FRB/US Simulation Basics

June 2018

This note covers some basic information needed to set up and execute simulations with the FRB/US model. Reference is made to many of the programs provided in the *programs* subdirectory:

- *example1.prg* runs a simple simulation under backward-looking VAR expectations;
- *example2.prg* runs a simple simulation under model-consistent (MC) expectations;
- *example3.prg* illustrates how to run a simulation in which the outcomes for the federal funds rate are constrained by the zero lower bound (ZLB) and also by threshold conditions that may delay the rise of the funds rate from the ZLB;
- *example4.prg* runs a simulation under MC expectations using a monetary policy rule that is not one of the policy alternatives included in FRB/US and that requires the addition of new MC expectations variables to FRB/US;
- *pings.prg* simulates impulse responses to key shocks under alternative assumptions about expectations formation;
- *ocpolicy.prg* runs a simulation in which the path of the federal funds rate is determined by optimal-control techniques to minimize a quadratic loss function.

These programs frequently use EViews string variables to define the names of key inputs (eg, models, databases, simulation ranges) so that the programs can be modified easily to work with alternative inputs.

1. Expectations overview

FRB/US contains about twenty-five expectations terms, each of which is coded as a separate variable with its own equation. The name of each expectations variable starts with a Z: for example, *zpicxfe* is the expectation of next quarter's value of *picxfe*, which is the rate of inflation of core consumer prices. Each expectations variable can have either a VAR or MC solution. A VAR solution is based on a fixed-coefficient equation whose structure is consistent with the historical interrelationships among a small set of key macro variables and whose explanatory variables are observed contemporaneously or with a lag. In an MC solution, the expectation is required to be the same as the solved future value of the underlying variable. The MC equation for *zpicxfe* is simply *zpicxfe = picxfe(+1)*. The design of the FRB/US simulation programs permits some Z variables to have VAR solutions and others to have MC solutions.

EViews uses the Fair-Taylor (FT) algorithm to solve models with future-dated variables, but this method does not reliably or efficiently solve FRB/US when its expectations are MC. As a result, two more powerful MC algorithms have been specifically developed for EViews simulations of models with MC expectations; they are coded as EViews subroutines.[1] Because the inputs to these alternative algorithms are quite different from the inputs to the built-in EViews model *solve* procedure, the code

[1] See Flint Brayton, "Two Practical Algorithms for Solving Rational Expectations Models," Finance and Economics Discussion Series 2011-44. (http://www.federalreserve.gov/pubs/feds/2011/201144/201144pap.pdf)

required for a FRB/US simulation with one or more MC expectations differs from the code for a simulation in which all expectations are VAR-based. Specifically, the MC algorithms require two models, one containing the full set of FRB/US equations in which all expectations are coded using the VAR-based backward-looking identities, and one with a partial set of equations consisting of the forward-looking MC identities and an associated set of identities that defines expectations errors as the difference between the forward-looking and backward-looking solutions. The algorithms alternate between solutions of each model, adjusting the add factors (constant adjustments) on the backward-looking equations for those expectations that are MC after each pair of simulations, until the MC expectations errors are zero. The default *newton* algorithm uses a Newton approach to calculating the constant adjustments. The alternative *qnewton* algorithm uses a version of Broyden's quasi-Newton method. Both MC algorithms are extended path methods and thus require considerably longer simulation periods than those used for VAR expectations, even if the period for which solution values are actually of interest is the same.

2. Subroutines

The *subs* subdirectory of the FRB/US Model Package contains two libraries of subroutines. The subroutines in *master_libary.prg* must be made available to all simulations programs. The subroutines in *mce_solve_library.prg* must be made available to all programs that run simulations with MC expectations.

3. Using the *read_xml_model* add-in to load equations and coefficients

FRB/US equations and coefficients are stored in the *model.xml* file in the *mods* subdirectory. The provided programs use the *read_xml_model* add-in command to load this information into EViews. Instructions for installing this add-in can be found in the README.TXT file. The following table shows the input parameters available for use with the add-in.

read_xml_model input parameters

path	required	path to XML file
vars	optional (default is all eqs)	"var1 var2 …" or "-allbut var1 var2 …"
mod_f	required for MCE simulations	name to be given to MCE model
mce_vars	required for MCE simulations	"-all", "-mcap", "-mcap+wp", "-wp", "var1 var2 var3 …", "-allbut var1 var2 …"

For use in a simulation in which all expectations are VAR-based, the following code (*example1.prg*) loads all FRB/US non-expectations equations and the VAR-based formulas for all expectations variables into a single model. The model is automatically named *stdver*:

```
%modelpath = "../mods/model.xml"
read_xml_model(path=%modelpath)
```

The block of code shown next illustrates how to instruct *read_xml_model* to set up the two models required for a simulation with MC expectations. In this example (*example2.prg*), all expectations are MC

(%mcegroup="-all"), the first model (named *stdver* automatically) contains all FRB/US non-expectations equations and the VAR-based formulas for all expectations equations, and the second model (named *pfver*) contains the MC formulas for all expectations variables and an associated set of MC error equations. In *pfver*, the names of MC variables are modified so that the initial Z is replaced with W, in order to distinguish the expectations solutions of *pfver* from those of *stdver*, and the names of MC error variables have an E prefix.

```
%modelpath = "../mods/model.xml"
%mcemod = "pfver"
%mcegroup= "-all"
read_xml_model(path=%modelpath, mce_vars=%mcegroup,mod_f=%mcemod)
```

In addition to the "`-all`" designation, three other MCE group choices are available. The "`-mcap`" setting declares that only the expectations in asset pricing equations have MC solutions. The "`-wp`" setting declares that only the expectations in the price and wage setting equations have MC solutions. The "`-mcap+wp`" setting combines the "`-mcap`" and "`-wp`" groups. In addition to the MCE group options, the *mce_vars* parameter can also be assigned to a list of individual expectations variables to include or exclude as MC.

read_xml_model MCE variable groups

-all		
-mcap+wp		Other
-mcap	-wp	
zdivgr zgap05 zgap10 zgap30 zrff5 zrff10 zrff30 zpi10 zpi10f zpic30 zpib5 zpic58	zpicxfe zpieci	zecd zeco zeh zgapc2 zlhp zpi5 zebfi zyh zyhp zyht zynid

The optional *vars* parameter that is available in *read_xml_model* is used to load a subset of FRB/US equations. As shown in *example4.prg*, the commands

```
%vars=  "-allbut rffgen"
read_xml_model(path=%modelpath, mce_vars=%mcegroup,mod_f=%mcemod,vars=%vars)
```

load all equations but the one for *rffgen*.

In addition to creating model objects and coefficient vectors in the EViews workfile, *read_xml_model* creates several string variables.

read_xml_model output strings

m_model_name	name of the VAR-expectations model
m_variable_names	list of all variables in the VAR-expectations model
m_stoch	declaration of endogenous variables that are shocked in stochastic simulations
m_zvars	list of expectations variables designated MC (MCE simulations only)

The *m_zvars* output string is used by programs that run simulations in which at least some expectations are MC. The *m_stoch* string is used by programs that run stochastic simulations.

4. Data

The *data* subdirectory contains the *longbase* database, which is in EViews format and contains historical observations on all FRB/US variables and projected observations that extend more than 100 years in the future. For the first few years of the projection the values of a few key macro variables are based on the most recent "Economic Projections of Federal Reserve Board Members and Federal Reserve Bank Presidents" (SEP). Thereafter, the projections gradually converge to an illustrative but arbitrary steady-state growth path.[2] In the example programs, the string *%dbin* declares the database name.

As mentioned in the previous section, simulations with MC expectations require the use of a second model whose equations contain special W and E variables. These variables are not part of the input database. In all programs that run MC simulations, such as *example2.prg*, the *mce_create_auxillary* subroutine is called to create data for these variables. It sets the W variables equal to the Z variables and the E variables to zero. The subroutine requires the argument *m_zvars*, which is a string created by *read_xml_model* of the names of the expectations variables that are MC.

5. Monetary policy

There are seven basic options for setting the federal funds rate (*rff*). The options consist of five policy reaction functions and two exogenous policies that hold the nominal funds rate or the real funds rate at a pre-determined path. A specific option is chosen by setting one of an associated set of seven exogenous switch variables to 1 and the others to 0. The follow table indicates the names of the switches, their descriptions, and for the options associated with policy rules, the name of the associated rule equation. For more information see the HTML reference document on *FRBUS Equations*.

Switch variable	Description	Reaction function equation
dmpintay	inertial Taylor rule	rffintay
dmptay	Taylor rule	rfftay
dmptlr	Taylor rule with unemployment gap	rfftlr
dmpalt	estimated rule	rffalt
dmpgen	generalized rule	rffgen
dmpex	exogenous nominal funds rate	na
dmprr	exogenous real funds rate	na

[2] The initial parts of the projections of real GDP growth, core and overall PCE price inflation, the unemployment rate and the federal funds rate are based on the SEP. All other aspects of the projections included in the database should not be construed as reflecting the views of the FOMC or any of its participants.

The values of the switch variables can be specified directly or via the subroutine *set_mp*. For example, the command

```
call set_mp("dmpintay")
```

selects the inertial Taylor rule by setting *dmpintay*=1 and all other switches to 0. The subroutine obeys the current workfile sample, so it is possible, for example, to set up a simulation in which the federal funds rate is exogenous initially and then switches to one of the policy rules by executing a pair of calls to the subroutine for different sample periods.

The stability of FRB/US solutions over the lengthy simulation periods needed when expectations are MC requires that one of the endogenous policy rules be in force from some point not too far from the start of the simulation through its end.

The ZLB and thresholds (example3.prg)

The chosen basic option can be modified so that the outcome for the federal funds rate is subject to the zero lower bound (ZLB). Setting the exogenous variable *rffmin* to zero (or, more realistically, a small positive value) imposes the ZLB. Setting *rffmin* to a large negative number eliminates the constraint.

In addition, the timing of the liftoff of the federal funds rate from the ZLB can be determined by a version of the threshold criteria that appear in FOMC statements from December 2012 to January 2014. In this case, liftoff is delayed until either the unemployment rate falls below a threshold (given by the value of *lurtrsh*) or the expected inflation rate rises above a threshold (given by the value of *pitrsh*), when the exogenous switch variable *dmptrsh* = 1. Once either threshold is crossed, the endogenous switch variable *dmptr* becomes 1.0 and the policy rate is determined, with a one-quarter delay and subject to the ZLB, by the chosen policy option, and it continues to be set according to that option irrespective of the subsequent outcomes for unemployment and expected inflation.

Care needs to be exercised in the use of either the ZLB or the liftoff thresholds in simulations that impose baseline-tracking add factors. If the add factors on various key equations in the monetary sector are not zero (including the equations for *rff, rffrule, dmptlur, dmptpi, dmptmax*, and *dmptr*), the effects of the zero bound restriction and the threshold conditions may not be those intended. Moreover, because the equations for *dmptlur, dmptpi, dmptmax,* and *dmptr* are designed to restrict their values to the [0,1] interval, these equations should never have non-zero add factors.

The structure of the endogenous switch variable *dmptr* requires further comment. In the equation, *dmptr* equals to the maximum of its own lag, a switch variable (*dmptlur*) indicating whether the unemployment rate has fallen below its threshold, and a switch variable (*dmptpi*) indicating whether expected inflation has risen above its threshold. The inclusion of the own lag has the effect of making the threshold restrictions a one-time event, but it requires that the value of *dmptr* be zero in the quarter prior to one in which it is desired to have threshold conditions affect the solution value of the funds rate. If this quarter is the first simulation period, then it is simply a matter of setting the baseline

data on *dmptr* to zero in the quarter prior to the start of the simulation. Alternatively, in a simulation in which, for example, the unemployment rate is below its threshold initially and one wants to impose the threshold conditions only after the labor market weakens sufficiently, then the desired condition for *dmptr* also requires that the initial values of threshold variable *lurtrsh* (and perhaps *pitrsh* as well) be set to extreme values --- such as -9999 for *lurstsh* and +9999 for *pitrsh*. This is illustrated in *example3.prg.*

Equilibrium real federal funds rate for monetary policy reaction functions (rstar)

The monetary policy equations for *rffintay, rfftay, rfftlr,* and *rffgen* contain the variable *rstar*, which can be interpreted as the policymakers' estimate of the equilibrium real funds rate. The behavior of *rstar* is controlled by the value of the exogenous switch variable *drstar*. When *drstar =0*, *rstar* is exogenous; when *drstar=1*, *rstar* moves gradually toward the simulated value of the real funds rate. Because *drstar* is a timeseries, the value of *rstar* may be fixed during part of a simulation and allowed to vary endogenously during the other part. The endogenous setting is usually the best choice, at least after some simulation time has elapsed, in any experiment that changes the value of the equilibrium real rate of interest – which in FRB/US includes simulations of most types of permanent shocks. For simulations of transitory shocks, it may also be desirable to make *rstar* endogenous after some point, especially if the simulation interval is lengthy either by choice under VAR expectations or by necessity under MCE. Among the provided programs, *example2.prg*, *example4.prg* and *pings.prg* shift *rstar* from exogenous to endogenous after 40 simulation quarters.

6. Fiscal policy

The exogenous switch variables *dfpex, dfpsrp,* and *dfpdbt* can be specified directly or via the subroutine *set_fp* to select one of three possible settings for the behavior of the trend federal and state and local personal income tax rates, *trfpt* and *trspt*. When *dfpex* = 1.0 and the other switches are zero, the trend tax rates are exogenous; when *dfpsrp* = 1.0, the trends adjust to gradually stabilize the ratios of the two government surpluses to GDP at the values given by exogenous variables *gfsrt* and *gssrt*; when *dfpdbt* = 1.0, the trends adjust to gradually stabilize the debt-to-GDP ratios of the each government sector at the values given by the exogenous variables *gfdrt* and *gsdrt*.

The syntax for use of the subroutine is *set_fp("dfp…")*. For example,

```
call set_fp("dfpsrp")
```

selects surplus stabilizing fiscal policy by setting *dfpex*=0*, dfpsrp*=1*,* and *dfpdbt*=0. The subroutine obeys the current workfile sample, so it is possible, for example, to set up a simulation in which fiscal policy is exogenous initially and then switches to surplus targeting by executing a pair of calls to the subroutine for different sample periods.

The lengthy simulation periods associated with FRB/US simulations with MC expectations require that one of the endogenous fiscal policy options be in force from some point not too far from the start of the simulation through its end.

7. Add factors and baseline tracking

Each of the example programs sets up FRB/US so that all of its equations are assigned add factor variables (*addassign*) and their values are initialized (*addinit*) such that, in the absence of shocks, the simulated outcomes for all endogenous variables replicate the baseline database. Because the supplied database does not generally impose the various MC identities, the expectations in MC simulations with baseline tracking are MC only when expressed as deviations from baseline.

The EViews *addassign* command creates an add factor variable for each equation whose name is constructed by adding the suffix *_a* to the name of dependent variable. Each FRB/US equation also comes coded with a separate add factor variable whose name contains *_aerr* as a suffix. If the *_aerr* variables do not have values in the input database, they need to be given values (typically zero) in the simulation program.

8. Simulation

The default EViews model *solve* options are typically not precise enough for satisfactory solutions of FRB/US. The following command specifies a better choice of options.

```
{%varmod}.solveopt(o=b,g=12,z=1e-12)
```

(The string *%varmod* refers to a previously defined name of a FRB/US version.) FRB/US simulations that use the EViews quasi-Newton Broyden algorithm (*o=b*) usually converge successfully, although some of the equations in FRB/US contain functions with kinks, which is a situation which can be problematic for Newton-type solution algorithms. When convergence fails, especially if a "solver stalled" error message occurs, switching to the EViews Newton (*o=n*) or Gauss-Seidel (*o=g*) algorithms may work.

As noted earlier, the mechanics of running a FRB/US simulation with VAR expectations are different from those associated with MC expectations.

VAR expectations

Given the assumption of VAR expectations in *example1.prg*, its code for simulating the effects of a one-time 100 basis point upward shock to the chosen monetary policy rule (*rffintay)* can use the EViews model *solve* procedure and thus is quite simple.

```
smpl %simstart %simstart
rffintay_aerr = rffintay_aerr + 1
smpl %simstart %simend
{%varmod}.solve
```

The shock is applied to *rffintay_aerr*, but it could have been applied to *rffintay_a* with the same effect.

MC expectations

The code in *example2.prg* requires a call to the *mce_run* subroutine, instead of the EViews *solve* command, because of the assumption that some expectations are MC.

```
smpl %simstart %simstart
rffintay_aerr = rffintay_aerr + 1
%modstr = "mod_b=%varmod,mod_f=%mcemod,mce_vars=m_zvars"
%algstr = "meth=qnewton"
%simstr = "type=single"
smpl %simstart %simend
call mce_run(%modstr,%algstr,%simstr)
```

A comprehensive description of the syntax and uses of the *mce_run* subroutine is contained in the reference document *MCE Solve Users' Guide.* The subroutine has three arguments, each of which is a string. The first string *(%modstr*) contains information about the model, which can be organized in several different ways. The example uses a syntax that requires the name of the full model with VAR expectations (*mod_b* keyword), the name of the partial model with MC expectations (*mod_f* keyword), and the names of the expectations variables that are MC (*mce_vars* keyword, which is assigned the value of the *m_zvars* string created by the *read_xml_model* add-in). The second string (*%algstr*) contains information about the MC solution algorithm. The example selects the quasi-Newton algorithm, which is usually the fastest of the available algorithms when running a single FRB/US simulation. The third string (*%simstr*) selects the type of simulation. Possible types include single (*single*) and optimal control (*opt*, *opttc*) simulations. The four *example* programs execute *single* simulations. The program named *ocpolicy.prg* illustrates how to set up and run an optimal control program of the *opt* type.

Any FRB/US simulation with MC expectations requires that the simulation period be long enough that the solved values over the period of interest are unaffected by the adding an additional quarter to the simulation period. For a simulation of a transitory shock in which the period of interest is the first five years (20 quarters) of the simulation, a range of 60 years (240 quarters) is usually sufficient, though if the shock is highly persistent, a longer span may be necessary. Simulations of permanent shifts, such as a change in the target rate of inflation or the target ratio of federal debt to GDP, may also require longer simulation periods as well as the resetting of terminal conditions, as described in the next paragraph.

A FRB/US simulation with MC expectations requires terminal conditions for all variables whose future values appear in the partial model with the MC identities. In the case of the MC equation that was noted earlier, *zpicxfe = picxfe(+1)*, the value of *picxfe* must be defined in the first quarter after the end of the simulation. Some of the other MC identities contain as many as four leads. For simulations that do not affect the long-run values of the specific variables having leads, the values of these variables in the input database should provide satisfactory terminal conditions.[3] On the other hand, in simulations that do alter the set of long-run values, which includes simulations of most permanent shocks as well as a few types of transitory shocks, adjusting the baseline values of terminal data can significantly improve the simulation's convergence characteristics. Adjustments of this type are done automatically when the

[3] Given a maximum lead of four quarters, MC simulations must end no later than four quarters before the end of the input database.

%simstr argument to the *mce_run* subroutine contains the keyword *terminal*. When this option is invoked, terminal values are adjusted in the first simulation iteration based on the deviations from baseline of the far-end solution values taken from the version of the model in which all expectations are VAR-based. Although an approximation, this procedure usually works satisfactorily, because FRB/US has essentially the same long-run properties under VAR expectations as it does under MC expectations.

The *terminal* option is invoked in *pings.prg* when all expectations are specified as MC. Resetting terminal values is beneficial in this case, because the program's simulations of shocks to the level and growth rate of multi-factor productivity both alter the long-run levels of some variables requiring terminal values in the consumption and investment sectors.

9. Modifying the structure of FRB/US

There are two approaches to making changes to FRB/US equations. In one approach, changes are made after the *read_xml_model* add-in has been used to load the model into EViews, such as directly changing the values of a coefficient vector (*eg,* y_rfftay(3)=2) or, in EViews versions 8 or later, using the built-in *replace*, *append*, *merge*, and *drop* procedures to modify the functional forms of equations.

In the second approach to making changes to FRB/US equations, the *vars* parameter of the *read_xml_model* add-in is used to load only those equations that do not need modifications. Then, additional EViews commands are used to *append* or *merge* the new specifications of the remaining equations. This approach must be used when making changes to functional forms in EViews 7 and may be optionally used for this purpose in later versions of EViews. In the block of code that follows,

```
%vars = "-allbut rff"
read_xml_model(path=%modelpath,vars=%vars)
{%varmod}.append rff = <an alternative text specification>
```

all equations but the one for *rff* are loaded and an alternative text specification for RFF is appended for use in a simulation with VAR expectations. The code for a simulation with MC expectations requires a similar use of the *vars* parameter (*example4.prg*).

Modifications to FRB/US that introduce new MC expectations variables need special attention to ensure that they conform to the design of the MCE algorithms supplied in the FRB/US Model Package. As mentioned earlier, these algorithms require two models, one containing the full set of FRB/US equations in which all expectations are coded using backward-looking identities, and one with a partial set of equations consisting of the forward-looking MCE identities and an associated set of identities that defines expectations errors as the difference between the forward-and backward-looking solutions. In this framework, each MCE variable is associated with three equations and three variable names. The name of an expectations variable when it appears as the dependent variable in its backward-looking equation in the first model (and when it appears as an explanatory variable in other equations in that model) should start with the letter Z. Its name when it appears as the dependent variable in its forward-looking equation should replace the initial Z with a W. And the name of its expectations error should be formed by appending an E to the Z form of its name. See *example4.prg* for a concrete example of how to add new MCE variables.