constraint — Define and list constraints

Description

`constraint` defines, lists, and drops linear constraints. Constraints are for use by models that allow constrained estimation.

Constraints are defined by the `constraint` command. The currently defined constraints can be listed by either `constraint list` or `constraint dir`; both do the same thing. Existing constraints can be eliminated by `constraint drop`.

`constraint get` and `constraint free` are programmer's commands. `constraint get` returns the contents of the specified constraint in macro `r(contents)` and returns in scalar `r(defined)` 0 or 1—1 being returned if the constraint was defined. `constraint free` returns the number of a free (unused) constraint in macro `r(free)`.

Quick start

For single-equation models

Define constraint 1 to constrain the coefficient for `x1` to 0

```
constraint define 1 x1=0
```

Same as above

```
constraint 1 x1
```

Constrain coefficients for `x1` and `x2` to 0

```
constraint 2 x1 x2
```

Overwrite constraint 2 to constrain `x2` and `x3` to equality

```
constraint 2 x2 = x3
```

Constrain the coefficients for factor indicators `2.a` and `3.a` to equality

```
constraint 3 2.a = 3.a
```

Constrain the coefficient on `x1` to equal 1

```
constraint 4 x1 = 1
```

For multiple-equation models

Constrain coefficient for `x4` to 0 in all equations

```
constraint 11 x4
```

Constrain coefficients for `x4` and `x5` to equality in the equation for `y2`

```
constraint 12 [y2]x4 = [y2]x5
```

Constrain the coefficient for x5 to equality in equations for y1 and y2

```
constraint 13 [y1=y2] x5
```

Constrain coefficient for x1 to 0 in equation where the dependent variable equals cat2

```
constraint 14 [cat2] x1
```

Constrain the coefficients for factor indicators 1.a and 1.b to equality in the equation for category cat3

```
constraint 15 [cat3]: 1.a = 1.b
```

Constrain coefficients for x1 to equality in the equations for categories cat2 and cat3

```
constraint 16 [cat2=cat3]: x1
```

Listing constraints

List existing constraints

```
constraint dir
```

Same as above

```
constraint list
```

Menu

Statistics > Other > Manage constraints

Syntax

Define constraints

```
constraint [define] # [exp=exp | coeflist]
```

List constraints

```
constraint dir [numlist | _all]
```

```
constraint list [numlist | _all]
```

Drop constraints

```
constraint drop { numlist | _all }
```

Programmer's commands

```
constraint get #
```

```
constraint free
```

where *coeflist* is as defined in [R] **test** and # is restricted to the range 1 to 1,999, inclusive.

Remarks and examples

Using constraints is discussed in [R] **cnsreg**, [R] **mlogit**, and [R] **reg3**; this entry is concerned only with practical aspects of defining and manipulating constraints.

▷ Example 1

Constraints are numbered from 1 to 1,999, and we assign the number when we define the constraint:

```
. use https://www.stata-press.com/data/r19/sysdsn1
(Health insurance data)
. constraint 2 [indemnity]2.site = 0
```

The currently defined constraints can be listed by `constraint list`:

```
. constraint list
     2:  [indemnity]2.site = 0
```

`constraint drop` drops constraints:

```
. constraint drop 2
. constraint list
```

The empty list after `constraint list` indicates that no constraints are defined. Below, we demonstrate the various syntaxes allowed by `constraint`:

```
. constraint 1 [Indemnity]
. constraint 10 [Indemnity]: 1.site 2.site
. constraint 11 [Indemnity]: 3.site
. constraint 21 [Prepaid=Uninsure]: nonwhite
. constraint 30 [Prepaid]
. constraint 31 [Insure]
. constraint list
     1:  [Indemnity]
    10:  [Indemnity]: 1.site 2.site
    11:  [Indemnity]: 3.site
    21:  [Prepaid=Uninsure]: nonwhite
    30:  [Prepaid]
    31:  [Insure]
. constraint drop 21-25, 31
. constraint list
     1:  [Indemnity]
    10:  [Indemnity]: 1.site 2.site
    11:  [Indemnity]: 3.site
    30:  [Prepaid]
. constraint drop _all
. constraint list
```

◁

❑ Technical note

The `constraint` command does not check the syntax of the constraint itself because a constraint can be interpreted only in the context of a model. Thus, `constraint` is willing to define constraints that later will not make sense. Any errors in the constraints will be detected and mentioned at the time of estimation.

❑

Reference

Buis, M. L. 2012. Stata tip 108: On adding and constraining. *Stata Journal* 12: 342–344.

Also see

[R] **cnsreg** — Constrained linear regression

[P] **makecns** — Constrained estimation