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_

!shocks:measurement

    RES_LLF_

!equations:measurement

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

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';
```

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;
```

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.

```
113: 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
```

```
All measurement variables available.
```

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
- 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 =

VER0

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

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,'Model \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)

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

estimateParams

Iter	F-count	f(x)	Max constraint	Line search steplength	Directional derivative	First-order optimality	Procedure
0	32	1326.59	-0.015				
1	72	1325.44	-0.01992	0.00391	-1.17e+003	2.99e+003	
2	108	1319.9	-0.01868	0.0625	-395	7.17e+003	
3	145	1287.76	-0.01809	0.0313	-363	2.48e+003	
4	180	1263.01	-0.01583	0.125	-120	943	
5	216	1233.65	-0.01484	0.0625	-65.7	1.09e+003	
6	256	1229.56	-0.01636	0.00391	-25.2	637	
7	291	1192.8	-0.01432	0.125	-159	1.88e+003	
8	328	1174.96	-0.01676	0.0313	-132	1.14e+003	
9	363	1133.66	-0.01466	0.125	-120	1e+003	
10	398	1128.5	-0.01283	0.125	-123	1.2e+003	
11	438	1127.39	-0.01334	0.00391	-95.3	895	
12	475	1113.68	-0.01292	0.0313	-146	1.8e+003	
13	511	1102.98	-0.01211	0.0625	-84	1.05e+003	
14	548	1091.38	-0.01671	0.0313	-74.6	669	
15	584	1084.04	-0.01566	0.0625	-63.3	631	
16	620	1072.26	-0.01468	0.0625	-58	315	
17	656	1062.56	-0.01377	0.0625	-44.3	237	
18	692	1061.87	-0.01468	0.0625	-47.6	1.32e+003	
19	729	1058.98	-0.01422	0.0313	-20.2	463	
20	765	1051.38	-0.01333	0.0625	-35.6	1.32e+003	
21	804	1050.72	-0.01414	0.00781	-30.8	623	
22	842	1048.84	-0.01489	0.0156	-8.97	1.22e+003	
23	881	1048.54	-0.01501	0.00781	-35	691	
24	917	1048.48	-0.01579	0.0625	-25.2	426	
25	954	1045.31	-0.0162	0.0313	-13.7	482	
26	991	1044.24	-0.0157	0.0313	-9.64	413	
27	1029	1043.96	-0.01545	0.0156	-4.37	328	
28	1066	1042.38	-0.01597	0.0313	-50.1	217	
29	1103	1039.93	-0.01547	0.0313	-27.5	632	
30	1140	1039.89	-0.01547	0.0313	-10.1	203	
31	1177	1039.47	-0.01529	0.0313	-9.97	252	
32	1214	1039.47	-0.01509	0.0313	-21.2	323	
33	1252	1039.23	-0.0151	0.0156	-9.44	333	
34	1290	1039.15	-0.01499	0.0156	-4.47	244	
35	1326	1039.11	-0.01496	0.0625	-9.44	330	
36	1363	1038.67	-0.01512	0.0313	-11.2	274	
37	1400	1038.48	-0.01508	0.0313	-11.1	190	
38	1437	1037.63	-0.01501	0.0313	-12.2	90.1	
39	1476	1037.57	-0.015	0.00781	-9.57	137	
40	1513	1037.45	-0.0151	0.0313	-4.32	91.1	
41	1549	1037.38	-0.01522	0.0625	-22.8	303	
42	1586	1037.35	-0.01526	0.0313	-6.86	72.5	
43	1623	1037.01	-0.01528	0.0313	-21.9	62.5	
44	1657	1036.94	-0.01519	0.25	-8.68	191	
45	1694	1036.85	-0.01513	0.0313	-9.3	135	
46	1727	1036.54	-0.01514	0.5	-8.85	52.3	
47	1765	1036.52	-0.01514	0.0156	-3.62	31.7	

```
   48   1797      1036.49     -0.01517            1        -2.66         3.91
Optimization terminated: magnitude of search direction less than 2*options.TolX
 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/VersionVER0/Estimates/USVER0.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/
```

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';

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

```
Version to report: VER0.
```

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
(tp7e2001f1_e62e_4216_b0d9_456b16d2bb8a.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 tp7e2001f1_e62e_4216_b0d9_456b16d2bb8a.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] <USGraphEconomicActivityGapsVER0.pdf, id=21, 516.93124pt x 638.385pt>
<use USGraphEconomicActivityGapsVER0.pdf> [6 <USGraphEconomicActivityGapsVER0.p
df>] <USGraphOutputGapVER0.pdf, id=30, 516.93124pt x 636.3775pt>
<use USGraphOutputGapVER0.pdf> [7 <USGraphOutputGapVER0.pdf>]
<USGraphPotentialGrowthVER0.pdf, id=40, 516.93124pt x 636.3775pt>
<use USGraphPotentialGrowthVER0.pdf> [8 <USGraphPotentialGrowthVER0.pdf>]
<USGraphPotentialGrowthYoYVER0.pdf, id=49, 516.93124pt x 636.3775pt>
<use USGraphPotentialGrowthYoYVER0.pdf> [9 <USGraphPotentialGrowthYoYVER0.pdf>]
 <USProdQoQTrendGrowthVER0.pdf, id=58, 516.93124pt x 636.3775pt>
<use USProdQoQTrendGrowthVER0.pdf> [10 <USProdQoQTrendGrowthVER0.pdf>]
<USProdYoYTrendGrowthVER0.pdf, id=67, 516.93124pt x 636.3775pt>
<use USProdYoYTrendGrowthVER0.pdf> [11 <USProdYoYTrendGrowthVER0.pdf>]
<USOutputProductivtyTrendGrowthVER0.pdf, id=76, 516.93124pt x 636.3775pt>
<use USOutputProductivtyTrendGrowthVER0.pdf> [12 <USOutputProductivtyTrendGrowt
hVER0.pdf>] <USGraphUnemploymentRateVER0.pdf, id=85, 517.935pt x 636.3775pt>
<use USGraphUnemploymentRateVER0.pdf> [13 <USGraphUnemploymentRateVER0.pdf>]
<USEmployTrendGrowthVER0.pdf, id=95, 516.93124pt x 636.3775pt>
<use USEmployTrendGrowthVER0.pdf> [14 <USEmployTrendGrowthVER0.pdf>]
<USEmployTrendGrowthYoYVER0.pdf, id=104, 516.93124pt x 636.3775pt>
<use USEmployTrendGrowthYoYVER0.pdf> [15 <USEmployTrendGrowthYoYVER0.pdf>]
<USParticipationVER0.pdf, id=113, 516.93124pt x 636.3775pt>
<use USParticipationVER0.pdf> [16 <USParticipationVER0.pdf>]
<USLaborForceGrowthPopGrowthVER0.pdf, id=121, 516.93124pt x 636.3775pt>
<use USLaborForceGrowthPopGrowthVER0.pdf> [17 <USLaborForceGrowthPopGrowthVER0.
pdf>] <USGraphCapacityUtilisationVER0.pdf, id=130, 516.93124pt x 636.3775pt>
<use USGraphCapacityUtilisationVER0.pdf> [18 <USGraphCapacityUtilisationVER0.pd
f>] <USGraphRelativeVariancesVER0.pdf, id=139, 513.92pt x 640.3925pt>
<use USGraphRelativeVariancesVER0.pdf> [19 <USGraphRelativeVariancesVER0.pdf>]
(estimTable.tex) [20] (calibTable.tex) [21] (RMSETable.tex) [22]
(RevisionsTable.tex) <USrealtimeYVER0.pdf, id=158, 516.93124pt x 630.355pt>
<use USrealtimeYVER0.pdf> [23] [24 <USrealtimeYVER0.pdf>]
<USrealtimeYHPVER0.pdf, id=172, 516.93124pt x 629.35126pt>
<use USrealtimeYHPVER0.pdf> [25 <USrealtimeYHPVER0.pdf>]
(tp7e2001f1_e62e_4216_b0d9_456b16d2bb8a.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 tp7e2001f1_e62e_4216_b0d9_456b16d2bb8a.pdf (25 pages, 169915
bytes).
Transcript written on tp7e2001f1_e62e_4216_b0d9_456b16d2bb8a.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.
```