Neural Decoding in Motor Cortex

Chenchen Zhou

March 7, 2013

ABSTRACT

In this project, I conduct neural decoding in motor cortex using three important inference methods in dynamic systems. The first method is classical Kalman Filter(KF). The second method is Inhomogeneous Poisson Process(IPP). The third method is Hidden Markov Model(HMM).

I apply statistical analysis on data of observed neural activity from brain cortex and data of external behaviors. I use these three methods to identify three different models to understand the brain mechanism and make inferences about the external behaviors based on those models.

I analysis the overall performances of the three methods(KF, IPP, HMM). In addition, I compare performances of KF,IPP,HMM. In KF model, I compare Kalman Filter Algorithm to Sequential Monte Carlo Algorithm with different sample size. In IPP model, I compare Point Process Algorithm with Sequential Monte Carlo Algorithm with different sample size.

Keywords: Kalman Filter, Inhomogeneous Poisson Process, Hidden Markov Model, Sequential Monte Carlo, inference, decoding

CONTENTS

1. INTRODUCTION

The topic which uses observations from a time-series to estimate the underlying process is very valuable. Among them,Hidden Markov Models(HMM) are especially known for their very wide application in temporal pattern recognition such as speech, handwriting, gesture recognition and in steering, robotics, engineering design,etc. In this project, I study neural decoding in Motor Cortex by using this model.

Let $x_1, x_2, ...x_t, x_{t+1}, ...$form a Markov process and these states are not directly visible. However,the observations $y_1, y_2, ...y_t, y_{t+1}, ...$,are visible. And for all t, the observation y_t is conditionally independent of x_τ, for $\tau \neq t$. This model is called Hidden Markov Model.
To be more clear, A Hidden Markov Model must satisfy:

$$P(x_t|x_1, x_2, ...x_{t-1}) = P(x_t|x_{t-1}) \qquad and \qquad P(y_t|y_1, y_2, ...y_{t-1}, x_t) = P(y_t|x_t)$$

Our goal is to use those observations $\{y_t\}$ to estimate those hidden states $\{x_t\}$. In this project I use filtering to make inferences. In this case the goal is to estimate a state x_t using observations up to t. In other words, the goal is to find

$$\widehat{x_t} = argmax_{x_t} P(x_t|y_1, y_2, ...y_t) \qquad or \qquad \widehat{x_t} = E(x_t|y_1, y_2, ...y_t)$$

Before estimating, we make assumptions to the probability models. With respect to different assumptions, we have different models. We first use training data to identify those probability models. Based on different models, we apply different algorithms to estimate the hidden states in test data.

2. **DATA DESCRIPTION**

In this project, I use observed neural activity from brain cortex in research animals to infer the unknown related external behaviors. To be more specific,I use two different datasets. One contains training data which is used to identify model, the other contains testing data to make inferences. Each set has two variables: kin and rate. 'kin' is the kinematic state of the hand which includes x-position, y-position, x-velocity, and y-velocity. This data is that we are interested and need us to infer. 'rate' is the spiking rates of 42 neurons, where the rate at each time is the number of spikes within 70ms. This data is observable and we rely on them to make inference. Use the kin and rate in the training data to identify the model. In the test data, use the identified model and rate to infer kin, and compare the estimate with the true kin. In this project, the comparison only focus on the hand positions, which are the first two components in kin.

To be more clear, in this project:

$X = (x_1, x_2, ...x_T) : x_t \in R^4$, kinematics (x-pos, y-pos,x-velocity,y-velocity). Those data are in kin, and they are the hidden states.

$Y = (y_1, y_2, ...y_T) : y_t \in R^{42}$ firing rates of 42 neurons, Those data are in rate, and they are the observed data.

3. COMPUTATIONAL METHODS

I have introduced the HMM in introduction. To illustrate methods I used in this project, I restate the HMM and our goal.

$x_1, x_2, ...x_t, x_{t+1}, ...$form a Markov process and these states are not directly visible. However,the observations $y_1, y_2, ...y_t, y_{t+1}, ...$,are visible. And for all t, the observation y_t is conditionally independent of x_τ, for $\tau \neq t$. This model is called Hidden Markov Model.
So a Hidden Markov Model must satisfy:

$$P(x_t|x_1, x_2, ...x_{t-1}) = P(x_t|x_{t-1}) \qquad and \qquad P(y_t|y_1, y_2, ...y_{t-1}, x_t) = P(y_t|x_t)$$

Our goal is to use those observations y_t to estimate those hidden states x_t. In this project I use filtering to make inferences. In this case the goal is to estimate a state x_t using observations up to t. In other words, the goal is to find

$$\widehat{x_t} = argmax_{x_t} P(x_t|y_1, y_2, ...y_t) \qquad or \qquad \widehat{x_t} = E(x_t|y_1, y_2, ...y_t)$$

Kalman Filter Model:

In this method, the model we assume is that all random variables have Gaussian distributions and they are linearly related. To be more clear, I define the notation as follows:

$x_k \in R^d$: internal state at kth frame (hidden random variable, kin(k,:) in this project).

$y_k \in R^c$: measurement at kth frame (observable random variable, rate(k,:) in this project).

$X_k = [x_1, x_2, ..., x_k]^T$: history up to time step k

$Y_k = [y_1, y_2, ...y_k]^T$: history up to time step k

System equation(prior model):

$x_{k+1} = A_k x_k + w_k \qquad w_k \in N(0, W_k) \qquad k = 1, 2... \quad A_k \in R^{d\times d}, \quad W_k \in R^{d\times d}$

This is same as:

$$x_{k+1}|x_k \sim N(A_k x_k, W_k)$$

Measurement equation(likely model):

$y_k = H_k x_k + q_k \qquad q_k \in N(0, Q_k) \qquad k = 1, 2... \quad H_k \in R^{c\times d}, \quad Q_k \in R^{c\times c}$

This is same as

$$y_k|x_k \sim N(H_k x_k, Q_k)$$

To make computation much easier, the model also assume that A_k, W_k, H_k, Q_k are constants. So there are only four matrix to be identified in this model. The A, H, W, Q can be estimated by maximizing the joint probability $p(X_m, Y_m)$.Then we can get closed form to compute A,H,W,Q.

After identify model, two computational methods are applied to make inference. One is Kalman Filter Algorithm, the other is Sequential Monte Carlo Algorithm. Both methods are based on the prediction equations and update equation.
Prediction equation:

$$\begin{aligned} f(x_t|y_1, y_2, ...y_{t-1}) &= \int_{x_{t-1}} f(x_t, x_{t-1}|y_1, y_2, ...y_{t-1}) dx_{t-1} \\ &= \int_{x_{t-1}} f(x_t|x_{t-1}) f(x_{t-1}|y_1, y_2, ...y_{t-1}) dx_{t-1} \end{aligned}$$

Update equation:

$$\begin{aligned} f(x_t|y_1, y_2, ...y_t) &= \frac{f(x_t, y_1, y_2, ...y_t)}{f(y_1, y_2, ...y_t)} \\ &= \frac{f(y_t|x_t) f(x_t|y_1, y_2, ...y_{t-1})}{f(y_t|y_1, y_2, ...y_{t-1})} \end{aligned}$$

To estimate, I apply two algorithms.

Kalman Filter Algorithm:
Both likelihood and prior are Gaussian,this greatly simplify the computation as a Normal distribution is only determined by two parameters(mean and covariance). What's more, the two estimators $\widehat{x_t} = argmax_{x_t} P(x_t|y_1, y_2, ...y_t)$ and $\widehat{x_t} = E(x_t|y_1, y_2, ...y_t)$ are the same. So when estimate the x_k, Kalman Filter Algorithms doesn't have to generate samples.
Kalman Filter Algorithm have two steps. First is time update, in this step, the

algorithm predicts x_t by using $y_1, y_2, ..y_{t-1}$, the second step is measurement update, in this step, the algorithm predicts x_t corporate with t_{th} observation y_t. So the measurement step is like a corrector.

Sequential Monte Carlo Algorithm:
The idea of Sequential Monte Carlo Algorithm is to use Monte Carlo idea and generates a large number of samples from the posterior $f(x_t|y_1, y_2, ..y_t)$ at every time t to estimate $\widehat{x_t} = E(x_t|y_1, y_2, ...y_t)$.
This method is very useful when the system is not linear. Because in that case, writing an general form to compute posterior means at every steps is very difficult and always impossible.
To be more clear,the method is as follows:
if state vectors:$x_t \in R^d$, $y_t \in R^c$:
State equation(Prior): $x_{t+1} = F(x_t, w_t)$.
Observation equation(likelihood): $y_t = G(x_t, q_t)$.
In this algorithm, there are also two steps:
Prediction step:
This step uses samples from the posterior density at time t to generate samples from the prediction density $f(x_{t+1}|y_1, ..., y_t)$. As,

$$f(x_{t+1}|y_1, y_2, ...y_t) = \int_{x_t} f(x_{t+1}|x_t) f(x_t|y_1, y_2, ...y_t) dx_t$$

And the fact that:
If

$$x_t^i \sim f(x_t|y_1, ..., y_t)$$

and

$$\widetilde{x_{t+1}^i} \sim f(x_{t+1}|x_t^i)$$

, then

$$\widetilde{x_{t+1}^i} \sim f(x_{t+1}|y_1, ..., y_t)$$

So in this step, we can generate a sample set which is from $f(x_{t+1}|y_1, ..., y_t)$.

Update step:
This step uses the prediction set to generate samples from the posterior den-

sity $f(x_{t+1}|y_1, ..., y_{t+1})$ to estimate

$$\begin{aligned}
\widehat{x_{t+1}} &= E_f(x_{t+1}|y_1, y_2, ...y_{t+1}) \\
&= \int x_{t+1} f(x_{t+1}|y_1, y_2, ...y_{t+1}) dx_{t+1} \\
&= \int x_{t+1} \frac{f(x_{t+1}, y_1, y_2, ...y_{t+1})}{f(y_1, y_2, ...y_{t+1})} dx_{t+1} \\
&= \int x_{t+1} \frac{f(y_{t+1}|x_{t+1}) f(x_{t+1}|y_1, y_2, ...y_t)}{f(y_{t+1}|y_1, y_2, ...y_t)} dx_{t+1}
\end{aligned}$$

So I use the samples set which are generated in the first step and use Monte Carlo Method, we can get estimator.

Inhomogeneous Poisson Process Method(IPP):

In this method, the model we assume is different from that in Kalman Filter method. The state equation in IPP is still Normal Distribution, the difference is the observation equation. In this method, for y_k we assume that we assume a generalized linear model (GLM) with an inhomogeneous Poisson process conditioned on x_k. That is,

$$y_{k,c} \sim Poisson(\lambda_{k,c})$$

where

$$\lambda_{k,c} = \mu_c + \alpha_c^T x_k$$

We maximize the likelihood to identify this model. When maximizing the likelihood, I use Newton-Raphson method.
To approximate posterior we use two different algorithm. To approximate the posterior $f(x_t|y_1, y_2, ..y_t)$, I apply two algorithms. One is Sequential Monte Carlo (SMC). The other algorithm is Point Process Filter.
Point Process Filter
Point Process Filter is based on Laplace approximation by approximate the posterior at each time using a Gaussian distribution.

Hidden Markov Method(HMM):

Hidden Markov models (HMM) are appropriate for describing time-series data that are produced from a system that transitions in a random, Markovian manner between some number of states.In particular, an HMM is described by two random variables at every point in time t: the hidden state q_t and the observed emission y_t. $t = 1, ...T$.

To be more clear:
In this method:

$q_t \in 1, 2, ...N \qquad y_t \in 1, 2...K$

$P(q_t|q_1, q_2, ...q_{t-1}) = P(q_t|q_{t-1}) \quad and \quad P(y_t|y_1, y_2, ...y_{t-1}, q_1, q_2, ...q_t) = P(y_t|q_t)$

And transition and observation probabilities are independent of time: for all $t, s \in 1, ..., T$

$$\alpha_{nm} = p(q_t = m|q_{t-1} = n) = p(q_s = m|q_{s-1} = n)$$
$$\eta_{nk} = p(y_t = k|q_t = n) = p(y_s = k|q_s = n)$$

Based on this assumed model, we use maximum likelihood to estimate the hidden state, that is:

$$\hat{Q} = argmax_Q p(Q|Y) = argmax_Q p(Q, Y)$$

We can maximize the likelihood by dynamic programming.

$$p(Q, Y) = p(y_T|q_T)p(q_T|q_{T-1})....p(y_3|q_2)p(y_2|q_2)p(q_2|q_1)p(y_1|q_1)p(q_1)$$

The idea is: Given q_2, optimal q_1 can be determined; Given q_3, optimal q_2 can be determined;Given q_T, optimal q_{T-1} can be determined; Finally, find which value of q_T is optimal.

In this project:

$X = (x_1, x_2, ...x_T) : x_t \in R^2 \quad kinematics(x - pos, y - pos).$

$Y = (y_1, y_2, ...y_T) : y_t \in R^{42} \quad firingratesof42neurons$

To identify the model, as in HMM the hidden state Q is not observable, so the parameters can be identified by maximizing log-likelihood

$$logp(Y|\theta)$$

we can apply Exception Maximization(EM) algorithm to estimate parameters.
1. Expectation Step: Compute

$$Q(\theta|\theta_i, x_0) = E[logf(x_0, x_m|\theta)|\theta_i, x_0]$$

2. Maximization Step: Set

$$\theta_{i+1} = argmax_\theta Q(\theta|\theta_i, x_0)$$

where $x_0 = Y, x_m = Q$ In estimating, we estimate kinematics:

$$\hat{X} = argmax_X P(X, Y)$$

Simple case: x_t is one-dimensional component of the kinematics(e.g. x-pos), then we can discretize x_i to N values; that is,

$$x_t = r_n, \qquad n \in 1, 2...N$$

$y_t = (y_t^1, y_t^2, ...y_t^{42})$is firing rate of all neurons. They are all nonnegative integers. For each neuron c, we estimate

$$p(y_t^c = k | x_t = r_n)$$

We assume conditional independence of all neurons. That is,

$$p(y_t^1, y_t^2, ...y_t^{42} | x_t) = p(y_t^1 | x_t) p(y_t^2 | x_t)...p(y_t^{42} | x_t)$$

From above assumption, we can maximize the likelihood by dynamic programming,thus to estimate hidden states.

4. **RESULTS**

(a)

inference methods	accuracy in x-position	accuracy in y-position
Classical Kalman Filter	0.60812	0.85341
SMC(n=20)	0.35045	0.80809
SMC(n=50)	0.44426	0.84713
SMC(n=100)	0.55372	0.84671
SMC(n=500)	0.59422	0.85456

Tab. 4.1: Accuracy of Kalman Filter Method with two different inference methods(R^2 error)

inference methods	accuracy in x-position	accuracy in y-position
Point Process Filter	0.55976	0.81327
SMC(n=20)	0.10695	0.78126
SMC(n=50)	0.39838	0.81404
SMC(n=100)	0.41849	0.81469
SMC(n=500)	0.51919	0.8175

Tab. 4.2: Accuracy of Inhomogeneous Poisson Method with two different inference methods(R^2 error)

accuracy in x-position	accuracy in y-position
0.2186	0.5228

Tab. 4.3: Accuracy of Hidden Markov Model Method (R^2 error)

(b) **KF,IPP,HMM**

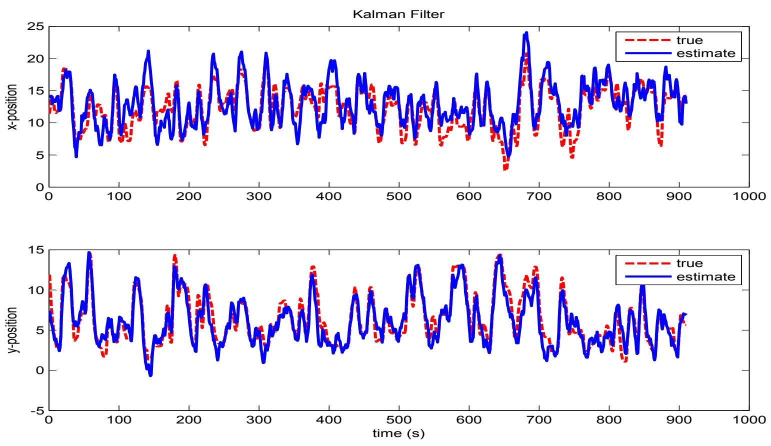

Fig. 4.1: Decoding accuracy of Kalman fiter in KF model

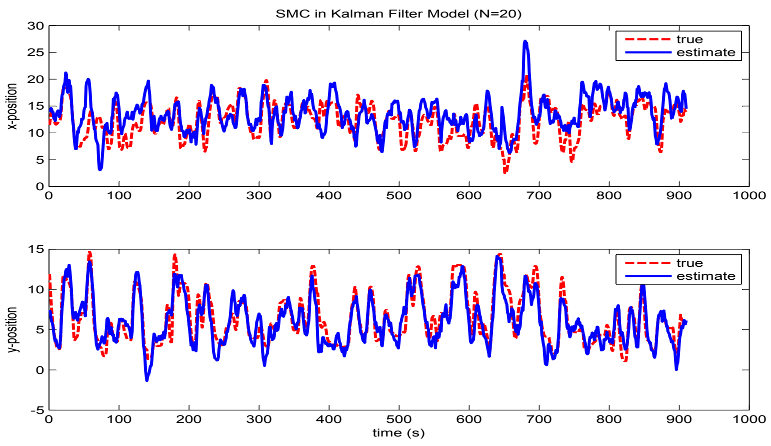

Fig. 4.2: Decoding accuracy of SMC in KF model(n=20)

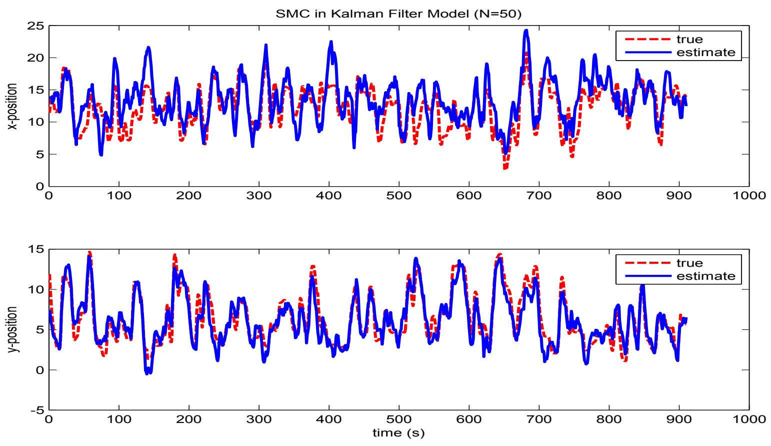

Fig. 4.3: Decoding accuracy of SMC in KF model(n=50)

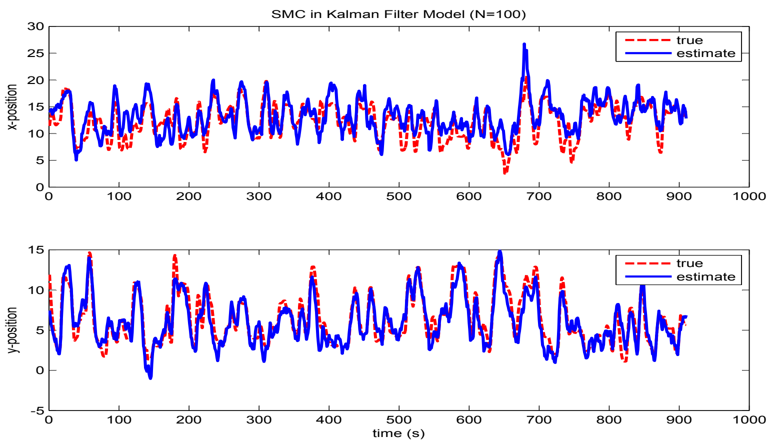

Fig. 4.4: Decoding accuracy of SMC in KF model(n=100)

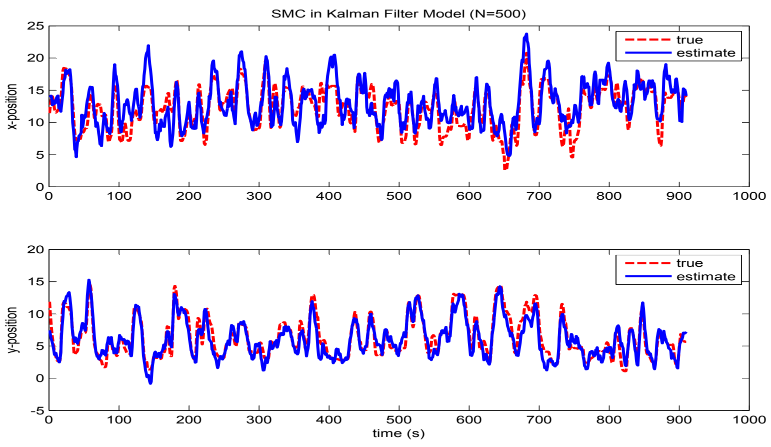

Fig. 4.5: Decoding accuracy of SMC in KF model(n=500)

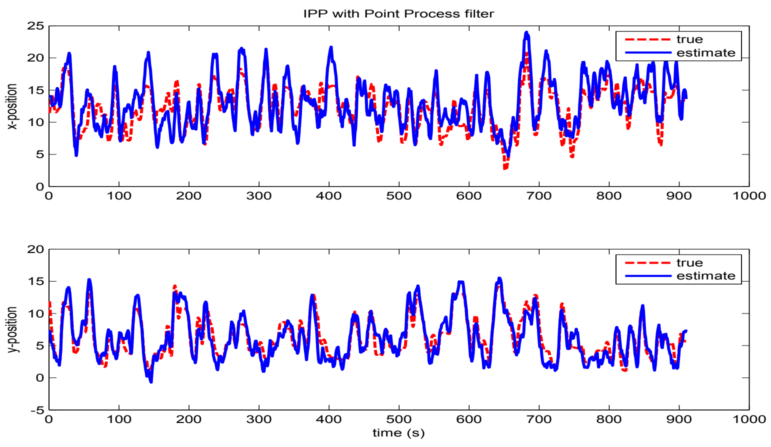

Fig. 4.6: Decoding accuracy(position) of Point Process filter in IPP model

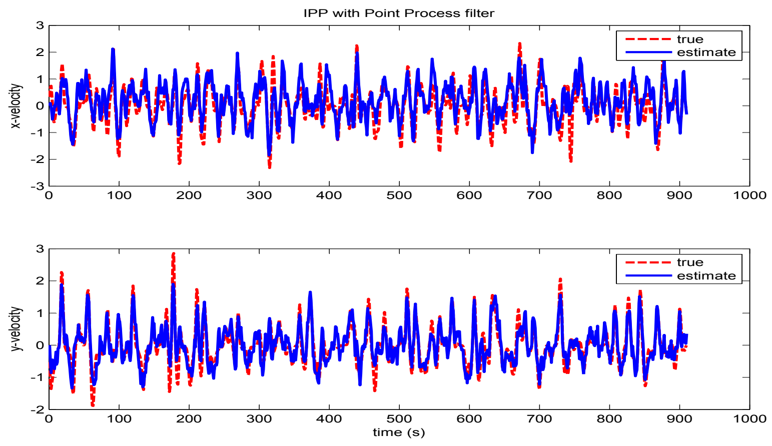

Fig. 4.7: Decoding accuracy(velocity) of Point Process filter in IPP model

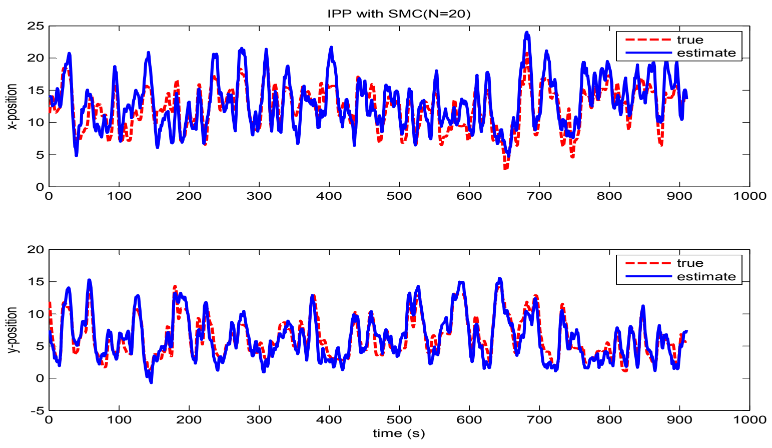

Fig. 4.8: Decoding accuracy(position) of SMC in IPP model(n=20)

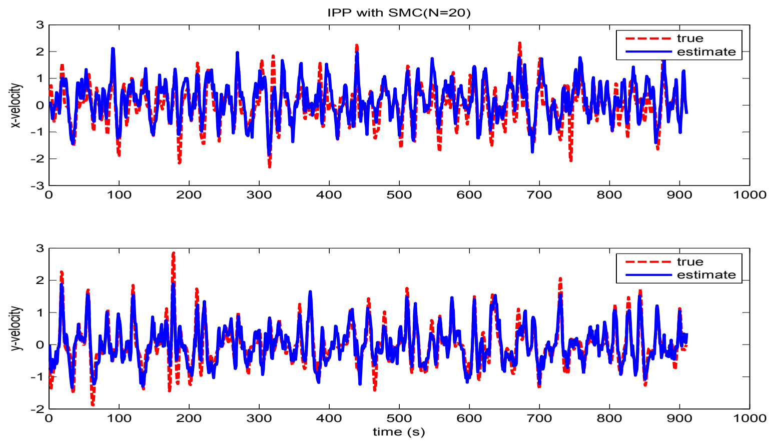

Fig. 4.9: Decoding accuracy(velocity) of SMC in IPP model(n=20)

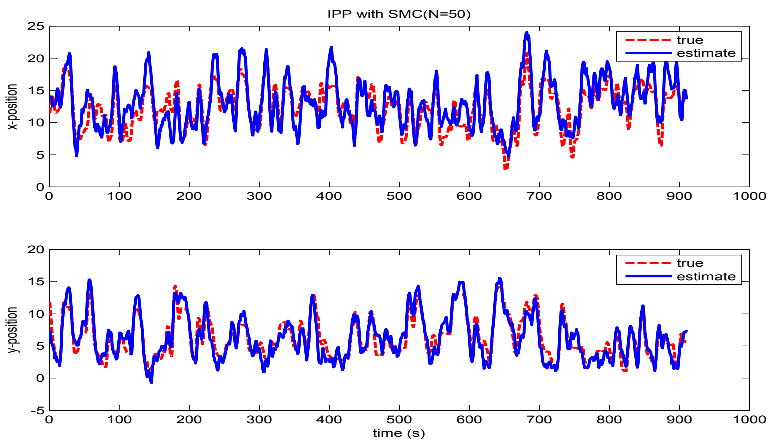

Fig. 4.10: Decoding accuracy(position) of SMC in IPP model(n=50)

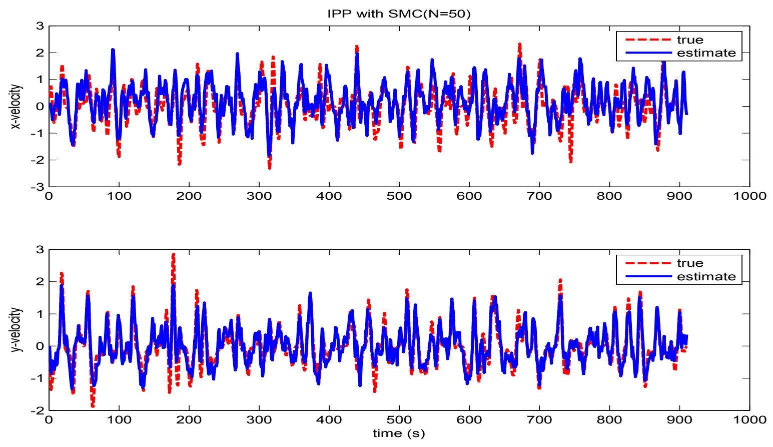

Fig. 4.11: Decoding accuracy(velocity) of SMC in IPP model(n=50)

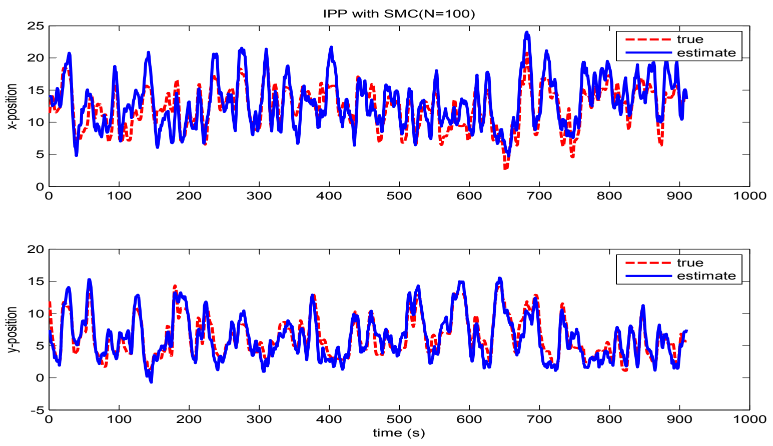

Fig. 4.12: Decoding accuracy(position) of SMC in IPP model(n=100)

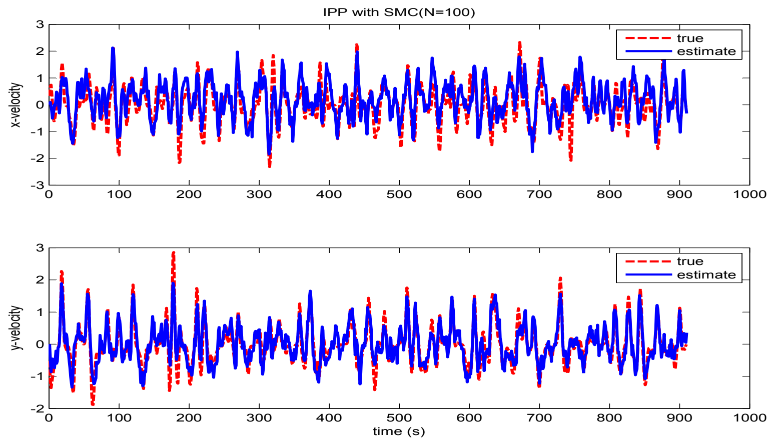

Fig. 4.13: Decoding accuracy(velocity) of SMC in IPP model(n=100)

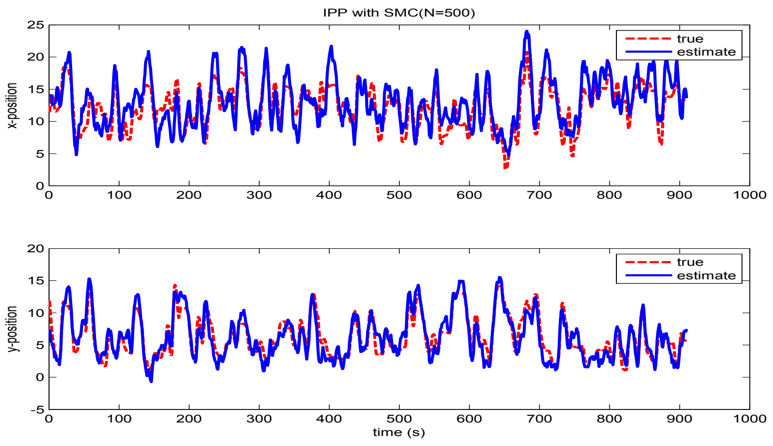

Fig. 4.14: Decoding accuracy(position) of SMC in IPP model(n=500)

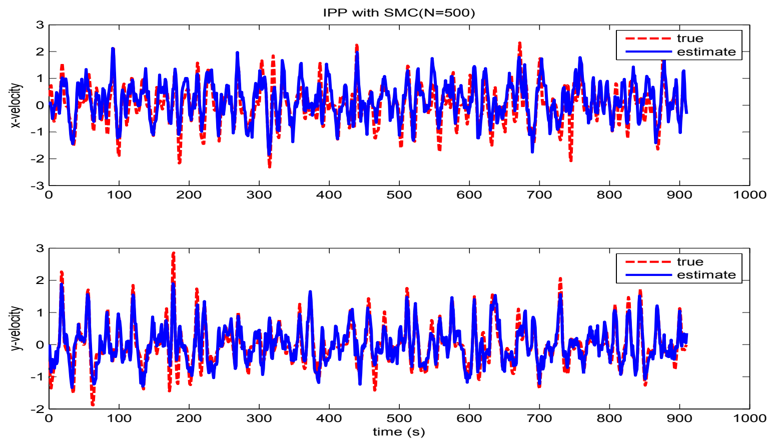

Fig. 4.15: Decoding accuracy(velocity) of SMC in IPP model(n=500)

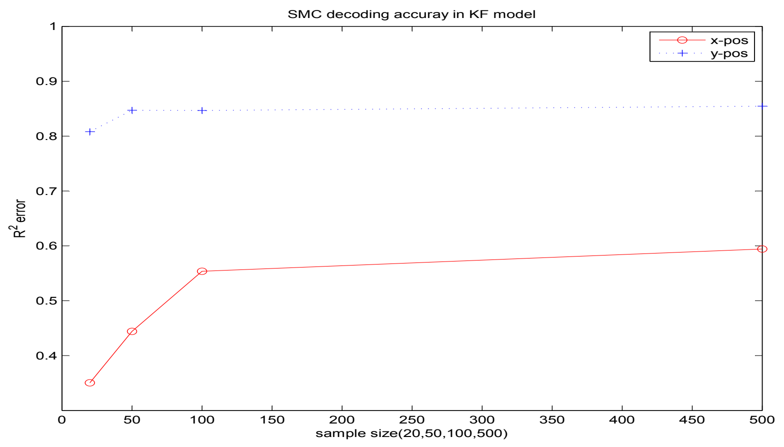

Fig. 4.16: SMC decoding accuracy in KF

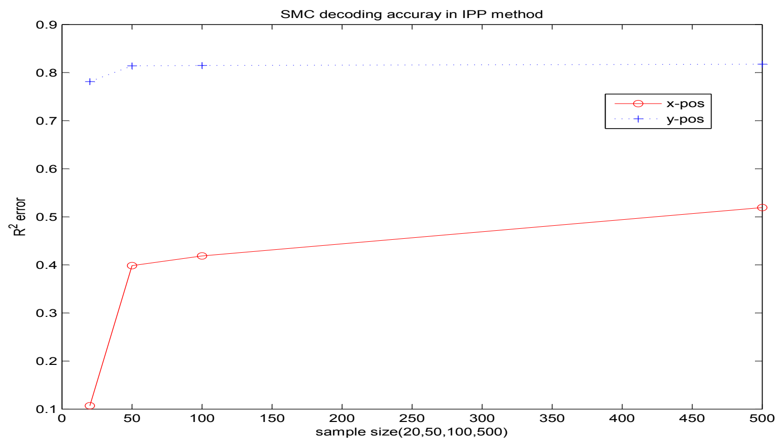

Fig. 4.17: SMC decoding accuracy in IPP

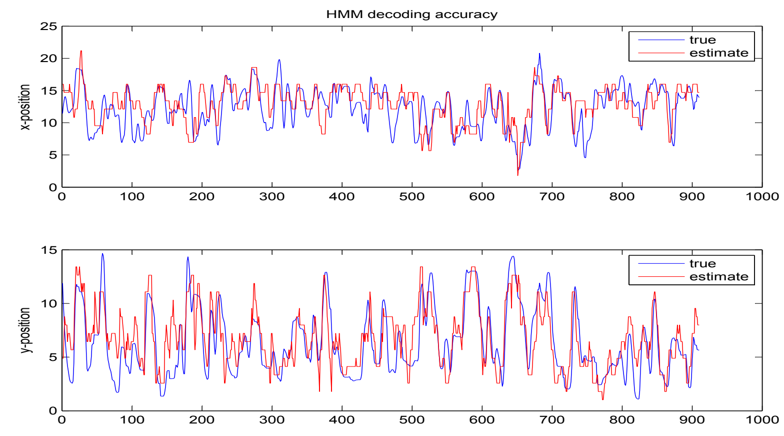

Fig. 4.18: Decoding accuracy of HMM

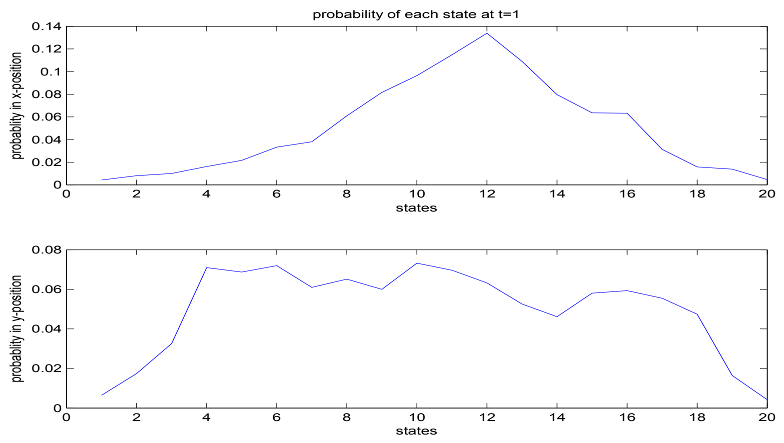

Fig. 4.19: Probability of kin at initial time $P((q_1) = n)$

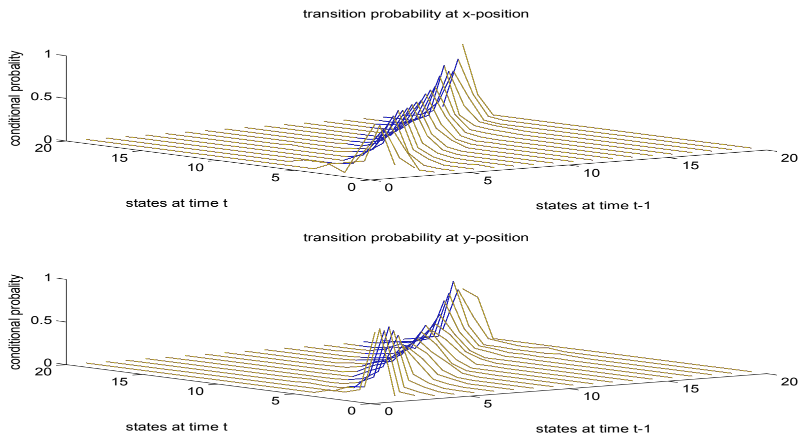

Fig. 4.20: transition probability $(P(q_t|q_{t-1}))$

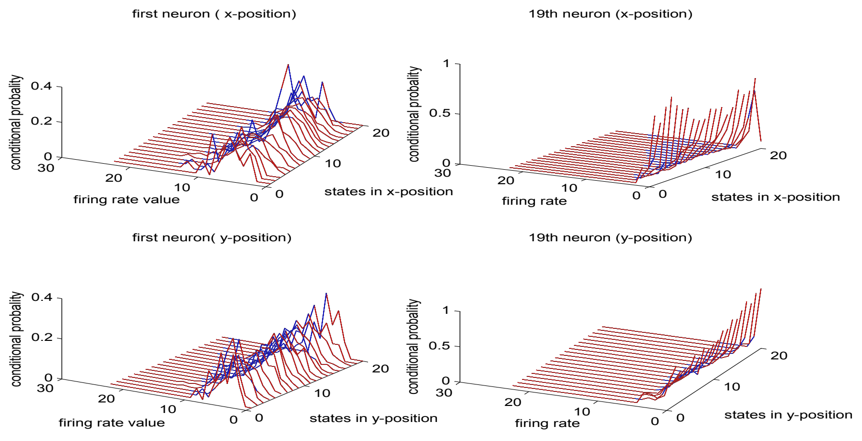

Fig. 4.21: conditional probability of firing rate $(P(y_t|q_t))$ of two representative neurons

5. DISCUSSIONS AND SUMMARY

Discussion:

(1)From table4.1, we can see that in KF model, Kalman filter has a very good estimation. The decoding accuracy of SMC is improving as the sample size is increasing. When the sample is 20,50,100, the accuracy of SMC is not as good as KF. When the sample reaches 500,SMC and KF have a similar decoding accuracy.
(2)From table4.2, we can see that in IPP model, Point Process filter has a good estimation. The decoding accuracy of SMC is improving as the sample size is increasing. When the sample is 20,50,100, the accuracy of SMC is not as good as KF. When the sample reaches 500,SMC and Point Process filter have a similar decoding accuracy.
(3)From table4.3, we can see that in HMM model, the decoding accuracy is very low which compare to KF and IPP.
(4)From fig4.16 and fig4.17, we can see that the decoding accuracy of SMC is increasing if we increase the sample size.
(5)KF model has best performance, and the second one is IPP method, HMM method is not satisfactory.
(6)From the fig4.20 and fig4.21. The probability from one state to another concentrate along the diagonal band, and the probability of firing rate condition on hand movements concentrate at one side. That means the probability from one state to another state is not fair. And for each neuron, the firing rate is concentrated along small interval respect to specific hand movement.
(7)Among all methods, the performance on y position is much better than x-position.
(8)As for the efficiency of those methods, KF takes a very short time to make inference; however, the SMC takes much more time. The decoding time may reach 50 seconds and more if the sample size is 500. For IPP model, point process filter has a similar time with KF filter in decoding, but it takes a longer time to identify the IPP model than KF model.For HMM method, the dynamic system make the estimate very efficient.

Summary:
In this project, I compare the three inference methods(KF,IPP,HMM) from

the view of accuracy and efficiency.I get the following summary.
(1) The KF model is the best one in this project, and IPP is also good, but not as good as KF. The HMM model is very bad with respect to this data.
(2) In KF, the classical KF filter algorithm is very efficient, it reaches a high accuracy in a very short time. In contrast, SMC in KF takes a long time to get the result. And to reach the same accuracy, SMC must increase the sample size which means it cost much computation.
(3) In IPP , the point process filter algorithm is more efficient than SMC and have a good . In contrast, SMC in IPP takes a long time to get the result. And to reach the same accuracy, SMC must increase the sample size which means it cost much computation.
(4)I do not suggest to use HMM to model neural decoding of hand movement, as it has a very low accuracy.

In conclude, KF model is best fit this data and classical KF algorithm enjoys a high accuracy and high efficiency.