```
%%% this file solves Ex 4.1, ECON 5300, Fall 2015
% note:
% - assumes that date and x have been saved in the file data.mat in proper
% format.


%% 0. housekeeping

clc             % clear the commands in the window
clear all;      % clear the workspace

%% 1. load data

load('data.mat','date','x');        % requires that data.mat is stored in the current directory

%% Ex 4.1(c)

% compute empirical averages of x
[T,N]     = size(x);                        % get dimensions of the data
xmean     = nanmean(x);                     % compute time series averages, use NaN ignoring mean
xmean_mat = repmat(xmean,T,1);              % replicate the variable mean T times

% compute the normalized series
xhat  = x./xmean_mat;                       % element-wise division with xmean_mat

% plot figure
scrsz = get(0,'Screensize');                                                    % get screensize
figure('Position',[scrsz(3)/8 scrsz(4)/8 scrsz(3)*3/4 scrsz(4)*3/4])            % 3/4 screensize figure
plot(datenum(date,'yyyyQQ'),log(xhat));                                         % plot normalized series
datetick('x','yyyyQQ');                                                         % convert date axis to proper format
ylabel('log(xhat)'); xlabel('date');                                            % add labels
legend('Household consumption','Household consumption (goods)','Household consumption (services)',...
    'Gross fixed capital formation','Gross fixed capital formation (mainland)','Gross capital formation',...
    'Total exports','Total imports','Gross domestic product','Gross domestic product (mainland)',...
    'Compensation of employees','Total hours worked','Location','Best')
```

```
%%% Ex 4.1 (d)

%% (i)
% estimate parameters of the linear filter

% define some useful variables
short       = sum(isnan(x(:,11)),1)+1;    % indicates first period of non-missing data in series 11 and 12
xhat_short  = xhat(short:T,:);            % choose submatrix such that no NaN's for series 11 and 12
T_short     = size(xhat_short,1);         % first dimension of the submatrix

% prepare inputs for the OLS procedure

% columns 1 to 10
y1 = log(xhat(:,1:10));
X1 = [ones(T,1),(1:T)'];

% columns 11 to 12
% limit the OLS to non-missing observations
y1_short = log(xhat_short(:,11:12));
X1_short = [ones(T_short,1),(short:T)'];

% get the OLS estimate
% beta will be a 2-by-N matrix, so there will be 2 estimates for each
% series

% columns 1 to 10
beta1(:,1:10)  = myols(y1,X1);

% columns 11 to 12
beta1(:,11:12) = myols(y1_short,X1_short);

% compute the trend component
xhat_trend1 = exp(X1*beta1);     % note that we ran the OLS in logs, thus we have to exponentiate for reaching levels

% cyclical component
xhat_cycle1 = exp(log(xhat)-log(xhat_trend1));
```

```
%% (ii)
% estimate parameter of the first difference filter

% prepare inputs for the OLS procedure

% columns 1 to 10
y2 = log(xhat(2:T,1:10))-log(xhat(1:T[redacted],1:10));        % first-difference is the dependent variable
% we loose one observation by taking the difference
X2 = ones(T-1,1);                                              % column of ones to estimate the constant

% columns 11 to 12
% limit the OLS to non-missing observations
y2_short = log(xhat_short([redacted]))-log(xhat_short(1:T_short-1,11:12));
X2_short = ones(T_short-1,1);

% get the OLS estimate
% beta will be a 1-by-N matrix, so there will be 1 estimate for each
% series

% columns 1 to 10
beta2(:,1:10)  = [redacted];

% columns 11 to 12
beta2(:,11:12) = [redacted];

% cyclical component
xhat_cycle2 = exp(log(xhat(2:T,:))-log(xhat(1:T-1,:))-X2*beta2);

%% (iii)

% initialize trend and cycle matrix of the same size as xhat
xhat_trend3 = NaN*ones(size(xhat));
xhat_cycle3 = NaN*ones(size(xhat));

% set smoothing parameter
```

```
lambda = 1600;                              % quarterly data

% estimate the components (in logs!) from the user-written HP-routine

% columns 1 to 10
[log_xhat_trend3,log_xhat_cycle3] = myhpfilter(log(xhat(:,1:10)),lambda);

xhat_trend3(:,1:10) = exp(log_xhat_trend3);
xhat_cycle3(:,1:10) = exp(log_xhat_cycle3);

% columns 11 to 12
[log_xhat_trend3,log_xhat_cycle3] = myhpfilter(log([redacted]),lambda);
xhat_trend3(short:end,11:12) = exp(log_xhat_trend3);
xhat_cycle3(short:end,11:12) = exp(log_xhat_cycle3);

%% Ex 4.1 (e)

% compute standard deviations
sd      = zeros(3,N);
sd(1,:) = nanstd(log(xhat_cycle1));         % use standard deviation that ignores NaN's in short series
sd(2,:) = nanstd(log(xhat_cycle2));
sd(3,:) = nanstd(log(xhat_cycle3));

% print table
filter  = {'Linear','First-difference','HP'};
varname = {'     Filter','x1','x2','x3','x4','x5','x6','x7','x8','x9','x10','x11','x12'};
fprintf('Standard deviations:\n \n' )
fprintf('%-16s %5s %5s %5s %5s %5s %5s %5s %5s %5s %5s %6s %5s %5s \n', varname{:});
fprintf('================= \n')
for i=1:3
    fprintf('%-16s: %5.2f %5.2f %5.2f %5.2f %5.2f %5.2f %5.2f %5.2f %5.2f %5.2f %5.2f %5.2f \n',...
        filter{i},sd(i,:));
end

%%  Ex 4.1 (f)
```

```
% compute cross correlations
s        = 5;          % number of lags/leads
corr_mat = zeros(N,2*s+1);
for n=1:N
    if n<=10    % long series
        corr_mat(n,:) = xcorr(log(xhat_cycle3(:,10)),log(xhat_cycle3(:,n)),s,'coeff')';
    else        % short series
        corr_mat(n,:) = xcorr(log(xhat_cycle3(short:end,10)),log(xhat_cycle3(short:end,n)),s,'coeff')';
    end
end

% print table
varname = {'x1','x2','x3','x4','x5','x6','x7','x8','x9','x10','x11','x12'};
lags    = {'Series','-5','-4','-3','-2','-1','0','1','2','3','4','5'};
fprintf('Cross-correlations:\n \n');
fprintf('%-7s %5s %6s %6s %6s %6s %6s %6s %6s %6s %6s %6s %6s \n', lags{:});
fprintf('======= \n');
for i=1:N
    fprintf('%-6s: %6.2f %6.2f %6.2f %6.2f %6.2f %6.2f %6.2f %6.2f %6.2f %6.2f %6.2f \n',...
        varname{i},corr_mat(i,:));
end

%% replicate Kydland and Prescott (1990, Chart 3)

% plot figure
scrsz = get(0,'Screensize');                                          % get screensize
figure('Position',[scrsz(3)/8 scrsz(4)/8 scrsz(3)*3/4 scrsz(4)*3/4])  % 3/4 screensize figure
plot(datenum(date,'yyyyQQ'),[log(xhat_cycle3(:,10)),log(xhat_cycle3(:,11))]);
datetick('x','yyyyQQ');                                               % convert date axis
ylabel('Deviation from trend (in %)'); xlabel('date');
xlim([datenum(date{short},'yyyyQQ'),datenum(date{end},'yyyyQQ')]);    % limit x-axis to 1995Q1-2014Q2
legend('GDP (mainland)','Hours Worked','Location','Best');
```

```
function beta = myols(y,X)
%% returns the OLS estimate for the generic equation: y=X*beta+e
% y is the matrix of dependent variables
% X is the matrix of regressors

% a few assertions:
T = size(y,1);

if (T~=size(X,1))
    error('Error: dimension of matrices y and X are inconsistent.');
end

% return the OLS estimate
% the backslash divide "\" is faster than taking the inverse
beta = [redacted]

end
```

```
function [trend,cycle] = myhpfilter(y,lambda)
%% HP-filters the series y and returns its trend and cycle components
% let:
% - N be the total number of variables
% - T be the number of time observations
% y is the data matrix with dimensions [TxN]
% lambda is the smoothing parameter

% get dimension of the data, y
[T,N] = size(y);

% assertions:
if T<3
    error('Error: T<3 detected, not enough data points to run the filter.');
elseif sum(sum(isnan(y)))>0
    error('Error: NaN''s in the data matrix detected.');
elseif T<N
    display('Warning: T<N detected, please make sure data matrix is provided in the format [TxN].');
end

% setup the main diagonal elements of the matrix A
d = repmat([1,-4,6,-4,1],T,1);

% set up matrix A, by filling up diagonal elements
A = diag(d(1:end-2,1),-2)+diag(d(1:end-1,2),-1)+diag(d(:,3),0)+c

% manually correct the irregular entries in the first two and the last two
% rows
A(1,1) = 1;
A(1,2) = -2;
A(2,1) =
A(2,2) =

A(end,end) =
A(end,end-1) =
A(end-1,end) =
```

```
A(end-1,end-1) = 5;

% the backslash divide "\" is faster than taking the inverse
trend = (lambda*A+diag(ones(T,1),0))\y;
cycle = y-trend;

end
```