ModYUKE.model

```
!variables:transition

    LGDP, LGDP_BAR, G, Y, GROWTH4, GROWTH, GROWTH4_BAR, GROWTH_BAR
    PIE4, PIELTE
    UNR, UNR_BAR, UNR_GAP, UNR_G
    CAPU, CAPU_BAR, CU_G, CAPU_GAP,
    'Population' LN, 'Population Growth' DLN,
    'Log Participation Rate' LP, 'Log Participation Rate Trend' LP_BAR,
    'Log Labor Force Trend' LLF_BAR, 'Log Labor Force' LLF,
    'Labor Force Trend Growth' DLLF_BAR, 'Labor Force Growth' DLLF,
    'Log Employment' LE, 'Employment Growth' DLE, 'Y/Y Employment Growth' DLE4,
    'Log Employment Trend' LE_BAR, 'Employment Trend Growth' DLE_BAR,
    'Y/Y Employment Trend Growth' DLE4_BAR,
    'Avg Labor Productivity' APL, 'Avg Labor Productivity Growth' DAPL,
    'Y/Y Avg Labor Productivity Growth' DAPL4,
    'Avg Labor Productivity Trend' APL_BAR, 'Avg Labor Productivity Trend Growth' DAPL_BAR,
    'Y/Y Avg Labor Productivity Trend Growth' DAPL4_BAR,
    'Participation gap' LP_GAP


!shocks:transition

    RES_APL_BAR, RES_G, RES_Y
    RES_PIE, RES_PIELTE
    RES_UNR_BAR, RES_UNR_G, RES_UNR_GAP
    RES_CAPU_BAR, RES_CU_G, RES_CAPU_GAP
    'Population Growth Shock' RES_DLN
    'Participation Rate Gap Shock' RES_LP
    'Participation Rate Trend Shock' RES_LP_BAR

!parameters

    lab_share, tau, growth_ss, rho1, rho2
    beta, omega
    parhist, lambda, unr_ss, alpha, phi1, phi2
    delta, kappa1, kappa2
    'Steady-State Population Growth' popgrowth_ss,
    'Steady-State Participation Rate' part_ss

    'Population Growth Dynamics' zeta
    'Discouraging Effect on Participation' theta2
    'Participate Rate Dynamics' eta
    theta1

!equations:transition

    % Output.
    LGDP = LGDP_BAR + Y;
    Y = rho1*Y{-1} - rho2/100*(PIE4{-1} - PIELTE{-1}) + RES_Y;
```

```
GROWTH4 = LGDP - LGDP{-4};
GROWTH = 4*(LGDP - LGDP{-1});
GROWTH4_BAR = LGDP_BAR - LGDP_BAR{-4};
GROWTH_BAR = 4*(LGDP_BAR - LGDP_BAR{-1});

% Inflation.
PIE4 = PIE4{-1} + beta*Y + omega*(Y - Y{-1}) + RES_PIE;
PIELTE = PIELTE{-1} + RES_PIELTE;

% Unemployment.
UNR = UNR_BAR - UNR_GAP;
UNR_BAR = UNR_BAR{-1} + UNR_G - parhist/100*Y{-1} - lambda/100*(UNR_BAR{-1} - unr_ss) + RES_UNR_BAR;
UNR_G = (1-alpha)*UNR_G{-1} + RES_UNR_G;
UNR_GAP = phi1*UNR_GAP{-1} + phi2*Y + RES_UNR_GAP;

% Capacity untilization.
CAPU = CAPU_BAR + CAPU_GAP;
CAPU_BAR = CAPU_BAR{-1} + CU_G + RES_CAPU_BAR;
CU_G = (1 - delta)*CU_G{-1} + RES_CU_G;
CAPU_GAP = kappa1*CAPU_GAP{-1} + kappa2*Y + RES_CAPU_GAP;

% Average productivity of labor.
APL_BAR = APL_BAR{-1} + G/4 + RES_APL_BAR;
G = (1-tau)*G{-1} + tau*growth_ss + RES_G;
DAPL_BAR = 4*(APL_BAR - APL_BAR{-1});
DAPL4_BAR = APL_BAR - APL_BAR{-4};
DAPL = 4*(APL - APL{-1});
DAPL4 = APL - APL{-4};

% Potential output is residually determined from this equation.
APL_BAR = LGDP_BAR - LE_BAR;
APL = LGDP - LE;

% Population.
DLN = (1-zeta)*DLN{-1} + zeta*popgrowth_ss + RES_DLN;
DLN = 4*(LN - LN{-1});

% Labor market participation.
LP_GAP = theta1*LP_GAP{-1} -100/part_ss*theta2*(UNR - UNR_BAR) + RES_LP;
LP_GAP = LP - LP_BAR;
LP_BAR = (1-eta)*LP_BAR{-1} + eta*100*log(part_ss/100) + RES_LP_BAR;

% Labor force trend.
LLF_BAR = LN + LP_BAR;
DLLF_BAR = 4*(LLF_BAR - LLF_BAR{-1});
LLF = LN + LP;
DLLF = 4*(LLF - LLF{-1});

% Employment trend.
% Mixed first-order expansion of UNR_BAR = 1 - E_BAR/LF_BAR;
```

```
    UNR_BAR - unr_ss = -(1-unr_ss/100)*(LE_BAR - LLF_BAR - 100*log(1-unr_ss/100));
    DLE_BAR = 4*(LE_BAR - LE_BAR{-1});
    DLE4_BAR = LE_BAR - LE_BAR{-4};
    UNR - unr_ss = -(1-unr_ss/100)*(LE - LLF - 100*log(1-unr_ss/100));
    DLE = 4*(LE - LE{-1});
    DLE4 = LE - LE{-4};

!variables:measurement

    GROWTH_, PIE4_, D_CAPU_, UNR_, PIE4_PIELTE_
    'Log Labour Force' LLF_, 'Log Participation Rate' LP_

    %These variables are added for tuning
    LP_BAR_
    DAPL_BAR_

!shocks:measurement

    RES_LLF_
    RES_DAPL_BAR_

!equations:measurement

    GROWTH_ = GROWTH;
    PIE4_ = PIE4;
    D_CAPU_ = CAPU - CAPU{-1};
    UNR_ = UNR;
    PIE4_PIELTE_ = PIE4 - PIELTE;
    LLF_ = LLF + RES_LLF_;
    LP_ = LP;

    %These equations are added for tuning
    LP_BAR_ = LP_BAR;
    DAPL_BAR_ = DAPL_BAR + RES_DAPL_BAR_;
```

readDefinitions

Set important dates and choose country

```
% |endhist| is the last period used for the estimation of the model
% parameters
endhist = qq(2009,4);

% |endproj| is the date until which the Kalman smoother is run
endproj = qq(2017,4);

% Type the two-letter country code of the country you would like to
% analyze. Make sure that file named data_|country|.csv containing data
% for that country exists in |InputData| subdirectory.
country = 'US';

% Choose a name for the version of the results
%version = 'VER0';
%version = 'VER1';
%version = 'VER2';
version = 'VER3';
```

readModel.m – Read and parametrize the model.

This m-file reads and parses the model code file `ModYUK.model`, and creates a model object in Matlab memory. The model object is then available for us to run the various exercises.

Contents

- Clearing the workspace
- First-pass parameters
- Read the model code, assign parameters and create a model object
- Alternative ways of assigning parameters
- Solve the model, and compute steady-state.
- Save the model object for future use.

Clearing the workspace

Clearing the workspace at the beginning of each new task is a really good idea.

```
clear all;
close all;
home();
```

First-pass parameters

We create a parameter database (note we use the word 'database' for a Matlab 'struct') and use it then to assign the model's parameters. Alternatively, you could first read the model, and only then create a parameter database, and assign the parameters. Whichever suits you better.

There is a parameter controlling the std deviation of each shock, called `'std_'` and the name of the shock. You needn't declare these in the model code file, they're created automatically, but you can treat them exactly the same way as the other parameters.

Finally, note that the values of parameters set here are immaterial for those parameters that are later estimated in `estimateParams.m`

```
% A new empty database (struct).
P = struct();

% Steady-state parameters.
P.unr_ss = 5.454376;
P.growth_ss = 1.5;
P.lab_share = 0.7;
P.popgrowth_ss = 1;
P.part_ss = 64;

% Transitory parameters.
P.alpha = 0.4630;
```

```
P.beta = 0.1485;
P.omega = 0.1547;
P.rho1 = 0.8687;
P.kappa1 = 0.4056;
P.phi1 = 0.6298;
P.phi2 = 0.2628;
P.tau = 0.2208;
P.delta = 0.4970;
P.kappa2 = 1.2899;
P.parhist = 3.37;
P.rho2 = 5.67;
P.lambda = 5;
P.zeta = 0.1;
P.eta = 0.02;
P.theta2 = 0.1;
P.theta1 = 0.9;

% Standard deviations of shocks
P.std_RES_APL_BAR = 1;
P.std_RES_Y = 0.7427;
P.std_RES_G = 1.1305;
P.std_RES_UNR_GAP = 0.1333;
P.std_RES_UNR_BAR = 0.0594;
P.std_RES_UNR_G = 0.0903;
P.std_RES_CAPU_GAP = 0.4137;
P.std_RES_CAPU_BAR = 0.2490;
P.std_RES_CU_G = 0.0715;
P.std_RES_PIE = 0.5349;
P.std_RES_PIELTE = 0.0955;
P.std_RES_DLN = 0.15;
P.std_RES_LP = 0.1;
P.std_RES_LP_BAR = 0.1;
P.std_RES_LLF_ = 1;
P.std_RES_DAPL_BAR_ = 1;
```

Read the model code, assign parameters and create a model object

The `model` function reads the model code file specified as the first input argument, and creates a model object `m` in Matlab memory.

- The 'linear'(=true) option indicates that this is a linear model (and some more efficient algorithms can be used).
- The 'assign'(=P) option allows us to assign the model's parameters straight away when creating the model object. Alternatively, the parameters may be assigned after the model object has been created.

```
m = model(...
    'ModYUKE.model',...
    'linear',true,...
    'assign',P);
display(m);
```

```
m =

linear model object: 1 parameterisation(s)
solution(s) available for no parameterisation(s)
user data: empty
```

Alternative ways of assigning parameters

What follows is alternative syntax for assigning or changing parameters. Remember to re-solve the model, and re-compute the steady state whenever you change some of the parameters (not needed, though, when you only change std deviations). Note that every time you use the `estimate` function the model is re-solved. Therefore, it is not imperative that we solve the model below. It is, however, good practice and we do it.

```
m = assign(m,P);
m = assign(m,'lambda',5,'tau',0.2208);
m.lambda = 5;
m.tau = 0.2208;
```

Solve the model, and compute steady-state.

```
m = solve(m);
m = sstate(m);
display(m);
```

```
m =

linear model object: 1 parameterisation(s)
solution(s) available for a total of 1 parameterisation(s)
user data: empty
```

Save the model object for future use.

Saved using the IRIS `savestruct` function.

```
114: savestruct('model.mat',m);
```

readData.m – Read data from a CSV file.

Contents

- Clear the workspace
- Define dates and the country of interest.
- Download data from CSV file and load model
- Create measurement variables
- Check if any measurement variables are missing from the database
- Save data structure for future use.

Clear the workspace

```
clear('all');
close('all');
home();
```

Define dates and the country of interest.

```
readDefinitions;
```

Download data from CSV file and load model

```
d = dbload(sprintf('InputData/data_%s.csv',country));
m = loadstruct('model.mat');
```

Create measurement variables

```
d.GROWTH_ = 4*diff(d.LGDP);
d.PIE4_ = d.PIE4;
d.D_CAPU_ = diff(d.CAPU);
d.UNR_ = d.UNR;
d.PIE4_PIELTE_ = d.PIE4 - d.PIELTE;
d.LP_ = 100*log(d.P/100);
d.LLF_ = 100*log(d.LF);
```

Check if any measurement variables are missing from the database

```
ylist = get(m,'ylist');
list = fieldnames(d);
missing = ylist - list;
if isempty(missing)
   disp('All measurement variables available.');
else
   disp('Measurement variables missing from the database:');
   disp(missing);
end
```

```
Measurement variables missing from the database:
    'LP_BAR_'    'DAPL_BAR_'
```

Save data structure for future use.

```
savestruct('data.mat',d);
```

estimateParams.m – Estimate parameters using maximum regularised likelihood.

Contents

- Clearing the workspace
- Load data and model
- Read important dates
- Display some summary information
- Adding priors on unobservables
- Some housekeeping
- Prior database.
- Estimate model.
- Run Kalman smoother and projections.
- Correct unit-root variables for an implicit constant.
- Save results in a country- and version- specific .mat file.

Clearing the workspace

```
clear all;
close all;
home();
```

Load data and model

```
d = loadstruct('data.mat');
m = loadstruct('model.mat');
```

Read important dates

```
readDefinitions;
```

Display some summary information

```
fprintf('%s...\n',country);
display(version);
starthist = get(d.GROWTH_,'start');
fprintf('start date = %s\n',dat2char(starthist));
fprintf('unr_ss = %g, growth_ss = %g, lab_share = %g\n',m.unr_ss,m.growth_ss, m.lab_share);
```

```
US...

version =

VER3

start date = 1967Q2
unr_ss = 5.45438, growth_ss = 1.5, lab_share = 0.7
```

Adding priors on unobservables

```
d.LP_BAR_ = tseries(starthist,100*log(60/100)); %initialize trend participation
d.DAPL_BAR_ = tseries(starthist:endhist,m.growth_ss*ones(endhist-starthist+1,1));
m.std_RES_DAPL_BAR_ = std(d.GROWTH_)/3; %uncertainty about your prior
```

Some housekeeping

```
if ~(exist(sprintf('Results/Version%s',version),'dir') == 7)
    mkdir(sprintf('Results/Version%s',version))
end

%Write a short description of current version to a text file
fid = fopen(sprintf('Results/Version%s/Description%s.txt',version,version),'w');
fprintf(fid,'Adding a prior on DAPL_BAR. \r\n'); %Notepad doesn't recognize \n as new line
fclose(fid);

% Save database to current version folder for reference purposes
savestruct(sprintf('Results/Version%s/Data%s.mat',version,version),d)
```

Prior database.

The prior for each parameter is a truncated normal distribution. Each parameter is assigned four numbers:

[prior mode, lower bound, upper bound, 1/(prior std dev before penalty)^2]

```
E = struct();

% Priors on model parameters
E.alpha   = [0.5, 0.1,  1,   1/0.05^2 ];
E.beta    = [0.4, 0.1,  Inf, 1/0.1^2 ];
E.omega   = [0.5, 0,    5,   1/0.1^2 ];
E.rho1    = [0.8, 0.1,  1,   1/0.05^2 ];
E.kappa1  = [0.1, 0.01, 1,   1/0.2^2 ];
```

```
E.phi1    = [0.8, 0,    1,   1/0.05^2 ];
E.phi2    = [0.3, 0,    1,   1/0.05^2 ];
E.tau     = [0.1, 0,    1,   1/0.05^2 ];
E.delta   = [0.5, 0,    1,   1/0.05^2 ];
E.kappa2  = [1.5, 0,    3,   1/0.5^2 ];
E.parhist = [5,   0,    100, 1/1^2 ];
E.rho2    = [5,   0,    100, 1/1^2 ];
E.lambda  = [1,   0.01, 100, 1/1^2 ];
E.zeta    = [0.1, 0,    1,  1/0.025^2];
E.eta     = [0.02,0,    1,   1/0.005^2];
E.theta2  = [0.1, 0,    1,   1/0.025^2];
E.theta1  = [0.9, 0,    1,   1/0.025^2];

% Priors on standard deviations of transition shocks
E.std_RES_Y        = [1,     0.01, 3, 1/0.1^2 ];
E.std_RES_G        = [1,     0.01, 3, 1/0.1^2 ];
E.std_RES_UNR_GAP  = [0.5,   0.01, 3, 1/0.1^2 ];
E.std_RES_UNR_BAR  = [0.02, 0.005, 3, 1/0.005^2 ];
E.std_RES_UNR_G    = [0.02, 0.005, 3, 1/0.005^2 ];
E.std_RES_CAPU_GAP = [0.4,   0.01, 3, 1/0.1^2 ];
E.std_RES_CAPU_BAR = [0.25,  0.01, 3, 1/0.05^2 ];
E.std_RES_CU_G     = [0.075, 0.01, 3, 1/0.01^2 ];
E.std_RES_PIE      = [0.5,   0.01, 3, 1/0.1^2 ];
E.std_RES_PIELTE   = [0.3,   0.01, 3, 1/0.1^2 ];
E.std_RES_APL_BAR  = [1/4,   0.01, 3, 1/(0.1/4)^2 ];
E.std_RES_DLN      = [0.15,   0.01, 3, 1/(0.15/4)^2 ];
E.std_RES_LP       = [0.15,   0.01, 3, 1/(0.15/4)^2 ];
E.std_RES_LP_BAR   = [0.1,    0.01, 3, 1/(0.1/4)^2 ];
```

Estimate model.

The call to the `estimate` function below uses the so-called maximum regularized likelihood estimation method (Ljung, 1999 – see presentation) to calculate estimates of the model parameters.

The output arguments of the `estimate` function are

- `Estar`, database with point estimates of model parameters
- `obj`, value of objective function (minus log-likelihood or weighted sum of prediction errors)
- `Grad`, Gradient at optimum,
- `Hess`, Hessian of log-posterior, and Hessian of log-prior (or penalty function) at optimum,
- `mstar`, a model object with parameter estimates updated

The input arguments of the `estimate` function are

- `m`, a model object to be estimated
- `d`, a database with measurement variables
- `starthist:endhist`, date range for estimation
- `E`, prior database
- `penalty`, a scalar that governs the overall tightness of the priors on the parameters (0 = maximum likelihood)

estimateParams

```
p = 0.1;
[Estar,obj,Grad,Hess,mstar] =...
    estimate(m,d,starthist:endhist,E,...
    'penalty',p,'tolx',1e-2,'tolfun',1e-2);
```

Iter	F-count	f(x)	Max constraint	Line search steplength	Directional derivative	First-order optimality	Procedure
0	32	1677.43	-0.015				
1	72	1666.34	-0.01992	0.00391	-1.58e+003	5.62e+003	
2	108	1576.21	-0.01868	0.0625	-472	8.29e+003	
3	145	1542.26	-0.01809	0.0313	-472	2.3e+003	
4	183	1535.44	-0.01781	0.0156	-171	423	
5	218	1506.32	-0.01558	0.125	-111	1.51e+003	
6	254	1476.61	-0.01461	0.0625	-43.4	1.38e+003	
7	293	1475.33	-0.0145	0.00781	-41.3	1.36e+003	
8	329	1457.26	-0.01359	0.0625	-141	731	
9	365	1434.23	-0.01274	0.0625	-104	625	
10	401	1425.32	-0.01194	0.0625	-83.8	563	
11	436	1422.7	-0.01045	0.125	-74.4	789	
12	472	1410.1	-0.009798	0.0625	-118	1.31e+003	
13	513	1409.69	-0.009779	0.00195	-41.4	1.34e+003	
14	550	1401.29	-0.009473	0.0313	-45	445	
15	590	1400.17	-0.01334	0.00391	-19.7	994	
16	625	1391.45	-0.01167	0.125	-62.1	1.68e+003	
17	662	1386.31	-0.01135	0.0313	-27	239	
18	698	1384.12	-0.01082	0.0625	-51.5	528	
19	735	1378.99	-0.01048	0.0313	-14.2	857	
20	771	1370.46	-0.009827	0.0625	-50.5	933	
21	807	1361.48	-0.01079	0.0625	-53.2	146	
22	843	1353.74	-0.01163	0.0625	-19	276	
23	880	1351.35	-0.01188	0.0313	-42.1	392	
24	920	1350.93	-0.01194	0.00391	-31.4	433	
25	956	1346.4	-0.01192	0.0625	-25.8	332	
26	993	1345.4	-0.01154	0.0313	-16.4	726	
27	1030	1344.43	-0.01184	0.0313	-13.1	424	
28	1067	1343.31	-0.01227	0.0313	-15.5	441	
29	1105	1342.63	-0.01208	0.0156	-6.52	1.88e+003	
30	1144	1342.5	-0.01208	0.00781	-5.62	311	
31	1182	1342.43	-0.01189	0.0156	-8.91	209	
32	1219	1341.59	-0.01152	0.0313	-9.84	1.01e+003	
33	1257	1341.21	-0.01134	0.0156	-23.5	792	
34	1294	1340.92	-0.01116	0.0313	-8.87	83.9	
35	1332	1340.71	-0.01169	0.0156	-4.18	229	
36	1371	1340.68	-0.01171	0.00781	-3.19	264	
37	1409	1340.62	-0.01177	0.0156	-2.42	266	
38	1446	1340.52	-0.0114	0.0313	-9.07	293	
39	1483	1340.44	-0.01104	0.0313	-3.19	415	

```
   40   1520      1340.43      -0.0107       0.0313        -5.76          282
   41   1559      1340.41     -0.01104      0.00781        -4.44         43.5
Optimization terminated: directional derivative predicts change in
  objective value less than options.TolFun and maximum constraint violation
  is less than options.TolCon.
No active inequalities.
```

Run Kalman smoother and projections.

A Kalman smoother yields the estimates of the unobservables based on the whole sample.

The output arguments of the `filter` function are

- `m1`, a model object with estimates of the unobservables
- `smooth`, a struct with `mean` and `std` databases containing the results of the Kalman smoother
- `se2`, the estimate of the overall variance scaling factor (when the `'relative'` option is set to true, which is by default)
- `pe`, a database with prediction errors for the measurement variables

The input arguments of the `filter` function are

- `mstar`, a model object to be filtered
- `d`, a database with measurement variables
- `starthist:endproj`, date range for filtration
- `'ahead'`,(=12), maximum number of steps ahead for which predictions of measurement variables are calculated

```
[m1,smooth,se2,delta,pe] = filter(mstar,d,starthist:endproj,'ahead',12);
f = smooth.mean;
f.std = smooth.std;
```

Correct unit-root variables for an implicit constant.

```
%This is done to retrieve the correct level of unit root variables
%observed in growth rates

date = get(d.LGDP,'last');
if ~isnan(date)
    add = d.LGDP(date) - f.LGDP(date);
    f.LGDP = f.LGDP + add;
    f.LGDP_BAR = f.LGDP_BAR + add;
end
date = get(d.PIE4,'last');
if ~isnan(date)
    add = d.PIE4(date) - f.PIE4(date);
    f.PIE4 = f.PIE4 + add;
    f.PIELTE = f.PIELTE + add;
```

```
end
date = get(d.CAPU,'last');
if ~isnan(date)
    add = d.CAPU(date) - f.CAPU(date);
    f.CAPU = f.CAPU + add;
    f.CAPU_BAR = f.CAPU_BAR + add;
end
```

Save results in a country- and version- specific .mat file.

```
dirs = {'Estimates'};
for ii = 1 : numel(dirs)
    if exist(sprintf('Results/Version%s/%s',version,dirs{ii}),'dir') ~= 7
        mkdir(sprintf('Results/Version%s/%s',version,dirs{ii}));
    end
end

filename = sprintf('Results/Version%s/Estimates/%s%s.mat',version,country,version);
fprintf('Saving %s.\n',filename);
save(filename,'f','mstar','Hess','Estar','E','starthist','p', 'pe');
```

```
Saving Results/VersionVER3/Estimates/USVER3.mat.
```

printReport.m – creates a detailed report of the results

Contents

- Clear workspace.
- Version to report. BE SURE TO MAKE CHANGES TO VERSION DESCRIPTION !!!
- Check for output directories.
- Create new empty report.
- Include model code listing.
- Add economic activity gaps graph.
- Add output gap graph.
- Add potential growth graph.
- Add YoY potential growth graph.
- Add productivity QoQ
- Add productivity YoY
- Add labor productivity and output trend growth
- Add unemployment rate graph.
- Add employment trend growth
- Add Y/Y employment trend growth
- Add participation graph
- Add labor force and population growth
- Add capacity utilization graph.
- Add relative variances graph.
- Create and include file with parameter estimate table.
- Create and include file with calibrated parameter table.
- Create and include file with RMSE table.
- Create and include file with revisions table.
- Add real-time graph.
- Add real-time HP graph.
- Compile PDF.
- Save output database as CSV.
- Housekeeping

Clear workspace.

Read definitions of dates and the list of countries.

```
clear('all');
close('all');
home();

readDefinitions;

%add the path to functions used througout the file to do certain tasks
addpath Functions/
```

printReport

Version to report. BE SURE TO MAKE CHANGES TO VERSION DESCRIPTION !!!

Each time you run `estimateParams`, the results are saved under a new version number. To access and report the correct results, you need to change the following line accordingly.

```
%version = 'VER0';
%version = 'VER1';
%version = 'VER2';
version = 'VER3';

fprintf('Version to report: %s.\n',version);
```

```
Version to report: VER3.
```

Check for output directories.

The result files will be moved to `Reports`, `Graphs`, `OutputData` `TexFiles` directories. Make sure they exist.

```
dirs = {'Reports','Graphs','Reports','OutputData','TexFiles'};
for i = 1 : numel(dirs)
    if exist(sprintf('Results/Version%s/%s',version,dirs{i}),'dir') ~= 7
        mkdir(sprintf('Results/Version%s/%s',version,dirs{i}));
    end
end

fprintf('%s...\n',country);
load(sprintf('Results/Version%s/Estimates/%s%s.mat',version,country,version),...
    'f','mstar','Hess','Estar','E','starthist','p', 'pe');

gray = 0.3*[1,1,1];
generalstyle = struct();
generalstyle.axes.xgrid = 'on';
generalstyle.axes.ygrid = 'on';
generalstyle.line.lineWidth = 2.5;
generalstyle.line.color = {gray,'blue','red','green'};
generalstyle.line.lineStyle = {'-','-','-','-'};
generalstyle.patch.faceColor = [0.85,0.85,1];
generalstyle.patch.edgeColor = 'none';
generalstyle.bar.faceColor = 0.6*[1,1,1];
generalstyle.bar.edgeColor = 'none';
generalstyle.bar.barWidth = 1;
```

```
US...
```

Create new empty report.

```
x = report.new();
```

Include model code listing.

```
x.include('The Model','MODYUKE.model','verbatim',true)

x.clearpage();
```

Add economic activity gaps graph.

```
thisstyle = generalstyle;
thisstyle.line.color = {'blue',[0,0.5,0],'red'};
thisstyle.line.lineStyle = '-';

x.figure('Economic activity gaps',...
    'range',starthist:endproj,'saveas',sprintf('%sGraphEconomicActivityGaps%s',country,version));

x.graph('','legend',true,'zeroline',true,'style',thisstyle,'highlight',endhist+1:endproj);
x.series('Inflation',f.PIE4,'plotfunc',@bar);
x.series('Output gap',f.Y);
x.series('Unemployment gap',f.UNR_GAP);
x.series('Capacity utilisation gap',f.CAPU_GAP);

x.clearpage();
```

Add output gap graph.

```
x.figure('Output Gap +/- 2 std devs (w/o model or parameter uncertainty)',...
    'range',starthist:endproj,'saveas',sprintf('%sGraphOutputGap%s',country,version));

x.graph('','legend',true,'highlight',endhist+1:endproj,'zeroline',true,'style',generalstyle);
x.series('Inflation',f.PIE4);
x.band('Output gap',f.Y,-2*f.std.Y,2*f.std.Y);

x.clearpage();
```

Add potential growth graph.

```
x.figure('Output Growth (Q/Q Annualized) +/- 2 std devs (w/o model or parameter uncertainty)',...
    'range',starthist:endproj,'saveas',sprintf('%sGraphPotentialGrowth%s',country,version));

x.graph('','legend',true,'style',generalstyle,'highlight',endhist+1:endproj);
x.series('Actual growth',f.GROWTH);
x.band('Potential growth +/- 2 std devs',f.GROWTH_BAR,-2*f.std.GROWTH_BAR,2*f.std.GROWTH_BAR);
x.clearpage();
```

Add YoY potential growth graph.

```
x.figure('Output Growth (Y/Y) +/- 2 std devs (w/o model or parameter uncertainty)',...
    'range',starthist:endproj,'saveas',sprintf('%sGraphPotentialGrowthYoY%s',country,version));

x.graph('','legend',true,'style',generalstyle,'highlight',endhist+1:endproj);
x.series('Actual',f.GROWTH4);
x.band('Potential +/- 2 std devs',f.GROWTH4_BAR,-2*f.std.GROWTH4_BAR,2*f.std.GROWTH4_BAR);
x.clearpage();
```

Add productivity QoQ

```
x.figure('Growth of Labor Productivity (Q/Q Annualized)','range',starthist:endproj,...
    'highlight',endhist+1:endproj,'saveas',sprintf('%sProdQoQTrendGrowth%s',country,version));

x.graph('','legend',true,'zeroline',true,'style',generalstyle);
x.series('Actual',f.DAPL);
x.band('Equilibrium +/- 2 std devs',f.DAPL_BAR,-2*f.std.DAPL_BAR,2*f.std.DAPL_BAR);
x.series('Steady-state productivity growth',tseries(starthist:endproj,f.growth_ss));
x.clearpage();
```

Add productivity YoY

```
x.figure('Growth of Labor Productivity (Y/Y)','range',starthist:endproj,...
    'highlight',endhist+1:endproj,'saveas',sprintf('%sProdYoYTrendGrowth%s',country,version));

x.graph('','legend',true,'zeroline',true,'style',generalstyle);
x.series('Actual',f.DAPL4);
x.band('Equilibrium +/- 2 std devs',f.DAPL4_BAR,-2*f.std.DAPL4_BAR,2*f.std.DAPL4_BAR);
x.series('Steady-state productivity growth',tseries(starthist:endproj,f.growth_ss));
x.clearpage();
```

Add labor productivity and output trend growth

```
x.figure('Labor Productivity and Output Trend Growth (Q/Q Annualized)','range',starthist:endproj,...
    'highlight',endhist+1:endproj,'saveas',sprintf('%sOutputProductivtyTrendGrowth%s',country,version));

x.graph('','legend',true,'zeroline',true,'style',generalstyle);
x.series('Output Trend',f.GROWTH_BAR);
x.series('Labor Productivity Trend',f.DAPL_BAR);
x.clearpage();
```

Add unemployment rate graph.

```
x.figure('Unemployment Rate',...
    'range',starthist:endproj,'saveas',sprintf('%sGraphUnemploymentRate%s',country,version));

x.graph('','legend',true,'legendlocation','North','highlight',endhist+1:endproj,'style',generalstyle);
x.series('Actual ',f.UNR);
x.band('Equilibrium +/- 2 std devs',f.UNR_BAR,-2*f.std.UNR_BAR,2*f.std.UNR_BAR);
x.series('Steady-state unemployment rate',tseries(starthist:endproj,f.unr_ss));

x.clearpage();
```

Add employment trend growth

```
x.figure('Growth of Employment (Q/Q Annualized)','range',starthist:endproj,...
    'highlight',endhist+1:endproj,'saveas',sprintf('%sEmployTrendGrowth%s',country,version));

x.graph('','legend',true,'zeroline',true,'style',generalstyle);
x.series('Actual',f.DLE);
x.band('Equilibrium +/- 2 std devs',f.DLE_BAR,-2*f.std.DLE_BAR,2*f.std.DLE_BAR);
x.clearpage();
```

Add Y/Y employment trend growth

```
x.figure('Growth of Employment (Y/Y Annualized)','range',starthist:endproj,...
    'highlight',endhist+1:endproj,'saveas',sprintf('%sEmployTrendGrowthYoY%s',country,version));

x.graph('','legend',true,'zeroline',true,'style',generalstyle);
x.series('Actual',f.DLE4);
x.band('Equilibrium +/- 2 std devs',f.DLE4_BAR,-2*f.std.DLE4_BAR,2*f.std.DLE4_BAR);
x.clearpage();
```

Add participation graph

```
x.figure('Participation Rate (in percent)','range',starthist:endproj,...
    'highlight',endhist+1:endproj,'saveas',sprintf('%sParticipation%s',country,version));

x.graph('','legend',true,'zeroline',true,'style',generalstyle);
x.series('Actual',100*exp(f.LP/100));
x.series('Trend',100*exp(f.LP_BAR/100));
x.clearpage();
```

Add labor force and population growth

```
x.figure('Labor Force And Population Growth (Q/Q Annualized)','range',starthist:endproj,...
    'highlight',endhist+1:endproj,'saveas',sprintf('%sLaborForceGrowthPopGrowth%s',country,version));

x.graph('','legend',true,'zeroline',true,'style',generalstyle);
thisbeg = get(f.DLLF,'start');
x.series('Actual',f.DLLF{thisbeg+1:endproj});
x.series('Trend',f.DLLF_BAR);
x.series('Population growth',f.DLN);
x.series('Steady-state population growth rate',tseries(starthist:endproj,f.popgrowth_ss));

x.clearpage();
```

Add capacity utilization graph.

```
x.figure('Equilibrium Capacity Utilization +/- 2 std devs (w/o model or parameter uncertainty)',...
    'range',starthist:endproj,'saveas',sprintf('%sGraphCapacityUtilisation%s',country,version));

x.graph('','legend',true,'highlight',endhist+1:endproj,'style',generalstyle);
x.series('Actual',f.CAPU);
x.band('Equilibrium+/- 2 std devs',f.CAPU_BAR,-2*f.std.CAPU_BAR,2*f.std.CAPU_BAR);

x.clearpage();
```

Add relative variances graph.

```
thisstyle = generalstyle;
thisstyle.line.color = {'blue','red'};

x.figure('Relative Variances of Potential Growth and Changes in the Output Gap at Different Horizons',...
    'range',1:20,'saveas',sprintf('%sGraphRelativeVariances%s',country,version));

[Rgap, Rpot] = computeRelativeVariances(f,starthist,endhist);
```

```
x.graph('','legend',true,'style',thisstyle);
x.series('Output gap',Rgap);
x.series('Potential output',Rpot);

x.clearpage();
```

Create and include file with parameter estimate table.

```
createEstimTable(Hess,E,mstar,p);
x.include('Regularized Maximum Likelihood','estimTable');

x.clearpage();
```

Create and include file with calibrated parameter table.

```
createCalibTable(mstar,Estar);
x.include('Calibrated Parameters','calibTable');

x.clearpage();
```

Create and include file with RMSE table.

- `pe` is a structure containing prediction errors for each of the measurement variables and different step-aheads. It is created as output of the `filter` function in `estiamteParams.m`
- `vlist` is a cell of strings containing the variable names for which you want RMSEs computed and displayed. It must be a subset of the observables defined in the model (i.e. the fieldnames of the `pe` structure)

```
vlist = {'GROWTH_','PIE4_','UNR_'};
createRMSETable(pe,vlist);
x.include('RMSEs','RMSETable');

x.clearpage();
```

Create and include file with revisions table.

```
[HPY0,Y0,targ_Y,targ_Y_hp] = createRevisionsTable(mstar,f,endhist,version);
x.include('Mean absolute revisions (according to most recent estimates)','RevisionsTable');
```

Add real-time graph.

```
x.figure('Real-time output gap (MV)',...
    'range',starthist:endhist,'saveas',sprintf('%srealtimeY%s',country,version));

x.graph('','legend',true,'zeroline',true,'style',generalstyle);
x.series('Y (real-time)',Y0);
x.series('Y (final estimate)',targ_Y);

x.clearpage()
```

Add real-time HP graph.

```
x.figure('Real-time output gap (HP)',...
    'range',starthist:endhist,'saveas',sprintf('%srealtimeYHP%s',country,version));

x.graph('','legend',true,'zeroline',true,'style',generalstyle);
x.series('Y (real-time)',HPY0);
x.series('Y (final estimate)',targ_Y_hp);

x.clearpage()
```

Compile PDF.

```
x.publish(sprintf('%sReport%s.pdf',country,version),...
    'title',sprintf('ModYUKE Report: %s%s',country,version),...
    'maketitle',true);
```

```
This is pdfTeX, Version 3.141592-1.40.7 (MiKTeX 2.7)
entering extended mode
(tp1d8ec944_c7de_451f_982d_989d6fe62788.tex
LaTeX2e <2005/12/01>
Babel <v3.8j> and hyphenation patterns for english, dumylang, nohyphenation, ge
rman, ngerman, french, loaded.
(D:\Apps\Miktex2_7\tex\latex\base\article.cls
Document Class: article 2005/09/16 v1.4f Standard LaTeX document class
(D:\Apps\Miktex2_7\tex\latex\base\size10.clo))
(D:\Apps\Miktex2_7\tex\latex\psnfss\mathptmx.sty)
(D:\Apps\Miktex2_7\tex\latex\graphics\graphicx.sty
(D:\Apps\Miktex2_7\tex\latex\graphics\keyval.sty)
(D:\Apps\Miktex2_7\tex\latex\graphics\graphics.sty
(D:\Apps\Miktex2_7\tex\latex\graphics\trig.sty)
(D:\Apps\Miktex2_7\tex\latex\00miktex\graphics.cfg)
(D:\Apps\Miktex2_7\tex\latex\graphics\pdftex.def)))
```

printReport

```
(D:\Apps\Miktex2_7\tex\latex\sectsty\sectsty.sty)
(D:\Apps\Miktex2_7\tex\latex\graphics\color.sty
(D:\Apps\Miktex2_7\tex\latex\00miktex\color.cfg))
(D:\Apps\Miktex2_7\tex\latex\rotating\rotating.sty
(D:\Apps\Miktex2_7\tex\latex\base\ifthen.sty))
(D:\Apps\Miktex2_7\tex\latex\geometry\geometry.sty
(D:\Apps\Miktex2_7\tex\latex\geometry\geometry.cfg))
No file tp1d8ec944_c7de_451f_982d_989d6fe62788.aux.
(D:\Apps\Miktex2_7\tex\latex\psnfss\ot1ptm.fd)
(D:\Apps\Miktex2_7\tex\context\base\supp-pdf.tex
[Loading MPS to PDF converter (version 2006.09.02).]
) (D:\Apps\Miktex2_7\tex\latex\psnfss\ot1ztmcm.fd)
(D:\Apps\Miktex2_7\tex\latex\psnfss\omlztmcm.fd)
(D:\Apps\Miktex2_7\tex\latex\psnfss\omsztmcm.fd)
(D:\Apps\Miktex2_7\tex\latex\psnfss\omxztmcm.fd) [1{C:/Documents and Settings/A
ll Users/Application Data/MiKTeX/2.7/pdftex/config/pdftex.map}] [2] [3]
[4] [5] <USGraphEconomicActivityGapsVER3.pdf, id=21, 516.93124pt x 638.385pt>
<use USGraphEconomicActivityGapsVER3.pdf> [6 <USGraphEconomicActivityGapsVER3.p
df>] <USGraphOutputGapVER3.pdf, id=30, 516.93124pt x 636.3775pt>
<use USGraphOutputGapVER3.pdf> [7 <USGraphOutputGapVER3.pdf>]
<USGraphPotentialGrowthVER3.pdf, id=40, 516.93124pt x 636.3775pt>
<use USGraphPotentialGrowthVER3.pdf> [8 <USGraphPotentialGrowthVER3.pdf>]
<USGraphPotentialGrowthYoYVER3.pdf, id=49, 516.93124pt x 637.38126pt>
<use USGraphPotentialGrowthYoYVER3.pdf> [9 <USGraphPotentialGrowthYoYVER3.pdf>]
 <USProdQoQTrendGrowthVER3.pdf, id=58, 516.93124pt x 636.3775pt>
<use USProdQoQTrendGrowthVER3.pdf> [10 <USProdQoQTrendGrowthVER3.pdf>]
<USProdYoYTrendGrowthVER3.pdf, id=67, 516.93124pt x 636.3775pt>
<use USProdYoYTrendGrowthVER3.pdf> [11 <USProdYoYTrendGrowthVER3.pdf>]
<USOutputProductivtyTrendGrowthVER3.pdf, id=76, 516.93124pt x 636.3775pt>
<use USOutputProductivtyTrendGrowthVER3.pdf> [12 <USOutputProductivtyTrendGrowt
hVER3.pdf>] <USGraphUnemploymentRateVER3.pdf, id=84, 517.935pt x 636.3775pt>
<use USGraphUnemploymentRateVER3.pdf> [13 <USGraphUnemploymentRateVER3.pdf>]
<USEmployTrendGrowthVER3.pdf, id=94, 516.93124pt x 637.38126pt>
<use USEmployTrendGrowthVER3.pdf> [14 <USEmployTrendGrowthVER3.pdf>]
<USEmployTrendGrowthYoYVER3.pdf, id=103, 516.93124pt x 636.3775pt>
<use USEmployTrendGrowthYoYVER3.pdf> [15 <USEmployTrendGrowthYoYVER3.pdf>]
<USParticipationVER3.pdf, id=112, 516.93124pt x 636.3775pt>
<use USParticipationVER3.pdf> [16 <USParticipationVER3.pdf>]
<USLaborForceGrowthPopGrowthVER3.pdf, id=120, 516.93124pt x 636.3775pt>
<use USLaborForceGrowthPopGrowthVER3.pdf> [17 <USLaborForceGrowthPopGrowthVER3.
pdf>] <USGraphCapacityUtilisationVER3.pdf, id=129, 516.93124pt x 636.3775pt>
<use USGraphCapacityUtilisationVER3.pdf> [18 <USGraphCapacityUtilisationVER3.pd
f>] <USGraphRelativeVariancesVER3.pdf, id=138, 513.92pt x 634.37pt>
<use USGraphRelativeVariancesVER3.pdf> [19 <USGraphRelativeVariancesVER3.pdf>]
(estimTable.tex) [20] (calibTable.tex) [21] (RMSETable.tex) [22]
(RevisionsTable.tex) <USrealtimeYVER3.pdf, id=157, 516.93124pt x 629.35126pt>
<use USrealtimeYVER3.pdf> [23] [24 <USrealtimeYVER3.pdf>]
<USrealtimeYHPVER3.pdf, id=171, 516.93124pt x 629.35126pt>
<use USrealtimeYHPVER3.pdf> [25 <USrealtimeYHPVER3.pdf>]
(tp1d8ec944_c7de_451f_982d_989d6fe62788.aux) ){D:/Apps/Miktex2_7/fonts/enc/dvip
```

```
s/fontname/8r.enc}<D:/Apps/Miktex2_7/fonts/type1/bluesky/cm/cmmi10.pfb><D:/Apps
/Miktex2_7/fonts/type1/bluesky/cm/cmsy10.pfb><D:/Apps/Miktex2_7/fonts/type1/blu
esky/cm/cmtt10.pfb><D:/Apps/Miktex2_7/fonts/type1/urw/times/utmb8a.pfb><D:/Apps
/Miktex2_7/fonts/type1/urw/times/utmr8a.pfb><D:/Apps/Miktex2_7/fonts/type1/urw/
times/utmr8a.pfb><D:/Apps/Miktex2_7/fonts/type1/urw/times/utmri8a.pfb>
Output written on tp1d8ec944_c7de_451f_982d_989d6fe62788.pdf (25 pages, 169774
bytes).
Transcript written on tp1d8ec944_c7de_451f_982d_989d6fe62788.log.
```

Save output database as CSV.

This database contains all variables including the estimates of the shocks and unobservables backed out by the Kalman filter.

```
f.std_UNR_BAR = f.std.UNR_BAR;
f.std_Y = f.std.Y;
f.std_G = f.std.G;
f.std_CAPU_BAR = f.std.CAPU_BAR;
f.std_UNR_BAR = f.std.UNR_BAR;
f.std_GROWTH_BAR = f.std.GROWTH_BAR;
f = rmfield(f,'std');
dbsave(f,sprintf('%sOutputData%s.csv',country,version),Inf,...
    'class',false,'comment',false,'format','%.16f');
```

Housekeeping

```
movefile(sprintf('%sOutputData%s.csv',country,version),...
    sprintf('Results/Version%s/OutputData/',version));
movefile(sprintf('%sReport%s.pdf',country,version),...
    sprintf('Results/Version%s/Reports/',version));
movefile(sprintf('%sGraph*%s.*',country,version),...
    sprintf('Results/Version%s/Graphs/',version));
movefile(sprintf('%srealtime*%s.*',country,version),...
    sprintf('Results/Version%s/Graphs/',version));
movefile(sprintf('%sLaborForceGrowthPopGrowth%s.*',country,version),...
    sprintf('Results/Version%s/Graphs/',version));
movefile(sprintf('%sOutputProductivtyTrendGrowth%s.*',country,version),...
    sprintf('Results/Version%s/Graphs/',version));
movefile(sprintf('%sParticipation%s.*',country,version),...
    sprintf('Results/Version%s/Graphs/',version));
movefile(sprintf('%sProdYoYTrendGrowth%s.*',country,version),...
    sprintf('Results/Version%s/Graphs/',version));
movefile(sprintf('%sProdQoQTrendGrowth%s.*',country,version),...
    sprintf('Results/Version%s/Graphs/',version));
movefile(sprintf('%sEmployTrendGrowth%s.*',country,version),...
    sprintf('Results/Version%s/Graphs/',version));
```

```
movefile(sprintf('%sEmployTrendGrowthYoY%s.*',country,version),...
    sprintf('Results/Version%s/Graphs/',version));
movefile('estimTable.tex',...
    sprintf('Results/Version%s/TexFiles/%sestimTable%s.tex',version,country,version));
movefile('calibTable.tex',...
    sprintf('Results/Version%s/TexFiles/%scalibTable%s.tex',version,country,version));
movefile('RMSETable.tex',...
    sprintf('Results/Version%s/TexFiles/%sRMSETable%s.tex',version,country,version));
movefile('RevisionsTable.tex',...
    sprintf('Results/Version%s/TexFiles/%sRevisionsTable%s.tex',version,country,version));


fprintf('%s Done.\n',country);
```

```
US Done.
```