MACFIXER – Macro Variable Fixing Program for InterDyme Models

Inforum
September 2009

Fixes, as used here, are ways to make a model work the way we want it to, not necessarily the way that emerges from its equations. The power that fixes give over a model can certainly be, and often has been, abused. Nonetheless, they have a legitimate role. Suppose, for example, we wish to consider the impacts of some event which the equations never dreamed of, like a natural disaster or a massive overhaul of the health care system. Then a fix is the natural way to convey to the model that the equations are not to be entirely trusted.

Interdyme has three types of fixes, those for macro variables, those for vectors and matrices, and a special type for industry outputs.

Macro Variable Fixes

Macro variable fixes are fixes applied to variables of type Tseries, which are defined using the Idbuild program described above. These fixes work very like those of models built with the G-Build combination, but also have much in common with the vector fixes described in the next section. The program that handles the macro variable fixes is called MacFixer. The input to MacFixer is a file prepared by the user with a text editor. It should have the extension .mfx . Once this file has been created, the program MacFixer is run by Model | MacFixer on the G main menu. The results of this program are written to a "macro fix bank", which is essentially a G bank, which can be read with G. The root name (the part of the filename before the dot) of the macro fix G bank is passed to MacFixer through the form which the above G command opens. It must also be passed to the simulation program by the form that opens on the command Model | RunDyme .

MacFixer requires a configuration file, called MACFIXER.CFG. It is created by G from the information provided on the form opened by the Model | MacFixer command. This form requires the name of the text input file, the root name of the G bank file used for base values for the index and growth-rate fixes (this would normally be the G bank created for use with the simulation program), the name of the G bank which will contain the values of the fixes, and the name of the output check file. This last file shows the values of each fix in each year, and serves as a check on the results in the binary file.

While it is up to the user to name files, it makes good sense to give files for the same simulation the same root name. A simulation that involves low defense expenditures, for example, could have a G bank file called LOWDEF.BNK, and a .mfx file called LOWDEF.MFX.

There are several varieties of macrofixes that may be given, and they are described in the list below:

skip

is the simplest type of fix. It simply skips the equation and uses the values in the model G bank. For example:

```
skip invn$35
```

would skip the equation for the macro variable invn$35, and use the value already in the model G bank.

ovr

overrides the result of the equation with the value of the time series given. Values between given years are linearly interpolated. In the example below, the macro fix program would calculate and override a fix series that starts in 1992, ends in 2000, and moves in a straight line between the two points. For example,

```
ovr uincome$
  92  154.1
  2000  182.3;
```

would override the value of the forecast of uincome$ with the values shown for the years shown. Note that year can be either 2-digits or 4-digits (they are all converted to 4-digits in the program).

mul

multiplies the equation's forecast by a factor specified by the data series on the following line. For example,

```
mul ulfi$
  1992  1.0
  1995  1.05
  2000  1.10;
```

multiplies the forecast results for the macrovariable ulfi$ by the factors shown. Values of the multiplicative fix between the years shown are linearly interpolated.

cta

does a constant term adjustment. That is, it adds or subtracts the value of the time series to the result of the equation. The time series is provided by the fix definition. For example,

```
cta nonagincome
   1992 .0001
   1995  200
   2000  180;
```

is a constant term adjustment for nonagricultural income from 1992 to 2000. Intermediate values are of course linearly interpolated.

ind, dind

is a variety of the override fix that specifies the time series as an index. There must be data in the vam file for the item being fixed up until at least the first year of the index series specified. The value for the item in that year is then moved by the index of the time series given by the fix lines. For example,

```
ind wag01
  1982  1.0  1.03  1.08  1.12  1.15
  1997  1.21  1.29  1.31  1.34;
```

will move the value of wag01 in 1982 forward by the rate of change of the series given, and will replace the calculated value of wag01 by this value when the model is run.

The “dind” version of the index fix is the “dynamic index fix”. This fix can start in any year, and does not rely on historical data being present in the databank. Rather, the fix is calculated based on the value of expression during the model solution for the first year of the fix.

gro, dgro

is a type of override fix that specifies the time series by growth rates. For the growth rate fix to be legal, there must be data in the vam file up until at least the year before the first year of the growth rate fix. Missing values of the growth rates are linearly interpolated.

```
gro wag01
  1993  3.1
  2000  3.4;
```

The “dgro” version is the “dynamic growth rate fix”. This fix can start in any year, and does not require data to be available in the databank for the starting year of the fix. The growth rate is always applied to the value of the variable in the previous period.

stp, dstp

is a step-growth fix. It is like “gro” except that a growth rate continues until a new one is provided. A value for the final period is necessary.

```
stp wag01
  93  4.1
  95  4.5
  2000 5.0;
```

The “dstp” version is analogous to the “dgro” fix, only the values of the fix are interpolated differently.

rho

is a rho-adjustment fix. This type of fix finds the error made by an equation in the last year for which there is data; in the next year, it multiplies this error by the given rho and adds to the value forecast by the equation; the next year it multiplies what it added in the first year by rho again and adds the result to the equation's forecast, and so on. For rho-adjustment fixes, the format is:

rho <depvar> <rho_value> <rho_set_date>

where

rho_value is the value of rho.

rho_set_date is the year in which the rho-adjustment error is to be calculated. If none is provided, it is set in the first year of the run.

for example:

```
rho invn$38 .40 1995
```

tells the model to apply a rho-adjustment to the variable invn$38 using the value .40 for rho, and starting the rho-adjustment in 1995.

A rho fix with a rho_set_date works like a "skip" in years before the rho_set_date. A variable can have a "rho" fix in conjunction with and a "cta","mul","ind" or other type fix. The rho adjustment is applied before the other fix.

eqn

is an equation fix. This type of fix lets you dynamically introduce a new equation relationship into the model at run time. The advantage of this type of fix is that users of the model who are not programmers can introduce their own assumed relationships into the model, without having to change the model program code. It is also helpful for prototyping a model, where you want to quickly try out different equation relationships to see how they work, before coding them into the model.

The equation fixes use the same expression syntax as used in the “f” command and other commands in G. The format for equation fixes for macrovariables is:

eqn <Macroname> = <expression>
<year> <value> [<value> <value> ...]
<year> <value> [<value> <value> ...]

where: <Macroname> is a legitimate name of a macrovariable, <expression> is a legitimate expression, as described below, and the <year> <value> entries are in the same format as the data for other fixes, but indicate the years for which the equation fix is to take effect. They also represent the time series for a special variable called “fixval”, which can be used within the equation expression. This “fixval” variable can be used wherever a vector or macrovariable could be used.

Just about any expression that is legal in G is legal for an equation fix, except that only a subset of functions are implemented. These functions are: @cum, @peak, @log, @exp, @sq, @sqrt, @pow, @fabs, @sin, @pct, @pos, @ifpos, @pct, @rand and @round.

Lagged values of any order can be used, with the constraint that they must not be before the starting year of the model G bank (DYME.BNK). Macrovariables are read directly from memory. Lagged values of vector variables are read from the Vam file. Therefore, you can use a lagged value of any vector as far back as the starting date of the Vam file, and you are not limited by whether or not that vector has been declared to store lagged values in memory in VAM.CFG.

Examples:

```
# Make the T bill rate equal to the average inflation plus some percent,
#   specified in “fixval”.
eqn rtb = .34*gnpinf + .33*gnpinf[1] + .33*gnpinf[2] + fixval
```

1998 1.0
2010 1.5;

fol

is a follow fix. The follow fix allows you to specify that a macrovariable should move like some other quantity, which may be specified as a general expression involving vector and macrovariables, just like the equation fix.

The general format for the follow fix is:

fol <Macroname> = <expression>
<year> <value> [<value> <value> ...]
<year> <value> [<value> <value> ...]

The variable “fixval” should not be used in the follow fix expression. Its purpose is to specify a growth rate to add to the growth of the expression.

For example, if we would like to specify that Medicaid transfer payments grow like real disposable income per capita, plus 0.1 per cent, we could write:

fol trhpmi = di87/pt
1997 0.1
2010 0.1

shr

is a share fix. This fixed is used to specify that the macrovariable should be a certain share of another variable or expression, with the share specified by the fix value. Actually, the “share” is just a multiplier, so it can be any number.

The general format of the share fix is:

shr <Macroname> = <expression>
<year> <value> [<value> <value> ...]
<year> <value> [<value> <value> ...]

In the share fix, the fix value is the multiplier or share to multiply by the right hand side expression.

When the input file as described above is ready and the macfixer.cfg file calls for its use, type "macfixer" at the DOS prompt to invoke the program Macfixer. When the model is running, calls to the "modify" function will apply the fixes, using the information in the macro fix G bank specified in the dyme.cfg for that run. Note that to view the fixes in the macro fix databank, specify the series name as the name of the macro variable, followed by a colon (':'), followed by a one-letter code signifying the type of fix.

These codes are as follows: skip ('k'), ovr ('o'), cta ('c'), ind ('i'), gro ('g'), stp ('s'), and rho ('r'). Therefore, to view a “cta” fix on the variable invn$38, do the following command in g:

```
ty invn$38:c
```

Macro fixes provide an alternative way to supply values of exogenous variables. Exogenous variables, to review, should be put into the "hist" bank in the process of running Idbuild. If the variable appears in no .sav file for a macro equation, then it is included in the PSEUDO.SAV file. The standard way of providing the values of the exogenous variables is then through "update" or other commands in Vam. Another possibility for providing exogenous values is to have a special run of G with the "hist" or other bank as the workspace bank. Finally, one can provide the exogenous values as macrofixes. For example, if we want disinc to be an exogenous variable, then -- however we are going to provide the values -- we need the statement

```
f disinc = disinc
```

in the "pseudo.sav" file. To use the macrofix method of assigning values, we need in the code of the model the statements

```
depend=disinc[t];
disinc[t] = disinc.modify(depend);
```

We could then provide the values with "ovr", "ind", "gro", or "stp" commands to the macrofix program, for example, by

```
gro disinc
   1995 3.0
   2000 3.5
   2005 4.0;
```

This method has the advantage of keeping all the fixes which constitute a scenario in one place. It also allows the use of the "gro" and "stp" fixes, which may be convenient. It has the disadvantage of adding an additional series to the banks which constitute the model and an additional statement within the model.

Please visit the Software pages of the Inforum web site for more details and to download the Fixer software: www.inforum.umd.edu/software/software.html.