement error from likelihood selection criteria is in accordance with what we expect. It is hard to choose a better selection criteria from those results, so we will report both prediction performance on test data set.

The essence of those process is to search for the closest local maxima to the global maxima as much as possible from different initial values. The initialized model we prefer from two criteria both gives quick convergence in likelihood and root mean square error. Figure 5.3 shows that the largest likelihood estimate is generated at early step from best RMSE metric initials ($Q = \text{diag}(0.01, \ldots, 0.01)$ and $R = \text{diag}(0.01, \ldots, 0.01)$) while it also gives a fairly good root mean square error. After that, the monotonic increase is no longer valid and start to fluctuate during the 50 iteration steps. The best likelihood metric initials ($Q = \text{diag}(0.05, \ldots, 0.05)$ and $R = \text{diag}(0.0001, \ldots, 0.0001)$) get to the largest likelihood at the intermediate step and also give a little fluctuation after reaching the largest likelihood. In the meanwhile, the largest likelihood fails to generate the smallest root mean square error.

Next, we would like to examine which proposed metrics is more reliable in the next day test set. We investigate the performance in the whole area in 24 cells and the core



(a) initial settings in terms of $Q = \text{diag}(0.01, \ldots, 0.01)$ and $R = \text{diag}(0.01, \ldots, 0.01)$



(b) initial settings in terms of $Q = \text{diag}(0.05, \ldots, 0.05)$ and $R = \text{diag}(0.0001, \ldots, 0.0001)$

Figure 5.3: Likelihood and RMSE performance during the iteration in two initial settings.

		First Hour	First two Hours
Whole Area	RMSE	0.780	0.783
	MAE	0.470	0.469
Core Region	RMSE	0.563	0.557
	MAE	0.455	0.439

Table 5.4: Performance when initialized from best RMSE metric

region in 8 cells during the first hour and first two hours respectively. In Table 5.4 and Table 5.5, whether it is one hour data or two times of them, the performance in core region is much better than the whole area. What deserves to be mentioned is the core region is the exact place we focus on which contains the most completely neighbouring information. The edge cells in the whole area $\mathcal{L}$ do not capture any information outside $\mathcal{L}$. In addition, no matter which metrics we choose, they are quite stable on different size of data. When we double the size of test set, the performance didn't change too much in root mean square error and mean absolute error. Therefore, the parameters in the model can be constant under the same weather conditions. With the help of all the quantities we give from both metrics, the best root mean square error metric stands out and shows more powerful prediction ability than the best likelihood metric.

		First Hour	First two Hours
Whole Area	RMSE	0.811	0.817
	MAE	0.512	0.516
Core Region	RMSE	0.626	0.632
	MAE	0.490	0.488

Table 5.5: Performance when initialized from best likelihood metric

The remarkable fact of prediction ability in test set is the root mean square error and mean absolute error are better than those in training set. From the perspective of statistics, the error is from two sources: bias and variance. The performance shows that variance from the test data set is lower than those in the training and validation set which makes this phenomenon happen. This could be the case because we use two separate days. It appears 165 times that there are not any bird movements in any of the 24 cells during the first day data while 92 times appear in the second day test set. Less extreme values could help to improve the prediction ability. We may optimistically conclude that the Kalman model surely captures some of the information and generate a good prediction under the same environment conditions.

5.4 Comparison with Lasso Regression

In this section, we will use lasso regression to fit this data set so that we could genuinely evaluate the sparse spatial Kalman model by comparison.

5.4.1 Performance in Lasso Regression

We still follow the spatial assumption we made in the beginning so that the current state in one cell is predicted by all the features in the neighbourhood set cells one time step before. In our particular case, the current mass in one cell is determined by previous mass and five other features (heading, latitude, longitude, mass correction and velocity) in the neighbourhood set cells. Those features are the ones after PCA feature selection. Therefore, there are 54 features in total for prediction model(6 features and 9 cells). In consideration of the large number of features and bias-variance tradeoff, we prefer a Lasso regression since Lasso does both parameter shrinkage and variable selection automatically. To choose the λ in the Lasso that minimizes L1 loss function, we perform a progressive validation instead since the cross validation is not suitable for the dependent time series data. We separate five folds from two-hour training set. The first forty-five minutes is the headmost subtraining set and validate the next fifteen minutes, then we unite the first subtraining set and validation set as the second subtraining set, namely the first 60 minutes, to validate the next fifteen minutes. We repeat it until we make full use of two-hour data. Table 5.6 shows all folds with subtraining set and validation set where the validation set in each fold is underlying with italic type. The average root mean square error and mean absolute error in five folds are then selected to assess the performance.

After we search for the best λ, we report the same performance evaluation metrics in test data set as we did in the last section in Table 5.7. Those results we obtained from Lasso are much better than Spatial Kalman filter model. Comparing different size of the test data set, Lasso is stable on them as well. In the core region, the prediction ability in Lasso is even twice as good as it in spatial Kalman model. However, the performance is getting a little worse in Lasso while it obtains much better prediction ability in Kalman

Fold 1	10:00-10:44	*10:45-10:59*		
Fold 2		10:00-10:59	*11:00-11:14*	
Fold 3			10:00-11:14	*11:15-11:29*
Fold 4				10:00-11:29	*11:30-11:44*
Fold 5					10:00-11:44	*11:45-11:59*

Table 5.6: Progressive Validation

model if we double the size of data in the core region. The self-righting process in Kalman model can help to refine the estimate so as to improve the prediction stability and Lasso is lack of this step. From the application point of view, spatial Kalman filter does not have any strengths in this data set. In contrast, a simple penalized regression is more suitable to cope with this practical project.

		First Hour	First two Hours
Whole Area	RMSE	0.588	0.606
	MAE	0.338	0.360
Core Region	RMSE	0.295	0.331
	MAE	0.236	0.263

Table 5.7: Performance in different area and different time period from Lasso

5.4.2 Comparison of Lasso and Sparse Kalman Filter

Expectation–Maximization algorithm for learning the parameters in sparse Kalman filter model can only convergence to a local optimization from almost any starting point. We can only approximate the possible best metric under limited computer computation resource and never get the theoretical best one. To reach the spatial structure, we apply to embed a greedy algorithm in the Expectation–Maximization procedure so that we can only approximate the maximum likelihood estimation. It is precisely because of this situation, the model we obtain in the end is the one that is close to one of the local optimization.

In the other hand, the number of parameters is dramatically increasing when we treat each parameter matrix as a spatial correlation map. When the number of observations is not large relative to the number of parameters, the technique is actually an ill-posed problem. The data we need should be large enough to be well studied. We may have problems with calibrating this complicated model's parameters if we do not have enough data. This will limit the accuracy and flexibility of the model. One practical solution is to reduce the number of parameters by adding more assumptions in the spatial system or add the penalty function to reduce the number of parameters.

Because of the initialization sensitivity in both Kalman filter and Expectation Maximization algorithm, inaccurate initialization and prior information of the filter is easy

to fail Kalman filter model applications. The matrix computation during the whole procedure may cause the numerical instability, such as approaching singularities of the log-likelihood function, the log-likelihood function not being bounded, etc. We apply some simple methods to avoid these problems in return for giving up the accuracy of the results.

In our particular Schiphol bird case, we make same environment condition supposition and drop those features which are expected to have a strong influence in terms of common sense due to the restriction of available data and computer computation power. This may lead to unstable prediction on further test data set. One of the issues resulting in prediction instability is the extreme value. Kalman filter assumes that both the system and observation models equations are linear and the state belief is Gaussian distributed. It would give low confidence on both tail part, even none on the extreme value. The variation in the number of extreme value will to a great extent determine how well the model performs. If we reverse the training set and test set given by last case, the performance would not be received as good as in the current order. In addition, it is inevitable that the model contains too many parameters to be estimated and those parameters can still be colinear. The results from Lasso indirectly confirm this point since Lasso has penalized 38 out of 79 parameters to 0 in the training data set. We therefore speculate that the prediction ability would improve if we would add a penalty function to regularize the coefficients in spatial Kalman model.

Chapter 6

Conclusion & Future Work

6.1 Conclusion

We present a new modified Kalman filter model combining the time dependence with spatial dependence. Unlike the nature of Kalman filter model deriving from physics model, we utilize it without any prior model knowledge except for the spatial relation. We design a greedy EM algorithm that can estimate sparse parameter matrices in Kalman filter model. The algorithm can solve the problem with capturing the time and spatial dependence simultaneously but reaches the approximate local optimization. The incidence matrices are applied to transition matrix and input control matrix in the sparse Kalman filter model that shows the relationship among all the locations. Since the Kalman model is based on the whole area, it will bring more parameters to estimate in the transition parameter matrix which will lead to overfitting and increasing run-time complexity.

6.2 Future work

As for future work, one of the practical challenges is to reduce the number of parameters in the transition matrix. Before deploying some algorithm, we can assume some of parameters are the same in terms of the prior information. This operation requires rederiving the updated equations in Expectation–Maximization algorithm under this constraint. In M step, we would still implement the greedy algorithm, but to take the derivation of each parameter in the matrix and obtain the maximum likelihood estimates instead of taking the derivation of each vector in the matrix. After that, we construct the transition parameter matrix by those maximum likelihood estimates. We can also include a penalty function during the Kalman recursion to achieve the goal of parameters reduction.

Bibliography

[1] Paul Zarchan, Howard Musoff *Fundamentals of Kalman filtering* 1st ed, American Institute of Aeronautics and Astronautics. 2000

[2] Greg Welch, Gary Bishop *An Introduction to the Kalman Filter* Lecture Note, University of North Carolina at Chapel Hill.Available from: http://www.cs.unc.edu/

[3] M.Yu Byron, V.Shenoy Krishna, Maneesh Sahani *Derivation of Kalman Filtering and Smoothing Equations* Stanford University, Tech. Rep., 2004.

[4] Christopher Bishop *Pattern Recognition and Machine Learning* 1st ed. New York: Springer 2006.

[5] Sam Roweis, Zoubin Ghahramani *A Unifying Review of Linear Gaussian Models* Neural Computation, 11(2), pp.305-345. 1999.

[6] Max Welling *The Kalman Filter* Lecture Note, California Institute of Technology Available from: http://www.stat.columbia.edu/

[7] Waldo Tobler *A computer movie simulating urban growth in the Detroit region* Economic Geography, 46, p.234, 1970.

[8] Xiaobai Yao *Research Issues in Spatio-temporal Data Mining* In White paper submitted to the University Consortium for Geographic Information Science (UCGIS) workshop on Geospatial Visualization and Knowledge Discovery, Lansdowne, Virginia, pages18–20,2013.

[9] Jean-Francois Mari,Florence Le Ber,El-Ghali Lazrak,et al *Using Markov Models to Mine Temporal and Spatial Data* Kimito Funatsu and Kiyoshi Hasegawa. New Fundamental Technologies in Data Mining, Intech, pp.561–584, 2011.

[10] Max A. Woodbury *Inverting modified matrices* Memorandum Rept. 42, Statistical Research Group, Princeton University, Princeton, NJ, 1950.

[11] K.Venkateswara Rao,A.Govardhan, K.V.Chalapati Rao *Spatiotemporal Data Mining: Issues, Tasks And Applications* International Journal of Computer Science & Engineering Survey, 3(1), pp.39-52.2012

[12] John R. Allan *The costs of birdstrikes and birdstrike prevention* In L.Clarke (Ed.), Human Conflicts with Wildlife: Economic Considerations, pp. 147–153.2002 .

[13] Robin Radar *3D Flex - Robin Radar Systems* [online] Robin Radar Systems. Available at: http://www.robinradar.com/3d-flex/ [Accessed 28 Nov. 2016].

[14] Luchtverkeersleider *Air Traffic Control the Netherlands* [online] Air Traffic Control the Netherlands. Available at: https://www.lvnl.nl/en [Accessed 28 Nov. 2016].

[15] Byron Boots *Learning Stable Linear Dynamical Systems* Master Thesis in Carnegie Mellon University,2009.

[16] Elizabeth Eli Holmes *Derivation of an EM Algorithm for Constrained and Unconstrained Multivariate Autoregressive State-Space Models* Northwest Fisheries Science Center, NOAA Fisheries,2725, 2012

[17] Schiphol Airport *Amsterdam Airport Schiphol* [online] Royal Schiphol Group. Available at: https://www.schiphol.nl/en/schiphol-group/ [Accessed 28 Nov. 2016].

[18] Sujit K. Sahu, Kanti V. Mardia *Recent trends in modeling spatio-temporal data* Proceedings of the Special Meeting on Statistics and Environment, pp. 69–83. 2005

[19] L.Sanchez, S.Infante, V.Griffin and D.Rey *Spatio-temporal dynamic model and parallelized ensemble Kalman filter for precipitation data* Brazilian Journal of Probability and Statistics, 30(4), pp.653-675. 2016

[20] Y.Bar-Shalom, X. Li and T.Kirubarajan, *Estimation with applications to tracking and navigation.* 1st ed. New York: Wiley, pp.208-209,334-338. 2001

[21] M M Shyam, Naren Naik,R M O Gemson and M R Ananthasayanam*Introduction to the Kalman filter and tuning its statistics for near optimal estimates and Cramer Rao bound* TR/EE2015/401, IIT Kanpur. 2015

[22] R.Opgen-Rhein and K.Strimmer,*From correlation to causation networks: a simple approximate learning algorithm and its application to high-dimensional plant gene expression data.* BMC Systems Biology, 1(1), p.37. 2007

[23] J.Schäfer and K.Strimmer.*A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics.* Statistical Applications in Genetics and Molecular Biology, 4(1). 2005

[24] N.Higham, *Computing the nearest correlation matrix - a problem from finance*; IMA Journal of Numerical Analysis 22, 329–343. 2002

[25] W.N.Venables and B.D.Ripley *Modern Applied Statistics with S-PLUS.* Third Edition. Springer. p.100. 1999

[26] S.Lê, J.Josse and F.Husson *FactoMineR : An R Package for Multivariate Analysis.* Journal of Statistical Software, 25(1).2008

[27] Y.Wang and N.L.Zhang *Severity of local maxima for the em algorithm: Experiences with hierarchical latent class models.* In Proceedings of the 3rd European Workshop on Probabilistic Graphical Models, pp. 301–308.2006

[28] M.R.Gupta and Y.Chen *Theory and use of the EM algorithm.* Foundations and Trends in Signal Processing, 4, 223–296.2011

[29] Z.Ghahramani *An Introduction to Hidden Markov Models and Bayesian Networks* International Journal of Pattern Recognition and Artificial Intelligence, vol. 15, no. 1, pp. 9–42, 2001.

[30] R.Penrose *A generalized inverse for matrices* Mathematical Proceedings of the Cambridge Philosophical Society, 51(3), 406-413, 1955

Appendix A

Notation

Table A.1: Notation and symbols

Symbol	Meaning
$z(t, s)$	Spatial temporal observations at time t and location s
$y(t, s)$	Spatial temporal true state at time t and location s
$\mathcal{L}$	Given space area
$\mathcal{S}$	Core region at risk in the given area
$\mathcal{T}$	NO. of observed time-points
$s = (i, j)$	Grid cell after segmentation, simplified as s
$N(s)$	Set of neighborhood cells for a given $s = (i, j)$
$\boldsymbol{z}_{1:t}(s)$	Observations in cell s from time 1 to t
$\boldsymbol{s}$	All separated cells after grid segmentation in area $\mathcal{L}$
$\boldsymbol{z}(t)$	Spatial temporal observations at time t in all cells $\boldsymbol{s}$
$\boldsymbol{y}(t)$	True state vector at time t in all cells $\boldsymbol{s}$
$\boldsymbol{u}(t)$	All input control features matrix at time t in all separated cells and consist of all $\boldsymbol{u}_n(t, \boldsymbol{s})$
$\boldsymbol{u}_n(t, \boldsymbol{s})$	the n-th features vector at time t in all separated cells
w_t	Process noise matrix in all cells
v_t	Measurement noise matrix in all cells
A	state transition matrix
$\vec{a_{(i,j)}}$	transition vector of cell (i, j) in transition matrix A
B	control input matrix
$\vec{b_{(i,j)}}$	control input vector of cell (i, j) in control input matrix B
A_s	Scalar of transition in each cells
$\mathcal{B}_n$	the n-th submatrix in the matrix B
Q	covariance matrix for process noise
R	covariance matrix for measurement noise
Q_s	Scalar covariance for process noise
R_s	Scalar covariance for measurement noise
$\{\boldsymbol{y}\}_{t_0:t_1}$	True state in all separated cells from time t_0 to t_1

Table A.1: Notation and symbols

Symbol	Meaning
$\{\boldsymbol{z}\}_{t_0:t_1}$	Measurements in all separated cells from time t_0 to t_1
$\{\boldsymbol{u}\}_{t_0:t_1}$	All input controal features in all separated cells from time t_0 to t_1
$P_t^{t^*}$	Expected state vector at time t given observations and input variables from time one to time t^*
$V_t^{t^*}$	State covariance matrix at time t given observations and input variables from time one to time t^*.
K_t	Kalman Gain at time t
J_t	Smoother Gain at time t
$V_{t+1,t}^T$	State covariance matrix between time t and time$t+1$ given observations and input variables from time one to time T.
$\boldsymbol{J}_{(i,j)}$	Indicator matrix for cell (i,j) in matrix A
$\boldsymbol{K}_{(i,j)}$	Indicator matrix for cell (i,j) in matrix B
π_1	Initial expected state in Kalman filter
V_1	Inditial variance in Kalman filter

Appendix B

Proof of Measurement Updated Equations in Kalman Filter

In the following paragraph, we will show how to derive three equations in Measurement Updated Equations (2.12).

Proof. In general, the logarithm of the probability density of the normally-distributed variable with mean μ and variance Σ is (to be simplified, we drop the constant part): [3]

$$\begin{aligned}\log P(x) &= -\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu) \\ &= -\frac{1}{2}x'\Sigma^{-1}x + x'\Sigma^{-1}\mu - \frac{1}{2}\mu'\Sigma^{-1}\mu \end{aligned} \tag{B.1}$$

To find the logarithm of the conditional probability in the linear dynamical system assumption, we have:

$$\begin{aligned}\log P(\boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:t},\{\boldsymbol{u}\}_{1:t}) &= \log P(\boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:t-1},z(t),u_{1:t-1},u(t)) \\ &= \log P(\boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:t-1},z(t),\{\boldsymbol{y}\}_{1:t-1}) + \log P(u(t)|\boldsymbol{y}(t),\{\boldsymbol{z}\}_{1:t-1},z(t),\{\boldsymbol{u}\}_{1:t-1}) \\ &\quad - \log P(u(t)|\{\boldsymbol{z}\}_{1:t-1},z(t),\{\boldsymbol{u}\}_{1:t-1}) \\ &= \log P(z(t)|\boldsymbol{y}(t),\{\boldsymbol{z}\}_{1:t-1},\{\boldsymbol{u}\}_{1:t-1}) + \log P(\boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:t-1},\{\boldsymbol{u}\}_{1:t-1}) + \dots \\ &= \log P(z(t)|\boldsymbol{y}(t)) + \log P(\boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:t-1},\{\boldsymbol{u}\}_{1:t-1}) + \dots \\ &= -\frac{1}{2}(z(t)-\boldsymbol{y}(t))'R^{-1}(z(t)-\boldsymbol{y}(t)) - \frac{1}{2}(\boldsymbol{y}(t)-P_t^{t-1})'(V_t^{t-1})^{-1}(\boldsymbol{y}(t)-P_t^{t-1}) \\ &\quad + \dots \\ &= -\frac{1}{2}\boldsymbol{y}(t)'(R^{-1}+(V_t^{t-1})^{-1})\boldsymbol{y}(t) + \boldsymbol{y}(t)'(R^{-1}z(t)+(V_t^{t-1})^{-1}P_t^{t-1}) + \dots \end{aligned} \tag{B.2}$$

In addition, we give the Woodbury matrix identity in the following application. [10] Suppose matrices A,B,C,and D are with appropriate dimensions, we have:

$$(A+BC^{-1}D)^{-1} = A^{-1} - A^{-1}B(C+DA^{-1}B)^{-1}DA^{-1} \tag{B.3}$$

Comparing the same structure in the first term of the equation (B.1) and (B.2) and apply the Woodbury matrix identity, we have

$$\begin{aligned} V_t^t &= (R^{-1} + (V_t^{t-1})^{-1})^{-1} \\ &= V_t^{t-1} - V_t^{t-1}(R + V_t^{t-1})^{-1}V_t^{t-1} \\ &= V_t^{t-1} - K_t V_t^{t-1} \end{aligned} \tag{B.4}$$

where K_t is called Kalman gain and

$$K_t = V_t^{t-1}(R + V_t^{t-1})^{-1} \tag{B.5}$$

Next, we need to combine the time update and measurement update estimates for the mean of the conditional normal distribution. Comparing the same structure in the second term of the equation (B.1) and (B.2) and use the equation (B.4) and (B.5), we have:

$$\begin{aligned} P_t^t &= V_t^t(R^{-1}z(t) + (V_t^{t-1})^{-1}P_t^{t-1}) \\ &= (V_t^{t-1} - V_t^{t-1}(R + V_t^{t-1})^{-1}V_t^{t-1})(R^{-1}z(t) + (V_t^{t-1})^{-1}P_t^{t-1}) \\ &= V_t^{t-1}(I - (R + V_t^{t-1})^{-1}V_t^{t-1})R^{-1}z(t) + (I - V_t^{t-1}(R + V_t^{t-1})^{-1})P_t^{t-1} \\ &= V_t^{t-1}((R + V_t^{t-1})^{-1}(R + V_t^{t-1}) - (R + V_t^{t-1})^{-1}V_t^{t-1})R^{-1}z(t) + (I - K_t)P_t^{t-1} \\ &= V_t^{t-1}(R + V_t^{t-1})^{-1}z_t + (I - K_t)P_t^{t-1} \\ &= K_t z(t) + (I - K_t)P_t^{t-1} \\ &= P_t^{t-1} + K_t(z(t) - P_t^{t-1}) \end{aligned} \tag{B.6}$$

The equation (B.5) can be written as:

$$K_t R K_t^T = V_t^{t-1}K_t^T - K_t V_t^{t-1}K_t^T \tag{B.7}$$

Thus, the equation (B.4) can be derived to a sum function:

$$\begin{aligned} V_t^t &= V_t^{t-1} - K_t V_t^{t-1} \\ &= (I - K_t)V_t^{t-1}(I - K_t)^T + (I - K_t)V_t^{t-1}K_t^T \\ &= (I - K_t)V_t^{t-1}(I - K_t)^T + K_t R K_t^T \end{aligned} \tag{B.8}$$

□

Appendix C

Proof of Kalman Smoother

C.1 Preliminary

In this part, we will give the expression of logarithmic probability density for the general case and the Kalman system.

In general, the logarithm of probability density of the two normally-distributed variable $[x_1, x_2]$ with mean $[\mu_1, \mu_2]$ refers to the form (to be simplified,we drop the constant part):

$$\begin{aligned}\log P(x_1, x_2) &= -\frac{1}{2}\begin{pmatrix} x_1 - \mu_1 \\ x_2 - \mu_2 \end{pmatrix}' \begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix} \begin{pmatrix} x_1 - \mu_1 \\ x_2 - \mu_2 \end{pmatrix} + \dots \\ &= -\frac{1}{2}x_1' S_{11} x_1 - \frac{1}{2}x_1' S_{12} x_2 - \frac{1}{2}x_2' S_{21} x_1 - \frac{1}{2}x_2' S_{22} x_2 + x_2'(S_{21}\mu_1 + S_{22}\mu_2) + \dots \end{aligned} \tag{C.1}$$

where those identities are followed if we introduce the notation in Kalman system into the covariance matrix:

$$\begin{aligned}\begin{pmatrix} V_{t+1}^T & V_{t+1,t}^T \\ V_{t,t+1}^T & V_t^T \end{pmatrix} &= \begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix}^{-1} \\ &= \begin{pmatrix} -(S_{21} - S_{22}S_{12}^{-1}S_{11})^{-1}S_{22}S_{12}^{-1} & (S_{21} - S_{22}S_{12}^{-1}S_{11})^{-1} \\ S_{12}^{-1} + S_{12}^{-1}S_{11}(S_{21} - S_{22}S_{12}^{-1}S_{11})^{-1}S_{22}S_{12}^{-1} & -S_{12}^{-1}S_{11}(S_{21} - S_{22}S_{12}^{-1}S_{11})^{-1} \end{pmatrix} \\ &= \begin{pmatrix} (S_{11} - S_{12}S_{22}^{-1}S_{21})^{-1} & -(S_{11} - S_{12}S_{22}^{-1}S_{21})^{-1}S_{12}S_{22}^{-1} \\ -S_{22}^{-1}S_{21}(S_{11} - S_{12}S_{22}^{-1}S_{21})^{-1} & (S_{22} - S_{21}S_{11}^{-1}S_{12})^{-1} \end{pmatrix} \\ &= \begin{pmatrix} (S_{11} - S_{12}S_{22}^{-1}S_{21})^{-1} & -(S_{11} - S_{12}S_{22}^{-1}S_{21})^{-1}S_{12}S_{22}^{-1} \\ -S_{22}^{-1}S_{21}(S_{11} - S_{12}S_{22}^{-1}S_{21})^{-1} & S_{22}^{-1} + S_{22}^{-1}S_{21}(S_{11} - S_{12}S_{22}^{-1}S_{21})^{-1}S_{12}S_{22}^{-1} \end{pmatrix} \end{aligned} \tag{C.2}$$

To find the logarithmic probability of variable $\boldsymbol{y}(t+1)$ and $\boldsymbol{y}(t)$ conditional on $\{\boldsymbol{z}\}_{1:t}$ and $\{\boldsymbol{u}\}_{1:t}$, we have:

$$
\begin{aligned}
\log P(\boldsymbol{y}(t+1), \boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:t}, \{\boldsymbol{u}\}_{1:t}) &= \log P(\boldsymbol{y}(t)|\boldsymbol{y}(t+1), \{\boldsymbol{z}\}_{1:T}, \{\boldsymbol{u}\}_{1:T}) + \log P(\boldsymbol{y}(t+1)|\{\boldsymbol{z}\}_{1:T}, \{\boldsymbol{u}\}_{1:T}) \\
&= \log P(\boldsymbol{y}(t)|\boldsymbol{y}(t+1), \{\boldsymbol{z}\}_{1:t}, \{\boldsymbol{u}\}_{1:t}) + \log P(\boldsymbol{y}(t+1)|\{\boldsymbol{z}\}_{1:T}, \{\boldsymbol{u}\}_{1:T}) \\
&= \log P(\boldsymbol{y}(t+1)|\boldsymbol{y}(t), \{\boldsymbol{z}\}_{1:t}, \{\boldsymbol{u}\}_{1:t}) + \log P(\boldsymbol{y}(t)|\{\boldsymbol{z}\}_{1:t}, \{\boldsymbol{u}\}_{1:t}) \\
&\quad - \log P(\boldsymbol{y}(t+1)|\{\boldsymbol{z}\}_{1:t}, \{\boldsymbol{u}\}_{1:t}) + \log P(\boldsymbol{y}(t+1)|\{\boldsymbol{z}\}_{1:T}, \{\boldsymbol{u}\}_{1:T}) \\
&= -\frac{1}{2}(\boldsymbol{y}(t+1) - A\boldsymbol{y}(t) - B\boldsymbol{u}(t))^{'} Q^{-1}(\boldsymbol{y}(t+1) - A\boldsymbol{y}(t) - B\boldsymbol{u}(t)) \\
&\quad - \frac{1}{2}(\boldsymbol{y}(t) - P_t^t)^{'}(V_t^t)^{-1}(\boldsymbol{y}_t - P_t^t) \\
&\quad + \frac{1}{2}(\boldsymbol{y}(t+1) - P_{t+1}^t)^{'}(V_{t+1}^t)^{-1}(\boldsymbol{y}(t+1) - P_{t+1}^t) \\
&\quad - \frac{1}{2}(\boldsymbol{y}(t+1) - P_{t+1}^T)^{'}(V_{t+1}^T)^{-1}(\boldsymbol{y}(t+1) - P_{t+1}^T) + \dots \\
&= -\frac{1}{2}y(t+1)^{'}(Q^{-1} - (V_{t+1}^t)^{-1} + (V_{t+1}^T)^{-1})\boldsymbol{y}(t+1) \\
&\quad - \frac{1}{2}\boldsymbol{y}(t+1)^{'}(-Q^{-1}A)\boldsymbol{y}(t) - \frac{1}{2}\boldsymbol{y}(t)^{'}(-A^{'}Q^{-1})\boldsymbol{y}(t+1) \\
&\quad - \frac{1}{2}\boldsymbol{y}(t)^{'}(A^{'}Q^{-1}A + (V_t^t)^{-1})\boldsymbol{y}(t) + \boldsymbol{y}(t)^{'}((V_t^t)^{-1}P_t^t - A^{'}Q^{-1}B\boldsymbol{u}(t)) \\
&\quad + \dots
\end{aligned} \tag{C.3}
$$

C.2 Variance Updated Equations

$$
\begin{aligned}
V_t^T &= V_t^t + J_t(V_{t+1}^T - V_{t+1}^t)J_t^{'} \\
V_{T,T-1}^T &= (I - K_T)AV_{T-1}^{T-1}
\end{aligned} \tag{C.4}
$$

where

$$J_t = V_t^t A^{'}(V_{t+1}^t)^{-1}$$

In the following paragraph, we will show how to derive two equations in (C.4).

Proof. Comparing the same structure in the first four terms in (C.1) and (C.3), we invoke the Woodbury matrix identity to calculate S_{22}^{-1} and $S_{22}^{-1}S_{21}$ by using the results of (B.3):

$$
\begin{aligned}
S_{22}^{-1} &= (A^{'}Q^{-1}A + (V_t^t)^{-1})^{-1} \\
&= V_t^t - V_t^t A^{'}(V_{t+1}^t)^{-1}AV_t^t \\
&= V_t^t - J_t V_{t+1}^t J_t^{'}
\end{aligned} \tag{C.5}
$$

where we could define

$$J_t = V_t^t A^{'}(V_{t+1}^t)^{-1} \tag{C.6}$$

Then we calculate $S_{22}^{-1}S_{21}$:

$$\begin{aligned} S_{22}^{-1}S_{21} &= -(V_t^t - J_t V_{t+1}^t J_t^{'})A^{'}Q^{-1} \\ &= -V_t^t A^{'}(I - (Q + AV_t^t A^{'})^{-1}AV_t^t A^{'})Q^{-1} \\ &= -V_t^t A^{'}(Q + AV_t^t A^{'})^{-1} \\ &= -J_t \end{aligned} \tag{C.7}$$

Next we could easily calculate V_t^T and $V_{t+1,t}^T$ by equations (C.2), (C.6), (C.7) and (C.8):

$$\begin{aligned} V_t^T &= S_{22}^{-1} + S_{22}^{-1}S_{21}(S_{11} - S_{12}S_{22}^{-1}S_{21})^{-1}S_{12}S_{22}^{-1} \\ &= V_t^t - J_t V_{t+1}^t J_t^{'} + (-J_t)V_{t+1}^T(-J_t^{'}) \\ &= V_t^t + J_t(V_{t+1}^T - V_{t+1}^t)J_t^{'} \end{aligned} \tag{C.8}$$

and

$$\begin{aligned} V_{t+1,t}^T &= -(S_{11} - S_{12}S_{22}^{-1}S_{21})^{-1}S_{12}S_{22}^{-1} \\ &= V_{t+1}^T J_t^{'} \\ &= (V_{t+1}^{t+1} + J_{t+1}(V_{t+2}^T - V_{t+2}^{t+1})J_{t+1}^{'})J_t^{'} \\ &= (V_{t+1}^{t+1} + J_{t+1}(V_{t+2,t+1}^T - AV_{t+1}^{t+1}))J_t^{'} \\ &= V_{t+1}^{t+1}J_t^{'} + J_{t+1}(V_{t+2,t+1}^T - AV_{t+1}^{t+1})J_t^{'} \end{aligned} \tag{C.9}$$

By using the equation (B.4), (C.7) and (C.9), this recursion is initialized with

$$\begin{aligned} V_{T,T-1}^T &= V_T^T J_{T-1}^{'} \\ &= (V_T^{T-1} - K_T V_T^{T-1})J_{T-1}^{'} \\ &= (I - K_T)AV_{T-1}^{T-1} \end{aligned} \tag{C.10}$$

C.3 Mean Updated Equations

Followed by the derivation of mean updated equation:

The mean $\{\boldsymbol{y}\}_t^T$ could be derived by comparing the fifth term between the equation (C.1) and (C.3) and then apply the results from equations (C.5) to (C.7):

$$\begin{aligned} P_t^T &= S_{22}^{-1}(V_t^t)^{-1}P_t^t - S_{22}^{-1}A^{'}Q^{-1}B\boldsymbol{u}(t) - S_{22}^{-1}S_{21}P_{t+1}^T \\ &= (I - J_t A)P_t^t - J_t B\boldsymbol{u}(t) + J_t P_{t+1}^T \\ &= P_t^t + J_t(P_{t+1}^T - AP_t^t - B\boldsymbol{u}(t)) \\ &= P_t^t + J_t(P_{t+1}^T - P_{t+1}^t) \end{aligned} \tag{C.11}$$

□