Particle Filter for mini project – Screenshots from class

Particle Filter Application

Initialization of Parameters

```
% initialization of all the parameters of the four tank system
clc; clear; close all;
A1 = 28; %(cm^2)
A2 = 32;
A3 = 28;
A4 = 32;
a1 = 0.071; a3 = 0.071; %(cm^2)
a2 = 0.057; a4 = 0.057;
kc = 0.5; % (V/cm)
g = 981; %(cm/s^2)
gamma1 = 0.7; gamma2 = 0.6;     % constants, determined from valve position
k1 = 3.33; k2 = 3.35; %[cm^3/Vs]
kc = 0.5; % [V/cm]
v1 = 3; v2 = 3; % (V)
h0 = [12.4; 12.7; 1.8; 1.4];
```

$$\boxed{p(\mathbf{x}_k|\mathbf{Z}_k) = \kappa\, p(\mathbf{z}_k|\mathbf{x}_k)\int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})d\mathbf{x}_{k-1}}$$

- ***Initialisation***, t = 0; (***Prior Step***)
  For ***i = 1, ..., N***, sample ***x(0) ~ p{x(0)}*** & set ***t = 1***

Particle Filter Algorithm

- ***Importance Sampling***
  - ***Prediction Step***
    For ***i = 1,...,N***, sample ***x(t,i) ~ p{x(t)|x(t-1, i)}***
  - ***Likelihood Step***
    For ***i = 1,...,N***, evaluate importance weights : ***w(t,i) = p{z(t)|x(t,i)***
    Normalise the Importance Weights
- ***Selection***
  - ***Update Step/Resampling Step***
    Resample with replacement, according to Importance Weights
    Set ***t = t + 1*** and return to ***Prediction Step***

Prior

- For a state vector of dimension (1 *x* 1)
  - **x(:,1) = sqrt(P0) * randn(N,1)**
  - **We obtain N samples (particles)**
- For a state vector of dimension (**n** *x* **1**)
  - **L = cholesky (P0)**
  - **x = randn(N, n) * L + ones(N, n) * x (n x 1)**
  - **We obtain *x* of order (N x n)**

$x(k+1) = A x(k) + B u(k) + w(k)$

P

Prior

- For a state vector of dimension (1 *x* 1)
  - **x(:,1) = sqrt(P0) * randn(N,1)**
  - **We obtain N samples (particles)**
- For a state vector of dimension (**n** *x* **1**)
  - **L = cholesky (P0)**
  - **x = randn(N, n) * L + ones(N, n) * x (n x 1)**
  - **We obtain *x* of order (N x n)**

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1N} \\ x_{21} & x_{22} & \cdots & x_{2N} \end{bmatrix} \quad 2 \times N$$

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \rightarrow N \qquad n \times N$$

$x = n \times 1$

$n = 4 \quad x^T = [x_1 \ x_2 \ x_3 \ x_4]$

$N = 100$

$$\begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ \vdots & & & \\ x_{1N} & \cdots & & \end{bmatrix} \quad (n \times N) \quad \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1N} \\ x_{21} & x_{22} & \cdots & x_{2N} \\ & & & \vdots \\ & & & x_{nN} \end{bmatrix}$$

Initialization

```
% initialization of the states and covariances
x0 = h0;
x_post = x0;
C = [kc 0 0 0; 0 kc 0 0];
n = 4;                  % number of states
P0 = 10^5 * eye(4);     % initial covariance
N = 500;                % number of particles
```

Prior

```
L = chol(P0);
x0 = (x0 * ones(1,N))' + randn(N,n) * L; %Adding covariance to each particle
% Initialization of the states
x1_post = x0(:,1);
x2_post = x0(:,2);
x3_post = x0(:,3);
x4_post = x0(:,4);
```

Initialization

```
% initialization of the states and covariances
x0 = h0;
x_post = x0;
C = [kc 0 0 0; 0 kc 0 0];
n = 4;                  % number of states
P0 = 10^5 * eye(4);     % initial covariance
N = 500;                % number of particles
```

Prior

```
L = chol(P0);
x0 = (x0 * ones(1,N))' + randn(N,n) * L; %Adding covariance to each particle
% Initialization of the states
x1_post = x0(:,1);
x2_post = x0(:,2);
x3_post = x0(:,3);
x4_post = x0(:,4);
```


$k-1^+$ k^- k^+

$\hat{x}(k^-) = A\hat{x}(k-1) + Bu(k-1)$

$P_{k^-} = AP_{k-1}^+ A^T + Q$

z_{k-1} z_k

$k-1$ k

$$\boxed{p(\mathbf{x}_k|\mathbf{Z}_k) = \kappa\; p(\mathbf{z}_k|\mathbf{x}_k)\int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})d\mathbf{x}_{k-1}}$$

- ***Initialisation***, t = 0; (***Prior Step)***

  For ***i = 1, ..., N***, sample ***x(0) ~ p{x(0)}*** & set ***t = 1***

- ***Importance Sampling***
  - ***Prediction Step***

    For ***i = 1,...,N***, sample ***x(t,i) ~ p{x(t)|x(t-1, i)}***
  - ***Likelihood Step***

    For ***i = 1,...,N***, evaluate importance weights : ***w(t,i) = p{z(t)|x(t,i)***

    Normalise the Importance Weights

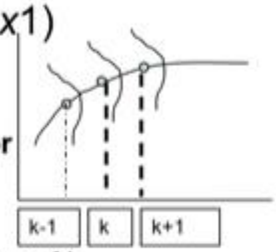
- ***Selection***
  - ***Update Step/Resampling Step***

    Resample with replacement, according to Importance Weights

    Set ***t = t + 1*** and return to ***Prediction Step***

Prediction

- *For a state vector of dimension* (1x1)
  - **w ~ sqrt (Q) * randn(N,1)**
  - **x(k-) = fn. { x, u } + w**
  - **We obtain predicted *x* as (N *x* 1) vector**

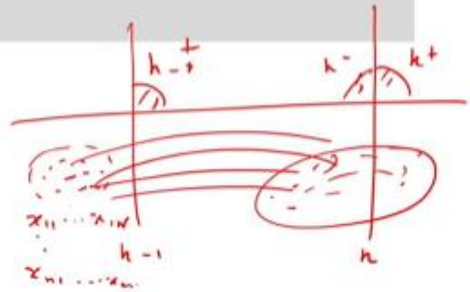


- *For a state vector of dimension* (**n** *x* **1**)
  - **w ~Cholesky (Q) * randn(N *x* n)**
  - **For i = 1 to N,**
    - **x*1*(k-) = fn. {x, u} + w(i, *1*)**
    - **:**
    - **x*n*(k-) = fn. {x, u} + w(i, *n*)**
  - **We obtain predicted x as (N x n) vector**

$p(x_k|z_k) = k \cdot p(z_k|x_k) \cdot [\, p(x_k|x_{k-1}) \, p(x_{k-1}|z_{k-1}) \,]$

```
% Prediction equation
for i = 1:N
    x1_pri(i) = -a1/A1*sqrt(2*g*x1_post(i)) + a3/A1*sqrt(2*g*x3_post(i)) + (gamma1*k1*v1)/A1
                + w1(i);
    x2_pri(i) = -a2/A2*sqrt(2*g*x2_post(i)) + a4/A2*sqrt(2*g*x4_post(i)) + (gamma2*k1*v2)/A2
                + w2(i);
    x3_pri(i) = -a3/A3*sqrt(2*g*x3_post(i)) + (1 - gamma2)*k2*v2/A3  +  w3(i);
    x4_pri(i) = -a4/A4*sqrt(2*g*x4_post(i)) + (1 - gamma1)*k1*v1/A4  +  w4(i);
end
```

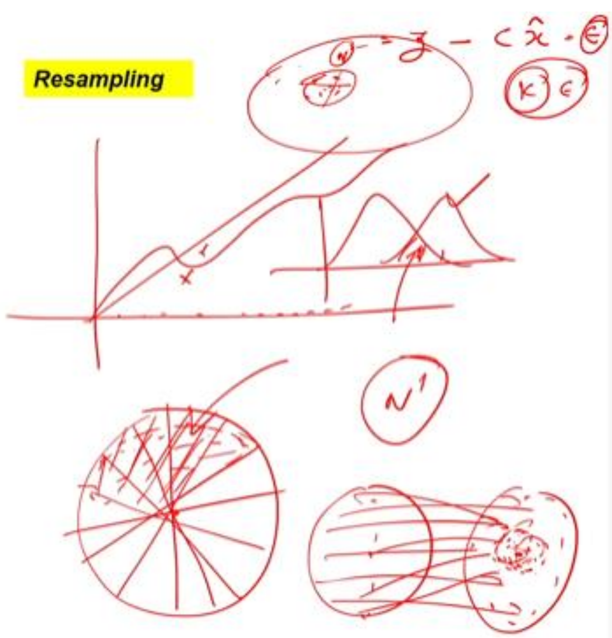
```
x_pri  = abs(x_pri);
x1_pri = abs(x1_pri);
x2_pri = abs(x2_pri);
x3_pri = abs(x3_pri);
x4_pri = abs(x4_pri);
```



As per sir, this is more than a pseudo code, almost the code!

```
M = length(wt);
Q = cumsum(wt);
indx = zeros(1, N);
T = linspace(0,1-1/N,N) + rand/N;
i = 1; j = 1;
while(i<=N && j<=M),
   while Q(j) < T(i)
      j = j + 1;
   end
   indx(i) = j;
    x1_post(i) = x1_pri(j);
    x2_post(i) = x2_pri(j);
    x3_post(i) = x3_pri(j);
    x4_post(i) = x4_pri(j);
   i = i + 1;
end
```

Resampling



Likelihood/Importance Weights

- *Considering an Exponential pdf*
  - ***v = z* ones(*N *x 1) – C ** $\hat{x}$ *(predicted)***
    (1 x n) (n x N)
  - **For i = 1 to N**

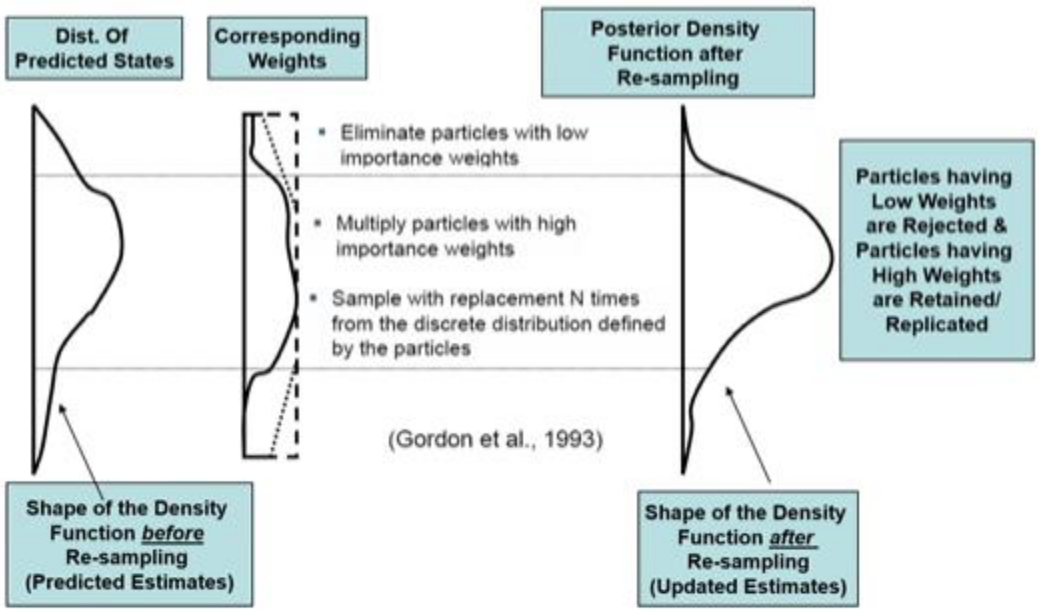
    $$q(i) = e^{-\frac{1}{2}[v(i,:)^T R^{-1} v(i,:)]}$$

- *Normalising the Weights*
  - ***For i = 1 to N***
    - ***Wt (i) = q(i) / sum(q)***

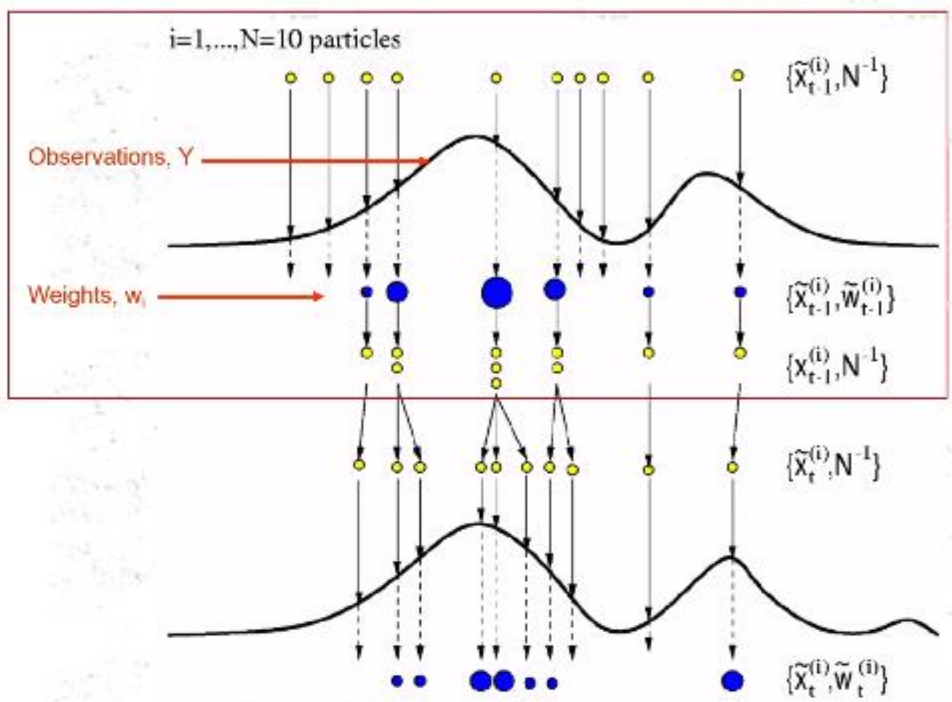
```
% Importance Weights (Likelihood Function)
z1 = 12.4;
z2 = 12.7;
z = [z1; z2];
z_true = z * ones(1,N);
R = 10 * eye(2);
z_est = C*x_pri;
 v = z_true - z_est;
 for i = 1:N
   q(i) = exp(-0.5 * (v(:,i)' * inv(R) * v(:,i)));
end
```

```
% Normalizing the weights
 for i = 1:N
   wt(i) = q(i)/sum(q);
end
```

Resampling



The Particle Filter Principle



Comparing Particle Filter and Kalman Filter

- ***Kalman Filter***
- ***Guess/Prior Knowledge*** :
  x(0), P(0), Q, R
- ***Prediction/Propagation*** :
  x(k-) = Ax(k-1+) + Bu(k)

  P(k-) = AP(k-1+)A' + Q
- ***Update***
  r = z - cx
  Obtaining Residues

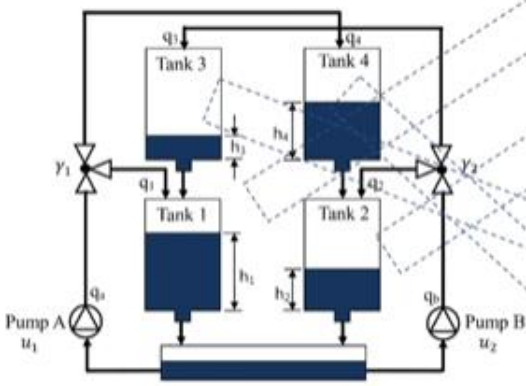
  K = ...
  x(k+) = x(k-) + K.r
  P(k+) = P(k-) – K...
- ***Repeat for t = 1:T***
- ***Obtain Aposteriori Density***

- ***Particle Filter***
- ***Prior Knowledge : P(0), Q, R***
- ***Prior :***
  x(0) = fn. {Sqrt/Chol. (P(0) . randn}
  (n x N) : N particles
- ***Prediction*** :
  For 1:N(particles)
  x(k-) = Ax(k-1+) + Bu(k) + Q
- ***Likelihood Function/ Importance Wt. Generation (N)***
  pdf{[z – cx(k-1)], R} = Wts.(N)
- ***Resampling*** *(of the* ***N*** *Weights)*
- ***Repeat for t = 1: T***
- ***Obtain Aposteriori Density***

In Kalman filter we linearized non-linear equations. (2 images below)

The 4-Tank Problem

A_i cross-section of Tank i;
a_i cross-section of the outlet hole;
h_i water level.



The voltage applied to Pump i is v_i and the corresponding flow is $k_i v_i$. The parameters $\gamma_1, \gamma_2 \in (0, 1)$ are determined from how the valves are set prior to an experiment. The flow to Tank 1 is $\gamma_1 k_1 v_1$ and the flow to Tank 4 is $(1-\gamma_1)k_1 v_1$ and similarly for Tank 2 and Tank 3. The acceleration of gravity is denoted g. The measured level signals are $k_c h_1$ and $k_c h_2$. The parameter

$$\frac{dh_1}{dt} = -\frac{a_1}{A_1}\sqrt{2gh_1} + \frac{a_3}{A_1}\sqrt{2gh_3} + \frac{\gamma_1 k_1}{A_1}v_1,$$

$$\frac{dh_2}{dt} = -\frac{a_2}{A_2}\sqrt{2gh_2} + \frac{a_4}{A_2}\sqrt{2gh_4} + \frac{\gamma_2 k_2}{A_2}v_2,$$

$$\frac{dh_3}{dt} = -\frac{a_3}{A_3}\sqrt{2gh_3} + \frac{(1-\gamma_2)k_2}{A_3}v_2,$$

$$\frac{dh_4}{dt} = -\frac{a_4}{A_4}\sqrt{2gh_4} + \frac{(1-\gamma_1)k_1}{A_4}v_1,$$

Linearized State Eqn. & Meas. Eqn.

$$\boxed{x_k = A_{k|k-1}x_{k-1} + B_{k-1}u_{k-1} + w_{k-1}}$$

$$A = \begin{bmatrix} -\frac{1}{T_1} & 0 & \frac{A_3}{A_1T_3} & 0 \\ 0 & -\frac{1}{T_2} & 0 & \frac{A_4}{A_2T_4} \\ 0 & 0 & -\frac{1}{T_3} & 0 \\ 0 & 0 & 0 & -\frac{1}{T_4} \end{bmatrix} \qquad B = \begin{bmatrix} \frac{\gamma_1 k_1}{A_1} & 0 \\ 0 & \frac{\gamma_2 k_2}{A_2} \\ 0 & \frac{(1-\gamma_2)k_2}{A_3} \\ \frac{(1-\gamma_1)k_1}{A_4} & 0 \end{bmatrix}$$

where the time constants are

$$T_i = \frac{A_i}{a_i}\sqrt{\frac{2h_i^0}{g}}, \qquad i = 1, \cdots, 4.$$

$$\boxed{z_k = H_k x_k + D_k u_k + v_k} \qquad H = \begin{bmatrix} k_c & 0 & 0 & 0 \\ 0 & k_c & 0 & 0 \end{bmatrix}$$