

```

%%%%% Maximum Likelihood Estimation for State-Space Model %%%%%

% Equation Numbers are from Technical Appendix
% The matlab program implements maximum likelihood estimation of state-space model
% for Case Study 2 in the book chapter.

% read in data file (in excel format)
% mytimeseries is a n x nts matrix where
%   n=length of time series, and
%   nts=number of time series

mytimeseries=xlsread(' [Insert file name] ');

% Find the dimension of the data
[n,nts]=size(mytimeseries);

% Set input parameters
niter=50; % Number of iterations
np=4;     % dimension of state vector
nq=2;     % dimension of observation vector
r=18     % no. of replicated time series
np2=np*np;

S0=.02*eye(np,np); % initial variance for the state vector
Phi=[2 0 -1 0; 0 2 0 -1;1 0 0 0; 0 1 0 0]; % Matrix Phi in the state eqt. for Case Study 2
Qf=.005*eye(np,np); % initial error variances in the state eqt.
Af=[1 0 0 0;0 1 0 0]; % Matrix A in the obs. eqt. for Case Study 2
Rf=[.05 0; 0 .05]; % Initial error variance in the obs. eqt.

B=zeros(nq,nq);
C11=zeros(nq,nq);
R22_1=zeros(nq,nq);

x0=zeros(np,n+1);
P0=zeros(np,np);
P00=zeros(np,np);
Pdiff=zeros(np,np);
p0=zeros(np2,n+1);
x1=zeros(np,n+1);
P1=zeros(np,np);
p1=zeros(np2,n+1);
P10=zeros(np,np);
p10=zeros(np2,n+1);
sd=zeros(n,r);
d=zeros(n,r);
Yf=zeros(n,2);
P0_b=zeros(np,np);

%%% Initialize
p0(:,1)=S0(:);

%%% Begin Iterations (E-M Algorithm)

for it=1:niter

    Rsum=zeros(nq,nq);
    Qsum=zeros(np,np);
    x0_b=zeros(np,n+1);
    p0_b=zeros(np2,n+1);

    for ir=1:r
        Yf=D(:,2*ir-1:2*ir);

        % ----- E-step starts here ----- %
        % Initial Mean for the State Vector
        x0(1:2,1)=Yf(1,:);

```

```

x0(3:4,1)=Yf(1,:)' ;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
la=0; % initialize a variable for calculating loglikelihood

%%%%% Forward Recursions (Kalman Filter) %%%%
for ik=2:n+1
% Initialize Observation Parameters
A=Af;
Q=Qf;
y=Yf(ik-1,:)' ;

% Adjust for missing data
i0=find(y>0);
i1=find(y==0);
n0=length(i0);
n1=length(i1);
if n0>0
R(i0,i0)=Rf(i0,i0);
end

if n1>0
y(i1)=zeros(n1,1);
A(i1,:)=zeros(n1,np);
end

% Begin recursions
% Equation (A.1)
x1(:,ik)=Phi*x0(:,ik-1);
P0(:)=p0(:,ik-1);

% Equation (A.2)
P1=Phi*P0*Phi'+Q;
p1(:,ik)=P1(:);

% Equation (A.5)
S=A*P1*A'+R;
K=P1*A'/S; % Kalman Gain

% Equation (A.3)
x0(:,ik)=x1(:,ik)+K*e;

% Equation (A.4)
P0=(eye(np,np)-K*A)*P1;
p0(:,ik)=P0(:);

% Compute filter residual
e=y-A*x1(:,ik);

% Increment -2 log L Equation (4)
la=la+log(det(S))+e'/S*e;

end % End of Forward Recursion (Kalman Filter)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% Backward Recursions (Kalman Smoother) %%%%

% Initialize Lag-One Smoother Covariance Equation (A.9)
P0(:)=p0(:,n);
P10=(eye(np,np)-K*A)*Phi*P0;
p10(:,n+1)=P10(:);

% Backward Recursion starts
for ik=n+1:-1:2

P0(:)=p0(:,ik-1);

```

```

    P1(:)=p1(:,ik);

% Equation (A.8)
J=P0*Phi'/P1;

% Equation (A.10): P_{t-1,t-2}^n
    if ik<n+1 & ik>1
        P10=C10*J';
        p10(:,ik)=P10(:);
    end

    if ik>2
        P10(:)=p10(:,ik);
        P0(:)=p0(:,ik-1);
        C10=P0+J*(P10-Phi*P0);
    end

% Equation (A.6)
    x0(:,ik-1)=x0(:,ik-1)+J*(x0(:,ik)-x1(:,ik));

% Equation (A.7)
    P0(:)=p0(:,ik-1);
    P00(:)=p0(:,ik);
    P1(:)=p1(:,ik);
    P0=P0+J*(P00-P1)*J';
    p0(:,ik-1)=P0(:);

    end % Backward recursion ends here
% -----End of E-Step ----- %

% Accumulate sums of products for update
S00=zeros(np,np);
S10=zeros(np,np);
S11=zeros(np,np);
R=zeros(nq,nq);
yp=zeros(nq,1);
y2=zeros(nq,1);

id=0;
% M-Step starts here (Updating Parameters)
for ik=2:n+1

% For Phi and Q update
    P00(:)=p0(:,ik-1);
    P10(:)=p10(:,ik);
    P0(:)=p0(:,ik);

% Equation (7)
    S00=S00+x0(:,ik-1)*x0(:,ik-1)'+P00;

% Equation (8)
    S10=S10+x0(:,ik)*x0(:,ik-1)'+P10;

% Equation (9)
    S11=S11+x0(:,ik)*x0(:,ik)'+P0;

    y=Yf(ik-1,:);
    i0=find(y>0);
    i1=find(y==0);
    n0=length(i0);
    n1=length(i1);

% Fully observed case
    if n0==nq
        e=y-Af*x0(:,ik);
        % Equation (3.31)
        R=R+(e*e'+Af*P0*Af')/n;
    end
end

```

```

% See Shumway and Stoffer (2000, Chp 4) for formulae need
% to handle missing data via E-M algorithm in State-Space Models
% (Fully unobserved and partially unobserved cases below)
% Ref: Shumway, R.H., & Stoffer, D.S. (2000).
% Time series analysis and its applications. New York: Springer.

% Fully unobserved case
    if n1==nq
        yp=Af*x0(:,ik);
        R=R+Rf/n;
    end

% Partially unobserved
    if n1>0 & n0>0
        B(i1,i0)=Rf(i1,i0)/Rf(i0,i0);

        % Predicted for unobserved part
        yp(i1)=Af(i1,:)*x0(:,ik);

% Observed part
        e(i0)=y(i0)-Af(i0,:)*x0(:,ik);
        y2(i1)=Af(i1,:)*x0(:,ik)+B(i1,i0)*e(i0);
        trm=(Af(i1,:)-B(i1,i0)*Af(i0,:))*P0;
        C11(i0,i0)=e(i0)*e(i0)'+Af(i0,:)*P0*Af(i0,:)' ;

% Increment estimator for covariance R
        R(i0,i0)=R(i0,i0)+C11(i0,i0)/n;
        R(i0,i1)=R(i0,i1)+C11(i0,i0)*B(i1,i0)'/n;
        R(i1,i0)=R(i0,i1)';
        R22_1(i1,i1)=Rf(i1,i1)-B(i1,i0)*Rf(i0,i1);
        R(i1,i1)=R(i1,i1)+(R22_1(i1,i1)+B(i1,i0)*C11(i0,i0)*B(i1,i0)')/n;

    end

    Q_temp=(S11-Phi*S10'-S10*Phi'+Phi*S00*Phi')/n;
    Q(1,1)=Q_temp(1,1); Q(2,2)=Q_temp(2,2);

end % End of M-step

Rsum=Rsum+R/r;
Qsum=Qsum+Q/r;
x0_b=x0_b+x0/r;
p0_b=p0_b+p0/r^2;

end % end of "ir" loop

Rf=Rsum;
Qf=Qsum;

% Calculation of -2 log L %
    if it>(niter-3)
        iteration=it
        m2_log_L=la
        Obs_cov=Rf
        Mod_cov=Qf
    end

end % end of "it" loop

```