SSY191 - Sensor Fusion and Nonlinear Filtering
Implementation of Home Assignment 01

Lucas Rath

Listings


```
clearvars; close all; clc;

%% Transformation of Gaussian random variables
% (a)

close all;

% p(x) parameters
mu_x = [10;0];
Sigma_x = diag([0.2 8]);

% apply linear transformation: h(x) = A*x + b
A = [1 1; 1 -1];
b = [0;0];
[mu_y, Sigma_y] = affineGaussianTransform(mu_x,Sigma_x, A, b);
% generate 3-sigma ellipse curve
xy = sigmaEllipse2D(mu_y,Sigma_y,3,100);


% Estimate mean and covariance of the linear transformation: h(x) = A*x + b
f = @(x) A*x+b;
N = 1000;
[mu_ya, Sigma_ya, y_s] = approxGaussianTransform(mu_x,Sigma_x,f,N);
xya = sigmaEllipse2D(mu_ya,Sigma_ya,3,100);


% Generate the plots
figure('Color','white','Position',[399  381  543  418]);
hold on, grid on;
cp = fp.getColor(1:2);
scp = scatter(y_s(1,:), y_s(2,:), 10,'filled', 'MarkerFaceColor',cp(1,:));
scp.MarkerFaceAlpha = 0.5;
h1a = plot(xya(1,:), xya(2,:), 'LineWidth',3, 'Color', cp(1,:));
s1a = scatter(mu_ya(1), mu_ya(2), 100, 'h','filled', 'MarkerFaceColor', cp(1,:), 'MarkerEdgeColor',
h1 = plot(xy(1,:), xy(2,:),'--', 'LineWidth',3, 'Color', cp(2,:));
sc1= scatter(mu_y(1), mu_y(2), 100, 'h','filled', 'MarkerFaceColor', cp(2,:), 'MarkerEdgeColor', cp(
sc1.MarkerFaceAlpha = 1;
xlim([0,20])
ylim([0,20])
xlabel 'y(1)', ylabel 'y(2)'
title({'Approximated and analytical computation of ',['transformed density p(y)=p(h(x)) - N=',num2st
legend({'Approx transformed samples', 'Approx. 3\sigma ellipse', 'Approx. mean', ...
        'Analytic  3\sigma ellipse','Analytic mean'})
% fp.savefig(sprintf('q1-N-%s',num2str(N)))
```

```
%% Transformation of Gaussian random variables
%  (b)

close all;

% p(x) parameters
mu_x = [10;0];
Sigma_x = diag([0.2 8]);

% Estimate mean and covariance of the nonlinear transformation: h(x) = A*x + b
f = @(x) [ (x(1,:).^2+x(2,:).^2).^0.5 ; atan2(x(2,:),x(1,:))];
N = 10;
[mu_ya, Sigma_ya, y_s] = approxGaussianTransform(mu_x,Sigma_x,f,N);
xya = sigmaEllipse2D(mu_ya,Sigma_ya,3,100);

% Generate the plots
figure('Color','white','Position',[399  381  543  418]);
hold on, grid on;
cp = fp.getColor(1:2);
scp = scatter(y_s(1,:), y_s(2,:), 10,'filled', 'MarkerFaceColor',cp(1,:));
scp.MarkerFaceAlpha = 0.3;
h1a = plot(xya(1,:), xya(2,:), 'LineWidth',3, 'Color', cp(1,:));
s1a = scatter(mu_ya(1), mu_ya(2), 150, 'h','filled', 'MarkerFaceColor', cp(2,:), 'MarkerEdgeColor',
xlim([8,13])
ylim([-1,1])
xlabel 'y(1)', ylabel 'y(2)'
title({'Approximated and analytical computation of ',['transformed density p(y)=p(h(x)) - N=',num2st
legend({'Approx transformed samples', 'Approx. 3\sigma ellipse', 'Approx. mean'})
% fp.savefig(sprintf('q1b-N-%s',num2str(N)))



%% Snow depth in Norway
% (a)

close all;

% covariance of the prior distribution
sigma2_x = 0.5^2;


% ANNA

% mean for the prior distribution
mu_x_A = 1.1;
% likelihood covariance Cov[y|x]
sigma2_r_A = 0.2^2;

% calculate joint distribution p(x,y)
[mu_xy_A, sigma_xy_A] = jointGaussian( mu_x_A, sigma2_x, sigma2_r_A );
% calculate 3-sigma ellipses
xy_A = sigmaEllipse2D( mu_xy_A, sigma_xy_A, 3,100);
% calculate principal component vector
[eigVec_A, eigval_A] = covm2pca(sigma_xy_A);

% generate the plots
figure('Color','white','Position',[328  486  395  380]);
hold on, grid on, axis equal;
cp = fp.getColor(1:2);
quiver(mu_xy_A(1),mu_xy_A(2), 2*eigVec_A(1,1), 2*eigVec_A(2,1), 'LineWidth',2,'MaxHeadSize',0.3,'col
h1a = plot(xy_A(1,:), xy_A(2,:), 'LineWidth',3, 'Color', cp(1,:),'DisplayName','3\sigma ellipse');
```

```
s1a = scatter(mu_xy_A(1), mu_xy_A(2), 100, 'h','filled', 'MarkerFaceColor', cp(1,:), 'MarkerEdgeColo
xlim([-1,3])
ylim([-1,3])
xlabel 'x', ylabel 'y'
title('Joint distribution p(x,y) for Anna (E[x]=1.1)')
legend('Location','northwest')
% fp.savefig(sprintf('q2a-Anna'))


% ELSE

% mean for the prior distribution
mu_x_E = 1;
% likelihood covariance Cov[y|x]
sigma2_r_E = 1^2;

% calculate joint distribution p(x,y)
[mu_xy_E, sigma_xy_E] = jointGaussian( mu_x_E, sigma2_x, sigma2_r_E );
% calculate 3-sigma ellipses
xy_E = sigmaEllipse2D( mu_xy_E, sigma_xy_E, 3,100 );
% calculate principal component vector
[eigVec_E, eigval_E] = covm2pca(sigma_xy_E);

% generate the plots
figure('Color','white','Position',[328  486  395  380]);
hold on, grid on, axis equal;
cp = fp.getColor(1:2);
quiver(mu_xy_E(1),mu_xy_E(2), 3*eigVec_E(1,1), 3*eigVec_E(2,1), 'LineWidth',2,'MaxHeadSize',0.3,'col
h1a = plot(xy_E(1,:), xy_E(2,:), 'LineWidth',3, 'Color', cp(1,:),'DisplayName','3\sigma ellipse');
s1a = scatter(mu_xy_E(1), mu_xy_E(2), 100, 'h','filled', 'MarkerFaceColor', cp(1,:), 'MarkerEdgeColo

xlim([-3 5])
ylim([-3,5])
xlabel 'x', ylabel 'y'
title('Joint distribution p(x,y) for Else (E[x]=1)')
legend('Location','southeast')
% fp.savefig(sprintf('q2a-Else'))


%% Snow depth in Norway
% (b)

close all;

y_A = 1;

% calculate the posterior p(x|y)
[mu_xgy_A, sigma2_xgy_A] = posteriorGaussian(mu_x_A, sigma2_x, y_A, sigma2_r_A)

% generate the plots
figure('Color','white','Position',[242  401  491  343]);
x = linspace(mu_xgy_A-4*sqrt(sigma2_xgy_A), mu_xgy_A+4*sqrt(sigma2_xgy_A), 100);
y = normpdf(x, mu_xgy_A, sqrt(sigma2_xgy_A));
plot(x,y, 'LineWidth',2)
title 'p(x_H | y_A=1)  for \mu_x=1.1', ylabel 'p(x_H | y_A=1)', xlabel 'x_H', grid on;
% fp.savefig(sprintf('q2b-Anna'))


y_E = 2;

% calculate the posterior p(x|y)
[mu_xgy_E, sigma2_xgy_E] = posteriorGaussian(mu_x_E, sigma2_x, y_E, sigma2_r_E)

figure('Color','white','Position',[242  401  491  343]);
```

```
x = linspace(mu_xgy_E-4*sqrt(sigma2_xgy_E), mu_xgy_E+4*sqrt(sigma2_xgy_E), 100);
y = normpdf(x, mu_xgy_E, sqrt(sigma2_xgy_E));
plot(x,y, 'LineWidth',2)
title 'p(x_K | y_E=2)  for \mu_x=1', ylabel 'p(x_K | y_E=2)', xlabel 'x_K', grid on;
% fp.savefig(sprintf('q2b-Else'))



%% MMSE and MAP estimates for Gaussian mixture posteriors

close all;
cp = fp.getColor(1:3);

% item a)

figure('Color','white','Position',[242  457  404  287]);
hold on, grid on;
x = linspace( -7 , 9 , 200);
y = 0.1 * normpdf(x, 1, sqrt(0.5)) + 0.9 * normpdf(x, 1, sqrt(9));
pl1 = plot(x,y, 'LineWidth',2 ,'DisplayName','p(\theta|y)');
title 'p(\theta|y) = 0.1*N(\theta; 1, 0.5) + 0.9*N(\theta; 1, 9)', ylabel 'p(\theta|y)', xlabel '\th

% calculate x_MAP and x_MMSE
[MAP, MAPi] = max(y);
xMAP  = x(MAPi);
xMMSE = gaussMixMMSEEst( [0.1 0.9], [1 1], 0);

% plot figures
sc1 = scatter( xMAP, MAP, 100, 'o','filled' ,'MarkerFaceColor',cp(2,:),'DisplayName','\theta_{MAP} =
plot( [xMAP xMAP] , [0,MAP], '--', 'LineWidth',2 )
xlim([min(x),max(x)])
legend([pl1 sc1])
fp.savefig(sprintf('q3-a'))


% item b)

figure('Color','white','Position',[242  457  404  287]);
hold on, grid on;
x = linspace( -10 , 10 , 200);
y = 0.49 * normpdf(x, 5, sqrt(2)) + 0.51 * normpdf(x, -5, sqrt(2));
pl1 = plot(x,y, 'LineWidth',2 ,'DisplayName','p(\theta|y)');
title 'p(\theta|y) = 0.49*N(\theta; 5, 2) + 0.51*N(\theta; -5, 2)', ylabel 'p(\theta|y)', xlabel '\t

% calculate x_MAP and x_MMSE
[MAP, MAPi] = max(y);
xMAP  = x(MAPi);
xMMSE = gaussMixMMSEEst( [0.49 0.51], [5 -5], 0);
[M,I] = min(abs(x-xMMSE));
MMSE = y(I);

% plot figures
sc1 = scatter( xMAP, MAP, 100, 'o','filled' ,'MarkerFaceColor',cp(2,:),'DisplayName','\theta_{MAP}',
plot( [xMAP xMAP] , [0,MAP], '--', 'LineWidth',2, 'Color', cp(2,:))
sc2 = scatter( xMMSE, MMSE, 100, 'o','filled' ,'MarkerFaceColor',cp(3,:),'DisplayName','\theta_{MMSE
plot( [xMMSE xMMSE] , [0,MMSE], '--', 'LineWidth',2 , 'Color', cp(3,:))
xlim([min(x),max(x)])
legend([pl1 sc1 sc2],'Location','southeast')
fp.savefig(sprintf('q3-b'))


% item c)
```

```
figure('Color','white','Position',[242  457  404  287]);
hold on, grid on;
x = linspace( -4 , 6 , 300);
y = 0.4 * normpdf(x, 1, sqrt(2)) + 0.6 * normpdf(x, 2, sqrt(1));
pl1 = plot(x,y, 'LineWidth',2 ,'DisplayName','p(\theta|y)');
title 'p(\theta|y) = 0.4*N(\theta; 1, 2) + 0.6*N(\theta; 2, 1)', ylabel 'p(\theta|y)', xlabel '\thet

[MAP, MAPi] = max(y);
xMAP  = x(MAPi);
xMMSE = gaussMixMMSEEst( [0.4 0.6], [1 2], 0);
[M,I] = min(abs(x-xMMSE));
MMSE = y(I);

sc1 = scatter( xMAP, MAP, 100, 'o','filled' ,'MarkerFaceColor',cp(2,:),'DisplayName','\theta_{MAP}',
plot( [xMAP xMAP] , [0,MAP], '--', 'LineWidth',2, 'Color', cp(2,:))

sc2 = scatter( xMMSE, MMSE, 100, 'o','filled' ,'MarkerFaceColor',cp(3,:),'DisplayName','\theta_{MMSE
plot( [xMMSE xMMSE] , [0,MMSE], '--', 'LineWidth',2 , 'Color', cp(3,:))

xlim([min(x),max(x)])
legend([pl1 sc1 sc2],'Location','northwest')
fp.savefig(sprintf('q3-c'))
```

```
function [pc,eigval] = covm2pca(covm)
    % calculate eigenvector of the covariance matrix
    [eigVec, eigval] = eig(covm);
    % sort eigenvectors by descendent order of eigenvalues
    % the eigenvector with the highest eigenvalue is the major axis
    [eigval,I] = sort(diag(eigval),'descend');
    pc = eigVec(:,I);
end
```