

November 3, 2010

MCONTROL_QP

Overview

The **mcontrol_qp** addin is an iterative procedure that calls an **R** quadratic programming function to solve 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 procedure permits inequality constraints to be imposed on linear functions of the model's endogenous variables. The number of control variables must be less than or equal to the number of target variables. For applications that do not involve constraints, **mcontrol_qp** tends to execute more slowly than the **mcontrol** addin. Both addins are generalizations of the EViews **control** procedure that solves a model when there is a single control, target, and trajectory. Use of **mcontrol_qp** requires access to R via the statconnDCOM communications module.

Let n be the number of control variables, m_o the number of target (objective) variables, c the number of constraints, m_c the number of variables appearing in the constraints, 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}_o as the $(m_oT \times 1)$ stacked vector of targets, \mathbf{y}_o^* as the $(m_oT \times 1)$ stacked vector of their desired paths, and \mathbf{y}_c^* $(m_cT \times 1)$ as the stacked vector of constraint variables. 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,

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

subject to the constraints

$$\mathbf{A} \mathbf{y}_c \geq \mathbf{b} \quad (2)$$

\mathbf{W} is a $(m_oT \times m_oT)$ symmetric matrix of weights, \mathbf{A} is a $(cT \times m_cT)$ matrix of constraint coefficients and \mathbf{b} is a $(cT \times 1)$ matrix of constraint constants.

Define y as the union of the y_o and y_c and m as the number of variables in y . The procedure requires estimates of the first derivatives of y with respect to the control variables. These are 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. 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.

Syntax

`model_name.mcontrol_qp(options) conts targs trajs [cnstr weights disc]`

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*.

Three arguments are optional: When included, the *cnstr* argument must be a text object in which the *ith* line has the form:

$$a_{i,1} * y_1 + \dots + a_{i,k} * y_k \geq b_i \quad (3)$$

The variables y_1, \dots, y_k must be the names of endogenous variables. The coefficients $a_{i,1}, \dots, a_{i,k}$ and constant b_i must be specific numerical values. Coefficients must be placed before variables and be followed by the “*” character. Coefficients that equal one (or minus one) may be omitted.

If the optional *weights* argument 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
<code>c = number</code>	Convergence criteria	1e-06
<code>m = integer</code>	Maximum number of iterations	15
<code>p = number</code>	Initial control variable perturbation factor for derivative simulations	0.01
<code>o = number</code>	0 for no output; 1 for output at end of execution; 2 for output each iteration	2
<code>debug</code>	Do not delete temporary matrices	

Convergence occurs when the solutions for the target variables obtained in R (based on a linearization of the model) differ from those obtained when the original model is solved in EViews by no more than the convergence criteria `c`.

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, subject to the constraint that the endogenous variable y_4 is greater than or equal to zero.

```
group conts x1 x2 x3
group targs y1 y2 y3
group trajs t1 t2 t3
text cc
cc.append y4 >= 0
smpl 2000q1 2008q4
testmod.mcontrol_qp conts targs trajs cc
```

In the second example, there is one control variable, three target variables, the elements of a diagonal weight matrix are contained in the series w_1 , w_2 , and w_3 , the iteration limit is 10, and target variable y_1 is constrained to be greater than or equal to endogenous variable y_4 .

```
group conts x1
group targs y1 y2 y3
group trajs t1 t2 t3
group weights w1 w2 w3
text cc
cc.append y1 - y4 >= 0
smpl 2000q1 2008q4
testmod.mcontrol_qp(m=10) conts targs trajs cc weights
```

Method

At each iteration (i), **mcontrol_qp** calls the R function *quadprog::solve.QP*(H, h, J, j) to minimize the quadratic expression $(.5\mathbf{x}'\mathbf{H}\mathbf{x} - \mathbf{h}'\mathbf{x})$ subject to the constraints $\mathbf{J}'\mathbf{x} \geq \mathbf{j}$. The inputs to the R function are created in two steps. First, perturbations simulations are run to compute the coefficient matrix \mathbf{D}_i of the linear approximation to the relationship between \mathbf{x} and \mathbf{y} .

$$(\mathbf{y}_i - \mathbf{y}_{i-1}) \approx \mathbf{D}_i(\mathbf{x}_i - \mathbf{x}_{i-1}) \quad (4)$$

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 absolute change in the control in the prior iteration. The four input matrices are then created:

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

$$\mathbf{h}_i = -2\mathbf{D}_i'\tilde{\mathbf{W}}(\mathbf{y}_{i-1} - \mathbf{D}_i\mathbf{x}_{i-1} - \mathbf{y}^*) \quad (6)$$

$$\mathbf{J}_i = \tilde{\mathbf{A}}\mathbf{D}_i \quad (7)$$

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

When the constraints contain endogenous variables that are not in the set of target variables, the vector of target variables is expanded to include these extra variables, the matrices \mathbf{W} , \mathbf{A} , and \mathbf{b} are enlarged with the appropriate number of zeros (and named $\tilde{\mathbf{W}}$, $\tilde{\mathbf{A}}$, and $\tilde{\mathbf{b}}$, respectively), and \mathbf{y}_o^* is lengthened with zeros to form \mathbf{y}^* . (In the latter case, the particular numerical values are not important, given that the associated weights are zero.)