Inference and Characterization of Planar Trajectories

Zhanglong Cao

a thesis submitted for the degree of

Doctor of Philosophy

at the University of Otago, Dunedin,

New Zealand.

Contents

List of Tables

List of Figures

All knowledge is, in the final analysis, history.
All science are, in the abstract, mathematics.
All judgments are, in their rationale, statistics.
– C. Radhakrishna Rao.

Abstract

This is the abstract of this thesis.

Acknowledgments

Thank you ALL.

This is an acknowledgment.

Chapter 1

Introduction

1.1 Problem Statement

The problem in the world I explore in this paper is....... The reason why it is an important problem is......

The lends I adopt to explore this problem is.... it is a useful lens because..........

I reach the following findings.......

This study contributes to.........

The Global Positioning System (abbreviated as GPS) is a space-based navigation system consisting 24 satellites running around the earth orbits at the altitude 202,00 km. These satellites cover 98% of the earth surface and at least 4 satellites are available at anytime from anywhere on the earth or near the earth orbit. With four or more satellites, a GPS receiver can provide geographic information to them and triangulate its location on the ground, such as longitude, latitude and elevation. These receivers are using passive locating technology, by which they can receive signals without transmitting any data. Therefore, GPS is used in plenty applications in military and general public, including aircraft tracking, vehicle navigation, surveying, astronomy and so on. Tractors, working on an orchard or vineyard, are often mounted with GPS units that provide position and velocity information to orchardist and landlords. With this information, an orchardist is able to follow the trajectory and motion patterns of tractors and monitor its working status.

Researchers are wondering how to get a more accurate estimation for the position of a moving vehicle and to track its moving trajectory instantly. That is the problem, which has been confusing researchers for a few years and I am exploring in this paper.

In 1960, R.E. Kalman published his famous paper describing a recursive solution to the discrete data linear filtering problem. The Kalman filter is a set of mathematical equations that provides an efficient computational means to estimate the state of a process in a recursive way, that minimizes the mean of the squared error Greg Welch (2006). Fernando Tusell gave a review of some R packages, which could be used to fit data with Kalman Filter methods Tusell (2011). However, limited to its property, Kalman Filter is tied up for a dynamic system, where the parameters and noise variances are unknown. In some dynamic systems, the variances are obtained based on the system identification algorithm, correlation method and least squares fusion criterion. To solve this issue, a self-tuning weighted measurement fusion Kalman filter is presented in Chenjian RAN (2010). Also, a new adaptive Kalman filter will be a good choice Oussalah and De Schutter (2001).

1.2 Spline Reconstruction

Kalman Filter is a good choice. And there probably a more simple and better way to fit our data – the splines. Hastie introduces several kinds of splines that we could use in his book Hastie *et al.* (2008). The core idea of splines is to augment the vector of inputs X with additional variables, then use linear models in this space of derived input features. Adding constraints to construct basis functions $h_i(x), i = 1, 2, \dots, n$, a linear basis expansion in X could be represented as

$$f(x) = \sum_{i=1}^{n} \theta_i h_i(x)$$

Once the basis functions $h_i(x)$ have been determined, the models are linear in the variables space.

Smoothing spline is a basic spline and could be used in two-dimensions. Mehdi Zamani proved that in two-dimensional spaces, splines have the advantages of simplicity and less computational operations Zamani (2010). The most important part of this method is to find the parameter λ. We have a process (function) to calculate the RSS (Residual Sum of Square), where the smallest RSS is, the best λ would be

$$RSS(f, \lambda) = \sum_{n=1}^{n} \{(y_i) - g(x_i)\}^2 + \lambda \int g''(x_i)^2$$

The function $g(x)$ that minimizes RSS is a natural cubic spline with knots at $x_1, ..., x_n$ Hastie *et al.* (2008).

Some ways are give to estimate the parameter Wood (2000), Kim and Gu (2004). However, we hope that λ could be a piecewise parameter, rather than a constant number, then we will have piecewise penalty functions. Grace Wahba introduced adaptive splines and a method to calculate piecewise parameters Donoho *et al.* (1995) and Ziyue Liu and Wensheng Guo improved the method Liu and Guo (2010). But to get these parameters, we have to compromise to use adaptive smoothing splines.

Using splines to fit data has been studies by many researchers. Some simple examples are given in the book Hastie *et al.* (2008). Moreover, Fujiichi Yoshimoto offers a method can treat not only data with a smooth underlying function, but also data with an underlying function having discontinuous points and/or cusps Yoshimoto *et al.* (2003). Dan Simon discussed a new type of algebraic spline, which is used to derive a filter for smoothing or interpolating discrete data points in eighth-order Simon (2004).

Based on these methods, Matthew decided to create a new spline. Temporarily, we name the new spline "Tractor Spline". This spline has some properties. This spline has two linear segments outside the knots and $n-1$ internal cubics. The spline needs $4n = 4(n-1)+2\times 2$ constraints per component. These are provided by specifying the values and first derivatives at the knots. This means $2n$ constraints per component but they count twice since they constrain the spline on both sides of the knot. After that, once we know the functions of tractor spline, the position of tractor at time t will be known. Moreover, we hope to integrate the details of orchard to improve predictions and begin to characterise the type of tractor trajectories at a higher level.

The observed position data set y_i always comes with some errors ε_i, which are assumed independent Gaussian distribution with variance σ_n^2. A popular method for finding $f(x)$ that fits these data is to augment/replace the vector of inputs $\mathbf{X}$ with additional variables, which are transformations of $\mathbf{X}$, and then use linear models in this new space of derived input features.Trevor Hastie (2009)

In regression problem, linear regression, linear discriminant analysis, logistic regression and separating hyperplanes all rely on a linear model. With the good property of linear model, easy to be interpreted and first order Taylor approximation to $f(t)$, it is more convenient to represent $f(t)$ by linear model. However, the true function $f(t)$ is unlikely to be an actual linear function in space $\mathbb{R}$. Researchers found some methods for moving beyond linearity. One of them is replacing the vector of inputs $\mathbf{T}$ with its transformations as new variables, and then use linear models in this new space of derived input features.

Denote by $h_m(t) : \mathbb{R} \mapsto \mathbb{R}$ the mth transformation of t, $m = 1, \cdots, M$. We then model

$$f(t) = \sum_{m=1}^{M} \beta_m h_m(t). \tag{1.1}$$

a linear basis expansion of $\mathbf{t}$ in $\mathbb{R}$, where $h_m(t)$ are named basis functions, β_m are coefficients. Once the basis functions h_m have been determined, the models are linear in these new variables, and the fitting proceeds as before.

Suppose we are given observed data $t_1, t_2, \cdots, t_n$ on interval $[0,1]$, satisfying $0 \leq t_1 < t_2 < \cdots < t_n \leq 1$. A piecewise polynomial function $f(t)$ can be obtained by dividing the interval into contiguous intervals $(t_1, t_2), \cdots, (t_{n-1}, t_n)$, and representing f by a separate polynomial in each interval. The points t_i are called knots. For example,

$$f(t) = d_i(t-t_i)^3 + c_i(t-t_i)^2 + b_i(t-t_i) + a_i, \tag{1.2}$$

for given coefficients d_i, c_i, b_i and a_i, where $t_i \leq t \leq t_{i+1}$, $i = 1, 2, \cdots, n$. f is a cubic spline on $[0,1]$ if (1) on each intervals f is a polynomial; (2) the polynomial pieces fit together at knots t_i in such a way that f itself and its first and second derivatives are continuous at each t_i. If the second and third derivatives of f are zero at 0 and 1, f is said to be a natural cubic spline. These conditions are called natural boundary conditions.

Over all spline functions $f(t)$ with two continuous derivatives fitting these observed data, the curve estimate $\hat{f}(t)$ will be defined to be the minimizer the following penalized residual sum of squares, **[edited expressions]**

$$\text{MSE}(f, \lambda) = \frac{1}{n}\sum_{i=1}^{n}(f(t_i) - y_i)^2 + \lambda \int_0^1 (f''(t))^2 dt \tag{1.3}$$

where λ is a fixed smoothing parameter, (t_i, y_i), $i = 1, \cdots, n$ are observed data and $0 \leq t_1 < t_2 < \cdots < t_n \leq 1$. In equation (3.2), the smoothing parameter λ controls the trade-off between over-fitting and bias,

$$\begin{cases} \lambda = 0: & f \text{ can be any function that interpolates the data,} \\ \lambda = \infty: & \text{the simple least squares line fit since no second derivative can be tolerated.} \end{cases} \tag{1.4}$$

In our case, the velocity data set v_i with some independent Gaussian distributed errors $\varepsilon_i \sim N(0, \frac{\sigma_n^2}{\gamma})$ are used to estimate $f(t)$ simultaneously. f is a linear combination of basis functions, as shown in equation (3.1), in the meantime, f' is a linear combination of the first derivative of these basis functions

$$f'(t) = \sum_{m=1}^{M} \alpha_m h'_m(t). \tag{1.5}$$

The velocity information is incorporated into MSE equation (3.2) by the addition of velocity term $(f'(t_i) - v_i)^2$. Then it becomes

$$\text{MSE}(f, \lambda, \gamma) = \frac{1}{n}\sum_{i=1}^{n}(f(t_i) - y_i)^2 + \frac{\gamma}{n}\sum_{i=1}^{n}(f'(t_i) - v_i)^2 + \lambda \int_0^1 (f''(t))^2 dt, \tag{1.6}$$

and $\hat{f}$ is the minimizer of the MSE equation (1.6).

In the model $y = f(t) + \varepsilon$, it is reasonable to assume that the observed data y_i is Gaussian distribution with mean $f(t_i)$ and variance σ_n^2. In a similar way, the velocity is estimated as $v = f'(t) + \frac{\varepsilon}{\gamma}$, where v_i is Gaussian distribution with mean $f'(t_i)$ and variance $\frac{\sigma_n^2}{\gamma}$. Then the joint distribution of $\mathbf{y}, \mathbf{v}, f(t)$ and $f'(t)$ is normal with zero mean and a covariance matrix, which can be estimated through Gaussian Process Regression.

Given a series of such fixes, I want to construct the most likely path taken by the tractor with some smart modelling of the movement. I then want to develop higher level characterisations of the trajectories and apply this to more general problems in curve and object recognition.

1.2.1 Cross Validation

1.2.2 K-Fold Cross Validation

Based on the procedure given by Wahba and Wold (1975), we follow the improved steps to calculate a K-fold cross validation.

Step 1. Remove the first data t_1 and last date t_n from the dataset.

Step 2. Divide dataset into k groups:

$$\begin{aligned}
&\text{Group 1} : t_2, t_{2+k}, \cdots \\
&\text{Group 2} : t_3, t_{3+k}, \cdots \\
&\vdots \\
&\text{Group k} : t_{k+1}, t_{2k+1}, \cdots
\end{aligned}$$

Step 3. Guess values of $\lambda_{down}, \lambda_{up}$ and γ.

Step 4. Delete the first group of data. Fit a smoothing spline to the first data, the rest groups of dataset and the last data, with $\lambda_{down}, \lambda_{up}$ and γ in step 3. Compute the sum of squared deviations of this smoothing spline from the deleted data points.

Step 5. Delete instead the second group of data. Fit a smoothing spline to the remaining data with $\lambda_{down}, \lambda_{up}$ and γ. Compute the sum of squared deviations of the spline from deleted data points.

Step 6. Repeat Step 5 for the 3rd, 4th, $\cdots$, kth group of data.

Step 7. Add the sums of squared deviations from steps 4 to 6 and divide by k. This is the cross validation score of three parameters $\lambda_{down}, \lambda_{up}$ and γ.

Step 8. Vary $\lambda_{down}, \lambda_{up}$ and γ systematically and repeat steps 4-7 until CV shows a minimum.

1.3 Gaussian Process Regression

A Gaussian Process is a collection of random variables, any finite number of which have a joint Gaussian distribution, Rasmussen and Williams (2006).

A GP is fully defined by its mean $m(t)$ and covariance $K(s,t)$ functions as

$$m(t) = \mathbb{E}[f(t)] \tag{1.7}$$

$$K(s,t) = \mathbb{E}[(f(s) - m(s))(f(t) - m(t))], \tag{1.8}$$

where s and t are two variables, and a function f distributed as such is denoted in form of

$$f \sim GP(m(t), K(s,t)). \tag{1.9}$$

Usually the mean function is assumed to be zero everywhere.

Given a set of input variables $\mathbf{T}$ for function $f(t)$ and the output $\mathbf{y} = f(\mathbf{T}) + \varepsilon$ with independent identically distributed Gaussian noise ε with variance σ_n^2, we can use the above definition to predict the value of the function $f_* = f(t_*)$ at a particular input t_*. As the noisy observations becoming

$$\text{cov}(y_p, y_q) = K(t_p, t_q) + \sigma_n^2 \delta_{pq} \tag{1.10}$$

where δ_{pq} is a Kronecker delta which is one iff $p = q$ and zero otherwise, the joint distribution of the observed outputs $\mathbf{y}$ and the estimated output f_* according to prior is

$$\begin{bmatrix} \mathbf{y} \\ f_* \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(\mathbf{T},\mathbf{T}) + \sigma_n^2 I & K(\mathbf{T}, t_*) \\ K(t_*, \mathbf{T}) & K(t_*, t_*) \end{bmatrix}\right). \tag{1.11}$$

The posterior distribution over the predicted value is obtained by conditioning on the observed data

$$f_* | \mathbf{y}, \mathbf{T}, t_* \sim N(\bar{f}_*, \text{cov}(f_*)) \tag{1.12}$$

where

$$\bar{f}_* = \mathbb{E}[f_*|\mathbf{y}, \mathbf{T}, t_*] = K(t_*, \mathbf{T})[K(\mathbf{T}, \mathbf{T}) + \sigma_n^2 I]^{-1}\mathbf{y}, \tag{1.13}$$

$$\text{cov}(f_*) = K(t_*, t_*) - K(t_*, \mathbf{T})[K(\mathbf{T}, \mathbf{T}) + \sigma_n^2 I]^{-1} K(\mathbf{T}, t_*). \tag{1.14}$$

We now add velocity information $\mathbf{v} = f'(\mathbf{T}) + \varepsilon'$, where ε' is independent distributed Gaussian noise with variance $\frac{\sigma_n^2}{\gamma}$.

It is expected that a position point y_i and velocity point v_i are all effected by other points $\mathbf{y}$ and $\mathbf{v}$. So the covariance matrix for $\mathbf{y}$ and $\mathbf{v}$ is

$$\Sigma(\mathbf{y}, \mathbf{v}) = \begin{bmatrix} \text{cov}(\mathbf{y}, \mathbf{y}) & \text{cov}(\mathbf{y}, \mathbf{v}) \\ \text{cov}(\mathbf{v}, \mathbf{y}) & \text{cov}(\mathbf{v}, \mathbf{v}) \end{bmatrix}, \tag{1.15}$$

where obviously $\text{cov}(\mathbf{y}, \mathbf{v}) = \text{cov}(\mathbf{v}, \mathbf{y})$. Then the joint distribution is

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{v} \end{bmatrix} \sim N(\mu_{y,v}, \Sigma_{y,v}). \tag{1.16}$$

Define f_* and f'_* the estimated position and velocity values at point t_*. From equation (1.15) and using similar idea, it is easily to get the covariance matrices

$$\begin{aligned} \Sigma(f_*, \mathbf{v}) &= \begin{bmatrix} \text{cov}(f_*, f_*) & \text{cov}(f_*, \mathbf{v}) \\ \text{cov}(\mathbf{v}, f_*) & \text{cov}(\mathbf{v}, \mathbf{v}) \end{bmatrix}, \\ \Sigma(\mathbf{y}, f'_*) &= \begin{bmatrix} \text{cov}(\mathbf{y}, \mathbf{y}) & \text{cov}(\mathbf{y}, f'_*) \\ \text{cov}(f'_*, \mathbf{y}) & \text{cov}(f'_*, f'_*) \end{bmatrix}, \\ \Sigma(f_*, f'_*) &= \begin{bmatrix} \text{cov}(f_*, f_*) & \text{cov}(f_*, f'_*) \\ \text{cov}(f'_*, f_*) & \text{cov}(f'_*, f'_*) \end{bmatrix}, \end{aligned} \tag{1.17}$$

[will need to give the form of these covariances at some point. in an appendix? i think you need discussion of how f' is related to f for a GP]

1.3.1 A Reproducing Kernel in Space $\mathbb{H}$

A Reproducing Kernel in Space $\mathbb{H}$.

1.4 Filtering Methods

1.5 Sequential Monte Carlo Markov Chain Algorithm

Chapter 2

Tractor Spline Theory

2.1 Writing Something

I'm thinking, I should start writing from Spline method, without adaptive terms on λ. An advanced Cross Validation method is given. Then when I tried to use this method to construct trajectories (application on real data), there appears some issues. So I brought adaptive terms.

It's a batch case method. Trajectory is reconstructed from a batch of data and some issues appeared. The numeric simulations proved that this new method is better. However, some issues still exist in real data application (long-gap-curve).

Then I will introduce Gaussian Process Regression and how it could estimate Smoothing Spline. Based on some references and tractor spline, I will prove some relationships between Tractor Spline and GPR. this is a spin-off results.

These are all batch method. Even if researchers could do some online smoothing, the computation time cost a lot.

Why don't we do online estimation by using some efficiency method? Introduce filters and other methods (Probably Kalman filter, particle filter and other filter) and dynamic linear regression models. Then the proposed method, parameters estimation and something else. It's an online case.

Parameter estimation methods: Metropolis-Hastings methods, better than MLE because the latter doesn't give the distribution of the estimates.

Moreover, I need to figure out how these different methods working and the advantages and disadvantages they have.

2.2 Introduction

In a vehicular system, the simplest way of getting the trajectory of a moving-object is connecting position points by a sequence of lines (line-based trajectory representation) in a 3-dimensional or 4-dimensional space-time Agarwal *et al.* (2003). Due to the noises generated from observation units, one can use regression method to find the best fitting returning the smallest sum square errors among all the sequences. Consider a regression model $y_i = f(t_i) + \epsilon_i$, where $a \leq t_1 < \cdots < t_n \leq b$ and $f \in C^2[a, b]$ is an unknown smooth function, $(\epsilon_i)_{i=1}^n \sim N(0, \sigma^2)$ are random errors. In a classical parametric regression, f is assumed having the form $f(x, \beta)$, which is known up to the data estimated parameters β Kim and Gu (2004). When $f(x, \beta)$ is linear in β, we will have a standard linear model. However, most of the natural moving objects return smooth trajectories without any angles. Therefore, a spline method is used to construct such trajectories. A curved-base method uses a parametric cubic function $P(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$ to obtain a spline that passes through any given sequence of joint position-velocity paired points $(x_1, v_1), (x_2, v_2), \cdots (x_n, v_n)$ Yu *et al.* (2004). In Yang and Sukkarieh (2010), an efficient and analytical continuous curvature path-smoothing algorithm based on parametric cubic Bézier curves is proposed. It can fit ordered sequential points smoothly. In computer (or computerized) numerical control (CNC), Altintas and Erkorkmaz Erkorkmaz and Altintas (2001) presented a quintic spline trajectory generation algorithm connecting a series of reference knots that produces continuous position, velocity and acceleration profiles.

However, a parametric approach only captures features contained in the preconceived class of functions Yao *et al.* (2005) and increases model bias. To avoid this, an alternative approach called nonparametric model was invented. Rather than giving a specified parameter, it is desired to reconstruct f from the data $y(t_i) \equiv y_i$, $i = 1, \cdots, n$ Craven and Wahba (1978). Smoothing spline estimate of the f function appears as a solution to the following minimization problem: Find $\hat{f} \in C^2[a, b]$ that minimizes the penalized residual sum of squares:

$$\text{RSS} = \sum_{j=1}^{n} (y_j - f(t_j))^2 + \lambda \int_a^b f''(t)^2 dt \tag{2.1}$$

for pre-specified value $\lambda > 0$ Aydin and Tuzemen (2012). In equation (2.1), the first part is residual sum square and it penalizes the lack of fit. The second part is roughness penalty term weighted by a smoothing parameter λ, which varies from 0 to $+\infty$ and establishes a trade-off between interpolation and a linear model. The motivation of roughness penalty term is from a formalization of a mechanical device: if a thin piece of flexible wood, called a spline, is bent to the shape of the graph g, then the leading term in the strain energy is proportional to $\int f''^2$ Green and Silverman (1993). The cost of equation (2.1) is determined not only by its goodness-of-fit to the data quantified by the residual sum of squares, but also by its roughness Schwarz (2012). For a given λ, minimizing equation (2.1) will give the best compromise between smoothness and goodness-of-fit. Notice that the first term in equation (2.1) depends only on the values of f at knots $t_i, i = 1, \cdots, n$. In the book, the authors show that the function that minimizes the roughness penalty for fixed values of $f(t_i)$ is a cubic spline: an interpolation of points via a continuous piecewise cubic function, with continuous first and second derivatives Green and Silverman (1993). The continuity requirements uniquely determine the interpolating spline, except at the boundariesSealfon *et al.* (2005).

A conventional smoothing spline is controlled by one single parameter, which controls the smoothness of a spline on the whole domain. A natural extension is to allow the smoothing parameter to vary as a penalty function of the independent variable, adapting to the change of roughness in different domains Silverman (1985), Donoho *et al.* (1995). In this way, a new objective function is formulated in the form of

$$\sum_{j=1}^{n}(y_j - f(x_j))^2 + \int_T \lambda(t) f''(t)^2 dt, \tag{2.2}$$

by minimizing which, the best estimation $\hat{f}$ can be found. This approach makes adaptive smoothing as a minimization problem with a new penalty term.

Similar to conventional smoothing spline problem, researchers are wondering how to choose the penalty function $\lambda(t)$. The fundamental idea of nonparametric smoothing is to let the data choose the amount of smoothness, which consequently decides the model complexity Gu (1998). Most of them focus on data driven criteria, such as cross validation (CV), generalized cross validation (GCV) Craven and Wahba (1978) and generalized maximum likelihood (GML) Wahba (1985). A new challenge is posed that the smoothing parameter becomes a function and is varying in domains. The structure of this penalty function controls the complexity on each domain and the whole final model. Liu and Guo proposed to approximate the penalty function with an indicator and extended the generalized likelihood to the adaptive smoothing spline Liu and Guo (2010).

In this paper, we propose an adaptive smoothing spline method based on Hermite Spline basis functions to get reconstruction of f and f' from noisy data $\mathbf{y}$ and $\mathbf{v}$. Rather than only using residuals of $f(t_i) - y_i$ term in the objective function in equation (2.2), we added the residuals of $f'(t_i) - v_i$ as a new term containing a new parameter γ. In this way, the spline keeps a balance on both observed $\mathbf{y}$ and $\mathbf{v}$. Using new generated basis functions, we reconstruct a smoothing spline on the whole interval $[a, b]$. Derived from the new objective function, an advanced cross validation formula of $f(t)$ and $f'(t)$ is given. This method can be used in either getting true signal from noisy data or moving-object database.

2.3 Tractor Spline

2.3.1 Objective Function

In a 2D curve nonparametric regression, consider n time points $t_{1:n}$, such that $a \leq t_1, \cdots, t_n \leq b$. Let $z_i = (x_i, y_i)$ and $w_i = (u_i, v_i)$ for $i = 1, \cdots, n$. We define a positive piecewise constant function $\lambda(t)$:

$$\lambda(t) = \lambda_i > 0, \tag{2.3}$$

where $t_i \leq t < t_{i+1}, t_0 = a, t_{n+1} = b$, that will control the curvature penalty of each interval. For a function $f : [a, b] \longrightarrow \mathbb{R}^2$ and $\gamma > 0$, define the objective function

$$J[f] = \frac{1}{n}\sum_{i=1}^{n}(f(t_i) - z_i)^2 + \frac{\gamma}{n}\sum_{i=1}^{n}(f'(t_i) - w_i)^2 + \sum_{i=0}^{n}\lambda_i\int_{t_i}^{t_{i+1}} f''(t)^2 dt, \tag{2.4}$$

where γ is a coefficient of the velocity information $\mathbf{v}$ and it weights the residuals between $\mathbf{f}'$ and $\mathbf{v}$, and $\lambda(t)$ is the smoothing parameter function.

Theorem 1. *For $n \geq 2$, the objective function $J[f]$ is minimized by a cubic spline that is linear outside the knots.*

The solution to the objective function (2.4) is called tractor spline.

In the following, we divide the 2D function $f(x, y)$ into two sub functions $f_x(t)$ on x-axis and $f_y(t)$ on y-axis. Compared with other parameters, choosing time t to be the parameter has some advantages: 1. The expressions of all the constraints are simpler Zhang *et al.* (2013); 2. It can be simply applied from 2-dimension to 3-dimension by adding an extra z-axis.

2.3.2 Basis Functions

Suppose we have a time series sequence of observed dataset $a = t_1 < t_2 < \cdots < t_n = b$. The function $f(t)$ (stands for $f_y(t)$) defined on this interval $[t_1, t_n]$ is called tractor spline, if it is the solution to the objective function (2.4). Then it has the following property: on each interior interval (t_i, t_{i+1}), $i = 2, \cdots, n-2$, $f(t)$ is a cubic polynomial, but on interval (t_1, t_2) and (t_{n-1}, t_n) can be linear; $f(t)$ fits together at each point t_i in such a way that $f(t)$ itself and its first derivatives are continuous at each t_i, $i = 2, \cdots, n-2$.

Using Hermite interpolation on an arbitrary interval $[t_i, t_{i+1}]$, the cubic spline basis functions can be constructed as follows

$$h_{00}^{(i)}(t) = \begin{cases} 2(\frac{t-t_i}{t_{i+1}-t_i})^3 - 3(\frac{t-t_i}{t_{i+1}-t_i})^2 + 1 & t_i \le t < t_{i+1} \\ 0 & \text{otherwise} \end{cases}, \tag{2.5}$$

$$h_{10}^{(i)}(t) = \begin{cases} \frac{(t-t_i)^3}{(t_{i+1}-t_i)^2} - 2\frac{(t-t_i)^2}{t_{i+1}-t_i} + (t-t_i) & t_i \le t < t_{i+1} \\ 0 & \text{otherwise} \end{cases}, \tag{2.6}$$

$$h_{01}^{(i)}(t) = \begin{cases} -2(\frac{t-t_i}{t_{i+1}-t_i})^3 + 3(\frac{t-t_i}{t_{i+1}-t_i})^2 & t_i \le t < t_{i+1} \\ 0 & \text{otherwise} \end{cases}, \tag{2.7}$$

$$h_{11}^{(i)}(t) = \begin{cases} \frac{(t-t_i)^3}{(t_{i+1}-t_i)^2} - \frac{(t-t_i)^2}{t_{i+1}-t_i} & t_i \le t < t_{i+1} \\ 0 & \text{otherwise} \end{cases}. \tag{2.8}$$

Then a Hermite spline $f^{(i)}(t)$ on interval $[t_i, t_{i+1})$ with points $p_i = \{y_i, v_i\}$ and $p_{i+1} = \{y_{i+1}, v_{i+1}\}$ can be expressed as

$$f^{(i)}(t) = h_{00}^{(i)}(t)y_i + h_{10}^{(i)}(t)v_i + h_{01}^{(i)}(t)y_{i+1} + h_{11}^{(i)}(t)v_{i+1}. \tag{2.9}$$

To construct a tractor spline on the entire interval $[t_1, t_n]$, the new basis functions are defined in such way, that $N_1 = h_{00}^{(1)}$, $N_2 = h_{10}^{(1)}$, and for all $k = 1, 2, \ldots, n-2$,

$$N_{2k+1} = \begin{cases} h_{01}^{(k)} + h_{00}^{(k+1)} & \text{if } t < t_n \\ 2(\frac{t-t_{n-1}}{t_n-t_{n-1}})^3 - 3(\frac{t-t_{n-1}}{t_n-t_{n-1s}})^2 + 1 & \text{if } t = t_n \end{cases}, \tag{2.10}$$

$$N_{2k+2} = \begin{cases} h_{11}^{(k)} + h_{10}^{(k+1)} & \text{if } t < t_n \\ \frac{(t-t_{n-1})^3}{(t_n-t_{n-1})^2} - 2\frac{(t-t_{n-1})^2}{t_n-t_{n-1}} + (t-t_{n-1}) & \text{if } t = t_n \end{cases}, \tag{2.11}$$

and

$$N_{2n-1} = \begin{cases} h_{01}^{(n-1)} & \text{if } t < t_n \\ -2(\frac{t-t_{n-1}}{t_n-t_{n-1}})^3 + 3(\frac{t-t_{n-1}}{t_n-t_{n-1}})^2 & \text{if } t = t_n \end{cases}, \tag{2.12}$$

$$N_{2n} = \begin{cases} h_{11}^{(n-1)} & \text{if } t < t_n \\ \frac{(t-t_{n-1})^3}{(t_n-t_{n-1})^2} - \frac{(t-t_{n-1})^2}{t_n-t_{n-1}} & \text{if } t = t_n \end{cases}. \tag{2.13}$$

Theorem 2. *On $[t_1, t_n]$, the functions $N_1, \dots, N_{2n}$ provide a basis for the set of functions which are continuous, have continuous first derivatives and are cubic on each open interval (t_i, t_{i+1}), where $i = 1, \cdots, n-1$.*

As independent basis functions, $N_1(t), \cdots, N_{2n}(t)$ span a $2n$ dimensional function space $\mathbb{H}$. For any $f \in \mathbb{H}$, it can be represented in the form of

$$f = \sum_{k=1}^{2n} \theta_k N_k(t), \tag{2.14}$$

where $\{\theta_k\}_{k=1}^{2n}$ are parameters.

Figure (2.1) presents two basis functions on an arbitrary interval $[t_k, t_{i+2})$ where they are continuous and differential. At the interior joint knot t_k, basis functions in the previous and following interval share the same position y_k and velocity v_k.

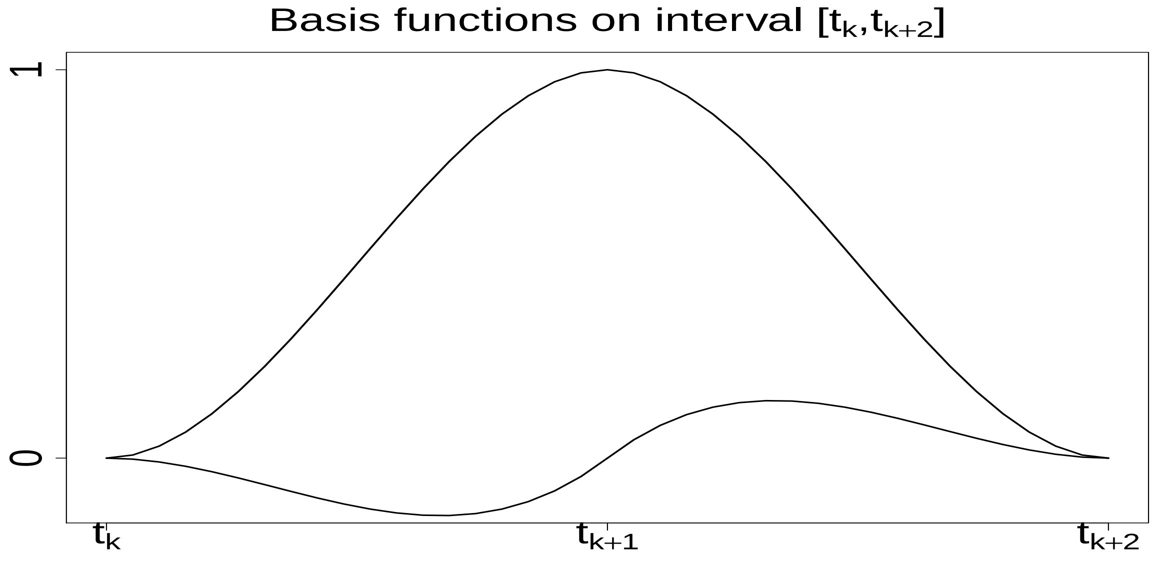


Figure 2.1: The two basis functions N_{2k+1} and N_{2k+2} on interval $[t_k, t_{k+2})$. It is apparently that these basis functions are continuous on this interval and have continuous first derivatives.

2.3.3 Solution to The Objective Function

Basis functions have been defined in the previous subsection, therefore the tractor spline $f(t)$ on $[a, b]$, where $a \le t_1 < t_2 < \cdots < t_{n-1} < t_n \le b$, can be found by minimizing objective function (2.4), which reduces to

$$\text{MSE}(\theta, \lambda, \gamma) = (\mathbf{y} - \mathbf{B}\theta)^\top (\mathbf{y} - \mathbf{B}\theta) + \gamma(\mathbf{v} - \mathbf{C}\theta)^\top (\mathbf{v} - \mathbf{C}\theta) + n\theta^\top \Omega_\lambda \theta, \tag{2.15}$$

where $\{\mathbf{B}\}_{ij} = N_j(t_i)$, $\{\mathbf{C}\}_{ij} = N'_j(t_i)$ and $\{\Omega_{2n}^{(k)}\}_{jk} = \int_{t_k}^{t_{k+1}} \lambda_k N''_j(t) N''_k(t) dt$. After substituting the series observation $t_1, \cdots, t_n$ into basis functions, we get $N_1(t_1) = 1, N_1(t_2) = 0, \cdots, N_{2k-1}(t_k) = 1, N_{2k}(t_k) = 0, \cdots, N_{2n-1}(t_n) = 1, N_{2n}(t_n) = 0$; and into first derivative of basis functions, we get $N'_1(t_1) = 0, N'_1(t_2) = 1, \cdots, N'_{2k-1}(t_k) = 0, N'_{2k}(t_k) = 1, \cdots, N'_{2n-1}(t_n) = 0, N'_{2n}(t_n) = 1$. That means the matrices $\mathbf{B}$ and $\mathbf{C}$ in MSE equation (2.15) are $n \times 2n$ dimensional and the elements are

$$\mathbf{B} = \{B\}_{ij} = \begin{cases} 1, & j = 2i - 1 \\ 0, & \text{otherwise} \end{cases} \tag{2.16}$$

$$\mathbf{C} = \{C\}_{ij} = \begin{cases} 1, & j = 2i \\ 0, & \text{otherwise} \end{cases} \tag{2.17}$$

where $i = 1, \cdots, n$. The k-th $\Omega_{\lambda_k}^{(k)}$ is a $2n \times 2n$ matrix and its details is in appendix. Then the penalty term is

$$\boldsymbol{\Omega}_\lambda = \sum_{k=1}^{n-1} \Omega_{\lambda_k}^{(k)}, \tag{2.18}$$

which is a bandwidth four matrix.

The solution to (2.15) is easily seen to be

$$\hat{\theta} = (\mathbf{B}^\top \mathbf{B} + \gamma \mathbf{C}^\top \mathbf{C} + n\Omega_\lambda)^{-1} (\mathbf{B}^\top \mathbf{y} + \gamma \mathbf{C}^\top \mathbf{v}) \tag{2.19}$$

a generalized ridge regression. Then the fitted smoothing spline is given by

$$\hat{f}(t) = \sum_{i=1}^{2n} N_i(t) \hat{\theta}_i \tag{2.20}$$

A smoothing spline with parameters $\lambda(t)$ and γ is an example of a linear smoother Trevor Hastie (2009). This is because the estimated parameters in equation (2.19) are a linear combination of y_i and v_i. Denote by $\hat{\mathbf{f}}$ and $\hat{\mathbf{f}}'$ the $2n$ vector of fitted values $\hat{f}(t_i)$ and $\hat{f}'(t_i)$ at the training points t_i. Then

$$\begin{aligned} \hat{\mathbf{f}} &= \mathbf{B}(\mathbf{B}^\top \mathbf{B} + \gamma \mathbf{C}^\top \mathbf{C} + n\Omega_\lambda)^{-1} (\mathbf{B}^\top \mathbf{y} + \gamma \mathbf{C}^\top \mathbf{v}) \\ &\triangleq \mathbf{S}_{\lambda,\gamma} \mathbf{y} + \gamma \mathbf{T}_{\lambda,\gamma} \mathbf{v} \end{aligned} \tag{2.21}$$

$$\begin{aligned} \hat{\mathbf{f}}' &= \mathbf{C}(\mathbf{B}^\top \mathbf{B} + \gamma \mathbf{C}^\top \mathbf{C} + n\Omega_\lambda)^{-1} (\mathbf{B}^\top \mathbf{y} + \gamma \mathbf{C}^\top \mathbf{v}) \\ &\triangleq \mathbf{U}_{\lambda,\gamma} \mathbf{y} + \gamma \mathbf{V}_{\lambda,\gamma} \mathbf{v} \end{aligned} \tag{2.22}$$

The fitted $\hat{\mathbf{f}}$ and $\hat{\mathbf{f}}'$ are linear in $\mathbf{y}$ and $\mathbf{v}$, and the finite linear operators $\mathbf{S}_{\lambda,\gamma}, \mathbf{T}_{\lambda,\gamma}, \mathbf{U}_{\lambda,\gamma}$ and $\mathbf{V}_{\lambda,\gamma}$ are known as the smoother matrices. One consequence of this linearity is that the recipe for producing $\hat{\mathbf{f}}$ and $\hat{\mathbf{f}}'$ from $\mathbf{y}$ and $\mathbf{v}$, do not depend on $\mathbf{y}$ and $\mathbf{v}$ themselves; $\mathbf{S}_{\lambda,\gamma}, \mathbf{T}_{\lambda,\gamma}, \mathbf{U}_{\lambda,\gamma}$ and $\mathbf{V}_{\lambda,\gamma}$ depend only on $t_i, \lambda(t)$ and γ.

Suppose in a traditional least squares fitting, $\mathbf{B}_\xi$ is $N \times M$ matrix of M cubic-spline basis functions evaluated at the N training points x_i, with knot sequence ξ and $M \ll N$. Then the vector of fitted spline values is given by

$$\hat{\mathbf{f}} = \mathbf{B}_\xi(\mathbf{B}_\xi^\top \mathbf{B}_\xi)^{-1}\mathbf{B}_\xi \mathbf{y} = \mathbf{H}_\xi \mathbf{y} \tag{2.23}$$

Here the linear operator $\mathbf{H}_\xi$ is a symmetric, positive semidefinite matrices, and $\mathbf{H}_\xi \mathbf{H}_\xi = \mathbf{H}_\xi$ (idempotent) Trevor Hastie (2009). In our case, it is easily seen that $\mathbf{S}_{\lambda,\gamma}, \mathbf{T}_{\lambda,\gamma}, \mathbf{U}_{\lambda,\gamma}$ and $\mathbf{V}_{\lambda,\gamma}$ are symmetric, positive semidefinite matrices as well. Additionally, by Cholesky decomposition

$$(\mathbf{B}^\top \mathbf{B} + \gamma \mathbf{C}^\top \mathbf{C} + n\Omega_\lambda)^{-1} = \mathbf{R}\mathbf{R}^\top, \tag{2.24}$$

it is easily to prove that $\mathbf{T}_{\lambda,\gamma} = \mathbf{B}\mathbf{R}\mathbf{R}^\top \mathbf{C}^\top$ and $\mathbf{U}_{\lambda,\gamma} = \mathbf{C}\mathbf{R}\mathbf{R}^\top \mathbf{B}^\top$, then we will have $\mathbf{T}_{\lambda,\gamma} = \mathbf{U}_{\lambda,\gamma}^\top$. When $\lambda = \gamma = 0$, the matrix $\mathbf{S}_{\lambda_0,\gamma_0} = \mathbf{B}(\mathbf{B}^\top \mathbf{B})^{-1}\mathbf{B}^\top$ is idempotent.

Additionally, by playing with $\lambda(t)$ and γ, tractor spline has the following property:

- if $\lambda(t)$ is a piecewise constant and $\gamma \neq 0$, then f and f' are continuous, f'' is piecewise linear but not continuous at knots;
- if $\lambda(t)$ is a piecewise constant and $\gamma = 0$, the same as above;
- if $\lambda(t) = \lambda$ is a constant and $\gamma \neq 0$, the same as above;
- if $\lambda(t) = \lambda$ is a constant and $\gamma = 0$, then f, f' are continuous, f'' is piecewise linear and continuous at knots.

2.3.4 Adjusted Penalty Term and Parameter Function

To get the reconstructed trajectory in a multi-dimensional space, one can use tractor spline to find the trajectory in each dimensions simultaneously can combine them together at the end. Additionally, sometimes the data is not recorded in equal space. Due to the property of Hermite spline, the combination of multi-dimensional reconstructions and non-equal space data will bring some issues. Imagining this situation that a vehicle is moving along the x-axis, its x position changes consequently, but its y position might stay the same. By fitting $\mathbf{x}$ and $\mathbf{u}$ on x-axis, the tractor spline $f_x(t)$ will give us a best fit which returns smallest errors to the objective function. While with the same parameter $\lambda(t)$ and γ, $f_y(t)$ will return a cubic curve. However, it should give us a straight line as we expected. Moreover, in some circumstances, the time mark increases but $\mathbf{f}$ and $\mathbf{f}'$ keep the same, or changes slightly. Facing this situation, the Hermite spline will return a wiggle in one dimension and a curve in two dimensions. To get a reliable reconstruction, we introduce an adjusted term $\frac{(\Delta t_i)^\alpha}{(\Delta d_i)^\beta}$, where $\alpha \geq 0$ and $\beta \geq 0$, to the penalty function $\lambda(t)$, which means that the tractor spline should be penalized by its real difference of Δd_i and Δt_i between two points p_i and p_{i+1}. With this term, when $\mathbf{u}$ goes down or equals to 0, it will make sure that the penalty function will be large enough and return a straight line rather than a curve in each dimension of x and y. Because of the unit of the penalty term is m^2/t^3, to keep the same scale in the space, α and β in the adjusted penalty term are chosen as 3 and 2. Then the final form of the penalty function is

$$\lambda(t) = \frac{(\Delta t_i)^3}{(\Delta d_i)^2}\lambda, \tag{2.25}$$

where $t_i \leq t < t_{i+1}$. Eventually in objective function there is one parameter λ controlling the curvature of tractor spline in different status, and another one parameter γ controlling the residuals of velocity.

2.4 Parameter Selection and Cross Validation

The problem of choosing the smoothing parameter is ubiquitous in curve estimation. And there are two different philosophical approaches to this question. The first one is to regard the free choice of smoothing parameter as an advantageous feature of the procedure. The other one is to find the parameter automatically by the data Green and Silverman (1993). We more prefer the latter one, use data to train our model and find the best parameters. The most well known method is cross-validation.

Assuming that the random errors has zero mean, the true regression curve $f(t)$ has the property that, if an observation y is taken at a point t, the value $f(t)$ is the best predictor of y in terms of returning a small value of $(y - f(t))^2$.

Now we focus on an observation y_i at point t_i as being a new observation by omitting it from the set of data, which are used to estimate $\hat{f}$. Denote by $\hat{f}^{(-i)}(t,\lambda)$ the estimated function from the remaining data, where λ is the smoothing parameter. Then $\hat{f}^{(-i)}(t,\lambda)$ minimizes

$$\frac{1}{n}\sum_{j\neq i}(y_j - f(t_j))^2 + \lambda\int f''^2 dt \tag{2.26}$$

and λ can be quantified by cross-validation score function

$$\mathrm{CV}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\{y_i - \hat{f}^{(-i)}(t_i, \lambda)\}^2. \tag{2.27}$$

The basis idea of cross-validation is to choose the value of λ that minimizes $\mathrm{CV}(\lambda)$.

An efficient way to calculate cross validation score is given by Green and Silverman (1993). Through the equation (2.23), we know that the value of the smoothing spline $\hat{f}$ depend linearly on the data y_i. Define the matrix $A(\lambda)$, which is a map vector of observed values y_i to predicted values $\hat{f}(t_i)$. Then we have

$$\hat{\mathbf{f}} = A(\lambda)\mathbf{y} \tag{2.28}$$

and the following lemma.

Lemma 1. *The cross validation score satisfies*

$$CV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{f}(t_i)}{1 - A_{ii}(\lambda)}\right)^2 \tag{2.29}$$

where $\hat{f}$ is the spline smoother calculated from the full data set $\{(t_i, y_i)\}$ with smoothing paramter λ.

For a tractor spline and its MSE function, there are two parameters need to be estimated λ and γ. Then the objective function (2.26) becomes

$$\frac{1}{n}\sum_{j\neq i}(y_j - f(t_j))^2 + \frac{\gamma}{n}\sum_{j\neq i}(v_j - f'(t_j))^2 + \int \lambda(t) f''^2 dt, \tag{2.30}$$

and the cross-validation score function is

$$\mathrm{CV}(\lambda, \gamma) = \frac{1}{n}\sum_{i=1}^{n}\{y_i - \hat{f}^{(-i)}(t_i, \lambda, \gamma)\}^2. \tag{2.31}$$

For a tractor spline, the parameter $\hat{\theta} = (B^\top B + \gamma C^\top C + n\Omega_\lambda)^{-1}(B^\top\mathbf{y} + \gamma C^\top \mathbf{v})$ gives us

$$\begin{aligned}\hat{\mathbf{f}} &= B\hat{\theta} = B(B^\top B + \gamma C^\top C + n\Omega_\lambda)^{-1}B^\top\mathbf{y} + B(B^\top B + \gamma C^\top C + n\Omega_\lambda)^{-1}C^\top\mathbf{v} \\ &= S\mathbf{y} + \gamma T\mathbf{v},\end{aligned} \tag{2.32}$$

$$\begin{aligned}\hat{\mathbf{f}}' &= C\hat{\theta} = C(B^\top B + \gamma C^\top C + n\Omega_\lambda)^{-1}B^\top\mathbf{y} + C(B^\top B + \gamma C^\top C + n\Omega_\lambda)^{-1}C^\top\mathbf{v} \\ &= U\mathbf{y} + \gamma V\mathbf{v}.\end{aligned} \tag{2.33}$$

From lemma 1, we can prove the following theorem.

Theorem 3. *The cross validation score of a tractor spline satisfies*

$$CV(\lambda, \gamma) = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{\hat{f}(t_i) - y_i + \gamma \frac{T_{ii}}{1-\gamma V_{ii}} (\hat{f}'(t_i) - v_i)}{1 - S_{ii} - \gamma \frac{T_{ii}}{1-\gamma V_{ii}} U_{ii}} \right)^2 \tag{2.34}$$

where $\hat{f}$ is the tractor spline smoother calculated from the full data set $\{(t_i, y_i, v_i)\}$ with smoothing parameter λ and γ.

The proof of Theorem 3 follows immediately from a lemma, and gives an expression for the deleted residuals $y_i - \hat{f}^{(-i)}(t_i)$ and $v_i - \hat{f}'^{(-i)}(t_i)$ in terms of $y_i - \hat{f}(t_i)$ and $v_i - \hat{f}'(t_i)$ respectively.

Lemma 2. *For fixed λ, γ and i, denote $\mathbf{f}^{(-i)}$ by the vector with components $f_j^{(-i)} = \hat{f}^{(-i)}(t_j, \lambda, \gamma)$, $\mathbf{f}'^{(-i)}$ by the vector with components $f_j'^{(-i)} = \hat{f}'^{(-i)}(t_j, \lambda, \gamma)$, and define vectors $\mathbf{y}^*$ and $\mathbf{v}^*$ by*

$$\begin{cases} y_j^* = y_j & j \neq i \\ y_i^* = \hat{f}^{(-i)}(t_i) & \textit{otherwise} \end{cases}, \tag{2.35}$$

$$\begin{cases} v_j^* = v_j & j \neq i \\ v_i^* = \hat{f}'^{(-i)}(t_i) & \textit{otherwise} \end{cases}. \tag{2.36}$$

Then

$$\hat{\mathbf{f}}^{(-i)} = S\mathbf{y}^* + \gamma T \mathbf{v}^* \tag{2.37}$$

$$\hat{\mathbf{f}}'^{(-i)} = U\mathbf{y}^* + \gamma V \mathbf{v}^* \tag{2.38}$$

2.5 Simulation

2.5.1 Numerical Examples

In this section, we examine the visual quality of the proposed method with four functions: Blocks, Bumps, HeaviSine and Doppler, which have been used in Donoho and Johnstone (1994), Donoho and Johnstone (1995) and Abramovich *et al.* (1998) because of their caricature features in imaging, spectroscopy and other scientific signal processing. However it is unfair for tractor spline fitting "jump" position in Blocks and Bumps function, because it fits position and velocity simultaneously and these points imply infinite first derivative in original functions, which are impossible for vehicles or individuals. In terms of this issue, we treat these functions as velocity, and use noise free points to generate accurate position data, then add noises back to them.

For calculating consideration, we use $n = 1024$ Nason (2010). Because all noises are randomly generated, for convenience of reinitialization and repetition of comparing, we set random seed as 2016. The noises are independent Gaussian distribution $\epsilon \sim N(0,1)$ and signal-to-noise ratio (SNR) is 7. These data are treated as velocity (first derivative). By setting initial position $y_0 = 0$, acceleration $a_0 = 0$ and using the following formula to calculate position

$$y_{i+1} = y_i + (v_i + v_{i+1})\frac{t_{i+1} - ti}{2}, \tag{2.39}$$

we can easily generate position data. Then we add some noises, which are independent Gaussian distribution $\epsilon \sim N(0,1)$ and SNR is 7. For wavelet reconstruction, we use the threshold policy of "*sure*" and "*BayesThresh*" with levels $j = 4, \cdots, 9$. Penalized B-spline is added in comparison. For tractor spline we have two parameters λ and γ. To evaluate the performance of the velocity term in objective function (2.4) and the adjusted penalty term in (2.25), the parameter γ is set as 0 in one reconstruction of tractor spline, whose objective function and solution become

$$J[f]_{\gamma=0} = \frac{1}{n}\sum_{i=1}^{n}(f(t_i) - y_i)^2 + \sum_{i=1}^{n-1}\lambda_i \int_{t_i}^{t_{i+1}} f''(t)^2 dt, \tag{2.40}$$

and

$$\hat{\theta}_{\gamma=0} = (\mathbf{B}^\top \mathbf{B} + n\Omega_\lambda)^{-1}\mathbf{B}^\top \mathbf{y} \tag{2.41}$$

and the adjusted penalty term in (2.25) was removed from another reconstruction, noted as "tractor spline without APT". Figure (2.2) to (2.5) display the original (velocity), generated position, wavelet with two different threshold methods, P-spline and three kinds of tractor spline fitted functions. The parameters λ and γ of a tractor spline are automatically selected from formula (2.34) by *optim*() function in *R*.

By comparing, we can see that all these methods can rebuild up the skeleton of generated trajectory. Wavelets(sure) method have more wiggles in interior interval than Wavelet(BayesThresh), and the latter one becomes fluctuation near boundary knots. P-spline gives much smoother fitting than wavelets, but the drawback is losing more specific details. Tractor spline without velocity might give us less fitting, as can be seen from Blocks and Bumps where there should be a straight line. Tractor spline without adjusted penalty term could also get over fitting when the direction changes more frequently than normal, although it catches specific feature in HeaviSine. The proposed tractor spline performs much better than other methods and returns the near-true trajectory reconstruction.

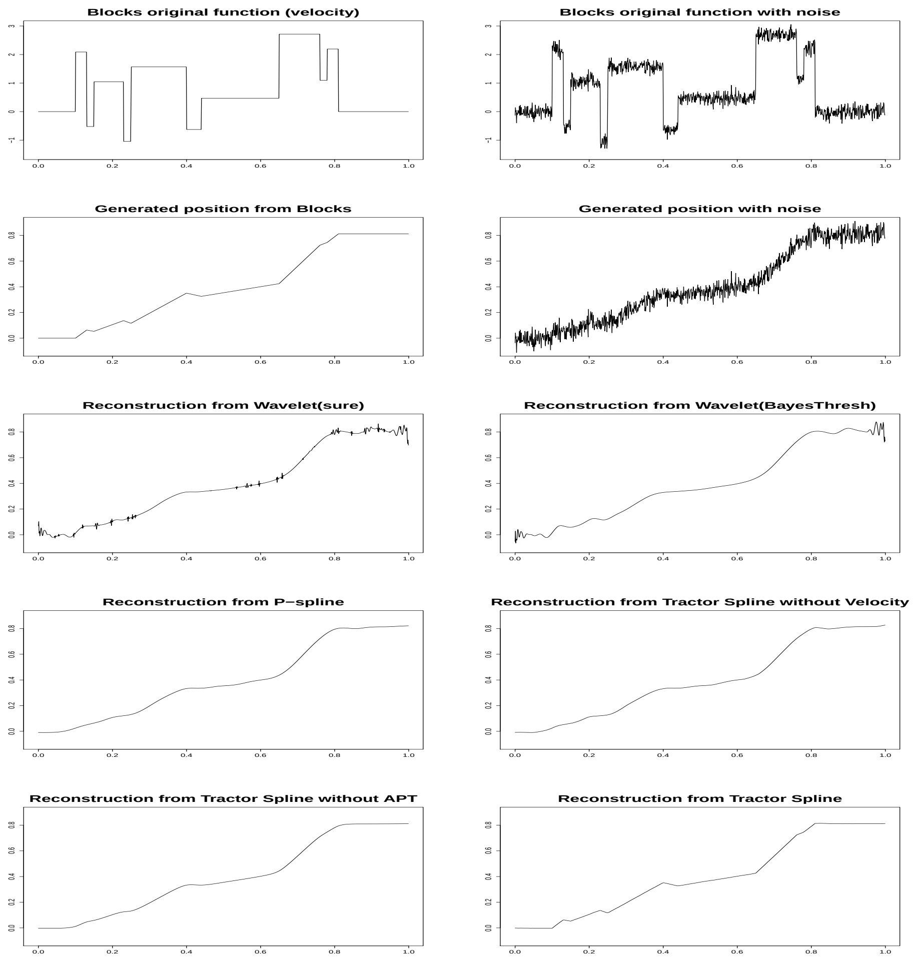


Figure 2.2: Numerical example: *Blocks*. (a) The true velocity function. (b) Velocity with Gaussian noise at SNR=7. (c) Generated position function. (d) Position with Gaussian noise at SNR=7. (e) Reconstruction from Wavelet with sure threshold. (f) Reconstruction from Wavelet with BayesThresh approach. (g) Reconstruction by P-spline. (h) Reconstruction by tractor spline setting $\gamma = 0$. (i) Reconstruction by tractor spline with normal penalty term. (j) Reconstruction by proposed tractor spline.

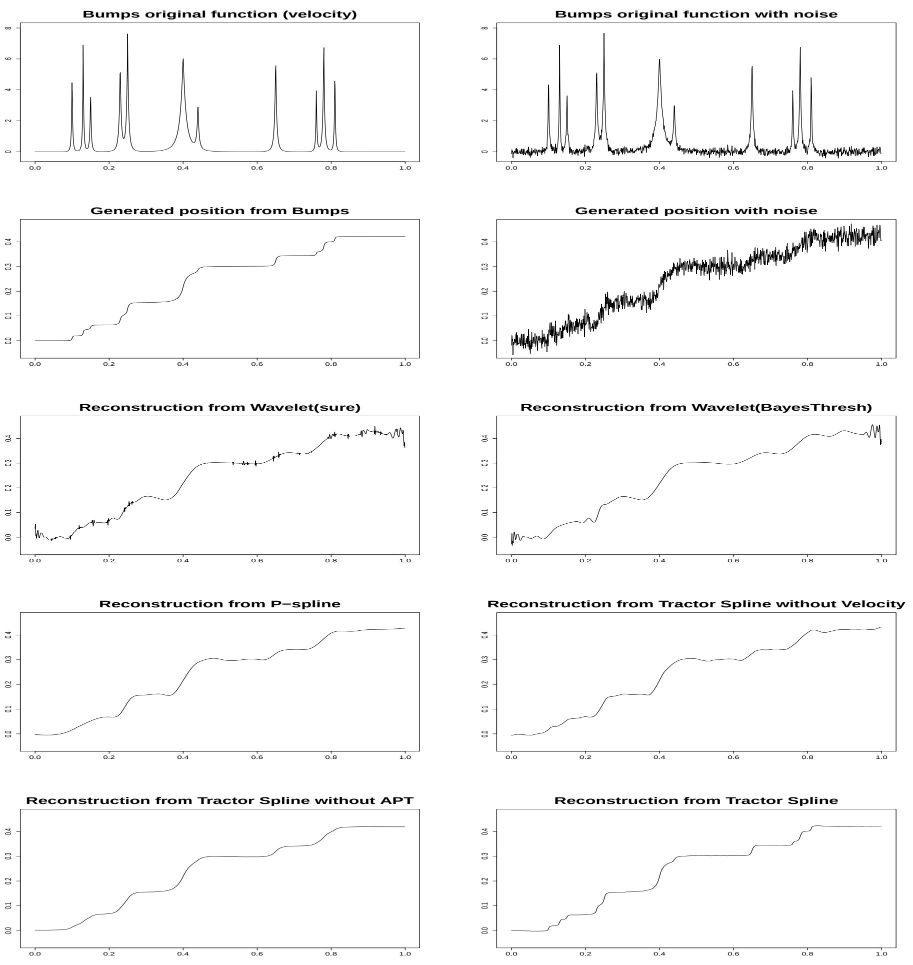


Figure 2.3: Numerical example: *Bumps*. (a) The true velocity function. (b) Velocity with Gaussian noise at SNR=7. (c) Generated position function. (d) Position with Gaussian noise at SNR=7. (e) Reconstruction from Wavelet with sure threshold. (f) Reconstruction from Wavelet with BayesThresh approach. (g) Reconstruction by P-spline. (h) Reconstruction by tractor spline setting $\gamma = 0$. (i) Reconstruction by tractor spline with normal penalty term. (j) Reconstruction by proposed tractor spline.

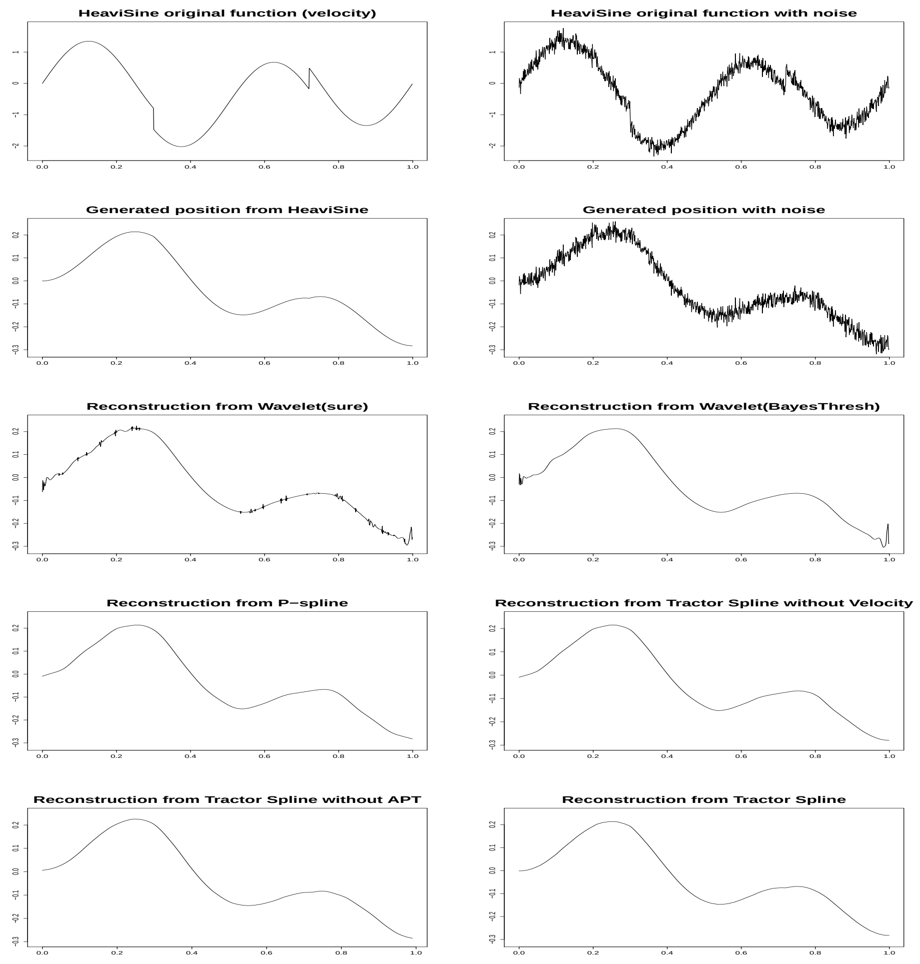


Figure 2.4: Numerical example: *HeaviSine*. (a) The true velocity function. (b) Velocity with Gaussian noise at SNR=7. (c) Generated position function. (d) Position with Gaussian noise at SNR=7. (e) Reconstruction from Wavelet with sure threshold. (f) Reconstruction from Wavelet with BayesThresh approach. (g) Reconstruction by P-spline. (h) Reconstruction by tractor spline setting $\gamma = 0$. (i) Reconstruction by tractor spline with normal penalty term. (j) Reconstruction by proposed tractor spline.

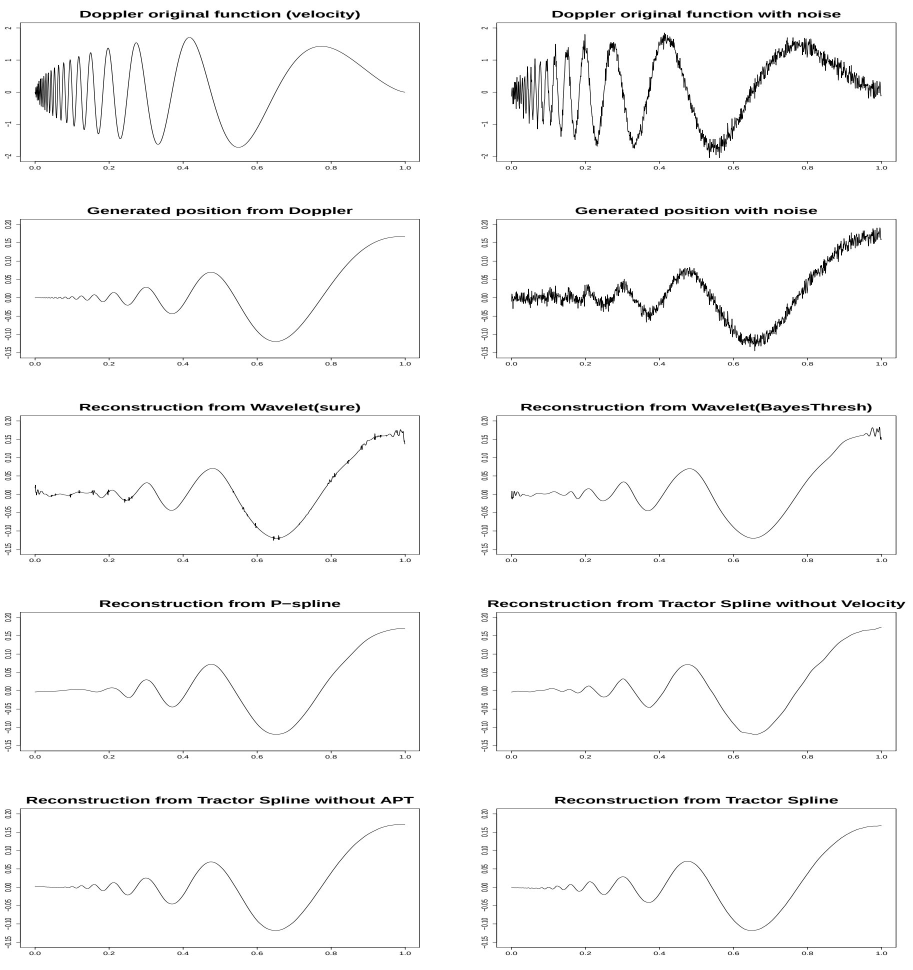


Figure 2.5: Numerical example: *Doppler*. (a) The true velocity function. (b) Velocity with Gaussian noise at SNR=7. (c) Generated position function. (d) Position with Gaussian noise at SNR=7. (e) Reconstruction from Wavelet with sure threshold. (f) Reconstruction from Wavelet with BayesThresh approach. (g) Reconstruction by P-spline. (h) Reconstruction by tractor spline setting $\gamma = 0$. (i) Reconstruction by tractor spline with normal penalty term. (j) Reconstruction by proposed tractor spline.

Figure 2.6 shows the estimated penalty function

$$\lambda(t) = \frac{(\Delta t)^3}{(\Delta d)^2}\lambda. \tag{2.42}$$

The left column illustrates the value of penalty function on different intervals and the right column is the projection on position. Bigger black dots present larger penalty values. It can be seen that $\lambda(t)$ adapts to the smoothness pattern of position and will be large where a long time gap may occur. The details of how this penalty function works will be explained in next subsection.

Figure 2.7 demonstrates the estimated velocity functions. Bt taking the first derivative of fitted tractor spine, it is easily to get the original four velocity functions. The fitting of velocity is not as smooth as that in position, because we only care about the smoothness of position rather than velocity in our cross-validation formula (3). However, velocity information dose help us reconstructing trajectory.

2.5.2 Evaluation

To examine the performance of tractor spline, we conducted a evaluation by comparing the mean square errors and true mean square errors, which are respectively calculated in

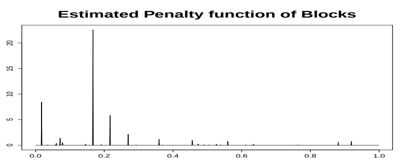
$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}_{\lambda,\gamma}(t_i))^2, \tag{2.43}$$

$$\text{TMSE} = \frac{1}{n}\sum_{i=1}^{n}(f(t_i) - \hat{f}_{\lambda,\gamma}(t_i))^2. \tag{2.44}$$

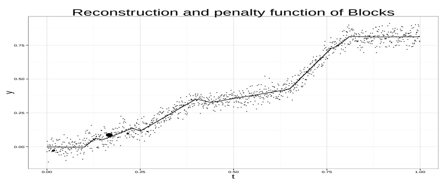
The results are shown in table 2.1 and 2.2. All of these methods have good performances in fitting noisy data. The differences of mean square error between these methods are not significant, as can be seen from table 2.1. The proposed method is not the best among these simulations according to MSE. However, from table 2.2, tractor spline returns the smallest true mean square errors. The difference is significant, that means the reconstruction from tractor spline is closer to the true trajectory.

2.6 Conclusion and Discussion

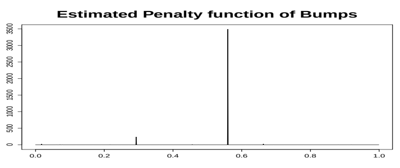
In this paper, we proposed a tractor spline model with first derivative and adaptive penalty terms to reconstruct trajectory. This method performs better when we know **z** and **w** information than other methods. Additionally, the reconstruction of a tractor



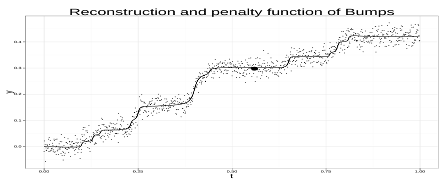
(a)



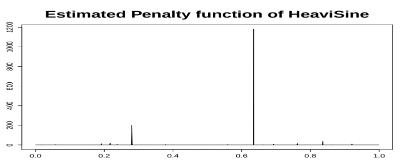
(b)



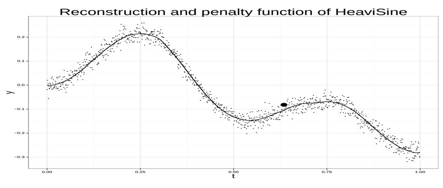
(c)



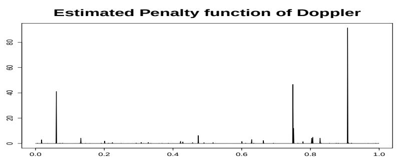
(d)



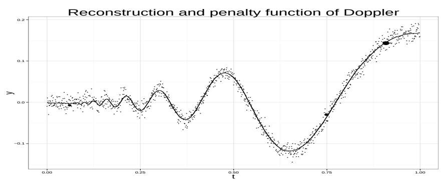
(e)



(f)



(g)



(h)

Figure 2.6: Estimated penalty functions. Left side shows how the value of $\lambda(t)$ changes on the interval. Right side projects $\lambda(t)$ into reconstructions. The bigger the blacks dots present, the larger the penalty values are.

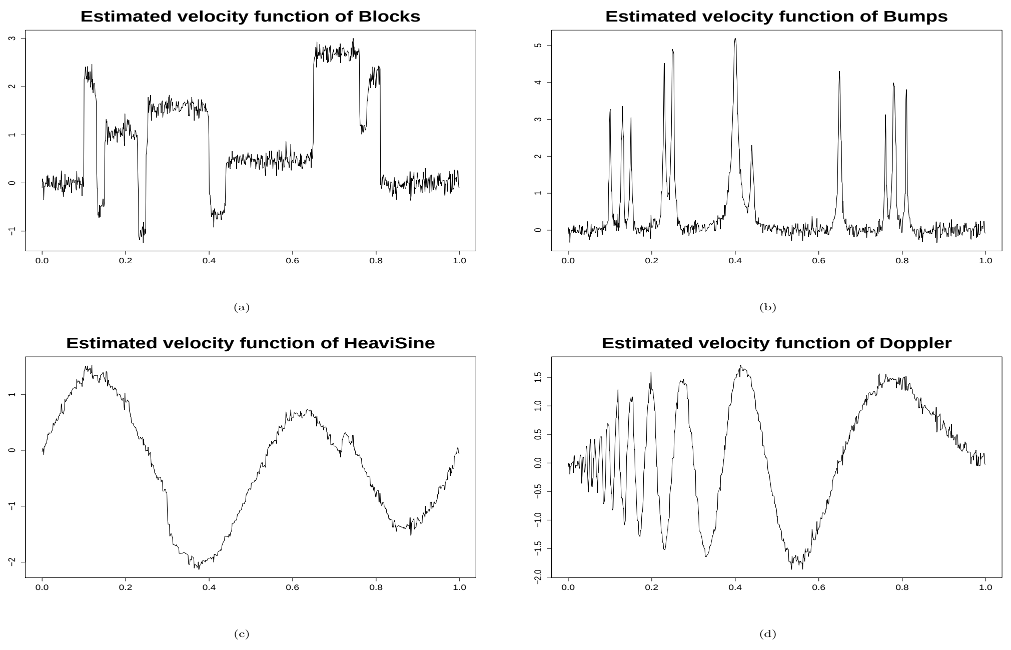


Figure 2.7: Estimated velocity functions by taking the first derivative of tractor spline. (a) Fitted *Blocks*. (b) Fitted *Bumps*. (c) Fitted *HeaviSine*. (d) Fitted *Doppler*.

Table 2.1: MSE. Mean square errors of different methods. The star sign (*) marks the smallest error among these methods under the same level. The difference is not significant.

MSE (10^{-4})	SNR	TS	TS$_{\gamma=0}$	TS$_{APT=0}$	P-spline	Wavelet(sure)	Wavelet(Bayes)
Blocks	7	16.53	15.99	16.69	16.14	*15.39	16.68
Blocks	3	89.79	*87.64	89.94	88.27	98.35	90.24
Bumps	7	4.40	4.19	4.55	4.33	*4.18	4.59
Bumps	3	23.93	*23.19	24.10	23.55	26.23	23.74
HeaviSine	7	4.16	4.01	4.16	4.02	*3.79	4.19
HeaviSine	3	22.63	*22.19	22.65	22.02	23.53	22.07
Doppler	7	1.15	*1.07	1.10	1.15	*1.07	1.13
Doppler	3	6.27	*5.94	6.28	6.05	6.85	6.29

Table 2.2: TMSE. True mean square errors of different methods. The star sign (*) marks the smallest error among these methods under the same level. The proposed tractor spline returns the smallest TMSE among all the methods under the same level except for *Doppler* with SNR=7. The differences are significant.

TMSE (10^{-6})	SNR	TS	TS$_{\gamma=0}$	TS$_{APT=0}$	P-spline	Wavelet(sure)	Wavelet(Bayes)
Blocks	7	*1.75	54.25	28.68	54.76	201.02	182.12
Blocks	3	*16.44	152.5	30.76	171.59	1138.08	712.36
Bumps	7	*1.64	23.44	21.10	24.21	71.71	69.26
Bumps	3	*8.51	77.78	37.12	77.52	330.77	238.79
HeaviSine	7	*1.53	7.80	1.56	9.54	55.37	44.88
HeaviSine	3	*8.21	33.56	8.49	34.26	240.72	110.49
Doppler	7	1.51	6.67	*1.08	8.26	14.87	12.01
Doppler	3	*8.10	22.14	8.25	19.95	81.48	50.33

spline contains $4 \times (n-1)$ parameters if we have n knots. By adding $2 \times (n-2)$ constrains, the original function and its first derivative are continuous on each interior knots, the degrees of freedom will be $4 \times (n-1) - 2 \times (n-2) = 2n$. Because there are n location and n velocity data, thus we don't need to specify more parameters or add more constrains on the model. Although the mean square error of tractor spline is not the smallest comparing with other methods, the true mean square error is the smallest most of the time. It means that the reconstruction is closer to the truth. In cross validation, we only focus on the errors of f ignoring that in f'. So the reconstruction of f' is not as smooth as that in f, which does not affect trajectory reconstruction. A drawback of tractor spline is the cost in finding local minimal parameters, where the cross validation algorithm returns a smaller score. So we optimized our code to make it runs as faster as it can.

Chapter 3

Gaussian Process Regression

3.1 Introduction

From Wikipedia, the free encyclopedia.

In probability theory and statistics, a Gaussian process is a particular kind of statistical model where observations occur in a continuous domain, e.g. time or space. In a Gaussian process, every point in some continuous input space is associated with a normally distributed random variable. Moreover, every finite collection of those random variables has a multivariate normal distribution, i.e. every finite linear combination of them is normally distributed. The distribution of a Gaussian process is the joint distribution of all those (infinitely many) random variables, and as such, it is a distribution over functions with a continuous domain, e.g. time or space.

Viewed as a machine learning algorithm, a Gaussian process uses lazy learning and a measure of the similarity between points (the kernel function) to predict the value for an unseen point from training data. The prediction is not just an estimate for that point, but also has uncertainty information it is a one-dimensional Gaussian distribution (which is the marginal distribution at that point).

For some kernel functions, matrix algebra can be used to calculate the predictions using the technique of Kriging. When a parameterized kernel is used, optimization software is typically used to fit a Gaussian process model.

The concept of Gaussian processes is named after Carl Friedrich Gauss because it is based on the notion of the Gaussian distribution (normal distribution). Gaussian processes can be seen as an infinite-dimensional generalization of multivariate normal distributions.

Gaussian processes are useful in statistical modeling, benefiting from properties inherited from the normal. For example, if a random process is modeled as a Gaussian process, the distributions of various derived quantities can be obtained explicitly. Such quantities include the average value of the process over a range of times and the error in estimating the average using sample values at a small set of times.

3.1.1 Spline

In interpolation and curve fitting, piecewise linear approximation may not have the practical significance of cubic spline, or even higher order, approximation. These "broken lines" are neither very smooth nor very efficient approximation. Researchers can go to piecewise polynomial approximation with higher order pieces De Boor *et al.* (1978), which is called spline method. A spline is a numeric function that is piecewise-defined by polynomial functions, and which possesses a high degree of smoothness at the places where the polynomial pieces connect (known as knots) Judd (1998)Chen (2009). Suppose we are given observed data $t_1, t_2, \cdots, t_n$ on interval $[0, 1]$, satisfying $0 \leq t_1 < t_2 < \cdots < t_n \leq 1$. A piecewise polynomial function $f(t)$ can be obtained by dividing the interval into contiguous intervals $(t_1, t_2), \cdots, (t_{n-1}, t_n)$ and represented by a separate polynomial in each interval. For any continuous $f \in \mathbb{C}^{(m)}[0, 1]$, it can be represented in a linear combination of basis functions $h_m(t)$, just as every vector in a vector space can be represented as a linear combination of basis vectors. So we have

$$f(t) = \sum_{m=1}^{M} \beta_m h_m(t), \tag{3.1}$$

where β_m are coefficients Ellis *et al.* (2009).

Suppose we were attempt to fit a model of $f(t)$ by least squares without any restrictions, the best fitting $\hat{f}(t)$ would go through every given data to reduce sum of squares to zero. Most of the time, the results are unsatisfactory as explanations of the given data. The roughness penalty approach is introduced to quantify the notion of a rapidly fluctuating curve and then to pose the estimation problem in a way that makes explicit the necessary compromise between varying rapidly fluctuation and slowly trend in curve estimation Green and Silverman (1993). By adding a penalty term $\int_0^1 (f^{(m)}(t))^2 dt$, the curve estimate $\hat{f}(t)$ exists and is unique over all spline functions $f(t)$ with $m-1$ continuous derivatives fitting observed data in the space $\mathbb{C}^{(m)}[0, 1]$, and it can be found by minimizing the following penalized mean residual sum squares

$$\mathrm{MSE}(f, \lambda) = \frac{1}{n} \sum_{i=1}^{n} (f(t_i) - y_i)^2 + \lambda \int_0^1 (f^{(m)}(t))^2 dt, \tag{3.2}$$

where λ is a fixed smoothing parameter, (t_i, y_i), $i = 1, \cdots, n$ are observed data and $0 \le t_1 < t_2 < \cdots < t_n \le 1$. In equation (3.2), the smoothing parameter λ controls the trade-off between over-fitting and bias. Smoothing spline provides a powerful tool to estimate nonparametric functions Hastie and Tibshirani (1990).

3.1.2 Gaussian Process Regression

A Gaussian Process is a collection of random variables, any finite number of which have a joint Gaussian distribution Rasmussen and Williams (2006). It is fully defined by its mean $m(t)$ and covariance $K(s,t)$ functions as

$$m(t) = \mathbb{E}[f(t)] \tag{3.3}$$

$$K(s,t) = \mathbb{E}[(f(s) - m(s))(f(t) - m(t))], \tag{3.4}$$

where s and t are two variables, and a function f distributed as such is denoted in form of

$$f \sim GP(m(t), K(s,t)). \tag{3.5}$$

Usually the mean function is assumed to be zero everywhere.

Given a set of input variables $\mathbf{T}$ for function $f(t)$ and the output $\mathbf{y} = f(\mathbf{T}) + \varepsilon$ with independent identically distributed Gaussian noise ε with variance σ_n^2, we can use the above definition to predict the value of the function $f_* = f(t_*)$ at a particular input t_*. As the noisy observations becoming

$$\mathrm{cov}(y_p, y_q) = K(t_p, t_q) + \sigma_n^2 \delta_{pq} \tag{3.6}$$

where δ_{pq} is a Kronecker delta which is one iff $p = q$ and zero otherwise, the joint distribution of the observed outputs $\mathbf{y}$ and the estimated output f_* according to prior is

$$\begin{bmatrix} \mathbf{y} \\ f_* \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(\mathbf{T},\mathbf{T}) + \sigma_n^2 I & K(\mathbf{T}, t_*) \\ K(t_*, \mathbf{T}) & K(t_*, t_*) \end{bmatrix}\right). \tag{3.7}$$

The posterior distribution over the predicted value is obtained by conditioning on the observed data

$$f_* \mid \mathbf{y}, \mathbf{T}, t_* \sim N(\bar{f}_*, \mathrm{cov}(f_*)) \tag{3.8}$$

where

$$\bar{f}_* = \mathbb{E}[f_* \mid \mathbf{y}, \mathbf{T}, t_*] = K(t_*, \mathbf{T})[K(\mathbf{T},\mathbf{T}) + \sigma_n^2 I]^{-1}\mathbf{y}, \tag{3.9}$$

$$\mathrm{cov}(f_*) = K(t_*, t_*) - K(t_*, \mathbf{T})[K(\mathbf{T},\mathbf{T}) + \sigma_n^2 I]^{-1} K(\mathbf{T}, t_*). \tag{3.10}$$

3.1.3 The Smoothing Spline as Bayes Estimates

It is possible to interpret the smoothing spline regression estimator as a Bayesian estimate when the mean function $r(.)$ is given an improper prior distribution. Berlinet and Thomas-Agnan (2011) Wahba (1990)

Consider the model

$$y_i = f(t_i) + \varepsilon_i, \tag{3.11}$$

where $i = 1, \ldots, n$, ε_i are i.i.d. Gaussian distributed noise with variance σ^2. Assume $f \in \mathbb{H}^{(m)}[0,1]$, where

$$\mathbb{H}^{(m)}[0,1] = \{f : f^{(\nu)}\text{absolutely continuous}, \nu = 0, \cdots, m-1, \int_0^1 (f^{(m)}(t)^2 dt < \infty\}. \tag{3.12}$$

A smoothing spline $\hat{f}_\lambda$ is the minimizer of objective function (3.2) Wang (1998). Equipped with an appropriate inner product

$$\langle f, g \rangle = \sum_{\nu=0}^{m-1} f^{(\nu)}(0) g^{(\nu)}(0) + \int_0^1 f^{(m)} g^{(m)} dt, \tag{3.13}$$

the space $\mathbb{H}^{(m)}[0,1]$ becomes a reproducing kernel Hilbert space.

Let $\phi_\nu(t) = \frac{t^{\nu-1}}{(\nu-1)!}$ where $\nu = 1, \cdots, m$ and $R_1(s,t) = \int_0^1 \frac{(s-u)_+^{m-1}}{(m-1)!} \frac{(t-u)_+^{m-1}}{(m-1)!} du$. Denote $S = \{\phi_\nu(t_i)\}_{n \times m}$ where $i = 1, \cdots, n, \nu = 1, \cdots, m$ and $Q = \{R_1(t_i, t_j)\}_{n \times n}$ where $i = 1, \cdots, n, j = 1, \cdots, n$. Kimeldorf and Wahba (1971) and Kimeldorf and Wahba (1970) proved that $\hat{f}_\lambda$ has the form

$$\hat{f}(t) = \sum_{\nu=1}^{m} d_\nu \phi_\nu(t) + \sum_{i=1}^{n} c_i R_1(t, t_i). \tag{3.14}$$

By denoting $M = Q + n\lambda I$, Gu (2013) found that the coefficients will be given by

$$\mathbf{c} = (M^{-1} - M^{-1} S (S^\top M^{-1} S)^{-1} S^\top M^{-1}) \mathbf{Y}, \tag{3.15}$$

$$\mathbf{d} = (S^\top M^{-1} S)^{-1} S^\top M^{-1} \mathbf{Y}. \tag{3.16}$$

3.2 A reproducing kernel on $\mathcal{C}^2_{p.w.}[0,1]$

The minimizer $f(x)$ of

$$\frac{1}{n} \sum_{i=1}^{n} (Y_i - f(x_i))^2 + \frac{\gamma}{n} \sum_{i=1}^{n} (v_i - f'(x_i))^2 + \lambda \int_0^1 f''^2 dx \tag{3.17}$$

in the space $\mathcal{C}^2_{p.w.}[0,1] = \{f : f, f'$ are continuous and f'' is piecewise continuous on $[0,1]\}$ is a tractor spline. Equipped with an appropriate inner product

$$(f,g) = f(0)g(0) + f'(0)g'(0) + \int_0^1 f''g''dx, \tag{3.18}$$

the space $\mathcal{C}^2_{p.w.}[0,1]$ is made a reproducing kernel Hilbert space. In fact, the representer $R_x(\cdot)$ is

$$R_x(y) = 1 + xy + \int_0^1 (x-u)_+(y-u)_+du. \tag{3.19}$$

It can be seen that $R_x(0) = 1, R'_x(0) = x$, and $R''_x(y) = (x-y)_+$.

The two terms of the reproducing kernel $R(x,y) = R_x(y) = R_0(x,y) + R_1(x,y)$, where

$$R_0(x,y) = 1 + xy \tag{3.20}$$

$$R_1(x,y) = \int_0^1 (x-u)_+(y-u)_+du \tag{3.21}$$

are both non-negative definite themselves.

Theorem 4. *If the reproducing kernel R of a space $\mathcal{H}$ on domain X can be decomposed into $R = R_0 + R_1$, where R_0 and R_1 are both non-negative definite, $R_0(x,\cdot), R_1(x,\cdot) \in \mathcal{H}$, $\forall x \in X$, and $(R_0(x,), R_1(y,)) = 0$, $\forall x,y \in X$, then the spaces $\mathcal{H}_0$ and $\mathcal{H}_1$ corresponding respectively to R_0 and R_1 form a tensor sum decomposition of $\mathcal{H}$. Conversely, if R_0 and R_1 are both non-negative definite and $\mathcal{H}_0 \cap \mathcal{H}_1 = \{0\}$, then $\mathcal{H} = \mathcal{H}_0 \bigoplus \mathcal{H}_1$ has a reproducing kernel $R = R_0 + R_1$.*

According to Theorem 1, R_0 can correspond the space of polynomials $\mathcal{H}_0 = \{f : f'' = 0\}$ with an inner product $(f,g)_0 = f(0)g(0) + f'(0)g'(0)$, and R_1 corresponds the orthogonal complement of $\mathcal{H}_0$

$$\mathcal{H}_1 = \{f : f(0) = 0, f'(0) = 0, \int_0^1 f''^2 dx < \infty\}$$

with inner product $(f,g)_1 = \int_0^1 f''g''dx$. Thus, $\mathcal{H}_0$ and $\mathcal{H}_1$ are two subspaces of the $\mathcal{C}^2_{p.w.}[0,1]$, and the reproducing kernel is $R_x(\cdot) = R_0(x,\cdot) + R_1(x,\cdot)$.

Define a new notation $\dot{R}(x,y) = \frac{\partial R}{\partial x}(x,y) = \frac{\partial R_0}{\partial x}(x,y) + \frac{\partial R_1}{\partial x}(x,y) = y + \int_0^x (y-u)_+du$. Obviously $\dot{R}_x(y) \in \mathcal{C}^2_{p.w.}[0,1]$. Additionally, we have $\dot{R}_x(0) = 0, \dot{R}'_x(0) = \frac{\partial \dot{R}_x}{\partial y}(0) = 1$, and $\dot{R}''_x(y) = \begin{cases} 0 & x \le y \\ 1 & x > y \end{cases}$. Then, for any $f \in \mathcal{C}^2_{p.w.}[0,1]$, it gives us

$$(\dot{R}_x, f) = \dot{R}_x(0)f(0) + \dot{R}'_x(0)f'(0) + \int_0^1 \dot{R}''_x f''du = f'(0) + \int_0^y f''du = f'(y).$$

It can be seen that the first term $\dot{R}_0 = y \in \mathcal{H}_0$, and the space spanned by the second term $\dot{R}_1 = \int_0^x (y-u)_+ du$, denoted as $\dot{\mathcal{H}}$, is not in $\mathcal{H}_1$, but $\dot{\mathcal{H}} \cap \mathcal{H}_1 \neq \emptyset$. Then we have a new space $\mathcal{H}_* = \dot{\mathcal{H}} \cup \mathcal{H}_1$. Thus the two new sub spaces in $\mathcal{C}^2_{p.w.}[0,1]$ are $\mathcal{H}_0$ and $\mathcal{H}_*$.

3.3 Computation of Polynomial Smoothing Splines

Given the sample points $x_j, j = 1, \cdots, n$ in equation (3.17) and noting that the space

$$\mathcal{A} = \{f : f = \sum_{j=1}^n \alpha_j R_1(x_j, \cdot) + \sum_{j=1}^n \beta_j \dot{R}_1(x_j, \cdot)\} \tag{3.22}$$

is a closed linear subspace of $\mathcal{H}_*$. Then $f \in \mathcal{C}^2_{p.w.}[0,1]$ can be written as

$$f(x) = d_1 + d_2 x + \sum_{j=1}^n c_j R_1(x_j, x) + \sum_{i=j}^n b_j \dot{R}_1(x_j, \cdot) + \rho(x) \tag{3.23}$$

where $\mathbf{d}, \mathbf{c}$ and $\mathbf{b}$ are coefficients, and $\rho(x) \in \mathcal{H}_* \ominus \mathcal{A}$.

The equation (3.17) can be written as

$$\begin{aligned}&\frac{1}{n}\sum_{i=1}^n \left(Y_i - d_1 - d_2 x - \sum_{j=1}^n c_j R_1(x_j, x_i) - \sum_{j=1}^n b_j \dot{R}_1(x_j, x_i) - \rho(x_i)\right)^2 \\ &\frac{\gamma}{n}\sum_{i=1}^n \left(V_i - d_2 - \sum_{j=1}^n c_j R_1'(x_j, x_i) - \sum_{j=1}^n b_j \dot{R}_1'(x_j, x_i) - \rho'(x_i)\right)^2 \\ &+\lambda \int_0^1 \left(\sum_{j=1}^n c_j R_1''(x_j, x) + \sum_{j=1}^n c_j \dot{R}_1''(x_j, x) + \rho''(x)\right)^2 dx\end{aligned}$$

By orthogonality, $\rho(x_i) = (R_1(x_i, \cdot), \rho) = 0$, $\rho'(x_i) = (\dot{R}_1(x_i, \cdot), \rho') = 0$, $i = 1, \cdots, n$. Denoting by

$$\begin{aligned} S = \{S_{ij}\}_{n\times 2} &= \begin{bmatrix} 1 & x_i \end{bmatrix}, \quad Q = \{Q_{ij}\}_{n\times n} = R_1(x_j, x_i), \quad P = \{P_{ij}\}_{n\times n} = \dot{R}_1(x_j, x_i), \\ S' = \{S'_{ij}\}_{n\times 2} &= \begin{bmatrix} 0 & 1 \end{bmatrix}, \quad Q' = \{Q'_{ij}\}_{n\times n} = R_1'(x_j, x_i), \quad P' = \{P'_{ij}\}_{n\times n} = \dot{R}_1'(x_j, x_i). \end{aligned}$$

and noting that $\int_0^1 R_1''(x_i, x) R_1''(x_j, x) dx = R_1(x_i, x_j)$, $\int_0^1 R_1''(x_i, x)\dot{R}_1''(x_j, x) dx = \int_0^v (x_i - x) dx = \dot{R}_1(x_j, x_i)$, and $\int_0^1 \dot{R}_1''(x_i, x)\dot{R}_1''(x_j, x) dx = \int_0^v 1 dx = \dot{R}_1'(x_i, x_j)$, where $v = \min(x_i, x_j)$, the above equation can be written as

$$\begin{aligned}(\mathbf{Y} - S\mathbf{d} - Q\mathbf{c} - P\mathbf{b})^\top(\mathbf{Y} - S\mathbf{d} - Q\mathbf{c} - P\mathbf{b}) + \gamma(\mathbf{V} - S'\mathbf{d} - Q'\mathbf{c} - P'\mathbf{b})^\top(\mathbf{V} - S'\mathbf{d} - Q'\mathbf{c} - P'\mathbf{b}) \\ + n\lambda(\mathbf{c}^\top Q\mathbf{c} + 2\mathbf{c}^\top P\mathbf{b} + \mathbf{b}^\top P'\mathbf{b}) + n\lambda(\rho, \rho).\end{aligned} \tag{3.24}$$

Note that ρ only appears in the third term in (3.24), which is minimised at $\rho = 0$. Hence, a polynomial smoothing spline resides in the space $\mathcal{H}_0 \oplus \mathcal{A}$ of finite dimension. Then the solution to (3.17) could be computed via minimization of the first two terms in (3.24) with respect to $\mathbf{d}$, $\mathbf{c}$ and $\mathbf{b}$.

3.4 Polynomial Smoothing Splines as Bayes Estimates

Now in the model $Y = f(x) + \epsilon$ and $V = f'(x) + \frac{\epsilon}{\gamma}$ where $\epsilon \sim N(0, \sigma^2)$, according to equation (3.23), for $f(x) \in \mathcal{C}^2_{p.w.}[0,1]$ and $x \in X$, we have

$$f(x) = (d_1 + d_2 x) + \sum_{i=1}^{n} c_i R_1(x_i, x) + \sum_{i=1}^{n} b_i \dot{R}_1(x_i, x). \tag{3.25}$$

The covariance functions for Y, V and f, f' are

$$\begin{aligned}
\mathbb{E}(f(x)f(y)) &= \tau^2 R_0(x,y) + \beta R_1(x,y) & \mathbb{E}(f(x)f'(y)) &= \tau^2 R_0'(x,y) + \beta R_1'(x,y) \\
\mathbb{E}(f'(x)f(y)) &= \tau^2 \dot{R}_0(x,y) + \beta \dot{R}_1(x,y) & \mathbb{E}(f'(x)f'(y)) &= \tau^2 \dot{R}_0'(x,y) + \beta \dot{R}_1'(x,y) \\
\mathbb{E}(y_i, y_j) &= \tau^2 R_0(x_i,x_j) + \beta R_1(x_i,x_j) & \mathbb{E}(v_i, v_j) &= \tau^2 \dot{R}_0'(x_i,x_j) + \beta \dot{R}_1'(x_i,x_j) \\
&\quad + \sigma^2 \delta_{ij} & &\quad + \frac{\sigma^2}{\gamma}\delta_{ij} \\
\mathbb{E}(v_i, y_j) &= \tau^2 \dot{R}_0(x_i,x_j) + \beta \dot{R}_1(x_i,x_j) & \mathbb{E}(y_i, v_j) &= \tau^2 R_0'(x_i,x_j) + \beta R_1'(x_i,x_j) \\
\mathbb{E}(y_i, f(x)) &= \tau^2 R_0(x_i,x) + \beta R_1(x_i,x) & \mathbb{E}(y_i, f'(x)) &= \tau^2 R_0'(x_i,x) + \beta R_1'(x_i,x) \\
\mathbb{E}(v_i, f(x)) &= \tau^2 \dot{R}_0(x_i,x) + \beta \dot{R}_1(x_i,x) & \mathbb{E}(v_i, f'(x)) &= \tau^2 \dot{R}_0'(x_i,x) + \beta \dot{R}_1'(x_i,x)
\end{aligned}$$

where $R(x,y)$ is taken from (3.19).

Observing $Y_i \sim N(f(x_i), \sigma^2)$ and $V_i \sim N(f(x_i), \frac{\sigma^2}{\gamma})$, the joint distribution of Y, V and $f(x)$ is normal with mean zero and a covariance matrix can be found. The posterior mean of $f(x)$ is

$$\begin{aligned}
\mathbb{E}(f \mid Y, V) &= \begin{bmatrix}\mathrm{cov}(Y,f) & \mathrm{cov}(f,V)\end{bmatrix} \begin{bmatrix}\mathrm{var}(Y) & \mathrm{cov}(Y,V) \\ \mathrm{cov}(V,Y) & \mathrm{var}(V)\end{bmatrix}^{-1} \begin{bmatrix}Y \\ V\end{bmatrix} \\
&= \begin{bmatrix}\tau^2\phi^\top S^\top + \beta\xi^\top & \tau^2\phi^\top S'^\top + \beta\psi^\top\end{bmatrix} \begin{bmatrix}\tau^2 SS^\top + \beta Q + \sigma^2 I & \tau^2 SS'^\top + \beta P \\ \tau^2 S'S^\top + \beta Q' & \tau^2 S'S'^\top + \beta P' + \frac{\sigma^2}{\gamma} I\end{bmatrix}^{-1} \begin{bmatrix}Y \\ V\end{bmatrix} \\
&= \begin{bmatrix}\rho\phi^\top S^\top + \xi^\top & \rho\phi^\top S'^\top + \psi^\top\end{bmatrix} \begin{bmatrix}\rho SS^\top + Q + n\lambda I & \rho SS'^\top + P \\ \rho S'S^\top + Q' & \rho S'S'^\top + P' + \frac{n\lambda}{\gamma} I\end{bmatrix}^{-1} \begin{bmatrix}Y \\ V\end{bmatrix} \\
&= (\phi\top\rho \begin{bmatrix}S \\ S'\end{bmatrix}^\top + \begin{bmatrix}\xi^\top & \psi^\top\end{bmatrix}) \left(\rho \begin{bmatrix}S \\ S'\end{bmatrix}^\top \begin{bmatrix}S \\ S'\end{bmatrix} + \begin{bmatrix}Q + n\lambda I & P \\ Q' & P' + \frac{n\lambda}{\gamma} I\end{bmatrix}\right)^{-1} \begin{bmatrix}Y \\ V\end{bmatrix} \\
&\triangleq \phi\top\rho T^\top \left(\rho T^\top T + M\right)^{-1} \begin{bmatrix}Y \\ V\end{bmatrix} + \begin{bmatrix}\xi^\top & \psi^\top\end{bmatrix} \left(\rho T^\top T + M\right)^{-1} \begin{bmatrix}Y \\ V\end{bmatrix}
\end{aligned} \tag{3.26}$$

where ϕ is 2×1 matrix with entry 1 and x, ξ is $n \times 1$ matrix with ith entry $R(x_i, x)$ and ψ is $n \times 1$ matrix with ith entry $\dot{R}(x_i, x)$, $\rho = \tau^2/\beta$ and $n\lambda = \sigma^2/\beta$.

Lemma 3. *Suppose M is symmetric and nonsingular and S is of full column rank.*

$$\begin{aligned}
&\lim_{\rho\to\infty} (\rho TT^\top + M)^{-1} = M^{-1} - M^{-1}T(T^\top M^{-1}T)^{-1}T^\top M^{-1}, \\
&\lim_{\rho\to\infty} \rho T^\top(\rho TT^\top + M)^{-1} = (T^\top M^{-1}T)^{-1}T^\top M^{-1}.
\end{aligned}$$

Setting $\rho \to \infty$ in equation (3.26) and applying Lemma 3, the posterior mean $\mathbb{E}(f(x) \mid \mathbf{Y}, \mathbf{V})$ is of the form $f = \phi^\top\mathbf{d} + \xi^\top\mathbf{c} + \psi^\top\mathbf{b}$, with the coefficients given by

$$\begin{aligned}
\mathbf{d} &= (T^\top M^{-1}T)^{-1}T^\top M^{-1} \begin{bmatrix}Y \\ V\end{bmatrix}, \\
\begin{bmatrix}\mathbf{c} \\ \mathbf{b}\end{bmatrix} &= (M^{-1} - M^{-1}T(T^\top M^{-1}T)^{-1}T^\top M^{-1}) \begin{bmatrix}Y \\ V\end{bmatrix},
\end{aligned}$$

where $T = \begin{bmatrix}S \\ S'\end{bmatrix}$ and $M = \begin{bmatrix}Q + n\lambda I & P \\ Q' & P' + \frac{n\lambda}{\gamma} I\end{bmatrix}$.

It is easy to verify that d, c, b are the solutions to

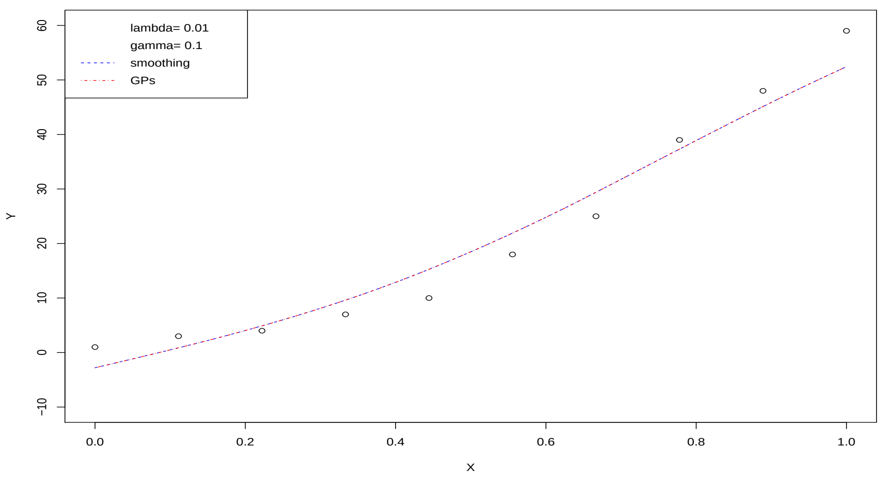
$$\begin{cases}
S^\top(Sd + Qc + Pb - Y) + \gamma S'^\top(S'd + P^\top c + S'^\top P'b - V) = 0, \\
Q(Sd + (Q + n\lambda I)c + Pb - Y) + P(\gamma S'd + \gamma P^\top c + (\gamma P' + n\lambda I)b - \gamma V) = 0, \\
P^\top(Sd + (Q + n\lambda I)c + Pb - Y) + P'(\gamma S'b + P^\top c + (\gamma P' + n\lambda I)b - \gamma V) = 0.
\end{cases}$$

3.5 Numeric Simulation of Smoothing Spline and GPR

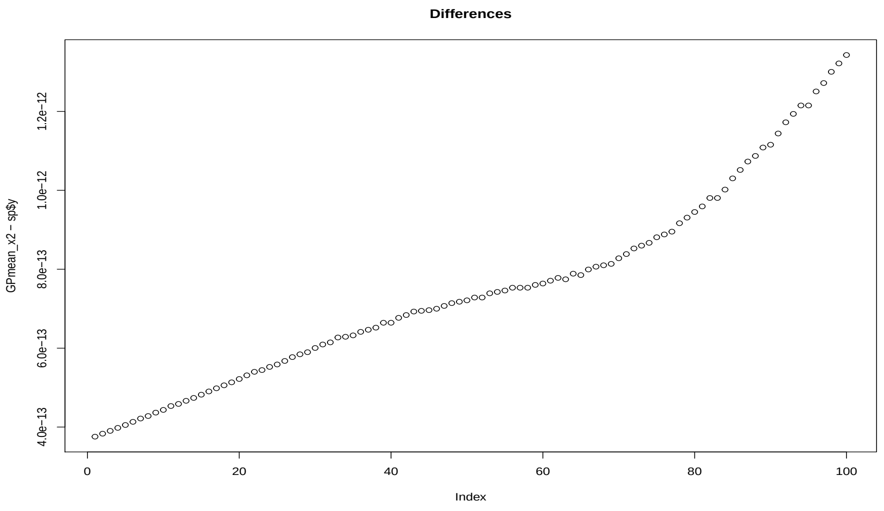
Given $\mathbf{X}$ a length-10 sequence from 0 to 1,

$$\mathbf{Y} = [1, 3, 4, 7, 10, 18, 25, 39, 48, 59]\,,$$

$v_i = \begin{cases} \frac{y_{i+1}-y_i}{x_{i+1}-x_i} & \text{if } 1 \leq i \leq 9 \\ 0 & \text{if } i = 10 \end{cases}$. A simulated result is in figure 1.



(a)



(b)

Figure 3.1: (a) Comparing two methods under the same parameters $\lambda = 0.01$ and $\gamma = 0.1$. In this graph, the blue line is reconstruction from tractor spline, the red line is the mean of Gaussian Process, which is the posterior $\mathbb{E}(f(x) \mid \mathbf{Y}, \mathbf{V})$. (b) The differences between two methods under the same parameters.

Chapter 4

Filtering Problem

4.1 State Space Models

State space models are the natural form of system models relying on the general concept of state. If we describe a system as an operator mapping from the space of inputs to the space of outputs, then we may need the entire input-output history of the system together with the planned input in order to compute the future output values Hangos *et al.* (2006). In an alternative way, by using new information at time t containing all the past information up to the current state and initial conditions to get the current output is possible, that is known as a sequential method. A genetic state space model consists of two sets of equations: state equation and output equation. The state equation describes the evolution of the true input and state variables sequentially as a function and passes the variable one after one, generally, with some noises. The output equation catches the input values and interprets it out by an algebraic equation. A general state space model looks like the following form

$$\text{State equation } x_t = G_t(x_{t-1}) + w_t, \tag{4.1}$$

$$\text{Output equation } y_t = F_t(x_t) + \epsilon_t \tag{4.2}$$

with an initial state x_0, where ϵ_t and w_t are noises passing through the process G_t and F_t. x_t are true status variables and y_t are output values. Many researchers have been interested in this model and its application because of its good property. It can be used to model univariate or multivariate time series, also in the presence of non-stationarity, structural changes, and irregular patterns Petris *et al.* (2009).

The most simple and important system is given by Gaussian linear state space models, also known by dynamic linear models (DLM), which defines a very general class of non-stationary time series models. Firstly, the model is linear, which means G_t and F_t are linear processes and satisfying linearity property. Secondly, the it is specified by a normal prior distribution for the p-dimensional state vector at initial state $t = 0$,

$$x_0 \sim N_p(m_,0, C_0)$$

and two independent zero mean normal distributed noises $\epsilon_t \sim N_p(0, V_t)$ and $w_t \sim N_p(0, W_t)$ Petris *et al.* (2009). The well known Kalman Filter is a particular algorithm that is used to solve state space models in the linear case. This was first derived by Kalman Kalman *et al.* (1960).

In a nonlinear state space model, the process G_t and F_t are no longer linear functions and the situation becomes more complicated. Here gives a simple nonlinear example of such a model, which has been used extensively in the literature for benchmarking numerical filtering techniques Kitagawa (1996) West (1993) Gordon *et al.* (1993) assuming the sequence is Markovian.

$$x_t = \frac{x_{t-1}}{2} + 25\frac{x_{t-1}}{1 + x_{t-1}^2} + 8\cos(1.2t) + u_t$$
$$y_t = \frac{x_t^2}{20} + v_t,$$

where $u_t \sim N(0, \sigma_u^2)$, $v_t \sim N(0, \sigma_v^2)$, $\sigma_u^2 = 10$ and $\sigma_t^2 = 1$ are considered fixed and known. The initial state $x_0 \sim N(0, 10)$. The assumption Markovian keeps the current state x_t only depending on the previous one step x_{t-1} and the observed y_t depending on x_t. A state-space is shown in the diagram below:

$$\begin{tikzcd}
\cdots \arrow[r] & x_{t-1} \arrow[r] \arrow[d] & x_t \arrow[r] \arrow[d] & x_{t+1} \arrow[r] \arrow[d] & \cdots & \text{truth} \\
\cdots & & & & \cdots & \\
\cdots & y_{t-1} & y_t & y_{t+1} & \cdots & \text{observation}
\end{tikzcd}$$

In applications, the process function G_t and F_t contain unknown parameters to be estimated De Jong (1988) and the target is to estimate the true states on sequential observations $y_t, \cdots, y_t$. Then it becomes to estimate a joint density of $p(x_{1:t}, \theta \mid y_{1:t})$, where $x_{1:t} = \{x_1, x_2, \cdots, x_t\}$ are the hidden states and $y_{1:t} = \{y_1, y_2, \cdots, y_t\}$ are the observed outcomes and θ is a set of unknown parameters.

4.2 Sequential Monte Carlo Method

The use of Monte Carlo methods for non-linear filtering can be traced back to the pioneering contributions of Handschin and Mayne (1969) Handschin and Mayne (1969) and Handschin (1970) Handschin (1970). These researchers tried to use an importance sampling paradigm to approximate the target distributions. Later on, an importance sampling algorithms were implemented sequentially in the non-linear filtering context. This algorithm is called sequential importance sampling, often abbreviated SIS, and has been known since the early 1970s. Limited by the power of computers and suffering from sample impoverishment or weight degeneracy, the SIS didn't develop very well until 1993. A particle filter algorithm was proposed to allow rejuvenation of the set of samples by duplicating the samples with high importance weights and, on the contrary, removing samples with low weights Cappé *et al.* (2009). Since then, sequential Monte Carlo (SMC) methods have been applied in many different fields including but not limited to computer vision, signal processing, control, econometrics, finance, robotics, and statistics Arnaud Doucet (2011) Ristic *et al.* (2004).

4.2.1 Filtering Problem and Estimation

Sequential Monte Carlo method, also known as particle filter, is a technique based on sampling and importance sampling methods to find the best state estimation given by Gordon in 1993 Gordon *et al.* (1993) and was the first successful application of sequential Monte Carlo techniques to the field of non-linear filtering Cappé *et al.* (2009). In the state space model, a generic particle filter estimates the posterior distribution of the hidden states using the observation measurement process. The filtering problem is to estimate sequentially the values of the hidden states x_k given the values of the observation process $y_{1:k}$ at any time step k. In another word, it is to find the value of $p(x_k \mid y_{1:k})$. The process is divided into two steps: prediction and updating. In the prediction step, the assumption of Markov Chain is the current status x_k only depends on the previous one x_{k-1}. Then we can calculate the probability of x_k by

$$\begin{aligned} p(x_k \mid y_{1:k-1}) &= \int p(x_k, x_{k-1} \mid y_{1:k-1}) dx_{k-1} \\ &= \int p(x_k \mid x_{k-1}, y_{1:k-1}) p(x_{k-1} \mid y_{1:k-1}) dx_{k-1} \\ &= \int p(x_k \mid x_{k-1}) p(x_{k-1} \mid y_{1:k-1}) dx_{k-1}. \end{aligned}$$

In the updating step, once $p(x_k \mid y_{1:k-1})$ is known, $p(x_k \mid y_{1:k})$ can be found by

$$\begin{aligned} p(x_k \mid y_{1:k}) =& \frac{p(y_k \mid x_k, y_{1:k-1}) p(x_k \mid y_{1:k-1})}{p(y_k \mid y_{1:k-1})} \\ =& \frac{p(y_k \mid x_k) p(x_k \mid y_{1:k-1})}{p(y_k \mid y_{1:k-1})}, \end{aligned}$$

where the normalization $p(y_k \mid y_{1:k-1}) = \int p(y_k \mid x_k)p(x_k \mid y_{1:k-1})dx_k$ Arulampalam *et al.* (2002).

Imagine that the stat space is partitioned as many parts, in which the particles are filled according to some probability measure. The higher probability, the denser the particles are concentrated. Suppose the particles $x_1, \cdots, x_N$ are drawn from the target probability density function $p(x)$, then these particles are used to estimate the expectation and variance of $f(x)$ by

$$\begin{aligned} E(f(x)) &= \int_a^b f(x)p(x)dx \\ Var(f(x)) &= E(f(x) - E(f(x)))^2 p(x)dx. \end{aligned}$$

Back to our target, the posterior distribution or density is empirically represented by a weighted sum of samples $x_1, \cdots, x_N$

$$\hat{p}(x_n \mid y_{1:k}) = \frac{1}{N}\sum_{i=1}^{N} \delta(x_n - x_n^{(i)}) \approx p(x_n \mid y_{1:k}),$$

where $f(x) = \delta(x_n - x_n^{(i)})$ is Dirac delta function. When N is sufficiently large, $\hat{p}(x_n \mid y_{1:k})$ approximates the true posterior $p(x_n \mid y_{1:k})$. By this approximation, the filtering problem becomes to get the expectation of current status

$$\begin{aligned} E(f(x_n)) &\approx \int f(x_n)\hat{p}(x_n \mid y_{1:k})dx_n \\ &= \frac{1}{N}\sum_{i=1}^{N} \int f(x_n)\delta(x_n - x_n^{(i)})dx_n \\ &= \frac{1}{N}\sum_{i=1}^{N} f(x_n^{(i)}). \end{aligned}$$

The expectation is the mean of the status of all particles $x_1, \cdots, x_N$.

However, the posterior distribution is unknown and impossible to sample from the true posterior. So some sampling methods are introduced.

4.2.2 Sampling Methods

Importance sampling

It is common to sample from an easy-to-implement distribution, the so-called proposal distribution $q(x \mid y)$, hence

$$\begin{aligned} E(f(x)) &= \int f(x_k) \frac{p(x_k \mid y_{1:k})}{q(x_k \mid y_{1:k})} q(x_k \mid y_{1:k}) dx_x \\ &= \int f(x_k) \frac{p(x_k) p(y_{1:k} \mid x_k)}{p(y_{1:k}) q(x_k \mid y_{1:k})} q(x_k \mid y_{1:k}) dx_x \\ &= \int f(x_k) \frac{W_k(x_k)}{p(y_{1:k})} q(x_k \mid y_{1:k}) dx_x, \end{aligned}$$

where $W_k(x_k) = \frac{p(x_k)p(y_{1:k}|x_k)}{q(x_k|y_{1:k})} \propto \frac{p(x_k|y_{1:k})}{q(x_k|y_{1:k})}$. Because $p(y_{1:k}) = \int p(y_{1:k} \mid x_k) p(x_k) dx_k$, so the above equation can be rewritten as

$$\begin{aligned} E(f(x)) &= \frac{1}{p(y_{1:k})} \int f(x_k) W_k(x_k) q(x_k \mid y_{1:k}) dx_k \\ &= \frac{\int f(x_k) W_k(x_k) q(x_k \mid y_{1:k}) dx_k}{\int p(y_{1:k} \mid x_k) p(x_k) dx_k} \\ &= \frac{\int f(x_k) W_k(x_k) q(x_k \mid y_{1:k}) dx_k}{\int W_k(x_k) q(x_k \mid y_{1:k}) dx_k} \\ &= \frac{E_{q(x_k|y_{1:k})}[W_k(x_k) f(x_k)]}{E_{q(x_k|y_{1:k})}[W_k(x_k)]}. \end{aligned}$$

To solve the above equation, we can use Monte Carlo method by drawing samples $\{x_k^{(i)}\}$ from $q(x_k \mid y_{1:k})$ and get their expectation, which approximate by

$$\begin{aligned} E(f(x_k)) &\approx \frac{\frac{1}{N}\sum_{i=1}^N W_k(x_k^{(i)}) f(x_k^{(i)})}{\frac{1}{N}\sum_{i=1}^N W_k(x_k^{(i)})} \\ &= \sum_{i=1}^N \tilde{W}_k(x_k^{(i)}) f(x_k^{(i)}), \end{aligned}$$

where $\tilde{W}_k(x_k^{(i)}) = \frac{W_k(x_k^{(i)})}{\sum_{i=1}^N W_k(x_k^{(i)})}$ is factorized weight. Each particles has its own weighted value, so the expectation is a weighted mean. However, the drawback of this method is that the computation is quite expensive. A smarter way is to update $W_k^{(i)}$ recursively. Suppose the proposal distribution

$$q(x_{0:k} \mid y_{1:k}) = q(x_{0:k-1} \mid y_{1:k-1}) q(x_k \mid x_{0:k-1}, y_{1:k}),$$

then the recursive form of the posterior distribution is

$$\begin{aligned}
p(x_{0:k} \mid y_{1:k}) &= \frac{p(y_k \mid x_{0:k}, y_{1:k-1})p(x_{0:k} \mid y_{1:k-1})}{p(y_k \mid y_{1:k-1})} \\
&= \frac{p(y_k \mid x_{0:k}, y_{1:k-1})p(x_k \mid x_{0:k-1}, y_{1:k-1})p(x_{0:k-1} \mid y_{1:k-1})}{p(y_k \mid y_{1:k-1})} \\
&= \frac{p(y_k \mid x_k)p(x_k \mid x_{k-1})p(x_{0:k-1} \mid y_{1:k-1})}{p(y_k \mid y_{1:k-1})} \\
&\propto p(y_k \mid x_k)p(x_k \mid x_{k-1})p(x_{0:k-1} \mid y_{1:k-1}),
\end{aligned}$$

the recursive form of the weights are

$$\begin{aligned}
W_k^{(i)} &\propto \frac{p(x_{0:k}^{(i)} \mid y_{1:k})}{q(x_{0:k}^{(i)} \mid y_{1:k})} \\
&= \frac{p(y_{1:k} \mid x_{0:k}^{(i)})p(x_k^{(i)} \mid x_{k-1}^{(i)})p(x_{0:k-1}^{(i)} \mid y_{1:k-1})}{q(x_k^{(i)} \mid x_{0:k-1}^{(i)}, y_k)q(x_{0:k-1}^{(i)} \mid y_{1:k-1})} \\
&= W_{k-1}^{(i)} \frac{p(y_{1:k} \mid x_{0:k}^{(i)})p(x_k^{(i)} \mid x_{k-1}^{(i)})}{q(x_k^{(i)} \mid x_{0:k-1}^{(i)}, y_k)}.
\end{aligned}$$

Sequential Importance Sampling and Resampling

In practice, we are interested in the current filtered estimate $p(x_k \mid y_{1:k})$ instead of $p(x_{0:k} \mid y_{1:k})$. Provided

$$q(x_k \mid x_{0:k-1}, y_{1:k}) = q(x_k \mid x_{k-1}, y_k),$$

the importance weights $W_k^{(i)}$ can be updated recursively

$$W_k^{(i)} \propto W_{k-1}^{(i)} \frac{p(y_k \mid x_k^{(i)})p(x_k^{(i)} \mid x_{k-1}^{(i)})}{q(x_k^{(i)} \mid x_{k-1}^{(i)}, y_k)}.$$

The problem of SIS filter is that the distribution of importance weights becomes more and more skewed as time increases. Hence, after some iterations, only very few particles have non-zero importance weights. This phenomenon is called *weight degeneracy* or *sample impoverishment* Arnaud Doucet (2011).

The effective sample size N_{eff} is suggested to monitor how bad the degeneration is, which is

$$N_{eff} = \frac{N}{1 + var(w_k^{*(i)})},$$

where $w_k^{*(i)} = \frac{p(x_k^{(i)}|y_{1:k})}{q(x_k^{(i)}|x_{k-1}^{(i)},y_{1:k})}$. The more different between the biggest weight and smallest weight, the worse the degeneration is. In practice, the effective sample size is approximated by

$$\hat{N}_{eff} \approx \frac{1}{\sum_{i=1}^{N}(w_k^{(i)})^2}.$$

If the value of N_{eff} is less than some threshold, some procedure should be used to avoid a worse degeneration. There are two ways one can do: choose an appropriate PDF for importance sampling, or use re-sampling after SIS.

The idea of resampling is keeping the same size of particles, replacing the low weights particles with new ones. As discussed before,

$$p(x_k \mid y_{1:k}) = \sum_{i=1}^{N} w_k^{(i)} \delta(x_k - x_k^{(i)}).$$

After resampling, it becomes

$$\tilde{p}(x_k \mid y_{1:k}) = \sum_{j=1}^{N} \frac{1}{N}\delta(x_k - x_k^{(j)}) = \sum_{i=1}^{N} \frac{n_i}{N}\delta(x_k - x_k^{(i)}),$$

where n_i represents how many times the new particles $x_k^{(j)}$ were duplicated from$x_k^{(i)}$.

Then the process of SIS particle filter with re-sampling is:

- Initial particles when $k = 0$. For $i = 1, \cdots, N$, draw samples $\{x_0^{(i)}\}$ from $p(x_0)$.
- For $k = 1, 2, \cdots$, run the process recursively
  - Importance sampling: draw sample $\{\tilde{x}_k^{(i)}\}_{i=1}^{N}$ from $q(x_k \mid y_{1:k})$, calculate their weights $\tilde{w}_k^{(i)}$ and normalize them.
  - Re-sampling: Re-sample $\{\tilde{x}_k^{(i)}, \tilde{w}_k^{(i)}\}$ and get a new set $\{x_k^{(i)}, \frac{1}{N}\}$.
  - Output the status at time k: $\hat{x}_k = \sum_{i=1}^{N} \tilde{x}_k^{(i)} \tilde{w}_k^{(i)}$.

In SIR, if we choose

$$q(x_k^{(i)} \mid x_{k-1}^{(i)}, y_k) = p(x_k^{(i)} \mid x_{k-1}^{(i)}),$$

the weights become

$$\begin{aligned} w_k^{(i)} &\propto w_{k-1}^{(i)} \frac{p(y_k \mid x_k^{(i)})p(x_k^{(i)} \mid x_{k-1}^{(i)})}{q(x_k^{(i)} \mid x_{k-1}^{(i)}, y_k)} \\ &\propto w_{k-1}^{(i)} p(y_k \mid x_k^{(i)}). \end{aligned}$$

Because $w_{k-1}^{(i)} = \frac{1}{N}$, thus we have $w_k^{(i)} \propto p(y_k \mid x_k^{(i)})$ and

$$w = \frac{1}{\sqrt{2\pi\Sigma}} exp\left(-\frac{1}{2}(y_{true} - y)\Sigma^{-1}(y_{true} - y)\right).$$

Delayed Acceptance Metropolis-Hastings Algorithm

Importance sampling works well only if the proposal density $q(x)$ is similar to $p(x)$. In large and complex problems it is difficult to create a single density $q(x)$ that has this property MacKay (2003). Here, we introduce the Metropolis-Hastings algorithm, which makes use of a proposal density $q(x)$ depending on the current state x_t instead. We assume that we can evaluate $p(\theta)$ for any θ. The transition probabilities should satisfy the detailed balance condition

$$\pi(\theta)p(\theta' \mid \theta) = \pi(\theta')p(\theta \mid \theta'),$$

that means transition from $\pi(\theta)$ to $\pi(\theta')$ has the same probability as that from $\pi(\theta')$ to $\pi(\theta)$. In sampling method, drawing θ_i first and then drawing θ_j should have the same probability as drawing θ_j and then drawing θ_i. However, in most situations, the details balance condition is not satisfied. Therefore, we introduce a function $\alpha(x, y)$ satisfying

$$p(\theta_i)q(\theta_i, \theta_j)\alpha(\theta_i, \theta_j) = p(\theta_j)q(\theta_j, \theta_i)\alpha(\theta_j, \theta_i).$$

A tentative new state θ' is generated from the proposal density $q(\theta'; \theta^{(t)})$. To decide whether to accept the new state, we compute the quantity

$$\alpha = \frac{p(\theta')}{p(\theta^{(t)})} \frac{q(\theta^{(t)}; \theta')}{q(\theta'; \theta^{(t)})}.$$

If $\alpha \geq 1$, then the new state is accepted. Otherwise, the new state is accepted with probability α. A drawback of MH algorithm is a large time consuming in calculating $p(\theta)$ if it's in a irregular structure. A delayed acceptance MH algorithm introduces a cheap approximation $\hat{p}(\theta)$ for $p(\theta)$ in two stages. In stage one, the quantity α_1 is found by a standard MH acceptance formula

$$\alpha_1 = \min\left\{1, \frac{\hat{p}(\theta^*)q(\theta \mid \theta^*)}{\hat{p}(\theta)q(\theta^* \mid \theta)}\right\}, \tag{4.3}$$

where $\hat{p}(\cdot)$ is a cheap estimation for θ and a simple form is $\hat{p}(\cdot) = N(\cdot \mid \hat{\theta}, \sigma)$. Once α_1 is accepted, the process goes into stage two and the acceptance probability α_2 is

$$\alpha_2 = \min\left\{1, \frac{p(\theta^*)\hat{p}(\theta)}{p(\theta)\hat{p}(\theta^*)}\right\}, \tag{4.4}$$

where the overall acceptance probability $\alpha_1\alpha_2$ ensures that detailed balance is satisfied with respect to $p(\cdot)$; however if a rejection occurs at stage one then the expensive evaluation of $p(\theta)$ at stage two is unnecessary.

In a random walk, the proposal density function $q(\cdot)$ can be chosen for some suitable normal distribution, and hence $q(\theta^* \mid \theta) = N(\theta^* \mid \theta, \epsilon^2)$ and $q(\theta \mid \theta^*) = N(\theta \mid \theta^*, \epsilon^2)$ cancel in the Delayed Acceptance MH process stage one **?**. Then in our case, the proposal $\theta' \sim N(\theta^{(t)}, \sigma)$ and the density q is symmetric, so it becomes

$$\alpha = \frac{p^*(\theta')}{p^*(\theta^{(t)})} = \frac{p(y_{1:t} \mid \theta')p(\theta')}{p(y_{1:t} \mid \theta^{(t)})p(\theta^{(t)})}. \tag{4.5}$$

4.3 Bayesian Parameter Estimation

The state transition density and the conditional likelihood function depend not only upon the dynamic state x_t, but also on a static parameter vector θ, which will be stressed by use of the notations $f(x_t \mid x_{t-1}, \theta)$ and $g(y_t \mid x_t, \theta)$. To estimate θ, we would consider a Bayesian method in the following two situations: off-line, estimating the parameters by a batch of data, and on-line, by an instant updated sequential data stream. Specifically, the advantage of Bayesian than maximum likelihood method is that the unknown parameter is considered random and assigned a suitable prior distribution, which is addressed from the experiences of researchers or a learning process and easily to be implemented in the algorithm of machine learning.

Generally, in the Bayesian setting, we choose a suitable prior density $p(\theta)$ for θ and compute the joint posterior density $p(x_{0:t}, \theta \mid y_{0:t})$ in the off-line case, or the sequence of posterior densities $\{p(x_{0:n}, \theta \mid y_{0:n})\}$ in the on-line setting Kantas *et al.* (2009).

4.3.1 Off-line Methods

In the off-line setting, the parameters can be estimated with non-sequential Monte Carlo methods, such as Markov Chain Monte Carlo Robert (2004). However, it is recognized that the sequential MC methods have some significant advantages in some certain cases, like Cappé *et al.* (2009) and Del Moral *et al.* (2006). Additionally, it is difficult to design an efficient MCMC sampling algorithm for a nonlinear non-Gaussian state space model. A Particle MCMC method is proposed by Andrieu *et al.* (2010), which is a new class of MCMC techniques relying on Standard MC methods to build efficient high dimensional proposal distributions.

PMMH jointly updates θ and $x_{0:t}$ for state space models. It proposes a new θ^* from a proposal density function $q(\theta^* \mid \theta)$, and then generates $x^*_{0:t}$ by running bootstrap particle filter with θ^*. The acceptance ratio of this sampler is

$$\begin{aligned}\alpha &= \min\left\{1, \frac{p(x^*_{0:t}, \theta^* \mid y_{0:t}) q((x_{0:t}, \theta) \mid (x^*_{0:t}, \theta^*))}{p(x_{0:t}, \theta \mid y_{0:t}) q((x^*_{0:t}, \theta^*) \mid (x_{0:t}, \theta))}\right\} \\ &= \min\left\{1, \frac{p_{\theta^*}(y_{0:t}) p(\theta^*) q(\theta \mid \theta^*)}{p_\theta(y_{0:t}) p(\theta) q(\theta^* \mid \theta)}\right\}.\end{aligned}$$

The PMMH sampler is an approximation of the ideal MMH sampler for sampling from $p(x^t, \theta \mid y^t)$. Apparently, the higher number of particles N the better the mixing properties of the algorithm, in contrast, the lower efficiency of computation.

4.3.2 On-line Methods

Putting the algorithms on-line means to update the parameters and states instantly as new observations coming into the data stream. For Bayesian dynamic models, however, the most natural option consists in treating the unknown parameter θ, using the state space representation, as a component of the state which has no dynamic evolution, also referred to as a static parameter Cappé *et al.* (2007).

The standard SMC is deficiency for on-line estimation. As a result of the successive resampling steps, after a certain time n, the approximation $\hat{p}(\theta \mid y^{1:t})$ will only contain a single unique value for θ. In other words, SMC approximation of the marginalized parameter posterior distribution is represented by a single Dirac delta function. It also causes error accumulation in successive Monte Carlo (MC) steps grows exponentially or polynomially in time.

The target is to estimate $p(\theta \mid y_{1:t})$ by

$$p(\theta \mid y_{1:t}) \propto p(y_{1:t} \mid \theta) p(\theta) \tag{4.6}$$

without introducing any bias or controlling the bias in states propagation. A pragmatic approach to reduce parameter sample degeneracy and error accumulation in successive MC approximations is to adding an artificial dynamic equation on θ Higuchi (2001) Kitagawa (1998), which gives

$$\theta_{n+1} = \theta_n + \varepsilon_{n+1}.$$

With a small artificial noise, SMC can now be applied to approximate $p(x^t, \theta \mid y^t)$. A related kernel density estimation method proposes a kernel density estimate of the target Liu and West (2001)

$$\hat{p}(\theta \mid y^t) = \frac{1}{N} \sum M(\theta - \theta_n^{(i)}).$$

Both of these methods require a significant amount of tuning.

A fixed-lag practical filtering is used to approximate

$$p(x_{0:n-L}, \theta \mid y_{0:n-1}) \approx p(x_{0:n-L}, \theta \mid y^n)$$

for L large enough in reference Polson *et al.* (2008). $x_{0:n-L}$ has very little influence on observations coming after n. The choice of the lag L is difficult and ther is a non-vanishing bias which is difficult to quantify.

A MCMC kernel with invariant density $p(x^t, \theta \mid y^t)$ is used in SMC algorithm. This method was firstly used in an on-line Bayesian parameter estimation, where the author in Andrieu *et al.* (1999) were using

$$K_n(x'_{1:t}, \theta' \mid x_{1:t}, \theta) = \delta_{x_{1:t}}(x'_{1:t}) p(\theta' \mid x_{1:t}, y_{1:t}),$$

where $p(y^t \mid \theta, x^t) = p(\theta \mid s_t(x^t, y^t))$ and $s_t(x^t, y^t)$ is a fixed-dimensional vector of sufficient statistics. MCMC can be used to maintain the diversity of the samples of θ. Here the stationary distribution for the MCMC will be the full joint posterior distribution of states and parameters and apply MH or Gibbs sampling separately to $p(\theta \mid x^t, y^t)$ and $p(x^t \mid \theta, y^t)$. However, this method is not feasible for large dataset.

Chapter 5

Monte Carlo Markov Chain

5.1 Combined State and Parameters Estimation of Sequential Monte Carlo Algorithm

To work out the best estimations for x and u, one has to solve the target function

$$p(X \mid Y) = \int p(X \mid Y, \theta) p(\theta \mid Y) d\theta. \tag{5.1}$$

The main work need to be done is finding an efficient way to sort out the integration in the above equation. Several methods can be used, such as cross validation, Expectation Maximization algorithm, Gibbs sampling and Metropolis-Hastings algorithm and so on. A Monte Carlo method is popular in research area solving this problem. Monte Carlo method is an algorithm that relies on repeated random sampling to obtain numerical results. To compute an integration of $\int f(x)dx$, one has to sampling as many independent x_i $(i = 1, \cdots, N)$ as possible and numerically to find $\frac{1}{N}\sum_i f(x_i)$ to approximate the target function.

In our target function, we have to sampling θ and use a numerical way to calculate its integration. Here are two ways of solving the sampling problem sequentially:

$$\begin{cases} \text{M1}: & p(\theta \mid Y_{1:t}, Y_{t+1}) \propto p(Y_{1:t}, Y_{t+1} \mid \theta) p(\theta) \\ \text{M2}: & p(\theta \mid Y_{1:t}, Y_{t+1}) \propto p(Y_{t+1} \mid \theta, Y_{1:t}) p(\theta \mid Y_{1:t}) \end{cases}.$$

NOTES: add more..........

5.1.1 General Linear Space

In one dimensional state space model, we consider the hidden state process $\{x_t, t \geq 1\}$ is a stationary and ergodic Markov process and transited by $f(x' \mid x)$. In this paper, we assume that the current state x_t only depends on the previous one step x_{t-1}, which is known as *AR(1)* model. As indicated by its name, the states are not observed directly but by another process $\{y_t, t \geq 1\}$, which is assumed depending on $\{x_t\}$ by the process $g(y \mid x)$ only and independent with each other. If the transition processes f and g are linear and normal distributed, we call this model *Linear Gaussian Model*, that can be written as

$$\begin{aligned} y_t \mid x_t &\sim N(\gamma x_t, \sigma^2) \\ x_t \mid x_{t-1} &\sim N(\phi x_{t-1}, \tau^2), \end{aligned}$$

where σ and τ are errors occurring in processes, γ and ϕ are static process parameters.

In a simple scenario, by assuming $\gamma = 1$ gives us the joint distribution for $x_{0:t}$ and $y_{1:t}$ as following

$$\begin{bmatrix} x \\ y \end{bmatrix} \sim N(0, \Sigma),$$

where Σ^{-1} looks like

$$\begin{bmatrix} \frac{1}{L^2} + \frac{\phi^2}{\tau^2} & \frac{-\phi}{\tau^2} & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \\ \frac{-\phi}{\tau^2} & \frac{1+\phi^2}{\tau^2} + \frac{1}{\sigma^2} & \cdots & 0 & -\frac{1}{\sigma^2} & 0 & \cdots & 0 \\ 0 & \frac{-\phi}{\tau^2} & \cdots & 0 & 0 & -\frac{1}{\sigma^2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\tau^2} + \frac{1}{\sigma^2} & 0 & 0 & \cdots & -\frac{1}{\sigma^2} \\ 0 & -\frac{1}{\sigma^2} & \cdots & 0 & \frac{1}{\sigma^2} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & \frac{1}{\sigma^2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & -\frac{1}{\sigma^2} & 0 & 0 & \cdots & \frac{1}{\sigma^2} \end{bmatrix},$$

is the general procedure matrix denoted as $\Sigma^{-1} = \begin{bmatrix} A & -B \\ -B & B \end{bmatrix}$. Its inverse is

$$\Sigma = \begin{bmatrix} (A-B)^{-1} & (A-B)^{-1} \\ (A-B)^{-1} & (I - A^{-1}B)^{-1}B^{-1} \end{bmatrix} \triangleq \begin{bmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{bmatrix} \tag{5.2}$$

the covariance matrix, where B is a $t \times t$ diagonal matrix with elements $\frac{1}{\sigma^2}$. The covariance matrices $\Sigma_{XX} = (A-B)^{-1}$ and $\Sigma_{YY} = (I - A^{-1}B)^{-1}B^{-1}$ are easily found. Here the parameter θ represents for the unknown parameters ϕ, σ, τ.

To find a recursive way of calculating the log likelihood posteriors, we introduce the Sherman-Morrison-Woodbury formula here first. In the late 1940s and the 1950s, Sherman and MorrisonSherman and Morrison (1950), Woodbury Woodbury (1950), Bartlett Bartlett (1951) and Bodewig Bodewig (1959) discovered the following result. The original Sherman-Morrison-Woodbury (for short SMW) formula has been used to consider the inverse of matrices Deng (2011). In this paper, we will consider the more generalized case.

Theorem 1.1 (Sherman-Morrison-Woodbury). Let $A \in B(H)$ and $G \in B(K)$ both be invertible, and $Y, Z \in B(K, H)$. Then $A + YGZ^*$ is invertible if and only if $G^{-1} + ZA^{-1}Y$ is invertible. In which case,

$$(A + YGZ^*)^{-1} = A^{-1} - A^{-1}Y(G^{-1} + ZA^{-1}Y)^{-1}ZA^{-1}. \tag{5.3}$$

A simple form of SMW formula is Sherman-Morrison formula represented in the following statement Bartlett (1951): Suppose $A \in R^{n\times n}$ is an invertible square matrix and $u, v \in R^n$ are column vectors. Then $A + uv\top$ is invertible $\iff 1 + u^\top A^{-1}v \neq 0$. If $A + uv\top$ is invertible, then its inverse is given by

$$(A + uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^TA^{-1}}{1 + v^TA^{-1}u}. \tag{5.4}$$

The Forecast Distribution $p(y_{t+1} \mid y_{1:t}, \theta)$

The joint distribution for y_{t+1} and $y_{1:t}$ is $p(y_{1:t+1} \mid \theta) \sim N(0, \Sigma_{YY})$, where $\Sigma_{YY} = (I - A^{-1}B)^{-1}B^{-1}$ is the covariance matrix given above. One may find the inverse of the covariance matrix

$$\Sigma_{YY}^{-1} = B(I - A^{-1}B) = \frac{1}{\sigma^4}(\sigma^2 I - A^{-1}) \triangleq \frac{1}{\sigma^4}\begin{bmatrix} Z_{t+1} & b_{t+1} \\ b_{t+1}^\top & K_{t+1} \end{bmatrix}.$$

Therefore, the original form of this covariance is

$$\Sigma_{YY} = \sigma^4 \begin{bmatrix} (Z - bK^{-1}b^\top)^{-1} & -Z^{-1}b(K - b^\top Z^{-1}b)^{-1} \\ -K^{-1}b^\top(Z - bK^{-1}b^\top)^{-1} & (K - b^\top Z^{-1}b)^{-1} \end{bmatrix}.$$

For sake of simplicity, here we are using Z to represent the $t \times t$ matrix Z_{t+1}, b to represent the $t \times 1$ vector b_{t+1} and K to represent the 1×1 constant K_{t+1}.

By denoting $C_{t+1} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$ and multiplying Σ_{YY}^{-1}, it gives us

$$\Sigma_{YY}^{-1} C_{t+1} = \frac{1}{\sigma^4}(\sigma^2 I - A^{-1}) C_{t+1} = \frac{1}{\sigma^4} \begin{bmatrix} b_{t+1} \\ K_{t+1} \end{bmatrix}.$$

In order to find b and K easily, one has to use Sherman-Morrison formula in the following way, that

$$A_{t+1}^{-1} C_{t+1} = \left(I - \frac{M_{t+1}^{-1} u_{t+1} u_{t+1}^\top}{1 + u_{t+1}^\top M_{t+1}^{-1} u_{t+1}} \right) M_{t+1}^{-1} C_{t+1}, \tag{5.5}$$

in which

$$M_{t+1}^{-1} C_{t+1} = \begin{bmatrix} A_t^{-1} & 0 \\ 0 & \sigma^2 \end{bmatrix} C_{t+1} = \sigma^2 C_{t+1},$$

$$u_{t+1}^\top C_{t+1} = \begin{bmatrix} 0 & \cdots & 0 & \frac{-\phi}{\tau} & \frac{1}{\tau} \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} = \frac{1}{\tau}.$$

Then the above equation becomes

$$A_{t+1}^{-1} C_{t+1} = \sigma^2 C_{t+1} - \frac{M_{t+1}^{-1} u_{t+1} \frac{\sigma^2}{\tau}}{1 + u^\top M_{t+1}^{-1} u}. \tag{5.6}$$

Moreover,

$$M_{t+1}^{-1} u_{t+1} = \begin{bmatrix} A_t^{-1} & 0 \\ 0 & \sigma^2 \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ -\frac{\phi}{\tau} \\ \frac{1}{\tau} \end{bmatrix} = \begin{bmatrix} A_t^{-1} & 0 \\ 0 & \sigma^2 \end{bmatrix} \begin{bmatrix} -\frac{\phi}{\tau} C_t \\ \frac{1}{\tau} \end{bmatrix} = \begin{bmatrix} -\frac{\phi}{\tau} A_t^{-1} C_t \\ \frac{\sigma^2}{\tau} \end{bmatrix},$$

$$u^\top M_{t+1}^{-1} u = \begin{bmatrix} 0 & \cdots & 0 & -\frac{\phi}{\tau} & \frac{1}{\tau} \end{bmatrix} \begin{bmatrix} -\frac{\phi}{\tau} A_t^{-1} C_t \\ \frac{\sigma^2}{\tau} \end{bmatrix} = \begin{bmatrix} -\frac{\phi}{\tau} C_t^\top & \frac{1}{\tau} \end{bmatrix} \begin{bmatrix} -\frac{\phi}{\tau} A_t^{-1} C_t \\ \frac{\sigma^2}{\tau} \end{bmatrix} = \frac{\phi^2}{\tau^2} C_t^\top A_t^{-1} C_t + \frac{\sigma^2}{\tau^2}.$$

Thus

$$\begin{aligned} A_{t+1}^{-1} C_{t+1} = \begin{bmatrix} -b_{t+1} \\ \sigma^2 - K_{t+1} \end{bmatrix} &= \sigma^2 C_{t+1} - \frac{1}{1 + \frac{\phi^2}{\tau^2} C_t^\top A_t^{-1} C_t + \frac{\sigma^2}{\tau^2}} \begin{bmatrix} -\frac{\phi \sigma^2}{\tau^2} A_t^{-1} C_t \\ \frac{\sigma^4}{\tau^2} \end{bmatrix} \\ &= \sigma^2 C_{t+1} - \frac{1}{\tau^2 + \phi^2 C_t^\top A_t^{-1} C_t + \sigma^2} \begin{bmatrix} -\phi \sigma^2 A_t^{-1} C_t \\ \sigma^4 \end{bmatrix} \end{aligned} \tag{5.7}$$

and

$$\sigma^2 - K_{t+1} = \sigma^2 - \frac{\sigma^4}{\tau^2 + \phi^2 C_t^\top A_t^{-1} C_t + \sigma^2} = \sigma^2 - \frac{\sigma^4}{\tau^2 + \sigma^2 + \phi^2(\sigma^2 - K_t)},$$

therefore

$$K_{t+1} = \frac{\sigma^4}{\tau^2 + \sigma^2 + \phi^2(\sigma^2 - K_t)}, \tag{5.8}$$

and

$$b_{t+1} = \begin{bmatrix} \frac{b_t \phi K_{t+1}}{\sigma^2} \\ \frac{K_{t+1}(\sigma^2+\tau^2)-\sigma^4}{\phi\sigma^2} \end{bmatrix}, \tag{5.9}$$

$$\begin{aligned}
\bar{\mu}_{t+1} &= 0 - \sigma^4 K^{-1} b^\top (Z - bK^{-1}b^\top)^{-1} \sigma^{-4} (Z - bK^{-1}b^\top) y_{1:t} \\
&= -K^{-1} b^\top y_{1:t} \\
&= \frac{\phi}{\sigma^2} K_t \bar{\mu}_t + \phi(1 - \frac{K_t}{\sigma^2}) y_t, \\
\bar{\Sigma}_{t+1} &= \sigma^4 (K - b^\top Z^{-1} b)^{-1} - \sigma^4 K^{-1} b^\top (Z - bK^{-1}b^\top)^{-1} (Z - bK^{-1}b^\top) Z^{-1} b (K - b^\top Z^{-1} b)^{-1} \\
&= \sigma^4 (I - K^{-1} b^\top Z^{-1} b)(K - b^\top Z^{-1} b)^{-1} \\
&= \sigma^4 K_{t+1}^{-1},
\end{aligned}$$

where $K_1 = \frac{\sigma^4}{\frac{\phi^2}{\tau^2} + \frac{1}{L^2}}$.

The Estimation Distribution $p(x_{t+1} \mid y_{1:t+1}, \theta)$

The joint distribution for x_{t+1} and $y_{1:t+1}$ is $p(x_{t+1}, y_{1:t+1} \mid \theta) \sim N(0, \Gamma)$, where

$$\Gamma = \begin{bmatrix} C_{t+1}^\top (A - B)^{-1} C_{t+1} & C_{t+1}^\top (A - B)^{-1} \\ (A - B)^{-1} C_{t+1} & (I - A^{-1}B)^{-1} B^{-1} \end{bmatrix},$$

where $C_{t+1}^\top$ is a $1 \times t+1$ vector $\begin{bmatrix} 0 & \cdots & 0 & 1 \end{bmatrix}_{1 \times t+1}$ retrieving the last column of a matrix. Because of

$$C_{t+1}^\top A_{t+1}^{-1} = \begin{bmatrix} -b_{t+1}^\top & \sigma^2 - K_{t+1} \end{bmatrix},$$

thus $x_{t+1} \mid y_{1:t+1} \sim N(\bar{\mu}^{(x)}_{t+1}, \bar{\sigma}^{(x)2}_{t+1})$, where

$$\begin{aligned}
\bar{\mu}^{(x)}_{t+1} &= \phi\hat{x}_t + C^\top_{t+1}(A-B)^{-1}B(I-A^{-1}B)y_{1:t+1} \\
&= \phi\hat{x}_t + C^\top_{t+1}A^{-1}By_{1:t+1} \\
&= \phi\hat{x}_t + \frac{1}{\sigma^2}C^\top_{t+1}A^{-1}y_{1:t+1} \\
&= 0 + \frac{1}{\sigma^2}\begin{bmatrix} -b^\top_{t+1} & \sigma^2 - K_{t+1}\end{bmatrix}\begin{bmatrix} y_{1:t} \\ y_{y+1}\end{bmatrix} \\
&= -\frac{1}{\sigma^2}b^\top_{t+1}y_{1:t} + (1-\frac{K_{t+1}}{\sigma^2})y_{t+1} \\
&= \frac{K_{t+1}\bar{\mu}_t}{\sigma^2} + (1-\frac{K_{t+1}}{\sigma^2})y_{t+1} \\
\bar{\sigma}^{(x)2}_{t+1} &= C^\top_{t+1}(A-B)^{-1}C_{t+1} - C^\top_{t+1}(A-B)^{-1}B(I-A^{-1}B)(A-B)^{-1}C_{t+1} \\
&= C^\top_{t+1}(A-B)^{-1}C_{t+1} - C^\top_{t+1}A^{-1}B(A-B)^{-1}C_{t+1} \\
&= C^\top_{t+1}A^{-1}C_{t+1} \\
&= \sigma^2 - K_{t+1}.
\end{aligned}$$

Approximations of The Parameters Posterior

Because of the covariance $\Sigma_{YY} = (I - A^{-1}B)^{-1}B^{-1}$, therefore the inverse is

$$\Sigma^{-1}_{YY} = B(I - A^{-1}B) = BA^{-1}\Sigma^{-1}_{XX}.$$

Given the Choleski decomposition $LL^\top = A$, we have

$$\begin{aligned}
\Sigma^{-1}_{YY} &= BL^{-\top}L^{-1}\Sigma^{-1}_{XX} \\
&= (L^{-1}B)^\top(L^{-1}\Sigma^{-1}_{XX}) \\
&= \text{solve}(L,B)^\top\text{solve}(L,\Sigma^{-1}_{XX}).
\end{aligned}$$

More usefully, by given another Choleski decomposition $RR^\top = A - B = \Sigma^{-1}_{XX}$,

$$\begin{aligned}
Y^\top\Sigma^{-1}_{YY}Y &= \text{solve}(L,BY)^\top\text{solve}(L,\Sigma^{-1}_{XX}Y) \\
&\triangleq W^\top\text{solve}(L,\Sigma^{-1}_{XX}Y)
\end{aligned} \tag{5.10}$$

$$\begin{aligned}
\det\Sigma^{-1}_{YY} &= \det B\det L^{-\top}\det L^{-1}\det R\det R^\top \\
&= \det B(\det L^{-1})^2(\det R)^2.
\end{aligned} \tag{5.11}$$

From the objective function, the posterior distribution of θ is

$$p(\theta \mid Y) \propto p(Y \mid \theta)p(\theta) \propto e^{-\frac{1}{2}Y\Sigma^{-1}_{YY}Y}\sqrt{\det\Sigma^{-1}_{YY}}p(\theta).$$

Then by taking natural logarithm on the posterior of θ and using the useful solutions in equations (5.26) and (5.27), we will have

$$\ln L(\theta) = -\frac{1}{2}Y^{\top}\Sigma_{YY}^{-1}Y + \frac{1}{2}\sum \ln \operatorname{tr}(B) - \sum \ln \operatorname{tr}(L) + \sum \ln \operatorname{tr}(R) + \ln p(\theta). \quad (5.12)$$

5.1.2 High Dimension Parameters Space of OU-Process

The Brownian motion is used to construct the Ornstein Uhlenbeck (OU) process, which has become a popular tool for modeling interest rates and vehicle moving. The derivative of the Brownian motion x_t does not exist at any point in time. Thus, if x_t represents the position of a particle, we might be interested in obtaining its velocity, which is the derivative of the motion. The OU process is an alternative model to the Brownian motion that overcomes the preceding problem. It does this by considering the velocity u_t of a Brownian motion at time t. Over a small time interval, two factors affect the change in velocity: the frictional resistance of the surrounding medium whose effect is proportional to u_t and the random impact of neighboring particles whose effect can be represented by a standard Wiener process. Thus, because mass times velocity equals force, we have that

$$m du_t = -\omega u_t dt + dW_t,$$

where $\omega > 0$ is called the friction coefficient and $m > 0$ is the mass. If we define $\gamma = \omega/m$ and $\lambda = 1/m$, we obtain the OU process with the following differential equation:

$$du_t = -\gamma u_t dt + \lambda dW_t. \quad (5.13)$$

The OU process is used to describe the velocity of a particle in a fluid and is encountered in statistical mechanics. It is the model of choice for random movement toward a concentration point. It is sometimes called a continuous-time Gauss Markov process, where a Gauss Markov process is a stochastic process that satisfies the requirements for both a Gaussian process and a Markov process. Because a Wiener process is both a Gaussian process and a Markov process, in addition to being a stationary independent increment process, it can be considered a Gauss-Markov process with independent increments Kijima (1997).

An OU-process model combing states and velocity is in the form of

$$\begin{cases} du_t = -\gamma u_t dt + \lambda dW_t, \\ dx_t = u_t dt + \xi dW_t'. \end{cases} \quad (5.14)$$

The solution can be found by integrating dt out, that gives us

$$\begin{cases} u_t &= u_{t-1}e^{-\gamma t} + \int_0^t \lambda e^{-\gamma(t-s)} dW_s, \\ x_t &= x_{t-1} + \frac{u_{t-1}}{\gamma}(1-e^{-\gamma t}) + \int_0^t \frac{\lambda}{\gamma} e^{\gamma s}\left(1-e^{-\gamma t}\right) dW_s + \int_0^t \xi dW_s'. \end{cases} \tag{5.15}$$

Therefore, the joint distribution is

$$\begin{bmatrix} x_t \\ u_t \end{bmatrix} \sim N\left(\begin{bmatrix} \mu_t^{(x)} \\ \mu_t^{(u)} \end{bmatrix}, \begin{bmatrix} \sigma_t^{(x)2} & \rho_t \sigma_t^{(x)} \sigma_t^{(u)} \\ \rho_t \sigma_t^{(x)} \sigma_t^{(u)} & \sigma_t^{(u)2} \end{bmatrix} \right), \tag{5.16}$$

where $\mu_t^{(x)}$ and $\mu_t^{(u)}$ are from the forward map process

$$\begin{bmatrix} \mu_t^{(x)} \\ \mu_t^{(u)} \end{bmatrix} = \begin{bmatrix} 1 & \frac{1-e^{-\gamma\Delta_t}}{\gamma} \\ 0 & e^{-\gamma\Delta_t} \end{bmatrix} \begin{bmatrix} x_{t-1}^{(x)} \\ u_{t-1} \end{bmatrix} \triangleq \Phi \begin{bmatrix} x_{t-1}^{(x)} \\ u_{t-1} \end{bmatrix}, \tag{5.17}$$

and

$$\begin{cases} \sigma_t^{(x)2} &= \frac{\lambda^2\left(e^{2\gamma\Delta_t}-1\right)\left(1-e^{-\gamma\Delta_t}\right)^2}{2\gamma^3} + \xi^2\Delta_t \\ \sigma_t^{(u)2} &= \frac{\lambda^2\left(1-e^{-2\gamma\Delta_t}\right)}{2\gamma} \\ \rho_t\sigma_t^{(x)}\sigma_t^{(u)} &= \frac{\lambda^2\left(e^{\gamma\Delta_t}-1\right)\left(1-e^{-2\gamma\Delta_t}\right)}{2\gamma^2} \end{cases}$$

In the above equations, $\Delta_t = T_t - T_{t-1}, \Delta_1 = 0$, $x_0 \sim N(0, L_x^2), u_0 \sim N(0, L_u^2)$, $\rho_t^2 = 1 - \frac{\xi^2\Delta_t}{\sigma_t^{(x)2}}$. To be useful, $1-\rho_t^2 = \frac{\sigma_t^{(x)^2}}{\xi^2\Delta_t}$.

Moreover, the independent observation processes are

$$\begin{cases} y_t = x_t + \epsilon_t, \\ v_t = u_t + \epsilon_t', \end{cases}$$

where $\epsilon_t \sim N(0,\sigma), \epsilon_t' \sim N(0,\tau)$ are normally distributed independent errors. Thus, the joint distribution of observations is

$$\begin{bmatrix} y_t \\ v_t \end{bmatrix} \sim N\left(\begin{bmatrix} x_t \\ u_t \end{bmatrix}, \begin{bmatrix} \sigma^2 & 0 \\ 0 & \tau^2 \end{bmatrix} \right). \tag{5.18}$$

Consequently, the parameter θ of an entire Ornstein-Uhlenbeck process is a set of five parameters from both hidden status and observation process, which is represented as $\theta = \{\gamma, \xi^2, \lambda, \sigma^2, \tau^2\}$.

Starting from the joint distribution of $x_{0:t}, u_{0:t}$ and $y_{1:t}, v_{1:t}$ by given θ, it can be found that

$$\left[\begin{matrix} \tilde{X} \\ \tilde{Y} \end{matrix} \middle| \theta \right] \sim N\left(0, \tilde{\Sigma}\right), \tag{5.19}$$

where $\tilde{X}$ represents for the hidden statues $\{x, u\}$, $\tilde{Y}$ represents for observed $\{y, v\}$, θ is the set of five parameters. The inverse of the covariance matrix $\tilde{\Sigma}^{-1}$ is the procedure matrix in the form of

$$\tilde{\Sigma}^{-1} = \begin{bmatrix} Q_{xx} & Q_{xu} & -\frac{1}{\sigma^2}I & 0 \\ Q_{ux} & Q_{uu} & 0 & -\frac{1}{\tau^2}I \\ -\frac{1}{\sigma^2}I & 0 & \frac{1}{\sigma^2}I & 0 \\ 0 & -\frac{1}{\tau^2}I & 0 & \frac{1}{\tau^2}I \end{bmatrix}.$$

To make the covariance matrix a more beautiful form and convenient computing, $\tilde{X}$, $\tilde{Y}$ and $\tilde{\Sigma}$ can be rearranged in a time series order, that makes $X = \{x_1, u_1, x_2, u_2, \cdots, x_t, u_t\}$, $Y = \{y_1, v_1, y_2, v_2, \cdots, y_t, v_t\}$ and the new procedure matrix Σ^{-1} looks like

$$\Sigma^{-1} = \begin{bmatrix} \sigma_{11}^{(x)2} + \frac{1}{\sigma^2} & \sigma_{11}^{(xu)2} & \cdots & \sigma_{1t}^{(x)2} & \sigma_{1t}^{(xu)2} & -\frac{1}{\sigma^2} & 0 & \cdots & 0 & 0 \\ \sigma_{11}^{(ux)2} & \sigma_{11}^{(u)2} + \frac{1}{\tau^2} & \cdots & \sigma_{1t}^{(ux)2} & \sigma_{1t}^{(x)2} & 0 & -\frac{1}{\tau^2} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \sigma_{t1}^{(x)2} & \sigma_{t1}^{(xu)2} & \cdots & \sigma_{tt}^{(x)2} + \frac{1}{\sigma^2} & \sigma_{tt}^{(xu)2} & 0 & 0 & \cdots & -\frac{1}{\sigma^2} & 0 \\ \sigma_{t1}^{(ux)2} & \sigma_{t1}^{(u)2} & \cdots & \sigma_{tt}^{(ux)2} & \sigma_{tt}^{(u)2} + \frac{1}{\tau^2} & 0 & 0 & \cdots & 0 & -\frac{1}{\tau^2} \\ -\frac{1}{\sigma^2} & 0 & \cdots & 0 & 0 & \frac{1}{\sigma^2} & 0 & \cdots & 0 & 0 \\ 0 & -\frac{1}{\tau^2} & \cdots & 0 & 0 & 0 & \frac{1}{\tau^2} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & -\frac{1}{\sigma^2} & 0 & 0 & 0 & \cdots & \frac{1}{\sigma^2} & 0 \\ 0 & 0 & \cdots & 0 & -\frac{1}{\tau^2} & 0 & 0 & \cdots & 0 & \frac{1}{\tau^2} \end{bmatrix} \triangleq \begin{bmatrix} A_t & - \\ -B_t^\top & \end{bmatrix}$$

where B_t is a $2t \times 2t$ diagonal matrix of observation errors at time t in the form of $\begin{bmatrix} \frac{1}{\sigma^2} & \cdot & \cdot & \cdot & \cdot \\ \cdot & \frac{1}{\tau^2} & \cdot & \cdot & \cdot \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \cdot & \cdot & \cdot & \frac{1}{\sigma^2} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \frac{1}{\tau^2} \end{bmatrix}$. In fact, the matrix A_t is a $2t \times 2t$ bandwidth six sparse matrix at time t in the process. Temporally, we are using A and B to represent the matrices A_t and B_t here. Then we may find the covariance matrix by calculating the inverse of the procedure matrix as

$$\begin{aligned}
\Sigma &= \begin{bmatrix} (A - B^\top B^{-1}B)^{-1} & -(A - B^\top B^{-1}B)^{-1}B^\top B^{-1} \\ -B^{-1}B(A - B^\top B^{-1}B)^{-1} & (B - B^\top A^{-1}B)^{-1} \end{bmatrix} \\
&= \begin{bmatrix} (A - B)^{-1} & (A - B)^{-1} \\ (A - B)^{-1} & (I - A^{-1}B)^{-1}B^{-1} \end{bmatrix} \\
&\triangleq \begin{bmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{bmatrix}.
\end{aligned}$$

The Forecast Distribution $p(Y_{t+1}|Y_{1:t}, \theta)$

We are now using the capital letter Y to represent the joint $\{y, v\}$ and $Y_{1:t} = \{y_1, v_1, y_2, v_2, \cdots, y_t, v_t\}$, $Y_{t+1} = \{y_{t+1}, v_{t+1}\}$. It is known that

$$\begin{aligned}
p(Y_{1:t}, \theta) &\sim N\left(0, \Sigma_{YY}^{(t)}\right) \\
p(Y_{t+1}, Y_{1:t}, \theta) &\sim N\left(0, \Sigma_{YY}^{(t+1)}\right) \\
p(Y_{t+1} \mid Y_{1:t}, \theta) &\sim N\left(\bar{\mu}_{t+1}, \bar{\Sigma}_{t+1}\right)
\end{aligned}$$

where the covariance matrix of the joint distribution is $\Sigma_{YY}^{(t+1)} = (I_{t+1} - A_{t+1}^{-1}B_{t+1})^{-1}B_{t+1}^{-1}$. Then, by taking its inverse, we will get

$$\Sigma_{YY}^{(t+1)(-1)} = B_{t+1}(I_{t+1} - A_{t+1}^{-1}B_{t+1}).$$

To be clear, the matrix B_t is short for the matrix $B_t(\sigma^2, \tau^2)$, which is $2t \times 2t$ diagonal matrix with elements $\frac{1}{\sigma^2}, \frac{1}{\tau^2}$ repeating for t times on its diagonal. For instance, the very simple $B_1(\sigma^2, \tau^2) = \begin{bmatrix} \frac{1}{\sigma^2} & 0 \\ 0 & \frac{1}{\tau^2} \end{bmatrix}_{2\times 2}$ is a 2×2 matrix.

Because of A is symmetric and invertible, B is the diagonal matrix defined as above, then they have the following property

$$\begin{aligned}
AB &= A^\top B^\top = (BA)^\top, \\
A^{-1}B &= A^{-\top}B^\top = (BA^{-1})^\top.
\end{aligned}$$

Followed up the form of $\Sigma_{YY}^{(t+1)(-1)}$, we can find out that

$$\begin{aligned}
\Sigma_{YY}^{(t+1)(-1)} &= B_{t+1}(I_{t+1} - A_{t+1}^{-1}B_{t+1}) \\
&= B_{t+1}(B_{t+1}^{-1} - A_{t+1}^{-1})B_{t+1} \\
&\triangleq \begin{bmatrix} B_t & 0 \\ 0 & B_1 \end{bmatrix} \begin{bmatrix} Z_{t+1} & b_{t+1} \\ b_{t+1}^\top & K_{t+1} \end{bmatrix} \begin{bmatrix} B_t & 0 \\ 0 & B_1 \end{bmatrix}
\end{aligned}$$

where Z_{t+1} is a $2t \times 2t$ matrix, b_{t+1} is a $2t \times 2$ matrix and K_{t+1} is a 2×2 matrix. Thus by taking its inverse again, we will get

$$\Sigma_{YY}^{(t+1)} = \begin{bmatrix} B_t^{-1}(Z_{t+1} - b_{t+1}K_{t+1}^{-1}b_{t+1}^\top)^{-1}B_t^{-1} & -B_t^{-1}Z_{t+1}^{-1}b_{t+1}(K_{t+1} - b_{t+1}^\top Z_{t+1}^{-1}b_{t+1})^{-1}B_1^{-1} \\ -B_1^{-1}K_{t+1}^{-1}b_{t+1}^\top(Z_{t+1} - b_{t+1}K_{t+1}^{-1}b_{t+1}^\top)^{-1}B_t^{-1} & B_1^{-1}(K_{t+1} - b_{t+1}^\top Z_{t+1}^{-1}b_{t+1})^{-1}B_1^{-1} \end{bmatrix}.$$

It is easy to find the relationship between A_{t+1} and A_t in the Sherman-Morrison-Woodbury form is

$$A_{t+1} = \begin{bmatrix} A_t & \cdot & \cdot \\ \cdot & \frac{1}{\sigma^2} & \cdot \\ \cdot & \cdot & \frac{1}{\tau^2} \end{bmatrix} + U_{t+1}U_{t+1}^\top \triangleq M_{t+1} + U_{t+1}U_{t+1}^\top,$$

where $M_{t+1} = \begin{bmatrix} A_t & \cdot & \cdot \\ \cdot & \frac{1}{\sigma^2} & \cdot \\ \cdot & \cdot & \frac{1}{\tau^2} \end{bmatrix} = \begin{bmatrix} A_t & 0 \\ 0 & B_1 \end{bmatrix}$ and its inverse is $M_{t+1}^{-1} = \begin{bmatrix} A_t^{-1} & 0 \\ 0 & B_1^{-1} \end{bmatrix}$. Additionallly, U is a $2t+2 \times 2$ matrix in the following form

$$U_{t+1} = \frac{1}{\sqrt{1-\rho_{t+1}^2}} \begin{bmatrix} \mathbf{0}_{2t-2} & \mathbf{0}_{2t-2} \\ \frac{1}{\sigma_{t+1}^{(x)}} & 0 \\ \frac{1-e^{-\gamma\Delta_{t+1}}}{\gamma\sigma_{t+1}^{(x)}} - \frac{\rho_{t+1}e^{-\gamma\Delta_{t+1}}}{\sigma_{t+1}^{(u)}} & \frac{\sqrt{1-\rho_{t+1}^2}e^{-\gamma\Delta_{t+1}}}{\sigma_{t+1}^{(u)}} \\ -\frac{1}{\sigma_{t+1}^{(x)}} & 0 \\ \frac{\rho_{t+1}}{\sigma_{t+1}^{(u)}} & -\frac{\sqrt{1-\rho_{t+1}^2}}{\sigma_{t+1}^{(u)}} \end{bmatrix} \triangleq \begin{bmatrix} C_tS_{t+1} \\ D_{t+1} \end{bmatrix},$$

denoted by $S_{t+1} = \frac{1}{\sqrt{1-\rho_{t+1}^2}} \begin{bmatrix} \frac{1}{\sigma_{t+1}^{(x)}} & 0 \\ \frac{1-e^{-\gamma\Delta_{t+1}}}{\gamma\sigma_{t+1}^{(x)}} - \frac{\rho_{t+1}e^{-\gamma\Delta_{t+1}}}{\sigma_{t+1}^{(u)}} & \frac{\sqrt{1-\rho_{t+1}^2}e^{-\gamma\Delta_{t+1}}}{\sigma_{t+1}^{(u)}} \end{bmatrix}$, $D_{t+1} = \frac{1}{\sqrt{1-\rho_{t+1}^2}} \begin{bmatrix} -\frac{1}{\sigma_{t+1}^{(x)}} & 0 \\ \frac{\rho_{t+1}}{\sigma_{t+1}^{(u)}} & -\frac{\sqrt{1}}{\sigma} \end{bmatrix}$

and $C_{t+1} = \begin{bmatrix} 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{0}_t \\ I_2 \end{bmatrix}$.

By post-multiplying $\Sigma_{YY}^{(t+1)(-1)}$ with C_{t+1}, it gives us

$$\begin{aligned}
\Sigma_{YY}^{(t+1)(-1)}C_{t+1} &= B_{t+1}(I_{t+1} - A_{t+1}^{-1}B_{t+1})C_{t+1} \\
&= \begin{bmatrix} B_t & 0 \\ 0 & B_1 \end{bmatrix} \left(\begin{bmatrix} B_t^{-1} & 0 \\ 0 & B_1^{-1} \end{bmatrix} - A_{t+1}^{-1} \right) \begin{bmatrix} B_t & 0 \\ 0 & B_1 \end{bmatrix} C_{t+1} \\
&= \begin{bmatrix} B_t & 0 \\ 0 & B_1 \end{bmatrix} \begin{bmatrix} Z_{t+1} & b_{t+1} \\ b_{t+1}^\top & K_{t+1} \end{bmatrix} \begin{bmatrix} B_t & 0 \\ 0 & B_1 \end{bmatrix} C_{t+1} \\
&= \begin{bmatrix} B_t & 0 \\ 0 & B_1 \end{bmatrix} \begin{bmatrix} b_{t+1}B_1 \\ K_{t+1}B_1 \end{bmatrix}.
\end{aligned}$$

and the property of A_{t+1}^{-1} is

$$A_{t+1}^{-1}C_{t+1} = \begin{bmatrix} -b_{t+1} \\ B_1^{-1} - K_{t+1} \end{bmatrix}.$$

Moreover, by pre-multiplying $C_{t+1}^\top$ on the left side of the above equation, we will have

$$C_{t+1}^\top A_{t+1}^{-1} C_{t+1} = B_1^{-1} - K_{t+1}, \tag{5.20}$$

$$K_{t+1} = B_1^{-1} - C_{t+1}^\top A_{t+1}^{-1} C_{t+1}. \tag{5.21}$$

We may use Sherman-Morrison-Woodbury formula to find the inverse of A_{t+1} in a recursive way, which is

$$A_{t+1}^{-1} = (M_{t+1} + U_{t+1}U_{t+1}^\top)^{-1} = M_{t+1}^{-1} - M_{t+1}^{-1}U_{t+1}(I + U_{t+1}^\top M_{t+1}^{-1} U_{t+1})^{-1} U_{t+1}^\top M_{t+1}^{-1}. \tag{5.22}$$

Consequently, it is easy to find that $M_{t+1}^{-1}C_{t+1} = \begin{bmatrix} 0 \\ B_1^{-1} \end{bmatrix}$ and

$$\begin{aligned}
A_{t+1}^{-1}C_{t+1} &= \begin{bmatrix} 0 \\ B_1^{-1} \end{bmatrix} - \begin{bmatrix} A_t^{-1} & 0 \\ 0 & B_1^{-1} \end{bmatrix} \begin{bmatrix} C_t S_{t+1} \\ D \end{bmatrix} (I + U_{t+1}^\top M_{t+1}^{-1} U_{t+1})^{-1} \begin{bmatrix} S_{t+1}^\top C_t^\top & D_{t+1}^\top \end{bmatrix} \begin{bmatrix} 0 \\ B_1^{-1} \end{bmatrix} \\
&= \begin{bmatrix} 0 \\ B_1^{-1} \end{bmatrix} - \begin{bmatrix} A_t^{-1} C_t S_{t+1} \\ B_1^{-1} D_{t+1} \end{bmatrix} (I + U_{t+1}^\top M_{t+1}^{-1} U_{t+1})^{-1} D_{t+1}^\top B_1^{-1} \\
&= \begin{bmatrix} 0 \\ B_1^{-1} \end{bmatrix} - \begin{bmatrix} A_t^{-1} C_t S_{t+1} \\ B_1^{-1} D_{t+1} \end{bmatrix} (I + S_{t+1}^\top C_t^\top A_t^{-1} C_t S_{t+1} + D_{t+1}^\top B_1^{-1} D_{t+1})^{-1} D_{t+1}^\top B_1^{-1} \\
&= \begin{bmatrix} 0 \\ B_1^{-1} \end{bmatrix} - \begin{bmatrix} A_t^{-1} C_t S_{t+1} \\ B_1^{-1} D_{t+1} \end{bmatrix} (I + S_{t+1}^\top (B_1^{-1} - K_t) S_{t+1} + D_{t+1}^\top B_1^{-1} D_{t+1})^{-1} D_{t+1}^\top B_1^{-1}.
\end{aligned}$$

Thus, by using the equation (5.20), we will get

$$K_{t+1} = B_1^{-1} D_{t+1} (I + S_{t+1}^\top (B_1^{-1} - K_t) S_{t+1} + D_{t+1}^\top B_1^{-1} D_{t+1})^{-1} D_{t+1}^\top B_1^{-1}, \tag{5.23}$$

and

$$\begin{aligned}
b_{t+1} &= A_t^{-1} C_t S_{t+1}(I + S_{t+1}^\top (B_1^{-1} - K_t) S_{t+1} + D_{t+1}^\top B_1^{-1} D_{t+1})^{-1} D_{t+1}^\top B_1^{-1} \\
&= \begin{bmatrix} -b_t \\ B_1^{-1} - K_t \end{bmatrix} S_{t+1}(I + S_{t+1}^\top (B_1^{-1} - K_t) S_{t+1} + D_{t+1}^\top B_1^{-1} D_{t+1})^{-1} D_{t+1}^\top B_1^{-1}.
\end{aligned}$$

To achieve the recursive updating formula, firstly we need to find the form of $b_{t+1}^\top B_t^2 Y_{1:t}$. In fact, it is

$$\begin{aligned}
b_{t+1}^\top B_t Y_{1:t} &= B_1^{-\top} D_{t+1}(I + S_{t+1}^\top (B_1^{-1} - K_t) S_{t+1} + D_{t+1}^\top B_1^{-1} D_{t+1})^{-\top} S_{t+1}^\top \begin{bmatrix} -b_t^\top & B_1^{-1} - K_t \end{bmatrix} B_t \begin{bmatrix} Y_{1:t-1} \\ Y_t \end{bmatrix} \\
&= B_1^{-\top} D_{t+1}(I + S_{t+1}^\top (B_1^{-1} - K_t) S_{t+1} + D_{t+1}^\top B_1^{-1} D_{t+1})^{-\top} S_{t+1}^\top \left(-b_t^\top B_{t-1} Y_{1:t-1} + (B_1^{-1} - K_t\right. \\
&= B_1^{-\top} D_{t+1}(I + S_{t+1}^\top (B_1^{-1} - K_t) S_{t+1} + D_{t+1}^\top B_1^{-1} D_{t+1})^{-\top} S_{t+1}^\top \left(K_t B_1 \bar{\mu}_t + (I - K_t B_1) Y_t\right),
\end{aligned}$$

By using equation (5.25) and simplifying the above equation, one can achieve a recursive updating form of the mean, which is

$$\begin{aligned}
\bar{\mu}_{t+1} &= -B_1 K_{t+1}^{-1} b_{t+1}^\top B_t Y_{1:t} \\
&= -D_{t+1}^{-\top} S_{t+1}^\top (K_t B_1 \bar{\mu}_t + (I - K_t B_1) Y_t) \\
&= -D_{t+1}^{-\top} S_{t+1}^\top (Y_t + K_t B_1 (\bar{\mu}_t - Y_t)),
\end{aligned}$$

where by simplifying $D^{-\top} S^\top$, one may find

$$D_{t+1}^{-\top} S_{t+1}^\top = \begin{bmatrix} -1 & -\frac{1 - e^{-\gamma \Delta_{t+1}}}{\gamma} \\ 0 & -e^{-\gamma \Delta_{t+1}} \end{bmatrix} = -\Phi_{t+1},$$

which is the negative of forward process. Then the final form of recursive updating formula are

$$\begin{cases} \bar{\mu}_{t+1} &= \Phi_{t+1} K_t B_1 \bar{\mu}_t + \Phi_{t+1}(I - K_t B_1) Y_t \\ \bar{\Sigma}_{t+1} &= (B_1 K_{t+1} B_1)^{-1} \end{cases}. \tag{5.24}$$

The matrix K_{t+1} is updated via

$$K_{t+1} = B_1^{-1} D_{t+1}(I + S_{t+1}^\top (B_1^{-1} - K_t) S_{t+1} + D_{t+1}^\top B_1^{-1} D_{t+1})^{-1} D_{t+1}^\top B_1^{-1}, \tag{5.25}$$

or updating its inverse in the following form makes the computation faster, that is

$$\begin{cases} K_{t+1}^{-1} &= B_1 D_{t+1}^{-\top} D_{t+1}^{-1} B_1 + B_1 \Phi_{t+1} (B_1^{-1} - K_t) \Phi_{t+1}^\top B_1 + B_1, \\ \bar{\Sigma}_{t+1} &= D_{t+1}^{-\top} D_{t+1}^{-1} + \Phi_{t+1} (B_1^{-1} - K_t) \Phi_{t+1}^\top + B_1^{-1} \end{cases}$$

and $K_1 = B_1^{-1} - A_1^{-1} = \begin{bmatrix} \frac{\sigma^4}{\sigma^2 + L_x^2} & 0 \\ 0 & \frac{\tau^4}{\tau^2 + L_u^2} \end{bmatrix}$.

The Estimation Distribution $p(X_{t+1}|Y_{1:t+1}, \theta)$

The filtering distribution of the state given parameters is $p(X_t \mid Y_{1:t}, \theta)$. To find its form, one can use the joint distribution of X_{t+1} and $Y_{1:t+1}$, which is $p(X_{t+1}, Y_{1:t+1} \mid \theta) \sim N(0, \Gamma)$, where

$$\Gamma = \begin{bmatrix} C_{t+1}^\top (A-B)^{-1} C_{t+1} & C_{t+1}^\top (A-B)^{-1} \\ (A-B)^{-1} C_{t+1} & (I - A^{-1}B)^{-1} B^{-1} \end{bmatrix}.$$

Because of

$$C_{t+1}^\top A_{t+1}^{-1} = \begin{bmatrix} -b_{t+1}^\top & B_1^{-1} - K_{t+1} \end{bmatrix},$$

then $X_{t+1} \mid Y_{1:t+1}, \theta \sim N(\bar{\mu}_{t+1}^{(X)}, \bar{\sigma}_{t+1}^{(X)2})$, where

$$\begin{aligned}
\bar{\mu}_{t+1}^{(X)} &= \Phi \hat{x}_t + C_{t+1}^\top (A-B)^{-1} B (I - A^{-1}B) Y_{1:t+1} \\
&= \Phi \hat{x}_t + C_{t+1}^\top A^{-1} B Y_{1:t+1} \\
&= 0 + \begin{bmatrix} -b_{t+1}^\top & B_1^{-1} - K_{t+1} \end{bmatrix} \begin{bmatrix} B_t & 0 \\ 0 & B_1 \end{bmatrix} \begin{bmatrix} Y_{1:t} \\ Y_{y+1} \end{bmatrix} \\
&= -b^\top B_t Y_{1:t} + (I - B_1 K_{t+1}) Y_{t+1} \\
&= K_{t+1} B_1 \bar{\mu}_{t+1} + (I - B_1 K_{t+1}) Y_{t+1} \\
\bar{\sigma}_{t+1}^{(X)2} &= C_{t+1}^\top (A-B)^{-1} C_{t+1} - C_{t+1}^\top (A-B)^{-1} B (I - A^{-1}B)(A-B)^{-1} C_{t+1} \\
&= C_{t+1}^\top (A-B)^{-1} C_{t+1} - C_{t+1}^\top A^{-1} B (A-B)^{-1} C_{t+1} \\
&= C_{t+1}^\top A^{-1} C_{t+1} \\
&= B_1^{-1} - K_{t+1}.
\end{aligned}$$

Approximations of The Parameters Posterior

From the joint distribution of X and Y, we can find that

$$\Sigma_{YY}^{-1} = B(I - A^{-1}B) = BA^{-1}\Sigma_{XX}^{-1}.$$

Given the Choleski decomposition $LL^\top = A$, we have

$$\begin{aligned}
\Sigma_{YY}^{-1} &= BL^{-\top} L^{-1} \Sigma_{XX}^{-1} \\
&= (L^{-1}B)^\top (L^{-1}\Sigma_{XX}^{-1}) \\
&= \text{solve}(L, B)^\top \text{solve}(L, \Sigma_{XX}^{-1}).
\end{aligned}$$

More usefully, by given another Choleski decomposition $RR^\top = A - B = \Sigma_{XX}^{-1}$,

$$\begin{aligned} Y^\top \Sigma_{YY}^{-1} Y &= \text{solve}(L, BY)^\top \text{solve}(L, \Sigma_{XX}^{-1} Y) \\ &\triangleq W^\top \text{solve}(L, \Sigma_{XX}^{-1} Y) \end{aligned} \tag{5.26}$$

$$\begin{aligned} \det \Sigma_{YY}^{-1} &= \det B \det L^{-\top} \det L^{-1} \det R \det R^\top \\ &= \det B (\det L^{-1})^2 (\det R)^2. \end{aligned} \tag{5.27}$$

From the objective function, the second term in the integral is

$$p(\theta \mid Y) \propto p(Y \mid \theta) p(\theta) \propto e^{-\frac{1}{2} Y \Sigma_{YY}^{-1} Y} \sqrt{\det \Sigma_{YY}^{-1}} P(\theta).$$

Then by taking natural logarithm on the posterior of θ and using the useful solutions in equations (5.26) and (5.27), we will have

$$\ln L(\theta) = -\frac{1}{2} Y^\top \Sigma_{YY}^{-1} Y + \frac{1}{2} \sum \ln \text{tr}(B) - \sum \ln \text{tr}(L) + \sum \ln \text{tr}(R). \tag{5.28}$$

5.2 Prior Distribution for Variance Parameters

The well known Hierarchical Linear Model, where the parameters vary at more then one level, was firstly introduced by Lindley and Smith in 1972 and 1973 Lindley and Smith (1972) Smith (1973). An extension of these models is non-linear Hierarchical Model. Hierarchical Model can be used on data with many levels, although 2-level models are the most common ones. The state space model in equations (4.1) and (4.2) is one of Hierarchical Linear Model if G_t and F_t are linear and non-linear model if G_t and F_t are non-linear processes. Researchers have made a few discussions and works on these both linear and non-linear models. In this section, we only discuss on the prior for variance parameters in these models.

Jonathan and Thomas in Stroud and Bengtsson (2007) have discussed a model, which is slightly different with a Gaussian state-space model in equations (4.1) and (4.2) from section one. The two errors ω_t and ϵ_t are assumed normally distributed as

$$\begin{aligned} \omega_t &\sim N(0, \alpha Q), \\ \epsilon_t &\sim N(0, \alpha R), \end{aligned}$$

where the two matrices R and Q are known and α is an unknown scale factor to be estimated. (Note that a perfect model is obtained by setting $Q = 0$.) Therefore, the density of Gaussian state-space model is

$$p(y_t \mid x_t, \alpha) = N(F(x_t), \alpha R),$$
$$p(x_t \mid x_{t-1}, \alpha) = N(G(x_{t-1}), \alpha Q).$$

The parameter α is assumed *Inverse Gamma* distribution.

Various non-informative and weakly-informative prior distributions have been suggested for scale parameters in hierarchical models. Andrew Gelman gave a discussion on prior distributions for variance parameters in hierarchical models in 2006 Gelman *et al.* (2006). General considerations include using invariance Jeffries (1961), maximum entropy Jaynes (1983) and agreement with classical estimators Box and Tiao (2011).

5.2.1 Priors Discussion

$http://andrewgelman.com/2007/07/18/informative_and/$

Informative and noninformative priors Posted by Andrew on 18 July 2007, 8:04 am Neal writes,

As I start your Bayesian stuff, can I ask you the same question I asked Boris a few years ago, namely, as you note, noninf priors simply represent the situation where we know very little and want the data to speak (so in the end not too far from the classical view). Can you point me to any social science (closer to ps is better) where people actually update, so that the prior in a second study is the posterior of the first (whether or not the two studies done by same person or not).

Equivalently point me to a study which uses non-inf priors. (as more than a toy i know the piece by gill and his student).

Btw do you know the old piece by Harry Roberts, saying that as a scientist all we can report is the likelihood, and that everyone should put their own prior in and then produce their own posterior. so all articles would just be a computer program which takes as input my prior and produces my posterior given the likelihood surface estimated by the author?

My reply: now I like weakly informative priors. But thats new since our books. Regarding informative priors in applied research, we can distinguish three categories:

(1) Prior distributions giving numerical information that is crucial to estimation of the model. This would be a traditional informative prior, which might come from a literature review or explicitly from an earlier data analysis.

(2) Prior distributions that are not supplying any controversial information but are strong enough to pull the data away from inappropriate inferences that are consistent

with the likelihood. This might be called a weakly informative prior.

(3) Prior distributions that are uniform, or nearly so, and basically allow the information from the likelihood to be interpreted probabilistically. These are noninformative priors, or maybe, in some cases, weakly informative.

I have examples of (1), (2), and (3) in my own applied research. Category (3) is the most common for me, but an example of (2) is my 1990 paper with King on seats-votes curves, where we fit a mixture model and used an informative prior to constrain the locations, scales, and masses of the three components. An example of (3) is my 1996 paper with Bois and Jiang where we used an informative prior distribution for several parameters in a toxicology model. We were careful to parameterize the model so that these priors made sense, and the model also had an interesting two-level structure which we discuss in that paper and also in Section 9.1 of Bayesian Data Analysis.

Regarding your question about models where people actually update: we did this in our radon analysis (see here) where the posterior distribution from a national data analysis (based on data from over 80,000 houses) gives inference for each county in the U.S., which is in turn used as the prior distribution for the radon level in your house, which in turn can be updated if you have information from a measurement in your house.

One of the convenient things about doing applied statistics is that eventually I can come up with an example for everything from my own experience. (This also makes it fun to write books.)

Regarding your last comment: yes, there is an idea that a Bayesian wants everyone else to be non-Bayesian so that he or she can do cleaner analyses. I discuss that idea in this talk from 2003 which Ive been too lazy to write up as a paper.

5.2.2 Discussion two

$http : //andrewgelman.com/2007/05/11/weakly_informat/$

Weakly informative priors Posted by Andrew on 11 May 2007, 1:01 pm

Bayesians traditionally consider prior distributions that (a) represent the actual state of subject-matter knowledge, or (b) are completely or essentially noninformative. We consider an alternative strategy: choosing priors that convey some generally useful information but clearly less than we actually have for the particular problem under study. We give some examples, including the Cauchy (0, 2.5) prior distribution for logistic regression coefficients, and then briefly discuss the major unsolved problem in Bayesian inference: the construction of models that are structured enough to learn from data but weak enough to learn from data.

Im speaking Monday on this at Jun Lius workshop on Monte Carlo methods at Harvard (my talk is 9:45-10:30am Monday at 104 Harvard Hall).

Heres the presentation. I think this is potentially a huge advance in how we think about Bayesian models.

Chapter 6

Data Simplify

6.1 Introduction

GPS devices are widely used in orchard planting and maintenance. This location-based system allows orchardist to check trajectory of tractors.

Usually, GPS units record more data than necessary, and causes more errors due to the shelter from branches.

Local simplification algorithms. This algorithm focuses on a couple of particular consecutive points. By analyzing the relationship among these points, a decision is made that which point can be deleted or retained. Distance threshold algorithm is one of these algorithms. All points, for which the distance to the preceding track point is less than a predetermined threshold is deleted. Direction changing algorithm is another one. The point is retained if the change in direction is greater than a predetermined threshold Ivanov (2012).

Global simplification algorithms. These algorithms have an overview of all tracked points. After analyzing the relationships among these points, a decision will be made about which one or more points to delete or retain. The Douglas-Peucker algorithm is one of them Douglas and Peucker (1973).

It is apparently that global simplification algorithms can be used on off-line data analyses and local simplification algorithms will perform better on on-line or real-time track simplification.

However, a pertinent algorithm is required in our case.

6.2 Preliminary

In this section, we elaborate an algorithm used in simplifying global trajectories.

Trajectory is a connection by a time series successive positions recorded by GPS devices. A classical GPS device records skeleton information, including time stamp, latitude, longitude, number of available satellites, etc. Recently, researchers try to enrich trajectory (called Semantic Trajectory) by adding background geographic information to discover meaningful pattern Ying *et al.* (2011). A TS simplification method, represented in Chen *et al.* (2009), consider both the skeleton information and semantic meanings of a trajectory when performing simplification.

In our case, a GPS log is a sequence time series GPS points $p_i \in P$, $P = \{p_1, p_2, \cdots, p_n\}$. Each GPS point p_i contains information of time stamp, latitude and longitude, and semantic information of velocity, heading direction and boom status, which can be written in form of

$$T = \{p_t = [x_t, y_t, v_t, \theta_t, b_t] | t \in \mathbb{R}\}. \tag{6.1}$$

Sequentially connect these points will give us a tractor trajectory.

For a tractor working on an orchard, there are two status of a boom, working and not working. These information is recorded by GPS units, and using $b = 1$ to represented for working and $b = 0$ for not working.

Segment A segment is a part of consecutive trajectory. Regarding to the status of boom, trajectory can be simply divided into two kinds of segment in our dataset, one is boom-working, the other is boom-not-working.

Direction. Direction θ denotes the heading direction of a tractor at a specific point location. This parameter uses north direction as a basis, in which way $0° \leq \theta < 360°$.

6.3 Track Simplification Algorithm

The first two steps are initial simplification to reduce some errors caused by misoperation and GPS units bugs.

- Step 1. Merging. If the length of a segment composed by consecutive boom working or not-working points is less than a threshold, merge this one into its backward segment.

- Step 2. Remove time stamp duplicated data.

Now only two types of segment points are left in GPS log, boom working and not-working, and the length of each segment is greater than the predetermined threshold.

The following algorithm is based on the relationship between a candidate point p_i and it's neighboring points p_{i-1} and p_{i+1}, and the importance of the p_i in the segment where it belongs to.

- Rule 1. The candidate point p_i is retained if it is not linear predictable or cannot be used for linear predicting. With the velocity information v_{i-1}, v_i at point p_{i-1}, p_i and time differences $\Delta t_{i-1} = |t_i - t_{i-1}|, \Delta t_i = |t_{i+1} - t_i|$, an estimated position can be calculated by $\hat{p}_i = \Delta t_{i-1} p_{i-1}$, $\hat{p}_{i+1} = \Delta t_i p_i$. If the distance $|\hat{p}_i - p_i|$ or $|\hat{p}_{i+1} - p_{i+1}|$ is less than a threshold, then the point p_i is not linear predictable or cannot be used for linear predicting.

- Rule 2. Select a candidate point p_i. Retain this point if the distance between p_i and p_{i-1} is greater than the threshold d, where d is the mean distances of these points $p_{i-1}, p_i, \cdots, p_{i+k}$ with same boom status $b_{i-1} = b_i = \cdots = b_{i+k}$.

- Rule 3. Neighbor Heading Changing. The candidate point p_i belongs to the track if $|\theta_i - \theta_{i-1}| + |\theta_i - \theta_{i+1}| > \theta$, where $|\theta_i - \theta_{i-1}|$ and $|\theta_i - \theta_{i+1}|$ are the direction changes between points p_i and p_{i-1} and between points p_i and p_{i+1}, θ is predefined threshold.

- Rule 4. The candidate point p_i belongs to the track if the boom status $b_i \neq b_{i-1}$.

Finally, the point p_i belongs to the track if Rule 1 = TRUE or Rule 2 = TRUE or Rule 3 = TRUE or Rule 4 = TRUE.

6.4 Evaluation

Errors are measured by Synchronized Euclidean Distance Lawson *et al.* (2011). SED measures the distances between the original and compressed trace at the same time. As shown in figure 1Lawson *et al.* (2011). $P_{t1}, \cdots, P_{t5}$ are original points. After simplification, the points P_{t2}, P_{t3} and P_{t4} were removed. The black curve is the original trajectory, in contrast, gray dash line is the simplified trajectory. The gray point P'_{t2} on simplified trajectory has the same time difference as the point P_{t2} on original trajectory does. Then the distance between P_{t2} and P'_{t2} is calculated.

Another way to calculate the difference between a GPS trace and its compressed version is to measure the perpendicular distance. This algorithms ignore the temporal

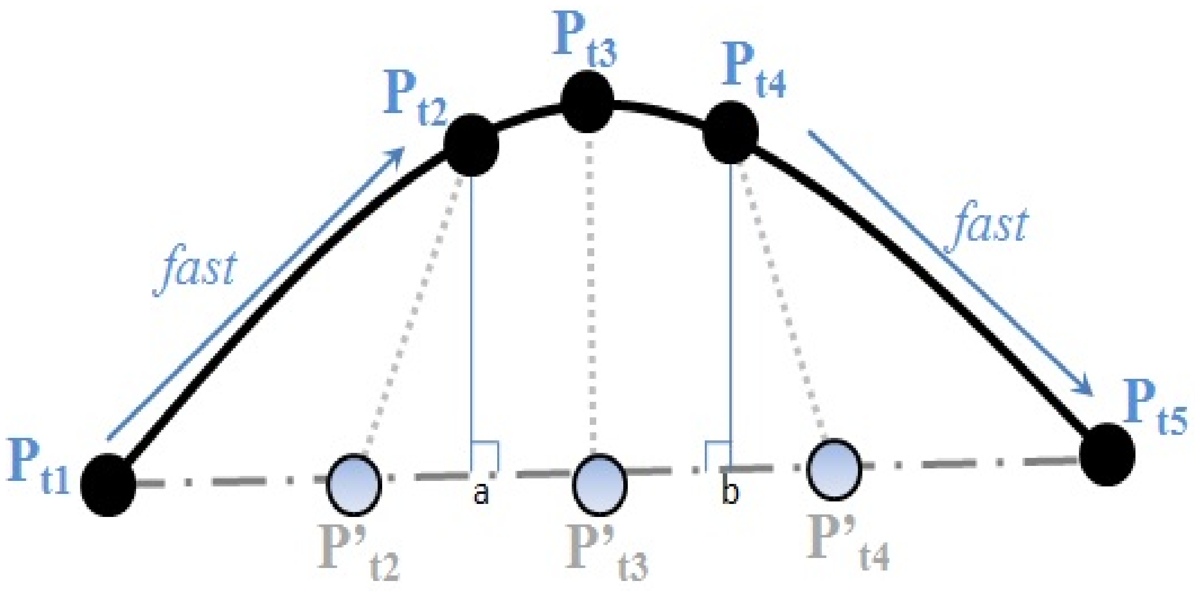


Figure 6.1: Synchronized Euclidean Distance

component and use simple perpendicular distance. The figure 2 Meratnia and Rolf (2004) expresses these difference clearly.

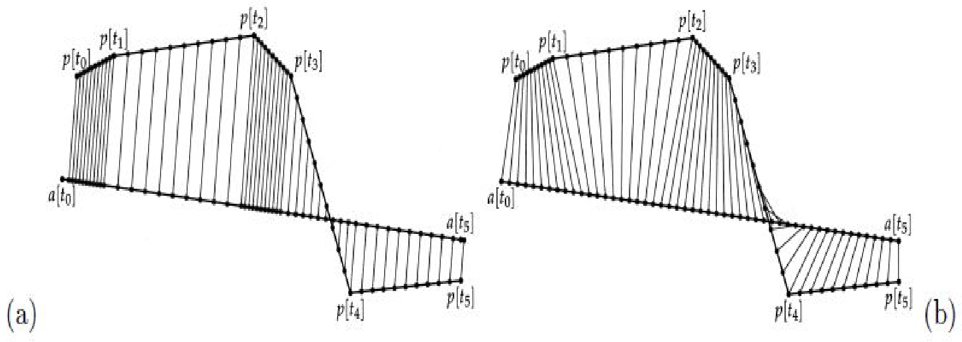


Figure 6.2: (a) error measured at fixed sampling rate as sum of perpendicular distance chords; (b) error measured at fixed sampling rates as sum of time-synchronous distance chords.

6.5 Adaptive Kalman Filter

Discrete Kalman Filter equations the describe the prediction step are

$$\begin{aligned}\hat{x}_k^- &= A\hat{x}_{k-1} + Bu_k \\ P_k^- &= AP_{k-1}A^\top + Q\end{aligned}$$

where $\hat{x}_k^-$ is a priori state estimate, $\hat{x}_k$ is a posteriori state estimate, A is status transition matrix, P_k^- is a priori estimate for error covariance, u_k is an input parameter and Q is process noise covariance.

Discrete Kalman Filter equations the describe the correct step are

$$\begin{aligned}K_k &= P_k^- H^\top (HP_k^- H^\top + R)^{-1} \\ \hat{x}_k &= \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \\ P_k &= (I - K_k H)P_k^-\end{aligned}$$

where K_k is the Kalman gain matrix, z_k is the observed data.

6.6 Experimental results

Our original data set has 1021 rows, each of them contains latitude, longitude, velocity, heading direction and boom information. Douglas-Peucker Algorithm, with distance threshold $0.205m$, retained 847 points. Tractor Algorithm, given a predictable distance being $5m$ and heading direction changing being $30°$, returned the same amount of points. Under the same circumstance, we calculated SED and other information.

Table 6.1: Error Comparing

	Original Data	**DP Algorithm**	**Tractor Algorithm**
Number of Points	1021	847	847
Tracked Distances(m)	74041.31	74038.33	74012.56
SED (m)	NA	1316.715	607.9587

Table (6.1) describe the results after simplifying with DP algorithm and Tractor Algorithm.

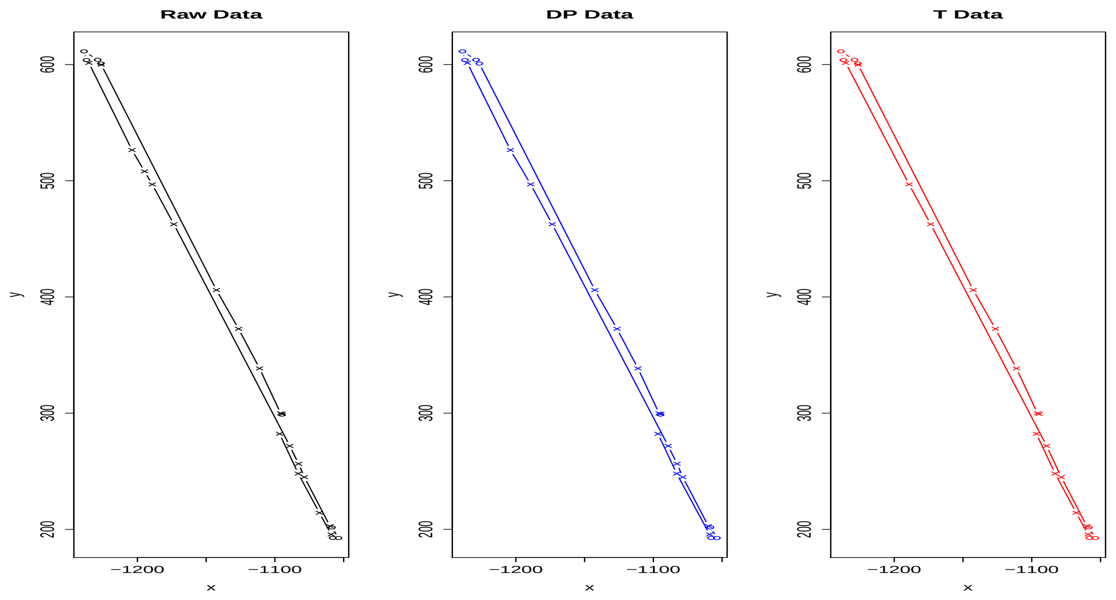


Figure 6.3: A segment start from time 2000 to 3000, recorded by GPS units. On the left side, it's the trajectory connected by raw data with 27 points. In the middle, it's the trajectory connected by simplified data with Douglas-Peucker Algorithm with 24 points. On the right side, it's the trajectory connected by simplified data with Tractor Simplification Algorithm with 23 points.

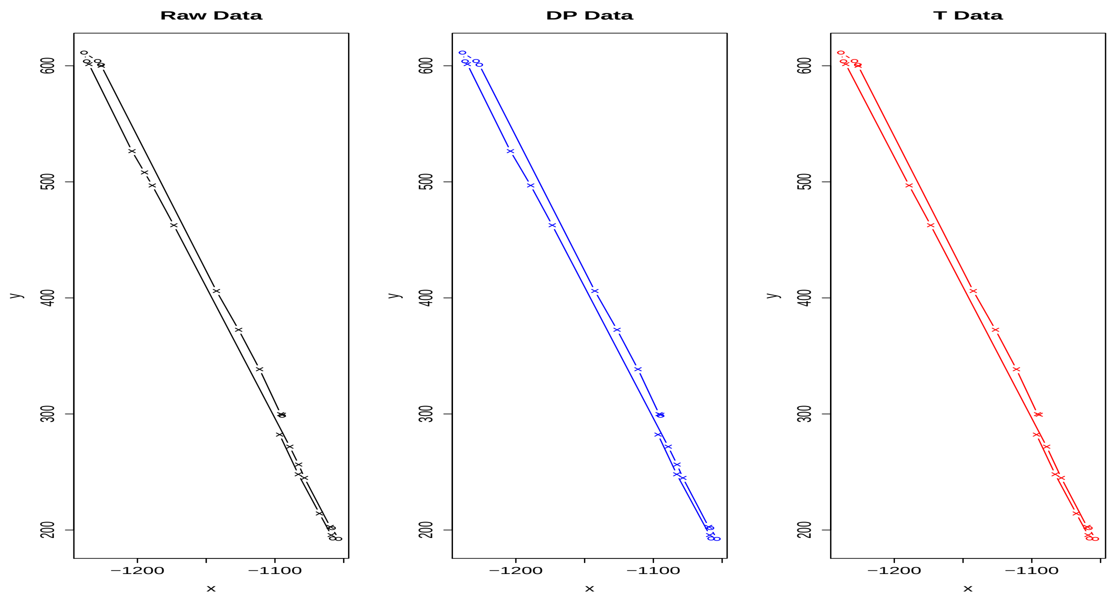


Figure 6.4: Trajectory fitted by Adaptive Kalman Filter. The errors of raw data, DP and tractor algorithm caused by AKF is 26.89217, 23.97877 and 23.97097 respectively.

Chapter 7

Future Work

Chapter 8

Summary

Appendices

Appendix A

Penalty

A.1 Penalty Matrix in (2.15)

The k-th $\Omega^{(k)}$ is a $2n \times 2n$ matrix in the form of

$$\begin{aligned}
\Omega^{(k)}_{2k-1,2k-1} &= \int_{t_k}^{t_{k+1}} \frac{d^2 h^{(k)}_{00}(t)}{dt^2} \frac{d^2 h^{(k)}_{00}(t)}{dt^2} dt = \frac{12}{\Delta_k^3} \\
\Omega^{(k)}_{2k-1,2k} &= \int_{t_k}^{t_{k+1}} \frac{d^2 h^{(k)}_{00}(t)}{dt^2} \frac{d^2 h^{(k)}_{10}(t)}{dt^2} dt = \frac{6}{\Delta_k^2} \\
\Omega^{(k)}_{2k-1,2k+1} &= \int_{t_k}^{t_{k+1}} \frac{d^2 h^{(k)}_{00}(t)}{dt^2} \frac{d^2 h^{(k+1)}_{01}(t)}{dt^2} dt = \frac{-12}{\Delta_k^3} \\
\Omega^{(k)}_{2k-1,2k+2} &= \int_{t_k}^{t_{k+1}} \frac{d^2 h^{(k)}_{00}(t)}{dt^2} \frac{d^2 h^{(k+1)}_{11}(t)}{dt^2} dt = \frac{6}{\Delta_k^2} \\
\Omega^{(k)}_{2k,2k} &= \int_{t_k}^{t_{k+1}} \frac{d^2 h^{(k)}_{10}(t)}{dt^2} \frac{d^2 h^{(k)}_{10}(t)}{dt^2} dt = \frac{4}{\Delta_k} \\
\Omega^{(k)}_{2k,2k+1} &= \int_{t_k}^{t_{k+1}} \frac{d^2 h^{(k)}_{10}(t)}{dt^2} \frac{d^2 h^{(k+1)}_{01}(t)}{dt^2} dt = \frac{-6}{\Delta_k^2} \\
\Omega^{(k)}_{2k,2k+2} &= \int_{t_k}^{t_{k+1}} \frac{d^2 h^{(k)}_{10}(t)}{dt^2} \frac{d^2 h^{(k+1)}_{11}(t)}{dt^2} dt = \frac{2}{\Delta_k} \\
\Omega^{(k)}_{2k+1,2k+1} &= \int_{t_k}^{t_{k+1}} \frac{d^2 h^{(k+1)}_{01}(t)}{dt^2} \frac{d^2 h^{(k+1)}_{01}(t)}{dt^2} dt = \frac{12}{\Delta_k^3} \\
\Omega^{(k)}_{2k+1,2k+2} &= \int_{t_k}^{t_{k+1}} \frac{d^2 h^{(k+1)}_{01}(t)}{dt^2} \frac{d^2 h^{(k+1)}_{11}(t)}{dt^2} dt = \frac{-6}{\Delta_k^2} \\
\Omega^{(k)}_{2k+2,2k+2} &= \int_{t_k}^{t_{k+1}} \frac{d^2 h^{(k+1)}_{11}(t)}{dt^2} \frac{d^2 h^{(k+1)}_{11}(t)}{dt^2} dt = \frac{4}{\Delta_k}
\end{aligned}$$

$k = 1, 2, \cdots, n-1$. It's a bandwidth four matrix. Then

$$\boldsymbol{\Omega} = \sum_{k=1}^{n-1} \Omega^{(k)}$$

A.2 Proof of Theorem 2

Proof. It is obviously that every basis functions are continuous on subinterval $[t_k, t_{k+1}]$. We firstly prove that these basis functions are independent. We have $2n$ basis functions and n knots. Then choose $t_1, \frac{t_1+t_2}{2}, t_2, \frac{t_2+t_3}{2}, \cdots, t_{n-1}, \frac{t_{n-1}+t_n}{3}, \frac{2(t_{n-1}+t_n)}{3}, t_n$ as new $2n$ knots, and denoted by $c_1, c_2, \cdots, c_{2n}$. Then the determinant is

$$\begin{aligned} D(c_1, c_2, \cdots, c_{2n}) &= \begin{vmatrix} N_1(c_1) & N_1(c_2) & \cdots & N_1(c_{2n}) \\ N_2(c_1) & N_2(c_2) & \cdots & N_2(c_{2n}) \\ \vdots & \vdots & \ddots & \vdots \\ N_{2n}(c_1) & N_{2n}(c_2) & \cdots & N_{2n}(c_{2n}) \end{vmatrix} \\ &= \begin{vmatrix} 1 & a_{12} & 0 & 0 & \cdots & 0 & 0 \\ 0 & a_{22} & 0 & 0 & \cdots & 0 & 0 \\ 0 & a_{32} & 1 & a_{34} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & a_{2n-1,2n-1} & 1 \\ 0 & 0 & 0 & 0 & \cdots & a_{2n,2n-1} & 0 \end{vmatrix}, \end{aligned} \tag{A.1}$$

$$\text{where } D(c_1, c_2, \cdots, c_{2n}) = \begin{cases} N_1(t_1) = 1 \\ N_1(\frac{t_1+t_2}{2}) = a_{12} \\ N_2(t_1) = 0 \\ N_2(\frac{t_1+t_2}{2}) = a_{22} \\ N_{2k+1}(\frac{t_k+t_{k+1}}{2}) = a_{2k+1,2k} & k = 1, 2, \cdots, 2n \\ N_{2k+1}(t_{k+1}) = 1 & k = 1, 2, \cdots, 2n \\ N_{2k+1}(\frac{t_{k+1}+t_{k+2}}{2}) = a_{2k+1,2k+2} & k = 1, 2, \cdots, 2n \\ N_{2k+2}(\frac{t_k+t_{k+1}}{2}) = a_{2k+2,2k} & k = 1, 2, \cdots, 2n \\ N_{2k+2}(\frac{t_{k+1}+t_{k+2}}{2}) = a_{2k+2,2k+2} & k = 1, 2, \cdots, 2n, \\ N_{2n-1}(t_{2n-1}) = 0 \\ N_{2n-1}(\frac{t_{2n-1}+t_{2n}}{3}) = a_{2n-1,2n-2} \\ N_{2n-1}(\frac{2(t_{2n-1}+t_{2n})}{3}) = a_{2n-1,2n-1} \\ N_{2n-1}(t_{2n}) = 1 \\ N_{2n}(\frac{t_{2n-1}+t_{2n}}{3}) = a_{2n,2n-2} \\ N_{2n}(\frac{2(t_{2n-1}+t_{2n})}{3}) = a_{2n,2n-1} \\ N_{2n}(t_{2n}) = 0 \\ 0 & otherwise \end{cases}$$

and $a_{ij} \neq 0$. After decomposing determinant D in equation (A.1), gives

$$\det D = \begin{vmatrix} a_{22} & 0 & 0 & \cdots & 0 & 0 \\ a_{32} & 1 & a_{34} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & a_{2n-1,2n-1} & 1 \\ 0 & 0 & 0 & \cdots & a_{2n,2n-1} & 0 \end{vmatrix} = a_{22} \begin{vmatrix} 1 & a_{34} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & a_{2n-1,2n-1} & 1 \\ 0 & 0 & \cdots & a_{2n,2n-1} & 0 \end{vmatrix}$$

$$= \cdots = a_{22}a_{44}\cdots a_{2n-4,2n-4} \begin{vmatrix} a_{2n-2,2n-2} & a_{2n-2,2n-1} & 0 \\ a_{2n-1,2n-2} & a_{2n-1,2n-1} & 1 \\ a_{2n,2n-2} & a_{2n,2n-1} & 0 \end{vmatrix}$$

$$= a_{22}a_{44}\cdots a_{2n-4,2n-4}(a_{2n-2,2n-1}a_{2n,2n-2} - a_{2n,2n-1}a_{2n-2,2n-2}) \neq 0.$$

With the conclusion in Lemma 1, $N_1(t), \ldots, N_{2n}(t)$ are linearly independent on interval $[t_1, t_n]$. Secondly, we prove that basis functions represent any cubic function on each interval $[t_k, t_{k+1}]$. Due to the definition of cubic spline, on interval $[t_k, t_{k+1}]$, a cubic spline $g(t)$ can be written in the form of

$$g(t) = d_k(t-t_k)^3 + c_k(t-t_k)^2 + b_k(t-t_k) + a_k, \text{for } t_k \le t \le t_{k+1} \tag{A.2}$$

For any $f(t)$ on $[t_1, t_n]$, it can be represented as $f(t) = \sum_{k=1}^{2n} \theta_k N_k(t)$. Then for $\forall t \in [t_k, t_{k+1}]$, we have

$$f(t) = \begin{cases} \theta_{2k-1}N_{2k-1}(t) + \theta_{2k}N_{2k}(t) + \theta_{2k+1}N_{2k+1}(t) + \theta_{2k+2}N_{2k+3}(t), & t_k \le t \le t_{k+1} \\ 0, & \text{otherwise} \end{cases},$$

thus

$$\begin{aligned} f(t) =& \theta_{2k-1}\{2(\frac{t-t_k}{t_{k+1}-t_k})^3 - 3(\frac{t-t_k}{t_{k+1}-t_k})^2 + 1\} \\ &+\theta_{2k}\{\frac{(t-t_k)^3}{(t_{k+1}-t_k)^2} - 2\frac{(t-t_k)^2}{t_{k+1}-t_k} + (t-t_k)\} \\ &+\theta_{2k+1}\{-2(\frac{t-t_k}{t_{k+1}-t_k})^3 + 3(\frac{t-t_k}{t_{k+1}-t_k})^2\} + \theta_{2k+2}\{\frac{(t-t_k)^3}{(t_{k+1}-t_k)^2} - \frac{(t-t_k)^2}{t_{k+1}-t_k}\}. \end{aligned}$$

After rearranging, we have

$$\begin{aligned} f(t) =& \{\frac{2\theta_{2k-1}}{(t_{k+1}-t_k)^3} + \frac{\theta_{2k}}{(t_{k+1}-t_k)^2} - \frac{2\theta_{2k+1}}{(t_{k+1}-t_k)^3} + \frac{\theta_{2k+2}}{(t_{k+1}-t_k)^2}\}(t-t_k)^3 \\ &+\{-\frac{3\theta_{2k-1}}{(t_{k+1}-t_k)^2} - \frac{2\theta_{2k}}{(t_{k+1}-t_k)} + \frac{3\theta_{2k+1}}{(t_{k+1}-t_k)^2} - \frac{\theta_{2k+2}}{(t_{k+1}-t_k)}\}(t-t_k)^2 \\ &+\theta_{2k}(t-t_k) + \theta_{2k-1} \end{aligned}$$

where coefficients are

$$\begin{cases} \theta_{2k-1} = a_k \\ \theta_{2k} = b_k \\ -\frac{3\theta_{2k-1}}{(t_{k+1}-t_k)^2} - \frac{2\theta_{2k}}{(t_{k+1}-t_k)} + \frac{3\theta_{2k+1}}{(t_{k+1}-t_k)^2} - \frac{\theta_{2k+2}}{(t_{k+1}-t_k)} = c_k \\ \frac{2\theta_{2k-1}}{(t_{k+1}-t_k)^3} + \frac{\theta_{2k}}{(t_{k+1}-t_k)^2} - \frac{2\theta_{2k+1}}{(t_{k+1}-t_k)^3} + \frac{\theta_{2k+2}}{(t_{k+1}-t_k)^2} = d_k \end{cases}$$

the resulting can always be solved for $\theta_{2k-1}, \theta_{2k}, \theta_{2k+1}, \theta_{2k+2}$ in terms of a_k, b_k, c_k, d_k on interval $[t_k, t_{k+1}]$. So cubic spline on each interval can be represented by basis functions.

Finally, we will prove basis functions are continuous on $[t_1, t_n]$. For any knot t_k, where $t_1 < t_k < t_n$, it is known that $f(t_k) = \theta_{2k-1}$. Moreover,

$$\lim_{t\to t_k+} f(t) = \lim_{t\to t_k+}(\theta_{2k-1}N_{2k-1}(t) + \theta_{2k}N_{2k}(t) + \theta_{2k+1}N_{2k+1}(t) + \theta_{2k+2}N_{2k+3}(t)) = \theta_{2k-1},$$

$$\lim_{t\to t_k-} f(t) = \lim_{t\to t_k-}(\theta_{2k-1}N_{2k-1}(t) + \theta_{2k}N_{2k}(t) + \theta_{2k+1}N_{2k+1}(t) + \theta_{2k+2}N_{2k+3}(t)) = \theta_{2k-1}.$$

So

$$\lim_{t \to t_k+} f(t) = \lim_{t \to t_k-} f(t) = f(t),$$

$f(t)$ is continuous at knots, and then continuous on whole interval $[t_1, t_n]$. $f(t)$ is a continuous cubic spline on interval $[t_1, t_n]$, then $f(t)$ has continuous first and second derivatives. $\square$

A.3 Proof of Lemma 2

Proof. For any spline f,

$$\begin{aligned}
&\frac{1}{n}\sum_{j=1}^{n}(y_j^* - f(t_j))^2 + \frac{\gamma}{n}\sum_{j=1}^{n}(v_j^* - f'(t_j)) + \lambda\int f''^2 \\
\geq&\frac{1}{n}\sum_{j\neq i}^{n}(y_j^* - f(t_j))^2 + \frac{\gamma}{n}\sum_{j\neq i}^{n}(v_j^* - f'(t_j)) + \lambda\int f''^2 \\
\geq&\frac{1}{n}\sum_{j\neq i}^{n}(y_j^* - \hat{f}^{(-i)}(t_j))^2 + \frac{\gamma}{n}\sum_{j\neq i}^{n}(v_j^* - \hat{f}'^{(-i)}(t_j)) + \lambda\int \hat{f}^{(-i)''2} \\
=&\frac{1}{n}\sum_{j=1}^{n}(y_j^* - \hat{f}^{(-i)}(t_j))^2 + \frac{\gamma}{n}\sum_{j=1}^{n}(v_j^* - \hat{f}'^{(-i)}(t_j)) + \lambda\int \hat{f}^{(-i)''2}
\end{aligned} \tag{A.3}$$

by the definition of $\hat{f}^{(-i)}$, $\hat{f}'^{(-i)}$ and $y_i^* = \hat{f}^{(-i)}(t_i)$, $v_i^* = \hat{f}'^{(-i)}(t_i)$. It follows that $\hat{f}^{(-i)}$ is the minimizer of the MSE function (*****), so that

$$\hat{\mathbf{f}}^{(-i)} = S\mathbf{y}^* + \gamma T\mathbf{v}^* \tag{A.4}$$

$$\hat{\mathbf{f}}'^{(-i)} = U\mathbf{y}^* + \gamma V\mathbf{v}^* \tag{A.5}$$

$\square$

A.4 Proof of Theorem 3

Proof.

$$\begin{aligned}\hat{f}^{(-i)}(t_i) - y_i =& \sum_{j=1}^{n} S_{ij} y_j^* + \gamma \sum_{j=1}^{n} T_{ij} v_j^* - y_i^* \\ =& \sum_{j \neq i}^{n} S_{ij} y_j + \gamma \sum_{j \neq i}^{n} T_{ij} v_j + S_{ii} \hat{f}^{(-i)}(t_i) + \gamma T_{ii} \hat{f}'^{(-i)}(t_i) - y_i \\ =& \sum_{j=1}^{n} S_{ij} y_j + \gamma \sum_{j=1}^{n} T_{ij} v_j + S_{ii}(\hat{f}^{(-i)}(t_i) - y_i) + \gamma T_{ii}(\hat{f}'^{(-i)}(t_i) - v_i) - y_i \\ =& (\hat{f}(t_i) - y_i) + S_{ii}(\hat{f}^{(-i)}(t_i) - y_i) + \gamma T_{ii}(\hat{f}'^{(-i)}(t_i) - v_i).\end{aligned} \tag{A.6}$$

Additionally,

$$\begin{aligned}\hat{f}'^{(-i)}(t_i) - v_i =& \sum_{j=1}^{n} U_{ij} y_j^* + \gamma \sum_{j=1}^{n} V_{ij} v_j^* - v_i^* \\ =& \sum_{j \neq i}^{n} U_{ij} y_j + \gamma \sum_{j \neq i}^{n} V_{ij} v_j + U_{ii} \hat{f}^{(-i)}(t_i) + \gamma V_{ii} \hat{f}'^{(-i)}(t_i) - v_i \\ =& \sum_{j=1}^{n} U_{ij} y_j + \gamma \sum_{j=1}^{n} V_{ij} v_j + U_{ii}(\hat{f}^{(-i)}(t_i) - y_i) + \gamma V_{ii}(\hat{f}'^{(-i)}(t_i) - v_i) - v_i \\ =& (\hat{f}'(t_i) - v_i) + U_{ii}(\hat{f}^{(-i)}(t_i) - y_i) + \gamma V_{ii}(\hat{f}'^{(-i)}(t_i) - v_i).\end{aligned} \tag{A.7}$$

Then

$$\hat{f}'^{(-i)}(t_i) - v_i = \frac{\hat{f}'(t_i) - v_i}{1 - \gamma V_{ii}} + \frac{U_{ii}(\hat{f}^{(-i)}(t_i) - y_i)}{1 - \gamma V_{ii}}. \tag{A.8}$$

After substituting equation (A.8) into (A.6), we get

$$\hat{f}^{(-i)}(t_i) - y_i = \frac{\hat{f}(t_i) - y_i + \gamma \frac{T_{ii}}{1-\gamma V_{ii}}(\hat{f}'(t_i) - v_i)}{1 - S_{ii} - \gamma \frac{T_{ii}}{1-\gamma V_{ii}} U_{ii}}. \tag{A.9}$$

Then

$$CV(\lambda, \gamma) = \frac{1}{n} \sum_{i=1}^{n} \frac{\hat{f}(t_i) - y_i + \gamma \frac{T_{ii}}{1-\gamma V_{ii}}(\hat{f}'(t_i) - v_i)}{1 - S_{ii} - \gamma \frac{T_{ii}}{1-\gamma V_{ii}} U_{ii}}. \tag{A.10}$$

□

Appendix B

Matrix

B.1 2D Matrix

$$\Sigma_t = \begin{bmatrix} \sigma_t^{(x)2} & \rho_t \sigma_t^{(x)} \sigma_t^{(u)} \\ \rho_t \sigma_t^{(x)} \sigma_t^{(u)} & \sigma_t^{(u)2} \end{bmatrix}$$

$$\Sigma_t^{-1} = \frac{1}{1-\rho_t^2} \begin{bmatrix} \frac{1}{\sigma_t^{(x)2}} & -\frac{\rho_t}{\sigma_t^{(x)}\sigma_t^{(u)}} \\ -\frac{\rho_t}{\sigma_t^{(x)}\sigma_t^{(u)}} & \frac{1}{\sigma_t^{(u)2}} \end{bmatrix}$$

$$M_t^\top = \begin{bmatrix} 1 & 0 \\ \frac{1-e^{-\gamma\Delta_t}}{\gamma} & e^{-\gamma\Delta_t} \end{bmatrix}$$

$$z_t = \begin{bmatrix} x_t \\ u_t \end{bmatrix}$$

$$M_t^\top \Sigma_t^{-1} = \frac{1}{1-\rho_t^2} \begin{bmatrix} \frac{1}{\sigma_t^{(x)2}} & -\frac{\rho_t}{\sigma_t^{(x)}\sigma_t^{(u)}} \\ \frac{1-e^{-\gamma\Delta_t}}{\gamma\sigma_t^{(x)2}} - \frac{\rho_t e^{-\gamma\Delta_t}}{\sigma_t^{(x)}\sigma_t^{(u)}} & -\frac{\rho_t(1-e^{-\gamma\Delta_t})}{\gamma\sigma_t^{(x)}\sigma_t^{(u)}} + \frac{e^{-\gamma\Delta_t}}{\sigma_t^{(u)2}} \end{bmatrix}$$

$$M_t^\top \Sigma_t^{-1} M_t = \frac{1}{1-\rho_t^2} \begin{bmatrix} \frac{1}{\sigma_t^{(x)2}} & \frac{1-e^{-\gamma\Delta_t}}{\gamma\sigma_t^{(x)2}} - \frac{\rho_t e^{-\gamma\Delta_t}}{\sigma_t^{(x)}\sigma_t^{(u)}} \\ \frac{1-e^{-\gamma\Delta_t}}{\gamma\sigma_t^{(x)2}} - \frac{\rho_t e^{-\gamma\Delta_t}}{\sigma_t^{(x)}\sigma_t^{(u)}} & \frac{(1-e^{-\gamma\Delta_t})^2}{\gamma^2\sigma_t^{(x)2}} - \frac{2\rho_t e^{-\gamma\Delta_t}(1-e^{-\gamma\Delta_t})}{\gamma\sigma_t^{(x)}\sigma_t^{(u)}} + \frac{e^{-2\gamma\Delta_t}}{\sigma_t^{(u)2}} \end{bmatrix}$$

$$\begin{bmatrix}
\frac{1-\rho_t^2}{\sigma_1^{(x)2}} + \frac{1}{\sigma_2^{(x)2}} & -\frac{1}{\sigma_2^{(x)2}} & 0 & 0 & \frac{1-e^{-\gamma\Delta_2}}{\gamma\sigma_2^{(x)2}} - \frac{\rho_t e^{-\gamma\Delta_2}}{\sigma_2^{(x)}\sigma_2^{(u)}} & \frac{\rho_t}{\sigma_2^{(x)}\sigma_2^{(u)}} & 0 & 0 \\
-\frac{1}{\sigma_2^{(x)2}} & \frac{1}{\sigma_2^{(x)2}} + \frac{1}{\sigma_3^{(x)2}} & -\frac{1}{\sigma_3^{(x)2}} & 0 & -\frac{1-e^{-\gamma\Delta_2}}{\gamma\sigma_2^{(x)2}} + \frac{\rho_t e^{-\gamma\Delta_2}}{\sigma_2^{(x)}\sigma_2^{(u)}} & \frac{1-e^{-\gamma\Delta_3}}{\gamma\sigma_3^{(x)2}} - \frac{\rho_t e^{-\gamma\Delta_3}}{\sigma_3^{(x)}\sigma_3^{(u)}} - \frac{\rho_t}{\sigma_2^{(x)}\sigma_2^{(u)}} & \frac{\rho_t}{\sigma_3^{(x)}\sigma_3^{(u)}} & 0 \\
0 & -\frac{1}{\sigma_3^{(x)2}} & \frac{1}{\sigma_3^{(x)2}} + \frac{1}{\sigma_4^{(x)2}} & -\frac{1}{\sigma_4^{(x)2}} & 0 & -\frac{1-e^{-\gamma\Delta_3}}{\gamma\sigma_3^{(x)2}} + \frac{\rho_t e^{-\gamma\Delta_3}}{\sigma_3^{(x)}\sigma_3^{(u)}} & \frac{1-e^{-\gamma\Delta_4}}{\gamma\sigma_4^{(x)2}} - \frac{\rho_t e^{-\gamma\Delta_4}}{\sigma_4^{(x)}\sigma_4^{(u)}} - \frac{\rho_t}{\sigma_3^{(x)}\sigma_3^{(u)}} & \frac{\rho_t}{\sigma_4^{(x)}\sigma_4^{(u)}} \\
0 & 0 & -\frac{1}{\sigma_4^{(x)2}} & \frac{1}{\sigma_4^{(x)2}} & 0 & 0 & -\frac{1-e^{-\gamma\Delta_4}}{\gamma\sigma_4^{(x)2}} + \frac{\rho_t e^{-\gamma\Delta_4}}{\sigma_4^{(x)}\sigma_4^{(u)}} & -\frac{\rho_t}{\sigma_4^{(x)}\sigma_4^{(u)}} \\
\top & \top & 0 & 0 & \frac{1-\rho_t^2}{\sigma_1^{(u)2}} + C_2 & -\frac{e^{-\gamma\Delta_2}}{\sigma_2^{(u)2}} + \frac{\rho_t(1-e^{-\gamma\Delta_2})}{\gamma\sigma_2^{(x)}\sigma_2^{(u)}} & 0 & 0 \\
\top & \top & \top & 0 & -\frac{e^{-\gamma\Delta_2}}{\sigma_2^{(u)2}} + \frac{\rho_t(1-e^{-\gamma\Delta_2})}{\gamma\sigma_2^{(x)}\sigma_2^{(u)}} & \frac{1}{\sigma_2^{(u)2}} + C_3 & -\frac{e^{-\gamma\Delta_3}}{\sigma_3^{(u)2}} + \frac{\rho_t(1-e^{-\gamma\Delta_3})}{\gamma\sigma_3^{(x)2}\sigma_3^{(u)}} & 0 \\
0 & \top & \top & \top & 0 & -\frac{e^{-\gamma\Delta_3}}{\sigma_3^{(u)2}} + \frac{\rho_t(1-e^{-\gamma\Delta_3})}{\gamma\sigma_3^{(x)}\sigma_3^{(u)}} & \frac{1}{\sigma_3^{(u)2}} + C_4 & -\frac{e^{-\gamma\Delta_4}}{\sigma_4^{(u)2}} + \frac{\rho_t(1-e^{-\gamma\Delta_4})}{\gamma\sigma_4^{(x)}\sigma_4^{(u)}} \\
0 & 0 & \top & \top & 0 & 0 & -\frac{e^{-\gamma\Delta_4}}{\sigma_4^{(u)2}} + \frac{\rho_t(1-e^{-\gamma\Delta_4})}{\gamma\sigma_4^{(x)}\sigma_4^{(u)}} & \frac{1}{\sigma_4^{(u)2}}
\end{bmatrix}$$

$$\frac{\cdots\gamma\Delta_t)^2}{\cdots x)^2} + \frac{e^{-2\gamma\Delta_t}}{\sigma_t^{(u)2}} - \frac{2\rho_t e^{-\gamma\Delta_t}(1-e^{-\gamma\Delta_t})}{\gamma\sigma_t^{(x)}\sigma_t^{(u)}},\ \Delta_t = T_t - T_{t-1}.$$

References

Abramovich, F., Sapatinas, T., and Silverman, B. W. (1998). Wavelet thresholding via a Bayesian approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *60*(4), 725–749.

Agarwal, P. K., Arge, L., and Erickson, J. (2003). Indexing moving points. *Journal of Computer and System Sciences*, *66*(1), 207–243.

Andrieu, C., De Freitas, N., and Doucet, A. (1999). Sequential MCMC for Bayesian model selection. In *Higher-Order Statistics, 1999. Proceedings of the IEEE Signal Processing Workshop on*, 130–134. IEEE.

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *72*(3), 269–342.

Arnaud Doucet, Nando de Freitas, N. G. (Ed.) (2011). *Sequential Monte Carlo Methods in Practice*. Springer-Verlag New York.

Arulampalam, M. S., Maskell, S., Gordon, N., and Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on signal processing*, *50*(2), 174–188.

Aydin, D. and Tuzemen, M. S. (2012). Smoothing parameter selection problem in nonparametric regression based on smoothing spline: A simulation study. *Journal of Applied Sciences*, *12*(7), 636.

Bartlett, M. S. (1951). An inverse matrix adjustment arising in discriminant analysis. *The Annals of Mathematical Statistics*, *22*(1), 107–111.

Berlinet, A. and Thomas-Agnan, C. (2011). *Reproducing kernel Hilbert spaces in probability and statistics*. Springer Science & Business Media.

Bodewig, E. (1959). Matrix Calculus, North.

Box, G. E. and Tiao, G. C. (2011). *Bayesian inference in statistical analysis*, Volume 40. John Wiley & Sons.

Cappé, O., Godsill, S. J., and Moulines, E. (2007). An overview of existing methods and recent advances in sequential Monte Carlo. *Proceedings of the IEEE*, *95*(5), 899–924.

Cappé, O., Moulines, E., and Rydén, T. (2009). Inference in hidden markov models. In *Proceedings of EUSFLAT Conference*, 14–16.

Chen, W.-K. (2009). *Feedback, nonlinear, and distributed circuits.* CRC Press.

Chen, Y., Jiang, K., Zheng, Y., Li, C., and Yu, N. (2009). Trajectory simplification method for location-based social networking services. In *Proceedings of the 2009 International Workshop on Location Based Social Networks*, 33–40. ACM.

Chenjian RAN, Z. D. (2010). Self-tuning weighted measurement fusion Kalman filter and its convergence. *J Control Theory Application*, *4*, 435–440.

Craven, P. and Wahba, G. (1978). Smoothing noisy data with spline functions. *Numerische Mathematik*, *31*(4), 377–403.

De Boor, C., De Boor, C., Mathématicien, E.-U., De Boor, C., and De Boor, C. (1978). *A practical guide to splines*, Volume 27. Springer-Verlag New York.

De Jong, P. (1988). The likelihood for a state space model. *Biometrika*, 165–169.

Del Moral, P., Doucet, A., and Jasra, A. (2006). Sequential monte carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *68*(3), 411–436.

Deng, C. Y. (2011). A generalization of the Sherman–Morrison–Woodbury formula. *Applied Mathematics Letters*, *24*(9), 1561–1564.

Donoho, D. L. and Johnstone, I. M. (1995). Adapting to unknown smoothness via wavelet shrinkage. *Journal of the american statistical association*, *90*(432), 1200–1224.

Donoho, D. L., Johnstone, I. M., and Kerkyacharian (1995). Wavelet shrinkage: asymptopia? *Journal of the Royal Statistical Society. Series B (Methodological)*, 301–369.

Donoho, D. L. and Johnstone, J. M. (1994). Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, *81*(3), 425–455.

Douglas, D. H. and Peucker, T. K. (1973). Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, *10*(2), 112–122.

Ellis, D., Sommerlade, E., and Reid, I. (2009). Modelling pedestrian trajectory patterns with gaussian processes. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, 1229–1234. IEEE.

Erkorkmaz, K. and Altintas, Y. (2001). High speed CNC system design. Part I: jerk limited trajectory generation and quintic spline interpolation. *International Journal of machine tools and manufacture*, *41*(9), 1323–1345.

Gelman, A. *et al.* (2006). Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper). *Bayesian analysis*, *1*(3), 515–534.

Gordon, N. J., Salmond, D. J., and Smith, A. F. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Proceedings F (Radar and Signal Processing)*, Volume 140, 107–113. IET.

Green, P. J. and Silverman, B. W. (1993). *Nonparametric regression and generalized linear models: a roughness penalty approach.* CRC Press.

Greg Welch, G. B. (2006). An Introduction to the Kalman Filter. *UNC-Chapel Hill*, *TR 95-041.*

Gu, C. (1998). Model indexing and smoothing parameter selection in nonparametric function estimation. *Statistica Sinica*, 607–623.

Gu, C. (2013). *Smoothing Spline ANOVA Models Second Edition.* Springer New York Heidelberg Dordrecht London.

Handschin, J. (1970). Monte Carlo techniques for prediction and filtering of non-linear stochastic processes. *Automatica*, *6*(4), 555–563.

Handschin, J. E. and Mayne, D. Q. (1969). Monte Carlo techniques to estimate the conditional expectation in multi-stage non-linear filtering. *International journal of control*, *9*(5), 547–559.

Hangos, K. M., Bokor, J., and Szederkényi, G. (2006). *Analysis and control of nonlinear process systems.* Springer Science & Business Media.

Hastie, T., Tibshirani, R., Friedman, J., and Franklin, J. (2008). *The elements of statistical learning: data mining, inference and prediction.* Springer.

Hastie, T. J. and Tibshirani, R. J. (1990). *Generalized additive models*, Volume 43. CRC Press.

Higuchi, T. (2001). Self-organizing time series model. In *Sequential Monte Carlo Methods in Practice*, 429–444. Springer.

Ivanov, R. (2012). Real-time GPS track simplification algorithm for outdoor navigation of visually impaired. *Journal of Network and Computer Applications*, *35*(5), 1559–1567.

Jaynes, E. T. (1983). Papers On Probability. *Statistics and Statistical Physics*.

Jeffries, H. (1961). Theory of probability.

Judd, K. L. (1998). *Numerical methods in economics.* MIT press.

Kalman, R. E. *et al.* (1960). A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, *82*(1), 35–45.

Kantas, N., Doucet, A., Singh, S., and Maciejowski, J. (2009). An overview of sequential Monte Carlo methods for parameter estimation. In *in General State-Space Models, in IFAC System Identification, no. Ml.* Citeseer.

Kijima, M. (1997). *Markov processes for stochastic modeling*, Volume 6. CRC Press.

Kim, Y.-J. and Gu, C. (2004). Smoothing spline Gaussian regression: more scalable computation via efficient approximation. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *66*(2), 337–356.

Kimeldorf, G. and Wahba, G. (1971). Some results on Tchebycheffian spline functions. *Journal of mathematical analysis and applications*, *33*(1), 82–95.

Kimeldorf, G. S. and Wahba, G. (1970). A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *The Annals of Mathematical Statistics*, *41*(2), 495–502.

Kitagawa, G. (1996). Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of computational and graphical statistics*, *5*(1), 1–25.

Kitagawa, G. (1998). A self-organizing state-space model. *Journal of the American Statistical Association*, 1203–1215.

Lawson, C. T., Ravi, S., and Hwang, J.-H. (2011). Compression and Mining of GPS Trace Data: New Techniques and Applications. Technical report, Technical Report. Region II University Transportation Research Center.

Lindley, D. V. and Smith, A. F. (1972). Bayes estimates for the linear model. *Journal of the Royal Statistical Society. Series B (Methodological)*, 1–41.

Liu, J. and West, M. (2001). Combined parameter and state estimation in simulation-based filtering. In *Sequential Monte Carlo methods in practice*, 197–223. Springer.

Liu, Z. and Guo, W. (2010). Data driven adaptive spline smoothing. *Statistica Sinica*, *20*(1143-1163).

MacKay, D. J. (2003). *Information theory, inference and learning algorithms*. Cambridge university press.

Meratnia, N. and Rolf, A. (2004). Spatiotemporal compression techniques for moving point objects. In *Advances in Database Technology-EDBT 2004*, 765–782. Springer.

Nason, G. (2010). *Wavelet methods in statistics with R*. Springer Science & Business Media.

Oussalah, M. and De Schutter, J. (2001). Adaptive Kalman filter for noise identification. *PROCEEDINGS OF THE INTERNATIONAL SEMINAR ON MODAL ANALYSIS*, *3*, 1225–1232.

Petris, G., Petrone, S., and Campagnoli, P. (2009). Dynamic linear models. *Dynamic Linear Models with R*, 31–84.

Polson, N. G., Stroud, J. R., and Müller, P. (2008). Practical filtering with sequential parameter learning. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *70*(2), 413–428.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. The MIT Press.

Ristic, B., Arulampalam, S., and Gordon, N. (2004). Beyond the Kalman filter. *IEEE Aerospace and Electronic Systems Magazine*, *19*(7), 37–38.

Robert, C. P. (2004). *Monte carlo methods*. Wiley Online Library.

Schwarz, K.-P. (2012). *Geodesy Beyond 2000: The Challenges of the First Decade, IAG General Assembly Birmingham, July 19–30, 1999*, Volume 121. Springer Science & Business Media.

Sealfon, C., Verde, L., and Jimenez, R. (2005). Smoothing spline primordial power spectrum reconstruction. *Physical Review D*, *72*(10), 103520.

Sherman, J. and Morrison, W. J. (1950). Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, *21*(1), 124–127.

Silverman, B. W. (1985). Some aspects of the spline smoothing approach to non-parametric regression curve fitting. *Journal of the Royal Statistical Society. Series B (Methodological)*, 1–52.

Simon, D. (2004). Data smoothing and interpolation using eighth-order algebraic splines. *Signal Processing, IEEE Transactions on*, *52*(4), 1136–1144.

Smith, A. F. (1973). A general Bayesian linear model. *Journal of the Royal Statistical Society. Series B (Methodological)*, 67–75.

Stroud, J. R. and Bengtsson, T. (2007). Sequential state and variance estimation within the ensemble Kalman filter. *Monthly Weather Review*, *135*(9), 3194–3208.

Trevor Hastie, Robert Tibshirani, J. F. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Second Edition*. Springer-Verlag.

Tusell, F. (2011). Kalman Filtering in R. *Journal of Statistical Software*, *39*(2).

Wahba, G. (1985). A comparison of GCV and GML for choosing the smoothing parameter in the generalized spline smoothing problem. *The Annals of Statistics*, 1378–1402.

Wahba, G. (1990). *Spline models for observational data*, Volume 59. Siam.

Wahba, G. and Wold, S. (1975). A completely automatic French curve: Fitting spline functions by cross validation. *Communications in Statistics-Theory and Methods*, *4*(1), 1–17.

Wang, Y. (1998). Smoothing spline models with correlated random errors. *Journal of the American Statistical Association*, *93*(441), 341–348.

West, M. (1993). Mixture models, Monte Carlo, Bayesian updating, and dynamic models. *Computing Science and Statistics*, 325–325.

Wood, S. N. (2000). Modelling and smoothing parameter estimation with multiple quadratic penalties. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *62*(2), 413–428.

Woodbury, M. A. (1950). Inverting modified matrices. *Memorandum report*, *42*(106), 336.

Yang, K. and Sukkarieh, S. (2010). An analytical continuous-curvature path-smoothing algorithm. *Robotics, IEEE Transactions on*, *26*(3), 561–568.

Yao, F., Müller, H.-G., Wang, J.-L., *et al.* (2005). Functional linear regression analysis for longitudinal data. *The Annals of Statistics*, *33*(6), 2873–2903.

Ying, J. J.-C., Lee, W.-C., Weng, T.-C., and Tseng, V. S. (2011). Semantic trajectory mining for location prediction. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 34–43. ACM.

Yoshimoto, F., Harada, T., and Yoshimoto, Y. (2003). Data fitting with a spline using a real-coded genetic algorithm. *Computer-Aided Design*, *35*(8), 751–760.

Yu, B., Kim, S. H., Bailey, T., and Gamboa, R. (2004). Curve-based representation of moving object trajectories. In *Database Engineering and Applications Symposium, 2004. IDEAS'04. Proceedings. International*, 419–425. IEEE.

Zamani, M. (2010). A simple 2-D interpolation model for analysis of nonlinear data. *Natural Science*, *2*(6), 641–645.

Zhang, K., Guo, J.-X., and Gao, X.-S. (2013). Cubic spline trajectory generation with axis jerk and tracking error constraints. *International Journal of Precision Engineering and Manufacturing*, *14*(7), 1141–1146.