Lab 6 - Particle Filter

Madhumita Krishnan

11/20/2018

1 Introduction

This report deals with the design and implementation of a Particle Filter. The Particle Filter Method is a Monte Carlo technique for solution of state estimation problem. It is a hypothesis tracker that approximates the filtered posterior distribution by a set of weighted particles.

The principal advantage of this method is that it does not rely on any local linearisation techniques nor any functional approximations. It can be used to model Non-Linear and Non-Gaussian systems.

The system in this report consists of a 1D position, moving on a line. It follows a motion pattern where the position zig-zags back and forth on a line.

2 Method

The method for implementation of the Particle Filter is outlined below by a predict-update cycle.

1. Each particle m is propagated through the state transition equation:

$$\{x_t^{(m)} = f(x_{t-1}^{(m)}, a_t^{(m)})\}_{m=1}^{M} \tag{1}$$

where the value $a_t^{(m)}$ represents the dynamic noise from $t-1$ to t. The state variables used for position and velocity are given by the state vector

$$X_t = \begin{bmatrix} x_t \\ \dot{x}_t \end{bmatrix} \tag{2}$$

The state transition equations that predicts future values of position and velocity of a 1D particle is given as follows:

$$f(x_t, a_t) = \begin{bmatrix} x_{t+1} = x_t + \dot{x}_t T \\ \dot{x}_{t+1} = \begin{cases} 2 & \text{if } x_t < -20 \\ \dot{x}_t + |a_t| & \text{if } -20 \leq x_t < 0 \\ \dot{x}_t - |a_t| & \text{if } 0 \leq x_t \leq 20 \\ -2 & \text{if } x_t > 20 \end{cases} \end{bmatrix} \tag{3}$$

2. Using the measurement vector y_t, the weight for each particle is updated:

$$\tilde{w}_t^{(m)} = w_{t-1}^{(m)} \cdot p(y_t | x_t^{(m)}) \tag{4}$$

The measurement vector is obtained from a sensor that measures the total magnetic strength in this problem. $p(y_t|x_t^{(m)})$ is determined by comparing the ideal measurement

of the particle with the actual measurement. The ideal measurement is given by Equation 5. Following this the probability is obainted by Equation 6.

$$g(x_t^{(m)}, 0) = \left[y_t^{(m)} = \frac{1}{\sqrt{2\pi}\sigma_m} \exp(\frac{-(x_t^{(m)} - x_{m1})^2}{2\sigma_m^2}) + \frac{1}{\sqrt{2\pi}\sigma_m} \exp(\frac{-(x_t^{(m)} - x_{m2})^2}{2\sigma_m^2}) \right] \tag{5}$$

$$p(y_t | x_t^{(m)}) = \frac{1}{\sqrt{2\pi}\sigma_n} \exp(\frac{-(y_t^{(m)} - y_t)^2}{2\sigma_n^2}) \tag{6}$$

3. The weights are normalized so they sum up to 1 using the Equation 7.

$$w_t^{(m)} = \frac{\tilde{w}_t^{(m)}}{\sum_{m=1}^{M} \tilde{w}_t^{(m)}} \tag{7}$$

4. The expected value is determined by Equation 8 and it is used as the output of the filter designed.

$$E[x_t] \approx \sum_{m=1}^{M} x_t^{(m)} \cdot w_t^{(m)} \tag{8}$$

5. Resampling is implemented if the effective sample size falls below the threshold value set initially. The effective sample size ESS describes how many particles have an appreciable weight. To determine ESS, the coefficient of variation statistic is first calculated using Equation 9.

$$\text{CV} = \frac{1}{M} \sum_{m=1}^{M} (M \cdot w^{(m)} - 1)^2 \tag{9}$$

The effective sample size can then be calculated as in Equation 10.

$$\text{ESS} = \frac{M}{1 + \text{CV}} \tag{10}$$

The method used to resample in this problem is called select with replacement. The particles with negligible weights are removed and replaced with copies of particles that have large weights.

6. Let $t = t + 1$; iterate.

3 Results

The Particle Filter was implemented from the above method using the values below.

1. $M = 100$
2. $\sigma_a = 0.0625$
3. $\sigma_n = 0.0039$

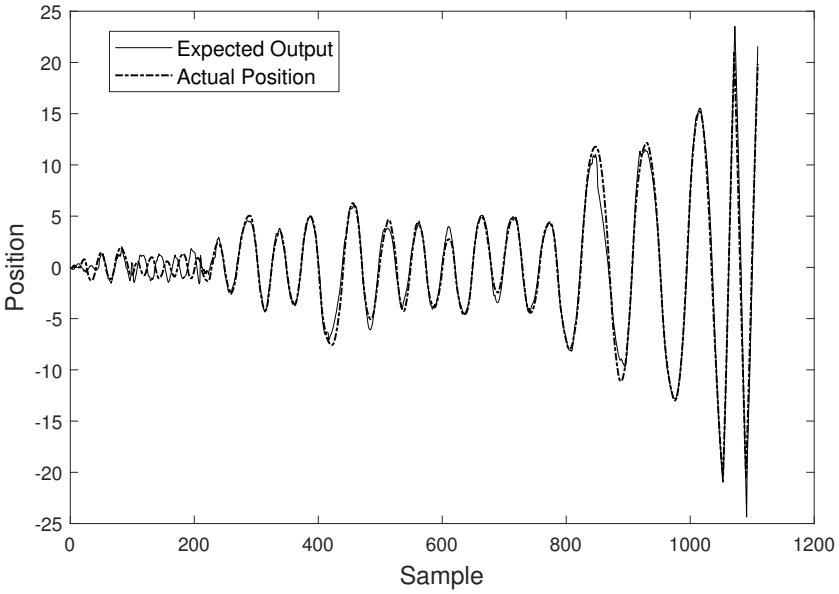


Figure 1: Tracking Output of the Particle Filter

4. $\sigma_m = 4.0$

The Expected Value which was obtained using Equation 8 is plotted with respect to the Samples. The plot obtained is shown in Figure 1. Figure 1 shows us that the Particle Filter output closely tracks the actual position given. A more closer look of the tracking can be seen in Figure 2.

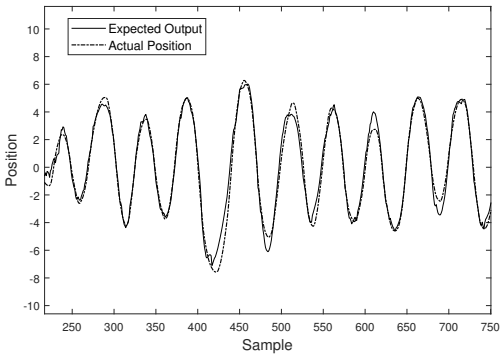


Figure 2: A Closer look at the Filter Tracking

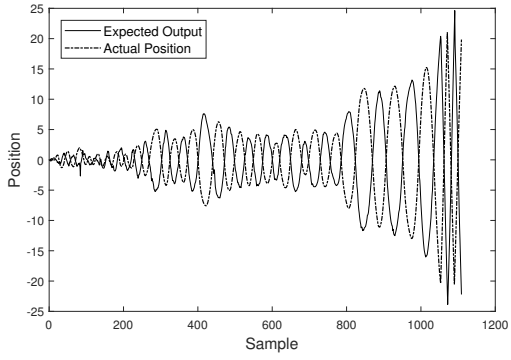
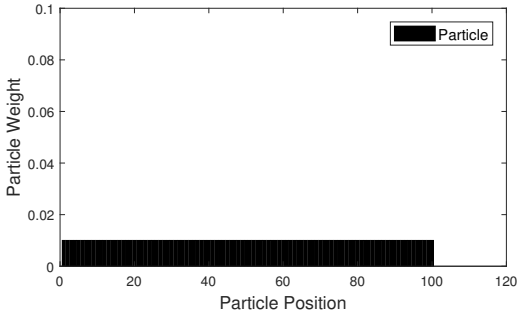
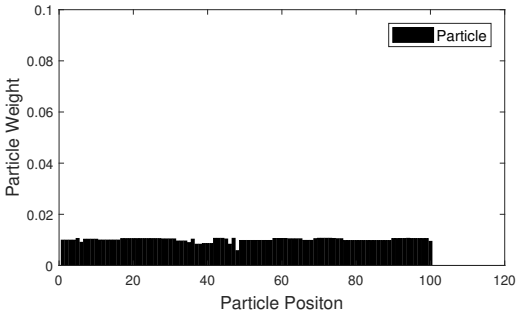
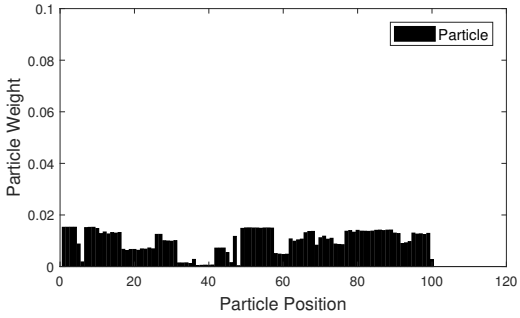


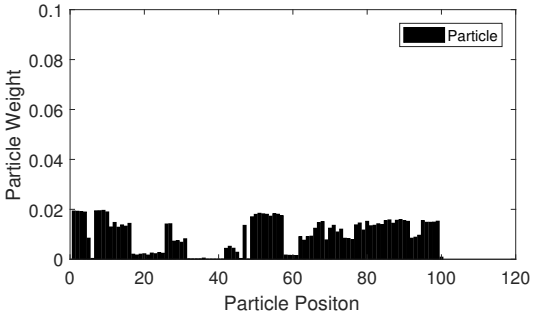
Figure 3: Out of Phase Tracking Output

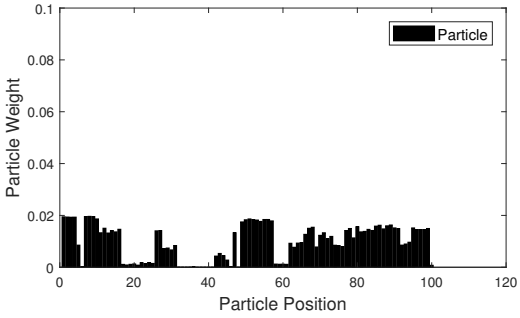
The resampling implementation is shown by plotting the weights of the particle with respect to the position of the particle. Figure 4 shows the steps of resampling. The process of initialization of weights is seen in Figure 4. It can also be seen that as the number of iterations increases the weights in the particles vary and the value of ESS also varies along with it. When the ESS drops below the threshold value the particles are resampled. Figure 5 shows the output of the Filter for which the resampling of particles is shown.

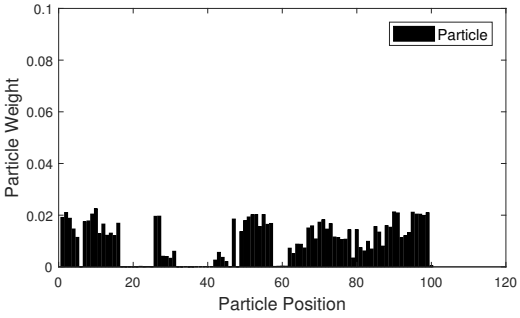

Particle
Particle Weight
Particle Position
0.1
0.08
0.06
0.04
0.02
0
0
20
40
60
80
100
120


Particle
Particle Weight
Particle Positon
0.1
0.08
0.06
0.04
0.02
0
0
20
40
60
80
100
120


Particle
Particle Weight
Particle Position
0.1
0.08
0.06
0.04
0.02
0
0
20
40
60
80
100
120


Particle
Particle Weight
Particle Positon
0.1
0.08
0.06
0.04
0.02
0
0
20
40
60
80
100
120


Particle
Particle Weight
Particle Position
0.1
0.08
0.06
0.04
0.02
0
0
20
40
60
80
100
120


Particle
Particle Weight
Particle Position
0.1
0.08
0.06
0.04
0.02
0
0
20
40
60
80
100
120

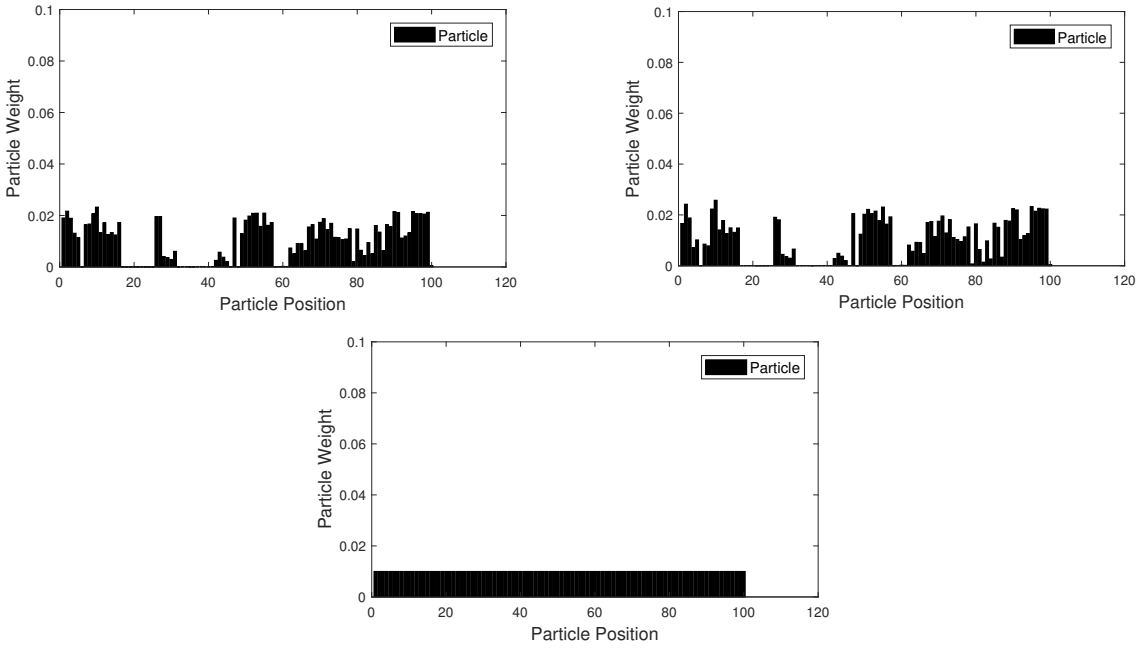


Figure 4: Weights of Particles from Initialization to Resampled Particles Weights

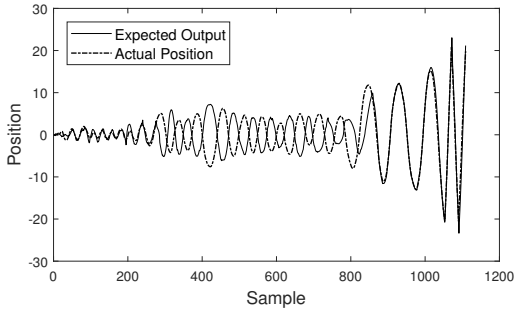


Figure 5: Output of the Particle Filter used
for the Resampling Example

4 Conclusion

Even thought Particle Filter Methods are computationally expensive, their popularity is increasing in real-time applications in diverse fields like Target Tracking and Robotics, Chemical Engineering, Computer Vision and Financial Econometrics. These methods work for any observation model. They also work for high dimensional systems as they are independent of the size of the system. The disadvantage is the lack of diversity. Over time samples tend to congregate. However to solve this problem of particle degradation, algorithms based on improved adaptive resampling are used.

5 Appendix

Code for Particle Filter

```
clc
clear
close all
x=zeros(2,1);
t=1;
T=1;
table=dlmread("magnets-data.txt");
xpos=zeros(2,1);
M=100;

sm=4.0;
sn=0.004;
xm1= -10;
xm2=  10;
actualpos=table(:,1);
actualvel=table(:,2);
sensor=table(:,3);
%State Initialization
xpos=zeros(1,M);
xvel=zeros(1,M);
xposprev=zeros(1,M);
xvelprev=zeros(1,M);

ytM=zeros(1,M);

%Weight initialization
wt=ones(1,M)*1/M;
wtprev=ones(1,M)*1/M;
wtn=ones(1,M)*1/M;

E=zeros(1,length(sensor));
```

```
%Plot Init
rc=0;
sf=15;
plt=0;
X2=1:M;

for t=1:length(sensor)

    for m=1:M

        at= randn*(0.0625);

        xpos(m)=xposprev(m)+(xvelprev(m)*1);
        if xposprev(m) < (-20)
            xvel(m)=2;
        elseif xposprev(m) >= (-20) && xposprev(m) < 0
            xvel(m)=xvelprev(m)+ abs(at);
        elseif xposprev(m) >= 0 && xposprev(m) <= 20
            xvel(m)=xvelprev(m)- abs(at);
        elseif xposprev(m) > 20
            xvel(m)=-2;
        end

        ytM(m)=(1/(sqrt(2*pi)*sm))*exp(-(xpos(m)-xm1)^2/(2*(sm^2)))+(1/(sqrt(2*pi)*sm))*

        p= (1/(sqrt(2*pi)*sn))*exp(-(ytM(m)-sensor(t))^2/(2*(sn^2)));
        wt(m)=wtprev(m)*p;

        wtprev(m)=wt(m);
        xposprev(m)=xpos(m);
        xvelprev(m)=xvel(m);
    end

    sum1=cumsum(wt);

    wtn=wt./sum1(M);
    mul1=(xpos .* wtn);
    sum2=cumsum(mul1);
    %Expected Output
    E(t)=sum2(M);

    mul2=M.*wtn;
    sub1=(mul2-1).^2;
    sum3=cumsum(sub1);
```

```
CV=1/M*(sum3(M));
ESS=M/(1+CV);
if(plt)
    figure(t)
    bar(X2,wtn)

    xlabel('Particle Position')
    ylabel('Particle Weight')
    legend('Particle')
end
%Resampling
if(ESS< 0.5*M)
    rc=rc+1;
    if(rc==sf)
        plt=1;
    end
    Q=cumsum(wtn);
    k=rand(1,M);
    K=sort(k);
    K(1,M+1)=1.0;
    i=1;
    j=1;
    while(i<=M)
        if (K(i)<=Q(j))
            Index(i)=j;
            i=i+1;
        else
            j=j+1;
        end
    end
    p1=1;
    while(p1<=M)
        xpos(p1)=xpos(Index(p1)) ;
        xposprev(p1)=xposprev(Index(p1));
        xvel(p1)=xvel(Index(p1));
        xvelprev(p1)=xvelprev(Index(p1));
        wtprev(p1)=1/M;
        wt(p1)=1/M;
        wtn(p1)=1/M;
        p1=p1+1;

    end
    if(plt)
        figure(t)
        bar(X2,wtn)
```

```
            xlabel('Particle Position')
            ylabel('Particle Weight')
            legend('Particle')
        end
        if(rc==sf+1)
            plt=0;
        end

    end

end

%Plot
X1=1:length(sensor);

figure(1)
plot(X1,E, 'k-');
hold on
plot(X1,actualpos,'k-.','LineWidth',1)
hold off
legend('Expected Output','Actual Position')
xlabel('Sample')
ylabel('Position')
```