Title

stata.com

irf table — Tables of IRFs, dynamic-multiplier functions, and FEVDs

Syntax

```
irf table [stat] [, options]
```

stat	Description
Main	
`irf`	impulse–response function
`oirf`	orthogonalized impulse–response function
`dm`	dynamic-multiplier function
`cirf`	cumulative impulse–response function
`coirf`	cumulative orthogonalized impulse–response function
`cdm`	cumulative dynamic-multiplier function
`fevd`	Cholesky forecast-error variance decomposition
`sirf`	structural impulse–response function
`sfevd`	structural forecast-error variance decomposition

If *stat* is not specified, all statistics are included, unless option `nostructural` is also specified, in which case `sirf` and `sfevd` are excluded. You may specify more than one *stat*.

options	Description
Main	
`set(`*filename*`)`	make *filename* active
`irf(`*irfnames*`)`	use *irfnames* IRF result sets
`impulse(`*impulsevar*`)`	use *impulsevar* as impulse variables
`response(`*endogvars*`)`	use endogenous variables as response variables
`individual`	make an individual table for each result set
`title("`*text*`")`	use *text* for overall table title
Options	
`level(#)`	set confidence level; default is `level(95)`
`noci`	suppress confidence intervals
`stderror`	include standard errors in the tables
`nostructural`	suppress `sirf` and `sfevd` from the default list of statistics
`step(#)`	use common maximum step horizon # for all tables

Menu

Statistics > Multivariate time series > IRF and FEVD analysis > Tables by impulse or response

Description

`irf table` makes a table from the specified IRF results.

The rows of the tables are the time since impulse. Each column represents a combination of `impulse()` variable and `response()` variable for a *stat* from the `irf()` results.

Options

Main

`set(`*filename*`)` specifies the file to be made active; see [TS] **irf set**. If `set()` is not specified, the active file is used.

All results are obtained from one IRF file. If you have results in different files that you want in one table, use `irf add` to copy results into one file; see [TS] **irf add**.

`irf(`*irfnames*`)` specifies the IRF result sets to be used. If `irf()` is not specified, all the results in the active IRF file are used. (Files often contain just one set of IRF results, saved under one *irfname*; in that case, those results are used. When there are multiple IRF results, you may also wish to specify the `individual` option.)

`impulse(`*impulsevar*`)` specifies the impulse variables for which the statistics are to be reported. If `impulse()` is not specified, each model variable, in turn, is used. *impulsevar* should be specified as an endogenous variable for all statistics except `dm` or `cdm`; for those, specify as an exogenous variable.

`response(`*endogvars*`)` specifies the response variables for which the statistics are to be reported. If `response()` is not specified, each endogenous variable, in turn, is used.

`individual` specifies that each set of IRF results be placed in its own table, with its own title and footer. By default, `irf table` places all the IRF results in one table with one title and one footer. `individual` may not be combined with `title()`.

`title("`*text*`")` specifies a title for the overall table.

Options

`level(#)` specifies the default confidence level, as a percentage, for confidence intervals, when they are reported. The default is `level(95)` or as set by `set level`; see [U] **20.7 Specifying the width of confidence intervals**.

`noci` suppresses reporting of the confidence intervals for each statistic. `noci` is assumed when the model was fit by `vec` because no confidence intervals were estimated.

`stderror` specifies that standard errors for each statistic also be included in the table.

`nostructural` specifies that *stat*, when not specified, exclude `sirf` and `sfevd`.

`step(#)` specifies the maximum step horizon for all tables. If `step()` is not specified, each table is constructed using all steps available.

Remarks and examples

stata.com

If you have not read [TS] **irf**, please do so.

Also see [TS] **irf graph**, which produces output in graphical form, and see [TS] **irf ctable**, which also produces tabular output. `irf ctable` is more difficult to use but provides more control over how tables are formed.

▷ Example 1

We have fit a model with `var`, and we saved the IRFs from two different orderings. The commands we previously used were

```
. use http://www.stata-press.com/data/r13/lutkepohl2
. var dln_inv dln_inc dln_consump
. irf set results4
. irf create ordera, step(8)
. irf create orderb, order(dln_inc dln_inv dln_consump) step(8)
```

We now wish to compare the two orderings:

```
. irf table oirf fevd, impulse(dln_inc) response(dln_consump) noci std
> title("Ordera versus orderb")
```

Ordera versus orderb

step	(1) oirf	(1) S.E.	(1) fevd	(1) S.E.
0	.005123	.000878	0	0
1	.001635	.000984	.288494	.077483
2	.002948	.000993	.294288	.073722
3	-.000221	.000662	.322454	.075562
4	.000811	.000586	.319227	.074063
5	.000462	.000333	.322579	.075019
6	.000044	.000275	.323552	.075371
7	.000151	.000162	.323383	.075314
8	.000091	.000114	.323499	.075386

step	(2) oirf	(2) S.E.	(2) fevd	(2) S.E.
0	.005461	.000925	0	0
1	.001578	.000988	.327807	.08159
2	.003307	.001042	.328795	.077519
3	-.00019	.000676	.370775	.080604
4	.000846	.000617	.366896	.079019
5	.000491	.000349	.370399	.079941
6	.000069	.000292	.371487	.080323
7	.000158	.000172	.371315	.080287
8	.000096	.000122	.371438	.080366

(1) irfname = ordera, impulse = dln_inc, and response = dln_consump
(2) irfname = orderb, impulse = dln_inc, and response = dln_consump

The output is displayed as a "single" table; because the table did not fit horizontally, it wrapped automatically. At the bottom of the table is a definition of the keys that appear at the top of each column. The results in the table above indicate that the orthogonalized IRFs do not change by much.

◁

▷ Example 2

Because the estimated FEVDs do change significantly, we might want to produce two tables that contain the estimated FEVDs and their 95% confidence intervals:

```
. irf table fevd, impulse(dln_inc) response(dln_consump) individual
```

Results from ordera

step	(1) fevd	(1) Lower	(1) Upper
0	0	0	0
1	.288494	.13663	.440357
2	.294288	.149797	.43878
3	.322454	.174356	.470552
4	.319227	.174066	.464389
5	.322579	.175544	.469613
6	.323552	.175826	.471277
7	.323383	.17577	.470995
8	.323499	.175744	.471253

95% lower and upper bounds reported
(1) irfname = ordera, impulse = dln_inc, and response = dln_consump

Results from orderb

step	(1) fevd	(1) Lower	(1) Upper
0	0	0	0
1	.327807	.167893	.487721
2	.328795	.17686	.48073
3	.370775	.212794	.528757
4	.366896	.212022	.52177
5	.370399	.213718	.52708
6	.371487	.214058	.528917
7	.371315	.213956	.528674
8	.371438	.213923	.528953

95% lower and upper bounds reported
(1) irfname = orderb, impulse = dln_inc, and response = dln_consump

Because we specified the `individual` option, the output contains two tables, one for each set of IRF results. Examining the results in the tables indicates that each of the estimated functions is well within the confidence interval of the other, so we conclude that the functions are not significantly different.

◁

❑ Technical note

Be careful in how you name variables when you fit models. Say that you fit one model with `var` and used time-series operators to form one of the endogenous variables

```
. var d.ln_inv ...
```

and in another model, you created a new variable:

```
. gen dln_inv = d.ln_inv
. var dln_inv ...
```

Say that you saved IRF results from both (perhaps they differ in the number of lags). Now you wish to use `irf table` to compare them. You would not be able to specify `response(d.ln_inv)` or `response(dln_inv)` because neither variable is in both models. Similarly, you could not specify `impulse(d.ln_inv)` or `impulse(dln_inv)` for the same reason.

All is not lost; if `impulse()` is not specified, all endogenous variables are used, and similarly if `response()` is not specified, so you could obtain the result you desired by simply not specifying the options, but you will also obtain a lot more, besides. If you want to specify the `impulse()` or `response()` options, be sure to name variables consistently.

Also, you may forget how the endogenous variables were named. If so, `irf describe, detail` can provide the answer. In `irf describe`'s output, the endogenous variables are listed next to `endog`.

❑

Stored results

If the `individual` option is not specified, `irf table` stores the following in `r()`:

Scalars

`r(ncols)`	number of columns in table
`r(k_umax)`	number of distinct keys
`r(k)`	number of specific table commands

Macros

`r(key#)`	#th key
`r(tnotes)`	list of keys applied to each column

If the `individual` option is specified, then for each *irfname*, `irf table` stores the following in `r()`:

Scalars

`r(`*irfname*`_ncols)`	number of columns in table for *irfname*
`r(`*irfname*`_k_umax)`	number of distinct keys in table for *irfname*
`r(`*irfname*`_k)`	number of specific table commands used to create table for *irfname*

Macros

`r(`*irfname*`_key#)`	#th key for *irfname* table
`r(`*irfname*`_tnotes)`	list of keys applied to each column in table for *irfname*

Also see

[TS] **irf** — Create and analyze IRFs, dynamic-multiplier functions, and FEVDs

[TS] **var intro** — Introduction to vector autoregressive models

[TS] **vec intro** — Introduction to vector error-correction models