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⟩≡
   ⟨*common setup* 38⟩
   ⟨*edo model* 10⟩
   ⟨*common stoch sim* 39⟩

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⟩


// XXXXXXXXXXXXXXXXXXXXX
// 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` 40.

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_ 43, beta_ 43, betas 40, delta_ 43, EC 40, EFFK 40, EIK 40, ePMKC 42, ePMKK 42, eR 42, eta_cnn 43, gam_pc 43, h 43, HC 40, HK 40, INFC 40, INFCNA 40, INFCNASS 43, INFCSS 43, INFK 40, INFKSS 43, KC 40, KCSS 43, KK 40, KKSS 43, L 40, MCC 40, MCK 40, mu_ 43, MUC 40, MUK 40, PFGAP 40, phi_pc 43, phi_u 43, PKB 40, QK 40, R 40, r_dinf 43, r_inf 43, r_y 43, RC 40, rho_R 43, RK 40, rpr 43, RSS 43, theta_c 43, theta_k 43, UC 40, UK 40, WC 40, WK 40, YC 40, YCSS 43, YK 40, and YKSS 43.

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)*INFW
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)*INFW
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+si
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+si
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+
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+
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 43, A_HK 43, beta_ 43, DIFFNORMGDP 40, DIFFREALEC 40, DIFFREALEIK 40, DIFFREALGDP 40, EC 40, ECD 40, ECH 40, EFFK 40, EIK 40, EIKSS 43, empC 40, empK 40, empSC 40, empSK 40, eXiL 42, gam_h 43, gam_ic 43, gam_wc 43, HC 40, HCSS 43, HG 40, HK 40, HKSS 43, HrC 40, HrK 40, HrSC 40, HrSK 40, HSC 40, HSCSS 43, HSK 40, HSKSS 43, INFC 40, INFK 40, INFWC 40, INFWCSS 43, INFWK 40, INFWKSS 43, KC 40, KK 40, L 40, Lpref 40, MUC 40, MUK 40, NORMINFGDP 40, phi_H 43, phi_ic 43, phi_wc 43, PKB 40, QK 40, s_k 43, sigmah 43, sigman 43, theta_wc 43, theta_wk 43, UHC 40, UHK 40, UHSC 40, UHSK 40, unemp 40, WC 40, WK 40, XiL 40, xsi_HrC 43, xsi_HrK 43, xsi_NC 43, xsi_NK 43, YC 40, YCSS 43, YK 40, and YKSS 43.

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 43, DIFFREALW 40, HCSS 43, HKSS 43, INFC 40, INFWC 40, and INFWK 40.

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 40, alpha_ 43, betarl 43, DIFFREALGDP 40, GAP 40, HC 40, HCSS 43, HK 40, HKSS 43, INFC 40, INFC10 40, INFCNA 40, INFCOR 40, INFGDP 40, INFK 40, MUC 40, PFGAP 40, PKB 40, s_ecdc 43, s_k 43, UC 40, UK 40, USS 43, YC 40, YCSS 43, YK 40, and YKSS 43.

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_ 43, delta_cd 43, ECD 40, ECDSS 43, EFFECD 40, eta_cd 43, gam_icd 43, h_cd 43, KD 40, L 40, LAGKD 40, MUK 40, phi_cd 43, PKB 40, QCD 40, and RCD 40.

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(R
```

Uses alpha_ 43, beta_ 43, betas 40, delta_ch 43, DIFFREALECD 40, DIFFREALECH 40, eB 42, ECD 40, ECH 40, ECHSS 43, eEFFECD 42, eEFFECH 42, eEFFK 42, EFFECD 40, EFFECH 40, EFFK 40, eHG 42, eLpref 42, eMUZK 42, eMUZM 42, eSTAR 42, eta_ch 43, eXiL 42, gam_ech 43, h_ch 43, HG 40, KCH 40, KD 40, L 40, LAGKCH 40, LAGKD 40, Lpref 40, mu_ 43, MUC 40, MUK 40, MUZK 40, MUZKSS 43, MUZM 40, MUZMSS 43, phi_ech 43, phi_u 43, QCH 40, QK 40, R 40, RC 40, RCH 40, rho_B 43, rho_EFFECD 43, rho_EFFECH 43, rho_EFFK 43, rho_HG 43, rho_lpref 43, rho_STAR 43, rho_XiL 43, RK 40, RL1 40, RL2 40, RL3 40, RL4 40, RL5 40, RL6 40, RL7 40, RT2 40, STAR 40, tp2 43, UC 40, UK 40, and XiL 40.

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 40, AH_obs 40, AHSS 43, DIFFREALEC 40, DIFFREALEC_obs 40, DIFFREALECD 40, DIFFREALECD_obs 40, DIFFREALECDSS 43, DIFFREALECDSS_obs 43, DIFFREALECH 40, DIFFREALECH_obs 40, DIFFREALECHSS 43, DIFFREALECHSS_obs 43, DIFFREALECSS 43, DIFFREALECSS_obs 43, DIFFREALEIK 40, DIFFREALEIK_obs 40, DIFFREALEIKSS 43, DIFFREALEIKSS_obs 43, DIFFREALGDP 40, DIFFREALGDP_obs 40, DIFFREALGDPSS 43, DIFFREALGDPSS_obs 43, DIFFREALW 40, DIFFREALW_obs 40, DIFFREALWSS 43, DIFFREALWSS_obs 43, INFCNA 40, INFCNA_obs 40, INFCNASS 43, INFCNASS_obs 43, INFCOR 40, INFCOR_obs 40, INFCORSS 43, INFCORSS_obs 43, INFK 40, INFK_obs 40, INFKSS 43, INFKSS_obs 43, R 40, R_obs 40, RSS 43, RSS_obs 43, RT2 40, RT2_obs 40, RT2SS 43, RT2SS_obs 43, unemp 40, unemp_obs 40, unempSS 43, and unempSS_obs 43.

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;

⟨edo steady state values 17⟩

%end_steady_state;

%trends;

⟨edo steady state trends 20⟩

%end_trends;


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


⟨edo steady state result return 21⟩
```

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

A.2.1 EDO Steady State Values

17 ⟨*edo steady state values* 17⟩≡ (16)

```
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 43, A_HK 43, AA 43, AHSS 43, alpha_ 43, beta_ 43, beta_0 43, beta_2 43, DD 43, delta_ 43, delta_cd 43, delta_ch 43, DIFFREALECDSS 43, DIFFREALECHSS 43, DIFFREALECSS 43, DIFFREALEIKSS 43, DIFFREALGDPSS 43, DIFFREALWSS 43, ECDSS 43, ECHSS 43, ECSS 43, EIKSS 43, empCSS 43, empKSS 43, empSCSS 43, empSKSS 43, eta_cd 43, eta_cd_eta_cnn 43, eta_ch 43, eta_ch_eta_cnn 43, eta_cnn 43, h 43, h_cd 43, h_ch 43, hc_hk 43, HCSS 43, HKSS 43, HrCSS 43, HrKSS 43, HrSCSS 43, HrSKSS 43, HSCSS 43, HSKSS 43, HSS 43, IMPHSSS 43, INFC10SS 43, INFCNASS 43, INFCORSS 43, INFCSS 43, INFGDPSS 43, INFKSS 43, INFWCSS 43, INFWKSS 43, KCDSS 43, KCHSS 43, KCSS 43, KKSS 43, LSS 43, MCCSS 43, MCKSS 43, mu_ 43, MUCSS 43, MUCSShabit 43, MUKSS 43, MUKSShabit 43, MUZCSS 43, MUZKSS 43, MUZMSS 43, ONE 43, pbeta 43, PKBSS 43, PYSS 43, QCDSS 43, QCHSS 43, QKSS 43, RCDSS 43, RCHSS 43, RCSS 43, RKSS 43, RL1SS 43, RL2SS 43, RL3SS 43, RL4SS 43, RL5SS 43, RL6SS 43, RL7SS 43, Rnr 43, rpr 43, RR 43, RSS 43, RT2SS 43, s_AS 43, s_c_ech 43, s_ecdc 43, s_k 43, s_k_ecd 43, s_k_eik 43, s_yc 43, sigmah 43, sigman 43, theta_c 43, theta_k 43, theta_wc 43, theta_wk 43, tp2 43, UCSS 43, UHCSS 43, UHKSS 43, UHSCSS 43, UHSKSS 43, UKSS 43, unempSS 43, USS 43, WCSS 43, WKSS 43, xsi_HrC 43, xsi_HrK 43, xsi_NC 43, xsi_NK 43, ycbi 43, ycbi_ykb 43, YCSS 43, ykb 43, YKSS 43, and YYSS 43.

A.2.2 EDO Steady State Trends

20 ⟨*edo steady state trends* 20⟩≡ (16)

```
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 43, DIFFREALECHSS_obs 43, DIFFREALECSS_obs 43, DIFFREALEIKSS_obs 43, DIFFREALGDPSS_obs 43, DIFFREALWSS_obs 43, INFCNASS_obs 43, INFCORSS_obs 43, INFCSS 43, INFKSS 43, INFKSS_obs 43, MUCSS 43, MUKSS 43, RSS 43, RSS_obs 43, RT2SS 43, RT2SS_obs 43, s_ecdc 43, s_k 43, unempSS 43, and unempSS_obs 43.

A.2.3 EDO Steady State Result Return

21 ⟨*edo steady state result return* 21⟩≡ (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 43, beta_ 43, DIFFREALECDSS 43, DIFFREALECDSS_obs 43, DIFFREALECHSS 43, DIFFREALECHSS_obs 43, DIFFREALECSS 43, DIFFREALECSS_obs 43, DIFFREALEIKSS 43, DIFFREALEIKSS_obs 43, DIFFREALGDPSS 43, DIFFREALGDPSS_obs 43, DIFFREALWSS 43, DIFFREALWSS_obs 43, ECDSS 43, ECHSS 43, ECSS 43, EIKSS 43, empCSS 43, empKSS 43, empSCSS 43, empSKSS 43, HCSS 43, HKSS 43, HrCSS 43, HrKSS 43, HrSCSS 43, HrSKSS 43, HSCSS 43, HSKSS 43, INFC10SS 43, INFCNASS 43, INFCNASS_obs 43, INFCORSS 43, INFCORSS_obs 43, INFCSS 43, INFGDPSS 43, INFKSS 43, INFKSS_obs 43, INFWCSS 43, INFWKSS 43, KCDSS 43, KCHSS 43, KCSS 43, KKSS 43, LSS 43, MCCSS 43, MCKSS 43, MUCSS 43, MUKSS 43, MUZKSS 43, MUZMSS 43, ONE 43, PKBSS 43, QCDSS 43, QCHSS 43, QKSS 43, RCDSS 43, RCHSS 43, RCSS 43, RKSS 43, RL1SS 43, RL2SS 43, RL3SS 43, RL4SS 43, RL5SS 43, RL6SS 43, RL7SS 43, RSS 43, RSS_obs 43, RT2SS 43, RT2SS_obs 43, UHCSS 43, UHKSS 43, UHSCSS 43, UHSKSS 43, unempSS 43, unempSS_obs 43, USS 43, WCSS 43, WKSS 43, YCSS 43, and YKSS 43.

A.3 linearized.mod

23 ⟨*srcedo/linearized.mod* 23⟩≡
  ⟨*common setup* 38⟩
  ⟨*linearized calibrated ME* 24⟩
  ⟨*common stoch sim* 39⟩

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

A.3.1 Linearized Calibrated Measured Equations

24 ⟨*linearized calibrated ME* 24⟩≡ (23)

```
  ⟨linearized cme model block 25⟩

  // XXXXXXXXXXXXXXXXXXXXXXXXXXXX
  // labor block
  // TOTAL LABOR INPUT (called "(LSS*exp(L))" in the paper, I kept the "H" notation of
  ⟨linearized cme model labor 26⟩

  // Identities
  ⟨linearized cme identities 27a⟩


  // XXXXXXXXXXXXXXXXXXXXX
  // Aggregate hours equals agg hours in each sector
  ⟨linearized cme hours 27b⟩

  // See Section 8: Data Identities

  // new equations
  // Durable Block

  ⟨linearized cme data identities 27c⟩

  // Housing Block
  ⟨linearized cme housing 28⟩

  //measurement_equations;
  ⟨linearized cme measurement 29⟩

  //end_measurement_equations;

  end;
```

Uses L 40 and LSS 43.

A.3.2 Linearized Calibrated ME Model

25 ⟨*linearized cme model block* 25⟩≡ (24)

```
(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_ 43, beta_ 43, betas 40, delta_ 43, EC 40, ECSS 43, EFFK 40, EIK 40, EIKSS 43, ePMKC 42, ePMKK 42, eR 42, eta_cnn 43, gam_pc 43, h 43, HC 40, HCSS 43, HK 40, HKSS 43, INFC 40, INFCNA 40, INFCNASS 43, INFCSS 43, INFK 40, INFKSS 43, KC 40, KCSS 43, KK 40, KKSS 43, L 40, LSS 43, MCC 40, MCCSS 43, MCK 40, MCKSS 43, mu_ 43, MUC 40, MUCSS 43, MUK 40, MUKSS 43, ONE 43, PFGAP 40, phi_pc 43, phi_u 43, PKB 40, PKBSS 43, QK 40, QKSS 43, R 40, r_dinf 43, r_inf 43, r_y 43, RC 40, RCSS 43, rho_R 43, RK 40, RKSS 43, rpr 43, RSS 43, theta_c 43, theta_k 43, UC 40, UK 40, USS 43, WC 40, WCSS 43, WK 40, WKSS 43, YC 40, YCSS 43, YK 40, and YKSS 43.

A.3.3 Linearized Calibrated ME Labor BLock

26 ⟨*linearized cme model labor* 26⟩≡ (24)

```
-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 43, A_HK 43, beta_ 43, DIFFNORMGDP 40, DIFFREALEC 40, DIFFREALECSS 43, DIFFREALEIK 40, DIFFREALEIKSS 43, DIFFREALGDP 40, DIFFREALGDPSS 43, EC 40, ECD 40, ECDSS 43, ECH 40, ECHSS 43, ECSS 43, EFFK 40, EIK 40, EIKSS 43, empC 40, empCSS 43, empK 40, empKSS 43, empSC 40, empSCSS 43, empSK 40, empSKSS 43, eXiL 42, gam_h 43, gam_ic 43, gam_wc 43, HC 40, HCSS 43, HG 40, HK 40, HKSS 43, HrC 40, HrCSS 43, HrK 40, HrKSS 43, HrSC 40, HrSCSS 43, HrSK 40, HrSKSS 43, HSC 40, HSCSS 43, HSK 40, HSKSS 43, INFC 40, INFCSS 43, INFK 40, INFKSS 43, INFWC 40, INFWCSS 43, INFWK 40, INFWKSS 43, KC 40, KCSS 43, KK 40, KKSS 43, L 40, Lpref 40, LSS 43, MUC 40, MUCSS 43, MUK 40, MUKSS 43, NORMINFGDP 40, ONE 43, phi_H 43, phi_ic 43, phi_wc 43, PKB 40, PKBSS 43, QK 40, QKSS 43, s_k 43, sigmah 43, sigman 43, theta_wc 43, theta_wk 43, UHC 40, UHCSS 43, UHK 40, UHKSS 43, UHSC 40, UHSCSS 43, UHSK 40, UHSKSS 43, unemp 40, unempSS 43, WC 40, WCSS 43, WK 40, WKSS 43, XiL 40, xsi_HrC 43, xsi_HrK 43, xsi_NC 43, xsi_NK 43, YC 40, YCSS 43, YK 40, and YKSS 43.

A.3.4 Linearized Calibrated ME Identities

27a ⟨*linearized cme identities* 27a⟩≡ (24)

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

Uses AHSS 43, DIFFREALW 40, DIFFREALWSS 43, HCSS 43, HKSS 43, INFC 40, INFCSS 43, INFWC 40, INFWCSS 43, INFWK 40, and INFWKSS 43.

A.3.5 Linearized Calibrated ME Hours

27b ⟨*linearized cme hours* 27b⟩≡ (24)

```
(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 40, AHSS 43, alpha_ 43, betarl 43, DIFFREALGDP 40, DIFFREALGDPSS 43, GAP 40, HC 40, HCSS 43, HK 40, HKSS 43, INFC 40, INFC10 40, INFC10SS 43, INFCNA 40, INFCNASS 43, INFCOR 40, INFCORSS 43, INFCSS 43, INFGDP 40, INFGDPSS 43, INFK 40, INFKSS 43, MUC 40, MUCSS 43, ONE 43, PFGAP 40, PKB 40, PKBSS 43, s_ecdc 43, s_k 43, UC 40, UK 40, USS 43, YC 40, YCSS 43, YK 40, and YKSS 43.

A.3.6 Linearized Calibrated ME Data Itentities

27c ⟨*linearized cme data identities* 27c⟩≡ (24)

```
(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_ 43, delta_cd 43, ECD 40, ECDSS 43, EFFECD 40, eta_cd 43, gam_icd 43, h_cd 43, KCDSS 43, KD 40, L 40, LAGKD 40, LSS 43, MUK 40, MUKSS 43, ONE 43, phi_cd 43, PKB 40, PKBSS 43, QCD 40, QCDSS 43, RCD 40, and RCDSS 43.

A.3.7 Linearized Calibrated ME Housing

28 ⟨*linearized cme housing* 28⟩≡ (24)

```
(LSS*exp(L))*(RCHSS*exp(RCH))-eta_ch/((KCHSS*exp(KCH(-1)))/(MUCSS*exp(MUC))-h_ch*(KC
(QCHSS*exp(QCH))-beta_*(1/(ONE*exp(EFFECH)))*(LSS*exp(L(+1)))/(LSS*exp(L))/(MUCSS*ex
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-
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((E
ln((DIFFREALECDSS*exp(DIFFREALECD)))-ln((MUKSS*exp(MUK)))-ln((ECDSS*exp(ECD)))+ln((E
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((RL
```

Uses alpha_ 43, beta_ 43, betas 40, delta_ch 43, DIFFREALECD 40, DIFFREALECDSS 43, DIFFREALECH 40, DIFFREALECHSS 43, eB 42, ECD 40, ECDSS 43, ECH 40, ECHSS 43, eEFFECD 42, eEFFECH 42, eEFFK 42, EFFECD 40, EFFECH 40, EFFK 40, eHG 42, eLpref 42, eMUZK 42, eMUZM 42, eSTAR 42, eta_ch 43, eXiL 42, gam_ech 43, h_ch 43, HG 40, KCDSS 43, KCH 40, KCHSS 43, KD 40, L 40, LAGKCH 40, LAGKD 40, Lpref 40, LSS 43, mu_ 43, MUC 40, MUCSS 43, MUK 40, MUKSS 43, MUZK 40, MUZKSS 43, MUZM 40, MUZMSS 43, ONE 43, phi_ech 43, phi_u 43, QCH 40, QCHSS 43, QK 40, QKSS 43, R 40, RC 40, RCH 40, RCHSS 43, RCSS 43, rho_B 43, rho_EFFECD 43, rho_EFFECH 43, rho_EFFK 43, rho_HG 43, rho_lpref 43, rho_STAR 43, rho_XiL 43, RK 40, RKSS 43, RL1 40, RL1SS 43, RL2 40, RL2SS 43, RL3 40, RL3SS 43, RL4 40, RL4SS 43, RL5 40, RL5SS 43, RL6 40, RL6SS 43, RL7 40, RL7SS 43, RSS 43, RT2 40, RT2SS 43, STAR 40, tp2 43, UC 40, UK 40, USS 43, and XiL 40.

A.3.8 Linearized Calibrated ME Measurement

29 ⟨*linearized cme measurement* 29⟩≡ (24)

```
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 40, AH_obs 40, DIFFREALEC 40, DIFFREALEC_obs 40, DIFFREALECD 40, DIFFREALECD_obs 40, DIFFREALECDSS_obs 43, DIFFREALECH 40, DIFFREALECH_obs 40, DIFFREALECHSS_obs 43, DIFFREALECSS_obs 43, DIFFREALEIK 40, DIFFREALEIK_obs 40, DIFFREALEIKSS_obs 43, DIFFREALGDP 40, DIFFREALGDP_obs 40, DIFFREALGDPSS_obs 43, DIFFREALW 40, DIFFREALW_obs 40, DIFFREALWSS_obs 43, INFCNA 40, INFCNA_obs 40, INFCNASS_obs 43, INFCOR 40, INFCOR_obs 40, INFCORSS_obs 43, INFK 40, INFK_obs 40, INFKSS_obs 43, R 40, R_obs 40, RSS_obs 43, RT2 40, RT2_obs 40, RT2SS_obs 43, unemp 40, unemp_obs 40, and unempSS_obs 43.

A.4 linearized_steadystate.m

30 ⟨*srcedo/linearized.steadystate.m* 30⟩≡

```
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;

⟨linearized steady state values 31⟩

%end_steady_state;

%trends;

⟨linearized steady state trends 34⟩

%end_trends;


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

⟨linearized steady state result return 35⟩
```

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

A.4.1 Linearized Steady State Values

31 ⟨*linearized 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 43, A_HK 43, AA 43, AHSS 43, alpha_ 43, beta_ 43, beta_0 43, beta_2 43, DD 43, delta_ 43, delta_cd 43, delta_ch 43, DIFFREALECDSS 43, DIFFREALECHSS 43, DIFFREALECSS 43, DIFFREALEIKSS 43, DIFFREALGDPSS 43, DIFFREALWSS 43, ECDSS 43, ECHSS 43, ECSS 43, EIKSS 43, empCSS 43, empKSS 43, empSCSS 43, empSKSS 43, eta_cd 43, eta_cd_eta_cnn 43, eta_ch 43, eta_ch_eta_cnn 43, eta_cnn 43, h 43, h_cd 43, h_ch 43, hc_hk 43, HCSS 43, HKSS 43, HrCSS 43, HrKSS 43, HrSCSS 43, HrSKSS 43, HSCSS 43, HSKSS 43, HSS 43, IMPHSSS 43, INFC10SS 43, INFCNASS 43, INFCORSS 43, INFCSS 43, INFGDPSS 43, INFKSS 43, INFWCSS 43, INFWKSS 43, KCDSS 43, KCHSS 43, KCSS 43, KKSS 43, LSS 43, MCCSS 43, MCKSS 43, mu_ 43, MUCSS 43, MUCSShabit 43, MUKSS 43, MUKSShabit 43, MUZCSS 43, MUZKSS 43, MUZMSS 43, ONE 43, pbeta 43, PKBSS 43, PYSS 43, QCDSS 43, QCHSS 43, QKSS 43, RCDSS 43, RCHSS 43, RCSS 43, RKSS 43, RL1SS 43, RL2SS 43, RL3SS 43, RL4SS 43, RL5SS 43, RL6SS 43, RL7SS 43, Rnr 43, rpr 43, RR 43, RSS 43, RT2SS 43, s_AS 43, s_c_ech 43, s_ecdc 43, s_k 43, s_k_ecd 43, s_k_eik 43, s_yc 43, sigmah 43, sigman 43, theta_c 43, theta_k 43, theta_wc 43, theta_wk 43, tp2 43, UCSS 43, UHCSS 43, UHKSS 43, UHSCSS 43, UHSKSS 43, UKSS 43, unempSS 43, USS 43, WCSS 43, WKSS 43, xsi_HrC 43, xsi_HrK 43, xsi_NC 43, xsi_NK 43, ycbi 43, ycbi_ykb 43, YCSS 43, ykb 43, YKSS 43, and YYSS 43.

A.4.2 Linearized Steady State Trends

34 ⟨*linearized 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` 43, `DIFFREALECHSS_obs` 43, `DIFFREALECSS_obs` 43, `DIFFREALEIKSS_obs` 43, `DIFFREALGDPSS_obs` 43, `DIFFREALWSS_obs` 43, `INFCNASS_obs` 43, `INFCORSS_obs` 43, `INFCSS` 43, `INFKSS` 43, `INFKSS_obs` 43, `MUCSS` 43, `MUKSS` 43, `RSS` 43, `RSS_obs` 43, `RT2SS` 43, `RT2SS_obs` 43, `s_ecdc` 43, `s_k` 43, `unempSS` 43, and `unempSS_obs` 43.

A.4.3 Linearized Steady State Result Return

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

```
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 43, DIFFREALECHSS_obs 43, DIFFREALECSS_obs 43, DIFFREALEIKSS_obs 43, DIFFREALGDPSS_obs 43, DIFFREALWSS_obs 43, INFCNASS_obs 43, INFCORSS_obs 43, INFKSS_obs 43, RSS_obs 43, RT2SS_obs 43, and unempSS_obs 43.

A.5 Common Routines

A.5.1 Common Model Setup

38 ⟨*common setup* 38⟩≡ (9 23)

```
⟨common var 40⟩


⟨common varexo 42⟩


⟨common parameters 43⟩


//estimated_params;
⟨common estimated params 47⟩
//end_estimated_params;

//calibrated_params;
⟨common calibrated params 48⟩
//end_calibrated_params;

//free_params;
⟨common free params 49a⟩
//end_free_params;

//calibrated ME


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

model;
```

A.5.2 Common Stochastic Simulation

39 ⟨*common stoch sim* 39⟩≡ (9 23)

```
⟨common varobs 49b⟩

shocks;
⟨common shocks 50⟩
end;

steady;

estimated_params;
⟨common steady estimated params 51⟩

⟨common stderr 52⟩
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

40 ⟨*common var* 40⟩≡ (38)

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

Defines:
- AH, used in chunks 13a, 15, 27b, and 29.
- AH_obs, used in chunks 15, 29, 49b, and 50.
- betas, used in chunks 11, 14, 25, and 28.
- DIFFNORMGDP, used in chunks 12a and 26.
- DIFFREALEC, used in chunks 12a, 15, 26, and 29.
- DIFFREALEC_obs, used in chunks 15, 29, 49b, and 50.
- DIFFREALECD, used in chunks 14, 15, 28, and 29.
- DIFFREALECD_obs, used in chunks 15, 29, 49b, and 50.
- DIFFREALECH, used in chunks 14, 15, 28, and 29.
- DIFFREALECH_obs, used in chunks 15, 29, 49b, and 50.
- DIFFREALEIK, used in chunks 12a, 15, 26, and 29.
- DIFFREALEIK_obs, used in chunks 15, 29, 49b, and 50.
- DIFFREALGDP, used in chunks 12a, 13a, 15, 26, 27b, and 29.
- DIFFREALGDP_obs, used in chunks 15, 29, 49b, and 50.
- DIFFREALW, used in chunks 12b, 15, 27a, and 29.
- DIFFREALW_obs, used in chunks 15, 29, 49b, and 50.
- EC, used in chunks 11, 12a, 25, and 26.
- ECD, used in chunks 12–14 and 26–28.
- ECH, used in chunks 12a, 14, 26, and 28.
- EFFECD, used in chunks 13b, 14, 27c, and 28.
- EFFECH, used in chunks 14 and 28.
- EFFK, used in chunks 11, 12a, 14, 25, 26, and 28.
- EIK, used in chunks 11, 12a, 25, and 26.
- empC, used in chunks 12a and 26.
- empK, used in chunks 12a and 26.
- empSC, used in chunks 12a and 26.
- empSK, used in chunks 12a and 26.
- GAP, used in chunks 13a and 27b.
- HC, used in chunks 11–13 and 25–27.
- HG, used in chunks 12a, 14, 26, and 28.
- HK, used in chunks 11–13 and 25–27.
- HrC, used in chunks 12a and 26.
- HrK, used in chunks 12a and 26.
- HrSC, used in chunks 12a and 26.
- HrSK, used in chunks 12a and 26.
- HSC, used in chunks 12a and 26.
- HSK, used in chunks 12a and 26.
- INFC, used in chunks 11–13 and 25–27.
- INFC10, used in chunks 13a and 27b.
- INFCNA, used in chunks 11, 13a, 15, 25, 27b, and 29.
- INFCNA_obs, used in chunks 15, 29, 49b, and 50.
- INFCOR, used in chunks 13a, 15, 27b, and 29.
- INFCOR_obs, used in chunks 15, 29, 49b, and 50.
- INFGDP, used in chunks 13a and 27b.
- INFK, used in chunks 11–13, 15, 25–27, and 29.
- INFK_obs, used in chunks 15, 29, 49b, and 50.
- INFWC, used in chunks 12, 26, and 27a.
- INFWK, used in chunks 12, 26, and 27a.
- KC, used in chunks 11, 12a, 25, and 26.
- KCH, used in chunks 14 and 28.
- KD, used in chunks 13b, 14, 27c, and 28.

KK, used in chunks 11, 12a, 25, and 26.
L, used in chunks 10–14 and 24–28.
LAGKCH, used in chunks 14 and 28.
LAGKD, used in chunks 13b, 14, 27c, and 28.
Lpref, used in chunks 12a, 14, 26, and 28.
MCC, used in chunks 11 and 25.
MCK, used in chunks 11 and 25.
MUC, used in chunks 11–14 and 25–28.
MUK, used in chunks 11–14 and 25–28.
MUZK, used in chunks 14 and 28.
MUZM, used in chunks 14 and 28.
NORMINFGDP, used in chunks 12a and 26.
PFGAP, used in chunks 11, 13a, 25, and 27b.
PKB, used in chunks 11–13 and 25–27.
QCD, used in chunks 13b and 27c.
QCH, used in chunks 14 and 28.
QK, used in chunks 11, 12a, 14, 25, 26, and 28.
R, used in chunks 11, 14, 15, 25, 28, and 29.
R_obs, used in chunks 15, 29, and 49b.
RC, used in chunks 11, 14, 25, and 28.
RCD, used in chunks 13b and 27c.
RCH, used in chunks 14 and 28.
RK, used in chunks 11, 14, 25, and 28.
RL1, used in chunks 14 and 28.
RL2, used in chunks 14 and 28.
RL3, used in chunks 14 and 28.
RL4, used in chunks 14 and 28.
RL5, used in chunks 14 and 28.
RL6, used in chunks 14 and 28.
RL7, used in chunks 14 and 28.
RT2, used in chunks 14, 15, 28, and 29.
RT2_obs, used in chunks 15, 29, 49b, and 50.
STAR, used in chunks 14 and 28.
UC, used in chunks 11, 13a, 14, 25, 27b, and 28.
UHC, used in chunks 12a and 26.
UHK, used in chunks 12a and 26.
UHSC, used in chunks 12a and 26.
UHSK, used in chunks 12a and 26.
UK, used in chunks 11, 13a, 14, 25, 27b, and 28.
unemp, used in chunks 12a, 15, 26, and 29.
unemp_obs, used in chunks 15, 29, 49b, and 50.
WC, used in chunks 11, 12a, 25, and 26.
WK, used in chunks 11, 12a, 25, and 26.
XiL, used in chunks 12a, 14, 26, and 28.
YC, used in chunks 11–13 and 25–27.
YK, used in chunks 11–13 and 25–27.

A.5.4 Common VarExo

42 ⟨*common varexo* 42⟩≡ (38)

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

Defines:

`eB`, used in chunks 14, 28, 50, and 52.
`eEFFECD`, used in chunks 14, 28, 50, and 52.
`eEFFECH`, used in chunks 14, 28, 50, and 52.
`eEFFK`, used in chunks 14, 28, 50, and 52.
`eHG`, used in chunks 14, 28, 50, and 52.
`eLpref`, used in chunks 14, 28, 50, and 52.
`eMUZK`, used in chunks 14, 28, 50, and 52.
`eMUZM`, used in chunks 14, 28, 50, and 52.
`ePMKC`, used in chunks 11, 25, 50, and 52.
`ePMKK`, used in chunks 11, 25, 50, and 52.
`eR`, used in chunks 11, 25, 50, and 52.
`eSTAR`, used in chunks 14, 28, 50, and 52.
`eXiL`, used in chunks 12a, 14, 26, 28, 50, and 52.

A.5.5 Common Parameters

43 ⟨*common parameters* 43⟩≡ (38)

```
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, 17, 26, 31, and 49a.
- `A_HK`, used in chunks 12a, 17, 26, 31, and 49a.
- `a_ks`, used in chunk 48.
- `AA`, used in chunks 17 and 31.
- `AHSS`, used in chunks 12b, 15, 17, 21, 27, and 31.
- `alpha_`, used in chunks 11, 13a, 14, 17, 25, 27b, 28, 31, and 48.
- `beta_`, used in chunks 11–14, 17, 21, 25–28, and 31.
- `beta_0`, used in chunks 17 and 31.
- `beta_2`, used in chunks 17 and 31.
- `betarl`, used in chunks 13a, 27b, and 48.
- `DD`, used in chunks 17 and 31.
- `delta_`, used in chunks 11, 17, 25, 31, and 48.
- `delta_cd`, used in chunks 13b, 17, 27c, 31, and 48.
- `delta_ch`, used in chunks 14, 17, 28, 31, and 48.
- `DIFFREALECDSS`, used in chunks 15, 17, 21, 28, and 31.
- `DIFFREALECDSS_obs`, used in chunks 15, 20, 21, 29, 34, and 35.
- `DIFFREALECHSS`, used in chunks 15, 17, 21, 28, and 31.
- `DIFFREALECHSS_obs`, used in chunks 15, 20, 21, 29, 34, and 35.
- `DIFFREALECSS`, used in chunks 15, 17, 21, 26, and 31.
- `DIFFREALECSS_obs`, used in chunks 15, 20, 21, 29, 34, and 35.
- `DIFFREALEIKSS`, used in chunks 15, 17, 21, 26, and 31.
- `DIFFREALEIKSS_obs`, used in chunks 15, 20, 21, 29, 34, and 35.
- `DIFFREALGDPSS`, used in chunks 15, 17, 21, 26, 27b, and 31.
- `DIFFREALGDPSS_obs`, used in chunks 15, 20, 21, 29, 34, and 35.
- `DIFFREALWSS`, used in chunks 15, 17, 21, 27a, and 31.
- `DIFFREALWSS_obs`, used in chunks 15, 20, 21, 29, 34, and 35.
- `ECDSS`, used in chunks 13b, 17, 21, 26–28, and 31.
- `ECHSS`, used in chunks 14, 17, 21, 26, 28, and 31.
- `ECSS`, used in chunks 17, 21, 25, 26, and 31.
- `EIKSS`, used in chunks 12a, 17, 21, 25, 26, and 31.
- `empCSS`, used in chunks 17, 21, 26, and 31.
- `empKSS`, used in chunks 17, 21, 26, and 31.
- `empSCSS`, used in chunks 17, 21, 26, and 31.
- `empSKSS`, used in chunks 17, 21, 26, and 31.

A.5.6 Common Estimated Params

47 ⟨*common estimated params* 47⟩≡ (38)

```
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 43, gam_wc 43, h 43, phi_cd 43, phi_ech 43, phi_H 43, phi_ic 43, phi_pc 43, phi_wc 43, r_inf 43, r_y 43, rho_B 43, rho_EFFECD 43, rho_EFFECH 43, rho_EFFK 43, rho_HG 43, rho_lpref 43, rho_R 43, rho_STAR 43, rho_XiL 43, sig_B 43, sig_EFFECD 43, sig_EFFECH 43, sig_EFFK 43, sig_HG 43, sig_lpref 43, sig_MUZK 43, sig_MUZM 43, sig_PMKC 43, sig_PMKK 43, sig_R 43, sig_STAR 43, sig_XiL 43, sigmah 43, sigman 43, and tp2 43.

A.5.7 Common Calibrated params

48 ⟨*common calibrated params* 48⟩≡ (38)

```
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` 43, `alpha_` 43, `betarl` 43, `delta_` 43, `delta_cd` 43, `delta_ch` 43, `g_y` 43, `gam_ech` 43, `gam_h` 43, `gam_ic` 43, `gam_icd` 43, `h_cd` 43, `h_ch` 43, `icoef` 43, `MUZKSS` 43, `MUZMSS` 43, `ONE` 43, `pbeta` 43, `phi_u` 43, `r_dinf` 43, `r_dy` 43, `rho_MUZK` 43, `rho_MUZM` 43, `rpr` 43, `s_AS` 43, `theta_c` 43, `theta_k` 43, and `unempSS` 43.

A.5.8 Common Free Params

49a ⟨*common free params* 49a⟩≡ (38)

```
//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` 43, `A_HK` 43, `eta_cd` 43, `eta_ch` 43, `eta_cnn` 43, `mu_` 43, `s_ecdc` 43, `s_k` 43, `theta_wc` 43, `theta_wk` 43, `xsi_HrC` 43, `xsi_HrK` 43, `xsi_NC` 43, and `xsi_NK` 43.

A.5.9 common Varobs

49b ⟨*common varobs* 49b⟩≡ (39)

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

Uses `AH_obs` 40, `DIFFREALEC_obs` 40, `DIFFREALECD_obs` 40, `DIFFREALECH_obs` 40, `DIFFREALEIK_obs` 40, `DIFFREALGDP_obs` 40, `DIFFREALW_obs` 40, `INFCNA_obs` 40, `INFCOR_obs` 40, `INFK_obs` 40, `R_obs` 40, `RT2_obs` 40, and `unemp_obs` 40.

A.5.10 Common Shocks

50 ⟨*common shocks* 50⟩≡ (39)

```
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 40, DIFFREALEC_obs 40, DIFFREALECD_obs 40, DIFFREALECH_obs 40, DIFFREALEIK_obs 40, DIFFREALGDP_obs 40, DIFFREALW_obs 40, eB 42, eEFFECD 42, eEFFECH 42, eEFFK 42, eHG 42, eLpref 42, eMUZK 42, eMUZM 42, ePMKC 42, ePMKK 42, eR 42, eSTAR 42, eXiL 42, INFCNA_obs 40, INFCOR_obs 40, INFK_obs 40, RT2_obs 40, sig_B 43, sig_EFFECD 43, sig_EFFECH 43, sig_EFFK 43, sig_HG 43, sig_lpref 43, sig_MUZK 43, sig_MUZM 43, sig_PMKC 43, sig_PMKK 43, sig_R 43, sig_STAR 43, sig_XiL 43, and unemp_obs 40.

A.5.11 Common Steady Estimated params

51 ⟨*common steady estimated params* 51⟩≡ (39)

```
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 43, gam_wc 43, h 43, phi_cd 43, phi_ech 43, phi_H 43, phi_ic 43, phi_pc 43, phi_wc 43, r_inf 43, r_y 43, rho_B 43, rho_EFFECD 43, rho_EFFECH 43, rho_EFFK 43, rho_HG 43, rho_lpref 43, rho_R 43, rho_STAR 43, rho_XiL 43, sigmah 43, sigman 43, and tp2 43.

A.5.12 Common Stderr

52 ⟨*common stderr* 52⟩≡ (39)

```
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` 42, `eEFFECD` 42, `eEFFECH` 42, `eEFFK` 42, `eHG` 42, `eLpref` 42, `eMUZK` 42, `eMUZM` 42, `ePMKC` 42, `ePMKK` 42, `eR` 42, `eSTAR` 42, and `eXiL` 42.

A.6 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