ACM 116 Problem Set 2 Solutions

Emmanuel Candés, Hannes Helgason and Lei Zhang

Problem 1

We follow in a similar manner as in class:

$$\begin{aligned} E|\hat{Y} - Y| &= E|T(X) - Y| = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |T(x) - y| f_{X,Y}(x,y)dxdy \quad (1)\\ &= \int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty} |T(x) - y| f_{Y|X}(y \mid x)dy\right]dx \quad (2)\end{aligned}$$

so if we minimize the inner integral in the last expression with respect to $T(x)$, we minimize $E|\hat{Y} - Y|$. For a fixed x, write $g(y) := f_{Y|X}(y \mid x)$, $t := T(x)$ and

$$H(t) := \int_{-\infty}^{\infty} |t - y| g(y)dy.$$

Note that $g(y)$ is a distribution function so its integral over the real axis is 1.

Before differentiating $H(t)$ with respect to t to find its minimizer, we need to do some algebra (we cannot simply differentiate inside the integral since $|t - y|$ is not differentiable where $t = y$). We have

$$\begin{aligned} H(t) &= \int_t^{\infty} (y - t)g(y)dy + \int_{-\infty}^{t} (t - y)g(y)dy \quad (3)\\ &= \int_t^{\infty} yg(y)dy - t\int_t^{\infty} g(y)dy + t\int_{-\infty}^{t} g(y)dy - \int_{-\infty}^{t} yg(y)dy. \quad (4)\end{aligned}$$

Now differentiating with respect to t (recalling our tools from calculus and that $g(y)$ is a distribution function) we get

$$\begin{aligned} H'(t) &= -tg(t) - \int_t^{\infty} g(y)dy - t(-g(t)) + \int_{-\infty}^{t} g(y)dy + tg(t) - tg(t) \quad (5)\\ &= -\int_t^{\infty} g(y)dy + \int_{-\infty}^{t} g(y)dy = -1 + 2\int_{-\infty}^{t} g(y)dy. \quad (6)\end{aligned}$$

Setting the derivative to zero gives us

$$\int_{-\infty}^{T(x)} f_{Y|X}(y \mid x)(y)dy = \int_{-\infty}^{t} g(y)dy = \frac{1}{2}.$$

Thus, the the estimator that minimizes $E|\hat{Y} - Y|$ is the *median of the posterior distribution*, $f_{Y|X}(y \mid x)$. That is, $\hat{Y} = T(x)$ is the number such that

$$P(Y \leq T(x) \mid X = x) = P(Y \geq T(x) \mid X = x) = 1/2.$$

Problem 2

(a) From the definition of conditioning,

$$\mathcal{L}(Y_t \mid Y_{t+1}, Y_{t-1}) = \frac{\mathcal{L}(Y_t, Y_{t+1} \mid Y_{t-1})}{\mathcal{L}(Y_{t+1} \mid Y_{t-1})} = \frac{\mathcal{L}(Y_{t+1} \mid Y_t, Y_{t-1})\ \mathcal{L}(Y_t \mid Y_{t-1})}{\mathcal{L}(Y_{t+1} \mid Y_{t-1})}$$

We can evaluate the distributions by appealing to the stochastic process.

$$\begin{aligned} Y_t &= \rho Y_{t-1} + \varepsilon_t \\ Y_t &\sim N(\rho\ Y_{t-1}, 1) \end{aligned}$$

$$\begin{aligned} Y_{t+1} &= \rho^2 Y_{t-1} + \rho\varepsilon_{t+1} + \varepsilon_t \\ Y_{t+1} &\sim N(\rho^2\ Y_{t-1}, \rho^2 + 1) \end{aligned}$$

The needed distributions are as follows:

$$\begin{aligned} \mathcal{L}(Y_t \mid Y_{t-1}) &= \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(y_t - \rho y_{t-1})^2}{2}\right) dy_t \\ \mathcal{L}(Y_{t+1} \mid Y_t, Y_{t-1}) &= \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(y_{t+1} - \rho y_t)^2}{2}\right) dy_{t+1} \\ \mathcal{L}(Y_{t+1} \mid Y_{t-1}) &= \frac{1}{\sqrt{2\pi}\sqrt{1+\rho^2}} \exp\left(-\frac{(y_{t+1} - \rho^2 y_{t-1})^2}{2(1+\rho^2)}\right) dy_{t+1} \end{aligned}$$

Giving,

$$\mathcal{L}(Y_t \mid Y_{t+1}, Y_{t-1}) = \frac{\sqrt{1+\rho^2}}{\sqrt{2\pi}} \frac{\exp\left(-\frac{y_t^2 - 2\rho y_t y_{t-1} + \rho^2 y_{t-1}^2}{2}\right) \exp\left(-\frac{y_{+1}t^2 - 2\rho y_{t+1} y_t + \rho^2 y_t^2}{2}\right)}{\exp\left(-\frac{y_{t+1}^2 - 2\rho y_{t+1} y_{t-1} + \rho^4 y_{t-1}^2}{2(1+\rho^2)}\right)}$$

After some algebra, this reduces to

$$\mathcal{L}(Y_t \mid Y_{t+1}, Y_{t-1}) = \frac{\sqrt{1+\rho^2}}{\sqrt{2\pi}} \exp\left(-\frac{1+\rho^2}{2}\left(y_t - \frac{\rho y_{t-1} + \rho y_{t+1}}{1+\rho^2}\right)^2\right)$$

We identify:

$$\begin{aligned} \mu_t &= \frac{\rho(y_{t-1} + y_{t+1})}{1+\rho^2} \\ \gamma &= \frac{1}{\sqrt{1+\rho^2}} \end{aligned}$$

(b) In matrix form, we can write

$$\mathbf{Y} := \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_{L+1} \\ Y_0 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 & \cdots & \rho \\ \rho & 1 & 0 & \cdots & \rho^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho^L & \rho^{L-1} & \rho^{L-2} & \cdots & \rho^{L+1} \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_{L+1} \\ Y_0 \end{pmatrix}}_{\varepsilon}.$$

(The reason why we choose this order of the entries in the vector $\mathbf{Y}$ will be clear soon.)
Lets assume $\varepsilon_0 := Y_0 \sim N(0,1)$ and that $\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_{L+1}$ are i.i.d. standard normal. Then we have that

$$Cov(\varepsilon_i, \varepsilon_j) = \begin{cases} 0 & \text{for } i \neq j \\ 1 & \text{for } i = j \end{cases} \qquad \text{and} \qquad E(\varepsilon) = \mathbf{0}.$$

Since ε follows a multivariate normal distribution, so is $\mathbf{Y}$ and we have

$$E(\mathbf{Y}) = \mathbf{A}E(\varepsilon) = \mathbf{0} \qquad \text{and} \qquad \mathbf{\Sigma} = \mathbf{Cov}(\mathbf{Y}) = \mathbf{ACov}(\varepsilon)\mathbf{A^T} = \mathbf{AA^T}.$$

Write

$$\Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}$$

where Σ_{11} is a $L \times L$ block matrix and Σ_{22} is a 2×2 block matrix, and

$$\mathbf{Y^{(1)}} = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_L \end{pmatrix} \qquad \text{and} \qquad \mathbf{Y^{(2)}} = \begin{pmatrix} Y_{L+1} \\ Y_0 \end{pmatrix}.$$

Then the conditional distribution of $Y^{(1)}$ given $Y^{(2)}$ is multivariate normal;

$$\mathcal{L}(Y^{(1)} \mid Y^{(2)}) \sim N(\mu', \Sigma')$$

where

$$\begin{aligned} \mu' &= E(\mathbf{Y^{(1)}}) + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{Y^{(2)}} - E(Y^{(2)}) && (7) \\ &= \Sigma_{12}\Sigma_{22}^{-1}\mathbf{Y^{(2)}} && (8) \\ \Sigma' &= \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}. && (9) \end{aligned}$$

These are the formulas we'll use in our code to do the sampling.

One way to calculate the L vector $\mu(y_0, y_{L+1})$ goes as follows:
Conditioning a multivariate normal on one of its components leaves a multivariate normal. To calculate the mean of Y_i, consider $\mathcal{L}(Y_i \mid Y_0, Y_{L+1})$.

$$\begin{aligned} \mathcal{L}(Y_i \mid Y_0, Y_{L+1}) &= \frac{\mathcal{L}(Y_i, Y_{L+1} \mid Y_0)}{\mathcal{L}(Y_{L+1} \mid Y_0)} \\ Y_i &\sim \rho^i y_0 + \rho^{i-1}\varepsilon_i + \cdots + \rho\varepsilon_2 + \varepsilon_1 \\ Y_{L+1} \mid Y_i = y_i &\sim \rho^{L+1-i} y_i + \rho^{L-i}\varepsilon_{L-i+1} + \cdots + \varepsilon_1 \\ Y_{L+1} &\sim \rho^{L+1} y_0 + \rho^{L-1}\varepsilon_{L+1} + \cdots + \varepsilon_1 \end{aligned}$$

Using the same method as in part a), we obtain for $\mathcal{L}(Y_t \mid Y_{t+1}, Y_{t-1})$:

$$\frac{\frac{1}{\sqrt{2\pi}\sqrt{\rho^{2(i-1)}+\cdots+\rho^2+1}} \exp\left(-\frac{(y_i-\rho^i y_0)^2}{2(\rho^{2(i-1)}+\cdots+\rho^2+1)}\right) \frac{1}{\sqrt{2\pi}\sqrt{\rho^{2(L-i)}+\cdots+\rho^2+1}} \exp\left(-\frac{(y_{L+1}-\rho^{L+1-i}y_i)^2}{2(\rho^{2(L-i)}+\cdots+\rho^2+1)}\right)}{\frac{1}{\sqrt{2\pi}\sqrt{\rho^{2L}+\cdots+\rho^2+1}} \exp\left(-\frac{(y_{L+1}-\rho^{L+1}y_0)^2}{2(\rho^{2L}+\cdots+\rho^2+1)}\right)}$$

As this distribution is normal, we know the exponent is of the form

$$-\frac{1}{2\sigma_i^2}(y_i - \mu_i)^2 = -\frac{1}{2}(Ay_i^2 - 2By_i + C)$$

$$\mu_i = \frac{B}{A}$$

After multiplying out the relevant terms, we observe

$$\mu_i = \frac{\left(\frac{\rho^i y_0}{\rho^{2(i-1)}+\cdots+\rho^2+1} + \frac{\rho^{L+1-i} y_{L+1}}{\rho^{2(L-i)}+\cdots+\rho^2+1}\right)}{\left(\frac{1}{\rho^{2(i-1)}+\cdots+\rho^2+1} + \frac{\rho^{2(L+1-i)}}{\rho^{2(L-i)}+\cdots+\rho^2+1}\right)}$$

$$= \frac{(\rho^{2L-i+2} - \rho^i)y_0 + (\rho^{L+i+1} - \rho^{L-i+1})y_{L+1}}{\rho^{2L+2} - 1}.$$

(c) See the accompanying `Matlab` file in the Appendix. To sample $Y_1, \ldots, Y_L$ from the distribution $\mathcal{L}(Y_1, \ldots, Y_L \mid Y_0 = y_0, Y_{L+1} = y_{L+1})$, we will use the formulas derived in part (b) and the `Matlab` function `MVNRND`, which generates $Y_1, \ldots, Y_L$ from a given mean vector and covariance matrix.

When using $L = 10$, we can run the code overnight without getting a sample. Using $L = 5$, we needed to draw about 852000 samples from the multivariate distribution when repeating the procedure 100 times.

(d) See the accompanying `Matlab` file in the Appendix. Using $L = 5$ needed about 420000 samples from the multivariate distribution to repeat the procedure 100 times.

This method managed to get a sample for $L = 10$ in a reasonable amount of time, or by using around 17000 to 18000 redraws.

(e) The method of part d is more efficient because it can reject individual components, where the method of part c requires rejecting the whole vector of 10 components (or 5 components in the case of $L = 5$).

Problem 3

(a) We can observe,

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_T \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & \ldots & 0 \\ \rho & 1 & 0 & \ldots & \vdots \\ \rho^2 & \rho & 1 & \ldots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho^{T-1} & \rho^{T-2} & \rho^{T-3} & \ldots & 1 \end{pmatrix} \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \varepsilon_T \end{pmatrix}$$

which we will write as $\vec{X} = \mathbf{A}\vec{\varepsilon}$. Each ε_i is an independent standard normal. Hence,

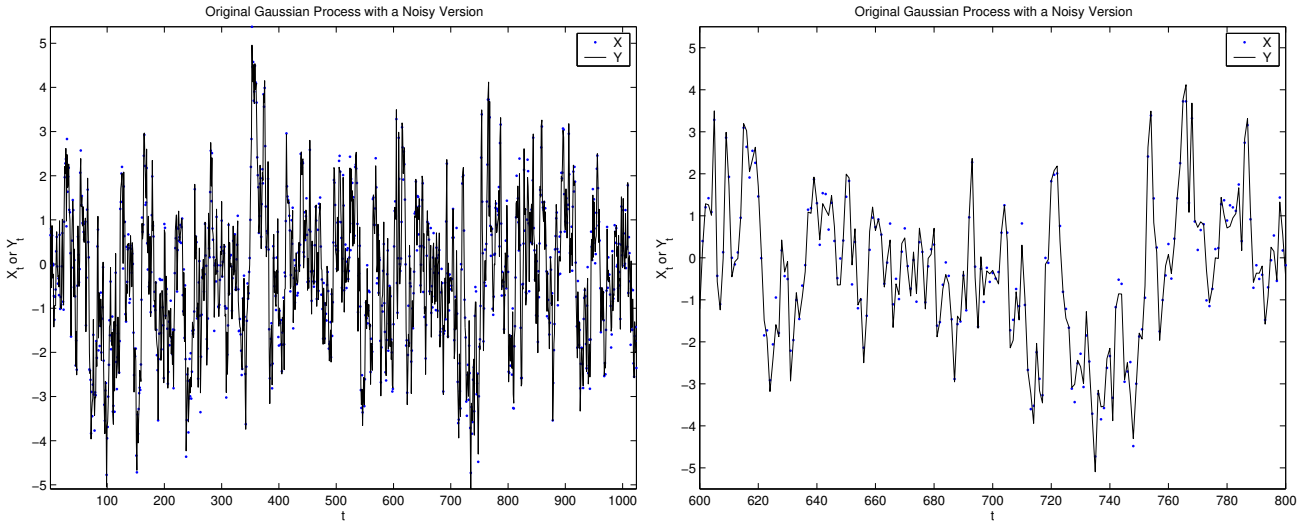
$$\vec{\varepsilon} \sim N(0, I_{T\times T})$$
$$\vec{X} \sim N(0, AA^t)$$

We can inspect the covariance matrix to be $\Sigma = AA^t$. The Σ_{ij} element can be written explicitly as $\rho^{|i-j|}\frac{\rho^{2Min(i,j)}-1}{\rho^2-1}$.

(b) There are several possible ways to sample $\vec{X}$:

- Successively multiply by ρ and add a standard normal.
- Sample $\vec{X}$ by sampling $\vec{\varepsilon}$, which has independent identically distributed $N(0, 1)$ components. We may then left multiply by A.
- Principal Component Analysis: $\Sigma = \Phi D \Phi^t$. The k-th column of Φ is φ_k, the k-th eigenvector. D is the diagonal matrix of eigenvalues λ_k^2 corresponding to each of the φ_k's.

(c) See the accompanying `Matlab` file in the Appendix. We sample $\vec{X}$ via Principal Components and add the normal noise elementwise. The following figures show a sample $\vec{X}$ vector and a noisy version $\vec{Y}$.



(d) See the accompanying `Matlab` file in the Appendix. From class, we seek to find

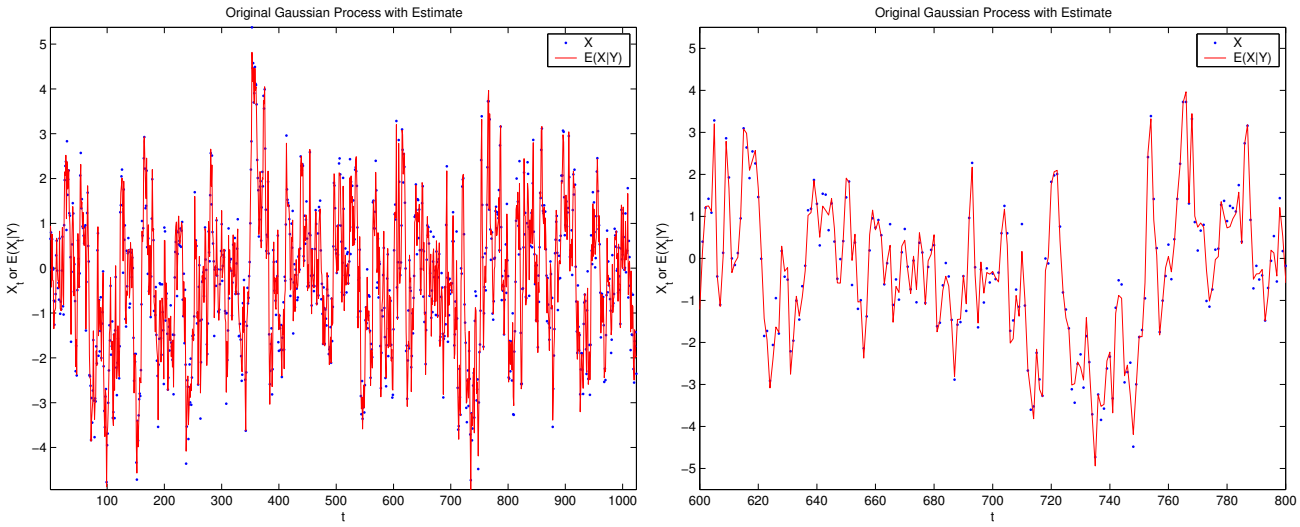
$$E(\vec{X} \mid \vec{Y}) = \sum_{k=1}^{T} \omega_k \langle \vec{Y}, \varphi_k \rangle \varphi_k, \text{ with}$$

$$\omega_k = \frac{\lambda_k^2}{\lambda_k^2 + \sigma^2}$$

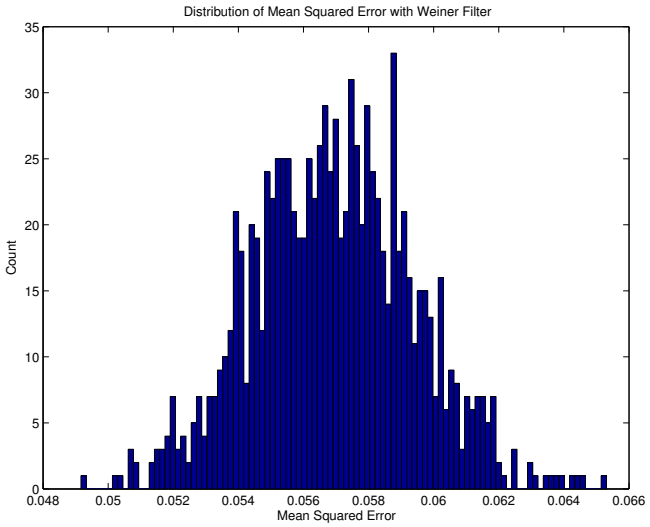
$$\langle \vec{Y}, \varphi_k \rangle = \sum_{t=1}^{T} Y_t \varphi_k(t)$$

Upon inspection, we may interpret $E(\vec{X} \mid \vec{Y})$ as the left multiplication of the matrix Φ against the column vector with components $\omega_k \langle \vec{Y}, \varphi_k \rangle$.

For the parameters $\rho = 0.8$, $\sigma = 0.25$, and $T = 1024$, a sample MSE is 0.0567. The following figures show the estimate of $\vec{X}$ along with $\vec{X}$.



(e) See the accompanying `Matlab` file in the Appendix. The following figure shows the empirical distribution of MSE for 1000 samples.



$$\text{Mean MSE} \approx 0.0569$$
$$\text{Std Dev of MSE} \approx 0.00247$$

(f) See the accompanying `Matlab` file in the Appendix. We seek to calculate $E(\|\vec{X} - E(\vec{X} \mid \vec{Y})\|^2)$. As per lecture, norm is preserved under diagonalization to components. If y_k, θ_k, and w_k are respectively $\langle \vec{Y}, \varphi_k \rangle$, $\langle \vec{X}, \varphi_k \rangle$, and $\langle \vec{W}, \varphi_k \rangle$:

$$\|\vec{X} - E(\vec{X} \mid \vec{Y})\|^2 = \|\omega_k y_k - \theta_k\|^2 = \sum_{k=1}^{T} (\omega_k y_k - \theta_k)^2$$

Also from lecture,

$$y_k = \theta_k + z_k$$
$$N(0, \lambda_k^2 + \sigma^2) = N(0, \lambda_k^2) + N(0, \sigma^2)$$

We need the distribution of the k-th element of the norm. Plugging in the definition of y_k,

$$\begin{aligned} \omega_k y_k - \theta_k &\sim N\big(0, (\omega_k - 1)^2 \lambda_k^2 + \omega_k^2 \sigma^2\big) \\ &\sim \sqrt{(\omega_k - 1)^2 \lambda_k^2 + \omega_k^2 \sigma^2}\; N(0,1) \\ (\omega_k y_k - \theta_k)^2 &\sim \big((\omega_k - 1)^2 \lambda_k^2 + \omega_k^2 \sigma^2\big) \chi_1^2 \end{aligned}$$

The square of a normal distribution follows a Chi-Squared distribution with 1 degree of freedom, denoted χ_1^2. The needed properties are:

$$E(\chi_1^2) = 1$$
$$Var(\chi_1^2) = 2$$

Finally,

$$\begin{aligned} E\left(\frac{1}{T}\sum_{t=1}^{T}(X_t - \hat{X}_t)^2\right) = E\left(\frac{1}{T}\sum_{k=1}^{T}(\omega_k y_k - \theta_k)^2\right) &= \frac{1}{T}\sum_{k=1}^{T}(\omega_k - 1)^2 \lambda_k^2 + \omega_k^2 \sigma^2 \\ &\approx 0.0569258 \end{aligned}$$

The computation of this number is included in the accompanying simulation file. We observe exceptional agreement between observation and theory. The difference is in the 5-th significant digit! Such accuracy should not be expected, and it will likely be worse upon subsequent sets of 1000 trials.

(g) *Bonus Question*: See the accompanying `Matlab` file in the Appendix. We proceed as in the end of part (f).

$$Var\left(\frac{1}{T}\sum_{t=1}^{T}(X_t-\hat{X}_t)^2\right)=\frac{1}{T^2}\sum_{k=1}^{T}Var\big((\omega_k y_k-\theta_k)^2\big)=\frac{1}{T^2}\sum_{k=1}^{T}2\big((\omega_k-1)^2\lambda_k^2+\omega_k^2\sigma^2\big)^2$$
$$\approx 6.355\times 10^{-6}$$

The first equality follows from independence of components, and the second follows by pulling out the multiplicative constant in front of the χ_1^2. As standard deviation is the square root of variance, standard deviation ≈ 0.002521. The agreement of observation and theory is reasonable as the difference occurs in the second significant digit.

Appendix

`Matlab`-code for Problem 2

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script is for running the numerical experiments
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
K = 100;
rho = 0.7;
L = 5;
% Let's run strategies 1 and 2 (from 1(c) and (d) respectively)
% K times
samplesdrawn1 = zeros(1,K);
samplesdrawn2 = zeros(1,K);
for k = 1:K,
  samplesdrawn1(k) = strategy1(rho,L);
  samplesdrawn2(k) = strategy2(rho,L);
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This code is for the sampling with the method described
% in 2(c)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function samplesdrawn = strategy1(rho,L)

y0 = 1;
ylast = 0;

Y2 = [ylast;y0];

% Create the matrix A from part 1(b)
A = zeros(L+2,L+2);
for i=1:L+1,
  for j=1:i+1,
    A(i,j) = rho^(i+1-j);
  end
end
A(L+2,1) = 1;

% Calculate the covariance matrix
```

```
E= A*A';

% Define variables for the covariance block matrices
E11 = E(1:L,1:L);
E12 = E(1:L,[L+1 L+2]);
E21 = E([L+1 L+2],1:L);
E22 = E([L+1 L+2],[L+1 L+2]);
% Find the mean and the variance of the cond. distribution
mu = E12*inv(E22)*Y2;
sigma = E11-E12*inv(E22)*E21;

% We'll use the Matlab function mvnrnd to generate
% Y1,...,YL
samplesdrawn = 0;
while (1), % we break when we get a "good" sample
  tmp = mvnrnd(mu,sigma);
  samplesdrawn = samplesdrawn + 1;
  % check whether the vector satisfies the constraint
  success = 1;
  if tmp(1) < 1 & tmp(L) > 0,
    % have to check whether the vector is ordered
    for l=2:L,
      if tmp(l) >= tmp(l-1),
        % Y(l) < Y(l-1) does not hold, have to get another sample
        success = 0;
        break;
      end
    end
    % we have a sample from the distribution!
  else
    success = 0;
  end
  if success == 1,
    break; % we have a sample from the distribution, so we break
  end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This code is for the sampling with the method described
% in 2(d)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function samplesdrawn = strategy2(rho,L)

% INITIALIZE
Y = zeros(1,L+2);
Y(1) = 1;
for i=2:L+1,
  Y(i) = (L+1-i)/(L+1);
end
% NOTE! Y(j) corresponds to Y_j-1

sigma = 1/(1+rho^2); % This value always stays the same
samplesdrawn = 0;
for n=1:100,
```

```
  for k=2:L+1,
    % Let's calculate the mean and variance
    % for L(Y1|Y0=1,Y2=y2)
    mu = rho/(1+rho^2)*(Y(k-1)+Y(k+1));
    % sample yk
    while 1, % we break the loop when a "good" sample is drawn
      Y(k) = normrnd(mu,sigma);
      samplesdrawn = samplesdrawn+1;
      % check whether the contraints hold
      if Y(k-1) > Y(k) & Y(k) > Y(k+1),
        break;
      end
    end
  end
end
```

Matlab-code for Problem 3

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script is for running the numerical experiments
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

rho=0.8;
T=1024;
E = zeros(T);
error =zeros(1,1000);
sigma = 1/4;

% Calculate the covariance matrix
for i = 1:T
  for j = 1:T
    E(i,j) = rho^(abs(i-j))*(rho^(2*min(i,j))-1)/(rho^2-1);
  end
end
[Phi,D] = eig(E);
d=diag(D);
w = d./(d+sigma^2);

% sampling for 3(c), mse for 3(d) and average mse for 3(e)
% strategy 1 is straightforward implementation, which is easy to
% understand
% strategy 2 is vectorized implementation, which is optimized for speed
% in matlab
strategy = 2;
if strategy == 1
  for n = 1:1000
    X = mvnrnd(zeros(1,T),E);
    Y = X + randn(1,T)*sigma;

    Xhat = (phi*diag(w)*phi'*Y')';
    error(n) = sum((X-Xhat).^2)/T;

    if n==1
```

```
      disp('3(d)');
      disp(sprintf('estimation error is %f',error(1)));
      figure
      plot(1:T,X,'b.',1:T,Y,'k-');
      legend('X','Y');
      ylabel('X_t or Y_t')
      title('Original Gaussian Process with a Noisy Version')
      figure
      plot(1:T,X,'b.',1:T,Xhat,'r-');
      legend('X','E(X|Y)');
      ylabel('X_t or E(X_t|Y)');
      title('Original Gaussian Process with Estimate');
    end
  end
elseif strategy == 2
  % faster vectorized implementation
  X = mvnrnd(zeros(1,T),E,1000);
  Y = X + randn(1000,T)*sigma;

  Xhat = (phi*diag(w)*phi'*Y')';
  error = sum((X-Xhat).^2,2)/T;

  disp('3(d)');
  disp(sprintf('estimation error is %f',error(1)));
  figure
  plot(1:T,X(1,:),'b.',1:T,Y(1,:),'k-');
  legend('X','Y');
  ylabel('X_t or Y_t')
  title('Original Gaussian Process with a Noisy Version')
  figure
  plot(1:T,X(1,:),'b.',1:T,Xhat(1,:),'r-');
  legend('X','E(X|Y)');
  ylabel('X_t or E(X_t|Y)');
  title('Original Gaussian Process with Estimate');
end

% histogram for 3(e)
figure
hist(error,100);
xlabel('Mean Squared Error');
ylabel('Count')
title('Distribution of Mean Squared Error with Weiner Filter');

disp('3(e)')
avg_error = sum(error)/1000
std_error = std(error)

% exact mse in 3(f)

disp('3(f)');
format long;
exact_mse=sum((w-1).^2.*d+w.^2*sigma^2)/T

% standard deviation in 3(g)
```

```
disp('3(g)')
std_mse=sqrt(2*sum(((w-1).^2.*d+w.^2*sigma^2).^2)/T^2)
```