Optimal Filtering, 2012–10–01
Due, 2012–10–08
Mats Bengtsson

Homework # 6

1. Problem 10.18 from Kailath's book

2. Tables 1–4 show Matlab functions to generate particle distibutions, representing a continuous random variable x with a specific distribution.

   a) Show analytically that all four functions generate particles corresponding to one and the same distribution of x and derive the PDF and CDF of that distribution.

   b) Use the four functions (available on the course web, Lect. 6) to generate particle distributions of suitable size and use these to numerically estimate the mean and variance of x. Which of the four sampling strategies gives the best accuracy of the mean and variance?

3. Continuation of the previous problem. Assume that $z = x^2 + u$, where x is the stochastic variable from the previous problem and u is uniformly distributed over $[-1, 1]$.

   a) Describe at least one method to generate a particle distribution for Z, given a particle distribution for X. Will the method work, no matter which of four strategies was used in the previous problem?

   b) Implement your method and use it to estimate values of

      - $\mathrm{E}[Z]$
      - $\mathrm{E}[Z^3]$
      - $P[Z > 40]$

   Again, continued from the previous problems. Assume now that we are observing the stochastic variable $Y = Z + W$, where the distribution of W is given in Fig. 1.

4. a) Describe how to update the weights of a particle representation of Z, to obtain a particle representation of the posterior distribution $f_{Z|Y}(z|Y = y)$.

   b) Implement the method and use it to determine estimates of

      - $\mathrm{E}[Z|Y = 60]$
      - $P[Z < 50|Y = 50]$

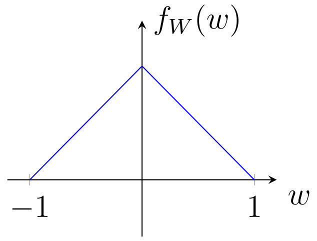

Figure 1: PDF of W.

Table 1: Algorithm 1 to generate N particles.

```
function [xp, w] = generateparticles1(N)
xmax=10;
xp=xmax*sqrt(rand(N,1));
w=ones(N,1)/N;
```

Table 2: Algorithm 2 to generate N particles.

```
function [xp, w] = generateparticles2(N)
xmax=10;
xp=xmax*rand(N,1);
w=xp/sum(xp);
```

Table 3: Algorithm 3 to generate N particles.

```
function [xp, w] = generateparticles3(N)
xmax=10;
lambda=1/5;
xp=-log(rand(N,1))/lambda;
w_unnormalized=xp.*exp(lambda*xp)/lambda;
w_unnormalized(xp>xmax)=0;
w=w_unnormalized/sum(w_unnormalized);
```

Table 4: Algorithm 4 to generate one out of N particles. The Matlab file on the course web gives N particles directly, but is slightly harder to read.

```
function [xp, w] = generateparticles4(N)
xmax=10;
sigma=7;
M=1.1*2/xmax/normpdf(xmax,0,sigma);
rejected=true;
while rejected
  xp=sigma*randn;
  u=rand;
  rejected = (xp<0 | xp>xmax | u >= (2*xp/xmax^2 ./ normpdf(xp,0,sigma)/M));
end
w=1/N; % If this is one of N generated particles
```