Estimated Dynamic Optimization (EDO) Model

Gary Young (editor)

June 26, 2016

Contents

Chapter 1

EDO model packages

The Estimated Dynamic Optimization (EDO) Model is available from the Federal Reserve Board of Governors website:

> The model package zip file (link below) contains the following files:
>
> - A readme file with basic instructions.
> - Dynare mod files to run two versions of the EDO model, one with variables in levels and the other with variables in log deviations from steady state. Both versions include the nonlinear equations derived from household and firm optimization problems. The version in levels more closely follows the derivations described in the article "Unemployment During the Great Recession in the EDO Model of the U.S. Economy", while the version in log deviations facilitates the reporting of simulation results. These programs solve the model, report some basic model statistics, and run some basic impulse response simulations.

Notice that the edo (zip) is actually contained in the "EDO variable listing (ZIP)" link and the mentioned "variable listing zip file" doesn't seem to be available anywhere:

> The variable listing zip file (link below) contains an HTML representation of the EDO model, showing linkages between variables, parameters, and equations.

> EDO variable listing (ZIP)

> NOTE: The programs for simulating the EDO model are written for use with the Dynare software package. The Dynare package can be downloaded without cost at www.dynare.org While Dynare itself is free, it requires the installation of either Matlab or Octave. Matlab is a commercial product available at www.mathworks.com. Octave is free-ware, and is available at www.gnu.org/software/octave

Dynare and octave were available in The Ubuntu Software Center for my computer. From the readme file, to execute the model:

> 5) Run the command "dynare linearized" or "dynare Dynare_edo" from the Matlab/Octave command line to run the two model versions

The model fails for me and when googling for the error "dynare linearized trans_A undefined" we get

> At the moment, Octave 4 is not supported yet. See https://github.com/DynareTeam/dynare/issues/1113

at "Trans_A Matrix error" and "Make Dynare compatible with Octave 4.0 #1113".

Appendices

Appendix A

Original Files

A.1 Dynare_edo_mod

9 ⟨*srcedo/Dynare.edo.mod* 9⟩≡

```
⟨edo var 11⟩


⟨edo varexo 13⟩


⟨edo parameters 14⟩


//estimated_params;
⟨edo estimated_params 18⟩
//end_estimated_params;

//calibrated_params;
⟨edo calibrated_params 19⟩
//end_calibrated_params;

//free_params;
⟨edo free_params 20⟩
//end_free_params;

//calibrated ME


⟨edo model 21⟩
```

```
options_.order = 1;
options_.jacobian_flag = 1;
options_.nonlin = 1;

stoch_simul(order=1,irf=40,nograph);
```

This code is written to file srcedo/Dynare.edo.mod.
Defines:
jacobian_flag, used in chunk 38.
nonlin, used in chunk 38.
options_, used in chunk 38.
order, used in chunk 38.
stoch_simul, used in chunk 38.

A.1.1 Dynare EDO Var

11 ⟨*edo var* 11⟩≡ (9)

```
var RC RK WC WK YC YK MCC MCK KC  KK PKB R L QK HC  HSC  HK HSK  UHC UHSC UHK UHSK empC  HrC  e
 DIFFREALGDP_obs DIFFREALEC_obs DIFFREALEIK_obs DIFFREALECD_obs DIFFREALECH_obs DIFFREALW_obs A
```

Defines:
- AH, used in chunks 24a, 26a, and 38.
- AH_obs, used in chunks 26, 27, and 38.
- betas, used in chunks 22, 25, and 38.
- DIFFNORMGDP, used in chunks 23a and 38.
- DIFFREALEC, used in chunks 23a, 26a, and 38.
- DIFFREALEC_obs, used in chunks 26, 27, and 38.
- DIFFREALECD, used in chunks 25, 26a, and 38.
- DIFFREALECD_obs, used in chunks 26, 27, and 38.
- DIFFREALECH, used in chunks 25, 26a, and 38.
- DIFFREALECH_obs, used in chunks 26, 27, and 38.
- DIFFREALEIK, used in chunks 23a, 26a, and 38.
- DIFFREALEIK_obs, used in chunks 26, 27, and 38.
- DIFFREALGDP, used in chunks 23a, 24a, 26a, and 38.
- DIFFREALGDP_obs, used in chunks 26, 27, and 38.
- DIFFREALW, used in chunks 23b, 26a, and 38.
- DIFFREALW_obs, used in chunks 26, 27, and 38.
- EC, used in chunks 22, 23a, and 38.
- ECD, used in chunks 23–25 and 38.
- ECH, used in chunks 23a, 25, and 38.
- EFFECD, used in chunks 24b, 25, and 38.
- EFFECH, used in chunks 25 and 38.
- EFFK, used in chunks 22, 23a, 25, and 38.
- EIK, used in chunks 22, 23a, and 38.
- empC, used in chunks 23a and 38.
- empK, used in chunks 23a and 38.
- empSC, used in chunks 23a and 38.
- empSK, used in chunks 23a and 38.
- GAP, used in chunks 24a and 38.
- HC, used in chunks 22–24 and 38.
- HG, used in chunks 23a, 25, and 38.
- HK, used in chunks 22–24 and 38.
- HrC, used in chunks 23a and 38.
- HrK, used in chunks 23a and 38.
- HrSC, used in chunks 23a and 38.
- HrSK, used in chunks 23a and 38.
- HSC, used in chunks 23a and 38.
- HSK, used in chunks 23a and 38.
- INFC, used in chunks 22–24 and 38.
- INFC10, used in chunks 24a and 38.
- INFCNA, used in chunks 22, 24a, 26a, and 38.
- INFCNA_obs, used in chunks 26, 27, and 38.
- INFCOR, used in chunks 24a, 26a, and 38.
- INFCOR_obs, used in chunks 26, 27, and 38.
- INFGDP, used in chunks 24a and 38.
- INFK, used in chunks 22–24, 26a, and 38.
- INFK_obs, used in chunks 26, 27, and 38.
- INFWC, used in chunks 23 and 38.
- INFWK, used in chunks 23 and 38.
- KC, used in chunks 22, 23a, and 38.
- KCH, used in chunks 25 and 38.
- KD, used in chunks 24b, 25, and 38.

A.1.2 Dynare EDO VarExo

13 ⟨*edo varexo* 13⟩≡ (9)

```
varexo eHG eXiL eLpref eR eMUZK eMUZM ePMKC ePMKK eEFFECH eEFFECD eEFFK eB eSTAR;
```

Defines:
- `eB`, used in chunks 25, 27, 29, and 38.
- `eEFFECD`, used in chunks 25, 27, 29, and 38.
- `eEFFECH`, used in chunks 25, 27, 29, and 38.
- `eEFFK`, used in chunks 25, 27, 29, and 38.
- `eHG`, used in chunks 25, 27, 29, and 38.
- `eLpref`, used in chunks 25, 27, 29, and 38.
- `eMUZK`, used in chunks 25, 27, 29, and 38.
- `eMUZM`, used in chunks 25, 27, 29, and 38.
- `ePMKC`, used in chunks 22, 27, 29, and 38.
- `ePMKK`, used in chunks 22, 27, 29, and 38.
- `eR`, used in chunks 22, 27, 29, and 38.
- `eSTAR`, used in chunks 25, 27, 29, and 38.
- `eXiL`, used in chunks 23a, 25, 27, 29, and 38.

A.1.3 Dynare EDO Parameters

14 ⟨*edo parameters* 14⟩≡ (9)

```
parameters
h r_inf r_y r_dy phi_pc phi_H phi_wc phi_ic phi_cd phi_ech gam_pc gam_wc gam_ic gam_
rho_EFFECD rho_HG rho_EFFECH tp2 ONE MUZMSS MUZKSS  r_dinf rpr phi_u rho_MUZK rho_MU
theta_k theta_wc theta_wk g_y a_ks s_AS gam_h gam_ech  s_k s_ecdc eta_cnn eta_cd eta_
icoef mu_ betarl MUZCSS RCSS RKSS WCSS WKSS YCSS YKSS MCCSS MCKSS KCSS KKSS LSS HCSS
MUCSS MUKSS AHSS ECDSS KCDSS QCDSS RCDSS ECHSS KCHSS QCHSS RCHSS UKSS UCSS USS MUKSS
INFCNASS INFCORSS INFC10SS  RT2SS beta_0 beta_2 beta_ PYSS AA DD RR
eta_cd_eta_cnn eta_ch_eta_cnn Rnr ycbi_ykb hc_hk HSS ycbi ykb YYSS s_k_ecd s_c_ech s_
sig_HG sig_XiL sig_lpref sig_R sig_MUZK sig_MUZM sig_PMKC sig_PMKK sig_EFFECH  sig_E
HSKSS HSCSS HrCSS HrKSS A_HC sigman sigmah A_HK xsi_NC xsi_HrC xsi_NK xsi_HrK rho_Xi
empCSS empKSS HrSKSS HrSCSS empSCSS empSKSS UHCSS UHKSS UHSCSS UHSKSS unempSS DIFFRE
DIFFREALECHSS DIFFREALEIKSS DIFFREALWSS  RL1SS RL2SS RL3SS RL4SS RL5SS
RL6SS RL7SS DIFFREALGDPSS_obs DIFFREALECSS_obs DIFFREALEIKSS_obs DIFFREALECDSS_obs
DIFFREALECHSS_obs DIFFREALWSS_obs INFCNASS_obs INFCORSS_obs INFKSS_obs
RSS_obs RT2SS_obs unempSS_obs;
```

Defines:

- A_HC, used in chunks 20, 23a, 31, 38, and 47.
- A_HK, used in chunks 20, 23a, 31, 38, and 47.
- a_ks, used in chunks 19 and 38.
- AA, used in chunks 31, 38, and 47.
- AHSS, used in chunks 23b, 26a, 31, 35, 38, and 47.
- alpha_, used in chunks 19, 22, 24a, 25, 31, 38, and 47.
- beta_, used in chunks 22–25, 31, 35, 38, and 47.
- beta_0, used in chunks 31, 38, and 47.
- beta_2, used in chunks 31, 38, and 47.
- betarl, used in chunks 19, 24a, and 38.
- DD, used in chunks 31, 38, and 47.
- delta_, used in chunks 19, 22, 31, 38, and 47.
- delta_cd, used in chunks 19, 24b, 31, 38, and 47.
- delta_ch, used in chunks 19, 25, 31, 38, and 47.
- DIFFREALECDSS, used in chunks 26a, 31, 35, 38, and 47.
- DIFFREALECDSS_obs, used in chunks 26a, 34, 35, 38, and 47.
- DIFFREALECHSS, used in chunks 26a, 31, 35, 38, and 47.
- DIFFREALECHSS_obs, used in chunks 26a, 34, 35, 38, and 47.
- DIFFREALECSS, used in chunks 26a, 31, 35, 38, and 47.
- DIFFREALECSS_obs, used in chunks 26a, 34, 35, 38, and 47.
- DIFFREALEIKSS, used in chunks 26a, 31, 35, 38, and 47.
- DIFFREALEIKSS_obs, used in chunks 26a, 34, 35, 38, and 47.
- DIFFREALGDPSS, used in chunks 26a, 31, 35, 38, and 47.
- DIFFREALGDPSS_obs, used in chunks 26a, 34, 35, 38, and 47.
- DIFFREALWSS, used in chunks 26a, 31, 35, 38, and 47.
- DIFFREALWSS_obs, used in chunks 26a, 34, 35, 38, and 47.
- ECDSS, used in chunks 24b, 31, 35, 38, and 47.
- ECHSS, used in chunks 25, 31, 35, 38, and 47.
- ECSS, used in chunks 31, 35, 38, and 47.
- EIKSS, used in chunks 23a, 31, 35, 38, and 47.
- empCSS, used in chunks 31, 35, 38, and 47.
- empKSS, used in chunks 31, 35, 38, and 47.
- empSCSS, used in chunks 31, 35, 38, and 47.
- empSKSS, used in chunks 31, 35, 38, and 47.

A.1.4 Dynare EDO Estimated Params

18 ⟨*edo estimated_params* 18⟩≡ (9)

```
h             = 0.715162417869797;
r_inf         = 1.46344163969035;
r_y           = 0.263123294207851;
phi_pc        = 3.54471453295450;
phi_H         = 3.22894079106560;
phi_wc        = 5.49395755514723;
phi_ic        = 0.253308786976374;
phi_cd        = 0.470089385005009;
phi_ech       = 9.13986886546163;
gam_pc        = 0.314488926051065;
gam_wc        = -0.230018833252054;
sigman        = 39.4075260618789;
sigmah        = 21.8859803402692;
rho_R         = 0.833200065745674;
rho_XiL       = 0.263567746111198;
rho_lpref     = 0.979092048897712;
rho_B         = 0.895267027146152;
rho_STAR      = 0.909187927454138;
rho_EFFK      = 0.937829274540004;
rho_EFFECD    = -0.240286975088701;
rho_HG        = 0.582395471123139;
rho_EFFECH    = 0.877235725078934;
tp2           = 0.000307314910763576;
sig_HG        = 0.579315931803017;
sig_XiL       = 2.49313873916751;
sig_lpref     = 5.66476748114241;
sig_R         = 0.124100461010359;
sig_MUZK      = 0.936167718269030;
sig_MUZM      = 0.597390920898135;
sig_PMKC      = 0.451830653200989;
sig_PMKK      = 0.685376191952156;
sig_EFFECH    = 0.514704527091087;
sig_EFFECD    = 9.11199585973990;
sig_EFFK      = 0.402779878811407;
sig_B         = 0.295232712196573;
sig_STAR      = 0.104877885500673;
```

Uses `gam_pc` 14, `gam_wc` 14, `h` 14, `phi_cd` 14, `phi_ech` 14, `phi_H` 14, `phi_ic` 14, `phi_pc` 14, `phi_wc` 14, `r_inf` 14, `r_y` 14, `rho_B` 14, `rho_EFFECD` 14, `rho_EFFECH` 14, `rho_EFFK` 14, `rho_HG` 14, `rho_lpref` 14, `rho_R` 14, `rho_STAR` 14, `rho_XiL` 14, `sig_B` 14, `sig_EFFECD` 14, `sig_EFFECH` 14, `sig_EFFK` 14, `sig_HG` 14, `sig_lpref` 14, `sig_MUZK` 14, `sig_MUZM` 14, `sig_PMKC` 14, `sig_PMKK` 14, `sig_R` 14, `sig_STAR` 14, `sig_XiL` 14, `sigmah` 14, `sigman` 14, and `tp2` 14.

A.1.5 Dynare EDO Calibrated Params

19 ⟨*edo calibrated_params* 19⟩≡ (9)

```
r_dy = 0;
ONE = 1;
MUZKSS = 1.009250;
MUZMSS = 1.001000;
gam_ic = 1.0;
gam_icd = 1.0;
r_dinf = 0;
rpr = 0.965;
phi_u = 1;
rho_MUZK = 0;
rho_MUZM = 0;
pbeta = 0.99862;
delta_ = 0.03;
h_cd = 0.0;
h_ch = 0.0;
delta_cd = 0.055;
delta_ch = 0.0035;
alpha_ = 0.26;
theta_c = 7;
theta_k = 7;
unempSS = .06;
g_y = 0.0;
a_ks = 0.2;
s_AS = 0.2;
gam_h = 1;
gam_ech = 1;
icoef = 3;
betarl = .958;
```

Uses `a_ks` 14, `alpha_` 14, `betarl` 14, `delta_` 14, `delta_cd` 14, `delta_ch` 14, `g_y` 14, `gam_ech` 14, `gam_h` 14, `gam_ic` 14, `gam_icd` 14, `h_cd` 14, `h_ch` 14, `icoef` 14, `MUZKSS` 14, `MUZMSS` 14, `ONE` 14, `pbeta` 14, `phi_u` 14, `r_dinf` 14, `r_dy` 14, `rho_MUZK` 14, `rho_MUZM` 14, `rpr` 14, `s_AS` 14, `theta_c` 14, `theta_k` 14, and `unempSS` 14.

A.1.6 Dynare EDO Free Params

20 ⟨*edo free_params* 20⟩≡ (9)

```
//A_HC;
//A_HK;
//xsi_NC;
//xsi_HrC;
//xsi_NK;
//xsi_HrK;
//theta_wc;
//theta_wk;
//infkbar;
//infcbar;
//infwcbar;
//infwkbar;
//Pybar;
//Yybar;
//mu_yc;
//mu_yk;
//s_k;
//s_ecdc;
//eta_cnn;
//eta_cd;
//eta_ch;
//mu_;
```

Uses A_HC 14, A_HK 14, eta_cd 14, eta_ch 14, eta_cnn 14, mu_ 14, s_ecdc 14, s_k 14, theta_wc 14, theta_wk 14, xsi_HrC 14, xsi_HrK 14, xsi_NC 14, and xsi_NK 14.

A.1.7 Dynare EDO Model

21 ⟨*edo model* 21⟩≡ (9)

```
//****************************
//MODEL BLOCK
//****************************

model;
⟨edo model prelim 22⟩

// XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
// labor block
// TOTAL LABOR INPUT (called "L" in the paper, I kept the "H" notation of the original EDO prg)
⟨edo model labor 23a⟩

// Identities
⟨edo model identities 23b⟩


// XXXXXXXXXXXXXXXXXXXXXX
// Aggregate hours equals agg hours in each sector
⟨edo model hours 24a⟩

// See Section 8: Data Identities

// new equations
// Durable Block

⟨edo model durables 24b⟩

// Housing Block
⟨edo model housing 25⟩

//measurement_equations;
⟨edo model measurement 26a⟩
//end_measurement_equations;
end;

⟨edo model varobs 26b⟩

shocks;
⟨edo model shocks 27⟩
end;

steady;
```

```
estimated_params;
⟨edo model estimated_params 28⟩

⟨edo model stderr 29⟩
end;
```

Uses L 11.

A.1.8 Dynare EDO Prelim

22 ⟨*edo model prelim* 22⟩≡ (21)

```
RC-MCC*YC/UC/KC(-1)*alpha_*MUK=0;
RK-MCK*YK/UK/KK(-1)*alpha_*MUK=0;
WC-MCC*YC/HC*(1-alpha_)=0;
WK-MCK*YK/HK*(1-alpha_)=0;
YC-(UC*KC(-1)/MUK)^alpha_*(HC)^(1-alpha_)=0;
YK-(UK*KK(-1)/MUK)^alpha_*(HK)^(1-alpha_)=0;
MCC*YC*theta_c-(theta_c-1)*YC-100*phi_pc*(INFC-gam_pc*INFC(-1)-(1-gam_pc)*INFCSS)*INI
MCK*YK*theta_k/PKB-(theta_k-1)*YK-100*phi_pc*(INFK-gam_pc*INFK(-1)-(1-gam_pc)*INFKSS)
QK-beta_*(1/EFFK)*(((1-delta_)*QK(+1)+RC(+1)*UC(+1))*L(+1)/MUK(+1)/L)=0;
QK-beta_*(1/EFFK)*(((1-delta_)*QK(+1)+RK(+1)*UK(+1))*L(+1)/MUK(+1)/L)=0;
L-betas*R/rpr/INFC(+1)/MUC(+1)*L(+1)=0;
ln(R/RSS)-rho_R*ln(R(-1)/RSS)-(1-rho_R)*(r_inf*ln(INFCNA/INFCNASS)+r_dinf*(ln(INFCNA)
L-eta_cnn/(EC-h*EC(-1)/MUC)+eta_cnn*beta_*h/(MUC(+1)*EC(+1)-h*EC)=0;
KK-(1-delta_)*KK(-1)/MUK+KC-(1-delta_)*KC(-1)/MUK-1*EIK+mu_*((UK^(1+1/phi_u)-1)/(1+1/
```

Uses alpha_ 14, beta_ 14, betas 11, delta_ 14, EC 11, EFFK 11, EIK 11, ePMKC 13, ePMKK 13, eR 13, eta_cnn 14, gam_pc 14, h 14, HC 11, HK 11, INFC 11, INFCNA 11, INFCNASS 14, INFCSS 14, INFK 11, INFKSS 14, KC 11, KCSS 14, KK 11, KKSS 14, L 11, MCC 11, MCK 11, mu_ 14, MUC 11, MUK 11, PFGAP 11, phi_pc 14, phi_u 14, PKB 11, QK 11, R 11, r_dinf 14, r_inf 14, r_y 14, RC 11, rho_R 14, RK 11, rpr 14, RSS 14, theta_c 14, theta_k 14, UC 11, UK 11, WC 11, WK 11, YC 11, YCSS 14, YK 11, and YKSS 14.

A.1.9 Dynare EDO Labor

23a ⟨*edo model labor* 23a⟩≡ (21)

```
-100+UHC*theta_wc-(theta_wc-1)*WC-100*phi_wc*(INFWC-gam_wc*INFWC(-1)-(1-gam_wc)*INFWCSS)*INFWC*
UHSC-WC+phi_H/10*(HSC/HSK-gam_h*HSC(-1)/HSK(-1)-(1-gam_h)*HSCSS/HSKSS);//+100*eXiL=0;
-100+UHK*theta_wk-(theta_wk-1)*WK-100*phi_wc*(INFWK-gam_wc*INFWK(-1)-(1-gam_wc)*INFWKSS)*INFWK*
UHSK-WK-phi_H/10*(HSC/HSK-gam_h*HSC(-1)/HSK(-1)-(1-gam_h)*HSCSS/HSKSS);//+100*eXiL=0;
UHC*L*Lpref-A_HC*((1+sigman)/(1+sigman/(1+sigmah)))*(HC)^(-1+(1+sigman)/(1+sigman/(1+sigmah)))=
UHSC*L*Lpref-A_HC*((1+sigman)/(1+sigman/(1+sigmah)))*(HSC)^(-1+(1+sigman)/(1+sigman/(1+sigmah))
UHK*L*Lpref-A_HK*((1+sigman)/(1+sigman/(1+sigmah)))*(HK)^(-1+(1+sigman)/(1+sigman/(1+sigmah)))=
UHSK*L*Lpref-A_HK*((1+sigman)/(1+sigman/(1+sigmah)))*(HSK)^(-1+(1+sigman)/(1+sigman/(1+sigmah))
empC-((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(-1/(1+sigmah+sigman))*HC^(1/(1+sigman/(1+sigmah)))=0;
HrC-((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(1/(1+sigmah))*empC^(sigman/(1+sigmah))=0;
empK-((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(-1/(1+sigmah+sigman))*HK^(1/(1+sigman/(1+sigmah)))=0;
HrK-((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(1/(1+sigmah))*empK^(sigman/(1+sigmah))=0;
empSC-((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(-1/(1+sigmah+sigman))*HSC^(1/(1+sigman/(1+sigmah)))=0
HrSC-((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(1/(1+sigmah))*empSC^(sigman/(1+sigmah))=0;
empSK-((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(-1/(1+sigmah+sigman))*HSK^(1/(1+sigman/(1+sigmah)))=0
HrSK-((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(1/(1+sigmah))*empSK^(sigman/(1+sigmah))=0;
unemp-(empSC+empSK-(empC+empK))/(empSC+empSK)=0;
PKB-(1-100*phi_ic*(EIK-gam_ic*EIK(-1)-(1-gam_ic)*EIKSS)/(KC(-1)+KK(-1))*MUK)*QK-beta_*(1/EFFK)*
YC-EC-ECH-0.2*YCSS*HG=0;
ln(INFWC)-ln(WC)+ln(WC(-1))-ln(MUC)-ln(INFC)=0;
ln(INFWK)-ln(WK)+ln(WK(-1))-ln(MUC)-ln(INFC)=0;
ln(INFK)-ln(INFC)-ln(PKB)+ln(PKB(-1))+ln(MUK)-ln(MUC)=0;
YK-EIK-ECD-0.2*YKSS*HG=0;
ln(DIFFNORMGDP)-(1-s_k)*(ln(YC)-ln(YC(-1)))-s_k*(ln(YK)-ln(YK(-1)))=0;
ln(NORMINFGDP)-s_k*(ln(PKB)-ln(PKB(-1)))=0;
ln(DIFFREALGDP)-ln(DIFFNORMGDP)-(1-s_k)*ln(MUC)-s_k*ln(MUK)=0;
ln(DIFFREALEC)-ln(EC)+ln(EC(-1))-ln(MUC)=0;
ln(DIFFREALEIK)-ln(EIK)+ln(EIK(-1))-ln(MUK)=0;
```

Uses A_HC 14, A_HK 14, beta_ 14, DIFFNORMGDP 11, DIFFREALEC 11, DIFFREALEIK 11, DIFFREALGDP 11, EC 11, ECD 11, ECH 11, EFFK 11, EIK 11, EIKSS 14, empC 11, empK 11, empSC 11, empSK 11, eXiL 13, gam_h 14, gam_ic 14, gam_wc 14, HC 11, HCSS 14, HG 11, HK 11, HKSS 14, HrC 11, HrK 11, HrSC 11, HrSK 11, HSC 11, HSCSS 14, HSK 11, HSKSS 14, INFC 11, INFK 11, INFWC 11, INFWCSS 14, INFWK 11, INFWKSS 14, KC 11, KK 11, L 11, Lpref 11, MUC 11, MUK 11, NORMINFGDP 11, phi_H 14, phi_ic 14, phi_wc 14, PKB 11, QK 11, s_k 14, sigmah 14, sigman 14, theta_wc 14, theta_wk 14, UHC 11, UHK 11, UHSC 11, UHSK 11, unemp 11, WC 11, WK 11, XiL 11, xsi_HrC 14, xsi_HrK 14, xsi_NC 14, xsi_NK 14, YC 11, YCSS 14, YK 11, and YKSS 14.

A.1.10 Dynare EDO Identities

23b ⟨*edo model identities* 23b⟩≡ (21)

```
ln(DIFFREALW)-HCSS/AHSS*(ln(INFWC))-HKSS/AHSS*(ln(INFWK))+ln(INFC)=0;
```

Uses AHSS 14, DIFFREALW 11, HCSS 14, HKSS 14, INFC 11, INFWC 11, and INFWK 11.

A.1.11 Dynare EDO Hours

24a ⟨*edo model hours* 24a⟩≡ (21)

```
AH-HC-HK=0;
ln(INFGDP)-ln(INFC)-ln(YC*MUC/YC(-1))+ln(DIFFREALGDP)-ln((1+PKB*YK/YC)/(1+PKB(-1)*YK(
ln(INFCNA)-(1-s_ecdc)*ln(INFC)-s_ecdc*ln(INFK)=0;
ln(INFCOR)-(1-s_ecdc)*ln(INFC)-s_ecdc*ln(INFK)=0;
ln(GAP)-(1-s_k)*ln(YC/YCSS)-s_k*ln(YK/YKSS)=0;
ln(PFGAP)-(1-alpha_)*((1-s_k)*ln(HC/HCSS)+s_k*ln(HK/HKSS))-alpha_*((1-s_k)*ln(UC/USS)
ln(INFC10)-betarl*ln(INFC10(+1))-(1-betarl)*ln(INFCOR)=0;
```

Uses AH 11, alpha_ 14, betarl 14, DIFFREALGDP 11, GAP 11, HC 11, HCSS 14, HK 11, HKSS 14, INFC 11, INFC10 11, INFCNA 11, INFCOR 11, INFGDP 11, INFK 11, MUC 11, PFGAP 11, PKB 11, s_ecdc 14, s_k 14, UC 11, UK 11, USS 14, YC 11, YCSS 14, YK 11, and YKSS 14.

A.1.12 Dynare EDO Durables

24b ⟨*edo model durables* 24b⟩≡ (21)

```
KD-(1-delta_cd)*KD(-1)/MUK-ECD=0;
L*RCD-eta_cd/(KD(-1)/MUK-h_cd*LAGKD(-1)/(MUK(-1)*MUK))+beta_*eta_cd*h_cd/(KD-h_cd*KD(
QCD-beta_*(1/EFFECD)*L(+1)/L/MUK(+1)*(RCD(+1)+(1-delta_cd)*QCD(+1))=0;
PKB-QCD*(1-100*phi_cd*(ECD-gam_icd*ECD(-1)-(1-gam_icd)*ECDSS)/KD(-1)*MUK) - beta_*(1/
```

Uses beta_ 14, delta_cd 14, ECD 11, ECDSS 14, EFFECD 11, eta_cd 14, gam_icd 14, h_cd 14, KD 11, L 11, LAGKD 11, MUK 11, phi_cd 14, PKB 11, QCD 11, and RCD 11.

A.1.13 Dynare EDO Housing

25 ⟨*edo model housing* 25⟩≡ (21)

```
L*RCH-eta_ch/(KCH(-1)/MUC-h_ch*LAGKCH(-1)/(MUC*MUC(-1)))+beta_*eta_ch*h_ch/(KCH-h_ch*KCH(-1)/MU
QCH-beta_*(1/EFFECH)*L(+1)/L/MUC(+1)*(RCH(+1)+(1-delta_ch)*QCH(+1))=0;
1*ECH+(1-delta_ch)*KCH(-1)/MUC-KCH=0;
1-QCH*(1-100*phi_ech*(ECH-gam_ech*ECH(-1)-(1-gam_ech)*ECHSS)/KCH(-1)*MUC) - beta_*(1/EFFECH)*10
ln(KD(-1))-ln(LAGKD)=0;
ln(KCH(-1))-ln(LAGKCH)=0;
RK-QK*mu_*UK^(1/phi_u)=0;
RC-QK*mu_*UC^(1/phi_u)=0;
ln(DIFFREALECH)-ln(MUC)-ln(ECH)+ln(ECH(-1))=0;
ln(DIFFREALECD)-ln(MUK)-ln(ECD)+ln(ECD(-1))=0;
ln(betas/beta_)-rho_B*ln(betas(-1)/beta_)-eB=0;
ln(XiL)-rho_XiL*ln(XiL(-1))-eXiL=0;
ln(Lpref)-rho_lpref*ln(Lpref(-1))-eLpref=0;
ln(EFFK)-rho_EFFK*ln(EFFK(-1))-eEFFK=0;
ln(MUZK/MUZKSS)-eMUZK=0;
ln(MUZM/MUZMSS)-eMUZM=0;
ln(HG)-rho_HG*ln(HG(-1))-eHG=0;
ln(MUC)-ln(MUZM)-alpha_*ln(MUZK)=0;
ln(MUK)-ln(MUZM)-ln(MUZK)=0;
ln(EFFECD)-rho_EFFECD*ln(EFFECD(-1))-eEFFECD=0;
ln(EFFECH)-rho_EFFECH*ln(EFFECH(-1))-eEFFECH=0;
ln(STAR)-rho_STAR*ln(STAR(-1))-eSTAR=0;
ln(RL1) - ln(R(+1))=0;
ln(RL2) - ln(RL1(+1))=0;
ln(RL3) - ln(RL2(+1))=0;
ln(RL4) - ln(RL3(+1))=0;
ln(RL5) - ln(RL4(+1))=0;
ln(RL6) - ln(RL5(+1))=0;
ln(RL7) - ln(RL6(+1))=0;
ln(RT2) - tp2 - 0.125*(ln(R) + ln(RL1) + ln(RL2) + ln(RL3) + ln(RL4) + ln(RL5) + ln(RL6) + ln(R
```

Uses alpha_ 14, beta_ 14, betas 11, delta_ch 14, DIFFREALECD 11, DIFFREALECH 11, eB 13, ECD 11, ECH 11, ECHSS 14, eEFFECD 13, eEFFECH 13, eEFFK 13, EFFECD 11, EFFECH 11, EFFK 11, eHG 13, eLpref 13, eMUZK 13, eMUZM 13, eSTAR 13, eta_ch 14, eXiL 13, gam_ech 14, h_ch 14, HG 11, KCH 11, KD 11, L 11, LAGKCH 11, LAGKD 11, Lpref 11, mu_ 14, MUC 11, MUK 11, MUZK 11, MUZKSS 14, MUZM 11, MUZMSS 14, phi_ech 14, phi_u 14, QCH 11, QK 11, R 11, RC 11, RCH 11, rho_B 14, rho_EFFECD 14, rho_EFFECH 14, rho_EFFK 14, rho_HG 14, rho_lpref 14, rho_STAR 14, rho_XiL 14, RK 11, RL1 11, RL2 11, RL3 11, RL4 11, RL5 11, RL6 11, RL7 11, RT2 11, STAR 11, tp2 14, UC 11, UK 11, and XiL 11.

A.1.14 Dynare EDO Model Measurement

26a ⟨*edo model measurement* 26a⟩≡ (21)

```
ln(DIFFREALGDP_obs/DIFFREALGDPSS_obs) = ln(DIFFREALGDP/DIFFREALGDPSS);
ln(DIFFREALEC_obs/DIFFREALECSS_obs)   = ln(DIFFREALEC/DIFFREALECSS);
ln(DIFFREALEIK_obs/DIFFREALEIKSS_obs) = ln(DIFFREALEIK/DIFFREALEIKSS);
ln(DIFFREALECD_obs/DIFFREALECDSS_obs) = ln(DIFFREALECD/DIFFREALECDSS);
ln(DIFFREALECH_obs/DIFFREALECHSS_obs) = ln(DIFFREALECH/DIFFREALECHSS);
ln(DIFFREALW_obs/DIFFREALWSS_obs)     = ln(DIFFREALW/DIFFREALWSS);
ln(AH_obs)                            = ln(AH/AHSS);
ln(INFCNA_obs/INFCNASS_obs)           = ln(INFCNA/INFCNASS);
ln(INFCOR_obs/INFCORSS_obs)           = ln(INFCOR/INFCORSS);
ln(INFK_obs/INFKSS_obs)               = ln(INFK/INFKSS);
ln(R_obs/RSS_obs)                     = ln(R/RSS);
ln(RT2_obs/RT2SS_obs)                 = ln(RT2/RT2SS);
ln(unemp_obs/unempSS_obs)             = ln(unemp/unempSS);
```

Uses AH 11, AH_obs 11, AHSS 14, DIFFREALEC 11, DIFFREALEC_obs 11, DIFFREALECD 11, DIFFREALECD_obs 11, DIFFREALECDSS 14, DIFFREALECDSS_obs 14, DIFFREALECH 11, DIFFREALECH_obs 11, DIFFREALECHSS 14, DIFFREALECHSS_obs 14, DIFFREALECSS 14, DIFFREALECSS_obs 14, DIFFREALEIK 11, DIFFREALEIK_obs 11, DIFFREALEIKSS 14, DIFFREALEIKSS_obs 14, DIFFREALGDP 11, DIFFREALGDP_obs 11, DIFFREALGDPSS 14, DIFFREALGDPSS_obs 14, DIFFREALW 11, DIFFREALW_obs 11, DIFFREALWSS 14, DIFFREALWSS_obs 14, INFCNA 11, INFCNA_obs 11, INFCNASS 14, INFCNASS_obs 14, INFCOR 11, INFCOR_obs 11, INFCORSS 14, INFCORSS_obs 14, INFK 11, INFK_obs 11, INFKSS 14, INFKSS_obs 14, R 11, R_obs 11, RSS 14, RSS_obs 14, RT2 11, RT2_obs 11, RT2SS 14, RT2SS_obs 14, unemp 11, unemp_obs 11, unempSS 14, and unempSS_obs 14.

A.1.15 Dynare EDO Model VarObs

26b ⟨*edo model varobs* 26b⟩≡ (21)

```
varobs DIFFREALGDP_obs DIFFREALEC_obs DIFFREALEIK_obs DIFFREALECD_obs DIFFREALECH_obs
```

Uses AH_obs 11, DIFFREALEC_obs 11, DIFFREALECD_obs 11, DIFFREALECH_obs 11, DIFFREALEIK_obs 11, DIFFREALGDP_obs 11, DIFFREALW_obs 11, INFCNA_obs 11, INFCOR_obs 11, INFK_obs 11, R_obs 11, RT2_obs 11, and unemp_obs 11.

A.1.16 Dynare EDO Shocks

27 ⟨*edo model shocks* 27⟩≡ (21)

```
var eHG;
stderr sig_HG;
var eXiL;
stderr sig_XiL;
var eLpref;
stderr sig_lpref;
var eR;
stderr sig_R;
var eMUZK;
stderr sig_MUZK;
var eMUZM;
stderr sig_MUZM;
var ePMKC;
stderr sig_PMKC;
var ePMKK;
stderr sig_PMKK;
var eEFFECH;
stderr sig_EFFECH;
var eEFFECD;
stderr sig_EFFECD;
var eEFFK;
stderr sig_EFFK;
var eB;
stderr sig_B;
var eSTAR;
stderr sig_STAR;


var  DIFFREALGDP_obs;
stderr 0.3;
var  DIFFREALEC_obs;
stderr 0.1;
var  DIFFREALEIK_obs;
stderr 1.5;
var DIFFREALECD_obs;
stderr 1.5;
var DIFFREALECH_obs;
stderr 1.5;
var DIFFREALW_obs;
stderr 0.3;
var AH_obs;
stderr 0.3;
var INFCNA_obs;
```

```
stderr 0.5;
var INFCOR_obs;
stderr 0.05;
var INFK_obs;
stderr 0.2;
var RT2_obs;
stderr 0.1;
var unemp_obs;
stderr 4;
```

Uses AH_obs 11, DIFFREALEC_obs 11, DIFFREALECD_obs 11, DIFFREALECH_obs 11, DIFFREALEIK_obs 11, DIFFREALGDP_obs 11, DIFFREALW_obs 11, eB 13, eEFFECD 13, eEFFECH 13, eEFFK 13, eHG 13, eLpref 13, eMUZK 13, eMUZM 13, ePMKC 13, ePMKK 13, eR 13, eSTAR 13, eXiL 13, INFCNA_obs 11, INFCOR_obs 11, INFK_obs 11, RT2_obs 11, sig_B 14, sig_EFFECD 14, sig_EFFECH 14, sig_EFFK 14, sig_HG 14, sig_lpref 14, sig_MUZK 14, sig_MUZM 14, sig_PMKC 14, sig_PMKK 14, sig_R 14, sig_STAR 14, sig_XiL 14, and unemp_obs 11.

A.1.17 Dynare EDO Model Estimated Params

28 ⟨*edo model estimated_params* 28⟩≡ (21)

```
h                 , .673       , -1        , 1    , uniform_pdf    ,,,-1
r_inf             , 1.461      , -999      , 999  , normal_pdf     , 1.5000
r_y               , 0.214      , -999      , 999  , normal_pdf     , 0.125
phi_pc            , 3.126      , 0         , 999  , gamma_pdf      , 4.0000
phi_H             , 4.064      , 0         , 999  , gamma_pdf      , 4.0000
phi_wc            , 5.119      , 0         , 999  , gamma_pdf      , 4.0000
phi_ic            , .325       , 0         , 999  , gamma_pdf      , 4.0000
phi_cd            , .651       , 0         , 999  , gamma_pdf      , 4.0000
phi_ech           , 10.948     , 0         , 999  , gamma_pdf      , 4.0000
gam_pc            , 0.386      , -999      , 999  , normal_pdf     , 0.000
gam_wc            , 0.213      , -999      , 999  , normal_pdf     , 0.000
sigman            , 1.25       , 0         , 999  , gamma_pdf      , 1.25
sigmah            , 10         , 0         , 999  , gamma_pdf      , 10
rho_R             , 0.654      , -1        , 1    , normal_pdf     , 0.5
rho_XiL           , 0.654      , -1        , 1    , normal_pdf     , 0.5
rho_lpref         , 0.954      , -1        , 1    , normal_pdf     , 0.5
rho_B             , 0.825      , -1        , 1    , normal_pdf     , 0
rho_STAR          , 0.825      , -1        , 1    , normal_pdf     , 0
rho_EFFK          , 0.850      , -1        , 1    , normal_pdf     , 0
rho_EFFECD        , .230       , -1        , 1    , normal_pdf     , 0
rho_HG            , 0.596      , 0         , 1    , beta_pdf       , 0.5
rho_EFFECH        , 0.844      , -1        , 1    , normal_pdf     , 0
tp2               , 0.001      , -999      , 999  , normal_pdf     , 0.0
```

Uses gam_pc 14, gam_wc 14, h 14, phi_cd 14, phi_ech 14, phi_H 14, phi_ic 14, phi_pc 14, phi_wc 14, r_inf 14, r_y 14, rho_B 14, rho_EFFECD 14, rho_EFFECH 14, rho_EFFK 14, rho_HG 14, rho_lpref 14, rho_R 14, rho_STAR 14, rho_XiL 14, sigmah 14, sigman 14, and tp2 14.

A.1.18 Dynare EDO Model Stderr

29 ⟨*edo model stderr* 29⟩≡ (21)

```
stderr eHG      , .745     , 0.0001    , 999  , inv_gamma_pdf , 1.772454    , Inf;
stderr eXiL     , 3.621    , 0.0001    , 999  , inv_gamma_pdf , 1.772454    , Inf;
stderr eLpref   , 1.621    , 0.0001    , 999  , inv_gamma_pdf , 1.772454    , Inf;
stderr eR       , 0.165    , 0.0001    , 999  , inv_gamma_pdf , 0.354491    , Inf;
stderr eMUZK    , .834     , 0.0001    , 999  , inv_gamma_pdf , 0.443113    , Inf;
stderr eMUZM    , .484     , 0.0001    , 999  , inv_gamma_pdf , 0.443113    , Inf;
stderr ePMKC    , .391     , 0.0001    , 999  , inv_gamma_pdf , 0.354491    , Inf;
stderr ePMKK    , .552     , 0.0001    , 999  , inv_gamma_pdf , 0.354491    , Inf;
stderr eEFFECH  , .526     , 0.0001    , 999  , inv_gamma_pdf , 1.772454    , Inf;
stderr eEFFECD  , 13.349   , 0.0001    , 999  , inv_gamma_pdf , 1.772454    , Inf;
stderr eEFFK    , .499     , 0.0001    , 999  , inv_gamma_pdf , 1.772454    , Inf;
stderr eB       , 0.5      , 0.0001    , 999  , inv_gamma_pdf , 1.772454    , Inf;
stderr eSTAR    , 0.05     , 0.0001    , 999  , inv_gamma_pdf , 0.354491    , Inf;
```

Uses `eB` 13, `eEFFECD` 13, `eEFFECH` 13, `eEFFK` 13, `eHG` 13, `eLpref` 13, `eMUZK` 13, `eMUZM` 13, `ePMKC` 13, `ePMKK` 13, `eR` 13, `eSTAR` 13, and `eXiL` 13.

A.2 Dynare_edo_steadystate.m

30 ⟨*srcedo/Dynare.edo.steadystate.m* 30⟩≡

```
function [ys,check] = unlinearized_edo_steadystate(ys,exe)
        global M_

check = 0;

NumberofParameters=M_.param_nbr;
for i=1:NumberofParameters
    paramname=deblank(M_.param_names(i,:));
    eval([paramname ' =M_.params(' int2str(i) ');']);
end;

%start_steady_state;

⟨edo steady state values 31⟩

%end_steady_state;

%trends;

⟨edo steady state trends 34⟩

%end_trends;


for i=1:NumberofParameters
    paramname=deblank(M_.param_names(i,:));
    eval(['M_.params(' int2str(i) ')=' paramname ';']);
end;


⟨edo steady state result return 35⟩
```

This code is written to file `srcedo/Dynare.edo.steadystate.m`.
Defines:
`unlinearized_edo_steadystate`, never used.

A.2.1 EDO Steady State Values

31 ⟨*edo steady state values* 31⟩≡ (30)

```
beta_0 = pbeta;
beta_2 = pbeta*rpr; % s.s. funds rate premium
beta_ = beta_2;
MUZCSS=1;
ONE=1;
USS=1;
MUKSS=MUZKSS*MUZMSS;
MUCSS=MUZKSS^alpha_*MUZMSS;
MUKSShabit=MUKSS;
MUCSShabit=MUCSS;
PKBSS=theta_k/(theta_k-1)*(theta_c-1)/theta_c;
PYSS=1;
MCCSS=(theta_c-1)/theta_c;
MCKSS=(theta_k-1)/theta_k;
RKSS=MUKSS/beta_2-(1-delta_);
RCSS=MUKSS/beta_2-(1-delta_);
RCHSS=MUCSS/beta_2-(1-delta_ch); % Housing sector
RCDSS=MUKSS/beta_2-(1-delta_cd); % Durable sector
USS=1;
mu_=RCSS;
AA=alpha_/RKSS*MCKSS;
DD = 0.135;
RR = 0.075;
eta_cnn=1;
eta_cd_eta_cnn=DD/((MUKSShabit-beta_2*h_cd)/(1-beta_2*h/MUCSShabit)*(1-h/MUCSShabit)/(1-h_cd/MU
eta_ch_eta_cnn=RR/((MUCSShabit-beta_2*h_ch)/(1-beta_2*h/MUCSShabit)*(1-h/MUCSShabit)/(1-h_ch/MU
eta_ch=eta_ch_eta_cnn;
eta_cd=eta_cd_eta_cnn;
DD=eta_cd_eta_cnn*(MUKSShabit-beta_2*h_cd)/(1-beta_2*h/MUCSShabit)*(1-h/MUCSShabit)/(1-h_cd/MUK
RR=eta_ch_eta_cnn*(MUCSShabit-beta_2*h_ch)/(1-beta_2*h/MUCSShabit)*(1-h/MUCSShabit)/(1-h_ch/MUC
Rnr=(1-(1-delta_)/MUKSS)*AA*MUKSS;
ycbi_ykb=((1-s_AS)-Rnr)/((DD*(1-s_AS)/(1+RR))+Rnr);
hc_hk=ycbi_ykb*(RCSS*MCKSS/(RKSS*MCCSS))^(alpha_/(1-alpha_));
HSS=0.25;
AHSS=HSS;
HKSS=HSS/(1+hc_hk);
HCSS=HSS-HKSS;
HrCSS=1/3;
HrKSS=1/3;
empCSS=HCSS/HrCSS;
empKSS=HKSS/HrKSS;
ycbi=HCSS*(AA)^(alpha_/(1-alpha_));
ykb=HKSS*(AA)^(alpha_/(1-alpha_));
```

```
YCSS=ycbi;
YKSS=ykb;
KCSS=AA*ycbi*MUKSS;
KKSS=AA*ykb*MUKSS;
ECHSS=RR/(1+RR)*ycbi*(1-s_AS);
ECSS=1/(1+RR)*ycbi*(1-s_AS);
ECDSS=DD*PKBSS*ECSS;
EIKSS=(1-(1-delta_)/MUKSS)*(KCSS+KKSS);
KCDSS=ECDSS/(1-(1-delta_cd)/MUKSS);
KCHSS=ECHSS/(1-(1-delta_ch)/MUCSS);
YYSS=(YCSS+YKSS*PKBSS)/PYSS;
s_k_ecd=ECDSS/YKSS;
s_c_ech=ECHSS/YCSS;
s_k_eik=EIKSS/YKSS;
s_yc = (YCSS/YYSS);
s_ecdc=PKBSS*ECDSS/(ECSS+PKBSS*ECDSS+(MUCSS/beta_2-1+delta_ch)*KCHSS);
INFCNASS=exp(.02/4);
INFCSS = INFCNASS*((MUZCSS/MUZKSS)^(1-alpha_))^(-s_ecdc);
INFCORSS=INFCNASS;
INFKSS=INFCSS*(MUZCSS/MUZKSS)^(1-alpha_);
INFWCSS=INFCSS*MUZKSS^alpha_*MUZMSS;
INFWKSS=INFWCSS;
RSS=INFCSS/beta_0*MUCSS;
RT2SS=exp(tp2)*RSS;
INFC10SS = INFCNASS;
IMPHSSS = RCHSS*KCHSS;
s_k=PKBSS*YKSS/YYSS;
INFGDPSS=INFCSS^(YCSS/YYSS)*INFKSS^(YKSS*PKBSS/(YYSS));
LSS=eta_cnn/(ECSS*(1-h/MUCSShabit))-eta_cnn*beta_2*h/(ECSS*(MUCSShabit-h));
WCSS=MCCSS*(1-alpha_)*YCSS/HCSS;
WKSS=MCKSS*(1-alpha_)*YKSS/HKSS;
xsiN_xsiH_C = ((HrCSS/empCSS)^(1+sigmah))/(1+1/sigmah);
xsiN_xsiH_K = ((HrKSS/empKSS)^(1+sigmah))/(1+1/sigmah);
gC = (1/(1+sigman) + 1/sigmah)*(xsiN_xsiH_C*(1+sigmah)/sigmah)^(-(1+sigman)/(1+sigman
markup_xsiN_C = (HCSS^((1+sigmah)*(1+sigman)/(1+sigmah+sigman)-1))*gC/(LSS*WCSS);
gK = (1/(1+sigman) + 1/sigmah)*(xsiN_xsiH_K*(1+sigmah)/sigmah)^(-(1+sigman)/(1+sigman
markup_xsiN_K = (HKSS^((1+sigmah)*(1+sigman)/(1+sigmah+sigman)-1))*gK/(LSS*WKSS);
markup_w = (1-unempSS)^((1+sigmah+sigman)/(1+sigmah) - 1 - sigman);
theta_wc = markup_w/(markup_w -1); theta_wk = theta_wc;
A_HC=LSS*(theta_wc-1)/theta_wc*WCSS/(((1+sigman)/(1+sigman/(1+sigmah)))*HCSS^(-1+(1+s
A_HK=LSS*(theta_wk-1)/theta_wk*WKSS/(((1+sigman)/(1+sigman/(1+sigmah)))*HKSS^(-1+(1+s
xsi_NC=A_HC/((1/(1+sigman)+1/sigmah)*(HCSS^sigman/HrCSS^(1+sigman+sigmah))^((1+sigman
xsi_NK=A_HK/((1/(1+sigman)+1/sigmah)*(HKSS^sigman/HrKSS^(1+sigman+sigmah))^((1+sigman
xsi_HrC=xsi_NC*(1+sigmah)/sigmah*(HCSS^sigman/HrCSS^(1+sigman+sigmah));
xsi_HrK=xsi_NK*(1+sigmah)/sigmah*(HKSS^sigman/HrKSS^(1+sigman+sigmah));
UHCSS=A_HC*((1+sigman)/(1+sigman/(1+sigmah)))*HCSS^(-1+(1+sigman)/(1+sigman/(1+sigmah
```

```
UHKSS=A_HK*((1+sigman)/(1+sigman/(1+sigmah)))*HKSS^(-1+(1+sigman)/(1+sigman/(1+sigmah)))/LSS;
HSCSS=(WCSS*LSS/(A_HC*((1+sigman)/(1+sigman/(1+sigmah)))))^(1/(-1+(1+sigman)/(1+sigman/(1+sigma
HSKSS=(WKSS*LSS/(A_HK*((1+sigman)/(1+sigman/(1+sigmah)))))^(1/(-1+(1+sigman)/(1+sigman/(1+sigma
empSCSS=((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(-1/(1+sigmah+sigman))*HSCSS^(1/(1+sigman/(1+sigmah)
empSKSS=((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(-1/(1+sigmah+sigman))*HSKSS^(1/(1+sigman/(1+sigmah)
HrSCSS=HSCSS/empSCSS;
HrSKSS=HSKSS/empSKSS;
UHSCSS=A_HC*((1+sigman)/(1+sigman/(1+sigmah)))*HSCSS^(-1+(1+sigman)/(1+sigman/(1+sigmah)))/LSS;
UHSKSS=A_HK*((1+sigman)/(1+sigman/(1+sigmah)))*HSKSS^(-1+(1+sigman)/(1+sigman/(1+sigmah)))/LSS;
unempSS=(empSCSS+empSKSS-(empCSS+empKSS))/(empSCSS+empSKSS);
QKSS=1;
QCDSS=1;
QCHSS=1;
UCSS=1;
UKSS=1;
XiBSS=1;
XiDSS=1;
XiHSS=1;
RL1SS=RSS;
RL2SS=RSS;
RL3SS=RSS;
RL4SS=RSS;
RL5SS=RSS;
RL6SS=RSS;
RL7SS=RSS;
DIFFREALECSS =exp( log(MUCSS));
DIFFREALEIKSS =exp( log(MUKSS));
DIFFREALECDSS =exp( log(MUKSS));
DIFFREALECHSS =exp( log(MUCSS));
DIFFREALWSS =exp( log(MUCSS) );
DIFFREALGDPSS =exp( (1-s_k)*log(MUCSS)+(s_k)*log(MUKSS));
```

Uses A_HC 14, A_HK 14, AA 14, AHSS 14, alpha_ 14, beta_ 14, beta_0 14, beta_2 14, DD 14, delta_ 14, delta_cd 14, delta_ch 14, DIFFREALECDSS 14, DIFFREALECHSS 14, DIFFREALECSS 14, DIFFREALEIKSS 14, DIFFREALGDPSS 14, DIFFREALWSS 14, ECDSS 14, ECHSS 14, ECSS 14, EIKSS 14, empCSS 14, empKSS 14, empSCSS 14, empSKSS 14, eta_cd 14, eta_cd_eta_cnn 14, eta_ch 14, eta_ch_eta_cnn 14, eta_cnn 14, h 14, h_cd 14, h_ch 14, hc_hk 14, HCSS 14, HKSS 14, HrCSS 14, HrKSS 14, HrSCSS 14, HrSKSS 14, HSCSS 14, HSKSS 14, HSS 14, IMPHSSS 14, INFC10SS 14, INFCNASS 14, INFCORSS 14, INFCSS 14, INFGDPSS 14, INFKSS 14, INFWCSS 14, INFWKSS 14, KCDSS 14, KCHSS 14, KCSS 14, KKSS 14, LSS 14, MCCSS 14, MCKSS 14, mu_ 14, MUCSS 14, MUCSShabit 14, MUKSS 14, MUKSShabit 14, MUZCSS 14, MUZKSS 14, MUZMSS 14, ONE 14, pbeta 14, PKBSS 14, PYSS 14, QCDSS 14, QCHSS 14, QKSS 14, RCDSS 14, RCHSS 14, RCSS 14, RKSS 14, RL1SS 14, RL2SS 14, RL3SS 14, RL4SS 14, RL5SS 14, RL6SS 14, RL7SS 14, Rnr 14, rpr 14, RR 14, RSS 14, RT2SS 14, s_AS 14, s_c_ech 14, s_ecdc 14, s_k 14, s_k_ecd 14, s_k_eik 14, s_yc 14, sigmah 14, sigman 14, theta_c 14, theta_k 14, theta_wc 14, theta_wk 14, tp2 14, UCSS 14, UHCSS 14, UHKSS 14, UHSCSS 14, UHSKSS 14, UKSS 14, unempSS 14, USS 14, WCSS 14, WKSS 14, xsi_HrC 14, xsi_HrK 14, xsi_NC 14, xsi_NK 14, ycbi 14, ycbi_ykb 14, YCSS 14, ykb 14, YKSS 14, and YYSS 14.

A.2.2 EDO Steady State Trends

34 ⟨*edo steady state trends* 34⟩≡ (30)

```
DIFFREALGDPSS_obs=(1-s_k)*log(MUCSS)*100+(s_k)*log(MUKSS)*100;
DIFFREALECSS_obs=log(MUCSS)*100;
DIFFREALEIKSS_obs=log(MUKSS)*100;
DIFFREALECDSS_obs=log(MUKSS)*100;
DIFFREALECHSS_obs=log(MUCSS)*100;
DIFFREALWSS_obs=log(MUCSS)*100;
INFCNASS_obs=(1-s_ecdc)*log(INFCSS)*100+s_ecdc*log(INFKSS)*100;
INFCORSS_obs=(1-s_ecdc)*log(INFCSS)*100+s_ecdc*log(INFKSS)*100;
INFKSS_obs=log(INFCSS)*100-log(MUKSS)*100+log(MUCSS)*100;
RSS_obs=log(RSS)*100;
RT2SS_obs=log(RT2SS)*100;
unempSS_obs=100*log(unempSS);
```

Uses DIFFREALECDSS_obs 14, DIFFREALECHSS_obs 14, DIFFREALECSS_obs 14, DIFFREALEIKSS_obs 14, DIFFREALGDPSS_obs 14, DIFFREALWSS_obs 14, INFCNASS_obs 14, INFCORSS_obs 14, INFCSS 14, INFKSS 14, INFKSS_obs 14, MUCSS 14, MUKSS 14, RSS 14, RSS_obs 14, RT2SS 14, RT2SS_obs 14, s_ecdc 14, s_k 14, unempSS 14, and unempSS_obs 14.

A.2.3 EDO Steady State Result Return

35 ⟨*edo steady state result return* 35⟩≡ (30)

```
ys = [
RCSS
RKSS
WCSS
WKSS
YCSS
YKSS
MCCSS
MCKSS
KCSS
KKSS
PKBSS
RSS
LSS
QKSS
HCSS
HSCSS
HKSS
HSKSS
UHCSS
UHSCSS
UHKSS
UHSKSS
empCSS
HrCSS
empKSS
HrKSS
empSCSS
HrSCSS
empSKSS
HrSKSS
unempSS
EIKSS
ECSS
INFWCSS
INFWKSS
INFCSS
INFKSS
ONE
ONE
DIFFREALGDPSS
DIFFREALECSS
DIFFREALEIKSS
```

```
DIFFREALWSS
AHSS
INFGDPSS
INFCNASS
INFCORSS
ONE
ONE
INFC10SS
ECDSS
KCDSS
RCDSS
QCDSS
KCHSS
RCHSS
ECHSS
QCHSS
KCDSS
KCHSS
USS
USS
DIFFREALECHSS
DIFFREALECDSS
beta_
ONE
ONE
ONE
MUZKSS
MUZMSS
ONE
MUCSS
MUKSS
ONE
ONE
ONE
RL1SS
RL2SS
RL3SS
RL4SS
RL5SS
RL6SS
RL7SS
RT2SS
DIFFREALGDPSS_obs
DIFFREALECSS_obs
DIFFREALEIKSS_obs
DIFFREALECDSS_obs
```

```
DIFFREALECHSS_obs
DIFFREALWSS_obs
ONE
INFCNASS_obs
INFCORSS_obs
INFKSS_obs
RSS_obs
RT2SS_obs
unempSS_obs
];
```

Uses AHSS 14, beta_ 14, DIFFREALECDSS 14, DIFFREALECDSS_obs 14, DIFFREALECHSS 14, DIFFREALECHSS_obs 14, DIFFREALECSS 14, DIFFREALECSS_obs 14, DIFFREALEIKSS 14, DIFFREALEIKSS_obs 14, DIFFREALGDPSS 14, DIFFREALGDPSS_obs 14, DIFFREALWSS 14, DIFFREALWSS_obs 14, ECDSS 14, ECHSS 14, ECSS 14, EIKSS 14, empCSS 14, empKSS 14, empSCSS 14, empSKSS 14, HCSS 14, HKSS 14, HrCSS 14, HrKSS 14, HrSCSS 14, HrSKSS 14, HSCSS 14, HSKSS 14, INFC10SS 14, INFCNASS 14, INFCNASS_obs 14, INFCORSS 14, INFCORSS_obs 14, INFCSS 14, INFGDPSS 14, INFKSS 14, INFKSS_obs 14, INFWCSS 14, INFWKSS 14, KCDSS 14, KCHSS 14, KCSS 14, KKSS 14, LSS 14, MCCSS 14, MCKSS 14, MUCSS 14, MUKSS 14, MUZKSS 14, MUZMSS 14, ONE 14, PKBSS 14, QCDSS 14, QCHSS 14, QKSS 14, RCDSS 14, RCHSS 14, RCSS 14, RKSS 14, RL1SS 14, RL2SS 14, RL3SS 14, RL4SS 14, RL5SS 14, RL6SS 14, RL7SS 14, RSS 14, RSS_obs 14, RT2SS 14, RT2SS_obs 14, UHCSS 14, UHKSS 14, UHSCSS 14, UHSKSS 14, unempSS 14, unempSS_obs 14, USS 14, WCSS 14, WKSS 14, YCSS 14, and YKSS 14.

A.3 linearized.mod

XXX YYY

38 ⟨*srcedo/linearized.mod* 38⟩≡

```
var RC RK WC WK YC YK MCC MCK KC  KK PKB R L QK HC  HSC  HK HSK  UHC UHSC UHK UHSK e
 DIFFREALGDP_obs DIFFREALEC_obs DIFFREALEIK_obs DIFFREALECD_obs DIFFREALECH_obs DIFF


varexo eHG eXiL eLpref eR eMUZK eMUZM ePMKC ePMKK eEFFECH eEFFECD eEFFK eB eSTAR;


parameters
h r_inf r_y r_dy phi_pc phi_H phi_wc phi_ic phi_cd phi_ech gam_pc gam_wc gam_ic gam_
rho_EFFECD rho_HG rho_EFFECH tp2 ONE MUZMSS MUZKSS  r_dinf rpr phi_u rho_MUZK rho_MU
theta_k theta_wc theta_wk g_y a_ks s_AS gam_h gam_ech  s_k s_ecdc eta_cnn eta_cd eta
icoef mu_ betarl MUZCSS RCSS RKSS WCSS WKSS YCSS YKSS MCCSS MCKSS KCSS KKSS LSS HCSS
MUCSS MUKSS AHSS ECDSS KCDSS QCDSS RCDSS ECHSS KCHSS QCHSS RCHSS UKSS UCSS USS MUKSS
INFCNASS INFCORSS INFC10SS  RT2SS beta_0 beta_2 beta_ PYSS AA DD RR
eta_cd_eta_cnn eta_ch_eta_cnn Rnr ycbi_ykb hc_hk HSS ycbi ykb YYSS s_k_ecd s_c_ech s
sig_HG sig_XiL sig_lpref sig_R sig_MUZK sig_MUZM sig_PMKC sig_PMKK sig_EFFECH  sig_E
HSKSS HSCSS HrCSS HrKSS A_HC sigman sigmah A_HK xsi_NC xsi_HrC xsi_NK xsi_HrK rho_Xi
empCSS empKSS HrSKSS HrSCSS empSCSS empSKSS UHCSS UHKSS UHSCSS UHSKSS unempSS DIFFRE
DIFFREALECHSS DIFFREALEIKSS DIFFREALWSS  RL1SS RL2SS RL3SS RL4SS RL5SS
RL6SS RL7SS DIFFREALGDPSS_obs DIFFREALECSS_obs DIFFREALEIKSS_obs DIFFREALECDSS_obs
DIFFREALECHSS_obs DIFFREALWSS_obs INFCNASS_obs INFCORSS_obs INFKSS_obs
RSS_obs RT2SS_obs unempSS_obs;


//estimated_params;
h               = 0.715162417869797;
r_inf           = 1.46344163969035;
r_y             = 0.263123294207851;
phi_pc          = 3.54471453295450;
phi_H           = 3.22894079106560;
phi_wc          = 5.49395755514723;
phi_ic          = 0.253308786976374;
phi_cd          = 0.470089385005009;
phi_ech         = 9.13986886546163;
gam_pc          = 0.314488926051065;
gam_wc          = -0.230018833252054;
sigman          = 39.4075260618789;
sigmah          = 21.8859803402692;
rho_R           = 0.833200065745674;
rho_XiL         = 0.263567746111198;
```

```
rho_lpref       = 0.979092048897712;
rho_B           = 0.895267027146152;
rho_STAR        = 0.909187927454138;
rho_EFFK        = 0.937829274540004;
rho_EFFECD      = -0.240286975088701;
rho_HG          = 0.582395471123139;
rho_EFFECH      = 0.877235725078934;
tp2             = 0.000307314910763576;
sig_HG          = 0.579315931803017;
sig_XiL         = 2.49313873916751;
sig_lpref       = 5.66476748114241;
sig_R           = 0.124100461010359;
sig_MUZK        = 0.936167718269030;
sig_MUZM        = 0.597390920898135;
sig_PMKC        = 0.451830653200989;
sig_PMKK        = 0.685376191952156;
sig_EFFECH      = 0.514704527091087;
sig_EFFECD      = 9.11199585973990;
sig_EFFK        = 0.402779878811407;
sig_B           = 0.295232712196573;
sig_STAR        = 0.104877885500673;
//end_estimated_params;

//calibrated_params;
r_dy = 0;
ONE = 1;
MUZKSS = 1.009250;
MUZMSS = 1.001000;
gam_ic = 1.0;
gam_icd = 1.0;
r_dinf = 0;
rpr = 0.965;
phi_u = 1;
rho_MUZK = 0;
rho_MUZM = 0;
pbeta = 0.99862;
delta_ = 0.03;
h_cd = 0.0;
h_ch = 0.0;
delta_cd = 0.055;
delta_ch = 0.0035;
alpha_ = 0.26;
theta_c = 7;
theta_k = 7;
unempSS = .06;
g_y = 0.0;
```

```
a_ks = 0.2;
s_AS = 0.2;
gam_h = 1;
gam_ech = 1;
icoef = 3;
betarl = .958;
//end_calibrated_params;

//free_params;
//A_HC;
//A_HK;
//xsi_NC;
//xsi_HrC;
//xsi_NK;
//xsi_HrK;
//theta_wc;
//theta_wk;
//infkbar;
//infcbar;
//infwcbar;
//infwkbar;
//Pybar;
//Yybar;
//mu_yc;
//mu_yk;
//s_k;
//s_ecdc;
//eta_cnn;
//eta_cd;
//eta_ch;
//mu_;
//end_free_params;

//calibrated ME


//****************************
//MODEL BLOCK
//****************************

model;
(RCSS*exp(RC))-(MCCSS*exp(MCC))*(YCSS*exp(YC))/(USS*exp(UC))/(KCSS*exp(KC(-1)))*alpha
(RKSS*exp(RK))-(MCKSS*exp(MCK))*(YKSS*exp(YK))/(USS*exp(UK))/(KKSS*exp(KK(-1)))*alpha
(WCSS*exp(WC))-(MCCSS*exp(MCC))*(YCSS*exp(YC))/(HCSS*exp(HC))*(1-alpha_)=0;
(WKSS*exp(WK))-(MCKSS*exp(MCK))*(YKSS*exp(YK))/(HKSS*exp(HK))*(1-alpha_)=0;
(YCSS*exp(YC))-((USS*exp(UC))*(KCSS*exp(KC(-1)))/(MUKSS*exp(MUK)))^alpha_*((HCSS*exp(
```

```
(YKSS*exp(YK))-((USS*exp(UK))*(KKSS*exp(KK(-1)))/(MUKSS*exp(MUK)))^alpha_*((HKSS*exp(HK)))^(1-a
(MCCSS*exp(MCC))*(YCSS*exp(YC))*theta_c-(theta_c-1)*(YCSS*exp(YC))-100*phi_pc*((INFCSS*exp(INFC
(MCKSS*exp(MCK))*(YKSS*exp(YK))*theta_k/(PKBSS*exp(PKB))-(theta_k-1)*(YKSS*exp(YK))-100*phi_pc*
(QKSS*exp(QK))-beta_*(1/(ONE*exp(EFFK)))*(((1-delta_)*(QKSS*exp(QK(+1)))+(RCSS*exp(RC(+1)))*(US
(QKSS*exp(QK))-beta_*(1/(ONE*exp(EFFK)))*(((1-delta_)*(QKSS*exp(QK(+1)))+(RKSS*exp(RK(+1)))*(US
(LSS*exp(L))-(beta_*exp(betas))*(RSS*exp(R))/rpr/(INFCSS*exp(INFC(+1)))/(MUCSS*exp(MUC(+1)))*(L
ln((RSS*exp(R))/RSS)-rho_R*ln((RSS*exp(R(-1)))/RSS)-(1-rho_R)*(r_inf*ln((INFCNASS*exp(INFCNA))/
(LSS*exp(L))-eta_cnn/((ECSS*exp(EC))-h*(ECSS*exp(EC(-1)))/(MUCSS*exp(MUC)))+eta_cnn*beta_*h/((M
(KKSS*exp(KK))-(1-delta_)*(KKSS*exp(KK(-1)))/(MUKSS*exp(MUK))+(KCSS*exp(KC))-(1-delta_)*(KCSS*e

// XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
// labor block
// TOTAL LABOR INPUT (called "(LSS*exp(L))" in the paper, I kept the "H" notation of the origin
-100+(UHCSS*exp(UHC))*theta_wc-(theta_wc-1)*(WCSS*exp(WC))-100*phi_wc*((INFWCSS*exp(INFWC))-gam
(UHSCSS*exp(UHSC))-(WCSS*exp(WC))+phi_H/10*((HSCSS*exp(HSC))/(HSKSS*exp(HSK))-gam_h*(HSCSS*exp(
-100+(UHKSS*exp(UHK))*theta_wk-(theta_wk-1)*(WKSS*exp(WK))-100*phi_wc*((INFWKSS*exp(INFWK))-gam
(UHSKSS*exp(UHSK))-(WKSS*exp(WK))-phi_H/10*((HSCSS*exp(HSC))/(HSKSS*exp(HSK))-gam_h*(HSCSS*exp(
(UHCSS*exp(UHC))*(LSS*exp(L))*(ONE*exp(Lpref))-A_HC*((1+sigman)/(1+sigman/(1+sigmah)))*((HCSS*e
(UHSCSS*exp(UHSC))*(LSS*exp(L))*(ONE*exp(Lpref))-A_HC*((1+sigman)/(1+sigman/(1+sigmah)))*((HSCS
(UHKSS*exp(UHK))*(LSS*exp(L))*(ONE*exp(Lpref))-A_HK*((1+sigman)/(1+sigman/(1+sigmah)))*((HKSS*e
(UHSKSS*exp(UHSK))*(LSS*exp(L))*(ONE*exp(Lpref))-A_HK*((1+sigman)/(1+sigman/(1+sigmah)))*((HSKS
(empCSS*exp(empC))-((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(-1/(1+sigmah+sigman))*(HCSS*exp(HC))^(1/
(HrCSS*exp(HrC))-((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(1/(1+sigmah))*(empCSS*exp(empC))^(sigman/(
(empKSS*exp(empK))-((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(-1/(1+sigmah+sigman))*(HKSS*exp(HK))^(1/
(HrKSS*exp(HrK))-((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(1/(1+sigmah))*(empKSS*exp(empK))^(sigman/(
(empSCSS*exp(empSC))-((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(-1/(1+sigmah+sigman))*(HSCSS*exp(HSC))
(HrSCSS*exp(HrSC))-((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(1/(1+sigmah))*(empSCSS*exp(empSC))^(sigm
(empSKSS*exp(empSK))-((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(-1/(1+sigmah+sigman))*(HSKSS*exp(HSK))
(HrSKSS*exp(HrSK))-((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(1/(1+sigmah))*(empSKSS*exp(empSK))^(sigm
(unempSS*exp(unemp))-((empSCSS*exp(empSC))+(empSKSS*exp(empSK))-((empCSS*exp(empC))+(empKSS*exp
(PKBSS*exp(PKB))-(1-100*phi_ic*((EIKSS*exp(EIK))-gam_ic*(EIKSS*exp(EIK(-1)))-(1-gam_ic)*EIKSS)/
(YCSS*exp(YC))-(ECSS*exp(EC))-(ECHSS*exp(ECH))-0.2*YCSS*(ONE*exp(HG))=0;
ln((INFWCSS*exp(INFWC)))-ln((WCSS*exp(WC)))+ln((WCSS*exp(WC(-1))))-ln((MUCSS*exp(MUC)))-ln((INF
ln((INFWKSS*exp(INFWK)))-ln((WKSS*exp(WK)))+ln((WKSS*exp(WK(-1))))-ln((MUCSS*exp(MUC)))-ln((INF
ln((INFKSS*exp(INFK)))-ln((INFCSS*exp(INFC)))-ln((PKBSS*exp(PKB)))+ln((PKBSS*exp(PKB(-1))))+ln(
(YKSS*exp(YK))-(EIKSS*exp(EIK))-(ECDSS*exp(ECD))-0.2*YKSS*(ONE*exp(HG))=0;
ln((ONE*exp(DIFFNORMGDP)))-(1-s_k)*(ln((YCSS*exp(YC)))-ln((YCSS*exp(YC(-1)))))-s_k*(ln((YKSS*ex
ln((ONE*exp(NORMINFGDP)))-s_k*(ln((PKBSS*exp(PKB)))-ln((PKBSS*exp(PKB(-1)))))=0;
ln((DIFFREALGDPSS*exp(DIFFREALGDP)))-ln((ONE*exp(DIFFNORMGDP)))-(1-s_k)*ln((MUCSS*exp(MUC)))-s_
ln((DIFFREALECSS*exp(DIFFREALEC)))-ln((ECSS*exp(EC))+ln((ECSS*exp(EC(-1))))-ln((MUCSS*exp(MUC)
ln((DIFFREALEIKSS*exp(DIFFREALEIK)))-ln((EIKSS*exp(EIK))+ln((EIKSS*exp(EIK(-1))))-ln((MUKSS*ex

// Identities
ln((DIFFREALWSS*exp(DIFFREALW)))-HCSS/AHSS*(ln((INFWCSS*exp(INFWC))))-HKSS/AHSS*(ln((INFWKSS*ex
```

```
// XXXXXXXXXXXXXXXXXXXXX
// Aggregate hours equals agg hours in each sector
(AHSS*exp(AH))-(HCSS*exp(HC))-(HKSS*exp(HK))=0;
ln((INFGDPSS*exp(INFGDP)))-ln((INFCSS*exp(INFC)))-ln((YCSS*exp(YC))*(MUCSS*exp(MUC)),
ln((INFCNASS*exp(INFCNA)))-(1-s_ecdc)*ln((INFCSS*exp(INFC)))-s_ecdc*ln((INFKSS*exp(IN
ln((INFCORSS*exp(INFCOR)))-(1-s_ecdc)*ln((INFCSS*exp(INFC)))-s_ecdc*ln((INFKSS*exp(IN
ln((ONE*exp(GAP)))-(1-s_k)*ln((YCSS*exp(YC))/YCSS)-s_k*ln((YKSS*exp(YK))/YKSS)=0;
ln((ONE*exp(PFGAP)))-(1-alpha_)*((1-s_k)*ln((HCSS*exp(HC))/HCSS)+s_k*ln((HKSS*exp(HK)
ln((INFC10SS*exp(INFC10)))-betarl*ln((INFC10SS*exp(INFC10(+1))))-(1-betarl)*ln((INFC(

// See Section 8: Data Identities

// new equations
// Durable Block

(KCDSS*exp(KD))-(1-delta_cd)*(KCDSS*exp(KD(-1)))/(MUKSS*exp(MUK))-(ECDSS*exp(ECD))=0;
(LSS*exp(L))*(RCDSS*exp(RCD))-eta_cd/((KCDSS*exp(KD(-1)))/(MUKSS*exp(MUK))-h_cd*(KCDS
(QCDSS*exp(QCD))-beta_*(1/(ONE*exp(EFFECD)))*(LSS*exp(L(+1)))/(LSS*exp(L))/(MUKSS*exp
(PKBSS*exp(PKB))-(QCDSS*exp(QCD))*(1-100*phi_cd*((ECDSS*exp(ECD))-gam_icd*(ECDSS*exp(

// Housing Block
(LSS*exp(L))*(RCHSS*exp(RCH))-eta_ch/((KCHSS*exp(KCH(-1)))/(MUCSS*exp(MUC))-h_ch*(KCH
(QCHSS*exp(QCH))-beta_*(1/(ONE*exp(EFFECH)))*(LSS*exp(L(+1)))/(LSS*exp(L))/(MUCSS*exp
1*(ECHSS*exp(ECH))+(1-delta_ch)*(KCHSS*exp(KCH(-1)))/(MUCSS*exp(MUC))-(KCHSS*exp(KCH)
1-(QCHSS*exp(QCH))*(1-100*phi_ech*((ECHSS*exp(ECH))-gam_ech*(ECHSS*exp(ECH(-1)))-(1-g
ln((KCDSS*exp(KD(-1))))-ln((KCDSS*exp(LAGKD)))=0;
ln((KCHSS*exp(KCH(-1))))-ln((KCHSS*exp(LAGKCH)))=0;
(RKSS*exp(RK))-(QKSS*exp(QK))*mu_*(USS*exp(UK))^(1/phi_u)=0;
(RCSS*exp(RC))-(QKSS*exp(QK))*mu_*(USS*exp(UC))^(1/phi_u)=0;
ln((DIFFREALECHSS*exp(DIFFREALECH)))-ln((MUCSS*exp(MUC)))-ln((ECHSS*exp(ECH)))+ln((EC
ln((DIFFREALECDSS*exp(DIFFREALECD)))-ln((MUKSS*exp(MUK)))-ln((ECDSS*exp(ECD)))+ln((EC
ln((beta_*exp(betas))/beta_)-rho_B*ln((beta_*exp(betas(-1)))/beta_)-eB=0;
ln((ONE*exp(XiL)))-rho_XiL*ln((ONE*exp(XiL(-1))))-eXiL=0;
ln((ONE*exp(Lpref)))-rho_lpref*ln((ONE*exp(Lpref(-1))))-eLpref=0;
ln((ONE*exp(EFFK)))-rho_EFFK*ln((ONE*exp(EFFK(-1))))-eEFFK=0;
ln((MUZKSS*exp(MUZK))/MUZKSS)-eMUZK=0;
ln((MUZMSS*exp(MUZM))/MUZMSS)-eMUZM=0;
ln((ONE*exp(HG)))-rho_HG*ln((ONE*exp(HG(-1))))-eHG=0;
ln((MUCSS*exp(MUC)))-ln((MUZMSS*exp(MUZM)))-alpha_*ln((MUZKSS*exp(MUZK)))=0;
ln((MUKSS*exp(MUK)))-ln((MUZMSS*exp(MUZM)))-ln((MUZKSS*exp(MUZK)))=0;
ln((ONE*exp(EFFECD)))-rho_EFFECD*ln((ONE*exp(EFFECD(-1))))-eEFFECD=0;
ln((ONE*exp(EFFECH)))-rho_EFFECH*ln((ONE*exp(EFFECH(-1))))-eEFFECH=0;
ln((ONE*exp(STAR)))-rho_STAR*ln((ONE*exp(STAR(-1))))-eSTAR=0;
ln((RL1SS*exp(RL1))) - ln((RSS*exp(R(+1))))=0;
ln((RL2SS*exp(RL2))) - ln((RL1SS*exp(RL1(+1))))=0;
ln((RL3SS*exp(RL3))) - ln((RL2SS*exp(RL2(+1))))=0;
```

```
ln((RL4SS*exp(RL4))) - ln((RL3SS*exp(RL3(+1))))=0;
ln((RL5SS*exp(RL5))) - ln((RL4SS*exp(RL4(+1))))=0;
ln((RL6SS*exp(RL6))) - ln((RL5SS*exp(RL5(+1))))=0;
ln((RL7SS*exp(RL7))) - ln((RL6SS*exp(RL6(+1))))=0;
ln((RT2SS*exp(RT2))) - tp2 - 0.125*(ln((RSS*exp(R))) + ln((RL1SS*exp(RL1))) + ln((RL2SS*exp(RL2

//measurement_equations;
DIFFREALGDP_obs = DIFFREALGDP + DIFFREALGDPSS_obs;
DIFFREALEC_obs = DIFFREALEC + DIFFREALECSS_obs;
DIFFREALEIK_obs = DIFFREALEIK + DIFFREALEIKSS_obs;
DIFFREALECD_obs = DIFFREALECD + DIFFREALECDSS_obs;
DIFFREALECH_obs = DIFFREALECH + DIFFREALECHSS_obs;
DIFFREALW_obs = DIFFREALW + DIFFREALWSS_obs;
AH_obs = AH;
INFCNA_obs = INFCNA + INFCNASS_obs;
INFCOR_obs = INFCOR + INFCORSS_obs;
INFK_obs = INFK + INFKSS_obs;
R_obs = R + RSS_obs;
RT2_obs = RT2 + RT2SS_obs;
unemp_obs = unemp + unempSS_obs;

//end_measurement_equations;

end;

varobs DIFFREALGDP_obs DIFFREALEC_obs DIFFREALEIK_obs DIFFREALECD_obs DIFFREALECH_obs DIFFREALW

shocks;
var eHG;
stderr sig_HG;
var eXiL;
stderr sig_XiL;
var eLpref;
stderr sig_lpref;
var eR;
stderr sig_R;
var eMUZK;
stderr sig_MUZK;
var eMUZM;
stderr sig_MUZM;
var ePMKC;
stderr sig_PMKC;
var ePMKK;
stderr sig_PMKK;
var eEFFECH;
stderr sig_EFFECH;
```

```
var eEFFECD;
stderr sig_EFFECD;
var eEFFK;
stderr sig_EFFK;
var eB;
stderr sig_B;
var eSTAR;
stderr sig_STAR;


var  DIFFREALGDP_obs;
stderr 0.3;
var  DIFFREALEC_obs;
stderr 0.1;
var  DIFFREALEIK_obs;
stderr 1.5;
var DIFFREALECD_obs;
stderr 1.5;
var DIFFREALECH_obs;
stderr 1.5;
var DIFFREALW_obs;
stderr 0.3;
var AH_obs;
stderr 0.3;
var INFCNA_obs;
stderr 0.5;
var INFCOR_obs;
stderr 0.05;
var INFK_obs;
stderr 0.2;
var RT2_obs;
stderr 0.1;
var unemp_obs;
stderr 4;
end;

steady;

estimated_params;
h              , .673         , -1             , 1    , uniform_pdf   ,,,-1
r_inf          , 1.461        , -999           , 999  , normal_pdf    , 1.5000
r_y            , 0.214        , -999           , 999  , normal_pdf    , 0.125
phi_pc         , 3.126        , 0              , 999  , gamma_pdf     , 4.0000
phi_H          , 4.064        , 0              , 999  , gamma_pdf     , 4.0000
phi_wc         , 5.119        , 0              , 999  , gamma_pdf     , 4.0000
phi_ic         , .325         , 0              , 999  , gamma_pdf     , 4.0000
```

```
phi_cd        , .651     , 0      , 999  , gamma_pdf     , 4.0000   , 4.000
phi_ech       , 10.948   , 0      , 999  , gamma_pdf     , 4.0000   , 4.000
gam_pc        , 0.386    , -999   , 999  , normal_pdf    , 0.000    , 0.250
gam_wc        , 0.213    , -999   , 999  , normal_pdf    , 0.000    , 0.250
sigman        , 1.25     , 0      , 999  , gamma_pdf     , 1.25     , 12.5^
sigmah        , 10       , 0      , 999  , gamma_pdf     , 10       , 100^.
rho_R         , 0.654    , -1     , 1    , normal_pdf    , 0.5      , 0.25;
rho_XiL       , 0.654    , -1     , 1    , normal_pdf    , 0.5      , 0.25;
rho_lpref     , 0.954    , -1     , 1    , normal_pdf    , 0.5      , 0.25;
rho_B         , 0.825    , -1     , 1    , normal_pdf    , 0        , 0.5;
rho_STAR      , 0.825    , -1     , 1    , normal_pdf    , 0        , 0.5;
rho_EFFK      , 0.850    , -1     , 1    , normal_pdf    , 0        , 0.5;
rho_EFFECD    , .230     , -1     , 1    , normal_pdf    , 0        , 0.5;
rho_HG        , 0.596    , 0      , 1    , beta_pdf      , 0.5      , 0.015
rho_EFFECH    , 0.844    , -1     , 1    , normal_pdf    , 0        , 0.5;
tp2           , 0.001    , -999   , 999  , normal_pdf    , 0.0      , 0.000

stderr eHG      , .745     , 0.0001 , 999  , inv_gamma_pdf , 1.772454 , Inf;
stderr eXiL     , 3.621    , 0.0001 , 999  , inv_gamma_pdf , 1.772454 , Inf;
stderr eLpref   , 1.621    , 0.0001 , 999  , inv_gamma_pdf , 1.772454 , Inf;
stderr eR       , 0.165    , 0.0001 , 999  , inv_gamma_pdf , 0.354491 , Inf;
stderr eMUZK    , .834     , 0.0001 , 999  , inv_gamma_pdf , 0.443113 , Inf;
stderr eMUZM    , .484     , 0.0001 , 999  , inv_gamma_pdf , 0.443113 , Inf;
stderr ePMKC    , .391     , 0.0001 , 999  , inv_gamma_pdf , 0.354491 , Inf;
stderr ePMKK    , .552     , 0.0001 , 999  , inv_gamma_pdf , 0.354491 , Inf;
stderr eEFFECH  , .526     , 0.0001 , 999  , inv_gamma_pdf , 1.772454 , Inf;
stderr eEFFECD  , 13.349   , 0.0001 , 999  , inv_gamma_pdf , 1.772454 , Inf;
stderr eEFFK    , .499     , 0.0001 , 999  , inv_gamma_pdf , 1.772454 , Inf;
stderr eB       , 0.5      , 0.0001 , 999  , inv_gamma_pdf , 1.772454 , Inf;
stderr eSTAR    , 0.05     , 0.0001 , 999  , inv_gamma_pdf , 0.354491 , Inf;
end;


options_.order = 1;
options_.jacobian_flag = 1;
options_.nonlin = 1;

stoch_simul(order=1,irf=40,nograph);
```

This code is written to file `srcedo/linearized.mod`.
Uses A_HC 14, A_HK 14, a_ks 14, AA 14, AH 11, AH_obs 11, AHSS 14, alpha_ 14, beta_ 14, beta_0 14, beta_2 14, betarl 14, betas 11, DD 14, delta_ 14, delta_cd 14, delta_ch 14, DIFFNORMGDP 11, DIFFREALEC 11, DIFFREALEC_obs 11, DIFFREALECD 11, DIFFREALECD_obs 11, DIFFREALECDSS 14, DIFFREALECDSS_obs 14, DIFFREALECH 11, DIFFREALECH_obs 11, DIFFREALECHSS 14, DIFFREALECHSS_obs 14, DIFFREALECSS 14, DIFFREALECSS_obs 14, DIFFREALEIK 11, DIFFREALEIK_obs 11, DIFFREALEIKSS 14, DIFFREALEIKSS_obs 14, DIFFREALGDP 11, DIFFREALGDP_obs 11, DIFFREALGDPSS 14, DIFFREALGDPSS_obs 14,

A.4 linearized_steadystate.m

47 ⟨*srcedo/linearized.steadystate.m* 47⟩≡

```
function [ys,check] = linearized_steadystate(ys,exe)
        global M_

check = 0;

NumberofParameters=M_.param_nbr;
for i=1:NumberofParameters
    paramname=deblank(M_.param_names(i,:));
    eval([paramname ' =M_.params(' int2str(i) ');']);
end;

%start_steady_state;

beta_0 = pbeta;
beta_2 = pbeta*rpr; % s.s. funds rate premium
beta_ = beta_2;
MUZCSS=1;
ONE=1;
USS=1;
MUKSS=MUZKSS*MUZMSS;
MUCSS=MUZKSS^alpha_*MUZMSS;
MUKSShabit=MUKSS;
MUCSShabit=MUCSS;
PKBSS=theta_k/(theta_k-1)*(theta_c-1)/theta_c;
PYSS=1;
MCCSS=(theta_c-1)/theta_c;
MCKSS=(theta_k-1)/theta_k;
RKSS=MUKSS/beta_2-(1-delta_);
RCSS=MUKSS/beta_2-(1-delta_);
RCHSS=MUCSS/beta_2-(1-delta_ch); % Housing sector
RCDSS=MUKSS/beta_2-(1-delta_cd); % Durable sector
USS=1;
mu_=RCSS;
AA=alpha_/RKSS*MCKSS;
DD = 0.135;
RR = 0.075;
eta_cnn=1;
eta_cd_eta_cnn=DD/((MUKSShabit-beta_2*h_cd)/(1-beta_2*h/MUCSShabit)*(1-h/MUCSShabit)/(1-h_cd/MU
eta_ch_eta_cnn=RR/((MUCSShabit-beta_2*h_ch)/(1-beta_2*h/MUCSShabit)*(1-h/MUCSShabit)/(1-h_ch/MU
eta_ch=eta_ch_eta_cnn;
eta_cd=eta_cd_eta_cnn;
DD=eta_cd_eta_cnn*(MUKSShabit-beta_2*h_cd)/(1-beta_2*h/MUCSShabit)*(1-h/MUCSShabit)/(1-h_cd/MUK
RR=eta_ch_eta_cnn*(MUCSShabit-beta_2*h_ch)/(1-beta_2*h/MUCSShabit)*(1-h/MUCSShabit)/(1-h_ch/MUC
```

```
Rnr=(1-(1-delta_)/MUKSS)*AA*MUKSS;
ycbi_ykb=((1-s_AS)-Rnr)/((DD*(1-s_AS)/(1+RR))+Rnr);
hc_hk=ycbi_ykb*(RCSS*MCKSS/(RKSS*MCCSS))^(alpha_/(1-alpha_));
HSS=0.25;
AHSS=HSS;
HKSS=HSS/(1+hc_hk);
HCSS=HSS-HKSS;
HrCSS=1/3;
HrKSS=1/3;
empCSS=HCSS/HrCSS;
empKSS=HKSS/HrKSS;
ycbi=HCSS*(AA)^(alpha_/(1-alpha_));
ykb=HKSS*(AA)^(alpha_/(1-alpha_));
YCSS=ycbi;
YKSS=ykb;
KCSS=AA*ycbi*MUKSS;
KKSS=AA*ykb*MUKSS;
ECHSS=RR/(1+RR)*ycbi*(1-s_AS);
ECSS=1/(1+RR)*ycbi*(1-s_AS);
ECDSS=DD*PKBSS*ECSS;
EIKSS=(1-(1-delta_)/MUKSS)*(KCSS+KKSS);
KCDSS=ECDSS/(1-(1-delta_cd)/MUKSS);
KCHSS=ECHSS/(1-(1-delta_ch)/MUCSS);
YYSS=(YCSS+YKSS*PKBSS)/PYSS;
s_k_ecd=ECDSS/YKSS;
s_c_ech=ECHSS/YCSS;
s_k_eik=EIKSS/YKSS;
s_yc = (YCSS/YYSS);
s_ecdc=PKBSS*ECDSS/(ECSS+PKBSS*ECDSS+(MUCSS/beta_2-1+delta_ch)*KCHSS);
INFCNASS=exp(.02/4);
INFCSS = INFCNASS*((MUZCSS/MUZKSS)^(1-alpha_))^(-s_ecdc);
INFCORSS=INFCNASS;
INFKSS=INFCSS*(MUZCSS/MUZKSS)^(1-alpha_);
INFWCSS=INFCSS*MUZKSS^alpha_*MUZMSS;
INFWKSS=INFWCSS;
RSS=INFCSS/beta_0*MUCSS;
RT2SS=exp(tp2)*RSS;
INFC10SS = INFCNASS;
IMPHSSS = RCHSS*KCHSS;
s_k=PKBSS*YKSS/YYSS;
INFGDPSS=INFCSS^(YCSS/YYSS)*INFKSS^(YKSS*PKBSS/(YYSS));
LSS=eta_cnn/(ECSS*(1-h/MUCSShabit))-eta_cnn*beta_2*h/(ECSS*(MUCSShabit-h));
WCSS=MCCSS*(1-alpha_)*YCSS/HCSS;
WKSS=MCKSS*(1-alpha_)*YKSS/HKSS;
xsiN_xsiH_C = ((HrCSS/empCSS)^(1+sigmah))/(1+1/sigmah);
xsiN_xsiH_K = ((HrKSS/empKSS)^(1+sigmah))/(1+1/sigmah);
```

```
gC = (1/(1+sigman) + 1/sigmah)*(xsiN_xsiH_C*(1+sigmah)/sigmah)^(-(1+sigman)/(1+sigman+sigmah));
markup_xsiN_C = (HCSS^((1+sigmah)*(1+sigman)/(1+sigmah+sigman)-1))*gC/(LSS*WCSS);
gK = (1/(1+sigman) + 1/sigmah)*(xsiN_xsiH_K*(1+sigmah)/sigmah)^(-(1+sigman)/(1+sigman+sigmah));
markup_xsiN_K = (HKSS^((1+sigmah)*(1+sigman)/(1+sigmah+sigman)-1))*gK/(LSS*WKSS);
markup_w = (1-unempSS)^((1+sigmah+sigman)/(1+sigmah) - 1 - sigman);
theta_wc = markup_w/(markup_w -1); theta_wk = theta_wc;
A_HC=LSS*(theta_wc-1)/theta_wc*WCSS/(((1+sigman)/(1+sigman/(1+sigmah)))*HCSS^(-1+(1+sigman)/(1+
A_HK=LSS*(theta_wk-1)/theta_wk*WKSS/(((1+sigman)/(1+sigman/(1+sigmah)))*HKSS^(-1+(1+sigman)/(1+
xsi_NC=A_HC/((1/(1+sigman)+1/sigmah)*(HCSS^sigman/HrCSS^(1+sigman+sigmah))^((1+sigman)/(1+sigma
xsi_NK=A_HK/((1/(1+sigman)+1/sigmah)*(HKSS^sigman/HrKSS^(1+sigman+sigmah))^((1+sigman)/(1+sigma
xsi_HrC=xsi_NC*(1+sigmah)/sigmah*(HCSS^sigman/HrCSS^(1+sigman+sigmah));
xsi_HrK=xsi_NK*(1+sigmah)/sigmah*(HKSS^sigman/HrKSS^(1+sigman+sigmah));
UHCSS=A_HC*((1+sigman)/(1+sigman/(1+sigmah)))*HCSS^(-1+(1+sigman)/(1+sigman/(1+sigmah)))/LSS;
UHKSS=A_HK*((1+sigman)/(1+sigman/(1+sigmah)))*HKSS^(-1+(1+sigman)/(1+sigman/(1+sigmah)))/LSS;
HSCSS=(WCSS*LSS/(A_HC*((1+sigman)/(1+sigman/(1+sigmah)))))^(1/(-1+(1+sigman)/(1+sigman/(1+sigma
HSKSS=(WKSS*LSS/(A_HK*((1+sigman)/(1+sigman/(1+sigmah)))))^(1/(-1+(1+sigman)/(1+sigman/(1+sigma
empSCSS=((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(-1/(1+sigmah+sigman))*HSCSS^(1/(1+sigman/(1+sigmah)
empSKSS=((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(-1/(1+sigmah+sigman))*HSKSS^(1/(1+sigman/(1+sigmah)
HrSCSS=HSCSS/empSCSS;
HrSKSS=HSKSS/empSKSS;
UHSCSS=A_HC*((1+sigman)/(1+sigman/(1+sigmah)))*HSCSS^(-1+(1+sigman)/(1+sigman/(1+sigmah)))/LSS;
UHSKSS=A_HK*((1+sigman)/(1+sigman/(1+sigmah)))*HSKSS^(-1+(1+sigman)/(1+sigman/(1+sigmah)))/LSS;
unempSS=(empSCSS+empSKSS-(empCSS+empKSS))/(empSCSS+empSKSS);
QKSS=1;
QCDSS=1;
QCHSS=1;
UCSS=1;
UKSS=1;
XiBSS=1;
XiDSS=1;
XiHSS=1;
RL1SS=RSS;
RL2SS=RSS;
RL3SS=RSS;
RL4SS=RSS;
RL5SS=RSS;
RL6SS=RSS;
RL7SS=RSS;
DIFFREALECSS =exp( log(MUCSS));
DIFFREALEIKSS =exp( log(MUKSS));
DIFFREALECDSS =exp( log(MUKSS));
DIFFREALECHSS =exp( log(MUCSS));
DIFFREALWSS =exp( log(MUCSS) );
DIFFREALGDPSS =exp( (1-s_k)*log(MUCSS)+(s_k)*log(MUKSS));

%end_steady_state;
```

```
%trends;

DIFFREALGDPSS_obs=(1-s_k)*log(MUCSS)*100+(s_k)*log(MUKSS)*100;
DIFFREALECSS_obs=log(MUCSS)*100;
DIFFREALEIKSS_obs=log(MUKSS)*100;
DIFFREALECDSS_obs=log(MUKSS)*100;
DIFFREALECHSS_obs=log(MUCSS)*100;
DIFFREALWSS_obs=log(MUCSS)*100;
INFCNASS_obs=(1-s_ecdc)*log(INFCSS)*100+s_ecdc*log(INFKSS)*100;
INFCORSS_obs=(1-s_ecdc)*log(INFCSS)*100+s_ecdc*log(INFKSS)*100;
INFKSS_obs=log(INFCSS)*100-log(MUKSS)*100+log(MUCSS)*100;
RSS_obs=log(RSS)*100;
RT2SS_obs=log(RT2SS)*100;
unempSS_obs=100*log(unempSS);

%end_trends;


for i=1:NumberofParameters
    paramname=deblank(M_.param_names(i,:));
    eval(['M_.params(' int2str(i) ')=' paramname ';']);
end;

ys = [
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
```

```
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
```

```
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
DIFFREALGDPSS_obs
DIFFREALECSS_obs
DIFFREALEIKSS_obs
DIFFREALECDSS_obs
DIFFREALECHSS_obs
DIFFREALWSS_obs
0
INFCNASS_obs
INFCORSS_obs
INFKSS_obs
RSS_obs
RT2SS_obs
unempSS_obs
];
```

This code is written to file `srcedo/linearized.steadystate.m`.

Uses `A_HC` 14, `A_HK` 14, `AA` 14, `AHSS` 14, `alpha_` 14, `beta_` 14, `beta_0` 14, `beta_2` 14, `DD` 14, `delta_` 14, `delta_cd` 14, `delta_ch` 14, `DIFFREALECDSS` 14, `DIFFREALECDSS_obs` 14, `DIFFREALECHSS` 14, `DIFFREALECHSS_obs` 14, `DIFFREALECSS` 14, `DIFFREALECSS_obs` 14, `DIFFREALEIKSS` 14, `DIFFREALEIKSS_obs` 14, `DIFFREALGDPSS` 14, `DIFFREALGDPSS_obs` 14, `DIFFREALWSS` 14, `DIFFREALWSS_obs` 14, `ECDSS` 14, `ECHSS` 14, `ECSS` 14, `EIKSS` 14, `empCSS` 14, `empKSS` 14, `empSCSS` 14, `empSKSS` 14, `eta_cd` 14, `eta_cd_eta_cnn` 14, `eta_ch` 14, `eta_ch_eta_cnn` 14, `eta_cnn` 14, `h` 14, `h_cd` 14, `h_ch` 14, `hc_hk` 14, `HCSS` 14, `HKSS` 14, `HrCSS` 14, `HrKSS` 14, `HrSCSS` 14, `HrSKSS` 14, `HSCSS` 14, `HSKSS` 14, `HSS` 14, `IMPHSSS` 14, `INFC10SS` 14, `INFCNASS` 14, `INFCNASS_obs` 14, `INFCORSS` 14, `INFCORSS_obs` 14, `INFCSS` 14, `INFGDPSS` 14, `INFKSS` 14, `INFKSS_obs` 14, `INFWCSS` 14, `INFWKSS` 14, `KCDSS` 14, `KCHSS` 14, `KCSS` 14, `KKSS` 14, `LSS` 14, `MCCSS` 14, `MCKSS` 14, `mu_` 14, `MUCSS` 14, `MUCSShabit` 14, `MUKSS` 14, `MUKSShabit` 14, `MUZCSS` 14, `MUZKSS` 14, `MUZMSS` 14, `ONE` 14, `pbeta` 14, `PKBSS` 14, `PYSS` 14, `QCDSS` 14, `QCHSS` 14, `QKSS` 14, `RCDSS` 14, `RCHSS` 14, `RCSS` 14, `RKSS` 14, `RL1SS` 14, `RL2SS` 14, `RL3SS` 14, `RL4SS` 14, `RL5SS` 14, `RL6SS` 14, `RL7SS` 14, `Rnr` 14, `rpr` 14, `RR` 14, `RSS` 14, `RSS_obs` 14, `RT2SS` 14, `RT2SS_obs` 14, `s_AS` 14, `s_c_ech` 14, `s_ecdc` 14, `s_k` 14, `s_k_ecd` 14, `s_k_eik` 14, `s_yc` 14, `sigmah` 14, `sigman` 14, `theta_c` 14, `theta_k` 14, `theta_wc` 14, `theta_wk` 14, `tp2` 14, `UCSS` 14, `UHCSS` 14, `UHKSS` 14, `UHSCSS` 14, `UHSKSS` 14, `UKSS` 14, `unempSS` 14, `unempSS_obs` 14, `USS` 14, WCSS 14, WKSS 14, xsi_HrC 14, xsi_HrK 14, xsi_NC 14, xsi_NK 14, ycbi 14, ycbi_ykb 14, YCSS 14, ykb 14, YKSS 14, and YYSS 14.

A.5 readme.txt

53 ⟨*srcedo/readme.txt* 53⟩≡

```

How to run the model:
====================

In Matlab/Octave:

1) Download Dynare Version 4 from the Dynare website: http://www.dynare.org/
2) Download the EDO files in a folder you choose.
3) Start Matlab/Octave and change the current directory to the folder in step 2.
4) Link in Matlab/Octave the Dynare folder in the menu under file/Set Path (or use the command
   "addpath path/to/dynare").
5) Run the command "dynare linearized" or "dynare Dynare_edo" from the Matlab/Octave command li


Content of the EDO folder:
========================

Dynare_edo.mod: Dynare model file containing the latest estimated parameters and nonlinear mode

Dynare_edo_steadystate.mod: Dynare steady-state file computes the steady state of the model var

linearized.mod: Dynare model file containing the latest estimated parameters and nonlinear mode

linearized_steadystate.mod: Dynare steady-state file computes the steady state of the model var

readme.txt: The file you are currently reading.
```
This code is written to file srcedo/readme.txt.

Appendix B

Notes, Bibliography and Indexes

B.1 Chunks

B.2 Index