

Reverse Engineering the Eview Code in the FRB/US Model

Gary Young

June 19, 2016

Contents

1	Introduction	5
	Appendices	7
A	Routine Libraries	9
A.1	Master Library	9
A.1.1	quarterly date string shift	9
A.1.2	copy series into group	10
A.1.3	set group to zero	11
A.1.4	names of all series in group	11
A.1.5	create new group	12
A.1.6	interpolate unavailable observations	13
A.1.7	set fiscal policy option	14
A.1.8	set monetary policy option	15
A.1.9	set monetary policy fed funds rate	15
A.2	mce solve library	16
A.2.1	mce solve library change history	16
A.2.2	run model consistent expectations	18
A.2.3	determine endogenous and exogenous variables	25
A.2.4	determine default method, linesearch, and other options	28
A.2.5	parse options containing equal signs	32
A.2.6	parse options not containing equal signs	32
A.2.7	create common variables, strings, matrices, vectors, and tables	33
A.2.8	model consistent coefficient simulation	35
A.2.9	solve model consistent instrument values	44
A.2.10	compute derivatives of mce targets wrt mce instruments	46
A.2.11	model consistent coefficient non-monotone step-length pro- cedure	52
A.2.12	model consistent armijo optimization rule	54
A.2.13	variable terminal values	55
A.2.14	set options based on defaults and overrides	57
A.2.15	main unconstrained optimal control simulation	64
A.2.16	main optimal control simulation with inequality constraints	68

A.2.17	model consistent optimal time-consistent solution	73
A.2.18	compute derivatives of loss function targets wrt instruments	81
A.2.19	solve model consistent expectations model	83
A.2.20	convert mcz inequality constraints	85
A.2.21	shift left	88
A.2.22	find next delimiter	89
A.2.23	load frbus with transformed subsidiary model	90
A.2.24	create wage and expectation variables in forward looking model	94
B	File Contents	97
B.1	setup.sh	97
B.2	check.sh	98
B.3	srcEview/frbus.package/addins/ld.frbus.cfs/ld.frbus.cfs.prg . . .	99
B.4	srcEview/frbus.package/addins/ld.frbus.eqqs/ld.frbus.eqqs.prg . . .	100
B.5	srcEview/frbus.package/addins/ld.mce.cfs/ld.mce.cfs.prg	102
B.6	srcEview/frbus.package/addins/ld.mce.eqqs/ld.mce.eqqs.prg	105
B.7	srcEview/frbus.package/addins/ld.some.eqqs/ld.some.eqqs.prg . . .	110
B.8	srcEview/frbus.package/addins/ld.varinfo/ld.varinfo.prg	114
B.9	srcEview/frbus.package/addins/regadd.prg	116
B.10	srcEview/frbus.package/programs/example1.prg	117
B.11	srcEview/frbus.package/programs/example2.prg	120
B.12	srcEview/frbus.package/programs/example3.prg	124
B.13	srcEview/frbus.package/programs/example4.prg	130
B.14	srcEview/frbus.package/programs/ocpolicy.prg	136
B.15	srcEview/frbus.package/programs/pings.prg	144
B.16	srcEview/frbus.package/programs/plot.resids.prg	159
B.17	srcEview/frbus.package/programs/stochsim.prg	163
B.18	srcEview/frbus.package/subs/master.library.prg	175
B.19	srcEview/frbus.package/subs/mce.solve.library.prg	176
B.20	srcEview/mce.solve.package/example1.prg	181
B.21	srcEview/mce.solve.package/example2.prg	184
B.22	srcEview/mce.solve.package/example3.prg	187
B.23	srcEview/mce.solve.package/example4.prg	191
B.24	srcEview/mce.solve.package/example5.prg	194
B.25	srcEview/mce.solve.package/mce.solve.library.prg	197
B.26	srcEview/state.space.package/data.transformations.prg	198
B.27	srcEview/state.space.package/estimation.code.prg	201
B.28	srcEview/state.space.package/frbus.supply.estimation.prg	205
B.29	srcEview/state.space.package/frbus.supply.filter.prg	208
B.30	srcEview/state.space.package/initial.values.prg	211
C	Notes, Bibliography and Indexes	213
C.1	Chunks	213
C.2	Index	215

Chapter 1

Introduction

Appendices

Appendix A

Routine Libraries

A.1 Master Library

A.1.1 quarterly date string shift

9 \langle quarterly date string shift 9 $\rangle \equiv$ (175b)

```
subroutine dateshift(string %indate, string %outdate, scalar qtrshift)

' this subroutine takes the quarterly date string in %indate (ie, "2001q1") and shifts
' it qtrshift quarters, returning the result in %outdate

!dddd1 = @dateval(%indate,"yyyfq")
!dddd2 = @dateadd(!dddd1,qtrshift,"q")
%outdate = @datestr(!dddd2,"yyyfq")

endsub
```

Defines:

`dateshift`, used in chunks 13, 73, 124, and 136.

A.1.2 copy series into group

```

10  <copy series into group 10>≡ (175b)
    subroutine group2group(string %fromgroup, string %togroup, string %to_type)

    ' copies a group of series into another group.
    ,
    ' If %to_type = "suffix", then %togroup is interpreted as a suffix to be applied to
    ' %fromgroup and its series.
    ,
    ' If %to_type = "prefix", then %togroup is interpreted as a prefix to be applied to
    ' %fromgroup and its series.
    ,
    ' If %to_type = "group", the %togroup is interpreted as the name of a group that
    ' already exists.

    if %to_type = "group" then
        if {%fromgroup}.@count <> {%togroup}.@count then
            statusline ERROR in GROUP2GROUP: the two groups do not contain the same number
        endif
        for !ik1 = 1 to {%fromgroup}.@count
            %tmp = {%fromgroup}.@seriesname(!ik1)
            %tmp1 = {%togroup}.@seriesname(!ik1)
            {%tmp1} = {%tmp}
        next
    endif

    if %to_type = "suffix" then
        %tmpa = " "
        %tmpb = %fromgroup + %togroup
        for !ik1 = 1 to {%fromgroup}.@count
            %tmp = {%fromgroup}.@seriesname(!ik1) + %togroup
            %tmpa = %tmpa + " " + %tmp
            if @isobject(%tmp) then
                {%tmp} = {%fromgroup}(!ik1)
            else
                series {%tmp} = {%fromgroup}(!ik1)
            endif
        next
        group {%tmpb} {%tmpa}
    endif

    if %to_type = "prefix" then
        %tmpa = " "
        %tmpb = %togroup + %fromgroup
        for !ik1 = 1 to {%fromgroup}.@count

```

```

    %tmp = %togroup + {%fromgroup}.@seriesname(!ik1)
    %tmpa = %tmpa + " " + %tmp
    if @isobject(%tmp) then
        {%tmp} = {%fromgroup}(!ik1)
    else
        series {%tmp} = {%fromgroup}(!ik1)
    endif
    next
    group {%tmpb} {%tmpa}
endif

```

endsub

Defines:

group2group, never used.

A.1.3 set group to zero

11a $\langle \text{set group to zero 11a} \rangle \equiv$ (175b)

```

subroutine group2zero(string %group)

' set all series in an existin %group to zero

for !ik1 = 1 to {%group}.@count
    %tmp = {%group}.@seriesname(!ik1)
    {%tmp} = 0
next

```

endsub

Defines:

group2zero, used in chunks 117, 120, 124, 130, 136, 145, 160, and 164.

A.1.4 names of all series in group

11b $\langle \text{names of all series in group 11b} \rangle \equiv$ (175b)

```

subroutine groupnames2string(string %group, string %groupnames)

' creates a string of the names of all the series in a group

%groupnames = " "
for !ik1 = 1 to {%group}.@count
    %groupnames = %groupnames + " " + {%group}.@seriesname(!ik1)
next

```

endsub

Defines:

groupnames2string, used in chunk 164.

A.1.5 create new group

12 \langle create new group 12 $\rangle \equiv$ (175b)

```

subroutine groupnew(string %fromgroup, string %to)

' Creates a new group. The names of the new group and associated series are built up
' from %fromgroup with %to as a suffix. Series that do not yet exist are set to zero

%tmpa = " "
%tmpb = %fromgroup + %to
for !ik1 = 1 to {%fromgroup}.@count
    %tmp = {%fromgroup}.@seriesname(!ik1) + %to
    %tmpa = %tmpa + " " + %tmp
    if @isobject(%tmp) <> 1 then
        series {%tmp} = 0
    endif
next
group {%tmpb} {%tmpa}

endsub

```

Defines:

groupnew, used in chunks 117, 120, 124, 130, 136, 145, 160, and 164.

A.1.6 interpolate unavailable observations

```

13  <interpolate unavailable observations 13>≡ (175b)
    subroutine interp_lin(string %series_in, string %series_out, string %substart, string %subend)

    ' Subroutine that replaces NA values in a series with
    ' interpolated observations.  NA values at the beginning or end of the series are
    ' replaced with the first or last non-NA value.

    smpl %substart %subend

    ' *****
    ' check that series is not all NAs
    series tmp_check = ({%series_in} = NA)
    if @sum(tmp_check) = @obssmpl then
        statusline Error in interp_lin subroutine:  interpolation cannot be performed because series
        stop
    endif

    series tmp_ser = ({%series_in}<>NA)
    series tmp_id = @cumsum(tmp_ser)
    series tmp_next = @sumsby({%series_in},tmp_id(-1))
    series tmp_prev = @sumsby({%series_in},tmp_id)

    ' *****
    ' check for NAs at either beginning or end of sample

    ' test for NAs at beginning of sample
    series tmp_naprev = (tmp_prev = NA)
    !flag_prev = @max(tmp_naprev)
    if !flag_prev = 1 then
        'at this point, tmp_next will have an undesired NA in its first observation;
        'change it to equal its second observation
        smpl %substart %substart
        call dateshift(%substart,%nextqtr,1)
        tmp_next = @elem(tmp_next,%nextqtr)
        smpl %substart %subend if (tmp_prev = NA)
        tmp_prev = tmp_next
        smpl %substart %subend
    endif

    ' test for NAs at end of sample
    series tmp_nanext = (tmp_next = NA)
    !flag_next = @max(tmp_nanext)
    if !flag_next = 1 then
        smpl %substart %subend if (tmp_next = NA)
    endif

```

```

    tmp_next = tmp_prev
    smpl %substart %subend
    endif

    ' *****
series tmp_lambda = (@obsid-@minsby(@obsid,tmp_id))/@sumsby(1,tmp_id)
series {%series_out} = tmp_lambda*tmp_next + (1-tmp_lambda)*tmp_prev

delete tmp_ser tmp_prev tmp_next tmp_lambda tmp_id tmp_nanext tmp_naprev tmp_check

endsub
Defines:
    interp_lin, used in chunk 164.
Uses dateshift 9.

```

A.1.7 set fiscal policy option

```

14  <set fiscal policy option 14>≡ (175b)
    subroutine set_fp(string dfpxxx)

        %policy_options = "dfpex dfpsrp dfpdbl"
        %dfpxxx = @lower(dfpxxx)
        %dfpxxx = @replace(%dfpxxx," ","")

        !kz = @wfind(%policy_options,%dfpxxx)
        if !kz > 0 then
            for !izzz = 1 to @wcount(%policy_options)
                %ppp = @word(%policy_options,!izzz)
                if !izzz = !kz then
                    series {%ppp} = 1
                else
                    series {%ppp} = 0
                endif
            next
        else
            %err = %dfpxxx + " is not a valid fiscal policy option; execution terminated"
            @uiprompt(%err)
            stop
        endif

    endsub
Defines:
    set_fp, used in chunks 117, 120, 124, 130, 136, 145, and 164.

```

A.1.8 set monetary policy option

15a $\langle \text{set monetary policy option 15a} \rangle \equiv$ (175b)

```

subroutine set_mp(string dmpxxx)

    %policy_options = "dmpex dmprr dmptay dmptlr dmpintay dmpalt dmpgen"
    %dmpxxx = @lower(dmpxxx)
    %dmpxxx = @replace(%dmpxxx," ","")

    !kz = @wfind(%policy_options,%dmpxxx)
    if !kz > 0 then
        for !izzz = 1 to @wcount(%policy_options)
            %ppp = @word(%policy_options,!izzz)
            if !izzz = !kz then
                series {%ppp} = 1
            else
                series {%ppp} = 0
            endif
        next
    else
        %err = %dmpxxx + " is not a valid monetary policy option; execution terminated"
        @uiprompt(%err)
        stop
    endif

endsub

```

Defines:

`set_mp`, used in chunks 117, 120, 124, 130, 136, 145, and 164.

A.1.9 set monetary policy fed funds rate

15b $\langle \text{set monetary policy fed funds rate 15b} \rangle \equiv$ (175b)

```

subroutine set_mpvars2rff

    rfffix = rff
    rfftay = rffe
    rfftlr = rffe
    rffalt = rff
    rffintay = rffe
    rffgen = rffe
    rrfix = rffe - @movav(picxfe,4)

endsub

```

Defines:

`set_mpvars2rff`, never used.

A.2 mce solve library

A.2.1 mce solve library change history

```

16  <mce solve library change history 16>≡ (177)
    ' Changes (1/25/14)
    ,
    ' 1. Removed defaults from mce_load_frbus subroutine
    ,
    ' 2. Added make_frbus_mcevars subroutine
    ,
    ' Changes (1/22/14)
    ,
    ' 1. Added code so that the _$sufsim string variable is assigned the
    '     alias of the currently active scenario when the %mopts argument
    '     is a null string.
    ,
    ' 2. In subroutine mcz_sim, put the contents of string variable
    '     mcz_sim_options into another string, a change which for unknown
    '     reasons eliminates an unexplained Eviews shutdown when running
    '     a pair of simulations of which the first is type=single and the
    '     second is type=opt.

    ' Changes (1/8/14)
    ,
    ' 1. Added the "dontstop" option to the %sopt argument. When invoked,
    '     this option causes the evIEWS program that calls mce_run
    '     to continue executing when running a type=single simulation if
    '     (a) the solution iterations do not
    '     converge within the maximum number of permitted iterations or
    '     (b) the evIEWS solver generates an error when solving either
    '     the backward-looking or forward-looking models in subroutine
    '     mcz_solvit. In the case of nonconvergence the call to mce_run
    '     terminates with the string value %mce_finish = "no". In the
    '     case of a solver error, the call to mce_run terminates with the
    '     string value %mce_finish = "failed_solve". Otherwise,
    '     %mce_finish = "yes".
    ,
    ,
    ' Changes (2/21/12)
    ,
    ' 1. Fixed problem with code for unconstrained TC policy
    '     calculated via EViews matrices -- the solution at period t
    '     (t = 1, ... !ndrv) must go out through period
    '     the farthest period ever solved -- (!nevl + !ndrv - 1) --
    '     which requires that the opt derivative

```



```

'   matrix must span this many quarters.  The constrained TC
'
' 2. Wrote code for constrained TC policy in EViews when there is
'   a single instrument
'
' 3. Fixed code for constrained TC policy in R -- still need to do matlab

' Changes (2/16/12)
'
' 1. Fixed problem with constrained optimization when
'   number of evaluation periods is not the same as
'   the number of instrument periods
'   (subroutine mcz_opt_qp).  This undoes part of the
'   1/17/12 change #3
'
' Changes (2/13/12)
'
' 1. Added "d=" option for TC damping factor (!tcdamp)

' Changes (1/30/12)
'
' 1. Modified subroutine mca_opt_qp to set options in matlab
'   quadprog function call
'
' Changes (1/17/12)
'
' 1. Added new subroutine (mcz_opt_tc) and new simtype (opttc)
' 2. Added new keywords:  ideriv, for sopt string
'   /xopen, /xclose for R and matlab
' 3. Modified subroutine mcz_opt_qp so that the dimensions of the
'   initial and transformed constraint matrices are based on !ndrv not !nevl
' 4. Dropped the explicit optqp simtype
'

```

Uses make_frbus_mcevars 94, mce_load_frbus 90, mce_run 18, mcz_opt_qp 68, mcz_opt_tc 73,
mcz_sim 35, and mcz_solvit 44.

A.2.2 run model consistent expectations

```

18  <run model consistent expectations 18>≡ (177)
      subroutine mce_run(string m_opts, string a_opts, string s_opts)

      ' Driver program

      %mcestart = @word(@pagesmpl,1)
      %mceend = @word(@pagesmpl,2)

      ' *****
      ' 1. Examine m_opts string (defines or creates models, mce errors and instruments)
      ' *****
      '
      ' Case 1: null string (ie, "")
      '      => use existing models whose names are contained in %$_mod_b and
      '           %$_mod_f; use existing objects $_mce_instrus
      '           and $_mce_errs.
      ' Case 2: string contains the keywords "create" and "mod=<modname>"
      '      => a model named <modname> is in the workfile and contains explicit
      '           leads; parse it to create the objects $_mod_b, $_mod_f,
      '           $_mce_instrus, and $_mce_errs
      ' Case 3: string contains the keywords "mod_b=", "mod_f=", "mce_errs=", "mce_instrus="
      '           or the keywords "mod_b=", "mod_f=", "mce_vars="
      '      => each keyword must be assigned to a string variable, whose contents are
      '           used to define $_mod_b, $_mod_f, $_mce_errs, and
      '           $_mce_instrus

      if @isempty(m_opts) = 1 then
      ' *****
      ' case 1 code
      !m_case = 1
      !z1 = @isobject(%$_mod_b)
      !z2 = @isobject(%$_mod_f)
      !z3 = @isobject("$_mce_instrus")
      !z4 = @isobject("$_mce_errs")
      !zsum = !z1+!z2+!z3+!z4
      if !zsum <> 4 then
          %err = "When the first argument to mce_run is a null string, a previous call to
          %err = %err + "mce_run must have placed the names of the backward and "
          %err = %err + "forward looking models, list of mce instruments names "
          %err = %err + "and list of mce error names in various strings; at least one "
          %err = %err + "these strings either does not exist or contains the name of "
          %err = %err + "an object that does not exist. Execution terminates."

```

```

    @uiprompt(%err)
    stop
    endif
'find alias of active scenario
%endog_active = {%_$_mod_b}.@endoglist("@active")
%endog_actual = {%_$_mod_b}.@endoglist
%word1_active = @word(%endog_active,1)
%word1_actual = @word(%endog_actual,1)
!a1 = @strlen(%word1_active)- @strlen(%word1_actual)
string $_sufsim = @right(%word1_active,!a1)

else
m_opts = @lower(m_opts)
m_opts = @replace(m_opts," ","")
m_opts = @replace(m_opts,","," ")
m_opts = " " + m_opts + " "

if @instr(m_opts,"create") and @instr(m_opts,"mod=") then
' *****
' case 2 code
!m_case = 2
call mcz_equalopt("mod",m_opts)
if @len(%temp)>0 then
    %temp1 = @left(%temp,1)
    if %temp1 = "%" then
        %mod = @lower({%temp})
    else
        %mod = @lower(%temp)
    endif
    call mcz_parsemod({%mod})
endif
else
!z1 = @instr(m_opts,"mod_b=")
!z2 = @instr(m_opts,"mod_f=")
!z3 = @instr(m_opts,"mce_errs=")
!z4 = @instr(m_opts,"mce_instrus=")
!z5 = @instr(m_opts,"mce_vars=")
!zsum1 = (!z1>0)+(!z2>0)+(!z3>0)+(!z4>0)
!zsum2 = (!z1>0)+(!z2>0)+(!z5>0)
if !zsum1 = 4 or !zsum2 = 3 then
' *****
' case 3 code
!m_case = 3
call mcz_equalopt("mod_b",m_opts)
if @len(%temp)>0 then
    %temp = @lower({%temp})

```

```

        %$_mod_b = %temp
    endif
    call mcz_equalopt("mod_f",m_opts)
    if @len(%temp)>0 then
        %temp = @lower({%temp})
        %$_mod_f = %temp
    endif
    call mcz_equalopt("mce_errs",m_opts)
    if @len(%temp)>0 then
        %temp = @lower({%temp})
        group $_mce_errs {%temp}
    endif
    call mcz_equalopt("mce_instrus",m_opts)
    if @len(%temp)>0 then
        %temp = @lower({%temp})
        group $_mce_instrus {%temp}
    endif

    else
    ' *****
    ' m_opt string does not conform to a valid case
    @uiprompt("first argument to subroutine mce_run is incorrectly specified")
    stop
    endif
endif
endif

if !m_case = 2 or !m_case = 3 then
    call mcz_hasopt("adds",m_opts)
    if !hasflag = 1 then
        {%$_mod_b}.addassign @all
        {%$_mod_f}.addassign @all
    endif
    call mcz_hasopt("track",m_opts)
    if !hasflag = 1 then
        %track = "yes"
        %track_start = @word(@pagesmpl,1)
        %track_end = @word(@pagesmpl,2)
        call mcz_equalopt("tstart",m_opts)
        if @len(%temp)>0 then
            %track_start = @lower(%temp)
        endif
        call mcz_equalopt("tend",m_opts)
        if @len(%temp)>0 then
            %track_end = @lower(%temp)
        endif
    endif
endif

```

```

        smpl %track_start %track_end
        {%$_mod_b}.addinit(v=n) @all
        {%$_mod_f}.addinit(v=n) @all
    endif
endif

if !m_case = 2 then
    group $_mce_errs {%mod}_targs}
    group $_mce_instrus {%mod}_instrus}
endif

if !m_case = 3 then
    call mcz_equalopt("mce_vars",m_opts)
    if @len(%temp)>0 then
        %temp = @lower({%temp})
        %errs = @wcross("e",%temp)
        group $_mce_errs {%errs}
        %instrus = @wcross(%temp,"_a")
        group $_mce_instrus {%instrus}
    endif
endif

' *****
' 2. Examine a_opts string (specifies mce algorithm)
' *****
'
' Case 1: blank string and %existing_algos = "yes"
'         => do not call mcz_algo (use existing settings)
' Case 2: blank string and %existing_algos <> "yes"
'         => call mcz_algo to set default options
' Case 3: nonblank string
'         => call mcz_algo using string to set options overrides

if @isempty(a_opts) = 1 then
    if %existing_algos = "yes" then
        !a_case = 1
    else
        !a_case = 2
        call mcz_algo($_mod_b,$_mod_f," ",$_mce_instrus,$_mce_errs)
    endif
else
    !a_case = 3
    call mcz_algo($_mod_b,$_mod_f,a_opts,$_mce_instrus,$_mce_errs)
endif

```

```

' *****
' 3. Examine s_opts string (specifies type of simulation)
' *****

if @isempty(s_opts) = 1 then
    @uiprompt("error: no simulation action requested")
    stop
endif

' make a copy of s_opts for parsing within this subroutine
' (the original is passed to other subroutines for additional
' parsing)
string _$_opts = s_opts
$_opts = @lower($_opts)
$_opts = @replace($_opts," ","")
$_opts = @replace($_opts,","," ")
$_opts = " " + $_opts + " "

' check for keywords that pertain to all simulation types
call mcz_hasopt("scen",$_opts)
if !hasflag = 1 then
    call mcz_equalopt("suf",$_opts)
    if @len(%temp)>0 then
        string $_sufsim = @lower(%temp)
    else
        string $_sufsim = "_1"
    endif
    %sufsim = $_sufsim
    %scenname = "mce_sim" + $_sufsim
    {%$_mod_b}.scenario(n,a=%sufsim) %scenname
    {%$_mod_f}.scenario(n,a=%sufsim) %scenname
endif
call mcz_equalopt("solveopt",$_opts)
{%$_mod_b}.solveopt(o=n,g=12,z=1e-12)
{%$_mod_f}.solveopt(o=n,g=12,z=1e-12)
if @len(%temp)>0 then
    {%$_mod_b}.solveopt({{%temp}})
    {%$_mod_f}.solveopt({{%temp}})
endif
call mcz_equalopt("txt",$_opts)
if @len(%temp)>0 then
    for !j = 1 to {%temp}.@linecount
        %tmp = @lower({%temp}.@line(!j))
        {%tmp}
    next
endif

```

```

!mceshow = 1
call mcz_equalopt("o",_$_opts)
if @len(%temp)>0 then
    !mceshow = @val(%temp)
endif
call mcz_equalopt("sstart",_$_opts)
if @len(%temp)>0 then
    %mcestart = @lower(%temp)
endif
call mcz_equalopt("send",_$_opts)
if @len(%temp)>0 then
    %mceend = @lower(%temp)
endif
smpl %mcestart %mceend
!nqtrs = @obssmpl
call mcz_hasopt("cleanup",_$_opts)
if !hasflag = 1 then
    %cleanup = "yes"
else
    %cleanup = "no"
endif
call mcz_hasopt("dontstop",_$_opts)
if !hasflag = 1 then
    %dontstop = "yes"
    if @maxerrs - @errorcount < 2 then
        !tt = @errorcount + 2
        setmaxerrs !tt
    endif
else
    %dontstop = "no"
endif

' create various program variables and objects that are needed by
' all simulation types; compute initial Jacobian in some cases
call mcz_sim_setup
if %mcz_sim_setup = "err" then
    %mce_finish = "failed_solve"
    return
endif

' determine simulation type
call mcz_equalopt("type",_$_opts)
if @len(%temp) > 0 then
    %simtype = @lower(%temp)
    if %simtype = "single" then
        call mcz_sim(_$_opts)
    endif
endif

```

```

if %mcz_sim = "err" then
    %mce_finish = "failed_solve"
    return
endif
else
if %simtype = "opt" or %simtype = "opttc" then
    call mcz_equalopt("targs",_$_opts)
    if @len(%temp)>0 then
        %targs = @lower(%temp)
    else
        @uiprompt("error:  targs keyword is missing")
        stop
    endif
    call mcz_equalopt("instrus",_$_opts)
    if @len(%temp)>0 then
        %instrus = @lower(%temp)
    else
        @uiprompt("error:  instrus keyword is missing")
        stop
    endif
    call mcz_equalopt("cnstr",_$_opts)
    if @len(%temp)>0 then
        %cnstr = @lower(%temp)
        %cnstrflag = "yes"
    else
        text _$_blanktext
        %cnstr = "_$_blanktext"
        %cnstrflag = "no"
    endif
    call mcz_opt_setup(s_opts,{%instrus},{%targs},{%cnstr})
else
    @uiprompt("error:  invalid simtype")
    stop
endif
endif
else
    @uiprompt("error:  required simtype keyword not found")
    stop
endif

```

endsub

Defines:

mce_run, used in chunks 16, 33, 35, 120, 130, 136, 145, 181, 184, 187, 191, and 194.

Uses mcz_algo 28, mcz_equalopt 32a, mcz_hasopt 32b, mcz_opt_setup 57, mcz_parsemod 25, mcz_sim 35, and mcz_sim_setup 33.

A.2.3 determine endogenous and exogenous variables

```

25  <determine endogenous and exogenous variables 25>≡ (177)
    subroutine mcz_parsemod(model modo)
    ,
    ' This subroutine takes modo, a model with explicit leads, and creates
    ' four objects. The name of each created object starts with the name of the
    ' input model, which is denoted by <modname>.
    ,
    ' 1. Model _$_<modname>_b is the same as modo but with all leads replaced
    ' with exogenous variables
    ' 2. Model _$_<modname>_f contains the MCE error equations
    ' 3. String <modname>_instrus of the names of the added exogenous variables,
    ' which are the instruments to be used to drive the MCE errors to zero
    ' 4. String <modname>_targs of the names of the endogenous variables in
    ' <modname>_f, which are the names of MCE error variables

    ' preliminaries
    %ok_chars = "--+*(^ "
    freeze(_$_modtext) modo.text
    string _$_endog = @lower(modo.@endoglist)
    !nvars = @wcount(_$_endog)
    %leadnames = " "
    %errnames = " "

    %mm = @lower(modo.@name)
    %$_mod_b = "_$_" + %mm + "_b"
    %$_mod_f = "_$_" + %mm + "_f"
    model {%$_mod_b}
    model {%$_mod_f}

    ' create model _$_<modname>_b
    smpl @all
    for !i2 = 1 to _$_modtext.@linecount
        %tmp2 = @ltrim(@lower(_$_modtext.@line(!i2)))
        %aa = @left(%tmp2,1)
        if %aa <> "@" then
            for !i1 = 1 to !nvars
                !occurrence = 1
                %tmp1 = @word(_$_endog,!i1) + "("
                !kk = @instr(%tmp2,%tmp1)
                if !kk > 0 then
                    while !kk > 0
                        'three possibilities
                        ' 1. it is part of another variable name

```

```

' 2. it is a lag
' 3. it is a lead
,
'look at character in %tmp2 just before %tmp1 to make sure that
'is not part of a longer variable or coefficient name
%before = @mid(%tmp2, !kk-1,1)
if @instr(%ok_chars,%before) > 0 then
    %tmp3 = @mid(%tmp2,!kk)
    !kkll = @instr(%tmp3,"(")
    !kkrr = @instr(%tmp3,")")
    %laglead = @mid(%tmp3,!kkll, !kkrr-!kkll+1)
    !ll = 0 + {%laglead}
    if !ll <= 0 then 'it is a lag -- skip it
        !occurrence = !occurrence + 1
    endif
    if !ll > 0 then 'it is a lead -- define a new variable
        %aaa = @word(_$endog,!i1) + %laglead
        %bbb = @word(_$endog,!i1) + "_ld_" + @str(!ll)
        series {%bbb} = {%aaa}
        %tmp4 = @replace(%tmp2,%aaa,%bbb,1)
        %tmp2 = %tmp4
        %leadnames = %leadnames + " " + %bbb
    endif
    else 'it is part of another variable name -- skip it
        !occurrence = !occurrence + 1
    endif
    !kk = @instr(%tmp2,%tmp1,!occurrence)
wend
endif
next
endif
{$_$_mod_b}.append {%tmp2}
next

' create _$_<modname>_f
%leadnames = @wunique(%leadnames)
'smpl @all
for !i1 = 1 to @wcount(%leadnames)
    %tmp1 = @word(%leadnames,!i1)
    !k1 = @instr(%tmp1,"_ld_")
    %tmp2 = @left(%tmp1,!k1-1)
    %tmp3 = @mid(%tmp1,!k1+4)
    %tmp4 = "err_" + %tmp2 + "_" + @str(%tmp3)
    series {%tmp4} = 0
    %errnames = %errnames + " " + %tmp4
    %eqstring = %tmp4 + "=" + %tmp1 + "-" + %tmp2 + "(" + @str(%tmp3) + ")"

```

```
        {%$_mod_f}.append {%eqstring}
    next

' create mce instrument and target strings

string {%mm}_instrus = %leadnames
string {%mm}_targs = %errnames

endsub
```

Defines:

mcz_parsemod, used in chunk 18.

A.2.4 determine default method, linesearch, and other options

```

28  <determine default method, linesearch, and other options 28>≡ (177)
      subroutine mcz_algo(string mcz_mod_b, string mcz_mod_f, string mcz_algo_opts, group

      %existing_algos = "yes"

      mcz_algo_opts = @lower(mcz_algo_opts)
      mcz_algo_opts = @replace(mcz_algo_opts," ","")
      mcz_algo_opts = @replace(mcz_algo_opts,",",", ")
      mcz_algo_opts = " " + mcz_algo_opts + " "

      ' default values for method options
      %meth = "newton"
      %jinit = "interp"
      !nskip = 12
      !jtrigger = .5

      ' default values for linesearch options
      %linemeth = "armijo"
      !linetrigger = .9
      !mcelinemax = 10
      !lambda = 1.0
      !lrat = .5
      !mcz_step_max = 1.0

      ' default values for other options
      !mceconv = .00001
      !mcemaxiter = 20
      !mceptrb = .001
      !broymax = 600

      ' just in case
      %terminal = "no"

      ' are there overrides to defaults?
      if @len(mcz_algo_opts) > 0 then

      ' *****
      ' look for method option
      call mcz_equalopt("meth",mcz_algo_opts)
      if @len(%temp)>0 then
        %meth = @lower(%temp)
        if %meth = "broy" then
          %jinit = "bd"

```

```

        %jupdate = "na"
        %linemeth = "lmr"
        !mcemaxiter = 200
    endif
    if %meth = "qnewton" then
        %jinit = "bd"
        %jupdate = "na"
        %linemeth = "lmr"
        !mcemaxiter = 200
        call mcz_equalopt("broymax",mcz_algo_opts)
        if @len(%temp)>0 then
            !broymax = @val(%temp)
        endif
    endif
    if %meth = "ft" then
        %jinit = "identity"
        %jupdate = "na"
        %linemeth = "na"
        !mcemaxiter = 500
    endif
endif

' *****
' look for jinit and jt options
call mcz_equalopt("jinit",mcz_algo_opts)
if @len(%temp)>0 then
    %temp = @lower(%temp)
    if @instr(%temp,"interp") then
        !k1 = @instr(%temp,"(")
        !nskip = -@val(@mid(%temp,!k1))
        %tmp1 = @mid(%temp,!k1)
        %jinit = "interp"
    else
        %jinit = @lower(%temp)
    endif
endif
call mcz_equalopt("jt",mcz_algo_opts)
if @len(%temp)>0 then
    !jtrigger = @val(%temp)
endif

' *****
' look for jupdate option
%jupdate = %jinit
call mcz_equalopt("jupdate",mcz_algo_opts)
if @len(%temp)>0 then

```

```

%temp = @lower(%temp)
if @instr(%temp,"interp") then
    !k1 = @instr(%temp,"(")
    !nskip = -@val(@mid(%temp,!k1))
    %tmp1 = @mid(%temp,!k1)
    %jupdate = "interp"
else
    %jupdate = @lower(%temp)
endif
endif

' *****
' look for options related to linesearch
call mcz_equalopt("lmeth",mcz_algo_opts)
if @len(%temp)>0 then
    %linemeth = @lower(%temp)
endif
call mcz_equalopt("lt",mcz_algo_opts)
if @len(%temp)>0 then
    !linetrigger = @val(%temp)
endif
call mcz_equalopt("lmax",mcz_algo_opts)
if @len(%temp)>0 then
    !mcelinemax = @val(%temp)
endif
call mcz_equalopt("lambda",mcz_algo_opts)
if @len(%temp)>0 then
    !lambda = @val(%temp)
endif
call mcz_equalopt("stepmax",mcz_algo_opts)
if @len(%temp)>0 then
    !mcz_step_max = @val(%temp)
endif

' *****
' look for other options
call mcz_equalopt("c",mcz_algo_opts)
if @len(%temp)>0 then
    !mceconv = @val(%temp)
endif
call mcz_equalopt("m",mcz_algo_opts)
if @len(%temp)>0 then
    !mcemaxiter = @val(%temp)
endif
call mcz_equalopt("p",mcz_algo_opts)

```

```

        if @len(%temp)>0 then
            !mceptrb = @val(%temp)
        endif

    endif

' *****
' verify the MCE instrument and error arguments

!nmceinstrus = $_mce_instrus.@count
!nmcetargs = $_mce_errs.@count

if !nmcetargs <> !nmceinstrus then
    @uiprompt("Error: There must be as many mce errors as there are mce instruments.")
    stop
endif

' *****
' check that mce instruments are exogenous variables or add factors
' in the lag model
%exog_vnames = {%$_mod_b}.@exoglist
%adds_vnames = {%$_mod_b}.@addfactors
%exog_vnames = %exog_vnames + " " + %adds_vnames
%endog_vnames = {%$_mod_b}.@endoglist
for !i = 1 to $_mce_instrus.@count
    %vvv = $_mce_instrus.@seriesname(!i)
    !cc = @wfindnc(%exog_vnames,%vvv)
    if !cc = 0 then
        %errstring = "mce control variable " + %vvv + " is not an exogenous variable or add factor"
        @uiprompt(%errstring)
        stop
    endif
endfor

' *****
' check that mce errors are endogenous variables in the lead model
%endog_lnames = {%$_mod_f}.@endoglist
for !i = 1 to $_mce_errs.@count
    %vvv = $_mce_errs.@seriesname(!i)
    !cc = @wfindnc(%endog_lnames,%vvv)
    if !cc = 0 then
        %errstring = "mce error variable " + %vvv + " is not an endogenous variable in the lead model"
        @uiprompt(%errstring)
        stop
    endif
endfor

```

```
next
```

```
endsub
```

Defines:

mcz_algo, used in chunk 18.

Uses mcz_equalopt 32a.

A.2.5 parse options containing equal signs

32a \langle parse options containing equal signs 32a $\rangle \equiv$ (177)

```
subroutine mcz_equalopt(string optionstext,string opts)
```

```

' parse an option that contains an "=" sign
optionstext = " " + optionstext + "="
!k10 = @instr(opts,optionstext)
if !k10 > 0 then
    !k11 = @len(optionstext)
    %tmp10 = @mid(opts,!k10+!k11)
    !k12 = @instr(%tmp10," ")
    %temp = @left(%tmp10,!k12-1)
else
    %temp = ""
endif
```

```
endsub
```

Defines:

mcz_equalopt, used in chunks 18, 28, 35, 57, and 90.

A.2.6 parse options not containing equal signs

32b \langle parse options not containing equal signs 32b $\rangle \equiv$ (177)

```
subroutine mcz_hasopt(string optionstext,string opts)
```

```

' parse an option that does not contain an "=" sign
!k10 = @wfind(opts,optionstext)
if !k10 > 0 then
    !hasflag = 1
else
    !hasflag = 0
endif
```

```
endsub
```

Defines:

mcz_hasopt, used in chunks 18, 35, and 57.

A.2.7 create common variables, strings, matrices, vectors, and tables

```

33  <create common variables, strings, matrices, vectors, and tables 33>≡      (177)
      subroutine mcz_sim_setup

      ' create various program variables, strings, matrices, vectors, and tables
      ' that are common to all simulation types

      !tmcetargs = !nmcetargs * !nqtrs
      !tmceinstrus = !nmceinstrus * !nqtrs
      %mce_targ_names = _$_mce_errs.@members
      %mce_instru_names = _$_mce_instrus.@members

      string _$_mod_f_exog = @lower({%$_mod_f}.@exoglist)
      string _$_mod_b_endog = @lower({%$_mod_b}.@endoglist)
      %fvars = @wintersect(_$_mod_f_exog,_$_mod_b_endog)
      {%$_mod_f}.override {%fvars}

      !re_counter = 0

      ' *****
      ' create additional matrix/vector objects

      vector(!mce_maxiter+1) _$_mce_loss_vec
      vector(!mce_maxiter+1) _$_mce_conv_vec
      vector( !nmceinstrus*!nqtrs) _$_mce_direction = 0
      vector( !nmceinstrus*!nqtrs) _$_mce_instru_vec = 0
      vector(!nmcetargs*!nqtrs) _$_mce_gap_vec
      matrix _$_mce_ptrb_mat = @filledmatrix(!nqtrs,!nmceinstrus,!mceptrb)
      matrix(!nmcetargs*!nqtrs,1) _$_mce_targ_vec
      matrix(!tmcetargs,1) _$_mce_targ_dvec

      ' *****
      ' misc

      table(!mce_maxiter+2,6) mce_sim_stats

      ' *****
      ' compute initial jacobian except when it is an identity matrix
      ' or when its been created by a previous call to mce_run
      if %jinit <> "identity" and !a_case <> 1 then
          !mceetry = 1

```

```
    smpl %mcestart %mceend
    call mcz_solvit
    if %mcz_solvit = "err" then
        %mcz_sim_setup = "err"
        return
    endif

    call mcz_derivs
    endif
```

```
endsub
```

Defines:

 mcz_sim_setup, used in chunk 18.

Uses mce_run 18, mcz_derivs 46, and mcz_solvit 44.

A.2.8 model consistent coefficient simulation

```

35  <model consistent coefficient simulation 35>≡ (177)
      subroutine mcz_sim(string mcz_sim_options)

          %mcz_sim = "ok"

          ' *****
          ' 1. set options based on defaults and overrides in string mcz_sim_options
          ' *****

          ' for some unknown reason, sometimes evIEWS bombs unless mcz_sim_options
          ' is assigned to another string before processing it
          %mso = mcz_sim_options

          %mso = @lower(%mso)
          %mso = @replace(%mso," ","")
          %mso = @replace(%mso,","," ")
          %mso = " " + %mso + " "

          %terminal = "no"
          %mcevars_b = " "
          %mcevars_f = " "

          if @len(%mso) > 0 then
              call mcz_hasopt("terminal",%mso)
              if !hasflag = 1 then
                  %terminal = "yes"
                  call mcz_equalopt("mcevars_b",%mso)
                  if @len(%temp)>0 then
                      %mcevars_b = @lower(%temp)
                      call mcz_equalopt("mcevars_f",%mso)
                      if @len(%temp)>0 then
                          %mcevars_f = @lower(%temp)
                          if @wcount(%mcevars_b) <> @wcount(%mcevars_f) then
                              %estring = "Error: mcevars_b and mcevars_f have different numbers of variables"
                              @uiprompt(%estring)
                              stop
                          endif
                      endif
                  endif
              endif
          endif

          if %linemeth = "lmr" then
              !mhistory = 4
          
```

```

!tmin = .1
!tmax = .5
!gammak = 10(-4)
endif

' *****
' 2. set up table of iteration-by-iteration statistics
' *****

if @isobject("mce_sim_stats") then
    delete mce_sim_stats
endif
table(!mce_maxiter+2,6) mce_sim_stats
mce_sim_stats.setwidth(1:1) 6
mce_sim_stats.setwidth(2:6) 11
mce_sim_stats.setlines(a2:f2) +b
setcell(mce_sim_stats,1,1,"iter")
setcell(mce_sim_stats,1,2,"converge")
setcell(mce_sim_stats,2,2,"stat")
setcell(mce_sim_stats,1,3,"SSR")
setcell(mce_sim_stats,2,3,"stat")
setcell(mce_sim_stats,1,4,"step")
setcell(mce_sim_stats,2,4,"length")
setcell(mce_sim_stats,1,5,"step")
setcell(mce_sim_stats,2,5,"iters")
setcell(mce_sim_stats,1,6,"Newton MCE")
setcell(mce_sim_stats,2,6,"deriv's?")

' *****
' 3. information text file
' *****

if @isobject("mce_sim_text") then
    delete mce_sim_text
endif
text mce_sim_text
mce_sim_text.append Simulation start = {%mcestart}
mce_sim_text.append Simulation end   = {%mceend}
mce_sim_text.append MCE method = %meth
if %meth = "newton" then
    mce_sim_text.append -- Initial jacobian = %jinit
    if %jinit = "interp" then
        mce_sim_text.append ---- Jacobian interpolation parameter = {%!nskip}
    endif
endif

```

```

    mce_sim_text.append -- Recompute Jacobian based on jtrigger = {!jtrigger}
    mce_sim_text.append -- Recompute jacobian using method = {%jupdate}
    endif
if %meth = "broy" or %meth = "qnewton" then
    mce_sim_text.append -- Initial Jacobian approximation = %jinit
    if %jinit = "interp" then
        mce_sim_text.append ---- Interpolation parameter = {!nskip}
    endif
    if %meth = "qnewton" then
        mce_sim_text.append ---- QNewton iteration switch = {!broymax}
    endif
endif
if %meth = "ft" then
    mce_sim_text.append -- Fixed step size =    {!lambda}
endif
mce_sim_text.append Linesearch method = {%linemeth}
if %linemeth <> "na" then
    mce_sim_text.append -- Linesearch trigger = {!linetrigger}
    mce_sim_text.append -- Maximum linesearch iterations =    {!mcelinemax}
endif
mce_sim_text.append Convergence criteria = {!mceconv}
mce_sim_text.append Maximum number of MCE iterations = {!mce_maxiter}
mce_sim_text.append MCE instrument perturbation factor = {!mceptrib}
mce_sim_text.append Intermediate output level factor = {!mceshow}

mce_sim_text.append MCE instrument variables = {%mce_instru_names}
mce_sim_text.append MCE error variables = {%mce_targ_names}
mce_sim_text.append There are {!tmceinstrus} instrument and {!tmcetargs} error observations

!re_counter = !re_counter + 1

' *****
' 4. solution iterations
' *****

' *****
' initialize counters, switches, etc.

!mcetry = 0
%mce_converge = "no"
smpl %mcestart %mceend
_$_mce_instru_vec = @vec(@convert(_$_mce_instrus))

' *****
' start of iteration loop

```

```

while !mcetry <= !mce_maxiter and %mce_converge = "no"

    !mcetry = !mcetry + 1
    setcell(mce_sim_stats,!mcetry+2,1,!mcetry-1,0)

    vector _$_instru_prev = _$_mce_instru_vec
    vector _$_gap_prev = _$_mce_gap_vec

    !mcz_step = !mcz_step_max
    call mcz_solvit
    if %mcz_solvit = "err" then
        %mcz_sim = "err"
        return
    endif

    if !mcetry > 1 then
        !gamma = _$_mce_loss_vec(!mcetry)/_$_mce_loss_vec(!mcetry-1)
        !loss_prev = _$_mce_loss_vec(!mcetry-1)
        setcell(mce_sim_stats,!mcetry+2,5,1,0)
        if %linemeth <> "none" and !gamma > !linetrigger then
            if %linemeth = "lmr" then
                call mcz_lmr
                if %mcz_lmr = "err" then
                    %mcz_sim = "err"
                    return
                endif
            endif
            if %linemeth = "armijo" then
                call mcz_armijo
                if %mcz_armijo = "err" then
                    %mcz_sim = "err"
                    return
                endif
            endif
        endif
    endif

    if !mceshow < 3 then
        statusline mce solution, iteration !mcetry, f(x) = !nconv
    endif

    ' test for convergence or iteration limit
    mce_sim_stats(!mcetry+2,2) = _$_mce_conv_vec(!mcetry)
    if _$_mce_conv_vec(!mcetry) < !mceconv then
        %mce_converge = "yes"
        mce_sim_text.append At iteration {!mcetry}, convergence
    endif

```

```

        %mce_finish = "yes"
    endif
    if !mcetry = !mcemaxiter and _$mce_conv_vec(!mcetry) >= !mceconv then
        mce_sim_text.append No convergence in {!mcemaxiter} iterations
        if %dontstop = "yes" then
            %mce_finish = "no"
            %mce_converge = "yes"
            mce_sim_text.append Terminating call to mce_run, but execution continues
        else
            @uiprompt("No convergence in maximum number of iterations. Execution terminating.")
            stop
        endif
    endif
endif

' continue if not converged
if %mce_converge = "no" then

' *****
' Newton algorithm (optionally update MCE jacobian)
if %meth = "newton" then
    if !mcetry = 1 then
        if %jinit = "identity" then
            matrix _$mce_der_mat = !dfactor*@identity(!nmceinstrus*!nqtrs)
        endif
        if %jinit = "bd" then
            for !ijj = 1 to !nmcetargs
                !r = (!ijj-1)*!nqtrs
                matrix _$mce_der_mat_{!ijj} = @subextract(_$mce_der_mat,!r+1,!r+1,!r+!nqtrs,!r+!nqtrs)
            next
            delete(noerr) _$mce_der_mat
        endif
        if %jinit <> "identity" then
            mce_sim_stats(!mcetry+3,6) = "yes"
        endif
    else
        if (!gamma >= !jtrigger and %jupdate <> "none") then
            %jinit_bac = %jinit
            %jinit = %jupdate
            call mcz_derivs
            %jinit = %jinit_bac
        endif
    endif

    if %jinit = "bd" then
        for !ijj = 1 to !nmcetargs

```

```

        vector _$vec_adds = -$_mce_der_mat_{!ijj}*@subextract($_mce_gap_vec, (
        matplace($_mce_direction, _$vec_adds, (!ijj-1)*!nqtrs+1, 1)
        next
    else
        $_mce_direction = -($_mce_der_mat*$_mce_gap_vec)
    endif

endif

endif

' *****
' Broyden algorithms
if %meth = "broy" or %meth = "qnewton" then
    if !mcetry = 1 then
        if %jinit <> "identity" then
            $_mce_direction = -($_mce_der_mat*$_mce_gap_vec)
        else
            matrix $_mce_der_mat = @identity(!nmceinstrus*!nqtrs)
            $_mce_direction = -$_mce_gap_vec
        endif
        if %jinit = "bd" then
            for !ijj = 1 to !nmcetargs
                !r = (!ijj-1)*!nqtrs
                matrix $_mce_der_mat_{!ijj} = @subextract($_mce_der_mat, !r+1, !r+1, !r)
            next
            if %bmeth = "qnewton" then
                delete(noerr) $_mce_der_mat
            endif
        endif
    endif
    if !mcetry > 1 then
        vector $_instru_delta = $_mce_instru_vec - $_instru_prev
        vector $_gap_delta = $_mce_gap_vec - $_gap_prev

        ' *****
        if %meth = "broy" then
            matrix $_jy = $_mce_der_mat*$_gap_delta
            matrix $_sj = @transpose($_instru_delta)*$_mce_der_mat
            scalar $_sjf = @sum(@transpose($_instru_delta)*$_jy)
            $_mce_der_mat = $_mce_der_mat + (($_instru_delta - $_jy)*$_sj)/$_sjf
            $_mce_direction = -($_mce_der_mat*$_mce_gap_vec)
        endif

        ' *****
        if %meth = "qnewton" then
            if !mcetry = 2 then
                ' create some matrices on first pass

```



```

vector(!broymax) _$_stp_nrm
matrix(!tmceinstrus,!broymax) _$_stp
vector(!broymax) _$_lam_rec
matrix(!tmceinstrus,1) _$_z
vector(!broymax) _$_counter
for !iq = 1 to !broymax
    _$_counter(!iq) = !iq
next
endif

if !mcetry <= !broymax + 1 then
    !q = !mcetry-1
    !f = !q
else
    !q = @mod(!mcetry-1,!broymax) + 1
    !f = !broymax
call shiftleft(_$_counter,1)
endif

colplace(_$_stp,_$_instru_delta,!q)
_$_stp_nrm(!q) = @norm(_$_instru_delta,2)
_$_lam_rec(!q) = !mcz_step

if %jinit = "bd" and !mcetry <= !broymax + 1 then
    for !ijj = 1 to !nmcetargs
        vector _$_vec_adds = -_$_mce_der_mat_{!ijj}* @subextract(_$_mce_gap_vec,(!ijj-1
        matplace(_$_z,_$_vec_adds,(!ijj-1)*!nqtrs+1,1)
    next
else
    _$_z = -_$_mce_gap_vec
endif

if !mcetry > 2 then
    for !kbroy = 2 to !f
        !k0 = _$_counter(!kbroy)
        !k1 = _$_counter(!kbroy - 1)
        !a = _$_lam_rec(!k1)/_$_lam_rec(!k0)
        !b = _$_lam_rec(!k1) - 1
        _$_z = _$_z + (!a*@columnextract(_$_stp,!k0)+!b*@columnextract(_$_stp,!k1))*(@t
    next
endif
!nrm2 = _$_stp_nrm(!q)^2
!lam = _$_lam_rec(!q)
!stz = @sum(@transpose(_$_instru_delta)*_$_z)
_$_mce_direction = (!nrm2*_$_z-(1-!lam)*!stz*_$_instru_delta)/(!nrm2-!lam*!stz)
endif

```

```

        endif
    endif

    ' *****
    ' Fair-Taylor algorithm
    if %meth = "ft" then
        _$_mce_direction = -!lambda*$_mce_gap_vec
    endif
    ' *****

    endif

wend

' *****
' 5. final steps
' *****

scalar _$_iterations = !mcetry

if !mceshow = 2 then
    close mce_sim_stats
endif

if !mceshow = 1 or !mceshow = 2 then
    if @isobject("mce_sim_spool") then
        delete mce_sim_spool
    endif
    spool mce_sim_spool
    mce_sim_spool.append mce_sim_stats
    mce_sim_spool.append mce_sim_text
    mce_sim_spool.name untitled01 mce_sim_stats
    mce_sim_spool.name untitled02 mce_sim_text
    show mce_sim_spool
endif

if !mceshow < 3 then
    statusline mcz_sim finished
endif

if %simtype = "single" and %cleanup = "yes" then
    delete(noerr) _$_*
endif

```

endsub

Defines:

`mcz_sim`, used in chunks 16, 18, 68, 73, 81, and 83.

Uses `mce_run` 18, `mcz_armijo` 54, `mcz_derivs` 46, `mcz_equalopt` 32a, `mcz_hasopt` 32b, `mcz_lmr` 52, `mcz_solvit` 44, and `shiftright` 88.

A.2.9 solve model consistent instrument values

```

44  <solve model consistent instrument values 44>≡ (177)
      subroutine mcz_solvit

      ' This subroutine first sets the MCE instrument values based on the current
      ' optimal direction and choice of step size, and then solves the models

      %mcz_solvit = "ok"

      ' update instrument values based on current direction and step size
      if !mcetry > 1 then
        mce_sim_stats(!mcetry+2,4) = !mcz_step
        $_mce_instru_vec = $_instru_prev + !mcz_step*$_mce_direction
        matrix $_tmp_mat = @unvec($_mce_instru_vec,!nqtrs)
        mtos($_tmp_mat,$_mce_instrus)
      endif

      ' solve lag model
      smpl %mcestart %mceend
      !err_before = @errorcount
      {%$_mod_b}.solve
      !err_after = @errorcount
      if !err_after > !err_before then
        if %dontstop = "yes" then
          %mcz_solvit = "err"
          return
        else
          @uiprompt("Error in solving lag model: execution terminating")
          stop
        endif
      endif

      ' optionally set terminal conditions on first iteration
      if !mcetry = 1 and %terminal = "yes" then
        call mcz_terminal
      endif

      ' solve lead model
      smpl %mcestart %mceend
      !err_before = @errorcount
      {%$_mod_f}.solve
      !err_after = @errorcount
      if !err_after > !err_before then
        if %dontstop = "yes" then
          %mcz_solvit = "err"

```

```

        return
    else
        @uiprompt("Error in solving lead model: execution terminating")
        stop
    endif
endif

' create group of the solution values of the mce target variables
' on the first iteration
if !mcetry = 1 then
    {%$_mod_f}.makegroup $_mce_errs_sols {%mce_targ_names}
endif

' compute mce error functions
$_mce_targ_vec = @vec(@convert($_mce_errs_sols))
$_mce_gap_vec = $_mce_targ_vec
!nloss = @norm($_mce_gap_vec,2)^2
mce_sim_stats(!mcetry+2,3) = !nloss
$_mce_loss_vec(!mcetry) = !nloss
!nconv = @max(@abs($_mce_gap_vec))
$_mce_conv_vec(!mcetry) = !nconv
if !mcetry > 1 then
    !gamma = !nloss/$_mce_loss_vec(!mcetry-1)
endif

if !mceshow = 2 then
    show mce_sim_stats
endif

endsub

```

Defines:

mcz_solvit, used in chunks 16, 33, 35, 52, and 54.

Uses mcz_terminal 55.

A.2.10 compute derivatives of mce targets wrt mce instruments

46 *(compute derivatives of mce targets wrt mce instruments 46)*≡ (177)
 subroutine mcz_derivs

'This subroutine computes the derivatives of the mce targets wrt the mce instruments

```
mce_sim_stats(!mcetry+2,6) = "yes"
matrix _$_mce_der_mat = @filledmatrix(!nmceinstrus*!nqtrs,!nmcetargs*!nqtrs,0)
```

```
if !mcetry > 1 then
  '$_mce_ptrb_mat = @abs(@unvec(!mcz_step*$_mce_direction,!nqtrs)) + 1e-6
  $_mce_ptrb_mat = @abs(@unvec(!mcz_step*$_mce_direction,!nqtrs)) + 1e-4
endif
```

```
smpl %mcestart %mceend
$_mce_targ_vec = @vec(@convert($_mce_errs_sols))
```

```
!maxlead = 1
```

```
' *****
```

```
' construct vector that determines when exact derivatives
' need to be computed
```

```
' $_dvec > 0 => period in which derivatives are to be simulated
' $_dvec = 1 => but derivatives do not have to be spread/interpolated forward or back
' $_dvec = 2 => and derivatives have to be spread forward down diagonals
' $_dvec = 3 => and derivatives have to be spread backward up diagonals
' $_dvec = 4 => and derivatives have to be interpolated backward along diagonals
' $_dvec = 5 => hybrid
vector(!nqtrs) $_dvec = 0
```

```
' *****
```

```
if %jinit = "every" then
  $_dvec = $_dvec + 1
endif
```

```
' *****
```

```
if %jinit = "interp" then
  $_dvec(1) = 1
  $_dvec(!nqtrs-!maxlead) = 4
  for !ij0 = (1+!nskip) to (!nqtrs-!maxlead-1) step !nskip
    $_dvec(!ij0) = 4
  next
  if !maxlead > 0 then
```

```

        for !ij0 = (!nqtrs-!maxlead+1) to !nqtrs
            _$_dvec(!ij0) = 1
        next
    endif
    if _$_dvec(!nqtrs-!maxlead-1) = 4 then
        _$_dvec(!nqtrs-!maxlead) = 1
    endif
endif

' *****
if %jinit = "bd" then
    !bd_col = @floor(!nqtrs/2)
    _$_dvec(!bd_col) = 5
endif

' *****
' code modified 5/27/10 to reduce the number of derivative
' sims by 1 per mc variable -- will work only when
' maxlead = 1
if %jinit = "linear" then
    _$_dvec(1) = 2
    _$_dvec(!nqtrs) = 3
    ' _$_dvec(!nqtrs-!maxlead) = 3
    'for !ij0 = !nqtrs - !maxlead + 1 to !nqtrs
    ' _$_dvec(!ij0) = 1
    ' next
endif

' *****
' *****
' Outer loop: specifies which instrument is shocked
for !ij1 = 1 to !nmceinstrus
    %instru_name = _$_mce_instrus.@seriesname(!ij1)
    statusline computing MCE derivatives for instrument %instru_name

' *****
' Middle loop: simulates effects of instrument shock for selected time periods
!skip = 0
for !ij2 = 1 to !nqtrs
    !skip = !skip + 1
    if _$_dvec(!ij2) > 0 then
        !perturbit = _$_mce_ptrb_mat(!ij2,!ij1)
        !col = (!ij1-1)*!nqtrs + !ij2
        smpl %mcestart + !ij2-1 %mcestart + !ij2-1
        {%instru_name} = {%instru_name} + !perturbit
    end
end

```

```

    smpl %mcestart %mceend
    {%$_mod_b}.solve
    {%$_mod_f}.solve
    $_mce_targ_dvec = @vec(@convert($_mce_errs_sols))
    matplace($_mce_der_mat,($_mce_targ_dvec-$_mce_targ_vec)/!perturbit,1,!col)
    smpl %mcestart + !ij2-1 %mcestart + !ij2-1
    {%instru_name} = {%instru_name} - !perturbit

' *****
' Inner loop: place and/or interpolates derivatives
if %jinit <> "every" then
    for !ij3 = 1 to !nmceinstrus
        !row = (!ij3-1)*!nqtrs + 1
        matrix $_tempa = @subextract($_mce_der_mat,!row,!col,!row+!nqtrs-1,!col)

' *****
    if %jinit = "linear" then

' forward loop moves the derivative diagonally down to the right
' when $_dvec = 2
    if $_dvec(!ij2) = 2 then
        for !ij4 = 1 to (!nqtrs - !maxlead - 1)
            matrix $_tempb = @subextract($_tempa,1,1,!nqtrs-!ij4,1)
            matplace($_mce_der_mat,$_tempb,!row+!ij4,!col+!ij4)
        next
    endif

' backward loop moves the derivative diagonally up to the left
' when $_dvec = 3
    if $_dvec(!ij2) = 3 then
        !ij5 = !nqtrs - !maxlead - 1
        for !ij4 = 1 to !ij5
            matrix $_tempb = @subextract($_tempa,(1+!ij4),1,!nqtrs-!maxlead,!ij4)
            matplace($_mce_der_mat,$_tempb,!row,!col-!ij4)
        next
    endif

' fill unassigned elements of last !maxlead rows with zeros when $_dvec
if $_dvec(!ij2) = 3 then
    !ij5 = !nqtrs - 2
    matrix(1,!ij5) $_tempc = 0
    for !ij4 = 1 to !maxlead
        matplace($_mce_der_mat,$_tempc,!row+!nqtrs-!ij4,!col-!nqtrs+2)
    next
endif

```



```
endif
```

```
' *****
if (%jinit = "interp") and (_$_dvec(!ij2) = 4) then
    !nn2 = !nqtrs-!skip+1
    'fill in columns between current and previous derivative
    '$_tempa holds current derivative, $_tempb holds previous derivative

    matrix $_tempa1 = @subextract($_tempa,1,1,!skip,1)
    matrix $_tempa2 = @subextract($_tempa,!skip+1,1,!nqtrs,1)

    'previous derivative
    matrix $_tempb = @subextract($_mce_der_mat,!row,!col-!skip,!row+!nqtrs-1,!col-!skip,1)
    matrix $_tempb1 = @subextract($_tempb,1,1,!nn2-1,1)
    matrix $_tempb2 = @subextract($_tempb,!nn2,1,!nqtrs,1)

    'forward loop moves lower (ie, nonoverlapping) portion of $_tempb diagonally down
    for !ij4 = 1 to !skip-1
        matrix $_tempb2a = @subextract($_tempb2,1,1,!skip-!ij4,1)
        matplace($_mce_der_mat,$_tempb2a,!row+!nn2-1+!ij4,!col-!skip+!ij4)
    next

    'backward loop moves upper (ie, nonoverlapping) portion of $_tempa diagonally up
    for !ij4 = 1 to !skip-1
        matrix $_tempa1a = @subextract($_tempa1,1+!ij4,1,!skip,1)
        matplace($_mce_der_mat,$_tempa1a,!row,!col-!ij4)
    next

    'overlap loop interpolates upper part of $_tempb and lower part of $_tempa
    for !ij4 = 1 to !skip-1
        matrix $_tempc = ((!skip-!ij4)/!skip)*$_tempb1 + (!ij4/!skip)*$_tempa2
        matplace($_mce_der_mat,$_tempc,!row+!ij4,!col-!skip+!ij4)
    next

    'optionally interpolate elements of last !maxlead rows across each row
    !qqqq = 0
    if !qqqq = 1 then
        !nn3 = !nqtrs - !maxlead + 1
        matrix $_tempa3 = @subextract($_tempa,!nn3,1,!nqtrs,1)
        matrix $_tempb3 = @subextract($_tempb,!nn3,1,!nqtrs,1)
        for !ij4 = 1 to !skip-1
            matrix $_tempc = ((!skip-!ij4)/!skip)*$_tempb3 + (!ij4/!skip)*$_tempa3
            matplace($_mce_der_mat,$_tempc,!row+!nn3-1,!col-!skip+!ij4)
        next
    endif
endif
```

```

endif

'*****
'The bd method takes account of only two own-derivative elements --
'one on the diagonal and one on the first super diagonal. The latter is
'important for one-lead euler equations.
',
'there is some crude coding here, reflecting the fact that the full column
'of the derivatives matrix has already been filled in and needs to be
'zeroed out

if (%jinit = "bd") and (_$_dvec(!ij2) = 5) and (!ij1=!ij3) then
  matplace(_$mce_der_mat,0*$_mce_targ_dvec,1,!col)
  !diag = $_tempa(!bd_col,1)
  vector(!nqtrs-1) $_tempb = $_tempa(!bd_col-1,1)
  'vector(!nqtrs-1) $_tempb1 = $_tempa(!bd_col+1,1)
  matrix(!nqtrs,!nqtrs) $_tempc
  matrix $_tempc = !diag*@identity(!nqtrs) + @makediagonal($_tempb,1)
  'matrix $_tempc = !diag*@identity(!nqtrs) + @makediagonal($_tempb,1) +
  $_tempc = @inverse($_tempc)
  matplace(_$mce_der_mat,$_tempc,!row,!row)
endif

'*****

next
!skip = 0
endif
endif
next
next
' *****

delete(noerr) $_tempa $_tempb $_tempc $_tempd

'recalculate current solution
smpl %mcestart %mceend
{$_$_mod_b}.solve
{$_$_mod_f}.solve

' *****
' invert derivative matrix (unless using bd method, in which case
' it's already inverted)
if %jinit <> "bd" then

```

```
statusline inverting MCE derivative matrix
_$_mce_der_mat = @inverse(_$_mce_der_mat)
endif
```

```
endsub
```

Defines:

`mcz.derivs`, used in chunks 33 and 35.

A.2.11 model consistent coefficient non-monotone step-length procedure

52 \langle model consistent coefficient non-monotone step-length procedure 52 $\rangle \equiv$ (177)

```

subroutine mcz_lmr

  %mcz_lmr = "ok"

  !step_pos = !mcz_step
  !step_neg = !mcz_step
  %line_converge = "no"
  !nk = _$mce_loss_vec(1)/(!mcetry)^2
  !loss_prev = _$mce_loss_vec(!mcetry-1)
  if !mhistory >= (!mcetry-1) then
    vector losshist = @subextract(_$mce_loss_vec,1,1,!mcetry-1,1)
  else
    vector losshist = @subextract(_$mce_loss_vec,!mcetry-!mhistory,1,!mcetry-1,1)
  endif
  !fkbar = @max(losshist)
  for !j = 1 to !mcelinemax
    if !j > 1 then
      !mcz_step = !step_pos
      call mcz_solvit
      if %mcz_solvit = "err" then
        %mcz_lmr = "err"
        return
      endif
    endif
    if !nloss <= (!fkbar + !nk - !gammak*!step_pos^2*!loss_prev) then
      %line_converge = "yes"
      exitloop
    else
      !nloss_pos = !nloss
      !mcz_step = -!step_neg
      call mcz_solvit
      if %mcz_solvit = "err" then
        %mcz_lmr = "err"
        return
      endif
      if !nloss <= (!fkbar + !nk - !gammak*!step_neg^2*!loss_prev) then
        %line_converge = "yes"
        exitloop
      endif
      !nloss_neg = !nloss
    endif
  !alphat = !step_pos^2*!loss_prev/(!nloss_pos + (2*!step_pos-1)*!loss_prev)

```

```

    if !alphan < (!tmin*!step_pos) then
        !step_pos = !tmin*!step_pos
    else
        if !alphan > !tmax*!step_pos then
            !step_pos = !tmax*!step_pos
        else
            !step_pos = !alphan
        endif
    endif
    !alphan = !step_neg^2*!loss_prev/(!nloss_neg + (2*!step_neg-1)*!loss_prev)
    if !alphan < (!tmin*!step_neg) then
        !step_neg = !tmin*!step_neg
    else
        if !alphan > !tmax*!step_neg then
            !step_neg = !tmax*!step_neg
        else
            !step_neg = !alphan
        endif
    endif
    next
    setcell(mce_sim_stats,!mccetry+2,5,!j,0)

```

endsub

Defines:

 mccz_lmr, used in chunk 35.

Uses mccz_solvit 44.

A.2.12 model consistent armijo optimization rule

54 \langle model consistent armijo optimization rule 54 $\rangle \equiv$ (177)

```

subroutine mcz_armijo

  %mcz_armijo = "ok"

  !iarm = 0
  while ((!nloss > (1-.01*!mcz_step)*!loss_prev) and !iarm < !mcelinemax)
    !mcz_step = !lrat * !mcz_step
    call mcz_solvit
    if %mcz_solvit = "err" then
      %mcz_armijo = "err"
      return
    endif
    !iarm = !iarm + 1
  wend
  setcell(mce_sim_stats,!mcetry+2,5,!iarm,0)

endsub

```

Defines:
 mcz_armijo, used in chunk 35.
 Uses mcz_solvit 44.

A.2.13 variable terminal values

```

55  {variable terminal values 55}≡ (177)
    subroutine mcz_terminal

        ' Set terminal values for all variables in _$_mod_f that might
        ' appear with leads, based on the simulated values of the corresponding
        ' variables in _$_mod_b at the end of the simulation period.
        ' Terminal values of stationary variables equal their last simulated
        ' level. Terminal values of nonstationary are based on extrapolating their
        ' simulated growth rates.
        ,

        ' This subroutine is usually called only once per MCE simulation and only
        ' for MCE simulations of permanent shocks. Multiple calls to this
        ' subroutine may result in convergence problems as a result of terminal
        ' values that drift from iteration to iteration.

        ' %fvars (variables that are exogenous in _$_mod_f and
        ' also endogenous in _$_mod_b)

        {$_$_mod_b}.makegroup _$_fvars_sols {%fvars}

        for !ijj = 1 to @wcount(%fvars)
            %tmpa = @lower(@word(%fvars,!ijj))
            %tmpb = _$_fvars_sols.@seriesname(!ijj)
            smpl %mceend + 8 %mceend + 8
            !tmpmean = @abs(@mean({%tmpa}))
            if !tmpmean > .001 then
                series _$_tmpgrowth = @abs(@movav(d({%tmpa},0,1)/{%tmpa}(-1),8))
            else
                series _$_tmpgrowth = 0
            endif
            !tmpscal = @mean(_$_tmpgrowth)
            if (!tmpscal < .001) then
                'stationary case
                smpl %mceend + 1 @last
                {%tmpb} = {%tmpb}(-1)
            else
                'nonstationary case
                smpl %mceend %mceend
                series _$_tmpgrowth = @movav(d({%tmpb},0,1)/{%tmpb}(-1),8)
                !tmpscal = @mean(_$_tmpgrowth) + 1
                smpl %mceend + 1 @last
                {%tmpb} = !tmpscal*{%tmpb}(-1)
            endif
        endfor
    endsubroutine

```

```

    {%tmpa} = {%tmpb}
    next

' mcevars_f (mce variables in $_mod_f, if any) have terminal conditions
' based on the values of mcevars_b (proxies for mce variables in $_mod_b)

if @wcount(%mcevars_f) > 0 then
    {%$_mod_b}.makegroup $_mcevars_b_sols {%mcevars_b}
    %mcevars_f1 = {%mcevars_f}
    for !ijj = 1 to @wcount(%mcevars_f1)
        %tmpa = @lower(@word(%mcevars_f1,!ijj))
        %tmpb = $_mcevars_b_sols.@seriesname(!ijj)
        smpl %mceend + 8 %mceend + 8
        !tmpmean = @abs(@mean({%tmpa}))
        if !tmpmean > .001 then
            series $_tmpgrowth = @abs(@movav(d({%tmpa},0,1)/{%tmpa}(-1),8))
        else
            series $_tmpgrowth = 0
        endif
        !tmpscal = @mean($_tmpgrowth)
        if (!tmpscal < .001) then
            'stationary case
            smpl %mceend + 1 @last
            {%tmpb} = {%tmpb}(-1)
        else
            'nonstationary case
            smpl %mceend %mceend
            series $_tmpgrowth = @movav(d({%tmpb},0,1)/{%tmpb}(-1),8)
            !tmpscal = @mean($_tmpgrowth) + 1
            smpl %mceend + 1 @last
            {%tmpb} = !tmpscal*{%tmpb}(-1)
        endif
        smpl %mceend + 1 %mceend + 8
        {%tmpa} = {%tmpb}
    next
endif

mce_sim_text.append Terminal conditions set at iteration {%mctry}

' once terminal conditions have been set once, turn the switch that
' calls this subroutine off
%terminal = "no"

endsub

```

Defines:

mcz_terminal, used in chunk 44.

A.2.14 set options based on defaults and overrides

```

57  <set options based on defaults and overrides 57>≡ (177)
    subroutine mcz_opt_setup(string mcz_opt_options, group mcz_opt_instrus, group mcz_opt_targs, te

        statusline mcz_opt_setup

' *****
' set up values of options based on defaults and overrides in string mcz_opt_options

    mcz_opt_options = @lower(mcz_opt_options)
    mcz_opt_options = @replace(mcz_opt_options," ","")
    mcz_opt_options = @replace(mcz_opt_options,",",", ")
    mcz_opt_options = " " + mcz_opt_options + " "

' default values for options

    if %simtype <> "opttc" then
        !optmaxiter = 15
        !optconv = 1e-05
    else
        !optmaxiter = 50
        !optconv = 1e-06
        !tcdamp = 1
    endif

    !optlinemax = 10
    !optptrb = .01
    !opt_step_max = 1.0
    !optshow = 3
    !mceshow = 3

    %evlstart = %mcestart
    %drvstart = %mcestart
    %freq = @pagefreq
    %evlend = @datestr(@dateadd(@dateval(%evlstart),59,%freq))
    %drvend = @datestr(@dateadd(@dateval(%drvstart),39,%freq))

    %ideriv = "yes"
    %xopen = "yes"
    %xclose = "yes"

' if imposing constraints (mcz_opt_qp)
    %qpswitch = "r"

```

```

' are there overrides to defaults?
if @len(mcz_opt_options) > 0 then
    %opts = "yes"
else
    %opts = "no"
endif

if %opts = "yes" then
    call mcz_equalopt("m",mcz_opt_options)
    if @len(%temp)>0 then
        !optmaxiter = @val(%temp)
    endif
    call mcz_equalopt("d",mcz_opt_options)
    if @len(%temp)>0 then
        !tcdamp = @val(%temp)
    endif
    call mcz_equalopt("c",mcz_opt_options)
    if @len(%temp)>0 then
        !optconv = @val(%temp)
    endif
    call mcz_equalopt("lmax",mcz_opt_options)
    if @len(%temp)>0 then
        !optlinemax = @val(%temp)
    endif
    call mcz_equalopt("p",mcz_opt_options)
    if @len(%temp)>0 then
        !optptrb = @val(%temp)
    endif
    call mcz_equalopt("stepmax",mcz_opt_options)
    if @len(%temp)>0 then
        !optstepmax = @val(%temp)
    endif
    call mcz_equalopt("oo",mcz_opt_options)
    if @len(%temp)>0 then
        !optshow = @val(%temp)
        !mceshow = @val(%temp)
    endif
    call mcz_equalopt("lstart",mcz_opt_options)
    if @len(%temp)>0 then
        %evlstart = @lower(%temp)
    endif
    call mcz_equalopt("lend",mcz_opt_options)
    if @len(%temp)>0 then
        %evlend = @lower(%temp)
    endif
    call mcz_equalopt("istart",mcz_opt_options)

```

```

    if @len(%temp)>0 then
        %drvstart = @lower(%temp)
    endif
    call mcz_equalopt("iend",mcz_opt_options)
    if @len(%temp)>0 then
        %drvend = @lower(%temp)
    endif
    call mcz_equalopt("ideriv",mcz_opt_options)
    if @len(%temp)>0 then
        %ideriv = @lower(%temp)
    endif
    call mcz_hasopt("terminal",mcz_opt_options)
    if !hasflag = 1 then
        %terminal = "yes"
    endif
    call mcz_hasopt("matlab",mcz_opt_options)
    if !hasflag = 1 then
        %qpswitch = "matlab"
    endif
    call mcz_hasopt("/xopen",mcz_opt_options)
    if !hasflag = 1 then
        %xopen = "no"
    endif
    call mcz_hasopt("/xclose",mcz_opt_options)
    if !hasflag = 1 then
        %xclose = "no"
    endif

endif

```

```

' *****

```

```

' Some preliminaries

```

```

' copy group subroutine arguments into objects with fixed names so that they can be

```

```

' easily accessed by other subroutines

```

```

copy mcz_opt_instrus      _$opt_instrus

```

```

copy mcz_opt_targs       _$opt_targs

```

```

smpl %evlstart %evlend

```

```

!nevl = @obssmpl

```

```

smpl %drvstart %drvend

```

```

!ndrv = @obssmpl

```

```

' *****

```

```

' Examine the inequality constraints, put them in a table,
' and, if necessary, augment the list of target variables to
' include all constraint variables
if %cnstrflag = "yes" then
  svector _$opt_cnstr = mcz_opt_cnstr.@svectornb
  !nconstraints = @rows(_$opt_cnstr)
  if !nconstraints > 0 then
    table(1,1) _$opt_cnstr_tab
    string _$opt_cnstr_vars = " "
    call mcz_constraints(_$opt_cnstr, _$opt_cnstr_tab, _$opt_cnstr_vars)
    %extra_targets = @wnotin(@upper(_$opt_cnstr_vars), _$opt_targs.@members)
    %extra_targets = @unique(%extra_targets)
    !nextra = @wcount(%extra_targets)
    if !nextra > 0 then
      for !qq = 1 to !nextra
        %newtarg = @word(%extra_targets, !qq)
        _$opt_targs.add {%newtarg}
        %newtarg_t = %newtarg + "_t"
        %newtarg_w = %newtarg + "_w"
        smpl @all
        series {%newtarg_t} = 0
        series {%newtarg_w} = 0
      next
    endif
  else
    @uiprompt("Error: The cnstr keyword is assigned to an empty text file")
    stop
  endif
endif

' *****
' More preliminaries

!noptinstrus = _$opt_instrus.@count
!nopttargs = _$opt_targs.@count

!topttargs = !nopttargs * !nevl
!toptinstrus = !noptinstrus * !ndrv
if !topttargs < !toptinstrus then
  statusline Error: more instruments than targets
  stop
endif

matrix _$opt_ptrb_mat = @filledmatrix(!ndrv, !noptinstrus, !optptrb)

```

```

vector(!optmaxiter+1) _$_opt_loss_vec

if %ideriv <> "no" then
    matrix _$_opt_der_mat = @filledmatrix(!toptinstrus,!topttargs,0)
endif

$_opt_targ_names = _$_opt_targs.@members
$_opt_des_names = @wcross($_opt_targ_names,"_t")
group _$_opt_des {%opt_des_names}
smpl %evlstart %evlend
matrix _$_opt_des_vec = @vec(@convert(_$_opt_des))

' if number of instruments and targets is the same, the optimal loss is zero,
' shortcut formulas can be used, and weights are not necessary
if !toptinstrus = !topttargs then
    !zero_loss = 1
    matrix _$_opt_wt_mat = @identity(!topttargs)
else
    !zero_loss = 0
    smpl %evlstart %evlend
    $_opt_wt_names = @wcross($_opt_targ_names,"_w")
    group _$_opt_wts {%opt_wt_names}
    matrix _$_opt_wt_mat = @makediagonal(@vec(@convert(_$_opt_wts)))
endif

' *****
' text file

text mce_opt_text

if %cnstrflag = "yes" then
    mce_opt_text.append constrained optimization using {%qpswitch}:  xopen = {%xopen}; xclose =
else
    mce_opt_text.append unconstrained optimization (EViews)
endif

if %simtype = "opt" then
    mce_opt_text.append optimization type = committment
    mce_opt_text.append simulation period:  {%mcestart} - {%mceend}
    mce_opt_text.append loss evaluation period: {%evlstart} - {%evlend}
    mce_opt_text.append instrument setting period: {%drvstart} - {%drvend}
    mce_opt_text.append max number of optimization iterations = !optmaxiter
    mce_opt_text.append max number of line search steps per iteration = !optlinemax
endif

```

```

if %simtype = "opttc" then
    mce_opt_text.append optimization type = time-consistent nash (discretion)
    mce_opt_text.append first instrument setting period: {%drvstart}
    mce_opt_text.append first simulation period: {%mcestart} - {%mceend}
    mce_opt_text.append first loss evaluation period: {%evlstart} - {%evlend}
    mce_opt_text.append last instrument setting period: {%drvend}
    mce_opt_text.append max number of backward-induction iterations = !optmaxiter
endif
mce_opt_text.append convergence criteria = !optconv
mce_opt_text.append output control parameter = !optshow
mce_opt_text.append compute instrument derivs? = {%ideriv}
mce_opt_text.append instrument perturbation factor = !optptrb

' *****
' table of iteration-by-iteration statistics

if @isobject("mce_opt_stats") then
    delete mce_opt_stats
endif
table(!optmaxiter+2,7) mce_opt_stats
mce_opt_stats.setwidth(1:1) 5
mce_opt_stats.setwidth(1:5) 12
mce_opt_stats.setlines(a2:g2) +b
setcell(mce_opt_stats,1,1,"iter")
setcell(mce_opt_stats,1,2,"f(x)")
setcell(mce_opt_stats,1,3,"step size")
setcell(mce_opt_stats,1,4,"convergence")
setcell(mce_opt_stats,2,4,"statistic")
setcell(mce_opt_stats,1,5,"linearity")
setcell(mce_opt_stats,2,5,"statistic")

' *****
' call appropriate optimization subroutine

if %simtype = "opt" and %cnstrflag = "no" then
    call mcz_opt
endif
if %simtype = "opt" and %cnstrflag = "yes" then
    call mcz_opt_qp
endif
if %simtype = "opttc" then
    call mcz_opt_tc
endif

```

```

' *****
' spool
  if @isobject("mce_opt_spool") then
    delete mce_opt_spool
  endif
  spool mce_opt_spool
  if %simtype = "opt" then
    mce_opt_spool.append mce_opt_stats
    mce_opt_spool.append mce_opt_text
    mce_opt_spool.name untitled01 mce_opt_stats
    mce_opt_spool.name untitled02 mce_opt_text
  endif
  if %simtype = "opttc" then
    mce_opt_spool.append mce_opt_text
    mce_opt_spool.name untitled01 mce_opt_text
  endif

  if !optshow <> 4 then
    show mce_opt_spool
    close mce_opt_stats
  endif

  if %cleanup = "yes" then
    delete(noerr) _$*
  endif

endsub

```

Defines:

mcz_opt_setup, used in chunk 18.

Uses mcz_constraints 85, mcz_equalopt 32a, mcz_hasopt 32b, mcz_opt 64, mcz_opt_qp 68,
and mcz_opt_tc 73.

A.2.15 main unconstrained optimal control simulation

```

64  <main unconstrained optimal control simulation 64>≡ (177)
    subroutine mcz_opt

    'Main subroutine for unconstrained optimal control simulations

    ' *****
    ' initialize counters and switches

    !opttry = 0
    %opt_converge = "no"
    !optnonlin = 0
    !optpchloss = 100

    ' *****
    ' initial solution

    smpl %mcestart %mceend
    !opt_step = 0
    !opt_step_prev = 0
    call mcz_opt_solve
    if !opttry = 0 and !optloss = 0 then
        %opt_converge = "yes"
        mce_opt_text.append At iteration {!opttry}, convergence
    endif

    ' *****
    ' iterate to minimize loss
    smpl %mcestart %mceend
    while !opttry <= !optmaxiter and %opt_converge = "no"

        !opttry = !opttry + 1

    ' *****
    ' compute instrument derivatives, hessian, gradient, direction, and
    ' predicted loss assuming the model is linear
    if (%ideriv <> "no" and (!opttry = 1 or @abs(!optnonlin) > .1)) then
        call mcz_opt_deriv
    endif
    if !zero_loss = 0 then

```



```

        matrix _$opt_hess = 2*$opt_der_mat*$opt_wt_mat*@transpose(_$opt_der_mat)
    else
        matrix _$opt_hess = 2*@transpose(_$opt_der_mat)
    endif
    if @issingular(_$opt_hess) = 1 then
        %errstring = "Hessian is singular at iteration " + @str(!opttry)
        @uiprompt(%errstring)
        stop
    endif
    matrix _$opt_hessinv = @inverse(_$opt_hess)
    if !zero_loss = 0 then
        matrix _$opt_grad = 2*$opt_der_mat*$opt_wt_mat*$opt_gap_vec
    else
        matrix _$opt_grad = 2*$opt_gap_vec
    endif
    vector _$opt_direction = _$opt_hessinv*$opt_grad
    matrix _$opt_gap_vec_p = _$opt_gap_vec - @transpose(_$opt_der_mat) * _$opt_direction
    !optloss_p = @sum(@transpose(_$opt_gap_vec_p)*_$opt_wt_mat*$opt_gap_vec_p)

' *****
' solve model and compute loss
!opt_step = !opt_step_max
!opt_step_prev = 0
call mcz_opt_solve

' *****
' test for nonlinearity
' !optnonlin is the ratio of the actual to predicted percentage reduction in loss less 1.0;
' the closer the model is to being linear, the closer !optnonlin is to zero
!ddloss = (_$opt_loss_vec(!opttry)-_$opt_loss_vec(!opttry+1))
if !ddloss <> 0 then
    !optnonlin = (_$opt_loss_vec(!opttry+1)-!optloss_p) / !ddloss
else
    !optnonlin = 100
endif
mce_opt_stats(!opttry+3,5) = !optnonlin

' *****
' Search for a better a step size (Armijo condition and backtracking)
!opt_loss_nderiv = -@transpose(_$opt_grad)*_$opt_direction
!kk = .01
!iarm = 0
while (_$opt_loss_vec(!opttry+1) > _$opt_loss_vec(!opttry) + !kk*!opt_step*!opt_loss_nderiv
    and !iarm < !optlinemax
    !iarm = !iarm + 1
    !opt_step_prev = !opt_step

```

```

!opt_step = .5*!opt_step
call mcz_opt_solve
wend

' *****
' Test for convergence
if !zero_loss = 1 then
' equal number of targets and controls, full rank derivative matrix
' => loss has to be less than !optconv
mce_opt_stats(!opttry+3,4) = _$opt_loss_vec(!opttry+1)
if _$opt_loss_vec(!opttry+1) < !optconv then
    %opt_converge = "yes"
endif
else
' !zero_loss = 0 => percentage change in loss from previous iteration has
' to be less than !optconv
!optpchloss = 1-(_$opt_loss_vec(!opttry+1)/_$opt_loss_vec(!opttry))
mce_opt_stats(!opttry+3,4) = !optpchloss
if !optpchloss < !optconv then
    %opt_converge = "yes"
    mce_opt_text.append At iteration {!opttry}, convergence
endif
endif

if !opttry = 1 and @abs(!optnonlin) < 1e-7 and !opt_step = 1 then
    %opt_converge = "yes"
    mce_opt_text.append At iteration 1, convergence assumed because model is linear
endif

if !opttry = !optmaxiter then
    mce_opt_text.append At iteration {!opttry}, no convergence in {!optmaxiter} iterations
    !continue = @uiprompt("Maximum number of optimization iterations reached: Continue?")
    if !continue = 2 then
        stop
    else
        %opt_converge = "yes"
    endif
endif

if !optshow <> 4 then
    show mce_opt_stats
endif
wend

```

June 19, 2016

frbuseview.nw 67

endsub

Defines:

mcz_opt, used in chunk 57.

Uses **mcz_opt.deriv** 81 and **mcz_opt.solve** 83.

A.2.16 main optimal control simulation with inequality constraints

```

68  <main optimal control simulation with inequality constraints 68>≡ (177)
    subroutine mcz_opt_qp

      ' Main subroutine for optimal control simulations with inequality constraints
      ' (requires either R or matlab)

      ' The original problem:
      '
      '   choose x to minimize  $z = (y-y^*)'W(y-y^*)$ 
      '   under the constraint  $Cy \geq c$ 
      '
      '   where at each iteration the linearized relationship between
      '   the target variables (y) and instrument variables (x) is
      '   given by
      '
      '    $y = B'x + k$ 
      '
      ' The transformed problem: Solve out y
      '
      '   choose x to minimize  $z = x'BWB'x + 2(k-y^*)'WB'x + \text{const}$ 
      '   under the constraint  $CB'x \geq c - Ck$ 
      '
      ' The R command -- quadprog::solve.QP(D,d,A,b) -- solves
      '
      '   min  $[0.5 * x' D x - d'x]$  with the constraint  $A'x \geq b$ 
      '
      ' The matlab command -- x = quadprog(D,d,A,b) -- solves
      '
      '   min  $[0.5 * x' D x + d'x]$  with the constraint  $Ax \leq b$ 
      '
      '
      '           R           matlab
      '   D       2BWB'       same
      '   d'       2(k-y*)'WB'  -2(k-y*)'WB'
      '   A'       CB'         -CB'
      '   b       c-Ck         -(c-Ck)
      '
      '   where  $k = y - B'x$ 
      '
      ' *****
      ' Create constraint matrices (C,c)-- dimensions are based on the number
      ' of periods in which the instruments are set (!ndrv) not the number of

```

```

' periods in which the loss function is evaluated (!nevl)

if %cnstrflag = "yes" then
  matrix(!nevl * !nconstraints,!nevl*!nopttargs) _$_opt_cnstr_mat = 0
  matrix(!nevl * !nconstraints,1) _$_opt_cnstr_vec = 0
  for !i = 1 to !nconstraints
    !nterms = @val(_$_opt_cnstr_tab(!i,1))
    for !k = 1 to !nterms
      %var = @upper(_$_opt_cnstr_tab(!i,2*!k))
      !coef = @val(_$_opt_cnstr_tab(!i,2*!k+1))
      !j = @wfindnc(%opt_targ_names,%var)
      for !l = 1 to !nevl
        _$_opt_cnstr_mat((!i-1)*!nevl+!l,(!j-1)*!nevl+!l) = !coef
      next
    next
    for !l = 1 to !nevl
      !coef = @val(_$_opt_cnstr_tab(!i,2*!nterms+2))
      _$_opt_cnstr_vec((!i-1)*!nevl+!l) = !coef
    next
  next
else
  !nconstraints = 1
  matrix(!nevl,!nevl*!nopttargs) _$_opt_cnstr_mat = 0
  matrix(!nevl,1) _$_opt_cnstr_vec = 0
endif

' *****
' initialize counters and switches

!opttry = 0
%opt_converge = "no"
!optnonlin = 0
!optpchloss = 100

if %qpswitch = "r" and %xopen = "yes" then
  xopen(type=r, case=lower)
endif
if %qpswitch = "matlab" and %xopen = "yes" then
  xopen(type=m, case=lower)
endif

' *****
' compute an initial solution, without reference to any constraints

```

```

smpl %mcestart %mceend
!opt_step = 0
!opt_step_prev = 0
call mcz_opt_solve

' *****
' iterate to minimize loss
smpl %mcestart %mceend
while !opttry <= !optmaxiter and %opt_converge = "no"

    !opttry = !opttry + 1

' *****
' compute instrument derivatives
if %ideriv <> "no" then
    call mcz_opt_deriv
endif

' *****
' compute qp matrices
smpl %drvstart %drvend
vector $_opt_instru_vec = @vec(@convert($_opt_instrus))
matrix $_qp_k_vec = $_opt_targ_vec - @transpose($_opt_der_mat) * $_opt_instru_vec
matrix qp_d_mat = 2 * $_opt_der_mat * $_opt_wt_mat * @transpose($_opt_der_mat)
matrix qp_d_vec = -2 * $_opt_der_mat * @transpose($_opt_wt_mat) *
                * ($_qp_k_vec - $_opt_des_vec)

matrix qp_a_mat = @transpose($_opt_cnstr_mat * @transpose($_opt_der_mat))
matrix qp_b_vec = $_opt_cnstr_vec - $_opt_cnstr_mat *
                * ($_qp_k_vec - $_opt_des_vec)

if !ndrv <> !nevl then
    qp_a_mat = @subextract(qp_a_mat,1,1,!ndrv,!ndrv)
    qp_b_vec = @subextract(qp_b_vec,1,1,!ndrv,1)
endif

xput qp_d_mat qp_a_mat qp_d_vec qp_b_vec

' *****
' *****
' R code

if %qpswitch = "r" then
    if ((!ndrv = 1) and (!noptinstrus = 1)) then
        xrun "dim(qp_d_mat) <- c(1,1);"
    
```

```

        xrun "dim(qp_a_mat) <- c(1,1);"
        xrun "dim(qp_d_vec) <- c(1,1);"
        xrun "dim(qp_b_vec) <- c(1,1);"
    endif
    xrun "QP.results <- quadprog::solve.QP(qp_d_mat,qp_d_vec,qp_a_mat,qp_b_vec);"
    xrun "r_xhat <- QP.results$solution;"
    xrun "r_xhat <- as.matrix(r_xhat);"
    xget(type = matrix, name = _$qp_instru_vec) r_xhat
endif

' *****
' *****
' matlab code

if %qpswitch = "matlab" then
    xrun "qp_d_vec = -qp_d_vec;"
    xrun "qp_a_mat = -qp_a_mat'";
    xrun "qp_b_vec = -qp_b_vec;"
    xrun "aeq = [];"
    xrun "beq = [];"
    xrun "lb = [];"
    xrun "ub = [];"
    xrun "xx0 = [];"
    xrun "options = optimset('Algorithm','active-set','LargeScale','off');"
    xrun "matlab_xhat = quadprog(qp_d_mat,qp_d_vec,qp_a_mat,qp_b_vec,aeq,beq,lb,ub,xx0,options)"
    xget matlab_xhat
    %tmp = matlab_xhat.@type
    if %tmp = "SCALAR" then
        matrix(1,1) _$qp_instru_vec = matlab_xhat
    else
        matrix _$qp_instru_vec = matlab_xhat
        delete matlab_xhat
    endif
endif

' *****
' *****

' values of target variables in qp (linearized) solution
matrix _$qp_targ_vec = @transpose(_$opt_der_mat)*_$qp_instru_vec + _$qp_k_vec

' values of target variables in original model
smpl %drvstart %drvend
matrix _$tt1 = @unvec(@vec(_$qp_instru_vec),!ndrv)
mtos(_$tt1,_$opt_instrus)
smpl %mcestart %mceend

```

```

call mcz_sim(" ")
smpl %evlstart %evlend
matrix _$_opt_targ_vec = @vec(@convert(_$_opt_targs_sols))
matrix _$_opt_gap_vec = _$_opt_targ_vec - _$_opt_des_vec
!optloss = @sum(@transpose(_$_opt_gap_vec)*_$_opt_wt_mat*_$_opt_gap_vec)
setcell(mce_opt_stats,!opttry+2,1,!opttry,0)

' convergence test is based on the maximum difference between
' the target variable values in the linearized and original models
!conv_stat = @max(abs(_$_qp_targ_vec-_$_opt_targ_vec))
mce_opt_stats(!opttry+2,2) = !optloss
mce_opt_stats(!opttry+2,4) = !conv_stat

if !optshow <> 4 then
    show mce_opt_stats
endif

if !conv_stat < !optconv then
    %opt_converge = "yes"
else
    endif
endif

wend

delete(noerr) qp_d_mat qp_a_mat qp_d_vec qp_b_vec

endsub

```

Defines:

mcz_opt_qp, used in chunks 16 and 57.

Uses mcz_opt_deriv 81, mcz_opt_solve 83, and mcz_sim 35.

A.2.17 model consistent optimal time-consistent solution

73 $\langle \text{model consistent optimal time-consistent solution 73} \rangle \equiv$ (177)

subroutine mcz_opt_tc

```

' 1. For a linear model, this code finds the exact time-consistent solution;
'   for a nonlinear model, the calculated solution is based on the
'   linearization of the model as given by the derivatives of the target
'   variables wrt the instrument variables along the baseline.
'
' 2. The basic data matrices include observations for (!nevl+!ndrv-1) periods,
'   which is also the interval over which tracking adds are required.
'
' 3. The simulation, derivative and evaluation periods are assumed to have the
'   same starting date
'
' 4. The solution method is backward induction; if there are no inequality constraints,
'   the method executes in EViews (using matrices); if inequality constraints
'   are present, the method executes in EViews if there is a single
'   instrument and in Matlab or R (depending on the
'   setting of a switch) if there are multiple instruments (which requires
'   a quadratic programming algorithm),

' *****
' Create constraint matrices (C,c)-- dimensions are based on the fact that
' each policymaker sets the instruments for a single period

if %cnstrflag = "yes" then
  matrix(!nconstraints,!nopttargs) _$_opt_cnstr_mat = 0
  matrix(!nconstraints,1) _$_opt_cnstr_vec = 0
  for !i = 1 to !nconstraints
    !terms = @val(_$_opt_cnstr_tab(!i,1))
    for !k = 1 to !terms
      %var = @upper(_$_opt_cnstr_tab(!i,2*!k))
      !coef = @val(_$_opt_cnstr_tab(!i,2*!k+1))
      !j = @wfindnc(%opt_targ_names,%var)
      _$_opt_cnstr_mat(!i,!j) = !coef
    next
    !coef = @val(_$_opt_cnstr_tab(!i,2*!terms+2))
    _$_opt_cnstr_vec(!i) = !coef
  next
endif

```

```

' *****
' initial solution, instrument derivatives, and data matrices

' initial simulation
  smpl %mcestart %mceend
  if %terminal = "yes" then
    call mcz_sim("terminal")
    %terminal = "no"
  else
    call mcz_sim(" ")
  endif
  {%$_mod_b}.makegroup $_opt_targs_sols {%opt_targ_names}
  !ntot = !ndrv + !nevl - 1

' derivative matrices and submatrices
' $_opt_der_mat (this is the basic matrix; tc_dmat is its transpose)
' tc_dmat          effect of !ndrv periods of instruments on !ntot periods of targets
' tc_dmat_short    effect of !ndrv periods of instruments on !nevl periods of targets
' tc_dmat1         effect of period 1 instrument on !nevl periods of targets
' tc_dmat2         effect of period 1 instrument on period 1 targets
  if %ideriv <> "no" then
    !opttry = 1
    matrix $_opt_der_mat = @filledmatrix(!toptinstrus,!nopttargs*!ntot,0)
    %evlend_b = %evlend
    %freq = @pagefreq
    !ntot1 = !ntot-1
    %evlend = @datestr(@dateadd(@dateval(%simstart),!ntot1,%freq))
    'call dateshift(%simstart,%evlend,!ntot-1)
    call mcz_opt_deriv
    %evlend = %evlend_b
    matrix tc_dmat = @transpose($_opt_der_mat)
    matrix tc_dmat_short = @filledmatrix(!nevl*!nopttargs,!ndrv*!noptinstrus,0)
    for !i = 1 to !toptinstrus
      for !j = 1 to !nopttargs
        !r1 = !ntot*(!j-1)+1
        !r2 = !r1 + !nevl - 1
        matplace(tc_dmat_short,@subextract(tc_dmat,!r1,!i,!r2,!i),!nevl*(!j-1)+1,!i)
      next
    next
    matrix tc_dmat1 = @filledmatrix(!nevl*!nopttargs,!noptinstrus,0)
    for !i = 1 to !noptinstrus
      matplace(tc_dmat1,@columnextract(tc_dmat_short,(!i-1)*!ndrv+1),1,!noptinstrus)
    next
    matrix tc_dmat2 = @filledmatrix(!noptinstrus,!nopttargs,0)
    for !i = 1 to !noptinstrus

```

```

        for !j = 1 to !nopttargs
            tc_dmat2(!i,!j) = tc_dmat1((!j-1)*!nevl+1,!i)
        next
    next
endif

' initial simulation again (is this necessary)
call mcz_sim(" ")

' data matrices
smp1 %simstart {%evlend} + !ndrv - 1
stom(_$opt_targs_sols,tc_ymat)
stom(_$opt_des,tc_ystarmat)
stom(_$opt_instrus,tc_xmat)
matrix tc_wmat = _$opt_wt_mat

' (no constraints) => eviews;
' (1 constraint -- single period) => eviews
' (multiple constraints) => R or Matlab
if %cnstrflag = "no" then
    %iii = "eviews"
else
    if !noptinstrus = 1 then
        %iii = "eviews"
    else
        %iii = %qpswitch
    endif
endif

scalar tc_iter = 0
scalar tc_stat = 100
scalar tc_itmax = !optmaxiter
scalar tc_conv = !optconv

' *****
' unconstrained backward induction => EViews

if %cnstrflag = "no" then
    while (tc_stat > tc_conv) and tc_iter <= tc_itmax

        matrix tc_xb = tc_xmat
        matrix tc_yb = tc_ymat

        for !i = !ndrv to 1 step -1
            !n1 = !i+!nevl-1

```

```

!n2 = !i+!ndrv-1
vector tc_y0v = @vec(@subextract(tc_ymat,!i,1,!n1,!nopttargs))
vector tc_ys0v = @vec(@subextract(tc_ystarmat,!i,1,!n1,!nopttargs))
vector tc_x0v = @vec(@subextract(tc_xmat,!i,1,!n2,!noptinstrus))
vector tc_x0_1v = @vec(@subextract(tc_xmat,!i,1,!i,!noptinstrus))
vector tc_xbv = @vec(@subextract(tc_xb,!i,1,!n2,!noptinstrus))

vector tc_y1v = tc_y0v + tc_dmat_short*(tc_x0v-tc_xbv)

matrix tc_g = @transpose(tc_dmat1)*tc_wmat*(tc_y1v-tc_ys0v)
matrix tc_h = @transpose(tc_dmat1)*tc_wmat*tc_dmat1
vector tc_x1_1v = tc_x0_1v - !tcdamp*@inverse(tc_h)*tc_g
matrix tc_x1_1 = @unvec(tc_x1_1v,!noptinstrus)
matplace(tc_xmat,tc_x1_1,!i,1)
next

vector tc_x0v = @vec(@subextract(tc_xmat,1,1,!ndrv,!noptinstrus))
vector tc_y0v = @vec(@subextract(tc_ymat,1,1,!ntot,!nopttargs))
vector tc_y1v = tc_y0v + tc_dmat*(tc_x0v-tc_xbv)
matrix tc_y1 = @unvec(tc_y1v,!ntot)
matplace(tc_ymat,tc_y1,1,1)

tc_iter = tc_iter + 1
if tc_iter > 1 then
    tc_stat = @max(@abs(tc_xmat-tc_xmatprev))
endif

matrix tc_xmatprev = tc_xmat
wend
endif

' *****
' constrained backward induction

if %cnstrflag = "yes" then

    matrix tc_cmat = _$_opt_cnstr_mat
    matrix tc_cvec = _$_opt_cnstr_vec

' *****
' single-period constraint => eviews
if %iii = "eviews" then

    statusline constrained TC optimization in EViews

```

```

matrix mat_a = tc_dmat2*@transpose(tc_cmat)

while (tc_stat > tc_conv) and tc_iter <= tc_itmax

    matrix tc_xb = tc_xmat
    matrix tc_yb = tc_ymat

    for !i = !ndrv to 1 step -1
        !n1 = !i+!nevl-1
        !n2 = !i+!ndrv-1

        ' vectors of observations in the current loss and instrument periods
        vector tc_y0v = @vec(@subextract(tc_ymat,!i,1,!n1,!nopttargs))
        vector tc_ys0v = @vec(@subextract(tc_ystarmat,!i,1,!n1,!nopttargs))
        vector tc_x0v = @vec(@subextract(tc_xmat,!i,1,!n2,!noptinstrus))
        vector tc_x0_1v = @vec(@subextract(tc_xmat,!i,1,!i,!noptinstrus))
        vector tc_xbv = @vec(@subextract(tc_xb,!i,1,!n2,!noptinstrus))

        ' vector of target variable values based on current instrument values
        ' (ie, solve model)
        vector tc_y1v = tc_y0v + tc_dmat_short*(tc_x0v-tc_xbv)

        ' unconstrained current optimal instrument value
        matrix tc_g = @transpose(tc_dmat1)*tc_wmat*(tc_y1v-tc_ys0v)
        matrix tc_h = @transpose(tc_dmat1)*tc_wmat*tc_dmat1
        vector tc_x1_1v = tc_x0_1v - @inverse(tc_h)*tc_g

        ' check constraint
        matrix tc_y1m = @unvec(tc_y1v,!nevl)
        matrix tc_ysm = @unvec(tc_ys0v,!nevl)
        matrix mattemp = @rowextract(tc_y1m,1) - (@transpose(tc_x0_1v)*tc_dmat2) - @rowextrac
        matrix mat_b = tc_cvec - tc_cmat*@transpose(mattemp)
        scalar testit = @sum(mat_b)/@sum(mat_a)
        if @sum(tc_x1_1v) < testit then
            tc_x1_1v = testit
            !tc_iter = tc_iter
        endif
        vector tc_x1_1v = (1-!tcdamp)*tc_x0_1v + !tcdamp* tc_x1_1v

        ' place current optimal instrument value in instrument matrix
        matrix tc_x1_1 = @unvec(tc_x1_1v,!noptinstrus)
        matplace(tc_xmat,tc_x1_1,!i,1)

    next

```

```

' solve model
  vector tc_x0v = @vec(@subextract(tc_xmat,1,1,!ndrv,!noptinstrus))
  vector tc_y0v = @vec(@subextract(tc_ymat,1,1,!ntot,!nopttargs))
  vector tc_y1v = tc_y0v + tc_dmat*(tc_x0v-tc_xbv)
  matrix tc_y1 = @unvec(tc_y1v,!ntot)
  matplace(tc_ymat,tc_y1,1,1)

  tc_iter = tc_iter + 1

  if tc_iter > 1 then
    tc_stat = @max(@abs(tc_xmat-tc_xmatprev))
  endif

  matrix tc_xmatprev = tc_xmat
  wend
endif

' *****
' multiple constraints => R or matlab

if %iii = "r" or %iii = "matlab" then

  scalar tc_noptinstrus = !noptinstrus
  scalar tc_nopttargs = !nopttargs
  scalar tc_nevl = !nevl
  scalar tc_ndrv = !ndrv
  scalar tc_damp = !tcdamp

  if %xopen = "yes" then
    if %iii = "r" then
      xopen(type=r)
      xrun library("quadprog")
      %wd = "" + %rpath + ""
      xrun setwd({%wd})
    else
      xopen(type=m)
      xrun addpath {%mpath}
    endif
  endif

  xput tc_ymat tc_ystarmat tc_xmat tc_wmat tc_cmat tc_cvec
  xput tc_nopttargs tc_noptinstrus tc_nevl tc_ndrv tc_itmax tc_conv
  xput tc_dmat tc_dmat1 tc_dmat2 tc_dmat_short tc_damp

```

```

        if %iii = "r" then
            xrun source("tcoc_r.R")
        else
            xrun tcoc_m
        endif

        xget tc_xmat
        xget tc_ymat
        xget tc_iter
        !tc_iter = tc_iter

        if %xclose = "yes" then
            xclose
        endif

        delete(noerr) tc_ystarmat tc_dmat tc_wmat tc_cmat tc_cvec
        delete(noerr) tc_nopttargs tc_noptinstrus tc_nevl tc_ndrv tc_conv
        delete(noerr) tc_damp tc_dmat1 tc_dmat2 tc_dmatshort

        endif
    endif

' *****
' examine linearized solution

    if tc_iter >= tc_itmax then
        @uiprompt("iteration limit reached:  time-consistent iterations did not converge")
        'stop
    endif

' write instrument and target values back into their corresponding series
    smpl %evlstart {%evlstart} + !nevl + !ndrv - 2
    mtos(tc_xmat, _$opt_instrus)
    mtos(tc_ymat, _$opt_targs_sols)

' value of loss function for linearized solution
    smpl %evlstart %evlend
    matrix _$opt_targ_vec = @vec(@convert(_$opt_targs_sols))
    matrix _$opt_gap_vec = _$opt_targ_vec - _$opt_des_vec
    !optloss_lin = @sum(@transpose(_$opt_gap_vec)*_ $opt_wt_mat*_ $opt_gap_vec)
    stom(_$opt_targs_sols, tc_ymat_lin)

' simulate EViews model
    smpl %mcestart %mceend
    call mcz_sim(" ")
    smpl %evlstart %evlend

```

```

stom(_$opt_targs_sols,tc_ymat_ev)
!maxdiff = @max(@abs(tc_ymat_ev-tc_ymat_lin))
matrix _$opt_targ_vec = @vec(@convert(_$opt_targs_sols))
matrix _$opt_gap_vec = _$opt_targ_vec - _$opt_des_vec
!optloss_ev = @sum(@transpose(_$opt_gap_vec)*_$opt_wt_mat*$_$opt_gap_vec)

!tciter = tc_iter
mce_opt_text.append time-consistent solution
mce_opt_text.append --- backward-induction iterations = !tciter
mce_opt_text.append --- TC damping factor = !tcdamp
mce_opt_text.append --- linearized solution loss = !optloss_lin
mce_opt_text.append --- EViews solution loss = !optloss_ev
mce_opt_text.append --- max diff btwn target vars in linear and EViews sols = !maxdiff

delete(noerr) tc_ymat tc_xmat tc_iter tc_ymat_lin tc_ymat_ev
delete(noerr) tc_itmax tc_stat

endsub

```

Defines:

`mcz_opt_tc`, used in chunks 16 and 57.

Uses `dateshift` 9, `mcz_opt_deriv` 81, and `mcz_sim` 35.

A.2.18 compute derivatives of loss function targets wrt instruments

```

81  <compute derivatives of loss function targets wrt instruments 81>≡ (177)
    subroutine mcz_opt_deriv
    ,
    ' requires that the following be defined
    ,
    ' !optshow, mce_opt_stats, %evlstart, %evlend
    ' $_opt_targs_sols, !noptinstrus, $_opt_instrus, !drv,
    ' $_opt_ptrb_mat, $_opt_der_mat

    ' This subroutine computes the derivatives of the loss function targets wrt the instruments

    smpl @all

    !optshow_bac = !optshow
    !optshow = 3

    'make a copy of the current values of the mce instruments;
    'they will be restored after each derivative is computed
    smpl %mcestart %mceend
    stom($_mce_instrus,$_instrus_bac)

    'do not reset terminal conditions when computing derivatives
    %set_terminal = "no"

    smpl %evlstart %evlend
    matrix $_opt_targ_vec = @vec(@convert($_opt_targs_sols))

    '*****
    'derivatives loop
    statusline computing instrument derivatives at optimization iteration !optry
    for !zi = 1 to !noptinstrus
        %instru_name = $_opt_instrus.@seriesname(!zi)
        for !zj = 1 to !ndrv
            !perturbit = $_opt_ptrb_mat(!zj,!zi)
            smpl %mcestart + !zj-1 %mcestart + !zj-1
            {%instru_name} = {%instru_name} + !perturbit
            call mcz_sim(" ")
            smpl %evlstart %evlend
            matrix $_opt_targ_dvec = @vec(@convert($_opt_targs_sols))
            matplace($_opt_der_mat,@transpose($_opt_targ_dvec-$_opt_targ_vec)/!perturbit,(!zi-1)*!)
            smpl %mcestart + !zj-1 %mcestart + !zj-1
            {%instru_name} = {%instru_name} - !perturbit
        'restore the original values of the add factors on

```

```
      'the z variable equations
      smpl %mcestart %mceend
      mtos(_$instrus_bac,$_mce_instrus)
      next
    next

    smpl %mcestart %mceend

    !optshow = !optshow_bac

  endsub
```

Defines:

`mcz_opt_deriv`, used in chunks 64, 68, and 73.
Uses `mcz_sim` 35.

A.2.19 solve model consistent expectations model

```

83  <solve model consistent expectations model 83>≡ (177)
    subroutine mcz_opt_solve

' This subroutine first sets the instrument values, based on the current
' optimal direction and choice of step size, and then solves the model

    statusline optimization solution, iteration !opttry
    smpl %mcestart %mceend

'compute instrument values based on current direction and step size
if !opttry > 0 then
    mce_opt_stats(!opttry+3,3) = !opt_step
    smpl %drvstart %drvend
    for !i = 1 to !noptinstrus
        vector _$_temp_adj = @subextract(_$_opt_direction,(!i-1)*!ndrv+1,1,!i*!ndrv,1)
        mtos(_$_temp_adj,_$_temp_ser)
        %y1 = _$_opt_instrus.@seriesname(!i)
        {%y1} = {%y1} - (!opt_step-!opt_step_prev)*_$_temp_ser
    next
endif

'solve model
if !opttry = 0 and %terminal = "yes" then
    call mcz_sim("terminal")
    %terminal = "no"
else
    call mcz_sim(" ")
endif

if !opttry = 0 then
    {%$_mod_b}.makegroup _$_opt_targs_sols {%opt_targ_names}
endif

'compute value of loss function
smpl %evlstart %evlend
matrix _$_opt_targ_vec = @vec(@convert(_$_opt_targs_sols))
matrix _$_opt_gap_vec = _$_opt_targ_vec - _$_opt_des_vec
!optloss = @sum(@transpose(_$_opt_gap_vec)*_$_opt_wt_mat*_$_opt_gap_vec)
setcell(mce_opt_stats,!opttry+3,1,!opttry,0)
mce_opt_stats(!opttry+3,2) = !optloss
_$_opt_loss_vec(!opttry+1) = !optloss

if !optshow = 2 then
    show mce_opt_stats

```

`endif``endsub`

Defines:

`mcz_opt_solve`, used in chunks 64 and 68.Uses `mcz_sim` 35.

A.2.20 convert mcz inequality constraints

```

85  <convert mcz inequality constraints 85>≡ (177)
    subroutine local mcz_constraints(svector sss, table ctable,string cvarnames)

' This subroutine converts the inequality constraint text into a table

' subroutine arguments:
'
' inputs:
'
'   sss                svector of constraint text
'
' outputs:
'
'   $_opt_cnstr_tab    table of constraint variables and coefficients
'   $_opt_cnstr_vars   string of names of variables in constraints

' *****

!nconstraints = @rows(sss)
scalar loc
scalar sign

for !i = 1 to !nconstraints
    %rrr = sss(!i)

' *****
' *****
' separate the left and right sides of each constraint

!kk1 = @instr(%rrr,">=")
if !kk1 = 0 then
    @uiprompt("each constraint must contain an ">=" term")
    stop
endif
%rrrl = @left(%rrr,!kk1-1)
if @isempty(%rrrl) = 1 then
    @uiprompt("each constraint must contain terms to the left of ">="")
    stop
endif
%rrrr = @mid(%rrr,!kk1+2)
if @isempty(%rrrr) = 1 then
    @uiprompt("each constraint must contain a value to the right of ">="")
    stop

```

```

endif

' *****
' *****
' process the left side of each constraint
'
' 1. split %rrrl into individual terms (based on + and - characters);
'    the main challenge concerns the first term, which may or may not
'    have a leading + or - attached
' 2. then split each term into 2 parts (based on * character)
' 3. then identify which part is coefficient and which is variable

!zz = 0      'flag to indicate end of left hand side of a constraint
!nterms = 1
vector(100) split_locs = 0
vector(100) split_signs = 0
%rrrlx = %rrrl

' find boundaries and signs of each term
while !zz = 0

' first term: determine whether its sign is explicit or implicit
if !nterms = 1 then
    call find_next_delimit(%rrrlx,loc,sign)
    if loc = 0 then                'implicit leading sign
        split_locs(!nterms) = 0
        split_signs(!nterms) = 1
    else
        %rrrl1 = @left(%rrrlx,loc-1)
        if @isempty(%rrrl1) = 1 then    'explicit leading sign
            split_locs(!nterms) = loc
            split_signs(!nterms) = sign
            %rrrlx = @mid(%rrrlx,loc+1)
        else
            'implicit leading sign
            split_locs(!nterms) = 0
            split_signs(!nterms) = 1
        endif
    endif
endif
endif

call find_next_delimit(%rrrlx,loc,sign)
if loc = 0 then                'last term
    split_locs(!nterms+1) = @length(%rrrlx)
    !zz = 1
else

```

```

        split_locs(!nterms+1) = loc
        split_signs(!nterms+1) = sign
        %rrrlx = @mid(%rrrlx,loc+1)
    endif

    !nterms = !nterms + 1
wend

for !k = 2 to !nterms
    split_locs(!k) = split_locs(!k-1) + split_locs(!k)
next

!nterms = !nterms - 1

' parse each term into coefficient times variable (they must
' appear in that order, although coefficients = 1 can be
' omitted), and store coefficient and variable in ctable
for !k = 1 to !nterms
    %term = @mid(%rrrl,split_locs(!k)+1,split_locs(!k+1)-split_locs(!k)-1)
    if @isempty(%term) = 1 then
        @uiprompt("term is empty:  illegal constraint specification")
        stop
    else
        !ii = @instr(%term,"*")
        if !ii > 0 then
            %coef = @left(%term,!ii-1)
            !coef = split_signs(!k) * @val(%coef)
            %var = @mid(%term,!ii+1)
        else
            !coef = split_signs(!k)
            %var = %term
        endif
        %var = @trim(%var)
    endif

    ctable(!i,2*!k) = %var
    ctable(!i,2*!k+1) = !coef
    cvarnames = cvarnames + " " + %var
next

' *****
' *****
' process the right side of each constraint

```

```

        ctable(!i,2*!nterms+2) = @val(%rrrr)
        ctable(!i,1) = !nterms

    next

    cvarnames = @wunique(cvarnames)

endsub

```

Defines:

`mcz.constraints`, used in chunk 57.

Uses `find_next_delimit` 89.

A.2.21 shift left

```

88   $\langle \textit{shift left 88} \rangle \equiv$  (177)
    subroutine local shiftleft(vector abc, scalar nshift)

        !rows = @rows(abc)
        vector v1 = @subextract(abc,1,1,nshift,1)
        vector v2 = @subextract(abc,nshift+1,1,!rows,1)
        matplace(abc,v2,1)
        matplace(abc,v1,!rows - nshift + 1)
    endsub

```

endsub

Defines:

`shiftleft`, used in chunk 35.

A.2.22 find next delimiter

89 $\langle \textit{find next delimiter 89} \rangle \equiv$ (177)

```

subroutine local find_next_delimit(string %instring, scalar loc, scalar sign)

    !ttp = @instr(%instring, "+")
    !ttm = @instr(%instring, "-")
    if !ttp = 0 or !ttm = 0 then
        loc = !ttp + !ttm
        if !ttp = 0 and !ttm = 0 then
            sign = 1
        else
            sign = (!ttp > 0) - (!ttm > 0)
        endif
    else
        loc = !ttp*(!ttp < !ttm) + !ttm*(!ttm < !ttp)
        sign = (!ttp < !ttm) - (!ttm < !ttp)
    endif

endsub

Defines:
    find_next.delimit, used in chunk 85.

```

A.2.23 load frbus with transformed subsidiary model

```

90  <load frbus with transformed subsidiary model 90>≡ (177)
      subroutine mce_load_frbus(string frbus_opts)

      ' take two corresponding FRB/US models (eg, stdver and pfver) and make
      ' the transformations needed to the second model so that the pair of
      ' models can be used with the mcz_solve_subs programs

      ' parameters
      '   required:  mce_vars, mod_b, mod_f, path_b, path_f
      '   optional:  allbut, only
      ,

      !allbut = 0
      !only = 0

      frbus_opts = @lower(frbus_opts)
      frbus_opts = @replace(frbus_opts," ","")
      frbus_opts = @replace(frbus_opts,","," ")
      frbus_opts = " " + frbus_opts + " "

      if @isempty(frbus_opts) = 0 then
        call mcz_equalopt("mce_vars",frbus_opts)
        if @len(%temp)>0 then
          %mce_vars = @lower({%temp})
        else
          @uiprompt("Error:  mce_load_frbus sub requires the mce_vars argument")
          stop
        endif
        call mcz_equalopt("mod_b",frbus_opts)
        if @len(%temp)>0 then
          %mod_b = {%temp}
        else
          @uiprompt("Error:  mce_load_frbus sub requires the mod_b argument")
          stop
        endif
        call mcz_equalopt("mod_f",frbus_opts)
        if @len(%temp)>0 then
          %mod_f = {%temp}
        else
          @uiprompt("Error:  mce_load_frbus sub requires the mod_f argument")
          stop
        endif
        call mcz_equalopt("path_b",frbus_opts)
        if @len(%temp)>0 then

```

```

    %path_b = {%temp}
  else
    @uiprompt("Error: mce_load_frbus sub requires the path_b argument")
    stop
  endif
call mcz_equalopt("path_f",frbus_opts)
if @len(%temp)>0 then
  %path_f = {%temp}
  else
    @uiprompt("Error: mce_load_frbus sub requires the path_f argument")
    stop
  endif
call mcz_equalopt("allbut",frbus_opts)
if @len(%temp)>0 then
  !allbut = 1
  %allbut = {%temp}
  else
    call mcz_equalopt("only",frbus_opts)
    if @len(%temp)>0 then
      !only = 1
      %only = {%temp}
    endif
  endif
endif
endif
endif

```

' Added so that users may ask directly for a group of forward-looking equations instead of
' passing in a list that may change in the future.

' Model consistent Asset Pricing

```
%s_mcap = " zdivgr zgap05 zgap10 zpi10f zpic30 zrff10 zrff5 zgap30 zrff30 zpi10 zpib5 zpic58"
```

' Wages and prices

```
%s_wp = " zpidxfe zpieci "
```

' Others - all PAC expectations

```
%s_other = " zecd zeco zeh zgapc2 zlhp zpi5 zvpd zvps zvpi zxnfbd zxnfbz zxnfbz ziyh ziyh ziyh"
```

```

if %mce_vars = "-all" then
  %mce_vars = %s_mcap + %s_wp + %s_other
endif
if %mce_vars = "-mcap" then
  %mce_vars = %s_mcap
endif
if %mce_vars = "-wp" then
  %mce_vars = %s_wp
endif

```

```

if %mce_vars = "-mcap+wp" then
    %mce_vars = %s_mcap + %s_wp
endif
if @left(%mce_vars, 7) = "-allbut" then
    %tmp = %mce_vars
    %s_remove = @replace(%tmp, "-allbut", "")
    %mce_vars = @wnotin(%s_mcap + %s_wp + %s_other, %s_remove)
endif
string zvar_list = %mce_vars
%zvars = %mce_vars

' backward-looking model

ld_frbus_cfs(modelname=%mod_b,modelpath=%path_b)
if !allbut = 1 then
    %tmp = "allbut " + %allbut
    ld_some_eqs(modelname=%mod_b,modelpath=%path_b,eqnames=%tmp)
endif
if !only = 1 then
    %tmp = %only
    ld_some_eqs(modelname=%mod_b,modelpath=%path_b,eqnames=%tmp)
endif
if !allbut = 0 and !only = 0 then
    ld_frbus_eqs(modelname=%mod_b,modelpath=%path_b)
endif

' model with mce equations and errors

ld_mce_eqs(pfname=%mod_f,pfpath=%path_f,mcename=%mod_f,mceeqs=%mce_vars)
ld_mce_cfs(pfname=%mod_f,pfpath=%path_f,mceeqs=%mce_vars)

!nmcevars = @wcount(%mce_vars)
%evars = @wcross("e",%mce_vars)
for !i = 1 to !nmcevars
    %tmp = @word(%mce_vars,!i)
    %tmpw = " w" + @mid(%tmp,2)
    if %tmp <> "zyh" and %tmp <> "zyhp" and %tmp <> "zyht" then
        %tmp1 = @word(%evars,!i) + "=" + @word(%mce_vars,!i) + "-" + %tmpw
    else
        %tmp1 = @word(%evars,!i) + "= log(" + @word(%mce_vars,!i) + "/" + %tmpw + ")"
    endif
    {%mod_f}.append {%tmp1}
next

endsub

```

Defines:

`mce_load_frbus`, used in chunks 16, 120, 130, 136, and 145.

Uses `ld_frbus_cfs` 99a 99b, `ld_frbus_eqs` 100b 101, `ld_mce_cfs` 102b 103, `ld_mce_eqs` 105 106,
`ld_some_eqs` 110 111, and `mcz.equalopt` 32a.

A.2.24 create wage and expectation variables in forward looking model

```

94  <create wage and expectation variables in forward looking model 94>≡      (177)
      subroutine make_frbus_mcevars(string frbus_mcevars)

      ' Create data for the w and e variables in the operational forward looking
      ' model over the workfile sample currently in effect
      ,
      ' The input string contains the names of the expectations variables that
      ' are to have MC solutions

      %mce_vars = frbus_mcevars

      ' The user may ask directly for a group of forward-looking equations instead of
      ' passing in a list that may change in the future.

      ' Model consistent Asset Pricing
      %s_mcap = " zdivgr zgap05 zgap10 zpi10f zpic30 zrff10 zrff5 zgap30 zrff30 zpi10 zp

      ' Wages and prices
      %s_wp = " zpicxfe zpieci "

      ' Others - all PAC expectations
      %s_other = " zecd zeco zeh zgapc2 zlhp zpi5 zvpd zvps zvpi zxnfbd zxnfbz zxnfbz zyl

      if %mce_vars = "-all" then
        %mce_vars = %s_mcap + %s_wp + %s_other
      endif
      if %mce_vars = "-mcap" then
        %mce_vars = %s_mcap
      endif
      if %mce_vars = "-wp" then
        %mce_vars = %s_wp
      endif
      if %mce_vars = "-mcap+wp" then
        %mce_vars = %s_mcap + %s_wp
      endif
      if @left(%mce_vars, 7) = "-allbut" then
        %tmp = %mce_vars
        %s_remove = @replace(%tmp, "-allbut", "")
        %mce_vars = @wnotin(%s_mcap + %s_wp + %s_other, %s_remove)
      endif

      ' Data for extra variables associated with MC expectations

```

```
smpl @all
for !i = 1 to @wcount(%mce_vars)
    %tmp = @word(%mce_vars,!i)
    %wtmp = "w" + @mid(%tmp,2)
    %wtmp_aerr = %wtmp + "_aerr"
    %etmp = "e" + @word(%mce_vars,!i)
    series {%wtmp} = {%tmp}
    series {%wtmp_aerr} = 0
    series {%etmp} = 0
next
```

endsub

Defines:

`make_frbus_mcevars`, used in chunks 16, 120, 130, 136, and 145.

Appendix B

File Contents

Notice that I've had to change underscores in file and directory names when I create them with noweb. I use the "setup.sh" bash script to create the directory structure, and the "check.sh" bash script to compare the renamed Eview files to the originals. I have to ignore white space in the comparison because the original source files are not consistent with using either Microsoft or Linux line ending characters.

B.1 setup.sh

```
97  <setup.sh 97>≡  
    mkdir -p srcEview/{frbus.package,mce.solve.package,state.space.package}  
    mkdir -p srcEview/frbus.package/{addins,programs,subs}  
    mkdir -p srcEview/frbus.package/addins/{ld.frbus.cfs,ld.frbus.eq,ld.mce.cfs,ld.mce.eq,ld.some
```

This code is written to file setup.sh.

B.2 check.sh

98 \langle check.sh 98 $\rangle \equiv$

```
cfile=frbus_package/addins/ld_frbus_cfs/ld_frbus_cfs.prg; ofile=srcEview/frbus.package/ld_frbus_cfs.prg;
cfile=frbus_package/addins/ld_frbus_eqs/ld_frbus_eqs.prg; ofile=srcEview/frbus.package/ld_frbus_eqs.prg;
cfile=frbus_package/addins/ld_mce_cfs/ld_mce_cfs.prg; ofile=srcEview/frbus.package/ld_mce_cfs.prg;
cfile=frbus_package/addins/ld_mce_eqs/ld_mce_eqs.prg; ofile=srcEview/frbus.package/ld_mce_eqs.prg;
cfile=frbus_package/addins/ld_some_eqs/ld_some_eqs.prg; ofile=srcEview/frbus.package/ld_some_eqs.prg;
cfile=frbus_package/addins/ld_varinfo/ld_varinfo.prg; ofile=srcEview/frbus.package/ld_varinfo.prg;
cfile=frbus_package/addins/regadd.prg; ofile=srcEview/frbus.package/addins/regadd.prg;
cfile=frbus_package/programs/example1.prg; ofile=srcEview/frbus.package/programs/example1.prg;
cfile=frbus_package/programs/example2.prg; ofile=srcEview/frbus.package/programs/example2.prg;
cfile=frbus_package/programs/example3.prg; ofile=srcEview/frbus.package/programs/example3.prg;
cfile=frbus_package/programs/example4.prg; ofile=srcEview/frbus.package/programs/example4.prg;
cfile=frbus_package/programs/ocpolicy.prg; ofile=srcEview/frbus.package/programs/ocpolicy.prg;
cfile=frbus_package/programs/pings.prg; ofile=srcEview/frbus.package/programs/pings.prg;
cfile=frbus_package/programs/plot_resids.prg; ofile=srcEview/frbus.package/programs/plot_resids.prg;
cfile=frbus_package/programs/stochsim.prg; ofile=srcEview/frbus.package/programs/stochsim.prg;
cfile=frbus_package/subs/master_library.prg; ofile=srcEview/frbus.package/subs/master_library.prg;
cfile=frbus_package/subs/mce_solve_library.prg; ofile=srcEview/frbus.package/subs/mce_solve_library.prg;
cfile=mce_solve_package/example1.prg; ofile=srcEview/mce.solve.package/example1.prg;
cfile=mce_solve_package/example2.prg; ofile=srcEview/mce.solve.package/example2.prg;
cfile=mce_solve_package/example3.prg; ofile=srcEview/mce.solve.package/example3.prg;
cfile=mce_solve_package/example4.prg; ofile=srcEview/mce.solve.package/example4.prg;
cfile=mce_solve_package/example5.prg; ofile=srcEview/mce.solve.package/example5.prg;
cfile=mce_solve_package/mce_solve_library.prg; ofile=srcEview/mce.solve.package/mce_solve_library.prg;
cfile=state_space_package/data_transformations.prg; ofile=srcEview/state.space.package/data_transformations.prg;
cfile=state_space_package/estimation_code.prg; ofile=srcEview/state.space.package/estimation_code.prg;
cfile=state_space_package/frbus_supply_estimation.prg; ofile=srcEview/state.space.package/frbus_supply_estimation.prg;
cfile=state_space_package/frbus_supply_filter.prg; ofile=srcEview/state.space.package/frbus_supply_filter.prg;
cfile=state_space_package/initial_values.prg; ofile=srcEview/state.space.package/initial_values.prg;
```

This code is written to file check.sh.

Uses data_transformations 198, example1 117 181, example2 120 184, example3 124 187, example4 130 191, example5 194, frbus_supply_filter 208, frbus_supply_estimation 205, initial_values 211, ld_frbus_cfs 99a 99b, ld_frbus_eqs 100b 101, ld_mce_cfs 102b 103, ld_mce_eqs 105 106, ld_some_eqs 110 111, ld_varinfo 114 115, mce_solve_library 176 197, ocpolicy 136, pings 144, plot_resids 159, regadd 116b, and stochsim 163.

B.3 srcEview/frbus.package/addins/ld.frbus.cfs/ld.frbus.cfs.prg

99a $\langle \text{srcEview/frbus.package/addins/ld.frbus.cfs/ld.frbus.cfs.prg 99a} \rangle \equiv$
 $\langle \text{load frbus coefficients 99b} \rangle$

$\langle \text{load frbus coefficients call 100a} \rangle$

This code is written to file `srcEview/frbus.package/addins/ld.frbus.cfs/ld.frbus.cfs.prg`.

Defines:

`ld_frbus_cfs`, used in chunks 90, 98, 100a, 116b, 117, 124, 160, and 164.

99b $\langle \text{load frbus coefficients 99b} \rangle \equiv$ (99a)
`subroutine ld_frbus_cfs(string %mname, string %mpath)`

'Load coefficients for frbus version %mname from a text file in directory %mpath
 'that has been previously created by the script eq_docs2evIEWS.

```
%cpath = %mpath + %mname + "_coeffs.txt"
delete(noerr) coefpath
text coefpath
coefpath.append(file) %cpath
svector coefpathv = coefpath.@svectornb

for !i = 1 to 900
  svector cofname = @wsplit(coefpathv(!i))
  %y1 = cofname(1)
  if @left(%y1,6) = "theend" then
    exitloop
  endif
  %y2 = cofname(2)
  %y3 = cofname(3)
  coef({%y2}) {%y1}
  {%y1}.fill {%y3}
next
```

`endsub`

Defines:

`ld_frbus_cfs`, used in chunks 90, 98, 100a, 116b, 117, 124, 160, and 164.

B.4 srcEview/frbus.package/addins/ld.frbus.eqs/ld.frbus.eqs.p

⟨load frbus equations call 102a⟩

Defines:

`ld_frbus_eqs`, used in chunks 90, 98, 102a, 116b, 117, 124, 160, and 164.

```

101  <load frbus equations 101>≡ (100b)
      subroutine ld_frbus_eqs(string %mname, string %mpath)

      'Create evIEWS model %mname and load it with the equations for frbus version
      '%mname that are in a text file in directory %mpath that was previously created
      'by the script eq_docs2evIEWS.

      %epath = %mpath + %mname
      if @fileexist(%epath) <> 1 then
        %epath = %mpath + %mname + "_eqs.txt"
      endif
      delete(noerr) eqtext
      text eqtext
      eqtext.append(file) %epath
      svector eqtextv = eqtext.@svectornb

      model {%mname}

      !eqnum = 0
      %eqcode = " "

      for !i = 1 to 3000
        %y = eqtextv(!i)
        if @isempty(%y) = 0 then
          'string is not blank
          if @left(%y,6) = "theend" then
            %x = @replace(%eqcode," _"," ")
            {%mname}.append {%x}
            exitloop
          endif

          !k = @instr(%y,":")
          if !k > 0 then
            'string contains the start of a new equation
            !eqnum = !eqnum + 1
            if !eqnum > 0 then
              %x = @replace(%eqcode," _"," ")
              {%mname}.append {%x}
            endif
            %eqcode = @mid(%y,!k+1)

          else
            'string contains the continuation of an equation
            %eqcode = %eqcode + %y
          endif
        endif
      endfor
    
```

```

        endif
    next

```

```

endsub

```

Defines:

ld_frbus_eqs, used in chunks 90, 98, 102a, 116b, 117, 124, 160, and 164.

```

102a  <load frbus equations call 102a>≡ (100b)
      if @len(@option(1)) < 1 or @len(@option(2)) < 1 then
          @uiprompt("Error: ld_frbus_eqs requires model name and model path")
          stop
      endif

      %temp = @equaloption("modelname")
      if @len(%temp)>0 then
          %mname = %temp
      endif
      %temp = @equaloption("modelpath")
      if @len(%temp)>0 then
          %mpath = %temp
      endif

      call ld_frbus_eqs(%mname, %mpath)

```

Uses ld_frbus_eqs 100b 101.

B.5 srcEview/frbus.package/addins/ld.mce.cfs/ld.mce.cfs.prg

```

102b  <srcEview/frbus.package/addins/ld.mce.cfs/ld.mce.cfs.prg 102b>≡
      <load mce coefficients 103>

      <load mce coefficients call 104>

```

This code is written to file srcEview/frbus.package/addins/ld.mce.cfs/ld.mce.cfs.prg.

Defines:

ld_mce_cfs, used in chunks 90, 98, 104, and 116b.

```

103  <load mce coefficients 103>≡ (102b)
      subroutine ld_mce_cfs(string %pfname, string %pfpath, string %mceeqs)

      'This subroutine is used for setting up the particular type of frbus simulations
      'with model-consistent expectations in which a separate model is created
      'containing only those expectations equations chosen to have model-consistent
      'solutions. For those expectations variables, the initial z character is replaced
      'with w.

      'Load coefficients for frbus version %pfname from a text file in directory %pfpath
      'that has been previously created by the script eq_docs2evIEWS. Only those
      'coefficient vectors whose names are in the string %mceeqs are stored.

      %cpath = %pfpath + %pfname + "_coeffs.txt"
      delete(noerr) coefpathpv
      text coefpathpv
      coefpathpv.append(file) %cpath
      svector coefpathv = coefpathpv.@svectornb

      for !i = 1 to 900
        svector cofname = @wsplit(coefpathv(!i))
        %y1 = cofname(1)
        if @left(%y1,6) = "theend" then
          exitloop
        endif
        %y2 = cofname(2)
        %y3 = cofname(3)
        for !j = 1 to @wcount(%mceeqs)
          %z = @lower(@word(%mceeqs,!j))
          %z1 = @mid(%y1,3)
          if %z = %z1 then
            %y1 = @replace(%y1,"z","w")
            coef({%y2}) {%y1}
            {%y1}.fill {%y3}
            exitloop
          endif
        next
      next

      endsub

Defines:
      ld_mce_cfs, used in chunks 90, 98, 104, and 116b.

```

```

104  (load mce coefficients call 104)≡ (102b)
      if @len(@option(1)) < 1 or @len(@option(2)) < 1 or @len(@option(3)) < 1 then
        @uiprompt("Error: ld_mce_cfs requires four parameters: pf model name, pf model path,
        stop
      endif

      %temp = @equaloption("pfname")
      if @len(%temp)>0 then
        %pfname = %temp
      endif
      %temp = @equaloption("pfpath")
      if @len(%temp)>0 then
        %pfpath = %temp
      endif
      %temp = @equaloption("mceeqs")
      if @len(%temp)>0 then
        %mceeqs = %temp
      endif
      '      ' Added so that users may ask directly for a group of forward-looking equations
      '      ' passing in a list that may change in the future.
      '      ' Model consistent Asset Pricing
      '      %s_mcap = " zdivgr zgap05 zgap10 zpi10f zpic30 zrff10 zrff5 zgap30 zrff30 zpi10
      '      ' Wages and prices
      '      %s_wp = " zpicxfe zpieci "
      '      ' Others - all PAC expectations
      '      %s_other = " zecd zeco zeh zgapc2 zlhp zpi5 zpi10 zpib5 zvpd zvps zvpi zxnfbd z
      '      if %mceeqs = "-all" then
      '        %mceeqs = %s_mcap + %s_wp + %s_other
      '      endif
      '      if %mceeqs = "-mcap" then
      '        %mceeqs = %s_mcap
      '      endif
      '      if %mceeqs = "-wp" then
      '        %mceeqs = %s_wp
      '      endif
      '      if %mceeqs = "-mcap+wp" then
      '        %mceeqs = %s_mcap + %s_wp
      '      endif
      '      if @left(%mceeqs, 7) = "-allbut" then
      '        %tmp = %mceeqs
      '        %s_remove = @replace(%tmp, "-allbut", "")
      '        %mceeqs = @wnotin(%s_mcap + %s_wp + %s_other, %s_remove)
      '      endif
      '      string zvar_list = %mceeqs
      '      scalar nzvars = @wcount(%mceeqs)
      '      group zvars {%mceeqs}

```



```
call ld_mce_cfs(%pfname, %pfpath, %mceeqs)
```

Uses `ld_mce_cfs` 102b 103.

B.6 `srcEview/frbus.package/addins/ld.mce.eqs/ld.mce.eqs.prg`

105 `<srcEview/frbus.package/addins/ld.mce.eqs/ld.mce.eqs.prg 105>≡`

```
<load mce equations 106>
```

```
<load mce equations call 109>
```

This code is written to file `srcEview/frbus.package/addins/ld.mce.eqs/ld.mce.eqs.prg`.

Defines:

```
ld_mce_eqs, used in chunks 90, 98, 109, and 116b.
```

```

106  (load mce equations 106)≡ (105)
      subroutine ld_mce_eqs(string %pfname, string %pfpath, string %mcename, string %mceeqs

      'This subroutine is used for setting up the particular type of frbus simulations
      'with model-consistent expectations in which a separate model is created
      'containing only those expectations equations chosen to have model-consistent
      'solutions. For those expectations variables, the initial z character is replaced
      'with w.

      'Create evIEWS model %mcename and load it with the equations for frbus version
      '%pfname that are in a text file in directory %pfpath that was previously created
      'by the script eq_docs2evIEWS. Only those equations whose names are in the
      'string %mceeqs are included.

      ' revised 2/11/13 to add a check that version %pfname contains an equation for each
      ' name in %mceeqs

      %epath = %pfpath + %pfname + "_eqs.txt"
      delete(noerr) eqtextp
      text eqtextp
      eqtextp.append(file) %epath
      svector eqtextpv = eqtextp.@svectornb

      %coded = " "

      model {%mcename}

      !eqnum = 0
      %eqnew = " "
      %eqold = " "
      %eqcode = " "
      %eqcodeold = " "

      for !i = 1 to 3000

          %appendit = "no"
          %exitloop = "no"
          %y = eqtextpv(!i)
          if @isempty(%y) = 0 then
              'string is not blank
              if @left(%y,6) = "theend" then
                  'string contains end-of-file flag
                  %appendit = "yes"
                  %exitloop = "yes"
                  %eqold = %eqnew

```

```

        %eqcodeold = %eqcode
    endif

    !k = @instr(%y,":")
    if !k > 0 then
        'string contains the start of a new equation
        %appendit = "yes"
        !eqnum = !eqnum + 1
        if !eqnum > 0 then
            %eqold = %eqnew
            %eqcodeold = %eqcode
        endif
        %eqnew = @left(%y,!k-1)
        %eqcode = @mid(%y,!k+1)
    else
        'string contains the continuation of an equation
        %eqcode = %eqcode + %y
    endif

    if %appendit = "yes" then
        'add equation to model only if it is one with mce expectations
        for !j = 1 to @wcount(%mceeqs)
            %z = @lower(@word(%mceeqs,!j))
            if %z = %eqold then
                %x = @replace(%eqcodeold," _"," ")
                %x = @replace(%x,"z","w")
                {%mcename}.append {%x}
                %coded = %coded + " " + %z
            exitloop
            endif
        next
    endif

    if %exitloop = "yes" then
        exitloop
    endif

    endif
next

' check that model %mcename contains an equation for each variable in %mceeqs
!j = @wcount(%mceeqs)
!k = @wcount(%coded)
if !j <> !k then
    %z = @wnotin(@lower(%mceeqs),@lower(%coded))
    %errstring = "Error in ld_mce_eqs addin: Model " + %mcename

```

```
%errstring = %errstring + " does not contain equation(s) for variable(s): " + %  
%errstring = %errstring + ". Execution stopped."  
@uiprompt(%errstring)  
stop  
endif
```

```
endsub
```

Defines:

`ld.mce_eqs`, used in chunks 90, 98, 109, and 116b.

```

109  <load mce equations call 109>= (105)
      if @len(@option(1)) < 1 or @len(@option(2)) < 1 or @len(@option(3)) < 1 or @len(@option(4)) < 1
        @uiprompt("Error: ld_mce_eqs requires five parameters: pf model name, pf model path, mce mod
        stop
      endif

      %temp = @equaloption("pfname")
      if @len(%temp)>0 then
        %pfname = %temp
      endif
      %temp = @equaloption("pfpath")
      if @len(%temp)>0 then
        %pfpath = %temp
      endif
      %temp = @equaloption("mcename")
      if @len(%temp)>0 then
        %mcename = %temp
      endif
      %temp = @equaloption("mceeqs")
      if @len(%temp) >0 then
        %mceeqs = %temp
      endif
      ' ' Added so that users may ask directly for a group of forward-looking equations instead of
      ' ' passing in a list that may change in the future.
      ' ' Model consistent Asset Pricing
      ' %s_mcap = " zdivgr zgap05 zgap10 zpi10f zpic30 zrff10 zrff5 zgap30 zrff30 zpi10 zpib5 "
      ' ' Wages and prices
      ' %s_wp = " zpicxfe zpieci "
      ' ' Others - all PAC expectations
      ' %s_other = " zecd zeco zeh zgapc2 zlhp zpi5 zpi10 zpib5 zvpd zvps zvpi zxbd zxbs zxbi zyh
      ' if %mceeqs = "-all" then
      '   %mceeqs = %s_mcap + %s_wp + %s_other
      ' endif
      ' if %mceeqs = "-mcap" then
      '   %mceeqs = %s_mcap
      ' endif
      ' if %mceeqs = "-wp" then
      '   %mceeqs = %s_wp
      ' endif
      ' if %mceeqs = "-mcap+wp" then
      '   %mceeqs = %s_mcap + %s_wp
      ' endif
      ' if @left(%mceeqs, 7) = "-allbut" then
      '   %tmp = %mceeqs
      '   %s_remove = @replace(%tmp, "-allbut", "")
      '   %mceeqs = @wnotin(%s_mcap + %s_wp + %s_other, %s_remove)

```

```

'    endif
'    string zvar_list = %mceeqs
'    scalar nzvars = @wcount(%mceeqs)
'    group zvars {%mceeqs}

call ld_mce_eqs(%pfname, %pfpath, %mcename, %mceeqs)

```

Uses `ld_mce_eqs` 105 106.

B.7 srcEview/frbus.package/addins/ld.some.eq/ld.some.eq.pr

```

110 <srcEview/frbus.package/addins/ld.some.eq/ld.some.eq.prg 110>≡
    <load some equations 111>
    <load some equations call 113>

```

This code is written to file `srcEview/frbus.package/addins/ld.some.eq/ld.some.eq.prg`.

Defines:

`ld.some.eq`, used in chunks 90, 98, 113, and 116b.

```

111  <load some equations 111>≡ (110)
      subroutine ld_some_eqs(string %mname, string %mpath, string %eqnames)

      ' Create evIEWS model %mname and load it with selected equations for frbus version
      ' %mname that are in a text file in directory %mpath that was previously created
      ' by the script eq_docs2evIEWS. Only those equations whose names match or do not match the
      ' equation names in the string %eqnames are included. If the first "word" in %eqnames is
      ' "allbut", then all equations but those listed will be included.

      '*****
      'parse equation string and put equation names in a table

      %allbut = "no"

      string zlist = " "
      zlist = zlist + %eqnames
      zlist = @trim(zlist)
      if @isempty(zlist) = 1 then
        @uiprompt("Error: input string to subroutine load_selected_equtions is empty!!")
        stop
      endif

      if @wfind(@upper(zlist),"ALLBUT") = 1 then
        %allbut = "yes"
        zlist = @wdrop(zlist,"ALLBUT")
        zlist = @wdrop(zlist,"allbut")
      endif

      '*****
      'parse equation file

      %epath = %mpath + %mname + "_eqs.txt"
      delete(noerr) eqtext
      text eqtext
      eqtext.append(file) %epath
      svector eqtextv = eqtext.@svectornb

      model {%mname}

      !eqnum = 0
      %eqnew = " "
      %eqold = " "
      %eqcode = " "
      %eqcodeold = " "

```

```

for !i = 1 to 3000
  %appendit = "no"
  %exitloop = "no"
  %y = eqtextv(!i)
  if @isempty(%y) = 0 then
    'string is not blank
    if @left(%y,6) = "theend" then
      %appendit = "yes"
      %exitloop = "yes"
      %eqold = %eqnew
      %eqcodeold = %eqcode
    endif

    !k = @instr(%y,":")
    if !k > 0 then
      'string contains the start of a new equation
      %appendit = "yes"
      !eqnum = !eqnum + 1
      if !eqnum > 0 then
        %eqold = %eqnew
        %eqcodeold = %eqcode
      endif
      %eqnew = @left(%y,!k-1)
      %eqcode = @mid(%y,!k+1)

    else
      'string contains the continuation of an equation
      %eqcode = %eqcode + %y
    endif

    if %appendit = "yes" then
      'check whether equation should be added to model
      !zswitch = 0
      for !j = 1 to @wcount(zlist)
        %z = @lower(@word(zlist,!j))
        if %z = %eqold then
          !zswitch = 1
        endif
        if %allbut = "no" and !zswitch = 1 then
          %x = @replace(%eqcodeold," _"," ")
          {%mname}.append {%x}
          exitloop
        endif
        if %allbut = "yes" and !j = @wcount(zlist) and !zswitch = 0 then
          %x = @replace(%eqcodeold," _"," ")

```



```

        {%mname}.append {%x}
      endif
    next
  endif

  if %exitloop = "yes" then
    exitloop
  endif

endif
next

```

endsub

Defines:

ld_some_eqs, used in chunks 90, 98, 113, and 116b.

```

113 <load some equations call 113>≡ (110)
    if @len(@option(1)) < 1 or @len(@option(2)) < 1 or @len(@option(3)) < 1 then
      @uiprompt("Error: ld_some_eqs requires model name and model path and eqnames")
      stop
    endif

    %temp = @equaloption("modelname")
    if @len(%temp)>0 then
      %mname = %temp
    endif
    %temp = @equaloption("modelpath")
    if @len(%temp)>0 then
      %mpath = %temp
    endif
    %temp = @equaloption("eqnames")
    if @len(%temp)>0 then
      %eqnames = %temp
    endif

    call ld_some_eqs(%mname, %mpath, %eqnames)
  Uses ld_some_eqs 110 111.

```

B.8 `srcEview/frbus.package/addins/ld.varinfo/ld.varinfo.prg`

114 $\langle \textit{srcEview/frbus.package/addins/ld.varinfo/ld.varinfo.prg} \text{ 114} \rangle \equiv$
 $\langle \textit{load variable information} \text{ 115} \rangle$

$\langle \textit{load variable information call} \text{ 116a} \rangle$

This code is written to file `srcEview/frbus.package/addins/ld.varinfo/ld.varinfo.prg`.

Defines:

`ld.varinfo`, used in chunks 98, 116, 160, and 164.

115 *(load variable information 115)*≡ (114)
 subroutine ld_varinfo(string %pathname)

 'Load varinfo information from %pathname in strings

```

text vinfo_text
vinfo_text.append(file) %pathname
svector varinfo = vinfo_text.@svectornb
string vinfo_vname = " "
string vinfo_vtype = " "
string vinfo_vrule = " "
string vinfo_sector = " "
string vinfo_stoch = " "
string vinfo_decomp = " "
for !i = 1 to 900
  %y1 = varinfo(!i)
  !ss = @instr(%y1," ")
  %vname = @mid(%y1,!ss+1,8)
  %vname = @rtrim(%vname)
  %vtype = @mid(%y1,!ss+107,1)
  %vrule = @mid(%y1,!ss+109,1)
  if %vrule = " " then
    %vrule = "0"
  endif
  %sector = @mid(%y1,!ss+120,1)
  if %sector = " " then
    %sector = "0"
  endif
  %stoch = @mid(%y1,!ss+128,2)
  %decomp = @mid(%y1,!ss+135,2)
  if %vname = "ZZZBLANK" then
    scalar vinfo_size = !i-1
    exitloop
  endif
  vinfo_vname = vinfo_vname + " " + %vname
  vinfo_vtype = vinfo_vtype + " " + %vtype
  vinfo_vrule = vinfo_vrule + " " + %vrule
  vinfo_sector = vinfo_sector + " " + %sector
  vinfo_stoch = vinfo_stoch + " " + %stoch
  vinfo_decomp = vinfo_decomp + " " + %decomp
next

```

endsub

Defines:

ld_varinfo, used in chunks 98, 116, 160, and 164.

116a *<load variable information call 116a>*≡ (114)

```

    if @len(@option(1)) < 1 then
        @uiprompt("Error: ld_varinfo requires varinfo path")
        stop
    endif

    %temp = @equaloption("pathname")
    if @len(%temp)>0 then
        %pathname = %temp
    endif

    call ld_varinfo(%pathname)

```

Uses `ld_varinfo` 114 115.

B.9 srcEview/frbus.package/addins/regadd.prg

116b *<srcEview/frbus.package/addins/regadd.prg 116b>*≡

```

    addin(proc="ld_frbus_eqs",desc="Load FRB/US equations") ".\ld_frbus_eqs/ld_frbus_eqs.prg"
    addin(proc="ld_frbus_cfs",desc="Load FRB/US coefficients") ".\ld_frbus_cfs/ld_frbus_cfs.prg"
    addin(proc="ld_mce_eqs",desc="Load MCE equations") ".\ld_mce_eqs/ld_mce_eqs.prg"
    addin(proc="ld_mce_cfs",desc="Load MCE coefficients") ".\ld_mce_cfs/ld_mce_cfs.prg"
    addin(proc="ld_some_eqs",desc="Load some FRB/US equations") ".\ld_some_eqs/ld_some_eqs.prg"
    addin(proc="ld_varinfo",desc="Load variable information file") ".\ld_varinfo/ld_varinfo.prg"

```

This code is written to file `srcEview/frbus.package/addins/regadd.prg`.
 Defines:
 `regadd`, used in chunk 98.
 Uses `ld_frbus_cfs` 99a 99b, `ld_frbus_eqs` 100b 101, `ld_mce_cfs` 102b 103, `ld_mce_eqs` 105 106,
 `ld_some_eqs` 110 111, and `ld_varinfo` 114 115.

B.10 srcEview/frbus.package/programs/example1.prg

```

117 <srcEview/frbus.package/programs/example1.prg 117>≡
    ' Program for simple simulation under VAR expectations
    ,
    ' See FRB/US Simulation Basics document for information about
    ' this program

    ' *****
    ' Initial filename and parameter settings
    ' *****

    ' Subroutines
    include ../subs/master_library

    ' Workfile
    %wfstart = "1975q1"
    %wfend = "2030q4"
    %mainpage = "main"
    wfcreate(wf=aaa,page={%mainpage}) q {%wfstart} {%wfend}

    ' FRB/US model name and location
    %varmod = "stdver"
    %varpath = "../mods/"

    ' Input database
    %dbin = "../data/longbase"

    ' Simulation range
    %simstart = "2020q1"
    %simend = "2025q4"

    ' *****
    ' Retrieve data, model equations and coefficients, set
    ' policy options, and compute tracking residuals
    ' *****

    ' Load equations and coefficients
    ld_frbus_eqs(modelname=%varmod,modelpath=%varpath)
    ld_frbus_cfs(modelname=%varmod,modelpath=%varpath)

    ' Load data
    dbopen %dbin as longbase
    fetch(d=longbase) *

```

```

' Set monetary policy rule
  smpl @all
  call set_mp("dmpintay")

' Turn off zero bound and policy thresholds; hold policymaker's
' perceived equilibrium real interest rate constant
  smpl @all
  dmptrsh = 0
  rffmin = -9999
  drstar = 0

' Set fiscal policy
  smpl @all
  call set_fp("dfpsrp")

' Set _aerr variables to zero
  smpl @all
  {%varmod}.makegroup(a,n) endog @endog
  call groupnew("endog","_aerr")
  call group2zero("endog_aerr")

' Standard solution options
  {%varmod}.solveopt(o=b,g=12,z=1e-12)

' Assign baseline tracking add factors
  %suftrk = "_0"
  smpl %simstart %simend
  {%varmod}.addassign @all
  {%varmod}.addinit(v=n) @all
  {%varmod}.scenario(n,a={%suftrk}) "track"
  {%varmod}.solve
  scalar mm = @max(@abs(xgap{%suftrk}-xgap))
  if mm > .0001 then
    statusline dynamic tracking simulation failed for {%varmod}
    stop
  endif

' *****
' Simulate a shock to monetary policy rule
' *****

  %sufsim = "_1"
  {%varmod}.scenario(n,a={%sufsim}) "sim"

  smpl %simstart %simstart
  rffintay_aerr = rffintay_aerr + 1

```

```

smpl %simstart %simend
{%varmod}.solve

'*****
' Make a graph
'*****

smpl %simstart %simend
series zero = 0
series d_rff = rff{%sufsim} - rff
series d_rg10 = rg10{%sufsim} - rg10
series d_lur = lur{%sufsim} - lur
series d_pic4 = pic4{%sufsim} - pic4

graph fig1a.line zero d_rff
fig1a.addtext(t,just(c),font("arial",12)) Federal Funds Rate
fig1a.legend -display

graph fig1b.line zero d_rg10
fig1b.addtext(t,just(c),font("arial",12)) 10-Year Treasury Yield
fig1b.legend -display

graph fig1c.line zero d_lur
fig1c.addtext(t,just(c),font("arial",12)) Unemployment Rate
fig1c.legend -display

graph fig1d.line zero d_pic4
fig1d.addtext(t,just(c),font("arial",12)) Inflation Rate (4-Quarter)
fig1d.legend -display

graph fig1.merge fig1a fig1b fig1c fig1d
fig1.addtext(t,just(c),font("Arial",16)) Macroeconomic Effects of Funds Rate Perturbation\r(V
fig1.align(2,1,1.25)
show fig1

```

This code is written to file srcEview/frbus.package/programs/example1.prg.

Defines:

example1, used in chunk 98.

Uses group2zero 11a, groupnew 12, ld_frbus_cfs 99a 99b, ld_frbus_eqs 100b 101, set_fp 14,
and set_mp 15a.

B.11 srcEview/frbus.package/programs/example2.prg

```

120  <srcEview/frbus.package/programs/example2.prg 120>≡
      ' Program for simple simulation with MCE expectations
      ,
      ' The switch variables %mcvars_wp and %mcvars_all control whether
      ' the assumption of MC expectations extends beyond the financial
      ' sector

      ' See the Simulation Basics document for information about
      ' this program

      ' *****
      ' Initial filename and parameter settings
      ' *****

      ' Subroutines
      include ../subs/master_library
      include ../subs/mce_solve_library

      ' Workfile
      %wfstart = "1975q1"
      %wfend = "2100q4"
      %mainpage = "main"
      wfcreate(wf=aaa,page={%mainpage}) q {%wfstart} {%wfend}

      ' FRB/US model names and locations
      %varmod = "stdver"
      %varpath = "../mods/"
      %mcmmod = "pfver"
      %mcepath = "../mods/"

      ' Input database
      %dbin = "../data/longbase"

      ' Simulation range
      %simstart = "2020q1"
      %simend = "2069q4"

      ' *****
      ' Retrieve data, model equations and coefficients, set
      ' policy options, and compute tracking residuals
      ' *****

```



```

' Specify MC expectations variables
%mcvars_wp = "yes"
%mcvars_all = "no"

' MCE asset pricing
%zvars = "zdivgr zgap05 zgap10 zgap30 zrff5 zrff10 zrff30 zpi10 zpi10f zpic30 zpib5 zpic58 "

' MCE elsewhere
if %mcvars_wp = "yes" and %mcvars_all = "no" then
    %zvars = %zvars + "zpicxfe zpieci "
endif
if %mcvars_all = "yes" then
    %zvars = %zvars + "zpicxfe zpieci "
    %zvars = %zvars + "zecd zeco zeh zgapc2 zlhp zpi5 zvpd zvpi zvps zxbd zxbi zxbs zyh zyhp zy"
endif

' Load equations and coefficients
call mce_load_frbus("mce_vars=%zvars,mod_b=%varmod,path_b=%varpath,mod_f=%mcemod,path_f=%mcep

' Load data
dbopen %dbin as longdata
fetch(d=longdata) *

' Data for extra variables associated with MC expectations
smpl @all
call make_frbus_mcevars(%zvars)

' Set monetary policy
smpl @all
call set_mp("dmpintay")

' Turn off zero bound and policy thresholds; hold policymaker's
' perceived equilibrium real interest rate constant
smpl @all
dmptrsh = 0
rffmin = -9999
drstar = 0

' Set fiscal policy
smpl @all
call set_fp("dfpsrp")

' Set _aerr variables to zero
smpl @all
{%varmod}.makegroup(a,n) endog @endog
call groupnew("endog","_aerr")

```

```

call group2zero("endog_aerr")

' Standard solution options
{%varmod}.solveopt(o=b,g=12,z=1e-12)
{%mccmod}.solveopt(o=b,g=12,z=1e-12)

' Assign baseline tracking add factors
%suftrk = "_0"
smpl %simstart %simend
{%varmod}.addassign @all
{%varmod}.addinit(v=n) @all
{%varmod}.scenario(n,a={%suftrk}) "track"
{%varmod}.solve
scalar mm = @max(@abs(xgap{%suftrk}-xgap))
if mm > .0001 then
    statusline dynamic tracking simulation failed for {%varmod}
    stop
endif
{%mccmod}.addassign @all
{%mccmod}.addinit(v=n) @all

' *****
' Simulate a monetary policy shock
' *****

%sufsim = "_1"
{%varmod}.scenario(n,a={%sufsim}) "sim"
{%mccmod}.scenario(n,a={%sufsim}) "sim"

smpl %simstart %simstart
rffintay_a = rffintay_a + 1

%modstr = "mod_b=%varmod,mod_f=%mccmod,mce_vars=%zvars"
%algstr = "meth=qnewton"
%simstr = "type=single"
smpl %simstart %simend
call mce_run(%modstr,%algstr,%simstr)

' *****
' Make a graph
' *****

smpl %simstart %simstart + 39
series zero = 0
series d_rff = rff{%sufsim} - rff

```

```

series d_rg10 = rg10{%sufsim} - rg10
series d_lur = lur{%sufsim} - lur
series d_pic4 = pic4{%sufsim} - pic4

graph fig1a.line zero d_rff
fig1a.addtext(t,just(c),font("arial",12)) Federal Funds Rate
fig1a.legend -display

graph fig1b.line zero d_rg10
fig1b.addtext(t,just(c),font("arial",12)) 10-Year Treasury Yield
fig1b.legend -display

graph fig1c.line zero d_lur
fig1c.addtext(t,just(c),font("arial",12)) Unemployment Rate
fig1c.legend -display

graph fig1d.line zero d_pic4
fig1d.addtext(t,just(c),font("arial",12)) Inflation Rate (4-Quarter)
fig1d.legend -display

graph fig1.merge fig1a fig1b fig1c fig1d
%title = " Macroeconomic Effects of Funds Rate Shock\r"
if %mcvars_wp = "no" and %mcvars_all = "no" then
    %title = %title + "(MC Expectations in Asset Pricing)"
endif
if %mcvars_wp = "yes" and %mcvars_all = "no" then
    %title = %title + "(MC Expectations in Asset Pricing and Price-Wage Setting)"
endif
if %mcvars_all = "yes" then
    %title = %title + "(MC Expectations in All Sectors)"
endif
fig1.addtext(t,just(c),font("Arial",16)) {%title}
fig1.align(2,1,1.25)
show fig1

```

This code is written to file `srcEview/frbus.package/programs/example2.prg`.

Defines:

`example2`, used in chunk 98.

Uses `group2zero` 11a, `groupnew` 12, `make_frbus_mcevars` 94, `mce_load_frbus` 90, `mce_run` 18, `mce_solve_library` 176 197, `set_fp` 14, and `set_mp` 15a.

B.12 srcEview/frbus.package/programs/example3.prg

```

124  <srcEview/frbus.package/programs/example3.prg 124>≡
      ' Program for simulation under VAR expectations that illustrates how
      ' to set the monetary policy options that impose the zero lower bound
      ' on the funds rate and delay the liftoff of the funds rate from the
      ' ZLB until either the unemployment rate falls below a threshold or
      ' inflation rises above a threshold.
      '
      ' See FRB/US Simulation Basics document for general information about
      ' this program.

      ' Additional notes:

      ' 1. The scenario involves a set of negative aggregate demand
      ' shocks and a positive risk premium shock that start in 2003q3,
      ' when the baseline (historical) funds rate is about one percent.
      ' The shocks are equal to the equation errors actually observed
      ' in the four quarters starting in 2008q4.

      ' 2. To impose the ZLB set %zb = "yes" (rather than "no")

      ' 3. To impose the policy liftoff threshold conditions set both
      ' %zb = "yes" and %threshold = "yes". For illustrative purposes
      ' and reflecting the baseline conditions in 2003 and the years
      ' that immediately follow, the inflation threshold is set to 3.0
      ' and the unemployment threshold is set to 7.0, subject to the
      ' the adjustments described next.

      ' 4. Because the threshold conditions only make sense once the ZLB is
      ' binding, unemployment is above its threshold level (lurtrsh),
      ' and inflation is below its threshold (pitrsh), which is not the
      ' case in the initial simulation quarters, the program turns on the
      ' threshold code (using dmptrsh) in the 5th simulation quarter,
      ' at which point these conditions hold. In addition, for the threshold
      ' code to work properly, the endogenous switch variable dmptr must be
      ' zero in the quarter prior to the quarter in which the threshold code is
      ' turned on. This is accomplished by setting the baseline data on dmptr
      ' to zero and by setting the unemployment and inflation thresholds
      ' (lurtrsh, pitrsh) to values in the first four simulation quarters that
      ' would not flip the dmptr switch to one.

      ' 4. Choose one of the five available policy rules by setting
      ' %policy to one of rffintay, rfftay, rfftlr, rffalt, or rffgen.

      ' 5. If neither the ZLB or thresholds are imposed, the monetary policy

```

```

' equations have baseline-tracking adds and the simulation is
' a standard deviations-from-baseline exercise.

' 6. If either the ZLB or thresholds are imposed, the add factors on
' monetary policy equations are set to zero after the tracking adds
' are computed so that the ZLB and threshold conditions are based on the
' actual simulated outcomes for the funds rate and inflation and unemployment,
' not their deviations from baseline.

' *****
' Initial filename and parameter settings
' *****

' Subroutines
include ../subs/master_library

' Workfile
%wfstart = "1975q1"
%wfend = "2012q4"
%mainpage = "main"
wfcreate(wf=aaa,page={%mainpage}) q {%wfstart} {%wfend}

' FRB/US model name and location
%varmod = "stdver"
%varpath = "../mods/"

' Input database
%dbin = "../data/longbase"

' Simulation range
%simstart = "2003q3"
%simend = "2008q2"

' Policy
%zb = "yes"
%threshold = "yes"
%policy = "rfftay"

' *****
' Retrieve data, model equations and coefficients, set
' policy options, and compute tracking residuals
' *****

' Load equations and coefficients
ld_frbus_eqs(modelname=%varmod,modelpath=%varpath)

```

```

ld_frbus_cfs(modelname=%varmod,modelpath=%varpath)

' Load data
dbopen %dbin as longbase
fetch(d=longbase) *

' Set monetary policy rule
smpl @all
%policydmp = @replace(%policy,"rff","dmp")
call set_mp(%policydmp)

' Set ZLB
if %zb = "yes" then
    rffmin = .125
else
    rffmin = -9999
endif

' Set threshold variables
if %threshold = "yes" then
    if %zb = "no" then
        @uiprompt("When policy thresholds are imposed, the zero bound must also be imposed")
        stop
    endif
    smpl @all
    call dateshift(%simstart,%quarter4,3)
' thresholds (dmptrsh and dmptr) not active in first 4 qtrs
    smpl %simstart - 1 %quarter4
    dmptrsh = 0
    lurtrsh = -9999
    pitrsh = 9999
    dmptr = 0
' thresholds (dmptrsh and dmptr) active starting in qtr 5
    smpl %quarter4 + 1 %simend
    dmptrsh = 1
    lurtrsh = 7.0
    pitrsh = 3.0
    smpl @all
    else
    smpl @all
    dmptrsh = 0
    endif

smpl @all
drstar = 0

```

```

' Set fiscal policy
  smpl @all
  call set_fp("dfpsrp")

' Set _aerr variables to zero
  smpl @all
  {%varmod}.makegroup(a,n) endog @endog
  call groupnew("endog","_aerr")
  call group2zero("endog_aerr")

' Standard solution options
  {%varmod}.solveopt(o=b,g=12,z=1e-12)

' Assign baseline tracking add factors
  %suftrk = "_0"
  smpl %simstart 2012q4
  {%varmod}.addassign @all
  {%varmod}.addinit(v=n) @all
  {%varmod}.scenario(n,a={%suftrk}) "track"
  {%varmod}.solve
  scalar mm = @max(@abs(xgap{%suftrk}-xgap))
  if mm > .0001 then
    statusline dynamic tracking simulation failed for {%varmod}
    stop
  endif

' Set monetary policy add factors to zero when ZLB or threshold are
' imposed

if %zb = "yes" then
  smpl @all
  {%policy}_a = 0
  rffrule_a = 0
  rffe_a = 0
  if %threshold = "yes" then
    dmptpi_a = 0
    dmptlur_a = 0
    dmptmax_a = 0
    dmptr_a = 0
  endif
endif

' *****
' Simulation
' *****

```

```

%sufsim = "_1"
{%varmod}.scenario(n,a={%sufsim}) "sim"

' shock values are taken from equation residuals for 2008q4-2009q3
eco_a.fill(o=%simstart) -.006, -.006, -.011, -.001
ecd_a.fill(o=%simstart) -.091, -.018, -.021, .029
eh_a.fill(o=%simstart) -.076, -.078, -.040, .073
epd_a.fill(o=%simstart) -.096, -.062, .014, .032
eps_a.fill(o=%simstart) -.018, -.046, -.036, -.017
rbbbp_a.fill(o=%simstart) 2.70, 0.38, -0.89, -1.35

smpl %simstart %simend
{%varmod}.solve

'*****
' Make a graph
'*****

smpl %simstart %simend

graph fig1a.line rff rff{%sufsim}
fig1a.addtext(t,just(c),font("arial",12)) Federal Funds Rate
fig1a.legend -display

graph fig1b.line rg10 rg10{%sufsim}
fig1b.addtext(t,just(c),font("arial",12)) 10-Year Treasury Yield
fig1b.legend -display

graph fig1c.line lur lur{%sufsim}
fig1c.addtext(t,just(c),font("arial",12)) Unemployment Rate
fig1c.legend -display

graph fig1d.line pic4 pic4{%sufsim}
fig1d.addtext(t,just(c),font("arial",12)) Inflation Rate (4-Quarter)
fig1d.legend -display

%title = "Macroeconomic Effects of Negative AD Shock\r(VAR Expectations"
%title = %title + "; Policy = " + %policy + ")"
if %zb = "yes" and %threshold = "no" then
    %title = %title + "\r(ZLB Imposed)"
endif
if %zb = "yes" and %threshold = "yes" then
    %title = %title + "\r(ZLB and Thresholds Imposed)"
endif

```



```
graph fig1.merge fig1a fig1b fig1c fig1d
fig1.addtext(t,just(c),font("Arial",16)) {%title}
fig1.addtext(b,just(c),font("Arial",16)) Blue:  Actual;  Red:  Simulated
fig1.align(2,1,1.25)
show fig1
```

This code is written to file `srcEview/frbus.package/programs/example3.prg`.

Defines:

`example3`, used in chunk 98.

Uses `dateshift` 9, `group2zero` 11a, `groupnew` 12, `ld_frbus_cfs` 99a 99b,
`ld_frbus_eqs` 100b 101, `set_fp` 14, and `set_mp` 15a.

B.13 srcEview/frbus.package/programs/example4.prg

```

130  <srcEview/frbus.package/programs/example4.prg 130>≡
      ' This MCE example program illustrates:
      ,
      ' 1. how to use a monetary policy rule that is not one of the policy
      '   alternatives included in FRB/US;
      ' 2. how to add new MCE expectations variables;
      ' 3. how to drop one of the regular FRB/US equations as part of the process
      '   of loading the model

      ' Most of the code needed illustrate these issues is located between
      ' the "start of new code" and "end of new code" comments below

      ' The switch variables %mcvars_wp and %mcvars_all control whether
      ' the assumption of MC expectations extends beyond the financial
      ' sector

      ' See the Simulation Basics document for information about
      ' this program

      ' *****
      ' Initial filename and parameter settings
      ' *****

      ' Subroutines
      include ../subs/master_library
      include ../subs/mce_solve_library

      ' Workfile
      %wfstart = "1975q1"
      %wfend = "2100q4"
      %mainpage = "main"
      wfcreate(wf=aaa,page={%mainpage}) q {%wfstart} {%wfend}

      ' FRB/US model names and locations
      %varmod = "stdver"
      %varpath = "../mods/"
      %mcmmod = "pfver"
      %mcepath = "../mods/"

      ' Input database
      %dbin = "../data/longbase"

      ' Simulation range

```

```

%simstart = "2010q1"
%simend   = "2069q4"

' *****
' Retrieve data, model equations and coefficients, set
' policy options, and compute tracking residuals
' *****

' Specify MC expectations variables
%mcvars_wp = "no"
%mcvars_all = "yes"

' MCE asset pricing
%zvars = "zdivgr zgap05 zgap10 zgap30 zrff5 zrff10 zrff30 zpi10 zpi10f zpic30 zpib5 zpic58 "

' MCE elsewhere
if %mcvars_wp = "yes" and %mcvars_all = "no" then
    %zvars = %zvars + "zpicxfe zpieci "
endif
if %mcvars_all = "yes" then
    %zvars = %zvars + "zpicxfe zpieci "
    %zvars = %zvars + "zecd zeco zeh zgapc2 zlhp zpi5 zvpd zvpi zvps zxbd zxbi zxbs zyh zyhp zy"
endif

' Load equations and coefficients

' drop one of the FRB/US monetary policy rule equations (rffgen) so that it can be
' replaced below with an alternative rule
%allbut = "rffgen"
call mce_load_frbus("mce_vars=%zvars,mod_b=%varmod,path_b=%varpath,mod_f=%mcemod,path_f=%mcepath")

' Load data
dbopen %dbin as longdata
fetch(d=longdata) *

' *****
' *****
' start of new code (aside from the change above to mce_load_frbus and
' the change below to the call to set_mp)

' Code a first-difference interest rate rule as rffgen. The first-difference rule depends on
' the expected output gap three quarters ahead (zgap3) and on expected 4-qtr inflation three
' quarters ahead (zpic43). The name of each new expectation must start with a "z".
{%varmod}.append rffgen-rffgen_aerr = rffe(-1) + .5*(zpic43-pitarg) + .5*(zgap3-zgap3(-4))

```

```

' Add the MCE definitions of zgap3 and zpic43 to the forward-looking model,
' noting that the MCE names of these variables must start with a "w" rather than a "z"
  {%mccmod}.append wgap3-wgap3_aerr = xgap2(3)
  {%mccmod}.append wpic43-wpic43_aerr = picx4(3)
' Add expectations error equations to the MCE model
  {%mccmod}.append ezgap3 = zgap3-wgap3
  {%mccmod}.append ezipic43 = zpic43-wpic43

' Add to the backward-looking model simple equations for the new expectations variables
' Technically, these equations should be the appropriate VAR expectations formulas, but
' because in this program these expectations will always be MCE, the form of their
' backward-looking identities is not very important.
  {%varmod}.append zgap3-zgap3_aerr = .5*xgap2(-1)
  {%varmod}.append zpic43-zpic43_aerr = .5*picx4(-1)+.5*ptr(-1)

' Add the new MCE variables to the %zvars string
  %zvars = %zvars + " zgap3 zpic43"

' Define baseline values of the new expectations variables
  smpl @all
  series zgap3 = xgap2(3)
  series zpic43 = picx4(3)

' Make sure that the baseline data for rffgen matches the baseline data for rffe
  rffgen = rffe

' end of new code
' *****
' *****

' Data for extra variables associated with MC expectations
  smpl @all
  call make_frbus_mcevars(%zvars)

' Set monetary policy to use the first-difference policy rule (coded as rffgen)
  smpl @all
  call set_mp("dmpgen")

' Turn off zero bound and policy thresholds; hold policymaker's
' perceived equilibrium real interest rate constant
  smpl @all
  dmptrsh = 0
  rffmin = -9999
  drstar = 0

' Set fiscal policy

```

```

    smpl @all
    call set_fp("dfpsrp")

' Set _aerr variables to zero
    smpl @all
    {%varmod}.makegroup(a,n) endog @endog
    call groupnew("endog","_aerr")
    call group2zero("endog_aerr")

' Standard solution options
    {%varmod}.solveopt(o=b,g=12,z=1e-12)
    {%mccmod}.solveopt(o=b,g=12,z=1e-12)

' Assign baseline tracking add factors
    %suftrk = "_0"
    smpl %simstart %simend
    {%varmod}.addassign @all
    {%varmod}.addinit(v=n) @all
    {%varmod}.scenario(n,a={%suftrk}) "track"
    {%varmod}.solve
    scalar mm = @max(@abs(xgap{%suftrk}-xgap))
    if mm > .0001 then
        statusline dynamic tracking simulation failed for {%varmod}
        stop
    endif
    {%mccmod}.addassign @all
    {%mccmod}.addinit(v=n) @all

' *****
' Simulate the effects of a one-percent consumption shock
' *****

    %sufsim = "_1"
    {%varmod}.scenario(n,a={%sufsim}) "sim"
    {%mccmod}.scenario(n,a={%sufsim}) "sim"

    smpl %simstart %simstart
    eco_a = eco_a + .01

    %modstr = "mod_b=%varmod,mod_f=%mccmod,mce_vars=%zvars"
    %algstr = "meth=qnewton"
    %simstr = "type=single"
    smpl %simstart %simend
    call mce_run(%modstr,%algstr,%simstr)

```

```

'*****
' Make a graph
'*****

    smpl %simstart %simstart + 39
    series zero = 0
    series d_rff = rff{%sufsim} - rff
    series d_rg10 = rg10{%sufsim} - rg10
    series d_lur = lur{%sufsim} - lur
    series d_pic4 = pic4{%sufsim} - pic4

    graph fig1a.line zero d_rff
    fig1a.addtext(t,just(c),font("arial",12)) Federal Funds Rate
    fig1a.legend -display

    graph fig1b.line zero d_rg10
    fig1b.addtext(t,just(c),font("arial",12)) 10-Year Treasury Yield
    fig1b.legend -display

    graph fig1c.line zero d_lur
    fig1c.addtext(t,just(c),font("arial",12)) Unemployment Rate
    fig1c.legend -display

    graph fig1d.line zero d_pic4
    fig1d.addtext(t,just(c),font("arial",12)) Inflation Rate (4-Quarter)
    fig1d.legend -display

    graph fig1.merge fig1a fig1b fig1c fig1d
    %title = " Macroeconomic Effects of a Shock to Consumption\r"
    if %mcvars_wp = "no" and %mcvars_all = "no" then
        %title = %title + "(MC Expectations in Asset Pricing)"
    endif
    if %mcvars_wp = "yes" and %mcvars_all = "no" then
        %title = %title + "(MC Expectations in Asset Pricing and Price-Wage Setting)"
    endif
    if %mcvars_all = "yes" then
        %title = %title + "(MC Expectations in All Sectors)"
    endif
    fig1.addtext(t,just(c),font("Arial",16)) {%title}
    fig1.align(2,1,1.25)
    show fig1

```

This code is written to file srcEview/frbus.package/programs/example4.prg.

Defines:

example4, used in chunk 98.

June 19, 2016

frbuseview.nw 135

Uses `group2zero` 11a, `groupnew` 12, `make_frbus_mcevars` 94, `mce_load_frbus` 90, `mce_run` 18, `mce_solve_library` 176 197, `set_fp` 14, and `set_mp` 15a.

B.14 srcEview/frbus.package/programs/ocpolicy.prg

```

136 <srcEview/frbus.package/programs/ocpolicy.prg 136>≡
    ' Routine to simulate how the SEP baseline forecast would change if
    ' policymakers commit to a path for the federal funds rate that is
    ' determined by optimal-control (OC) techniques to minimize a
    ' quadratic loss function.
    ,
    ' Detailed information on the mechanics of the OC algorithm and the
    ' various required and optional parameters that set up and guide its
    ' execution is available in the MCE Solve Users Guide in the
    ' documentation directory. Most relevant is the part of section 5
    ' that describes the "opt" simulation type as well as table 7.

    ' As specified below, the loss function penalizes equally weighted
    ' squared deviations of the unemployment rate from the natural rate,
    ' squared deviations of inflation from a 2 percent, and squared
    ' quarterly changes in the funds rate.

    ' In the SEP baseline, agents with model-consistent (MC) expectations
    ' are initially assumed to project that the funds rate will follow the
    ' baseline path and to set their baseline expectations
    ' accordingly. At the start of the optimal control simulation, however,
    ' these agents immediately and fully revise their expectations to be
    ' consistent with the revision to the funds rate path that occurs under
    ' optimal control -- that is, agents have rational expectations and
    ' announced policy actions are completely credible.
    ,
    ' The experiment can be run with the zero lower bound (ZLB) imposed
    ' (%zerobound = "yes") or not imposed (%zerobound = "no"). When
    ' the ZLB is imposed, a penalty term is added to the loss function.

    ' *****
    ' Initial filename and parameter settings
    ' *****

    ' Subroutines
    include ../subs/master_library
    include ../subs/mce_solve_library

    ' Workfile
    %wfstart = "1975q1"
    %wfend = "2100q4"
    %mainpage = "main"
    wfcreate(wf=aaa,page={%mainpage}) q {%wfstart} {%wfend}

```



```

' FRB/US model names and locations
%varmod = "stdver"
%varpath = "../mods/"
%mcemod = "pfver"
%ncepath = "../mods/"

' Input database
%dbin = "../data/longbase"

' Simulation range
%simstart = "2014q4"
%simend = "2070q4"

' Primary loss function parameters: The value of the policy instrument
' is chosen optimally from %drvstart to %drvend (60 qtrs) to minimize
' the loss function from %evlstart to %evlend (80 qtrs). The three
' arguments of the period loss function are weighted by the the
' weight parameters and over time losses are discounted at the rate
' %discount
%evlstart = %simstart
%drvstart = %simstart
call dateshift(%evlstart,%evlend,79)
call dateshift(%drvstart,%drvend,59)
%discount = ".99"
%u_weight = "1.0"
%p_weight = "1.0"
%r_weight = "1.0"

' Optionally impose the zero lower bound
%zerobound = "yes"

' *****
' Retrieve data, model equations and coefficients, set
' policy options, and compute tracking residuals
' *****

' Specify MC expectations variables
%mcvars_wp = "yes"
%mcvars_all = "no"

' MCE asset pricing
%zvars = "zdivgr zgap05 zgap10 zgap30 zrff5 zrff10 zrff30 zpi10 zpi10f zpic30 zpib5 zpic58 "

' MCE elsewhere
if %mcvars_wp = "yes" and %mcvars_all = "no" then

```

```

    %zvars = %zvars + "zpicxfe zpieci "
    endif
    if %mcvars_all = "yes" then
        %zvars = %zvars + "zpicxfe zpieci "
        %zvars = %zvars + "zecd zeco zeh zgapc2 zlhp zpi5 zvpd zvpi zvps zxbd zxnfbz zxnfbz"
    endif

' Load equations and coefficients
call mce_load_frbus("mce_vars=%zvars,mod_b=%varmod,path_b=%varpath,mod_f=%mcemod,p

' Add a ugap equation
{%varmod}.append ugap - ugap_aerr = lur - lurnat

' Load data
dbopen %dbin as longdata
fetch(d=longdata) *

' Define SEP-consistent ustar and ugap series; this step is needed because
' the baseline value of lurnat may not be fully SEP-consistent in the
' short-to-medium ruun
smpl @all
series ustar = lurnat
smpl %simstart 2025q4
ustar = 5.35
smpl @all
series ugap = lur-ustar
series ugap_aerr = 0

' Data for extra variables associated with MC expectations
smpl @all
call make_frbus_mcevars(%zvars)

' Set monetary policy option (the residual on the equation of
' the chosen option is the OC policy instrument)
smpl @all
call set_mp("dmptay")

' Initially turn off zero lower bound; if %zerobound = "yes", it will be
' imposed below by adding a penalty term to the loss function.
smpl @all
rffmin = -9999

' Turn off policy thresholds
dmptrsh = 0

' Let the perceived equilibrium real interest rate vary
drstar = 1

```

```

' Set fiscal policy so that it is exogenous for first 20 qtrs and then
' turns on debt targeting rule
  smpl %simstart %simstart + 19
  call set_fp("dfpex")
  smpl %simstart + 20 %simend
  call set_fp("dfpdbl")

' Set _aerr variables to zero
  smpl @all
  {%varmod}.makegroup(a,n) endog @endog
  call groupnew("endog","_aerr")
  call group2zero("endog_aerr")

' Standard solution options
  {%varmod}.solveopt(o=b,g=12,z=1e-12)
  {%mccmod}.solveopt(o=b,g=12,z=1e-12)

' Assign baseline tracking add factors
  %suftrk = "_0"
  smpl %simstart %simend
  {%varmod}.addassign @all
  {%varmod}.addinit(v=n) @all
  {%varmod}.scenario(n,a={%suftrk}) "track"
  {%varmod}.solve
  scalar mm = @max(@abs(xgap{%suftrk}-xgap))
  if mm > .0001 then
    statusline dynamic tracking simulation failed for {%varmod}
    stop
  endif
  {%mccmod}.addassign @all
  {%mccmod}.addinit(v=n) @all

' *****
' optimal policy setup
' *****

' The policy instrument is a time varying constant in the equation
' for the selected policy rule
  group opt_instrus rfftay_aerr

' Loss function variables (unemployment gap, 4-qtr PCE inflation,
' and the first difference of the federal funds rate)
  group opt_targs ugap pic4 delrff

```

```

' The desired paths of the loss function variables are specified in
' series with "_t" suffix.
smpl @all
series ugap_t = 0
series delrff_t = 0
series pic4_t = 2.0

' The weights on the loss function arguments are specified in
' series with "_w" suffix.
series ugap_w = @val(%u_weight)
series pic4_w = @val(%p_weight)
series delrff_w = @val(%r_weight)

!discount = @val(%discount)
smpl %simstart+1 %simend
ugap_w = !discount * ugap_w(-1)
pic4_w = !discount * pic4_w(-1)
delrff_w = !discount * delrff_w(-1)

' Zero bound penalty function
if %zerobound = "yes" then
  {%varmod}.append penalty - penalty_aerr = _
    @recode(rff<(rff_lo_bnd+rff_lo_shift), _
      3.0*((rff_lo_bnd+rff_lo_shift)-rff), _
      0) _
  + @recode(rff<(rff_lo_bnd+rff_lo_shift), _
    .10*exp(10*(rff-(rff_lo_bnd+rff_lo_shift))), _
    .10*exp(-20*(rff-(rff_lo_bnd+rff_lo_shift))))
smpl @all
series rff_lo_bnd = .125
series rff_lo_shift = .00
series penalty = 0
series penalty_aerr = 0
series rffmin = -9999
%penalty_weight = "10.0"
opt_targs.add penalty
smpl @all
series penalty_a = 0
series penalty_t = 0
series penalty_w = @val(%penalty_weight)
smpl %simstart+1 %simend
penalty_w = !discount * penalty_w(-1)
endif

```

```

'*****
' optimal policy simulation
'*****

% sufcontrol = "_1"

' In %simstr, the "type=opt" string designates a commitment-based
' OC simulation. The required "instrus" and "targs" keywords point to
' the groups containing the policy instrument(s) and target variables.
' FRB/US simulations of this type generally run much more quickly with
' the newton MCE algorithm than they do with qnewton.
%modstr = "mod_b=%varmod,mod_f=%mce,mod_mce_vars=%zvars"
%algstr = "jinit=interp(4), meth=newton"
%simstr = "type=opt,instrus=opt_instrus,targs=opt_targs"
%simstr = %simstr + ",scen,suf=" + %sufcontrol + ",solveopt=%sopt"
%simstr = %simstr + ",lend=" + %evlend + ",iend=" + %drvend + ",lmax=20"
smpl {%simstart} {%simend}
call mce_run(%modstr,%algstr,%simstr)

' When the ZLB is imposed, run the OC algorithm a second time,
' after adjusting the intercept of the penalty function,
' to hit ZLB more closely
if %zerobound = "yes" then
    smpl if rff{%sufcontrol} < .2
    rff_lo_shift = .125 - rff{%sufcontrol}
    %modstr = ""
    %algstr = ""
    %simstr = "type=opt,instrus=opt_instrus,targs=opt_targs"
    %simstr = %simstr + ",solveopt=%sopt"
    %simstr = %simstr + ",lend=" + %evlend + ",iend=" + %drvend + ",lmax=20"
    smpl {%simstart} {%simend}
    call mce_run(%modstr,%algstr,%simstr)
endif

'*****
' graph results
'*****

call dateshift(%simstart,%graphstart,-8)
call dateshift(%simstart,%graphend,32)

smpl %graphstart %graphend
graph fig1a.line rff{%sufcontrol} rff
fig1a.options size(7,4.2)
fig1a.legend display -inbox position(3.8,2.8) font("arial",15)

```

```

fig1a.datelabel format(yy)
fig1a.addtext(6.4,-.30,font("arial",13),keep) percent
fig1a.axis(left) font("arial",15)
fig1a.axis(bottom) font("arial",15)
fig1a.setelem(1) lcolor(red) legend("optimal control") lwidth(2)
fig1a.setelem(2) lcolor(black) legend("SEP-consistent baseline") lwidth(2)
fig1a.addtext(t,just(c),font("arial",18)) Federal Funds Rate

```

```

smpl %graphstart %graphend
graph fig1b.line rg10{%sufcontrol} rg10
fig1b.options size(7,4.2)
fig1b.legend display -inbox position(3.8,2.8) font("arial",15)
fig1b.datelabel format(yy)
fig1b.addtext(6.4,-.30,font("arial",13),keep) percent
fig1b.axis(left) font("arial",15)
fig1b.axis(bottom) font("arial",15)
fig1b.setelem(1) lcolor(red) legend("optimal control") lwidth(2)
fig1b.setelem(2) lcolor(black) legend("SEP-consistent baseline") lwidth(2)
fig1b.addtext(t,just(c),font("arial",18)) 10-Year Treasury Yield

```

```

smpl %graphstart %graphend
graph fig1c.line lur{%sufcontrol} lur
fig1c.options size(7,4.2)
fig1c.legend display -inbox position(3.9,0.3) font("arial",15)
fig1c.datelabel format(yy)
fig1c.addtext(6.4,-.30,font("arial",13),keep) percent
fig1c.axis(left) font("arial",15)
fig1c.axis(bottom) font("arial",15)
fig1c.setelem(1) lcolor(red) legend("optimal control") lwidth(2)
fig1c.setelem(2) lcolor(black) legend("SEP-consistent baseline") lwidth(2)
fig1c.addtext(t,just(c),font("arial",18)) Unemployment Rate

```

```

smpl %graphstart %graphend
graph fig1d.line pic4{%sufcontrol} pic4
fig1d.options size(7,4.2)
fig1d.legend display -inbox position(0.5,0.2) font("arial",15)
fig1d.datelabel format(yy)
fig1d.addtext(6.4,-.30,font("arial",13),keep) percent
fig1d.axis(left) font("arial",15)
fig1d.axis(bottom) font("arial",15)
fig1d.setelem(1) lcolor(red) legend("optimal control") lwidth(2)
fig1d.setelem(2) lcolor(black) legend("SEP-consistent baseline") lwidth(2)
fig1d.addtext(t,just(c),font("arial",18)) PCE Inflation Rate (4-Quarter)

```

```

graph fig1.merge fig1a fig1b fig1c fig1d
if %mcvars_wp = "no" and %mcvars_all = "no" then
    %title = "Macroeconomic Effects of Optimal-Control Policy with Rational Expectations only i
endif
if %mcvars_wp = "yes" and %mcvars_all = "no" then
    %title = "Macroeconomic Effects of Optimal-Control Policy\n With Rational Expectationsin Fi
endif
if %mcvars_all = "yes" then
    %title = "Macroeconomic Effects of Optimal-Control Policy With Full Rational Expectations"
endif

if %zerobound = "yes" then
    %title = %title + "\rZLB Imposed"
else
    %title = %title + "\rZLB not Imposed"
endif

fig1.addtext(t,just(c),font("Arial",20)) {%title}
fig1.align(2,1,1.25)
show fig1

```

This code is written to file srcEview/frbus.package/programs/ocpolicy.prg.

Defines:

ocpolicy, used in chunk 98.

Uses dateshift 9, group2zero 11a, groupnew 12, make_frbus_mcevars 94, mce_load_frbus 90,
mce_run 18, mce_solve_library 176 197, set_fp 14, and set_mp 15a.

B.15 srcEview/frbus.package/programs/pings.prg

```

144  <srcEview/frbus.package/programs/pings.prg 144>≡
      <simulate six ping simulations, aka simple IRFs 145>
      '*****
      '*****
      '*****
      <copy it 153>

      '*****
      '*****
      '*****
      <plot it 154>

      '*****
      '*****
      '*****
      <graph it 155>

```

This code is written to file srcEview/frbus.package/programs/pings.prg.

Defines:

pings, used in chunks 98 and 145.


```

145  <simulate six ping simulations, aka simple IRFs 145>≡ (144)
      ' Simulate six ping simulations (AKA simple IRFs)
      ,
      ' Notes:
      ,
      ' 1. Choose between VAR expectations and several MCE alternatives
      ' with the %mcevars parameter.
      ,
      ' - %mcevars = "none"    => VAR expectations everywhere
      ' - %mcevars = "mcap"   => MCE in asset pricing, VAR expectations elsewhere
      ' - %mcevars = "mcapwp" => MCE in asset pricing and price-wage setting;
      '   VAR expectations elsewhere
      ' - %mcevars = "all"    => MCE everywhere
      ,
      ' Note that even when %mcevars = "none", the program does many
      ' of the setup steps for an MCE simulation even though it never
      ' uses what they create.

      ' 2. Seven of the pings are one-time shocks to the residual of an
      ' equation whose structure contains a large autoregressive
      ' element. The remaining ping involves a permanent increase in the
      ' level of trend MFP.

      ' 3. The eight pings are:
      ,
      ' - A 100 basis point upward shock to the rffintay monetary
      '   policy rule
      ' - An increase in federal purchases equal to one percent of
      '   baseline GDP
      ' - A one percent permanent increase in the level of trend MPF
      ' - A 100 bp increase in the equity premium
      ' - A $10 per barrel increase in the price of oil
      ' - A 1 percent (ar) increase in the growth rate of
      '   multifactor productivity
      ' - Increases of 100 basis points to the 10-year Treasury term premium,
      '   75 basis points to the 5-year premium, and 30 basis points to the
      '   30-year premium
      ' - A 10 percent increase in the (real) exchange rate

      ' *****
      ' Initial filename and parameter settings
      ' *****

      ' Subroutines
      include ../subs/master_library
      include ../subs/mce_solve_library

```

```

' Workfile
  %wfstart = "1975q1"
  %wfend = "2100q4"
  %mainpage = "main"
  wfcreate(wf=aaa,page={%mainpage}) q {%wfstart} {%wfend}

' FRB/US model names and locations
  %varmod = "stdver"
  %varpath = "../mods/"
  %mcemod = "pfver"
  %mcepath = "../mods/"

' Input database
  %dbin = "../data/longbase"

' Simulation range
  %simstart = "2020q1"
  %simend = "2069q4"

' Choose an expectations option ("none" => VAR expectations, "mcap", "mcapwp", "all")
  %mcevars = "none"

' *****
' Retrieve data, model equations and coefficients, set
' policy options, and compute tracking residuals
' *****

' MCE variable setup
  if %mcevars = "none" then
    %zvars = "zpic58 "
  else
    %zvars = "zdivgr zgap05 zgap10 zgap30 zrff5 zrff10 zrff30 zpi10 zpi10f zpic30 zp
    if %mcevars = "mcapwp" or %mcevars = "all" then
      %zvars = %zvars + "zpicxfe zpieci "
    endif
    if %mcevars = "all" then
      %zvars = %zvars + "zecd zeco zeh zgapc2 zlhp zpi5 zvpd zvpi zvps zxbd zxbi zxb
    endif
  endif

' Load equations and coefficients
  call mce_load_frbus("mce_vars=%zvars,mod_b=%varmod,path_b=%varpath,mod_f=%mcemod,p

' Load data
  dbopen %dbin as longdata

```

```

    fetch(d=longdata) *

' Data for extra variables associated with MC expectations
  smpl @all
  call make_frbus_mcevars(%zvars)

' Set monetary policy
  smpl @all
  call set_mp("dmpintay")

' Turn off zero bound and policy thresholds; hold policymaker's
' perceived equilibrium real interest rate constant
  smpl @all
  dmptrsh = 0
  rffmin = -9999
  drstar = 0

' Set fiscal policy
  smpl @all
  call set_fp("dfpsrp")

' Set _aerr variables to zero
  smpl @all
  {%varmod}.makegroup(a,n) endog @endog
  call groupnew("endog","_aerr")
  call group2zero("endog_aerr")

' Standard solution options
  {%varmod}.solveopt(o=b,g=12,z=1e-12)
  {%mccmod}.solveopt(o=b,g=12,z=1e-12)

' Assign baseline tracking add factors
  %suftrk = "_0"
  smpl %simstart %simend
  {%varmod}.addassign @all
  {%varmod}.addinit(v=n) @all
  {%varmod}.scenario(n,a={%suftrk}) "track"
  {%varmod}.solve
  scalar mm = @max(@abs(xgap{%suftrk}-xgap))
  if mm > .0001 then
    statusline dynamic tracking simulation failed for {%varmod}
    stop
  endif
  {%mccmod}.addassign @all
  {%mccmod}.addinit(v=n) @all

```

```

' *****
' Ping simulations
' *****

%suf = "_1"
{%varmod}.scenario(n,a={%suf}) "ping"
{%mccmod}.scenario(n,a={%suf}) "ping"

' *****
' Federal Funds Rate: RFF ping

%ping = "rff"

smpl %simstart %simstart
rffintay_aerr = rffintay_aerr + 1
smpl %simstart %simend
if %mcevars = "none" then
    {%varmod}.solve
else
    %modstr = "mod_b=%varmod,mod_f=%mccmod,mce_vars=%zvars"
    %algstr = "meth=qnewton"
    %simstr = "type=single,solveopt=%sopt,suf=" + %suf
    call mce_run(%modstr,%algstr,%simstr)
endif
smpl %simstart %simstart
rffintay_aerr = rffintay_aerr - 1
call copyit

' *****
' Treasury Term Premium: RG10P, RG5P, and RG30P ping

%ping = "prem"
smpl @all
series rg30p_aerr = 0
smpl %simstart %simstart
rg10p_aerr = rg10p_aerr + 1
rg5p_aerr = rg5p_aerr + .75
rg30p_aerr = rg30p_aerr + .35
smpl %simstart %simend
if %mcevars = "none" then
    {%varmod}.solve
else

```

```

    %modstr = "mod_b=%varmod,mod_f=%mcecm,mod_g=%mcevars=%zvars"
    %algstr = "meth=qnewton"
    %simstr = "type=single,solveopt=%sopt,suf=" + %suf
    call mce_run(%modstr,%algstr,%simstr)
    endif
    smpl %simstart %simstart
    rg10p_aerr = rg10p_aerr - 1
    rg5p_aerr = rg5p_aerr - 0.75
    rg30p_aerr = rg30p_aerr - .35
    call copyit
    smpl %simstart %simend
    series rg10p_{%ping} = rg10p{%suf} - rg10p

' *****
' Federal Purchases: EGFO ping

%ping = "eg"
    smpl %simstart %simstart
    egfo_aerr = egfo_aerr + .01*xgdpn/egfn
    smpl %simstart %simend
    if %mcevars = "none" then
        {%varmod}.solve
    else
        %modstr = "mod_b=%varmod,mod_f=%mcecm,mod_g=%mcevars=%zvars"
        %algstr = "meth=qnewton"
        %simstr = "type=single,solveopt=%sopt,suf=" + %suf
        call mce_run(%modstr,%algstr,%simstr)
    endif
    smpl %simstart %simstart
    egfo_aerr = egfo_aerr - .01*xgdpn/egfn
    call copyit
    smpl %simstart %simend
    series egfn_shr_{%ping} = 100*(egfn{%suf}/xgdpn{%suf} - egfn/xgdpn)

' *****
' Equity Premium: REQP ping

%ping = "reqp"
    smpl %simstart %simstart
    reqp_aerr = reqp_aerr + 1
    smpl %simstart %simend
    if %mcevars = "none" then
        {%varmod}.solve
    else
        %modstr = "mod_b=%varmod,mod_f=%mcecm,mod_g=%mcevars=%zvars"
        %algstr = "meth=qnewton"

```

```

%simstr = "type=single,solveopt=%sopt,suf=" + %suf
call mce_run(%modstr,%algstr,%simstr)
endif
smpl %simstart %simstart
reqp_aerr = reqp_aerr - 1
call copyit
smpl %simstart %simend
series reqp_{%ping} = reqp{%suf} - reqp

' *****
' Oil Prices: POILR ping

%ping = "oil"
smpl %simstart %simstart
poilr_aerr = poilr_aerr + 10/pxb
smpl %simstart %simend
if %mcevars = "none" then
    {%varmod}.solve
else
    %modstr = "mod_b=%varmod,mod_f=%mccmod,mce_vars=%zvars"
    %algstr = "meth=qnewton"
    %simstr = "type=single,solveopt=%sopt,suf=" + %suf
    call mce_run(%modstr,%algstr,%simstr)
endif
smpl %simstart %simstart
poilr_aerr = poilr_aerr - 10/pxb
call copyit
smpl %simstart %simend
series poil_{%ping} = poil{%suf} - poil

' *****
' Exchange Rate: FPXRR ping

%ping = "exch"
smpl %simstart %simstart
series shock_fpxr = log(1.1)
fpxrr_aerr = fpxrr_aerr + shock_fpxr

smpl %simstart %simend
if %mcevars = "none" then
    {%varmod}.solve
else
    %modstr = "mod_b=%varmod,mod_f=%mccmod,mce_vars=%zvars"

```

```

    %algstr = "meth=qnewton"
    %simstr = "type=single,solveopt=%sopt,suf=" + %suf
    call mce_run(%modstr,%algstr,%simstr)
    endif

series fpxr_{%ping} = fpxr{%suf} - fpxr
smpl %simstart %simstart
fpxrr_aerr = fpxrr_aerr - shock_fpxr
call copyit

' *****
' HMFPT ping

%ping = "hmfpt"
smpl %simstart %simstart
hmfpt_aerr = hmfpt_aerr + 1
smpl %simstart %simend
if %mcevars = "none" then
    {%varmod}.solve
else
    %modstr = "mod_b=%varmod,mod_f=%mcevars,mce_vars=%zvars"
    %algstr = "meth=qnewton"
    %simstr = "type=single,solveopt=%sopt,suf=" + %suf
    call mce_run(%modstr,%algstr,%simstr)
endif
smpl %simstart %simstart
hmfpt_aerr = hmfpt_aerr - 1
call copyit
smpl %simstart %simend
series hmfpt_{%ping} = hmfpt{%suf} - hmfpt

' *****
' MFPT ping

%ping = "mfp"
smpl %simstart %simstart
mfpt_aerr = mfpt_aerr + .01
smpl %simstart %simend
if %mcevars = "none" then
    {%varmod}.solve
else
    %modstr = "mod_b=%varmod,mod_f=%mcevars,mce_vars=%zvars"
    %algstr = "meth=qnewton"
    %simstr = "type=single,solveopt=%sopt,suf=" + %suf

```

```

        call mce_run(%modstr,%algstr,%simstr)
    endif
    smpl %simstart %simstart
    mfpt_aerr = mfpt_aerr - .01
    call copyit
    smpl %simstart %simend
    series mfpt_{%ping} = 100*(mfpt{%suf}/mfpt - 1)

' *****
' Individual ping graphs
' *****
    call graphit

' *****
' Composite figures
' *****

    if %mcevars = "none" then
        %exp = "VAR Expectations"
    endif
    if %mcevars = "mcap" then
        %exp = "MC (MCAP) Expectations"
    endif
    if %mcevars = "mcapwp" then
        %exp = "MC (MCAP+WP) Expectations"
    endif
    if %mcevars = "all" then
        %exp = "MC (ALL) Expectations"
    endif

    %t1 = "FRB/US Ping Simulations:  " + %exp + " -- I"
    %t2 = "FRB/US Ping Simulations:  " + %exp + " -- II"
    %t3 = "FRB/US Ping Simulations:  " + %exp + " -- III"

' Figure 1

    graph fig_1.merge gr_rff gr_eg gr_reqp
    fig_1.align(3,.4,1.0)
    fig_1.addtext(t,just(c),font(12)) %t1
    show fig_1

' Figure 2

```



```

graph fig_2.merge gr_oil gr_hmfp gr_mfp
fig_2.align(3,.4,1.0)
fig_2.addtext(t,just(c),font(12)) %t2
show fig_2

```

' Figure 3

```

graph fig_3.merge gr_prem gr_exch
fig_3.align(3,.4,1.0)
fig_3.addtext(t,just(c),font(12)) %t3
show fig_3

```

Uses copyit 153, graphit 155, group2zero 11a, groupnew 12, make_frbus_mcevars 94,
 mce_load_frbus 90, mce_run 18, mce_solve_library 176 197, pings 144, set_fp 14,
 and set_mp 15a.

153 $\langle copy\ it\ 153 \rangle \equiv$ (144)
 subroutine copyit

```

smpl %simstart %simend
series picnia_{%ping} = picnia{%suf} - picnia
series pic4_{%ping} = pic4{%suf} - pic4
series picx4_{%ping} = picx4{%suf} - picx4
series picxfe_{%ping} = picxfe{%suf} - picxfe
series xgap2_{%ping} = xgap2{%suf} - xgap2
series lur_{%ping} = lur{%suf} - lur
series rff_{%ping} = rff{%suf} - rff

```

endsub

Defines:

copyit, used in chunk 145.

```

154  <plot it 154>≡ (144)
      subroutine plotit(string %grname, string %width, string %height, string %var1, str

      graph {%grname}.line {%var1} zero
      {%grname}.options size({%width},{%height}) -inbox
      {%grname}.setelem(1) linewidth(3) linepattern(1) linecolor(black)
      {%grname}.addtext(t,just(c),font(9)) %title
      {%grname}.addtext(0,-.15,font(8),just("r")) %units
      {%grname}.datelabel format(yyyy)
      {%grname}.legend -display
      {%grname}.axis(b) font(9)
      {%grname}.axis(l) font(9)

      endsub

```

Defines:

plotit, used in chunk 155.

```

155  <graph it 155>≡ (144)
      subroutine graphit

      smpl %simstart %simstart + 39
      series zero = 0

      delete(noerr) gr_*

' RFF ping

      %ping = "rff"

      %name = %ping + "a"
      %var1 = "xgap2_" + %ping
      %tt = "Response of Output Gap\rto Funds Rate"
      call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

      %name = %ping + "b"
      %var1 = "picxfe_" + %ping
      %tt = "Response of Core Inflation\rto Funds Rate"
      call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

      %name = %ping + "c"
      %var1 = "rff_" + %ping
      %tt = "Response of Funds Rate\rto Funds Rate"
      call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

      graph gr_{%ping}.merge  {%ping}a {%ping}b {%ping}c
      gr_{%ping}.align(3,.40,.40)

' EGFO ping

      %ping = "eg"

      %name = %ping + "a"
      %var1 = "xgap2_" + %ping
      %tt = "Response of Output Gap\rto Federal Purch"
      call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

      %name = %ping + "b"
      %var1 = "picxfe_" + %ping
      %tt = "Response of Core Inflation\rto Federal Purch"

```

```

call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

%name = %ping + "c"
%var1 = "egfn_shr_" + %ping
%tt = "Response of Federal Purch\erto Federal Purch"
call plotit(%name,"2","1.5",%var1,%tt,"percent of GDP")

graph gr_{%ping}.merge  {%ping}a {%ping}b {%ping}c
gr_{%ping}.align(3,.40,.40)

' REQP ping

%ping = "reqp"

%name = %ping + "a"
%var1 = "xgap2_" + %ping
%tt = "Response of Output Gap\erto Equity Premium"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

%name = %ping + "b"
%var1 = "picxfe_" + %ping
%tt = "Response of Core Inflation\erto Equity Premium"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

%name = %ping + "c"
%var1 = "reqp_" + %ping
%tt = "Response of Equity Premium\erto Equity Premium"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

graph gr_{%ping}.merge  {%ping}a {%ping}b {%ping}c
gr_{%ping}.align(3,.40,.40)

' POILR ping

%ping = "oil"

%name = %ping + "a"
%var1 = "xgap2_" + %ping
%tt = "Response of Output Gap\erto Oil Price"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

%name = %ping + "b"
%var1 = "picxfe_" + %ping

```

```
%tt = "Response of Core Inflation\rto Oil Price"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

%name = %ping + "c"
%var1 = "poil_" + %ping
%tt = "Response of Oil Price\rto Oil Price"
call plotit(%name,"2","1.5",%var1,%tt,"dollars per barrel")

graph gr_{%ping}.merge {%ping}a {%ping}b {%ping}c
gr_{%ping}.align(3,.40,.40)
```

’ HMFPT ping

```
%ping = "hmfp"

%name = %ping + "a"
%var1 = "xgap2_" + %ping
%tt = "Response of Output Gap\rto MFP Growth"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

%name = %ping + "b"
%var1 = "picxfe_" + %ping
%tt = "Response of Core Inflation\rto MFP Growth"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

%name = %ping + "c"
%var1 = "hmftp_" + %ping
%tt = "Response of MFP Growth\rto MFP Growth"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

graph gr_{%ping}.merge {%ping}a {%ping}b {%ping}c
gr_{%ping}.align(3,.40,.40)
```

’ MFPT ping

```
%ping = "mfp"

%name = %ping + "a"
%var1 = "xgap2_" + %ping
%tt = "Response of Output Gap\rto MFP Level"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")
```

```

%name = %ping + "b"
%var1 = "picxfe_" + %ping
%tt = "Response of Core Inflation\rto MFP Level"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

%name = %ping + "c"
%var1 = "mfpt_" + %ping
%tt = "Response of MFP Level\rto MFP Level"
call plotit(%name,"2","1.5",%var1,%tt,"percent")

graph gr_{%ping}.merge {%ping}a {%ping}b {%ping}c
gr_{%ping}.align(3,.40,.40)

%ping = "prem"

%name = %ping + "a"
%var1 = "xgap2_" + %ping
%tt = "Response of Output Gap\rto Term Premium"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

%name = %ping + "b"
%var1 = "picxfe_" + %ping
%tt = "Response of Core Inflation\rto Term Premium"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

%name = %ping + "c"
%var1 = "rg10p_" + %ping
%tt = "Response of Term Premium (10-year) \rto Term Premium"
call plotit(%name,"2","1.5",%var1,%tt,"percent")

graph gr_{%ping}.merge {%ping}a {%ping}b {%ping}c
gr_{%ping}.align(3,.40,.40)

%ping = "exch"

%name = %ping + "a"
%var1 = "xgap2_" + %ping
%tt = "Response of Output Gap\rto Exchange Rate"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

%name = %ping + "b"
%var1 = "picxfe_" + %ping
%tt = "Response of Core Inflation\rto Exchange Rate"
call plotit(%name,"2","1.5",%var1,%tt,"percentage points")

```

```

%name = %ping + "c"
%var1 = "fpxr_" + %ping
%tt = "Response of Exchange Rate \rto Exchange Rate"
call plotit(%name,"2","1.5",%var1,%tt,"percent")

graph gr_{%ping}.merge {%ping}a {%ping}b {%ping}c
gr_{%ping}.align(3,.40,.40)
endsub

```

Defines:

`graphit`, used in chunk 145.

Uses `plotit` 154.

B.16 `srcEview/frbus.package/programs/plot.resids.prg`

159 $\langle \text{srcEview/frbus.package/programs/plot.resids.prg 159} \rangle \equiv$
 $\langle \text{plot historical residuals of key equations 160} \rangle$

```

' *****
' *****
 $\langle \text{find variable description 162} \rangle$ 

```

This code is written to file `srcEview/frbus.package/programs/plot.resids.prg`.

Defines:

`plot_resids`, used in chunk 98.

```

160  <plot historical residuals of key equations 160>≡ (159)
      ' Program to plot the historical residuals of key FRB/US equations
      ,
      ' Each residual has the same units of measurement as the
      ' the left hand side of its equation

      ' *****
      ' Initial filename and parameter settings
      ' *****

      ' Subroutines
      include ../subs/master_library

      ' Workfile
      %wfstart = "1975q1"
      %wfend = "2030q4"
      %mainpage = "main"
      wfcreate(wf=aaa,page={%mainpage}) q {%wfstart} {%wfend}

      ' FRB/US model name and location
      %varmod = "stdver"
      %varpath = "../mods/"
      %varinfo = "../mods/stdver_varinfo"

      ' Input database
      %dbin = "../data/longbase"

      ' Plot range
      %plotstart = "1980q1"
      %plotend = "2014q2"

      ' *****
      ' Retrieve data, model equations and coefficients
      ' *****

      ' Load equations and coefficients
      ld_frbus_eqs(modelname=%varmod,modelpath=%varpath)
      ld_frbus_cfs(modelname=%varmod,modelpath=%varpath)
      ld_varinfo(pathname=%varinfo)

      ' Load data
      dbopen %dbin as longbase
      smpl %plotstart-12 %plotend
      fetch(d=longbase) *

```



```

' Set _aerr variables to zero
  smpl @all
  {%varmod}.makegroup(a,n) endog @endog
  call groupnew("endog","_aerr")
  call group2zero("endog_aerr")

' Compute baseline tracking add factors
  smpl %plotstart %plotend
  {%varmod}.addassign @all
  {%varmod}.addinit(v=n) @all

' *****
' Plots
' *****

%plotvars = "eco ecd eh epd epi eps ki ex emo lfpr lhp leo lww"
%plotvars = %plotvars + " picxfe pieci pcer pcfr pcengr"
%plotvars = %plotvars + " rg5p rg10p rg30p rbbbp rcar rme rtbe rcgain"
%plotvars = %plotvars + " ynidn yniin yhibn"

smpl %plotstart %plotend
series zero = 0

spool plot_vars

!counter = 0
for !i = 1 to @wcount(%plotvars)
  %vname = @word(%plotvars,!i)
  call find_var_description
  !counter = !counter + 1
  graph gr_{%vname}.line zero {%vname}_a
  %title = %vname + ": " + %desc
  gr_{%vname}.addtext(t) %title
  gr_{%vname}.axis range(minmax)
  gr_{%vname}.options size(4,3)
  gr_{%vname}.legend -display

  plot_vars.append gr_{%vname}
  %index = "000" + @str(!counter)
  if !counter < 100 then
    %index = @right(%index,2)
  else
    %index = @right(%index,3)
  endif
  %name = "untitled" + %index
  plot_vars.name {%name} {%vname}

```

```
next
```

```
plot_vars.display
```

Uses `find_var_description` 162, `group2zero` 11a, `groupnew` 12, `ld_frbus_cfs` 99a 99b, `ld_frbus_eqs` 100b 101, and `ld_varinfo` 114 115.

This was originally written in Fortran! Look at the *ii*, *jj*, *kk* variable names. I had forgotten about that from when I briefly helped the programmer debug the original model some 45 years ago. Who would have expected nostalgia from something so prosaic. I never saw the original source code because I was working with only a hex dump.

162 *(find variable description 162)≡* (159)
 subroutine find_var_description

```

for !j = 1 to 500
  %vline = vinfo_text.@line(!j)
  !eq = @instr(%vline,"=")
  !zz = !eq-4
  %name = @lower(@rtrim(@ltrim(@mid(%vline,4,!zz))))

  if %vname = %name then
    %desc = " "
    !ii = !eq + 1
    !kk = @instr(%vline,"sector_")
    if !kk = 0 then
      !kk = @instr(%vline,"X.")
      !jj = !kk - 1 - !ii
    else
      !jj = !kk - 7 - !ii
    endif
    %desc = @rtrim(@mid(%vline,!ii,!jj))
    exitloop
  endif
next
endsub
```

Defines:

`find_var_description`, used in chunk 160.

B.17 srcEview/frbus.package/programs/stochsim.prg

163 $\langle \text{srcEview/frbus.package/programs/stochsim.prg 163} \rangle \equiv$
 $\langle \text{stochastic simulations under variable expectations 164} \rangle$

```
'*****
'*****
'*****
'*****
'Subroutines

'*****
'*****
 $\langle \text{form table 173} \rangle$ 

'*****
'*****
 $\langle \text{make statistics 174} \rangle$ 
```

This code is written to file srcEview/frbus.package/programs/stochsim.prg.

Defines:

stochsim, used in chunk 98.

164 *(stochastic simulations under variable expectations 164)*≡ (163)

```

' Program for stochastic sims under VAR expectations

' The stochastic shocks are bootstrapped from the de-meaned
' historical errors of stochastic equations. The parameters
' %residstart and %residend declare the historical error range.
' A list of stochastic equations is extracted from the file
' pointed to by %varinfo.

' The bootstrap procedure randomly draws one historical quarter
' at a time when the parameter %errorblock = 1; alternatively, if
' %errorblock = 2, then the procedure would randomly draw two
' successive quarters at a time.

' The stochastic replications are simulated in a simple loop,
' rather than using the built-in EViews stochastic simulation
' procedure, so that shocks in the first simulation quarter can
' be scaled down by the parameter %q1_shock_damp. This feature
' is useful when uncertainty about the first simulation quarter
' in real-time analysis by known information. The shocks are
' not rescaled when %q1_shock_damp = 1.

' Similarly, the parameter %rff_weight_q1 is also designed to be
' used in real-time analysis when the first simulation quarter
' corresponds to a quarter that is already under way. The
' parameter provides the fractional value to be given to the
' monetary policy rule; the remaining fractional value is given to
' an exogenous value.

' The document Simulation Basics discusses the effects of
' imposing the zero lower bound (ZLB) on the federal funds rate
' (%zerobound parameter) and imposing threshold conditions
' on the liftoff of the funds rate from a ZLB episode
' (%threshold parameter).

' *****
' Initial filename and parameter settings
' *****

' Subroutines
include ../subs/master_library

' Workfile
%wfstart = "1965q1"
%wfend = "2020q4"
%mainpage = "main"

```

```

wfcreate(wf=aaa,page={%mainpage}) q {%wfstart} {%wfend}

' FRB/US model name and location
%varmod = "stdver"
%varpath = "../mods/"
%varinfo = "../mods/stdver_varinfo"

' Input database
%dbin = "../data/longbase"

' Simulation range
%simstart = "2014q1"
%simend = "2018q4"

' Stochastic parameters
rndseed 12345
%errorblock = "1"
%residstart = "1970q1"
%residend = "2012q4"
%nsims = "1000"
%q1_shock_damp = ".5"
%dbout_series = "rff lur picxfe picnia picx4 xgap2 hggdp anngr"

' Monetary policy
%zerobound = "yes"
%threshold = "yes"
%rff_weight_q1 = ".25"

' *****
' Retrieve data, model equations and coefficients, set
' policy options, and compute tracking residuals
' *****

' Load equations, coefficients, and variable information
ld_frbus_eqs(modelname=%varmod,modelpath=%varpath)
ld_frbus_cfs(modelname=%varmod,modelpath=%varpath)
ld_varinfo(pathname=%varinfo)

' add 4-qtr gdp growth equation
{%varmod}.append anngr - anngr_aerr = 100*((xgdp/xgdp(-4))-1)

' Load data
dbopen %dbin as longbase
fetch(d=longbase) *
smpl @all
series anngr = 100*((xgdp/xgdp(-4))-1)

```

```

' Set monetary policy to inertial Taylor rule (dmpintay, rffintay)
  smpl @all
  call set_mp("dmpintay")
  if %zerobound = "yes" then
    rffmin = .250
  else
    rffmin = -9999
  endif
  if %threshold = "yes" and %zerobound = "no" then
    %err = "Error: policy threshold conditions can only be used when the ZLB is impos
    @uiprompt(%err)
    stop
  endif
  if %threshold = "yes" and %zerobound = "yes" then
    dmptrsh = 1
    dmptr = 0
  else
    dmptrsh = 0
  endif
  drstar = 0

  smpl {%simstart} {%simstart}
  dmpintay = @val(%rff_weight_q1)
  dmpex = 1 - dmpintay

' Set fiscal policy
  smpl @all
  call set_fp("dfpex")

  dmpstb = 1

' Set _aerr variables to zero
  smpl @all
  {%varmod}.makegroup(a,n) endog @endog
  call groupnew("endog","_aerr")
  call group2zero("endog_aerr")

' Standard solution options
  {%varmod}.solveopt(o=b,g=14,z=1e-14)

' Assign baseline tracking add factors
  %suftrk = "_0"
  smpl %residstart %simend
  {%varmod}.addassign @all
  {%varmod}.addinit(v=n) @all

```

```

{%varmod}.scenario(n,a={%suftrk}) "track"
{%varmod}.solve
scalar mm = @max(@abs(xgap{%suftrk}-xgap))
if mm > .0001 then
    statusline dynamic tracking simulation failed for {%varmod}
stop
endif

' *****
' More monetary policy settings
' *****

' if policy thresholds are turned on, set add factors on endogenous
' threshold switch variables to zero
if %threshold = "yes" then
    smpl @all
    dmptpi_a = 0
    dmptlur_a = 0
    dmptmax_a = 0
    dmptr_a = 0
endif

' if the zero bound is binding in part or all of the baseline, the
' adds (_a) on the policy rule and the funds rate equations are
' determined so as to satisfy the following conditions.
'
' a. the stochastic funds rate equals the maximum of the zero bound and
' the prediction of the chosen policy rule (this simply requires that
' rffe_a and rffe_aerr be zero)
' b. the prediction of the chosen policy rule is subject to _a add factors
' that are determined as follows:
' (1) in quarters when the zero bound is not binding in the baseline,
' the associated add factors equal the values that make
' the policy rule equation match the baseline funds rate under
' baseline conditions (this is satisfied by the tracking adds on
' the policy rule as long as the baseline value of the policy rule
' variable equals the baseline funds rate);
' (2) in quarters when the zero bound is binding in the baseline,
' the associated add factors are determined by linear interpolation
' of the add factors generated according to b(1) for the
' unbound quarters;
' (3) when the zero bound is binding in all baseline quarters, the
' policy rule add factors are zero;
' (4) the zero bound is assumed to be binding in the baseline whenever
' the baseline funds rate (rffe) is within 25 basis points of the

```

```

',          zero bound variable (rffmin).

smpl %simstart %simend
series not_constrnd = ((rffe - rffmin) >= .25)
!tmp_max = @max(not_constrnd)
!tmp_min = @min(not_constrnd)

' zero bound binding in some quarters
if (!tmp_max = 1) and (!tmp_min = 0) then
    smpl %simstart %simend
    rffe_a = 0
    rffe_aerr = 0
    rffintay_aerr = 0
    smpl %simstart %simend if (not_constrnd = 0)
    rffintay_a = NA
    %series_in = "rffintay_a"
    %series_out = %series_in + "_int"
    call interp_lin(%series_in,%series_out,%simstart,%simend)
    rffintay_a = {%series_out}
endif

' zero bound binding in all quarters
if (!tmp_max = 0) and (!tmp_min = 0) then
    smpl %simstart %simend
    rffe_a = 0
    rffe_aerr = 0
    rffintay_aerr = 0
    rffintay_a = 0
endif

' *****
' Stochastic shocks
' *****

' copy historical residuals into series whose names have _err suffixes
smpl %residstart %residend
copy *_a *_err

' use vinfo table to create list/group of equations to receive shocks
%tmp = " "
for !i = 1 to vinfo_size
    %vname = @word(vinfo_vname,!i)
    %stoch = @word(vinfo_stoch,!i)
    if %stoch <> "NO" then
        %tmp = %tmp + " " + %vname
    endif
endfor

```



```

        next
    group shock {%tmp}

' demean historical residuals and store them in a matrix

    smpl %residstart %residend
    %error_names = " "
    for !i = 1 to shock.@count
        %temp = shock.@seriesname(!i) + "_err"
        scalar mm = @mean({%temp})
        series {%temp} = {%temp} - mm
        %error_names = %error_names + " " + %temp
    next

    group errors {%error_names}
    smpl %residstart %residend
    stom(errors,errorformat)

' create table of error statistics
    !nrows = 3 + errors.@count
    !ncols = 3
    table(!nrows,!ncols) error_tab
    error_tab.setjust(r1c1:r{!nrows}c1) left
    error_tab.setwidth(1) 20
    error_tab(1,1) = "error"
    error_tab(1,2) = "mean"
    error_tab(1,3) = "std-dev"
    smpl %residstart %residend
    for !i = 1 to errors.@count
        series tseries = errors(!i)
        error_tab(!i+3,1) = errors.@seriesname(!i)
        error_tab(!i+3,2) = @mean(tseries)
        error_tab(!i+3,3) = @stdev(tseries)
    next

' miscellaneous

    smpl %simstart %simend
    scalar nqtrs = @obssmpl
    scalar nrepl = {%nsims}
    scalar nsims = {%nsims}
    scalar nerrors = @rows(errorformat)
    scalar bbbb = nqtrs-nerrors

    call groupnew("shock","_aerr")

```

```

group track {%dbout_series}
call groupnames2string("track",%tracknames)

' for tracked variables, set up matrices to hold stochastic results
for !i = 1 to track.@count
    %temp = track.@seriesname(!i)
    matrix(nqtrs,nrepl) {%temp}_mat
next

' *****
' Stochastic sims
' *****

%sufstoch = "_1"
{%varmod}.scenario(n,a={%sufstoch}) "stoch sims"

' *****
' stochastic simulation loop (sims are run one at a time)

smpl %simstart %simend
for !i = 1 to nrepl
    statusline running stochastic sim number !i
    ' draw nqtrs random rows from the matrix of historical errors,
    ' damp the shocks in the first drawn row, and load the shocks
    ' into the respective _aerr error series
    matrix stocherrors = @resample(errormat,bbbb,{%errorblock})
    for !j = 1 to @columns(stocherrors)
        stocherrors(1,!j) = @val(%q1_shock_damp) * stocherrors(1,!j)
    next
    mtos(stocherrors,shock_aerr)
    {%varmod}.solve
    ' store solution values
    for !j = 1 to track.@count
        %temp = track.@seriesname(!j)
        %temp1 = %temp + "_mat"
        stom({%temp}{%sufstoch},tmp)
        colplace({%temp1},tmp,!i)
    next
next

' *****
' Statistics
' *****

```

```

statusline computing statistics

smpl %simstart %simend
!index = 2
series year = @year
series quarter = @quarter
alpha yyyyqq = @str(year) + "Q" + @str(quarter)
!lqtr = @dtoo(%simstart) - 1
!nstats = 8

' create a summary table in which to store key results

call tableform("summary_tab","100")

' loop over each tracked variable,
for !ii1 = 1 to track.@count
    %trkname = track.@seriesname(!ii1)
' compute statistics
    call makestats(%trkname)

' load statistics into variable-specific table
    %tabname = %trkname + "_tab"
    call tableform(%tabname,@str(nqtrs+2))
    for !ii2 = 1 to nqtrs
        {%trkname}_tab(!ii2+2,1) = yyyyqq(!lqtr + !ii2)
        for !ii3 = 1 to !nstats
            {%trkname}_tab(!ii2+2,!ii3+1) = {%trkname}_stats(!ii2,!ii3)
        next
    next

' load statistics for each q4 observation into summary table
    !index = !index + 1
    summary_tab(!index,1) = %trkname
    !index = !index + 1
    for !ii2 = 1 to nqtrs
        if quarter(!lqtr+!ii2) = 4 then
            for !ii3 = 1 to !nstats + 1
                summary_tab(!index,!ii3) = {%trkname}_tab(!ii2+2,!ii3)
            next
            !index = !index + 1
        endif
    next

' make graph showing 70 and 90 percent bands
graph {%trkname}_graph.band {%trkname}_lo90 {%trkname}_hi90 {%trkname}_lo70 _
    {%trkname}_hi70 {%trkname}_base

```

```

    {%trkname}_graph.addtext(t) %trkname
    {%trkname}_graph.options size(6,4.5)

next

' *****
' summary graph

lur_graph.addtext(t,just(c),font("arial",12)) Unemployment Rate
rff_graph.addtext(t,just(c),font("arial",12)) Federal Funds Rate
picx4_graph.addtext(t,just(c),font("arial",12)) 4-qtr Core Inflation Rate
anngr_graph.addtext(t,just(c),font("arial",12)) 4-qtr Real GDP Growth Rate
lur_graph.legend -display
rff_graph.legend -display
picx4_graph.legend -display
anngr_graph.legend -display

graph summary_graph.merge lur_graph rff_graph picx4_graph anngr_graph
summary_graph.legend -display
summary_graph.addtext(t,just(c),font("Arial",16)) Stochastic Simulations\r(70 and 9
show summary_graph

' *****
' summary spool
spool results
summary_tab.deleterow(!index) 100
results.append summary_tab
results.append summary_graph
results.append error_tab
results.display

Uses group2zero 11a, groupnames2string 11b, groupnew 12, interp_lin 13,
ld_frbus_cfs 99a 99b, ld_frbus_eqs 100b 101, ld_varinfo 114 115, makestats 174,
set_fp 14, set_mp 15a, and tableform 173.

```

```

173  <form table 173>≡ (163)
      subroutine tableform(string %tablename, string %nrows)

          table(@val(%nrows),9) {%tablename}

          {%tablename}.setwidth(1:9) 8
          {%tablename}.setjust(r1c1:r{%nrows}c9) right
          {%tablename}.setformat(r2c2:r{%nrows}c9) f.3
          {%tablename}(1,1) = "qtr"
          {%tablename}(1,2) = "baseline"
          {%tablename}(1,3) = "mean"
          {%tablename}(1,4) = "median"
          {%tablename}(1,5) = "stdev"
          {%tablename}(1,6) = "90%-low"
          {%tablename}(1,7) = "90%-hi"
          {%tablename}(1,8) = "70%-low"
          {%tablename}(1,9) = "70%-hi"

          endsub

```

Defines:

`tableform`, used in chunk 164.

```

174  <make statistics 174>≡ (163)
      subroutine makestats(string %trkname)

          %trkmat = %trkname + "_mat"
          %statsmat = %trkname + "_stats"
          matrix(nqtrs,8) {%statsmat}

          smpl {%simstart} {%simend}

      ' loop over each simulation quarter
      for !ii2 = 1 to nqtrs
      ' put simulation replications for this quarter into matrix tempm1
      matrix tempm1 = @sort(@rowextract({%trkmat},!ii2))
      {%statsmat}(!ii2,1) = {%trkname}(!lqtr + !ii2)
      {%statsmat}(!ii2,2) = @mean(tempm1)
      {%statsmat}(!ii2,3) = tempm1(1,@floor(.50*nrepl))
      {%statsmat}(!ii2,4) = @stdev(tempm1)
      {%statsmat}(!ii2,5) = tempm1(1,@floor(.05*nrepl))
      {%statsmat}(!ii2,6) = tempm1(1,@floor(.95*nrepl))
      {%statsmat}(!ii2,7) = tempm1(1,@floor(.15*nrepl))
      {%statsmat}(!ii2,8) = tempm1(1,@floor(.85*nrepl))

      next

      ' also create individual series for each statistic
      series {%trkname}_base = 0
      series {%trkname}_mn = 0
      series {%trkname}_med = 0
      series {%trkname}_se = 0
      series {%trkname}_lo90 = 0
      series {%trkname}_hi90 = 0
      series {%trkname}_lo70 = 0
      series {%trkname}_hi70 = 0

      group {%trkname}_group {%trkname}_base {%trkname}_mn {%trkname}_med _
          {%trkname}_se {%trkname}_lo90 {%trkname}_hi90 {%trkname}_lo70 _
          {%trkname}_hi70
      mtos({%statsmat},{%trkname}_group)

      endsub

```

Defines:

makestats, used in chunk 164.

B.18 srcEview/frbus.package/subs/master.library.prg

175a $\langle \text{srcEview/frbus.package/subs/master.library.prg } 175a \rangle \equiv$
 $\langle \text{master library routines } 175b \rangle$

This code is written to file srcEview/frbus.package/subs/master.library.prg.

175b $\langle \text{master library routines } 175b \rangle \equiv$ (175a)

```

' *****
' *****
' *****
 $\langle \text{quarterly date string shift } 9 \rangle$ 

' *****
' *****
' *****
 $\langle \text{copy series into group } 10 \rangle$ 
' *****
' *****
' *****
 $\langle \text{set group to zero } 11a \rangle$ 
' *****
' *****
' *****
 $\langle \text{names of all series in group } 11b \rangle$ 
' *****
' *****
' *****
 $\langle \text{create new group } 12 \rangle$ 
' *****
' *****
' *****
 $\langle \text{interpolate unavailable observations } 13 \rangle$ 
' *****
' *****
' *****
 $\langle \text{set fiscal policy option } 14 \rangle$ 
' *****
' *****
' *****
 $\langle \text{set monetary policy option } 15a \rangle$ 
' *****
' *****
' *****
 $\langle \text{set monetary policy fed funds rate } 15b \rangle$ 

```

B.19 `srcEview/frbus.package/subs/mce.solve.library.prg`

176 `<srcEview/frbus.package/subs/mce.solve.library.prg 176>≡`
 `<mce solve library 177>`

This code is written to file `srcEview/frbus.package/subs/mce.solve.library.prg`.

Defines:

`mce_solve_library`, used in chunks 98, 120, 130, 136, 145, 181, 184, 187, 191, and 194.

177 $\langle mce \text{ solve library } 177 \rangle \equiv$ (176 197)
 $\langle mce \text{ solve library change history } 16 \rangle$

```
' *****
' *****
' *****
<run model consistent expectations 18>
```

```
' *****
' *****
' *****
<determine endogenous and exogenous variables 25>
```

```
' *****
' *****
' *****
<determine default method, linesearch, and other options 28>
```

```
' *****
' *****
' *****
<parse options containing equal signs 32a>
```

```
' *****
' *****
' *****
<parse options not containing equal signs 32b>
```

```
' *****
' *****
' *****
' *****
<create common variables, strings, matrices, vectors, and tables 33>
```

```
' *****
' *****
' *****
<model consistent coefficient simulation 35>
```

```
» *****
» *****
» *****
<solve model consistent instrument values 44>
```

```
» *****
» *****
» *****
<compute derivatives of mce targets wrt mce instruments 46>
```

```
» *****
» *****
» *****
<model consistent coefficient non-monotone step-length procedure 52>
```

```
» *****
» *****
» *****
<model consistent armijo optimization rule 54>
```

```
» *****
» *****
» *****
<variable terminal values 55>
```

```
» *****
» *****
» *****
<set options based on defaults and overrides 57>
```

```
» *****
» *****
» *****
```

⟨main unconstrained optimal control simulation 64⟩

```

³ *****
³ *****
³ *****
⟨main optimal control simulation with inequality constraints 68⟩

```

```

³ *****
³ *****
³ *****
⟨model consistent optimal time-consistent solution 73⟩

```

```

³ *****
³ *****
³ *****
⟨compute derivatives of loss function targets wrt instruments 81⟩

```

```

³ *****
³ *****
³ *****
⟨solve model consistent expectations model 83⟩

```

```

³ *****
³ *****
³ *****
⟨convert mcz inequality constraints 85⟩

```

```

³ *****
³ *****
³ *****
⟨shift left 88⟩

```

```

³ *****
³ *****
³ *****
⟨find next delimiter 89⟩

```

```
, *****  
, *****  
, *****  
<load frbus with transformed subsidiary model 90>  
  
, *****  
, *****  
, *****  
<create wage and expectation variables in forward looking model 94>
```

B.20 srcEview/mce.solve.package/example1.prg

```

181 <srcEview/mce.solve.package/example1.prg 181>≡
    ' This example illustrates:
    '
    ' - The automated approach to constructing the two operational
    '   models
    ' - How to define the multiplier shock in a text
    '   file whose lines are executed from within the call to
    '   mce_run
    ' - How to declare the model scenario and the scenario
    '   scenario alias within the call to mce_run
    ' - The use of the "linear" option of the "newton" algorithm
    '   for a linear model in which the maximum endogenous lead and
    '   lag is one period

    ' *****
    ' *****
    ' *****
    ' Section 1:  Workfile, model name, simulation range

    include mce_solve_library

    ' Workfile
    %wfstart = "2000q1"
    %wfend = "2100q4"
    %mainpage = "main"
    wfcreate(wf=aaa,page={%mainpage}) q {%wfstart} {%wfend}

    ' Model name
    %mod = "simple"

    ' Simulation range
    %simstart = "2001q1"
    %simend = "2025q4"

    ' *****
    ' *****
    ' *****
    ' Section 2:  Model, coefficients, and data

    ' equations

    model {%mod}
    {%mod}.append pinf = cp(1) * pinf(-1) + (.98-cp(1))*pinf(1)+ cp(2) * ygap

```

```

{%mod}.append rate = cr(1)*rate(-1)+(1-cr(1))*(cr(2)*pinf + cr(3)*ygap)
{%mod}.append ygap = cy(1) * ygap(-1) + (.98-cy(1))*ygap(1) + cy(2) * (rate - pinf

' coefficients

coef(2) cy
cy.fill .50, -.02
coef(2) cp
cp.fill .50, .02
coef(3) cr
cr.fill .75, 1.5, 0.5

' set all data to zero
smpl @all
%vars = {%mod}.@varlist
for !i = 1 to @wcount(%vars)
    %tmp = @word(%vars,!i)
    series {%tmp} = 0
next

' *****
' *****
' *****
' Section 3: Simulation

text shock1
shock1.append smpl {%simstart} {%simstart}
shock1.append series rate_a = rate_a + 1

%mopts = "create,mod=%mod,adds,track"
%aopts = "jinit=linear"
%sopts = "type=single,txt=shock1,scen,suf=_1"
smpl {%simstart} {%simend}
call mce_run(%mopts,%aopts,%sopts)
copy mce_sim_spool mce_sim_spool_1
show mce_sim_spool_1

series zero = 0
smpl %simstart %simstart + 39
graph gr1.line zero rate_1 pinf_1 ygap_1
gr1.addtext(t,c,font(14)) "Positive interest rate shock"
show gr1

```

This code is written to file `srcEview/mce.solve.package/example1.prg`.

Defines:

`example1`, used in chunk 98.

Uses `mce_run` 18 and `mce.solve.library` 176 197.

B.21 srcEview/mce.solve.package/example2.prg

```

184  <srcEview/mce.solve.package/example2.prg 184>≡
      ' This example illustrates:
      '
      '   - A manual approach to constructing the two operational
      '     models that mimics what the automated approach does
      '   - How to define the multiplier shock in a text
      '     file whose lines are executed from within the call to
      '     mce_run
      '   - How to declare the model scenario and the scenario
      '     scenario alias within the call to mce_run
      '   - The use of the "linear" option of the "newton" algorithm
      '     for a linear model in which the maximum endogenous lead and
      '     lag is one period

      '*****
      '*****
      '*****
      ' Section 1:  Workfile, model name, simulation range

      include  mce_solve_library

      ' Workfile
      %wfstart = "2000q1"
      %wfend = "2100q4"
      %mainpage = "main"
      wfcreate(wf=aaa,page={%mainpage}) q {%wfstart} {%wfend}

      ' Model names
      %modb = "simpleb"
      %modf = "simplef"

      ' Simulation range
      %simstart = "2001q1"
      %simend = "2025q4"

      '*****
      '*****
      '*****
      ' Section 2:  Model, coefficients, and data

      ' equations in backward-looking model

      model {%modb}

```



```

{%modb}.append pinf = cp(1) * pinf(-1) + (.98-cp(1))*zpinf+ cp(2) * ygap
{%modb}.append rate = cr(1)*rate(-1)+(1-cr(1))*(cr(2)*pinf + cr(3)*ygap)
{%modb}.append ygap = cy(1) * ygap(-1) + (.98-cy(1))*zygap + cy(2) * (rate - zpinf)

' equations in expectations errors model

model {%modf}
{%modf}.append ezpinf = zpinf - pinf(1)
{%modf}.append ezygap = zygap - ygap(1)

' coefficients

coef(2) cy
cy.fill .50, -.02
coef(2) cp
cp.fill .50, .02
coef(3) cr
cr.fill .75, 1.5, 0.5

' set all data to zero
smpl @all
%vars = {%modb}.@varlist
for !i = 1 to @wcount(%vars)
  %tmp = @word(%vars,!i)
  series {%tmp} = 0
next
%vars = {%modf}.@varlist
for !i = 1 to @wcount(%vars)
  %tmp = @word(%vars,!i)
  series {%tmp} = 0
next

' declare mce variables and instruments
%instrus = "zpinf zygap"
%errs = "ezpinf ezygap"

' *****
' *****
' *****

' Section 3: Simulation

text shock1
shock1.append smpl {%simstart} {%simstart}
shock1.append series rate_a = rate_a + 1

```

```
%mopts = "mod_b=%modb,mod_f=%modf,mce_instrus=%instrus,mce_errs=%errs,adds,track"
%aopts = "jinit=linear"
%sopts = "type=single,txt=shock1,scen,suf=_1"
smpl {%simstart} {%simend}
call mce_run(%mopts,%aopts,%sopts)
copy mce_sim_spool mce_sim_spool_1
show mce_sim_spool_1

series zero = 0
smpl %simstart %simstart + 39
graph gr1.line zero rate_1 pinf_1 ygap_1
gr1.addtext(t,c,font(14)) "Positive interest rate shock"
show gr1
```

This code is written to file `srcEview/mce.solve.package/example2.prg`.

Defines:

`example2`, used in chunk 98.

Uses `mce_run` 18 and `mce_solve_library` 176 197.

B.22 srcEview/mce.solve.package/example3.prg

```

187 <srcEview/mce.solve.package/example3.prg 187>≡
' This example illustrates:
,
' - Another manual approach to constructing the two operational
'   models that introduces new endogenous variables for the
'   expectations leads along with simple equations for the new
'   endogenous variables.
' - The MCE instruments are the add factors on the equations for
'   the new endogenous variables; this circumstance requires that
'   add factors be assigned to the operational models prior to
'   the call to mce_run.
' - The option of defining the multiplier shock in commands that
'   are executed prior to the call to mce_run, when the manual
'   approach is used.
' - The option of declaring model scenarios and the scenario
'   scenario alias prior to the call to mce_run, when the manual
'   approach is used.
' - The model no longer satisfies the conditions for which the
'   "jinit=linear" option of the "newton" algorithm is designed.
'   This example uses "jinit=interp(4)" to specify a
'   particular approximate Jacobian.
,
' - This example runs three simulations. In the second and third
'   simulations, the assignment of null strings to the first two
'   arguments of the mce_run subroutine causes the simulations to be
'   run with the same internal models and algorithm (including
'   the Newton MCE Jacobian) that were created or declared in the
'   first simulation.

' *****
' *****
' *****
' Section 1: Workfile, model name, simulation range

include mce_solve_library

' Workfile
%wfstart = "2000q1"
%wfend = "2100q4"
%mainpage = "main"
wfcreate(wf=aaa,page={%mainpage}) q {%wfstart} {%wfend}

' Model names
%modb = "simpleb"

```

```

%modf = "simplef"

' Simulation range
%simstart = "2001q1"
%simend = "2025q4"

'*****
'*****
'*****
' Section 2: Model, coefficients, and data

' equations in backward-looking model

model {%modb}
{%modb}.append pinf = cp(1) * pinf(-1) + (.98-cp(1))*zpinf+ cp(2) * ygap
{%modb}.append rate = cr(1)*rate(-1)+(1-cr(1))*(cr(2)*pinf + cr(3)*ygap)
{%modb}.append ygap = cy(1) * ygap(-1) + (.98-cy(1))*zygap + cy(2) * (rate - zpinf)
{%modb}.append zpinf = @movav(pinf(-1),4)
{%modb}.append zygap = @movav(ygap(-1),4)

' equations in expectations errors model

model {%modf}
{%modf}.append ezpinf = zpinf - pinf(1)
{%modf}.append ezygap = zygap - ygap(1)

' coefficients

coef(2) cy
cy.fill .50, -.02
coef(2) cp
cp.fill .50, .02
coef(3) cr
cr.fill .75, 1.5, 0.5

' set all data to zero
smpl @all
%vars = {%modb}.@varlist
for !i = 1 to @wcount(%vars)
    %tmp = @word(%vars,!i)
    series {%tmp} = 0
next
%vars = {%modf}.@varlist
for !i = 1 to @wcount(%vars)
    %tmp = @word(%vars,!i)

```

```

        series {%tmp} = 0
    next

' declare mce variables and instruments
%instrus = "zpinf_a zygap_a"
%errs = "ezpinf ezygap"

' assign tracking add factors
smpl %simstart %simend
{%modb}.addassign @all
{%modb}.addinit(v=n) @all
{%modf}.addassign @all
{%modf}.addinit(v=n) @all

' *****
' *****
' *****
' Section 3: Simulations

%sufm = "_1"
{%modb}.scenario(n,a=%sufm) "multiplier"
{%modf}.scenario(n,a=%sufm) "multiplier"

' Sim 1: interest rate shock

smpl {%simstart} {%simstart}
rate_a = rate_a + 1

%mopts = "mod_b=%modb,mod_f=%modf,mce_instrus=%instrus,mce_errs=%errs"
%aopts = "jinit=interp(4)"
%sopts = "type=single"
smpl {%simstart} {%simend}
call mce_run(%mopts,%aopts,%sopts)
smpl {%simstart} {%simstart}
rate_a = rate_a - 1

series zero = 0
smpl %simstart %simstart + 39
graph gr1.line zero rate{%sufm} pinf{%sufm} ygap{%sufm}
gr1.addtext(t,c,font(14)) "Positive interest rate shock"
show gr1

' Sim 2: output gap shock

```

```

smpl {%simstart} {%simstart}
ygap_a = ygap_a + 1

%mopts = ""
%aopts = ""
%sopts = "type=single"
smpl {%simstart} {%simend}
call mce_run(%mopts,%aopts,%sopts)
smpl {%simstart} {%simstart}
ygap_a = ygap_a - 1

series zero = 0
smpl %simstart %simstart + 39
graph gr2.line zero rate{%sufm} pinf{%sufm} ygap{%sufm}
gr2.addtext(t,c,font(14)) "Positive output gap shock"
show gr2

' Sim 3:  inflation shock

smpl {%simstart} {%simstart}
pinf_a = pinf_a + 1

%mopts = ""
%aopts = ""
%sopts = "type=single"
smpl {%simstart} {%simend}
call mce_run(%mopts,%aopts,%sopts)
smpl {%simstart} {%simstart}
pinf_a = pinf_a - 1

series zero = 0
smpl %simstart %simstart + 39
graph gr3.line zero rate{%sufm} pinf{%sufm} ygap{%sufm}
gr3.addtext(t,c,font(14)) "Positive inflation shock"
show gr3

```

This code is written to file `srcEview/mce.solve.package/example3.prg`.

Defines:

`example3`, used in chunk 98.

Uses `mce_run` 18 and `mce_solve_library` 176 197.

B.23 srcEview/mce.solve.package/example4.prg

```

191 <srcEview/mce.solve.package/example4.prg 191>≡
    ' This example illustrates:
    ,
    '   - The simulation of a nonlinear model (zero bound imposed)
    '     (because the baseline data is set to zero; the zero-bound
    '       is set illustratively to -1)
    '   - The use of the qnewton algorithm

    ' *****
    ' *****
    ' *****
    ' Section 1:  Workfile, model name, simulation range

    include mce_solve_library

    ' Workfile
    %wfstart = "2000q1"
    %wfend = "2100q4"
    %mainpage = "main"
    wfcreate(wf=aaa,page={%mainpage}) q {%wfstart} {%wfend}

    ' Model names
    %modb = "simpleb"
    %modf = "simplef"

    ' Simulation range
    %simstart = "2001q1"
    %simend = "2025q4"

    ' *****
    ' *****
    ' *****
    ' Section 2:  Model, coefficients, and data

    ' equations in backward-looking model

    model {%modb}
    {%modb}.append pinf = cp(1) * pinf(-1) + (.98-cp(1))*zpinf+ cp(2) * ygap
    {%modb}.append rate_u = cr(1)*rate(-1)+(1-cr(1))*(cr(2)*pinf + cr(3)*ygap)
    {%modb}.append rate = @recode( rate_u>rate_min,rate_u,rate_min)
    {%modb}.append ygap = cy(1) * ygap(-1) + (.98-cy(1))*zygap + cy(2) * (rate - zpinf)
    {%modb}.append zpinf = @movav(pinf(-1),4)
    {%modb}.append zygap = @movav(ygap(-1),4)

```

```

' equations in expectations errors model

model {%modf}
{%modf}.append ezpinf = zpinf - pinf(1)
{%modf}.append ezygap = zygap - ygap(1)

' coefficients

coef(2) cy
cy.fill .50, -.02
coef(2) cp
cp.fill .50, .02
coef(3) cr
cr.fill .75, 1.5, 0.5

' set all data to zero
smpl @all
%vars = {%modb}.@varlist
for !i = 1 to @wcount(%vars)
    %tmp = @word(%vars,!i)
    series {%tmp} = 0
next
%vars = {%modf}.@varlist
for !i = 1 to @wcount(%vars)
    %tmp = @word(%vars,!i)
    series {%tmp} = 0
next

' declare mce variables and instruments
%instrus = "zpinf_a zygap_a"
%errs = "ezpinf ezygap"

' assign tracking add factors
smpl %simstart %simend
{%modb}.addassign @all
{%modb}.addinit(v=n) @all
{%modf}.addassign @all
{%modf}.addinit(v=n) @all

' *****
' *****
' *****
' Section 3: Simulation

%zb = "yes"

```



```

%sufm = "_1"
{%modb}.scenario(n,a=%sufm) "multiplier"
{%modf}.scenario(n,a=%sufm) "multiplier"

if %zb = "yes" then
    smpl @all
    rate_min = -1
else
    rate_min = -9999
endif

smpl {%simstart} {%simstart}
ygap_a = ygap_a - 5

%mopts = "mod_b=%modb,mod_f=%modf,mce_instrus=%instrus,mce_errs=%errs"
%aopts = "meth=qnewton"
%sopts = "type=single"
smpl {%simstart} {%simend}
tic
call mce_run(%mopts,%aopts,%sopts)
scalar elapsed = @toc
show elapsed

series zero = 0
smpl %simstart %simstart + 39
graph gr1.line zero rate{%sufm} pinf{%sufm} ygap{%sufm}
%title = "Negative Output Shock"
if %zb = "yes" then
    %title = %title + "\r(zero bound imposed)"
else
    %title = %title + "\r(zero bound not imposed)"
endif
gr1.addtext(t,c,font(14)) %title
show gr1

```

This code is written to file `srcEview/mce.solve.package/example4.prg`.

Defines:

`example4`, used in chunk 98.

Uses `mce_run` 18 and `mce_solve_library` 176 197.

B.24 srcEview/mce.solve.package/example5.prg

```

194  <srcEview/mce.solve.package/example5.prg 194>≡
      ' This example illustrates the two optimal policy simulation types
      ,
      '   - Simulate the effects of a positive shock to the output gap
      '     using sequentially
      ,
      '     (a) the model's interest rate rule
      ,
      '     (b) the opt simulation type to find the optimal interest rate
      '     path under commitment
      ,
      '     (c) the opttc simulation type to find the optimal time-consistent
      '     or discretionary interest rate path; note that the solution in
      '     this case is only approximate
      ,
      '     In both (b) and (c) the policy instrument is the residual of
      '     the interest rule
      ,
      '   - The illustrative loss function penalizes equally weighted,
      '     discounted, squared deviations of the output gap,
      '     inflation, and the first difference of the interest rate.
      ,

      ' *****
      ' *****
      ' *****
      ' Section 1:  Workfile, model name, simulation range

      include  mce_solve_library

      ' Workfile
      %wfstart = "2000q1"
      %wfend = "2100q4"
      %mainpage = "main"
      wfcreate(wf=aaa,page={%mainpage}) q {%wfstart} {%wfend}

      ' Model name
      %mod = "simple"

      ' Simulation range
      %simstart = "2001q1"
      %simend = "2025q4"

      ' *****

```

```

'*****
'*****
' Section 2: Model, coefficients, and data

' equations

model {%mod}
{%mod}.append pinf = cp(1) * pinf(-1) + (.98-cp(1))*pinf(1)+ cp(2) * ygap
{%mod}.append rate - rate_aerr = cr(1)*rate(-1)+(1-cr(1))*(cr(2)*pinf + cr(3)*ygap)
{%mod}.append ygap = cy(1) * ygap(-1) + (.98-cy(1))*ygap(1) + cy(2) * (rate - pinf(1))
{%mod}.append drate = rate - rate(-1)

' coefficients

coef(2) cy
cy.fill .50, -.02
coef(2) cp
cp.fill .50, .02
coef(3) cr
cr.fill .75, 1.5, 0.5

' set all data to zero
smpl @all
%vars = {%mod}.@varlist
for !i = 1 to @wcount(%vars)
    %tmp = @word(%vars,!i)
    series {%tmp} = 0
next

'*****
'*****
'*****
' Section 3: Optimal policy setup

' targets, instruments
group opt_instrus rate_aerr
group opt_targs pinf ygap drate

' desired target trajectories
smpl @all
series pinf_t = 0
series ygap_t = 0
series drate_t = 0

' loss function weights

```

```

%discount = ".99"
%y_weight = "1.0"
%p_weight = "1.0"
%r_weight = "1.0"
smpl @all
series ygap_w = @val(%y_weight)
series pinf_w = @val(%p_weight)
series drate_w = @val(%r_weight)
!discount = @val(%discount)
smpl %simstart+1 %simend
ygap_w = !discount * ygap_w(-1)
pinf_w = !discount * pinf_w(-1)
drate_w = !discount * drate_w(-1)

'*****
'*****
'*****
' Section 3: Simulations

text shock1
shock1.append smpl {%simstart} {%simstart}
shock1.append series ygap_a = ygap_a + 3

' run the simulation using the monetary policy rule

%mopts = "create,mod=%mod,adds,track"
%aopts = "jinit=linear"
%sopts = "type= single,txt=shock1,scen,suf=_1"
smpl {%simstart} {%simend}
call mce_run(%mopts,%aopts,%sopts)

smpl @all
series rate_rule = rate_1
series pinf_rule = pinf_1
series ygap_rule = ygap_1

' run the simulation using opt

%mopts = ""
%aopts = ""
%sopts = "type=opt,instrus=opt_instrus,targs=opt_targs"
smpl {%simstart} {%simend}
call mce_run(%mopts,%aopts,%sopts)

```

```

smpl @all
series rate_opt = rate_1
series pinf_opt = pinf_1
series ygap_opt = ygap_1

' run the simulation using opttc

%mopts = ""
%aopts = ""
%sopts = "type=opttc,instrus=opt_instrus,targs=opt_targs"
smpl {%simstart} {%simend}
call mce_run(%mopts,%aopts,%sopts)

smpl @all
series rate_opttc = rate_1
series pinf_opttc = pinf_1
series ygap_opttc = ygap_1

'graph

smpl %simstart %simstart + 39
series zero = 0
graph gr1.line zero rate_rule rate_opt rate_opttc
graph gr2.line zero pinf_rule pinf_opt pinf_opttc
graph gr3.line zero ygap_rule ygap_opt ygap_opttc
graph gr4.merge gr1 gr2 gr3
%title = "Effects of Positive Output Shock Under Three Policy Responses"
%title = %title + "\r1. An intertial interest rate rule (_rule)"
%title = %title + "\r2. Optimal policy under commitment (_opt)"
%title = %title + "\r3. Optimal time-consistent policy (_opttc)"
gr4.addtext(t,c,font(14)) %title
show gr4

```

This code is written to file `srcEview/mce.solve.package/example5.prg`.

Defines:

`example5`, used in chunk 98.

Uses `mce_run` 18 and `mce_solve_library` 176 197.

B.25 `srcEview/mce.solve.package/mce.solve.library.prg`

197 $\langle \text{srcEview/mce.solve.package/mce.solve.library.prg } 197 \rangle \equiv$
 $\langle \text{mce solve library } 177 \rangle$

This code is written to file `srcEview/mce.solve.package/mce.solve.library.prg`.

Defines:

`mce_solve_library`, used in chunks 98, 120, 130, 136, 145, 181, 184, 187, 191, and 194.

B.26 srcEview/state.space.package/data.transformations.prg

198 *<srcEview/state.space.package/data.transformations.prg 198>*≡
 subroutine data_transformations()

' Subroutine for FRB/US state-space model package to
 ' transform raw data from EViews database to observables
 ' used in the estimation of the state-space model.

' 1. Data transformations

series lpop = 100*log(n16)

'Business - product side
 series nbp = xbn
 series bp = log(xb)*100 -lpop

'GDP
 series gdp = log(xgdp)*100 -lpop

'Buseness - income side
 series dsnst_q = xgdpn - xgdin
 series nbi = nbp - dsnst_q
 series rbi = nbi/(pxb/100)
 series bi = log(rbi)*100 -lpop

' Employment Business sector
 series eb = log(lep)*100 - lpop

' Workweek
 series hb = log(lhp)*100 - lpop
 series bww = hb - eb

' Employment Rate
 series erate = 100*log((100-lur)/100)

' Participation Rate
 series lfpr = 100*log(lf/n16)

' Variables for TMFP:
 series lks = 100*log(ks) - lpop
 series llqualt = 100*log(lqualt)
 series lveoa = 100*log(veoa)

' Series for the price equation
 series c1pcex = @pca(pcxfe)

```

series engylag_pcex = 0.5*(uces(-1)+uces(-2))*@pca(pcer(-1))
series sw_coreimp_pcex = 0.5*((emon/xgden)+(emon(-1)/xgden(-1)))*@pca(pmo/pcxfe)
series dum84 = @year>=1985
series frzbulg = 0    ''' Nixon wage-price control programs
smpl 1971q3 1974q1
frzbulg = 1
smpl 1974q2 1974q4
frzbulg = -3.666

```

```

' For the output-sector ratio, we use the median unbiased approach of Stock
' and Watson (1998). Thus, tau_oti is the ratio of the variances of the level
' and drift shocks.

```

```

scalar tau_oti = .033260 ' modifier for the drift totfactor (OSR*) error term

```

```

'*****

```

```

' 2. Prior starting values for states.

```

```

'   For mean zero level states, set prior to zero.
'   For other level states, set prior to a data-based
'       value near the start of the sample.
'   For most drift terms, set prior to zero.
'       Exception is trend MFP, set equal to sample average.

```

```

' Cycle

```

```

scalar    icycle =      0
scalar    icycle1 =      0
scalar    icycle2 =      0

```

```

' Measurement error

```

```

scalar    ie3p =      0
scalar    ie3i =      0

```

```

' Levels of trends

```

```

scalar    itotfactor =      @elem(gdp,%eststart) - @elem(bp,%eststart)
scalar    itmfp =      .965*(@elem(bp,%eststart) - 0.725*(@elem(eb,%eststart) _
      + @elem(bww,%eststart)) - 0.275*@elem(lks,%eststart) _
      - 0.725*@elem(llqualt,%eststart) _
      - (.035/.965)*@elem(lveoa,%eststart))
scalar    itww =      @elem(bww,%eststart)
scalar    ithtfactor =      @elem(eb,%eststart) - @elem(erate,%eststart) _
      - @elem(lfpr,%eststart)

```

```

scalar      itlfpr =          @elem(lfpr,%eststart)
scalar      iterate =        @elem(erate,%eststart)

' Initial drift terms
scalar      igtotfactor =      0.0
scalar      igtmfp =           1.7
scalar      igtww =            0.0
scalar      igthtfactor =      0.0
scalar      igtlfpr =          0.0

scalar nstate = 25

' The mpriors *must* be in the same order as the states are in the model object
vector(nstate) mpriors
mpriors.fill  0, ie3p, ie3i, 0, 0, _
              0, 0, icycle, icycle1, icycle2, _
              itotfactor, igtotfactor, itmfp, igtmfp, itww, _
              itww, igtww, ithtfactor, ithtfactor, igthtfactor, _
              itlfpr, itlfpr, igtlfpr, iterate, iterate

' Set starting values for variance priors
'   Variance priors set at a high value. In estimation, variance
'   drops sharply in early periods of estimation sample.

sym(nstate) vpriors
for !d = 1 to nstate
  vpriors(!d, !d) = 3
next

' A tighter prior for drift variances,
vpriors(12,12) = 1    ' igtotfactor
vpriors(14,14) = 1    ' igtmfp
vpriors(17,17) = 1    ' igtww
vpriors(20,20) = 1    ' igthtfactor

```

endsub

This code is written to file `srcEview/state.space.package/data.transformations.prg`.

Defines:

`data_transformations`, used in chunks 98, 205, and 208.

B.27 `srcEview/state.space.package/estimation.code.prg`

```

⟨srcEview/state.space.package/estimation.code.prg 201⟩≡
  subroutine ss_estimation()

```

```

' Subroutine for FRB/US state-space model package to
' estimate the model parameters.

' *****

' 1. Starting values for parameters.

' All of the coefficients in equations appear as beta or phi.
' Note that some values that are set here are hard coded in the estimation code below.

call initial_values

' *****

' Model

sspace {%modname} ' Declare a new state-space model object

' 2. Output equations.

' In the output equations, trend output is related to the capital stock, energy intensity,
' and trends for labor input using a production function. The parameters of the production
' function are hard-coded to the values in the FRB/US model.

' GDP observable
{%modname}.append @signal gdp = totfactor + tmfp/.965 + 0.725*(terate + tlfpr + thtfactor +
+ 0.275*lks + 0.725*llqualt + (.035/.965)*lveoa _
+ cycle + beta(11)*beta(6) + beta(11)*e3p + rexo
{%modname}.append @state rexo = [var=0.0000001^2]

' Buisnes sector product-side observable
{%modname}.append @signal bp = tmfp/.965 + 0.725*(terate + tlfpr + thtfactor + tww) _
+ 0.275*lks + 0.725*llqualt + (.035/.965)*lveoa _
+ beta(10)*cycle + beta(6) + e3p
{%modname}.append @state e3p = beta(602)*e3p(-1) + re3p
{%modname}.append @ename re3p 'business prod error
{%modname}.append @evar var(re3p) = (beta(125)^2)

' Business sector income-side observable
{%modname}.append @signal bi = tmfp/.965 + 0.725*(terate + tlfpr + thtfactor + tww) _
+ 0.275*lks + 0.725*llqualt + (.035/.965)*lveoa _
+ beta(10)*cycle - beta(6) + e3i

```

```

{%modname}.append @state e3i =    beta(602)*e3i(-1) + re3i
{%modname}.append @ename re3i 'business income error
{%modname}.append @evar var(re3i) =    (beta(126)^2)

' 3. Labor equations.

' Workweek observable
' To make this equation more consistent with the FRB/US workweek specification,
' a contemporaneous change in the cycle is included as well as the level of the
' cycle. In addition, the coefficient on the lagged gap term in this equation is
' hard-coded at a value similar to that implicit in the FRB/US model.

{%modname}.append @signal bww =    tww _
                                + phi(20)*(cycle-cycle1) _
                                + phi(22)*cycle1 _
                                + 0.82*(bww(-1)-tww1) _
                                + eww
{%modname}.append @state eww =    [var=beta(104)^2]

' Employment observable
{%modname}.append @signal eb =    terate + tlfpr + thtfactor + _
                                + phi(30)*cycle _
                                + phi(31)*(eb(-1)-(terate1 + tlfpr1 +thtfactor1)) _
                                + eeb
{%modname}.append @state eeb =    [var=beta(105)^2]

' Employment rate observable
{%modname}.append @signal erate =    terate + phi(50)*cycle _
                                + phi(51)*(erate(-1)-terate1) _
                                + eerate
{%modname}.append @state eerate =    [var=beta(106)^2]

' Labor force participation observable
{%modname}.append @signal lfpr =    tlfpr + phi(40)*cycle _
                                + phi(41)*(lfpr(-1)-tlfpr1) _
                                + elfpr
{%modname}.append @state elfpr =    [var=beta(107)^2]

' 4. Price observable
    {%modname}.append @ename ec
    {%modname}.append @evar var(ec) =    (beta(109)^2)
    {%modname}.append @signal c1pcex = _
    beta(401)*c1pcex(-1) + (1-beta(401))*(ptr(-1)+.1) _
    + beta(404)*(((.50*cycle + .33*cycle1 + .17*cycle2 )) ) _
    + beta(405)*@movav(engylag_pcex(-1),6) _
    + beta(406)*@movav(dum84*engylag_pcex(-1),6) _

```

```

    + beta(407)*frzbulg _
    + beta(408)*sw_coreimp_pcex + beta(409)*sw_coreimp_pcex(-1) _
    + ec

' 5. State equations

' Cycle state
{%modname}.append @state cycle = beta(1)*cycle(-1) + beta(2)*cycle1(-1) + ecycle
{%modname}.append @state cycle1 = cycle(-1)
{%modname}.append @state cycle2 = cycle1(-1)
{%modname}.append @ename ecycle
{%modname}.append @evar var(ecycle) = (beta(111)^2)

' Trends

' Totfactor = OSR* in F.R.(2011)
' tau_oti is taken from FR(2011)
{%modname}.append @state totfactor = totfactor(-1) + 0.25*.95*(gtotfactor(-1) ) + 0.25*.05*beta
                                etotfactor + 0.25*egtotfactor
{%modname}.append @state gtotfactor = .95*gtotfactor(-1) + .05*beta(213) + egtotfactor
{%modname}.append @ename etotfactor
{%modname}.append @evar var(etotfactor) = (beta(112)^2)
{%modname}.append @ename egtotfactor
{%modname}.append @evar var(egtotfactor) = (4*((tau_oti))*beta(112))^2

' Multi-Factor Productivity (OPH*)
{%modname}.append @state tmfp = tmfp(-1) + etmfp + 0.25*.95*(gtmfp(-1)) + 0.25*.05*beta(214)
{%modname}.append @state gtmfp = 0.95*gtmfp(-1) + 0.05*beta(214) + egtmfp
{%modname}.append @ename etmfp
{%modname}.append @evar var(etmfp) = beta(114)^2
{%modname}.append @ename egtmfp
{%modname}.append @evar var(egtmfp) = 0.14^2 ' beta(115)^2

' For the workweek, employment-sector ratio, and LFPR, the level variances are hard-coded.
' This hard-coding is done for convenience in production work. In particular, the imposed
' values are close to the values these parameters take on when they are freely estimated.
' However, the t-ratios for these parameters are not very high, and that imprecision leads
' to sluggish convergence; imposing these values shortens the time needed for estimation.

' Workweek (WW*)
{%modname}.append @state tww = tww(-1) + 0.25*.95*(gtww(-1) ) + etww + 0.25*.05*beta(216)+ 0.2
{%modname}.append @state tww1 = tww(-1)
{%modname}.append @state gtww = .95*gtww(-1) + .05*beta(216) + egtww
{%modname}.append @ename etww
{%modname}.append @ename egtww
{%modname}.append @evar var(etww) = 0.1^2

```

```

{%modname}.append @evar var(egttww) = beta(117)^2

' Employment Sector Ratio (ESR*)
{%modname}.append @state thtfactor = thtfactor(-1) + 0.25*.95*gtthtfactor(-1) + _
                                0.25*egthtfactor + ethtfactor
{%modname}.append @state thtfactor1 = thtfactor(-1)
{%modname}.append @state gtthtfactor = 0.95*gtthtfactor(-1) + egthtfactor
{%modname}.append @ename ethtfactor
{%modname}.append @evar var(ethtfactor) = .01^2 ' beta(118)^2
{%modname}.append @ename egthtfactor
{%modname}.append @evar var(egthtfactor) = (beta(119)^2)

' Labor Force Participation (LFPR*)
{%modname}.append @state tlfpr = tlfpr(-1) + 0.25*(.95*gtlfp(-1) + egtlfp) + etlfp
{%modname}.append @state tlfpr1 = tlfpr(-1)
{%modname}.append @state gtlfp = 0.95*gtlfp(-1) + egtlfp
{%modname}.append @ename etlfp
{%modname}.append @evar var(etlfp) = .05^2 ' beta(122)^2
{%modname}.append @ename egtlfp
{%modname}.append @evar var(egtlfp) = beta(123)^2

' Employment Rate (ER*), no drift
{%modname}.append @state terate = terate(-1) + eterate
{%modname}.append @state terate1 = terate(-1)
{%modname}.append @ename eterate
{%modname}.append @evar var(eterate) = beta(124)^2

{%modname}.append @vprior vpriors

{%modname}.append @mprior mpriors

freeze({%modname}_results) {%modname}.ml(m=100,showopts)

endsub

```

This code is written to file `srcEview/state.space.package/estimation.code.prg`.

Defines:

`ss.estimation`, used in chunk 205.

Uses `initial.values` 211.

B.28 srcEview/state.space.package/frbus.supply.estimation.prg

```

205 <srcEview/state.space.package/frbus.supply.estimation.prg 205>≡
    ' Program to estimate the FRB/US state-space model and
    ' generate estimates of model states.

    ' Subroutines:
    '   To transform data
    include "./data_transformations"

    '   To estimate the model
    include "./estimation_code"
    include "./initial_values"

    close @all
    wfcreate kf_data q 1949:1 2020:4

    %estend    = "2013q4"
    %eststart  = "1963q2"
    %datastart = "1949q1"

    sample estsample %eststart %estend
    sample datasample %datastart %estend
    %datasmpl = %datastart + " " + %estend
    %modname = "ss_model"

    ' *****
    ' Retrieve variables from the database

    ' Definitions of series in database
    '   See FRB/US model documentation for more complete descriptions

    ' XGDP   - GDP, cw 2009$
    ' XGDPN  - GDP
    ' XB     - BLS Business output, 2009$
    ' XBN    - BLS Business output
    ' XGDIN  - GDI

    ' PGDP   - Price index for GDP, cw
    ' PXB    - BLS Business price

    ' LEP    - Employment in business sector (employee and self-employed)

```

```

' LHP      - Aggregate labor hours, business sector (employee and self-employed)
' LUR      - Civilian unemployment rate (break adjusted)
' LF       - Civilian labor force (break adjusted)
' N16      - Noninstitutional population, aged 16 and over (break adjusted)

' KS       - Capital services, 2009 $
' LQUALT   - Labor quality, trend level
' VEOA     - Average energy-output ratio of existing capital stock

' PCXFE    - Price index for personal consumption expendits ex. food and energy, cw (N
' PCER     - Price index for personal consumption expenditures on energy (relative to I

' PMO      - Price index for imports ex. petroleum, cw
' UCES     - Energy share of nominal consumption expenditures
' EMON     - Imports of goods and services ex. petroleum
' XGDEN    - Nominal Absorption
' PTR      - 10-year expected inflation (Hoey/Philadelphia survey)

```

```

%dbin = "state_space_data"
dbopen %dbin as dbin

```

```

string varlist = " xgdpn xbn pgdp pxb pcxfe pcer xgdp xb lep uces emon xgden ptr "
varlist = varlist + " ks lqualt veoa lhp lur lf n16 pmo xgdin "

```

```

fetch(d=dbin) {varlist}

```

```

'*****
' Make model observable series, set priors

```

```

smpl datasample

```

```

call data_transformations

```

```

'*****
' Specify and estimate model, save for re-use

```

```

smpl estsample

```

```

statusline Estimating model
call ss_estimation

```

```

{%modname}.makefilter saved_results

```

```

'*****
' Create states

{%modname}.makestates(t=pred) *_prs      ' one-step ahead state predictions
{%modname}.makestates(t=filt) *_1        ' filter states
{%modname}.makestates(t=smooth) *_2      ' smoothed states
{%modname}.makestates(t=disturb) *err     ' estimate of the disturbances
{%modname}.makestates(t=filtse) *se      ' RMSE of the filtered states
{%modname}.makestates(t=smoothse) *se2    ' RMSE of the smoothed states

{%modname}.makesignals(t=pred) *_pr      ' one-step ahead signal predictions
{%modname}.makesignals(t=resid) *_res    ' error in one-step ahead signal predictions
{%modname}.makesignals(t=stdresid) *_sres ' standardized one-step ahead prediction residual

'*****
' Transformation into FRB/US mnemonics

series xbt = exp((tmfp_2/.965 + 0.725*(terate_2 + tlfpr_2 + thtfactor_2 + tww_2) + _
                0.275*lkls + 0.725*llqualt + (.035/.965)*lveoa + lpop)/100)
series hmfpt = gtmfp_2
series qlfpr = exp(tlfpr_2/100)
series hqlfpr = (gtlfpr_2/400)*qlfpr(-1)
series hqlww = gtww_2
series huxb = gtotfactor_2
series leppot = exp((tlfpr_2+terate_2+thtfactor_2)/100) * n16
series lurnat = 100*(1-exp(terate_2/100))
series qlww = exp(tww_2/100)
series uxbt = exp(totfactor_2/100)
series xgdpt = xbt * uxbt
series xgdo = exp(cycle_2/100)*xgdpt

wfsave saved_results_new

```

This code is written to file `srcEview/state.space.package/frbus.supply.estimation.prg`.
 Defines:

`frbus.supply.estimation`, used in chunk 98.

Uses `data.transformations` 198, `initial_values` 211, and `ss.estimation` 201.

B.29 srcEview/state.space.package/frbus.supply.filter.prg

```

208 <srcEview/state.space.package/frbus.supply.filter.prg 208>≡
    ' Program is part of the FRB/US state-space model package.
    ' frbus_supply_filter takes a previously estimated state-space model and
    ' generates estimates of model states.

    ' Subroutines:
    '   To transform data
    include "./data_transformations"

    close @all
    wfcreate kf_data q 1949:1 2020:4

    %estend      = "2014q3"
    %eststart    = "1963q2"
    %datastart   = "1949q1"

    sample estsample %eststart %estend
    sample datasample %datastart %estend

    %modname = "ss_model"

    '*****
    ' Retrieve variables from the database

    ' Definitions of series in database
    '   See FRB/US model documentation for more complete descriptions

    ' XGDP    - GDP, cw 2009$
    ' XGDPN   - GDP
    ' XB      - BLS NFB output, 2009$
    ' XBN     - BLS NFB output
    ' XGDIN   - GDI

    ' PGDP    - Price index for GDP, cw
    ' PXB     - BLS NFB price

    ' LEP     - Employment in nonfarm business sector (employee and self-employed)
    ' LHP     - Aggregate labor hours, nonfarm business sector (employee and self-employed)
    ' LUR     - Civilian unemployment rate (break adjusted)
    ' LF      - Civilian labor force (break adjusted)
    ' N16     - Noninstitutional population, aged 16 and over (break adjusted)

```



```

' KS      - Capital services, 2009 $
' LQUALT  - Labor quality, trend level
' VEOA    - Average energy-output ratio of existing capital stock

' PCXFE   - Price index for personal consumption expendits ex. food and energy, cw (NIA def.)
' PCER    - Price index for personal consumption expenditures on energy (relative to PCXFE)

' PMO     - Price index for imports ex. petroleum, cw
' UCES    - Energy share of nominal consumption expenditures
' EMON    - Imports of goods and services ex. petroleum
' XGDEN   - Nominal Absorption
' PTR     - 10-year expected inflation (Hoey/Philadelphia survey)

```

```

smpl datasample
%dbin = "./state_space_data"
dbopen %dbin as histdata
fetch(d=histdata) *

```

```

'*****
' Make model observable series, set priors

```

```

smpl datasample

```

```

call data_transformations

```

```

'*****
' Load saved model; run estimation step with previously estimated
' parameters as starting values (converges quickly).

```

```

smpl estsample

```

```

wfoopen saved_results
copy saved_results::untitled\saved_results kf_data::untitled\{%modname}
wfclose saved_results
freeze({%modname}_results) {%modname}.ml(m=100,showopts)

```

```

'*****
' Create states

```

```

{%modname}.makestates(t=pred) *_prs      ' one-step ahead state predictions
{%modname}.makestates(t=filt) *_1        ' filter states
{%modname}.makestates(t=smooth) *_2      ' smoothed states
{%modname}.makestates(t=disturb) *err    ' estimate of the disturbances

```

```

{%modname}.makestates(t=filtse) *se      ' RMSE of the filtered states
{%modname}.makestates(t=smoothse) *se2   ' RMSE of the smoothed states

{%modname}.makesignals(t=pred) *_pr      ' one-step ahead signal predictions
{%modname}.makesignals(t=resid) *_res    ' error in one-step ahead signal prediction
{%modname}.makesignals(t=stdresid) *_sres ' standardized one-step ahead prediction

'*****
' Transformation into FRB/US mnemonics

series xbt = exp((tmfp_2/.965 + 0.725*(terate_2 + tlfpr_2 + thtfactor_2 + tww_2) +
                 0.275*lkls + 0.725*llqualt + (.035/.965)*lveoa + lpop)/100)
series hmfpt = gtmfp_2
series qlfpr = exp(tlfpr_2/100)
series hqlfpr = (gtlfpr_2/400)*qlfpr(-1)
series hqlww = gtww_2
series huxb = gtotfactor_2
series leppot = exp((tlfpr_2+terate_2+thtfactor_2)/100) * n16
series lurnat = 100*(1-exp(terate_2/100))
series qlww = exp(tww_2/100)
series uxbt = exp(totfactor_2/100)
series xgdpt = xbt * uxbt
series xgdo = exp(cycle_2/100)*xgdpt

```

This code is written to file `srcEview/state.space.package/frbus.supply.filter.prg`.

Defines:

`frbus.supply.filter`, used in chunk 98.

Uses `data.transformations` 198.

B.30 srcEview/state.space.package/initial.values.prg

211 *<srcEview/state.space.package/initial.values.prg 211>*≡
 subroutine initial_values

```

coef(703) beta
coef(60) phi
beta(1)      = 1.497516
beta(2)      = -0.540927
beta(213)    = -0.328852
beta(214)    = 1.061453
beta(216)    = -0.266658
beta(10)     = 1.325310
beta(401)    = 0.625693
beta(404)    = 0.090059
beta(405)    = 0.440360
beta(406)    = -0.324208
beta(407)    = -0.435237
beta(408)    = 0.293555
beta(409)    = 0.253904
beta(6)      = 0.331366
beta(602)    = 0.927692

phi(20)      = 0.301616
phi(22)      = 0.020651
phi(30)      = 0.452297
phi(31)      = 0.649965
phi(40)      = 0.042475
phi(41)      = 0.727876
phi(50)      = 0.299981
phi(51)      = 0.528947

beta(105)    = 0.176896
beta(106)    = 0.086226
beta(107)    = 0.210419
beta(104)    = 0.229143
beta(100)    = 0.000001
beta(109)    = 0.803236
beta(111)    = 0.566205
beta(119)    = 0.099423
beta(123)    = 0.129325
beta(115)    = 0.140001
beta(117)    = 0.081062
beta(124)    = 0.136532
beta(118)    = 0.010000
```

```
beta(122)      = 0.050000
beta(114)      = 0.107852
beta(112)      = 0.044151
beta(126)      = 0.476765
beta(125)      = 0.534807
beta(11) = 1/beta(10)
```

```
endsub
```

This code is written to file `srcEview/state.space.package/initial.values.prg`.

Defines:

`initial_values`, used in chunks 98, 201, and 205.

Appendix C

Notes, Bibliography and Indexes

C.1 Chunks

<check.sh 98>
<compute derivatives of loss function targets wrt instruments 81>
<compute derivatives of mce targets wrt mce instruments 46>
<convert mcz inequality constraints 85>
<copy it 153>
<copy series into group 10>
<create common variables, strings, matrices, vectors, and tables 33>
<create new group 12>
<create wage and expectation variables in forward looking model 94>
<determine default method, linesearch, and other options 28>
<determine endogenous and exogenous variables 25>
<find next delimiter 89>
<find variable description 162>
<form table 173>
<graph it 155>
<interpolate unavailable observations 13>
<load frbus coefficients 99b>
<load frbus coefficients call 100a>
<load frbus equations 101>
<load frbus equations call 102a>
<load frbus with transformed subsidiary model 90>
<load mce coefficients 103>
<load mce coefficients call 104>
<load mce equations 106>
<load mce equations call 109>
<load some equations 111>

<load some equations call 113>
 <load variable information 115>
 <load variable information call 116a>
 <main optimal control simulation with inequality constraints 68>
 <main unconstrained optimal control simulation 64>
 <make statistics 174>
 <master library routines 175b>
 <mce solve library 177>
 <mce solve library change history 16>
 <model consistent armijo optimization rule 54>
 <model consistent coefficient non-monotone step-length procedure 52>
 <model consistent coefficient simulation 35>
 <model consistent optimal time-consistent solution 73>
 <names of all series in group 11b>
 <parse options containing equal signs 32a>
 <parse options not containing equal signs 32b>
 <plot historical residuals of key equations 160>
 <plot it 154>
 <quarterly date string shift 9>
 <run model consistent expectations 18>
 <set fiscal policy option 14>
 <set group to zero 11a>
 <set monetary policy fed funds rate 15b>
 <set monetary policy option 15a>
 <set options based on defaults and overrides 57>
 <setup.sh 97>
 <shift left 88>
 <simulate six ping simulations, aka simple IRFs 145>
 <solve model consistent expectations model 83>
 <solve model consistent instrument values 44>
 <srcEview/frbus.package/addins/ld.frbus.cfs/ld.frbus.cfs.prg 99a>
 <srcEview/frbus.package/addins/ld.frbus.eqs/ld.frbus.eqs.prg 100b>
 <srcEview/frbus.package/addins/ld.mce.cfs/ld.mce.cfs.prg 102b>
 <srcEview/frbus.package/addins/ld.mce.eqs/ld.mce.eqs.prg 105>
 <srcEview/frbus.package/addins/ld.some.eqs/ld.some.eqs.prg 110>
 <srcEview/frbus.package/addins/ld.varinfo/ld.varinfo.prg 114>
 <srcEview/frbus.package/addins/regadd.prg 116b>
 <srcEview/frbus.package/programs/example1.prg 117>
 <srcEview/frbus.package/programs/example2.prg 120>
 <srcEview/frbus.package/programs/example3.prg 124>
 <srcEview/frbus.package/programs/example4.prg 130>
 <srcEview/frbus.package/programs/ocpolicy.prg 136>
 <srcEview/frbus.package/programs/pings.prg 144>
 <srcEview/frbus.package/programs/plot.resids.prg 159>
 <srcEview/frbus.package/programs/stochsim.prg 163>
 <srcEview/frbus.package/subs/master.library.prg 175a>

<srcEview/frbus.package/subs/mce.solve.library.prg 176>
 <srcEview/mce.solve.package/example1.prg 181>
 <srcEview/mce.solve.package/example2.prg 184>
 <srcEview/mce.solve.package/example3.prg 187>
 <srcEview/mce.solve.package/example4.prg 191>
 <srcEview/mce.solve.package/example5.prg 194>
 <srcEview/mce.solve.package/mce.solve.library.prg 197>
 <srcEview/state.space.package/data.transformations.prg 198>
 <srcEview/state.space.package/estimation.code.prg 201>
 <srcEview/state.space.package/frbus.supply.estimation.prg 205>
 <srcEview/state.space.package/frbus.supply.filter.prg 208>
 <srcEview/state.space.package/initial.values.prg 211>
 <stochastic simulations under variable expectations 164>
 <variable terminal values 55>

C.2 Index

copyit: 145, [153](#)
 data_transformations: 98, [198](#), 205, 208
 dateshift: [9](#), 13, 73, 124, 136
 example1: 98, [117](#), [181](#)
 example2: 98, [120](#), [184](#)
 example3: 98, [124](#), [187](#)
 example4: 98, [130](#), [191](#)
 example5: 98, [194](#)
 find_next_delimit: 85, [89](#)
 find_var_description: 160, [162](#)
 frbus.supply.filter: 98, [208](#)
 frbus_supply_estimation: 98, [205](#)
 graphit: 145, [155](#)
 group2group: [10](#)
 group2zero: [11a](#), 117, 120, 124, 130, 136, 145, 160, 164
 groupnames2string: [11b](#), 164
 groupnew: [12](#), 117, 120, 124, 130, 136, 145, 160, 164
 initial.values: 98, 201, 205, [211](#)
 interp_lin: [13](#), 164
 ld_frbus_cfs: 90, 98, [99a](#), [99b](#), 100a, 116b, 117, 124, 160, 164
 ld_frbus_eqs: 90, 98, [100b](#), [101](#), 102a, 116b, 117, 124, 160, 164
 ld_mce_cfs: 90, 98, [102b](#), [103](#), 104, 116b
 ld_mce_eqs: 90, 98, [105](#), [106](#), 109, 116b
 ld_some_eqs: 90, 98, [110](#), [111](#), 113, 116b
 ld_varinfo: 98, [114](#), [115](#), 116a, 116b, 160, 164
 make_frbus_mcevars: 16, [94](#), 120, 130, 136, 145
 makestats: 164, [174](#)
 mce_load_frbus: 16, [90](#), 120, 130, 136, 145

mce_run: 16, [18](#), 33, 35, 120, 130, 136, 145, 181, 184, 187, 191, 194
mce_solve_library: 98, 120, 130, 136, 145, [176](#), 181, 184, 187, 191, 194, [197](#)
mcz_algo: 18, [28](#)
mcz_armijo: 35, [54](#)
mcz_constraints: 57, [85](#)
mcz_derivs: 33, 35, [46](#)
mcz_equalopt: 18, 28, [32a](#), 35, 57, 90
mcz_hasopt: 18, [32b](#), 35, 57
mcz_lmr: 35, [52](#)
mcz_opt: 57, [64](#)
mcz_opt_deriv: 64, 68, 73, [81](#)
mcz_opt_qp: 16, 57, [68](#)
mcz_opt_setup: 18, [57](#)
mcz_opt_solve: 64, 68, [83](#)
mcz_opt_tc: 16, 57, [73](#)
mcz_parsemod: 18, [25](#)
mcz_sim: 16, 18, [35](#), 68, 73, 81, 83
mcz_sim_setup: 18, [33](#)
mcz_solvit: 16, 33, 35, [44](#), 52, 54
mcz_terminal: 44, [55](#)
ocpolicy: 98, [136](#)
pings: 98, [144](#), 145
plot_resids: 98, [159](#)
plotit: [154](#), 155
regadd: 98, [116b](#)
set_fp: [14](#), 117, 120, 124, 130, 136, 145, 164
set_mp: [15a](#), 117, 120, 124, 130, 136, 145, 164
set_mpvars2rff: [15b](#)
shiftleft: 35, [88](#)
ss_estimation: [201](#), 205
stochsim: 98, [163](#)
tableform: 164, [173](#)