st_vlexists() — Use and manipulate value labels

Description   Syntax   Remarks and examples   Conformability   Diagnostics   Also see

Description

`st_vlexists(`*name*`)` returns 1 if value label *name* exists and returns 0 otherwise.

`st_vldir()` returns a string vector of names of all value labels.

`st_vldrop(`*name*`)` drops value label *name* if it exists.

`st_vlmap(`*name*`,` *values*`)` maps *values* through value label *name* and returns the result.

`st_vlsearch(`*name*`,` *text*`)` does the reverse; it returns the value corresponding to the text.

`st_vlload(`*name*`,` *values*`,` *text*`)` places value label *name* into *values* and *text*.

`st_vlmodify(`*name*`,` *values*`,` *text*`)` creates a new value label or modifies an existing one.

Syntax

real scalar	`st_vlexists(`*name*`)`
string vector	`st_vldir()`
void	`st_vldrop(`*name*`)`
string matrix	`st_vlmap(`*name*`,` *real matrix values*`)`
real matrix	`st_vlsearch(`*name*`,` *string matrix text*`)`
void	`st_vlload(`*name*`,` *values*`,` *text*`)`
void	`st_vlmodify(`*name*`,` *real colvector values*`,` *string colvector text*`)`

where *name* is *string scalar* and where the types of *values* and *text* in `st_vlload()` are irrelevant because they are replaced.

Remarks and examples

Value labels record a mapping from numeric values to text, and they always have a name. For instance, a value label named `sexlbl` might record that 1 corresponds to male and 2 to female. Value labels are attached to Stata numeric variables. If a Stata numeric variable had the value label `sexlbl` attached to it, then the 1s and 2s in the variable would be displayed as male and female, respectively. How other values would appear—if there were other values—would not be affected.

Remarks are presented under the following headings:

Value-label mapping

Let us consider value label `sexlbl` mapping 1 to male and 2 to female.
`st_vlmap("sexlbl",` *values*`)` would map the $r \times c$ matrix values through `sexlbl` and return an $r \times c$ string matrix containing the result. Any values for which there was no mapping would result in `""`. Thus

```
: res = st_vlmap("sexlbl", 1)

: res
male

: res = st_vlmap("sexlbl", (2,3,1))

: res
            1         2         3
    +-------------------------------+
  1 |  female                  male |
    +-------------------------------+
```

`st_vlsearch(`*name*`,` *text*`)` performs the reverse mapping:

```
: txt = st_vlsearch("sexlbl", ("female","","male"))

: txt
       1   2   3
    +-------------+
  1 |  2   .   1  |
    +-------------+
```

Value-label creation and editing

`st_vlmodify(`*name*`,` *values*`,` *text*`)` creates new value labels and modifies existing ones.

If value label `sexlbl` did not exist, coding

```
: st_vlmodify("sexlbl", (1\2), ("male"\"female"))
```

would create it. If the value label did previously exist, the above would modify it so that 1 now corresponds to male and 2 to female, regardless of what 1 or 2 previously corresponded to, if they corresponded to anything. Other mappings that might have been included in the value label remain unchanged. Thus

```
: st_vlmodify("sexlbl", 3, "unknown")
```

would add another mapping to the label. Value labels are deleted by specifying the text as `""`, so

```
: st_vlmodify("sexlbl", 3, "")
```

would remove the mapping for 3 (if there was a mapping). If you remove all the mappings, the value label itself is automatically dropped:

```
: st_vlmodify("sexlbl", (1\2), (""\""))
```

results in value label `sexlbl` being dropped if 1 and 2 were the final mappings in it.

Loading value labels

`st_vlload(`*name*`,` *values*`,` *text*`)` returns the value label in *values* and *text*, where you can do with it as you please. Thus you could code

```
st_vlload("sexlbl", values, text)
     ...
st_vldrop("sexlbl")
st_vlmodify("sexlbl", values, text)
```

Conformability

`st_vlexists(`*name*`)`:

name:	1×1
result:	1×1

`st_vldir()`:

result:	$1 \times \mathrm{n}$

`st_vldrop(`*name*`)`:

name:	1×1
result:	*void*

`st_vlmap(`*name*`,` *values*`)`:

name:	1×1
values:	$r \times c$
result:	$r \times c$

`st_vlsearch(`*name*`,` *text*`)`:

name:	1×1
text:	$r \times c$
result:	$r \times c$

`st_vlload(`*name*`,` *values*`,` *text*`)`:

input:

name:	1×1

output:

values:	$k \times 1$
text:	$k \times 1$

`st_vlmodify(`*name*`,` *values*`,` *text*`)`:

name:	1×1
values:	$m \times 1$
text:	$m \times 1$
result:	*void*

Diagnostics

The only conditions under which the above functions abort with error is when *name* is malformed or Mata is out of memory. Functions tolerate all other problems.

`st_vldir()` calls Stata command `label dir`; see [D] **label**.

`st_vldrop(`*name*`)` does nothing if value label *name* does not exist.

`st_vlmap(`*name*`,` *values*`)` returns `J(rows(values), cols(values), "")` if value label *name* does not exist. When the value label does exist, individual values for which there is no recorded mapping are returned as `""`.

`st_vlsearch(`*name*`,` *text*`)`: returns `J(rows(values), cols(values), .)` if value label *name* does not exist. When the value label does exist, individual text values for which there is no corresponding value are returned as `.` (missing).

`st_vlload(`*name*`,` *values*`,` *text*`)`: sets *values* and *text* to be 0×1 when value label *name* does not exist.

`st_vlmodify(`*name*`,` *values*`,` *text*`)`: creates the value label if it does not already exist. Value labels may map only integers and `.a`, `.b`, ..., `.z`. Attempts to insert a mapping for `.` are ignored. Noninteger values are truncated to integer values. If an element of *text* is `""`, then the corresponding mapping is removed.

Also see

[M-4] **Stata** — Stata interface functions