Estimated Dynamic Optimization (EDO) Model

Gary Young (editor)

June 27, 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⟩≡
  ⟨*common setup* 30⟩
  ⟨*edo model* 10⟩
  ⟨*common stoch sim* 31⟩

This code is written to file `srcedo/Dynare.edo.mod`.

A.1.1 Dynare EDO Model

10 ⟨*edo model* 10⟩≡ (9)

```
⟨edo model prelim 11⟩

// XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
// labor block
// TOTAL LABOR INPUT (called "L" in the paper, I kept the "H" notation of the origina
⟨edo model labor 12a⟩

// Identities
⟨edo model identities 12b⟩


// XXXXXXXXXXXXXXXXXXXXXX
// Aggregate hours equals agg hours in each sector
⟨edo model hours 13a⟩

// See Section 8: Data Identities

// new equations
// Durable Block

⟨edo model durables 13b⟩

// Housing Block
⟨edo model housing 14⟩

//measurement_equations;
⟨edo model measurement 15⟩
//end_measurement_equations;
end;
```

Uses L 32.

A.1.2 Dynare EDO Prelim

11 ⟨*edo model prelim* 11⟩≡ (10)

```
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)*INFC*YC+beta_
MCK*YK*theta_k/PKB-(theta_k-1)*YK-100*phi_pc*(INFK-gam_pc*INFK(-1)-(1-gam_pc)*INFKSS)*INFK*YK+b
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)-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/phi_u))*KK
```

Uses alpha_ 35, beta_ 35, betas 32, delta_ 35, EC 32, EFFK 32, EIK 32, ePMKC 34, ePMKK 34, eR 34, eta_cnn 35, gam_pc 35, h 35, HC 32, HK 32, INFC 32, INFCNA 32, INFCNASS 35, INFCSS 35, INFK 32, INFKSS 35, KC 32, KCSS 35, KK 32, KKSS 35, L 32, MCC 32, MCK 32, mu_ 35, MUC 32, MUK 32, PFGAP 32, phi_pc 35, phi_u 35, PKB 32, QK 32, R 32, r_dinf 35, r_inf 35, r_y 35, RC 32, rho_R 35, RK 32, rpr 35, RSS 35, theta_c 35, theta_k 35, UC 32, UK 32, WC 32, WK 32, YC 32, YCSS 35, YK 32, and YKSS 35.

A.1.3 Dynare EDO Labor

12a ⟨*edo model labor* 12a⟩≡ (10)

```
-100+UHC*theta_wc-(theta_wc-1)*WC-100*phi_wc*(INFWC-gam_wc*INFWC(-1)-(1-gam_wc)*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)*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+
UHSC*L*Lpref-A_HC*((1+sigman)/(1+sigman/(1+sigmah)))*(HSC)^(-1+(1+sigman)/(1+sigman/(
UHK*L*Lpref-A_HK*((1+sigman)/(1+sigman/(1+sigmah)))*(HK)^(-1+(1+sigman)/(1+sigman/(1+
UHSK*L*Lpref-A_HK*((1+sigman)/(1+sigman/(1+sigmah)))*(HSK)^(-1+(1+sigman)/(1+sigman/(
empC-((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(-1/(1+sigmah+sigman))*HC^(1/(1+sigman/(1+sig
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+sig
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+s
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+s
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_
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 35, A_HK 35, beta_ 35, DIFFNORMGDP 32, DIFFREALEC 32, DIFFREALEIK 32, DIFFREALGDP 32, EC 32, ECD 32, ECH 32, EFFK 32, EIK 32, EIKSS 35, empC 32, empK 32, empSC 32, empSK 32, eXiL 34, gam_h 35, gam_ic 35, gam_wc 35, HC 32, HCSS 35, HG 32, HK 32, HKSS 35, HrC 32, HrK 32, HrSC 32, HrSK 32, HSC 32, HSCSS 35, HSK 32, HSKSS 35, INFC 32, INFK 32, INFWC 32, INFWCSS 35, INFWK 32, INFWKSS 35, KC 32, KK 32, L 32, Lpref 32, MUC 32, MUK 32, NORMINFGDP 32, phi_H 35, phi_ic 35, phi_wc 35, PKB 32, QK 32, s_k 35, sigmah 35, sigman 35, theta_wc 35, theta_wk 35, UHC 32, UHK 32, UHSC 32, UHSK 32, unemp 32, WC 32, WK 32, XiL 32, xsi_HrC 35, xsi_HrK 35, xsi_NC 35, xsi_NK 35, YC 32, YCSS 35, YK 32, and YKSS 35.

A.1.4 Dynare EDO Identities

12b ⟨*edo model identities* 12b⟩≡ (10)

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

Uses AHSS 35, DIFFREALW 32, HCSS 35, HKSS 35, INFC 32, INFWC 32, and INFWK 32.

A.1.5 Dynare EDO Hours

13a ⟨*edo model hours* 13a⟩≡ (10)

```
AH-HC-HK=0;
ln(INFGDP)-ln(INFC)-ln(YC*MUC/YC(-1))+ln(DIFFREALGDP)-ln((1+PKB*YK/YC)/(1+PKB(-1)*YK(-1)/YC(-1)
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)+s_k*ln(UK
ln(INFC10)-betarl*ln(INFC10(+1))-(1-betarl)*ln(INFCOR)=0;
```

Uses AH 32, alpha_ 35, betarl 35, DIFFREALGDP 32, GAP 32, HC 32, HCSS 35, HK 32, HKSS 35, INFC 32, INFC10 32, INFCNA 32, INFCOR 32, INFGDP 32, INFK 32, MUC 32, PFGAP 32, PKB 32, s_ecdc 35, s_k 35, UC 32, UK 32, USS 35, YC 32, YCSS 35, YK 32, and YKSS 35.

A.1.6 Dynare EDO Durables

13b ⟨*edo model durables* 13b⟩≡ (10)

```
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(-1)/MUK)=0
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/EFFECD)*10
```

Uses beta_ 35, delta_cd 35, ECD 32, ECDSS 35, EFFECD 32, eta_cd 35, gam_icd 35, h_cd 35, KD 32, L 32, LAGKD 32, MUK 32, phi_cd 35, PKB 32, QCD 32, and RCD 32.

A.1.7 Dynare EDO Housing

14 ⟨*edo model housing* 14⟩≡ (10)

```
L*RCH-eta_ch/(KCH(-1)/MUC-h_ch*LAGKCH(-1)/(MUC*MUC(-1)))+beta_*eta_ch*h_ch/(KCH-h_ch
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/
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(
```

Uses alpha_ 35, beta_ 35, betas 32, delta_ch 35, DIFFREALECD 32, DIFFREALECH 32, eB 34, ECD 32, ECH 32, ECHSS 35, eEFFECD 34, eEFFECH 34, eEFFK 34, EFFECD 32, EFFECH 32, EFFK 32, eHG 34, eLpref 34, eMUZK 34, eMUZM 34, eSTAR 34, eta_ch 35, eXiL 34, gam_ech 35, h_ch 35, HG 32, KCH 32, KD 32, L 32, LAGKCH 32, LAGKD 32, Lpref 32, mu_ 35, MUC 32, MUK 32, MUZK 32, MUZKSS 35, MUZM 32, MUZMSS 35, phi_ech 35, phi_u 35, QCH 32, QK 32, R 32, RC 32, RCH 32, rho_B 35, rho_EFFECD 35, rho_EFFECH 35, rho_EFFK 35, rho_HG 35, rho_lpref 35, rho_STAR 35, rho_XiL 35, RK 32, RL1 32, RL2 32, RL3 32, RL4 32, RL5 32, RL6 32, RL7 32, RT2 32, STAR 32, tp2 35, UC 32, UK 32, and XiL 32.

A.1.8 Dynare EDO Model Measurement

15 ⟨*edo model measurement* 15⟩≡ (10)

```
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 32, AH_obs 32, AHSS 35, DIFFREALEC 32, DIFFREALEC_obs 32, DIFFREALECD 32, DIFFREALECD_obs 32, DIFFREALECDSS 35, DIFFREALECDSS_obs 35, DIFFREALECH 32, DIFFREALECH_obs 32, DIFFREALECHSS 35, DIFFREALECHSS_obs 35, DIFFREALECSS 35, DIFFREALECSS_obs 35, DIFFREALEIK 32, DIFFREALEIK_obs 32, DIFFREALEIKSS 35, DIFFREALEIKSS_obs 35, DIFFREALGDP 32, DIFFREALGDP_obs 32, DIFFREALGDPSS 35, DIFFREALGDPSS_obs 35, DIFFREALW 32, DIFFREALW_obs 32, DIFFREALWSS 35, DIFFREALWSS_obs 35, INFCNA 32, INFCNA_obs 32, INFCNASS 35, INFCNASS_obs 35, INFCOR 32, INFCOR_obs 32, INFCORSS 35, INFCORSS_obs 35, INFK 32, INFK_obs 32, INFKSS 35, INFKSS_obs 35, R 32, R_obs 32, RSS 35, RSS_obs 35, RT2 32, RT2_obs 32, RT2SS 35, RT2SS_obs 35, unemp 32, unemp_obs 32, unempSS 35, and unempSS_obs 35.

A.2 Dynare_edo_steadystate.m

16 ⟨*srcedo/Dynare.edo.steadystate.m* 16⟩≡

```
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;

⟨common steady state values 45⟩

%end_steady_state;

%trends;

⟨common steady state trends 48⟩

%end_trends;


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


⟨edo steady state result return 17⟩
```

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

A.2.1 EDO Steady State Result Return

17 ⟨*edo steady state result return* 17⟩≡ (16)

```
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 35, beta_ 35, DIFFREALECDSS 35, DIFFREALECDSS_obs 35, DIFFREALECHSS 35, DIFFREALECHSS_obs 35, DIFFREALECSS 35, DIFFREALECSS_obs 35, DIFFREALEIKSS 35, DIFFREALEIKSS_obs 35, DIFFREALGDPSS 35, DIFFREALGDPSS_obs 35, DIFFREALWSS 35, DIFFREALWSS_obs 35, ECDSS 35, ECHSS 35, ECSS 35, EIKSS 35, empCSS 35, empKSS 35, empSCSS 35, empSKSS 35, HCSS 35, HKSS 35, HrCSS 35, HrKSS 35, HrSCSS 35, HrSKSS 35, HSCSS 35, HSKSS 35, INFC10SS 35, INFCNASS 35, INFCNASS_obs 35, INFCORSS 35, INFCORSS_obs 35, INFCSS 35, INFGDPSS 35, INFKSS 35, INFKSS_obs 35, INFWCSS 35, INFWKSS 35, KCDSS 35, KCHSS 35, KCSS 35, KKSS 35, LSS 35, MCCSS 35, MCKSS 35, MUCSS 35, MUKSS 35, MUZKSS 35, MUZMSS 35, ONE 35, PKBSS 35, QCDSS 35, QCHSS 35, QKSS 35, RCDSS 35, RCHSS 35, RCSS 35, RKSS 35, RL1SS 35, RL2SS 35, RL3SS 35, RL4SS 35, RL5SS 35, RL6SS 35, RL7SS 35, RSS 35, RSS_obs 35, RT2SS 35, RT2SS_obs 35, UHCSS 35, UHKSS 35, UHSCSS 35, UHSKSS 35, unempSS 35, unempSS_obs 35, USS 35, WCSS 35, WKSS 35, YCSS 35, and YKSS 35.

A.3 linearized.mod

19 ⟨*srcedo/linearized.mod* 19⟩≡
  ⟨*common setup* 30⟩
  ⟨*linearized model* 20⟩
  ⟨*common stoch sim* 31⟩

This code is written to file srcedo/linearized.mod.

A.3.1 Linearized Calibrated Measured Equations

20 ⟨*linearized model* 20⟩≡ (19)

```
  ⟨linearized model prelim 21⟩

  // XXXXXXXXXXXXXXXXXXXXXXXXXXXX
  // labor block
  // TOTAL LABOR INPUT (called "(LSS*exp(L))" in the paper, I kept the "H" notation of
  ⟨linearized model labor 22⟩

  // Identities
  ⟨linearized model identities 23a⟩


  // XXXXXXXXXXXXXXXXXXXXX
  // Aggregate hours equals agg hours in each sector
  ⟨linearized model hours 23b⟩

  // See Section 8: Data Identities

  // new equations
  // Durable Block

  ⟨linearized model durables 23c⟩

  // Housing Block
  ⟨linearized model housing 24⟩

  //measurement_equations;
  ⟨linearized model measurement 25⟩

  //end_measurement_equations;

  end;
```

Uses `L` 32 and `LSS` 35.

A.3.2 Linearized Prelim

21 ⟨*linearized model prelim* 21⟩≡ (20)

```
(RCSS*exp(RC))-(MCCSS*exp(MCC))*(YCSS*exp(YC))/(USS*exp(UC))/(KCSS*exp(KC(-1)))*alpha_*(MUKSS*e
(RKSS*exp(RK))-(MCKSS*exp(MCK))*(YKSS*exp(YK))/(USS*exp(UK))/(KKSS*exp(KK(-1)))*alpha_*(MUKSS*e
(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(HC)))^(1-a
(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
```

Uses alpha_ 35, beta_ 35, betas 32, delta_ 35, EC 32, ECSS 35, EFFK 32, EIK 32, EIKSS 35, ePMKC 34, ePMKK 34, eR 34, eta_cnn 35, gam_pc 35, h 35, HC 32, HCSS 35, HK 32, HKSS 35, INFC 32, INFCNA 32, INFCNASS 35, INFCSS 35, INFK 32, INFKSS 35, KC 32, KCSS 35, KK 32, KKSS 35, L 32, LSS 35, MCC 32, MCCSS 35, MCK 32, MCKSS 35, mu_ 35, MUC 32, MUCSS 35, MUK 32, MUKSS 35, ONE 35, PFGAP 32, phi_pc 35, phi_u 35, PKB 32, PKBSS 35, QK 32, QKSS 35, R 32, r_dinf 35, r_inf 35, r_y 35, RC 32, RCSS 35, rho_R 35, RK 32, RKSS 35, rpr 35, RSS 35, theta_c 35, theta_k 35, UC 32, UK 32, USS 35, WC 32, WCSS 35, WK 32, WKSS 35, YC 32, YCSS 35, YK 32, and YKSS 35.

A.3.3 Linearized Calibrated ME Labor BLock

22 ⟨*linearized model labor* 22⟩≡ (20)

```
-100+(UHCSS*exp(UHC))*theta_wc-(theta_wc-1)*(WCSS*exp(WC))-100*phi_wc*((INFWCSS*exp(I
(UHSCSS*exp(UHSC))-(WCSS*exp(WC))+phi_H/10*((HSCSS*exp(HSC))/(HSKSS*exp(HSK))-gam_h*(
-100+(UHKSS*exp(UHK))*theta_wk-(theta_wk-1)*(WKSS*exp(WK))-100*phi_wc*((INFWKSS*exp(I
(UHSKSS*exp(UHSK))-(WKSS*exp(WK))-phi_H/10*((HSCSS*exp(HSC))/(HSKSS*exp(HSK))-gam_h*(
(UHCSS*exp(UHC))*(LSS*exp(L))*(ONE*exp(Lpref))-A_HC*((1+sigman)/(1+sigman/(1+sigmah))
(UHSCSS*exp(UHSC))*(LSS*exp(L))*(ONE*exp(Lpref))-A_HC*((1+sigman)/(1+sigman/(1+sigmah
(UHKSS*exp(UHK))*(LSS*exp(L))*(ONE*exp(Lpref))-A_HK*((1+sigman)/(1+sigman/(1+sigmah))
(UHSKSS*exp(UHSK))*(LSS*exp(L))*(ONE*exp(Lpref))-A_HK*((1+sigman)/(1+sigman/(1+sigmah
(empCSS*exp(empC))-((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(-1/(1+sigmah+sigman))*(HCSS*ex
(HrCSS*exp(HrC))-((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(1/(1+sigmah))*(empCSS*exp(empC))
(empKSS*exp(empK))-((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(-1/(1+sigmah+sigman))*(HKSS*ex
(HrKSS*exp(HrK))-((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(1/(1+sigmah))*(empKSS*exp(empK))
(empSCSS*exp(empSC))-((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(-1/(1+sigmah+sigman))*(HSCSS
(HrSCSS*exp(HrSC))-((1+sigmah)/sigmah*xsi_NC/xsi_HrC)^(1/(1+sigmah))*(empSCSS*exp(emp
(empSKSS*exp(empSK))-((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(-1/(1+sigmah+sigman))*(HSKSS
(HrSKSS*exp(HrSK))-((1+sigmah)/sigmah*xsi_NK/xsi_HrK)^(1/(1+sigmah))*(empSKSS*exp(emp
(unempSS*exp(unemp))-((empSCSS*exp(empSC))+(empSKSS*exp(empSK))-((empCSS*exp(empC))+(
(PKBSS*exp(PKB))-(1-100*phi_ic*((EIKSS*exp(EIK))-gam_ic*(EIKSS*exp(EIK(-1)))-(1-gam_i
(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((INFWKSS*exp(INFWK)))-ln((WKSS*exp(WK)))+ln((WKSS*exp(WK(-1))))-ln((MUCSS*exp(MUC)
ln((INFKSS*exp(INFK)))-ln((INFCSS*exp(INFC)))-ln((PKBSS*exp(PKB)))+ln((PKBSS*exp(PKB(
(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*(l
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
ln((DIFFREALECSS*exp(DIFFREALEC)))-ln((ECSS*exp(EC)))+ln((ECSS*exp(EC(-1))))-ln((MUCS
ln((DIFFREALEIKSS*exp(DIFFREALEIK)))-ln((EIKSS*exp(EIK)))+ln((EIKSS*exp(EIK(-1))))-ln
```

Uses A_HC 35, A_HK 35, beta_ 35, DIFFNORMGDP 32, DIFFREALEC 32, DIFFREALECSS 35, DIFFREALEIK 32, DIFFREALEIKSS 35, DIFFREALGDP 32, DIFFREALGDPSS 35, EC 32, ECD 32, ECDSS 35, ECH 32, ECHSS 35, ECSS 35, EFFK 32, EIK 32, EIKSS 35, empC 32, empCSS 35, empK 32, empKSS 35, empSC 32, empSCSS 35, empSK 32, empSKSS 35, eXiL 34, gam_h 35, gam_ic 35, gam_wc 35, HC 32, HCSS 35, HG 32, HK 32, HKSS 35, HrC 32, HrCSS 35, HrK 32, HrKSS 35, HrSC 32, HrSCSS 35, HrSK 32, HrSKSS 35, HSC 32, HSCSS 35, HSK 32, HSKSS 35, INFC 32, INFCSS 35, INFK 32, INFKSS 35, INFWC 32, INFWCSS 35, INFWK 32, INFWKSS 35, KC 32, KCSS 35, KK 32, KKSS 35, L 32, Lpref 32, LSS 35, MUC 32, MUCSS 35, MUK 32, MUKSS 35, NORMINFGDP 32, ONE 35, phi_H 35, phi_ic 35, phi_wc 35, PKB 32, PKBSS 35, QK 32, QKSS 35, s_k 35, sigmah 35, sigman 35, theta_wc 35, theta_wk 35, UHC 32, UHCSS 35, UHK 32, UHKSS 35, UHSC 32, UHSCSS 35, UHSK 32, UHSKSS 35, unemp 32, unempSS 35, WC 32, WCSS 35, WK 32, WKSS 35, XiL 32, xsi_HrC 35, xsi_HrK 35, xsi_NC 35, xsi_NK 35, YC 32, YCSS 35, YK 32, and YKSS 35.

A.3.4 Linearized Calibrated ME Identities

23a ⟨*linearized model identities* 23a⟩≡ (20)

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

Uses AHSS 35, DIFFREALW 32, DIFFREALWSS 35, HCSS 35, HKSS 35, INFC 32, INFCSS 35, INFWC 32, INFWCSS 35, INFWK 32, and INFWKSS 35.

A.3.5 Linearized Model Hours

23b ⟨*linearized model hours* 23b⟩≡ (20)

```
(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))/(YCSS*exp(
ln((INFCNASS*exp(INFCNA)))-(1-s_ecdc)*ln((INFCSS*exp(INFC)))-s_ecdc*ln((INFKSS*exp(INFK)))=0;
ln((INFCORSS*exp(INFCOR)))-(1-s_ecdc)*ln((INFCSS*exp(INFC)))-s_ecdc*ln((INFKSS*exp(INFK)))=0;
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))/HKSS))-a
ln((INFC10SS*exp(INFC10)))-betarl*ln((INFC10SS*exp(INFC10(+1))))-(1-betarl)*ln((INFCORSS*exp(IN
```

Uses AH 32, AHSS 35, alpha_ 35, betarl 35, DIFFREALGDP 32, DIFFREALGDPSS 35, GAP 32, HC 32, HCSS 35, HK 32, HKSS 35, INFC 32, INFC10 32, INFC10SS 35, INFCNA 32, INFCNASS 35, INFCOR 32, INFCORSS 35, INFCSS 35, INFGDP 32, INFGDPSS 35, INFK 32, INFKSS 35, MUC 32, MUCSS 35, ONE 35, PFGAP 32, PKB 32, PKBSS 35, s_ecdc 35, s_k 35, UC 32, UK 32, USS 35, YC 32, YCSS 35, YK 32, and YKSS 35.

A.3.6 Linearized durables

23c ⟨*linearized model durables* 23c⟩≡ (20)

```
(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*(KCDSS*exp(LAGK
(QCDSS*exp(QCD))-beta_*(1/(ONE*exp(EFFECD)))*(LSS*exp(L(+1)))/(LSS*exp(L))/(MUKSS*exp(MUK(+1)))
(PKBSS*exp(PKB))-(QCDSS*exp(QCD))*(1-100*phi_cd*((ECDSS*exp(ECD))-gam_icd*(ECDSS*exp(ECD(-1)))-
```

Uses beta_ 35, delta_cd 35, ECD 32, ECDSS 35, EFFECD 32, eta_cd 35, gam_icd 35, h_cd 35, KCDSS 35, KD 32, L 32, LAGKD 32, LSS 35, MUK 32, MUKSS 35, ONE 35, phi_cd 35, PKB 32, PKBSS 35, QCD 32, QCDSS 35, RCD 32, and RCDSS 35.

A.3.7 Linearized Calibrated ME Housing

24 ⟨*linearized model housing* 24⟩≡ (20)

```
(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((RL2
```

Uses alpha_ 35, beta_ 35, betas 32, delta_ch 35, DIFFREALECD 32, DIFFREALECDSS 35, DIFFREALECH 32, DIFFREALECHSS 35, eB 34, ECD 32, ECDSS 35, ECH 32, ECHSS 35, eEFFECD 34, eEFFECH 34, eEFFK 34, EFFECD 32, EFFECH 32, EFFK 32, eHG 34, eLpref 34, eMUZK 34, eMUZM 34, eSTAR 34, eta_ch 35, eXiL 34, gam_ech 35, h_ch 35, HG 32, KCDSS 35, KCH 32, KCHSS 35, KD 32, L 32, LAGKCH 32, LAGKD 32, Lpref 32, LSS 35, mu_ 35, MUC 32, MUCSS 35, MUK 32, MUKSS 35, MUZK 32, MUZKSS 35, MUZM 32, MUZMSS 35, ONE 35, phi_ech 35, phi_u 35, QCH 32, QCHSS 35, QK 32, QKSS 35, R 32, RC 32, RCH 32, RCHSS 35, RCSS 35, rho_B 35, rho_EFFECD 35, rho_EFFECH 35, rho_EFFK 35, rho_HG 35, rho_lpref 35, rho_STAR 35, rho_XiL 35, RK 32, RKSS 35, RL1 32, RL1SS 35, RL2 32, RL2SS 35, RL3 32, RL3SS 35, RL4 32, RL4SS 35, RL5 32, RL5SS 35, RL6 32, RL6SS 35, RL7 32, RL7SS 35, RSS 35, RT2 32, RT2SS 35, STAR 32, tp2 35, UC 32, UK 32, USS 35, and XiL 32.

A.3.8 Linearized Calibrated ME Measurement

25 ⟨*linearized model measurement* 25⟩≡ (20)

```
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;
```

Uses AH 32, AH_obs 32, DIFFREALEC 32, DIFFREALEC_obs 32, DIFFREALECD 32, DIFFREALECD_obs 32, DIFFREALECDSS_obs 35, DIFFREALECH 32, DIFFREALECH_obs 32, DIFFREALECHSS_obs 35, DIFFREALECSS_obs 35, DIFFREALEIK 32, DIFFREALEIK_obs 32, DIFFREALEIKSS_obs 35, DIFFREALGDP 32, DIFFREALGDP_obs 32, DIFFREALGDPSS_obs 35, DIFFREALW 32, DIFFREALW_obs 32, DIFFREALWSS_obs 35, INFCNA 32, INFCNA_obs 32, INFCNASS_obs 35, INFCOR 32, INFCOR_obs 32, INFCORSS_obs 35, INFK 32, INFK_obs 32, INFKSS_obs 35, R 32, R_obs 32, RSS_obs 35, RT2 32, RT2_obs 32, RT2SS_obs 35, unemp 32, unemp_obs 32, and unempSS_obs 35.

A.4 linearized_steadystate.m

26 ⟨*srcedo/linearized.steadystate.m* 26⟩≡

```
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;

⟨common steady state values 45⟩

%end_steady_state;

%trends;

⟨common steady state trends 48⟩

%end_trends;


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

⟨linearized steady state result return 27⟩
```

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

A.4.1 Linearized Steady State Result Return

27 ⟨*linearized steady state result return* 27⟩≡ (26)

```
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
  ];
```

Uses DIFFREALECDSS_obs 35, DIFFREALECHSS_obs 35, DIFFREALECSS_obs 35, DIFFREALEIKSS_obs 35, DIFFREALGDPSS_obs 35, DIFFREALWSS_obs 35, INFCNASS_obs 35, INFCORSS_obs 35, INFKSS_obs 35, RSS_obs 35, RT2SS_obs 35, and unempSS_obs 35.

A.5 Common Model Routines

A.5.1 Common Model Setup

30 ⟨*common setup* 30⟩≡ (9 19)

```
⟨common var 32⟩


⟨common varexo 34⟩


⟨common parameters 35⟩


//estimated_params;
⟨common estimated params 39⟩
//end_estimated_params;

//calibrated_params;
⟨common calibrated params 40⟩
//end_calibrated_params;

//free_params;
⟨common free params 41a⟩
//end_free_params;

//calibrated ME


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

model;
```

A.5.2 Common Stochastic Simulation

31 ⟨*common stoch sim* 31⟩≡ (9 19)

```
⟨common varobs 41b⟩

shocks;
⟨common shocks 42⟩
end;

steady;

estimated_params;
⟨common steady estimated params 43⟩

⟨common stderr 44⟩
end;


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

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

Defines:
- `jacobian_flag`, never used.
- `nonlin`, never used.
- `options_`, never used.
- `order`, never used.
- `stoch_simul`, never used.

A.5.3 Common Var

32 ⟨*common var* 32⟩≡ (30)
```
var RC RK WC WK YC YK MCC MCK KC  KK PKB R L QK HC  HSC  HK HSK  UHC UHSC UHK UHSK em
  DIFFREALGDP_obs DIFFREALEC_obs DIFFREALEIK_obs DIFFREALECD_obs DIFFREALECH_obs DIFFR
```
Defines:

AH, used in chunks 13a, 15, 23b, and 25.
AH_obs, used in chunks 15, 25, 41b, and 42.
betas, used in chunks 11, 14, 21, and 24.
DIFFNORMGDP, used in chunks 12a and 22.
DIFFREALEC, used in chunks 12a, 15, 22, and 25.
DIFFREALEC_obs, used in chunks 15, 25, 41b, and 42.
DIFFREALECD, used in chunks 14, 15, 24, and 25.
DIFFREALECD_obs, used in chunks 15, 25, 41b, and 42.
DIFFREALECH, used in chunks 14, 15, 24, and 25.
DIFFREALECH_obs, used in chunks 15, 25, 41b, and 42.
DIFFREALEIK, used in chunks 12a, 15, 22, and 25.
DIFFREALEIK_obs, used in chunks 15, 25, 41b, and 42.
DIFFREALGDP, used in chunks 12a, 13a, 15, 22, 23b, and 25.
DIFFREALGDP_obs, used in chunks 15, 25, 41b, and 42.
DIFFREALW, used in chunks 12b, 15, 23a, and 25.
DIFFREALW_obs, used in chunks 15, 25, 41b, and 42.
EC, used in chunks 11, 12a, 21, and 22.
ECD, used in chunks 12–14 and 22–24.
ECH, used in chunks 12a, 14, 22, and 24.
EFFECD, used in chunks 13b, 14, 23c, and 24.
EFFECH, used in chunks 14 and 24.
EFFK, used in chunks 11, 12a, 14, 21, 22, and 24.
EIK, used in chunks 11, 12a, 21, and 22.
empC, used in chunks 12a and 22.
empK, used in chunks 12a and 22.
empSC, used in chunks 12a and 22.
empSK, used in chunks 12a and 22.
GAP, used in chunks 13a and 23b.
HC, used in chunks 11–13 and 21–23.
HG, used in chunks 12a, 14, 22, and 24.
HK, used in chunks 11–13 and 21–23.
HrC, used in chunks 12a and 22.
HrK, used in chunks 12a and 22.
HrSC, used in chunks 12a and 22.
HrSK, used in chunks 12a and 22.
HSC, used in chunks 12a and 22.
HSK, used in chunks 12a and 22.
INFC, used in chunks 11–13 and 21–23.
INFC10, used in chunks 13a and 23b.
INFCNA, used in chunks 11, 13a, 15, 21, 23b, and 25.
INFCNA_obs, used in chunks 15, 25, 41b, and 42.
INFCOR, used in chunks 13a, 15, 23b, and 25.
INFCOR_obs, used in chunks 15, 25, 41b, and 42.
INFGDP, used in chunks 13a and 23b.
INFK, used in chunks 11–13, 15, 21–23, and 25.
INFK_obs, used in chunks 15, 25, 41b, and 42.
INFWC, used in chunks 12, 22, and 23a.
INFWK, used in chunks 12, 22, and 23a.
KC, used in chunks 11, 12a, 21, and 22.
KCH, used in chunks 14 and 24.
KD, used in chunks 13b, 14, 23c, and 24.

A.5.4 Common VarExo

34 ⟨*common varexo* 34⟩≡ (30)

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

Defines:
- `eB`, used in chunks 14, 24, 42, and 44.
- `eEFFECD`, used in chunks 14, 24, 42, and 44.
- `eEFFECH`, used in chunks 14, 24, 42, and 44.
- `eEFFK`, used in chunks 14, 24, 42, and 44.
- `eHG`, used in chunks 14, 24, 42, and 44.
- `eLpref`, used in chunks 14, 24, 42, and 44.
- `eMUZK`, used in chunks 14, 24, 42, and 44.
- `eMUZM`, used in chunks 14, 24, 42, and 44.
- `ePMKC`, used in chunks 11, 21, 42, and 44.
- `ePMKK`, used in chunks 11, 21, 42, and 44.
- `eR`, used in chunks 11, 21, 42, and 44.
- `eSTAR`, used in chunks 14, 24, 42, and 44.
- `eXiL`, used in chunks 12a, 14, 22, 24, 42, and 44.

A.5.5 Common Parameters

35 ⟨*common parameters* 35⟩≡ (30)

```
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_icd rho_R r
rho_EFFECD rho_HG rho_EFFECH tp2 ONE MUZMSS MUZKSS  r_dinf rpr phi_u rho_MUZK rho_MUZM pbeta de
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_ch
icoef mu_ betarl MUZCSS RCSS RKSS WCSS WKSS YCSS YKSS MCCSS MCKSS KCSS KKSS LSS HCSS HKSS QKSS
MUCSS MUKSS AHSS ECDSS KCDSS QCDSS RCDSS ECHSS KCHSS QCHSS RCHSS UKSS UCSS USS MUKSShabit MUCSS
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_k_eik s_yc
sig_HG sig_XiL sig_lpref sig_R sig_MUZK sig_MUZM sig_PMKC sig_PMKK sig_EFFECH  sig_EFFECD sig_E
HSKSS HSCSS HrCSS HrKSS A_HC sigman sigmah A_HK xsi_NC xsi_HrC xsi_NK xsi_HrK rho_XiL rho_lpref
empCSS empKSS HrSKSS HrSCSS empSCSS empSKSS UHCSS UHKSS UHSCSS UHSKSS unempSS DIFFREALGDPSS DIF
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 12a, 22, 41a, and 45.
- A_HK, used in chunks 12a, 22, 41a, and 45.
- a_ks, used in chunk 40.
- AA, used in chunk 45.
- AHSS, used in chunks 12b, 15, 17, 23, and 45.
- alpha_, used in chunks 11, 13a, 14, 21, 23b, 24, 40, and 45.
- beta_, used in chunks 11–14, 17, 21–24, and 45.
- beta_0, used in chunk 45.
- beta_2, used in chunk 45.
- betarl, used in chunks 13a, 23b, and 40.
- DD, used in chunk 45.
- delta_, used in chunks 11, 21, 40, and 45.
- delta_cd, used in chunks 13b, 23c, 40, and 45.
- delta_ch, used in chunks 14, 24, 40, and 45.
- DIFFREALECDSS, used in chunks 15, 17, 24, and 45.
- DIFFREALECDSS_obs, used in chunks 15, 17, 25, 27, and 48.
- DIFFREALECHSS, used in chunks 15, 17, 24, and 45.
- DIFFREALECHSS_obs, used in chunks 15, 17, 25, 27, and 48.
- DIFFREALECSS, used in chunks 15, 17, 22, and 45.
- DIFFREALECSS_obs, used in chunks 15, 17, 25, 27, and 48.
- DIFFREALEIKSS, used in chunks 15, 17, 22, and 45.
- DIFFREALEIKSS_obs, used in chunks 15, 17, 25, 27, and 48.
- DIFFREALGDPSS, used in chunks 15, 17, 22, 23b, and 45.
- DIFFREALGDPSS_obs, used in chunks 15, 17, 25, 27, and 48.
- DIFFREALWSS, used in chunks 15, 17, 23a, and 45.
- DIFFREALWSS_obs, used in chunks 15, 17, 25, 27, and 48.
- ECDSS, used in chunks 13b, 17, 22–24, and 45.
- ECHSS, used in chunks 14, 17, 22, 24, and 45.
- ECSS, used in chunks 17, 21, 22, and 45.
- EIKSS, used in chunks 12a, 17, 21, 22, and 45.
- empCSS, used in chunks 17, 22, and 45.
- empKSS, used in chunks 17, 22, and 45.
- empSCSS, used in chunks 17, 22, and 45.
- empSKSS, used in chunks 17, 22, and 45.

A.5.6 Common Estimated Params

39 ⟨*common estimated params* 39⟩≡ (30)

```
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` 35, `gam_wc` 35, `h` 35, `phi_cd` 35, `phi_ech` 35, `phi_H` 35, `phi_ic` 35, `phi_pc` 35, `phi_wc` 35, `r_inf` 35, `r_y` 35, `rho_B` 35, `rho_EFFECD` 35, `rho_EFFECH` 35, `rho_EFFK` 35, `rho_HG` 35, `rho_lpref` 35, `rho_R` 35, `rho_STAR` 35, `rho_XiL` 35, `sig_B` 35, `sig_EFFECD` 35, `sig_EFFECH` 35, `sig_EFFK` 35, `sig_HG` 35, `sig_lpref` 35, `sig_MUZK` 35, `sig_MUZM` 35, `sig_PMKC` 35, `sig_PMKK` 35, `sig_R` 35, `sig_STAR` 35, `sig_XiL` 35, `sigmah` 35, `sigman` 35, and `tp2` 35.

A.5.7 Common Calibrated params

40 ⟨*common calibrated params* 40⟩≡ (30)

```
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` 35, `alpha_` 35, `betarl` 35, `delta_` 35, `delta_cd` 35, `delta_ch` 35, `g_y` 35, `gam_ech` 35, `gam_h` 35, `gam_ic` 35, `gam_icd` 35, `h_cd` 35, `h_ch` 35, `icoef` 35, `MUZKSS` 35, `MUZMSS` 35, `ONE` 35, `pbeta` 35, `phi_u` 35, `r_dinf` 35, `r_dy` 35, `rho_MUZK` 35, `rho_MUZM` 35, `rpr` 35, `s_AS` 35, `theta_c` 35, `theta_k` 35, and `unempSS` 35.

A.5.8 Common Free Params

41a ⟨*common free params* 41a⟩≡ (30)

```
//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 35, A_HK 35, eta_cd 35, eta_ch 35, eta_cnn 35, mu_ 35, s_ecdc 35, s_k 35, theta_wc 35, theta_wk 35, xsi_HrC 35, xsi_HrK 35, xsi_NC 35, and xsi_NK 35.

A.5.9 common Varobs

41b ⟨*common varobs* 41b⟩≡ (31)

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

Uses AH_obs 32, DIFFREALEC_obs 32, DIFFREALECD_obs 32, DIFFREALECH_obs 32, DIFFREALEIK_obs 32, DIFFREALGDP_obs 32, DIFFREALW_obs 32, INFCNA_obs 32, INFCOR_obs 32, INFK_obs 32, R_obs 32, RT2_obs 32, and unemp_obs 32.

A.5.10 Common Shocks

42 ⟨*common shocks* 42⟩≡ (31)

```
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 32, DIFFREALEC_obs 32, DIFFREALECD_obs 32, DIFFREALECH_obs 32, DIFFREALEIK_obs 32, DIFFREALGDP_obs 32, DIFFREALW_obs 32, eB 34, eEFFECD 34, eEFFECH 34, eEFFK 34, eHG 34, eLpref 34, eMUZK 34, eMUZM 34, ePMKC 34, ePMKK 34, eR 34, eSTAR 34, eXiL 34, INFCNA_obs 32, INFCOR_obs 32, INFK_obs 32, RT2_obs 32, sig_B 35, sig_EFFECD 35, sig_EFFECH 35, sig_EFFK 35, sig_HG 35, sig_lpref 35, sig_MUZK 35, sig_MUZM 35, sig_PMKC 35, sig_PMKK 35, sig_R 35, sig_STAR 35, sig_XiL 35, and unemp_obs 32.

A.5.11 Common Steady Estimated params

43 ⟨*common steady estimated params* 43⟩≡ (31)

```
h             , .673     , -1     , 1    , uniform_pdf   ,,,-1      ,1;
r_inf         , 1.461    , -999   , 999  , normal_pdf    , 1.5000   , 0.062
r_y           , 0.214    , -999   , 999  , normal_pdf    , 0.125    , 0.125
phi_pc        , 3.126    , 0      , 999  , gamma_pdf     , 4.0000   , 4.000
phi_H         , 4.064    , 0      , 999  , gamma_pdf     , 4.0000   , 4.000
phi_wc        , 5.119    , 0      , 999  , gamma_pdf     , 4.0000   , 4.000
phi_ic        , .325     , 0      , 999  , gamma_pdf     , 4.0000   , 4.000
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
```

Uses gam_pc 35, gam_wc 35, h 35, phi_cd 35, phi_ech 35, phi_H 35, phi_ic 35, phi_pc 35, phi_wc 35, r_inf 35, r_y 35, rho_B 35, rho_EFFECD 35, rho_EFFECH 35, rho_EFFK 35, rho_HG 35, rho_lpref 35, rho_R 35, rho_STAR 35, rho_XiL 35, sigmah 35, sigman 35, and tp2 35.

A.5.12 Common Stderr

44 ⟨*common stderr* 44⟩≡ (31)

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

Uses `eB` 34, `eEFFECD` 34, `eEFFECH` 34, `eEFFK` 34, `eHG` 34, `eLpref` 34, `eMUZK` 34, `eMUZM` 34, `ePMKC` 34, `ePMKK` 34, `eR` 34, `eSTAR` 34, and `eXiL` 34.

A.6 Common Steady State Routines

A.6.1 Common Steady State Values

45 ⟨*common steady state values* 45⟩≡ (16 26)

```
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)))/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));
```

Uses A_HC 35, A_HK 35, AA 35, AHSS 35, alpha_ 35, beta_ 35, beta_0 35, beta_2 35, DD 35, delta_ 35, delta_cd 35, delta_ch 35, DIFFREALECDSS 35, DIFFREALECHSS 35, DIFFREALECSS 35, DIFFREALEIKSS 35, DIFFREALGDPSS 35, DIFFREALWSS 35, ECDSS 35, ECHSS 35, ECSS 35, EIKSS 35, empCSS 35, empKSS 35, empSCSS 35, empSKSS 35, eta_cd 35, eta_cd_eta_cnn 35, eta_ch 35, eta_ch_eta_cnn 35, eta_cnn 35, h 35, h_cd 35, h_ch 35, hc_hk 35, HCSS 35, HKSS 35, HrCSS 35, HrKSS 35, HrSCSS 35, HrSKSS 35, HSCSS 35, HSKSS 35, HSS 35, IMPHSSS 35, INFC10SS 35, INFCNASS 35, INFCORSS 35, INFCSS 35, INFGDPSS 35, INFKSS 35, INFWCSS 35, INFWKSS 35, KCDSS 35, KCHSS 35, KCSS 35, KKSS 35, LSS 35, MCCSS 35, MCKSS 35, mu_ 35, MUCSS 35, MUCSShabit 35, MUKSS 35, MUKSShabit 35, MUZCSS 35, MUZKSS 35, MUZMSS 35, ONE 35, pbeta 35, PKBSS 35, PYSS 35, QCDSS 35, QCHSS 35, QKSS 35, RCDSS 35, RCHSS 35, RCSS 35, RKSS 35, RL1SS 35, RL2SS 35, RL3SS 35, RL4SS 35, RL5SS 35, RL6SS 35, RL7SS 35, Rnr 35, rpr 35, RR 35, RSS 35, RT2SS 35, s_AS 35, s_c_ech 35, s_ecdc 35, s_k 35, s_k_ecd 35, s_k_eik 35, s_yc 35, sigmah 35, sigman 35, theta_c 35, theta_k 35, theta_wc 35, theta_wk 35, tp2 35, UCSS 35, UHCSS 35, UHKSS 35, UHSCSS 35, UHSKSS 35, UKSS 35, unempSS 35, USS 35, WCSS 35, WKSS 35, xsi_HrC 35, xsi_HrK 35, xsi_NC 35, xsi_NK 35, ycbi 35, ycbi_ykb 35, YCSS 35, ykb 35, YKSS 35, and YYSS 35.

A.6.2 Common Steady State Trends

48 ⟨*common steady state trends* 48⟩≡ (16 26)

```
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 35, DIFFREALECHSS_obs 35, DIFFREALECSS_obs 35, DIFFREALEIKSS_obs 35, DIFFREALGDPSS_obs 35, DIFFREALWSS_obs 35, INFCNASS_obs 35, INFCORSS_obs 35, INFCSS 35, INFKSS 35, INFKSS_obs 35, MUCSS 35, MUKSS 35, RSS 35, RSS_obs 35, RT2SS 35, RT2SS_obs 35, s_ecdc 35, s_k 35, unempSS 35, and unempSS_obs 35.

A.7 readme.txt

49 ⟨*srcedo/readme.txt* 49⟩≡

```
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