

September 10, 2010

# MCONTROL

## Overview

The **mcontrol** addin is an iterative procedure that solves a model for the values of a group of exogenous (*control*) variables that minimize a quadratic function of the deviations of a group of endogenous (*target*) variables from desired paths (*trajectories*). The number of control variables must be less than or equal to the number of target variables. **mcontrol** is a generalization of the EViews **control** procedure that solves a model when there is a single control, target, and trajectory.

Let  $n$  be the number of control variables,  $m$  the number of target variables, and  $T$  the number of periods in the current workfile sample. Denote  $\mathbf{x}$  as the  $(nT \times 1)$  stacked vector of control variables,  $\mathbf{y}$  as the  $(mT \times 1)$  stacked vector of targets, and  $\mathbf{y}^*$  as the  $(mT \times 1)$  stacked vector of their desired paths. Each vector is ordered by variable (ie,  $\mathbf{x} = [x_{1,1}, \dots, x_{1,T}, x_{2,1}, \dots, x_{n,T}]$ ). The procedure finds the value of  $\mathbf{x}$  than minimizes the quadratic function,

$$f = (\mathbf{y} - \mathbf{y}^*)' \mathbf{W} (\mathbf{y} - \mathbf{y}^*) \quad (1)$$

where  $\mathbf{W}$  is a  $(mT \times mT)$  symmetric matrix of weights.

The procedure requires estimates of the first (gradient) and second (hessian) derivatives of  $f$  with respect to the control variables. The gradient is calculated from a set of  $nT$  perturbation simulations, each of which involves a ping to one of the control variables in one of the simulation periods. The hessian is estimated under the assumption that the model is linear. For a linear model, the procedure will find the solution in one iteration. When the model is nonlinear, more than one iteration will be required, and convergence to the global minimum may not be achieved.

A special case occurs when the number of control variables equals the number of target variables and the model's relationship between the controls and targets has full rank at the minimum of  $f$ . Under these conditions, the minimum is zero for any weight matrix of full rank, including the identity matrix.

## Syntax

model\_name.mcontrol(*options*) *conts* *targs* *trajs* [*weights* *discount*]

Three arguments are required: *conts*, a group object containing the exogenous control variables; *targs*, a group object containing the endogenous target variables; and *trajs*, a group object containing the desired trajectories of the target variables. The *ith* series in *trajs* is the desired trajectory for the *ith* series in *targs*. The *weights* argument is optional; if it is omitted,  $\mathbf{W}$  is assumed to be the identity matrix. When included, the *weights* argument may be either a group or (symmetric) matrix object. If *weights* is a group,  $\mathbf{W}$  is constructed as a diagonal matrix from the series in the group. In this case, the *ith* series in *weights* contains the weights associated with the *ith* target variable. When *weights* is a sym or matrix object, it may have size  $mT$  or  $m$ . In the former instance,  $\mathbf{W}$  equals the matrix object itself. Alternatively, when *weights* has size  $m$ , it is interpreted as containing the loss function weights for the first simulation period only. The weights in subsequent simulation periods are either set equal the first-period weights or decline period-by-period at the rate given by the optional *discount* scalar. In this case,  $\mathbf{W}$  would have a block diagonal structure if the control and target vectors were arranged by date rather than variable.

## Options

	description	default
c = number	Convergence criteria	1e-06
m = integer	Maximum number of iterations	15
l = integer	Maximum number of line search steps	9
t = number	Nonlinearity threshold	0.10
p = number	Initial control variable perturbation factor for gradient simulations	0.01
o = number	0 for no output; 1 for output at end of execution; 2 for output each iteration	2
na	Permit NAs in desired values	
g	Execute gradient (linearization) simulations every iteration	
debug	Do not delete temporary matrices	

Except when the minimum of the loss function is zero, convergence occurs when the percentage change in the loss function from one iteration to the

next is less than the convergence criteria **c**. The exception occurs when there are an equal number of controls and targets, in which case convergence requires that the loss value itself be less than the criteria. Gradient simulations are run every iteration if the **g** option is included. Otherwise, gradient simulations are computed in the first iteration, and thereafter they are run only when a nonlinearity statistic exceeds a threshold that can be set using option **t**. The nonlinearity statistic and its threshold also regulate whether step-size optimization takes place at an iteration. The default setting of **t** is .10, a value that seems to do well for models with modest nonlinearity. For models that are very nonlinear, a value of **t** that is closer to zero may be preferred.

When the **na** option is included, the algorithm interprets each NA value in a desired trajectory as indicating that the corresponding target variable has no desired value at that time period. The algorithm deletes the NA observations by shortening the length of **y\*** and **y** from  $mT$  to  $mT - k$ , where  $k$  is the number of NAs. The treatment of control variables when NAs are present in the desired trajectories depends on whether the number of control variables is the same as ( $n = m$ ) or fewer than ( $n < m$ ) the number of target variables. In the former case, the length of the control vector, **x**, is also shortened by  $k$  through the removal of the corresponding control variable observations. In the latter case, the control vector is not adjusted, but execution will proceed only if  $nT \leq mT - k$ . If the **na** option is not included, an NA causes execution to terminate.

The **o** and **debug** options control the “showing” of three table and text files: **mcontrol\_stats** contains key iteration statistics; **mcontrol\_text** presents information about the procedure’s inputs and design; and **mcontrol\_work** has a list of temporary matrices and vectors. When the output option **o** is set at its default of 2, **mcontrol\_stats** is updated and “shown” in the user’s EViews window each iteration, thus allowing the progress of the procedure to be monitored. At the end of a successful execution, a spool file named **mcontrol\_spool** containing **mcontrol\_stats** and **mcontrol\_text** is created and also “shown”. When **o** = 1, the final spool is displayed but the real-time iteration information is not. No output is displayed when **o** = 0. When the **debug** option is included, **mcontrol\_work** is added to **mcontrol\_spool**, the spool is displayed irrespective of the value of option **o**, and the temporary objects are not deleted from the workfile.

## Examples

In the first example, the set of exogenous variables in model *testmod* contains

the control variables  $x_1$ ,  $x_2$ , and  $x_3$ , the set of endogenous variables in the model includes the target variables  $y_1$ ,  $y_2$ , and  $y_3$ , and the desired trajectories of the target variables are contained in the series  $t_1$ ,  $t_2$ , and  $t_3$ . The weight matrix defaults to the identity matrix; the procedure will thus try to find the values of the control variables that minimize the unweighted sum of squares of gaps between the targets and their desired trajectories over the 2001q1-2008q4 sample. If the relationship between the control variables and target variables in the model has full rank, the minimized value will be zero.

```
group conts x1 x2 x3
group targs y1 y2 y3
group trajs t1 t2 t3
smpl 2000q1 2008q4
testmod.mcontrol conts targs trajs
```

In the second example, there is one control variable, three target variables, and the elements of a diagonal weight matrix are contained in the series  $w_1$ ,  $w_2$ , and  $w_3$ . The iteration limit is set to 10.

```
group conts x1
group targs y1 y2 y3
group trajs t1 t2 t3
group weights w1 w2 w3
smpl 2000q1 2008q4
testmod.mcontrol(m=10) conts targs trajs weights
```

The third example uses a  $(m \times m)$  symmetric matrix to specify the loss function weights for the first simulation period. The weights for each subsequent period are 0.9 times the weights in previous period. Weights on all intertemporal loss terms are zero.

```
group conts x1
group targs y1 y2 y3
group trajs t1 t2 t3
sym(3) weights
weights.fill 1,.5,.5,1,.5,1
scalar disc = .9
smpl 2000q1 2008q4
testmod.mcontrol conts targs trajs weights disc
```

## Method

For a linear model, the relationship between the control and target variables can be exactly expressed in terms of a coefficient matrix  $\mathbf{D}$  and baseline values  $\mathbf{x}_0$  and  $\mathbf{y}_0$ ,

$$(\mathbf{y} - \mathbf{y}_0) = \mathbf{D}(\mathbf{x} - \mathbf{x}_0) \quad (2)$$

As is well known in this case, the values of the controls that minimize the quadratic loss function are given by,

$$\hat{\mathbf{x}} = \mathbf{x}_0 - \mathbf{H}^{-1}\mathbf{g} \quad (3)$$

where  $\mathbf{g}$  is the vector of first derivatives (gradient) of the loss function with respect to the controls and  $\mathbf{H}$  is the matrix of second derivatives (hessian),

$$\mathbf{g} = 2\mathbf{D}'\mathbf{W}(\mathbf{y}_0 - \mathbf{y}^*) \quad (4)$$

$$\mathbf{H} = 2\mathbf{D}'\mathbf{W}\mathbf{D} \quad (5)$$

When the number of control variables is the same as the number of target variables and  $\mathbf{D}$  is thus square, equation 3 simplifies to an expression in which the weight matrix no longer appears,

$$\hat{\mathbf{x}} = \mathbf{x}_0 - \mathbf{D}^{-1}(\mathbf{y}_0 - \mathbf{y}^*) \quad (6)$$

The **mcontrol** algorithm is designed to work with both linear and non-linear models. The linear case requires a single solution iteration; nonlinear models require a sequence of iterations. At iteration  $i$ , the formula for updating the control variable takes the form of either

$$\mathbf{x}_i = \mathbf{x}_{i-1} - \alpha_i \mathbf{H}_i^{-1} \mathbf{g}_i \quad (7)$$

or

$$\mathbf{x}_i = \mathbf{x}_{i-1} - \alpha_i \mathbf{D}_i^{-1}(\mathbf{y}_{i-1} - \mathbf{y}^*) \quad (8)$$

depending on the relative number of control and target variables. Each formula includes the step size parameter  $\alpha$ .

The first and most costly part of each iteration executes  $nT$  control perturbation simulations to calculate the coefficient matrix  $\mathbf{D}_i$  of the linear approximation to the relationship between the controls and targets,

$$(\mathbf{y}_i - \mathbf{y}_{i-1}) \approx \mathbf{D}_i(\mathbf{x}_i - \mathbf{x}_{i-1}) \quad (9)$$

In the first iteration, the size of the perturbations defaults to 0.01 unless a different value is given by the optional parameter  $\mathbf{p}$ . In subsequent iterations, the perturbation applied to each control equals the sum of 1e-06 and the change in the control in the prior iteration.

Some nonlinear models may be close enough to being linear that  $\mathbf{D}$  need not be calculated every iteration. Unless the  $\mathbf{g}$  option is included, the algorithm automatically runs perturbation simulations only in the first iteration. On subsequent iterations, the decision whether to rerun perturbation simulations is based on the value of a statistic measuring the model's degree of nonlinearity in the neighborhood of the solutions in the current and previous iterations. Let  $f_{i,\alpha}$  be the value of the loss function at iteration  $i$  given step size  $\alpha$ ,  $f_{i-1,\hat{\alpha}}$  the value of the loss function at iteration  $i-1$  at the step size that was chosen during that iteration, and  $f_{i/i-1,1}$  the value of the loss function predicted for iteration  $i$  at the end of iteration  $i-1$  assuming the model is linear. The nonlinearity statistic,  $z$ , is defined as the difference from unity of the ratio of the actual to predicted decrease in the value of the loss function at a step size of 1.0,

$$z_i = 1 - (f_{i-1,\hat{\alpha}} - f_{i,1}) / (f_{i-1,\hat{\alpha}} - f_{i/i-1,1}) \quad (10)$$

If the model is linear,  $z_i = 0$ . Perturbation simulations are executed to update the estimate of  $\mathbf{D}$  only if the absolute value of  $z_i$  exceeds the value of option  $\mathbf{t}$  or its default of 0.10. At the default value, simulations are run only when the actual decrease in the loss differs from the predicted decrease by more than 10 percent.

When there are fewer control variables than target variables, the second part of each iteration computes the gradient and hessian,

$$\mathbf{g}_i = 2\mathbf{D}_i'\mathbf{W}(\mathbf{y}_{i-1} - \mathbf{y}^*) \quad (11)$$

$$\mathbf{H}_i = 2\mathbf{D}_i'\mathbf{W}\mathbf{D}_i \quad (12)$$

If there are as many control variables as target variables, the only calculation needed in this part is the inverse of  $\mathbf{D}$ .

The third part of each iteration, step-size optimization, takes place only if, at a step size of 1.0, the value of the loss function is higher than in the previous iteration or the absolute value of the nonlinearity statistic  $\mathbf{z}$  is larger than its threshold. In the former case, the step search starts from a step size close to zero and increases the step size until a local minimum is found or the maximum number of line search steps is hit. In the latter case, the line search starts from a step size of 1.0, evaluates whether to increase or decrease the step size, and then proceeds with ever larger or smaller steps until a local minimum is found or the step maximum is hit.