CARNEGIE MELLON UNIVERSITY
APPLIED STOCHASTIC PROCESSES
(COURSE 18-751)
HOMEWORK 9

Daniel Marew

October 31, 2017

I collaborated with :

Nebyou Yismaw
Daniel Nkemelu
Agatha Niwomugizi

Q.1

(a)

1) a) The R.V. total number of tickets sold, is a sum of random number of R.Vs.

$$Z = x_1 + x_2 + x_3 + \cdots x_N$$

$$N = \text{Poisson}(5) \quad , \quad X = \text{Poisson}(2)$$

$$P_X(x) = \begin{cases} \dfrac{\mu^x e^{-\mu}}{x!} & x = 0, 1, 2 \ldots \\ 0 & \end{cases}$$

$$\Theta_X(s) = \sum_{x=0}^{\infty} \frac{e^{sx} \mu^x e^{-\mu}}{x!} = \sum_{x=0}^{\infty} \frac{(e^s \mu)^x}{x!} e^{-\mu}$$

$$= e^{(e^s \mu)} e^{-\mu}$$

$$= e^{\mu(e^s - 1)}$$

$$\therefore \Theta_N(s) = e^{5(e^s - 1)} \qquad \Theta_X(s) = e^{2(e^s - 1)}$$

$$\Theta_Z(s) = \Theta_N(\ln \Theta_X(s))$$

$$= e^{5(e^{\ln \Theta_X(s)} - 1)} = e^{5(\Theta_X - 1)}$$

$$= e^{5(e^{2(e^s - 1)} - 1)}$$

(b)

$$\Theta_X(s) = \sum_{x=-\infty}^{\infty} e^{sx} P_X[X=x]$$

$$= \sum_{x=0}^{\infty} e^{sx} P_X[X=x] \quad , \text{ non-negative}$$

$$= e^{s\cdot 0} P_X[X=0] + \sum_{x=1}^{\infty} e^{sx} P_X[X=x]$$

$$= P_X[X=0] + \sum_{x=1}^{\infty} e^{sx} P_X[X=x]$$

$$\lim_{s\to-\infty} \Theta_X(s) = \lim_{s\to-\infty} \left(P_X[X=0] + \sum_{x=1}^{\infty} e^{sx} P_X[X=x] \right)$$

$$= P_X[X=0] + \lim_{s\to-\infty} \sum_{x=1}^{\infty} e^{sx} P_X[X=x]$$

$$= P_X[X=0] + \sum_{x=1}^{\infty} \lim_{s\to-\infty} e^{sx} P_X[X=x]$$

$$= P_X[X=0] + \sum_{x=1}^{\infty} P_X[X=x] \lim_{s\to-\infty} \overset{0}{e^{sx}}$$

$$\lim_{s\to-\infty} e^{sx} = 0 \Rightarrow \quad = P_X[X=0] + \sum_{x=1}^{\infty} P_X[X=x]\cdot 0$$

$$= \underline{\underline{P_X[X=0]}}$$

$\therefore$ For Discrete non-negative R.Vs $\lim_{s\to-\infty} \Theta_X(s) = P_X[X=0]$

$\Theta_Z(s) = e^{5(e^{2(e^s-1)}-1)}$, Z is a discrete non-negative R.V

$$\therefore \lim_{s\to-\infty} e^{5(e^{2(e^s-1)}-1)} = P[Z=0]$$

$$P_Z[Z=0] = \lim_{s\to-\infty} e^{5(e^{2(e^s-1)2}-1)} \qquad \lim_{s\to-\infty} e^s = 0$$

$$= e^{5(e^{-2}-1)}$$

$$= 0.0133$$

$$\therefore \underline{\underline{P_Z[Z=0] = 0.0133}}$$

(c,d)

(c) $\mu_z = \mu_x \times \mu_N = 2 \times 5 = \underline{\underline{10}}$ $\quad \sigma_N^2 = \mu_N \quad \sigma_x^2 = \mu_x$

$$\sigma_z^2 = \sigma_N^2 \mu_x^2 + \sigma_x^2 \mu_N$$
$$= 5 \times 2^2 + 2 \times 5$$
$$= 30$$

$\Rightarrow \mu_z = 10 \quad \sigma_z^2 = 30 \Rightarrow s.d = \underline{\underline{\sqrt{30}}} = \underline{\underline{5.477}}$

(d) We want to find N such that $P[z < N] > 0.99$

$$\Rightarrow 1 - P[z > N] > 0.99$$
$$\Rightarrow P[z > N] < 0.01$$

Using markov bound

$$P[z > N] < \frac{\mu_z}{N} = 0.01$$
$$\Rightarrow P[z > N] < 0.01 \Rightarrow \frac{\mu_z}{N} = 0.01$$
$$\Rightarrow N = \frac{\mu_z}{0.01} = \frac{10}{0.01} = \underline{\underline{1000}}$$

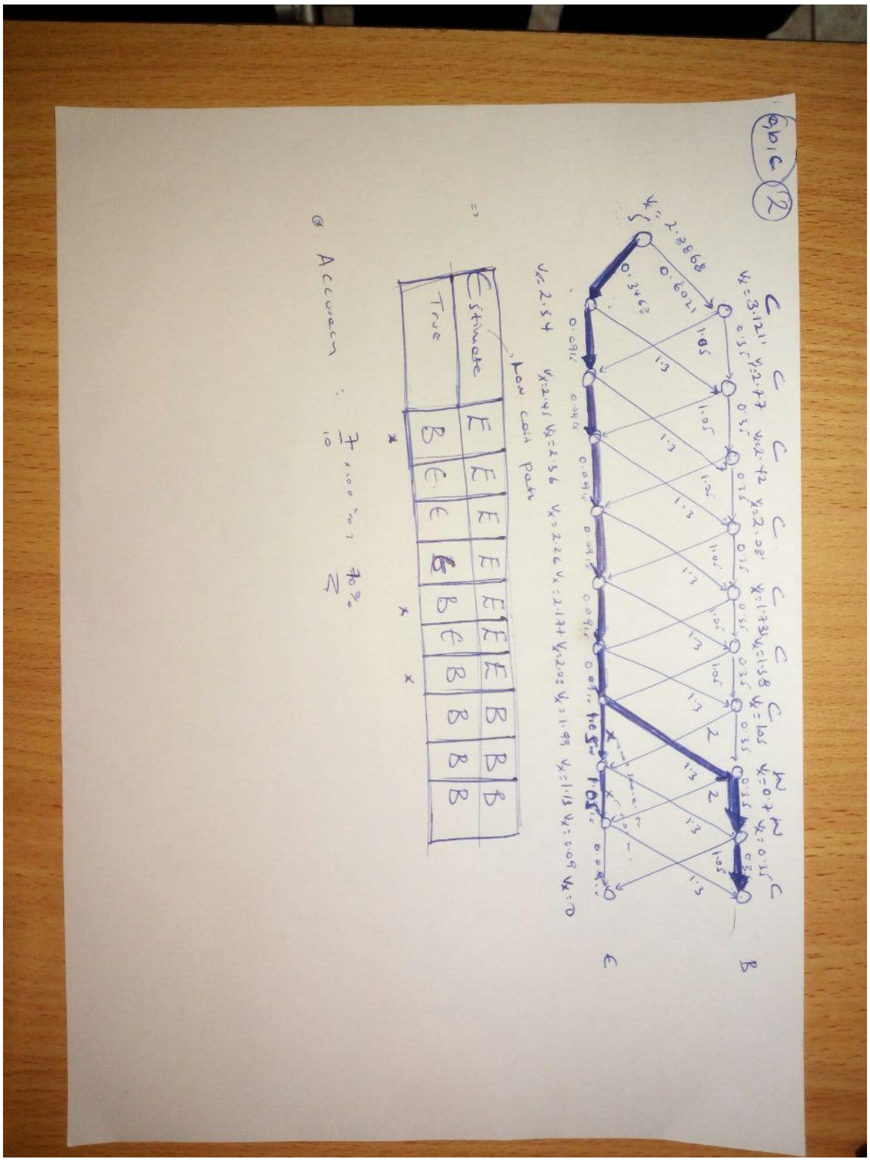
For tighter bound we use Chebishev bound.

$$P[\,|z - \mu_z| > N] < \frac{\sigma_z^2}{N^2} = 0.01$$
$$\Rightarrow N = \sqrt{\frac{\sigma_z^2}{0.01}} = \sqrt{\frac{30}{0.01}} = \sqrt{3000} = 54.77 = 55$$
$$\Rightarrow P[\,|z - 10| > 55] < 0.01$$
$$P[z > 65] + P[\cancel{z < -45}] < 0.01$$
$$P[z > 65] < 0.01$$
$$\Rightarrow P[z > 65] > 0.99$$

we take $N = \underline{65}$

Q.2

(a,b,c)



(d)

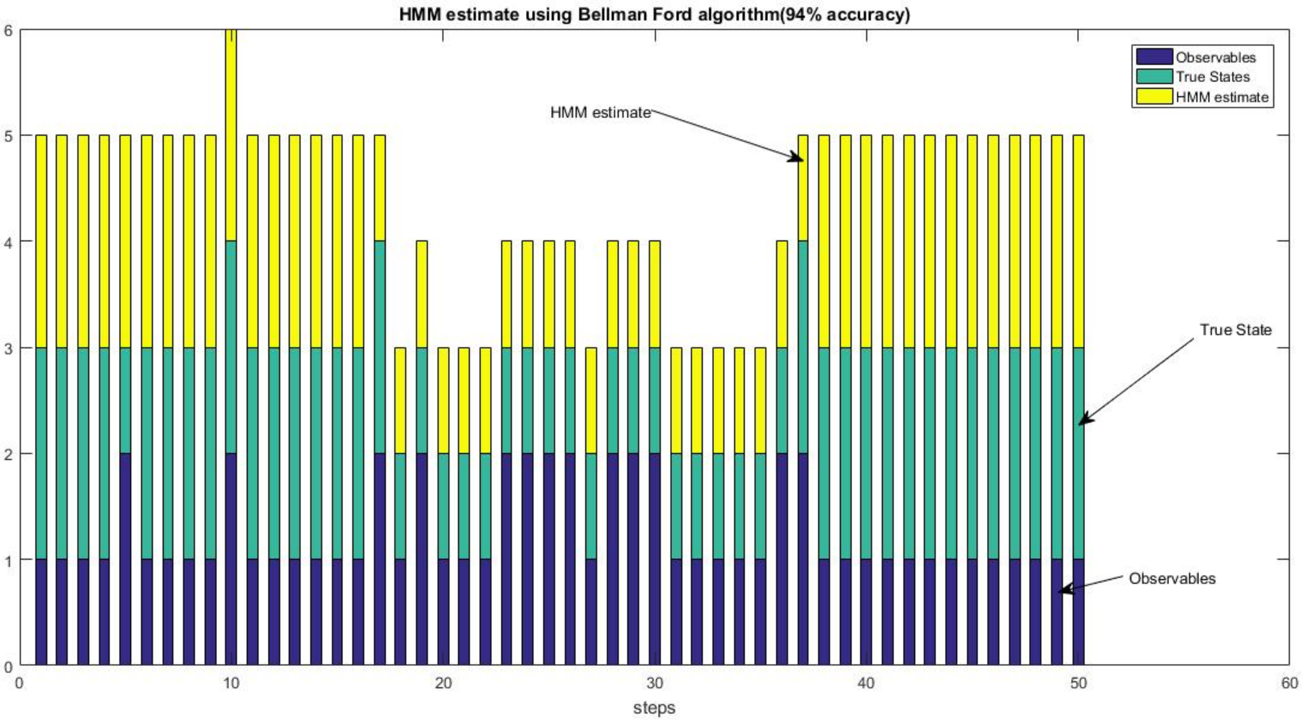


Figure 1: HMM with Bellman Ford (94% accuracy)

$Accuracy = 94\%$
Errors typically occur at transitions
Interpretation of the graph

Observables (length of the bar)

1: Correct
2: Wrong

True states and Estimate(length of the bar)

1: Bored
2: Engaged

To check the accuracy of the estimate, compare the lengths of the estimate and the true value.

Q.3 Kalman Filter

(a)

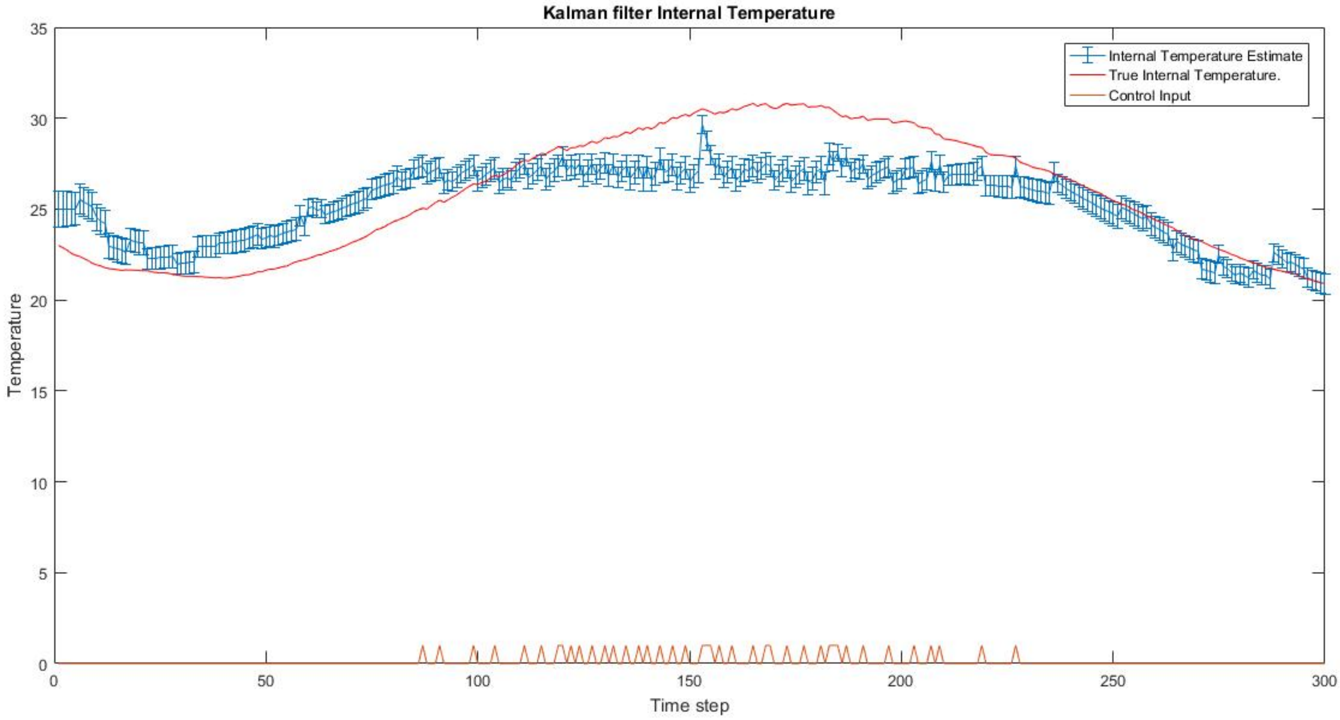


Figure 2: Kalman Filter for Tempreture control

(b)

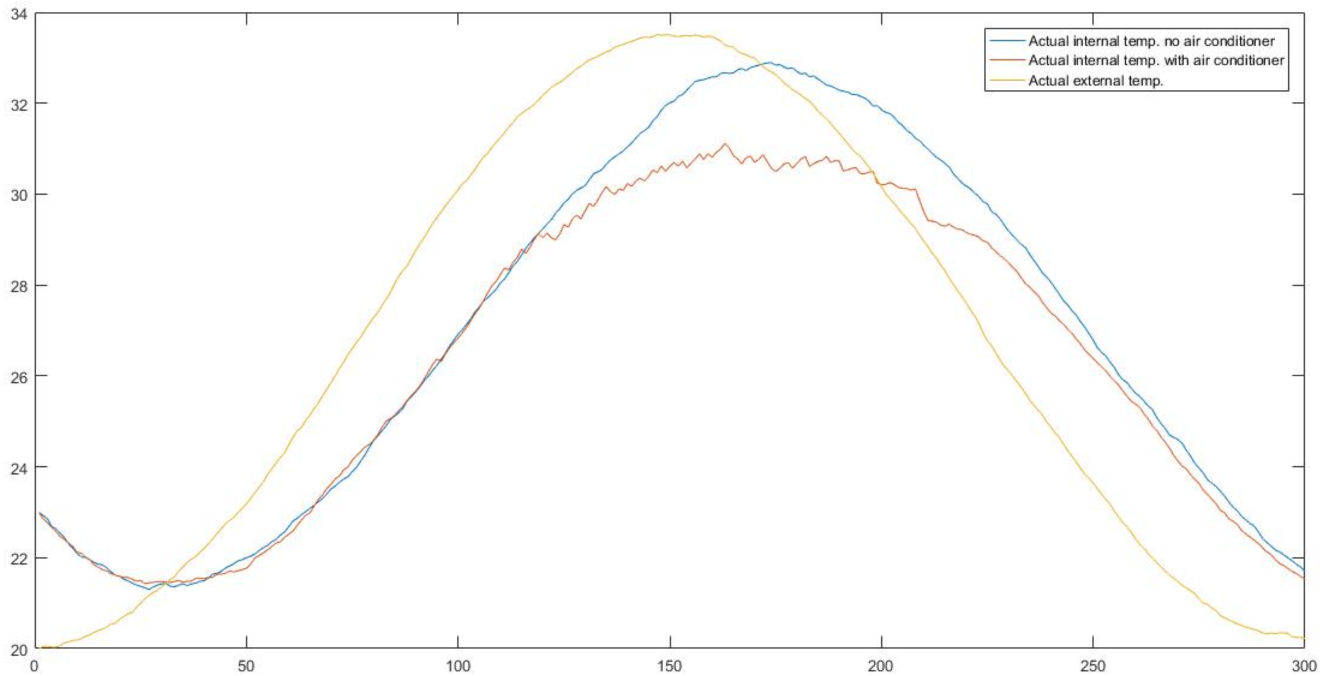


Figure 3: Kalman Filter for Tempreture control

The internal temperature swings with in an acceptable range (doesn't go beyond 29-30 degrees) where as internal temperature without the air conditioner could go upto 33 degrees .

Code Appendix

```
function [tempGraph,VX,stateS,HMMestimate] = create_graph(P,O,Y,
    N,PI_0)
tempGraph = zeros(4,N-1);
VX = zeros(2,N);
VX(:,N) = zeros(2,1);
selectedEdges = zeros(2,N);
for i=1:N-1
    currentO = O(:,Y(i+1));
    currentO = [currentO(:),currentO(:)];
    tempGraph(:,i) = -log10(P(:).*currentO(:));
end

for i=N-1:-1:1
    [VX(1,i),selectedEdges(1,i+1)]= min(tempGraph(1:2,i) + VX(:,
    i+1));
    [VX(2,i),selectedEdges(2,i+1)]= min(tempGraph(3:4,i) + VX(:,
    i+1));
end

currentO = O(:,Y(1));
stateSedges = -log10(PI_0.*currentO(:));
[stateS,stateSbestedge] = min(stateSedges + VX(1));
stateSbestedge
selectedEdges
HMMestimate = zeros(1,N);
HMMestimate(1) = stateSbestedge;
for i=2:N
HMMestimate(i) = selectedEdges(HMMestimate(i-1),i);
end
end
```

-

```
clear;
clc;
P = [0.9,0.1;0.1,0.9];
O = [0.5,0.5;0.9,0.1];

%Bored = 1;
%Engaged = 2
%Correct = 1
%Wrong = 2


[states,observables] = simMC(50,P,O);


display_states(states);
display_observables(observables);

[edgeWeights,VX,stateS,HMMestimate] = create_graph(P,O,
    observables,50,[0.5;0.5]);
```

```
display_states(HMMestimate);
accuracy = 100*(sum(HMMestimate==states)/50);
fprintf('Accuracy of HMM estimate:%2.2f percent\n',accuracy);
figure
bar([observables' states' HMMestimate'], 0.5, 'stack');
xlabel('steps')
legend('Observables', 'True States', 'HMM estimate');
```

```
clear;
clc;
A=[0.95 0.05;0 1];%state transtion matric
H=[1 0;0 1];%measurment transition
X= zeros(2,300);% blind prediction
Var = zeros(2,300);
Xh=X;%estimate
P = zeros(2,2,300);%estimate covariance
K=P;%gain
Xh(:,1)= [25,25];%initial state estimate
numberOfTimeSteps = 299;
Q = [0.04,0 ; 0,0.01];%model covariance
R = [4 0;0 1];%measurement covariance
Z = zeros(2,300); %measurment
turnOn = 0;%comand to turn on an off air conditioner
u_k = 0;%control input
P(:,:,1)= eye(2);%initialize estimate covariance to some big
    value
U_k = zeros(300,1);%hold command history
B = [-1,0]';%B
meanEstimateInternal = zeros(300,1);
varianceEstimate = zeros(300,1);
alpha =0.25;%contol the MC
beta  =0.5;%control the MC
mcP = [1-alpha,alpha;beta,1-beta];%markov chain transition
    matrix
state = 1;%start state of MC
Ztrue = zeros(2,300);
Xtrue = Ztrue;
Xtrue(:,1) = [23,20];
ZtrueNoConditioner = zeros(2,300);
XtrueNoConditioner = ZtrueNoConditioner;
XtrueNoConditioner(:,1) = [23,20];

for n=1:numberOfTimeSteps
    u_k = turnOn;
    U_k(n) = u_k;
    [Xtrue(:,n+1),Ztrue(:,n+1)] = get_measurment(Xtrue(:,n),u_k,
    n);%get measurement at constant rate

    [XtrueNoConditioner(:,n+1),ZtrueNoConditioner(:,n+1)] =
    get_measurment(XtrueNoConditioner(:,n),0,n);%get measurement
    at constant rate
    if state==1%if in prediction state


```

```
        Xh(:,n+1)  = A*Xh(:,n) + 0.1*sin((2*pi/300)*n) + B*u_k ; %prediction
        P(:,:,n+1) = A*P(:,:,n)*A' + Q;%predicted covariance
        state = discrete(mcP(state,:));%get next state
        fprintf('prediction\n');

    end
    while state==2%while measurment is available update

        %get measurement
        [Xtrue(:,n+1),Ztrue(:,n+1)] = get_measurment(Xtrue(:,n),u_k,n);%get measurement at constant rate
        [XtrueNoConditioner(:,n+1),ZtrueNoConditioner(:,n+1)] = get_measurment(XtrueNoConditioner(:,n),0,n);%get measurement at constant rate

        Z(:,n) = Ztrue(:,n) + [normrnd(0,4); normrnd(0,1)];

        K = P(:,:,n+1) * H' * ((H*P(:,:,n+1)*H' +R))^(-1);
        P(:,:,n+1) = (eye(2)-K*H)*P(:,:,n+1);
        Xh(:,n+1)= Xh(:,n+1)+K*(Z(:,n)-H*Xh(:,n+1));

        fprintf('fusion\n');
        state = discrete(mcP(state,:));

    end
    meanEstimateInternal(n) = Xh(1,n+1);
    varianceEstimate(n) = P(1,1,n+1);
    %if prob. temp>28 is >10%=0.1
    if qfunc((28-meanEstimateInternal(n))/sqrt(varianceEstimate(n)))>0.1
        turnOn = 1;
    else
        turnOn = 0;
    end
end


internalTempError = sqrt(P(1,1,:));%error matric
externalTempError = sqrt(P(2,2,:));
figure;
errorbar(1:300,Xh(1,:),internalTempError(:));
hold on;
%plot(Z(1,:));
%hold on;
plot(Xtrue(1,:),'r');
%plot(Ztrue(1,:),'b--');
plot(U_k(:,1));
title('Kalman filter Internal Temperature');
xlabel('Time step')
ylabel('Temperature')
legend('Internal Temperature Estimate','True Internal Temperature.','Control Input');
```

```
figure;
plot(XtrueNoConditioner(1,:),'DisplayName','Actual internal temp. no air conditioner');
hold on;
plot(Xtrue(1,:),'DisplayName','Actual internal temp. with air conditioner');
plot(Xtrue(2,:),'DisplayName','Actual external temp.');
legend('show')
```

```
function [X_k,Z_k] = get_measurment(X_k_1,u_k,n)
A = [0.96,0.04;0,1];
H = eye(2);
B = [-0.2;0];
V = [normrnd(0,1.5);normrnd(0,1.5)];
W = [normrnd(0,0.03);normrnd(0,0.02)];

X_k = A*X_k_1 + [0;0.14*sin((2*pi/300)*n)] + B*u_k + W ;
Z_k = H*X_k + V ;

end
```