Title

stata.com

irf graph — Graphs of IRFs, dynamic-multiplier functions, and FEVDs

Description Quick start Menu Syntax
Options Remarks and examples Stored results Also see

Description

`irf graph` graphs impulse–response functions (IRFs), dynamic-multiplier functions, and forecast-error variance decompositions (FEVDs) over time.

Quick start

Graph impulse–response function for dependent variables `y1` and `y2` given an unexpected shock to `y1`

```
irf graph irf, impulse(y1) response(y2)
```

As above, but for orthogonalized shocks

```
irf graph oirf, impulse(y1) response(y2)
```

As above, but begin the plot with the third forecast period

```
irf graph oirf, impulse(y1) response(y2) lstep(3)
```

As above, but with a separate graph for each IRF in the current IRF file

```
irf graph oirf, impulse(y1) response(y2) lstep(3) individual
```

Note: `irf` commands can be used after `var`, `svar`, `vec`, `arima`, or `arfima`; see [TS] **var**, [TS] **var svar**, [TS] **vec**, [TS] **arima**, or [TS] **arfima**.

Menu

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

Syntax

```
irf graph stat [ , options ]
```

stat	Description
`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

Notes: 1. No statistic may appear more than once.
2. If confidence intervals are included (the default), only two statistics may be included.
3. If confidence intervals are suppressed (option `noci`), up to four statistics may be included.

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
`noci`	suppress confidence bands
`level(#)`	set confidence level; default is `level(95)`
`lstep(#)`	use # for first step
`ustep(#)`	use # for maximum step
Advanced	
`individual`	graph each combination individually
`iname(`*namestub* `[, replace])`	stub for naming the individual graphs
`isaving(`*filenamestub* `[, replace])`	stub for saving the individual graphs to files
Plots	
`plot#opts(`*cline_options*`)`	affect rendition of the line plotting the # *stat*
CI plots	
`ci#opts(`*area_options*`)`	affect rendition of the confidence interval for the # *stat*
Y axis, X axis, Titles, Legend, Overall	
twoway_options	any options other than `by()` documented in [G-3] ***twoway_options***
`byopts(`*by_option*`)`	how subgraphs are combined, labeled, etc.

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.

`irf(`*irfnames*`)` specifies the IRF result sets to be used. If `irf()` is not specified, each of the results in the active IRF file is used. (Files often contain just one set of IRF results saved under one *irfname*; in that case, those results are used.)

`impulse(`*impulsevar*`)` and `response(`*endogvars*`)` specify the impulse and response variables. Usually one of each is specified, and one graph is drawn. If multiple variables are specified, a separate subgraph is drawn for each impulse–response combination. If `impulse()` and `response()` are not specified, subgraphs are drawn for all combinations of impulse and response variables.

impulsevar should be specified as an endogenous variable for all statistics except `dm` or `cdm`; for those, specify as an exogenous variable.

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

`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] 20.8 Specifying the width of confidence intervals**. Also see [TS] **irf cgraph** for a graph command that allows the confidence level to vary over the graphs.

`lstep(#)` specifies the first step, or period, to be included in the graphs. `lstep(0)` is the default.

`ustep(#)`, $\# \geq 1$, specifies the maximum step, or period, to be included in the graphs.

Advanced

`individual` specifies that each graph be displayed individually. By default, `irf graph` combines the subgraphs into one image. When `individual` is specified, `byopts()` may not be specified, but the `isaving()` and `iname()` options may be specified.

`iname(`*namestub* [`, replace`]`)` specifies that the ith individual graph be stored in memory under the name *namestubi*, which must be a valid Stata name of 24 characters or fewer. `iname()` may be specified only with the `individual` option.

`isaving(`*filenamestub* [`, replace`]`)` specifies that the ith individual graph should be saved to disk in the current working directory under the name *filenamestubi*`.gph`. `isaving()` may be specified only when the `individual` option is also specified.

Plots

`plot1opts(`*cline_options*`)`, ..., `plot4opts(`*cline_options*`)` affect the rendition of the plotted statistics (the *stat*). `plot1opts()` affects the rendition of the first statistic; `plot2opts()`, the second; and so on. *cline_options* are as described in [G-3] ***cline_options***.

CI plots

`ci1opts(`*area_options*`)` and `ci2opts(`*area_options*`)` affect the rendition of the confidence intervals for the first (`ci1opts()`) and second (`ci2opts()`) statistics in *stat*. *area_options* are as described in [G-3] ***area_options***.

Y axis, X axis, Titles, Legend, Overall

twoway_options are any of the options documented in [G-3] ***twoway_options***, excluding `by()`. These include options for titling the graph (see [G-3] ***title_options***) and for saving the graph to disk (see [G-3] ***saving_option***). The `saving()` and `name()` options may not be combined with the `individual` option.

`byopts(`*by_option*`)` is as documented in [G-3] ***by_option*** and may not be specified when `individual` is specified. `byopts()` affects how the subgraphs are combined, labeled, etc.

Remarks and examples

stata.com

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

Also see [TS] **irf cgraph**, which produces combined graphs; [TS] **irf ograph**, which produces overlaid graphs; and [TS] **irf table**, which displays results in tabular form.

`irf graph` produces one or more graphs and displays them arrayed into one image unless the `individual` option is specified, in which case the individual graphs are displayed separately. Each individual graph consists of all the specified *stat* and represents one impulse–response combination.

Because all the specified *stat* appear on the same graph, putting together statistics with very different scales is not recommended. For instance, sometimes `sirf` and `oirf` are on similar scales while `irf` is on a different scale. In such cases, combining `sirf` and `oirf` on the same graph looks fine, but combining either with `irf` produces an uninformative graph.

▷ Example 1

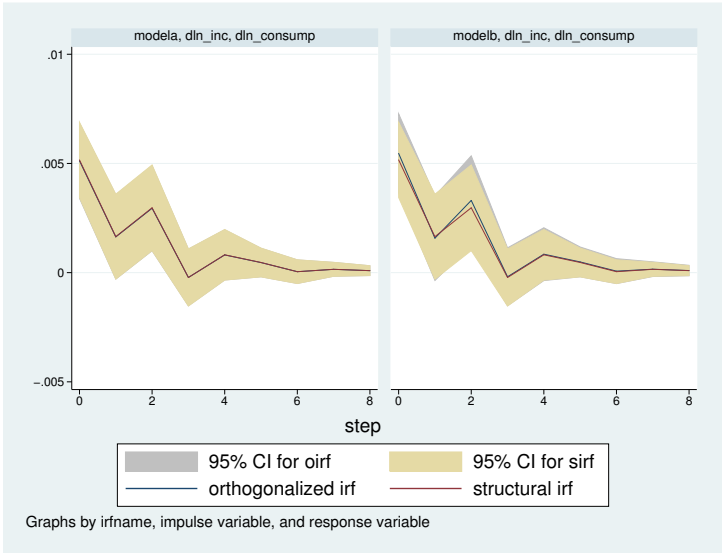
Suppose that we have results generated from two different SVAR models. We want to know whether the shapes of the structural IRFs and the structural FEVDs are similar in the two models. We are also interested in knowing whether the structural IRFs and the structural FEVDs differ significantly from their Cholesky counterparts.

Filling in the background, we have previously issued the commands:

```
. use http://www.stata-press.com/data/r15/lutkepohl2
. mat a = (., 0, 0\0,.,0\.,.,.)
. mat b = I(3)
. svar dln_inv dln_inc dln_consump, aeq(a) beq(b)
. irf create modela, set(results3) step(8)
. svar dln_inc dln_inv dln_consump, aeq(a) beq(b)
. irf create modelb, step(8)
```

To see whether the shapes of the structural IRFs and the structural FEVDs are similar in the two models, we type

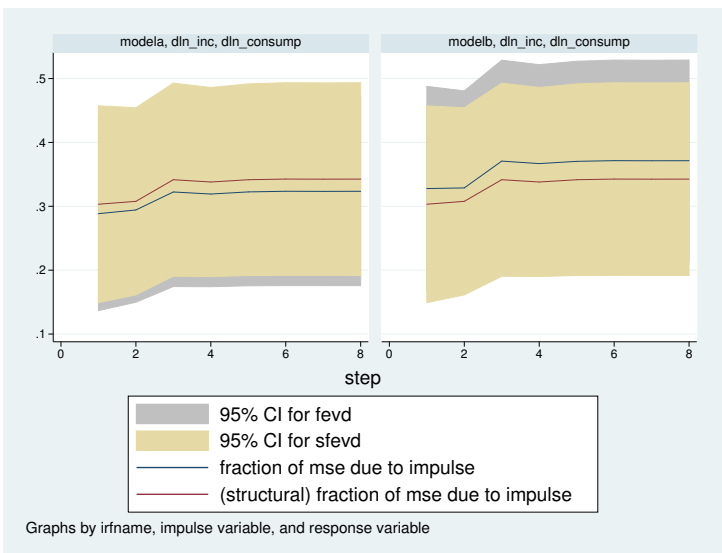
```
. irf graph oirf sirf, impulse(dln_inc) response(dln_consump)
```



The graph reveals that the `oirf` and the `sirf` estimates are essentially the same for both models and that the shapes of the functions are very similar for the two models.

To see whether the structural IRFs and the structural FEVDs differ significantly from their Cholesky counterparts, we type

```
. irf graph fevd sfevd, impulse(dln_inc) response(dln_consump) lstep(1)
> legend(cols(1))
```



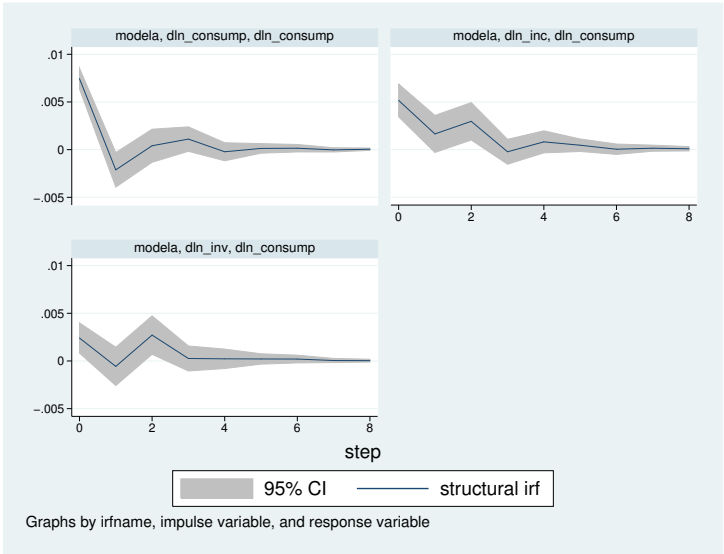
This combined graph reveals that the shapes of these functions are also similar for the two models. However, the graph illuminates one minor difference between them: In `modela`, the estimated structural FEVD is slightly larger than the Cholesky-based estimates, whereas in `modelb` the Cholesky-based estimates are slightly larger than the structural estimates. For both models, however, the structural estimates are close to the center of the wide confidence intervals for the two estimates.

◁

▷ Example 2

Let's focus on the results from `modela`. Suppose that we were interested in examining how `dln_consump` responded to impulses in its own structural innovations, structural innovations to `dln_inc`, and structural innovations to `dln_inv`. We type

```
. irf graph sirf, irf(modela) response(dln_consump)
```



The upper-left graph shows the structural IRF of an innovation in `dln_consump` on `dln_consump`. It indicates that the identification restrictions used in `modela` imply that a positive shock to `dln_consump` causes an increase in `dln_consump`, followed by a decrease, followed by an increase, and so on, until the effect dies out after roughly 5 periods.

The upper-right graph shows the structural IRF of an innovation in `dln_inc` on `dln_consump`, indicating that a positive shock to `dln_inc` causes an increase in `dln_consump`, which dies out after 4 or 5 periods.

◁

❑ Technical note

[TS] **irf table** contains a technical note warning you to be careful in naming variables when you fit models. What is said there applies equally here.

❑

Stored results

`irf graph` stores the following in `r()`:

Scalars

`r(k)`	number of graphs

Macros

`r(stats)`	*statlist*	`r(byopts)`	contents of `byopts()`
`r(irfname)`	*resultslist*	`r(saving)`	supplied `saving()` option
`r(impulse)`	*impulselist*	`r(name)`	supplied `name()` option
`r(response)`	*responselist*	`r(individual)`	`individual` or blank
`r(plot#)`	contents of `plot#opts()`	`r(isaving)`	contents of `saving()`
`r(ci)`	level applied to confidence intervals or `noci`	`r(iname)`	contents of `name()`
		`r(subtitle#)`	subtitle for individual graph #
`r(ciopts#)`	contents of `ci#opts()`		

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