irf cgraph — Combined graphs of IRFs, dynamic-multiplier functions, and FEVDs

Description

`irf cgraph` makes a graph or a combined graph of IRF results. A graph is drawn for specified combinations of named IRF results, impulse variables, response variables, and statistics. `irf cgraph` combines these graphs into one image, unless separate graphs are requested.

`irf cgraph` operates on the active IRF file; see [TS] **irf set**.

Quick start

Combine graphs of an orthogonalized IRF `myirf` and cumulative IRF `mycirf` for dependent variables `y1` and `y2`

```
irf cgraph (myirf y1 y2 oirf) (mycirf y1 y2 cirf)
```

Same as above, but suppress confidence bands and add a title

```
irf cgraph (myirf y1 y2 oirf) (mycirf y1 y2 cirf), noci ///
  title("My Title")
```

Note: `irf` commands can be used after `var`, `svar`, `ivsvar`, `vec`, `arima`, `arfima`, `lpirf`, `ivlpirf`, `dsge`, `dsgenl`, or `xtvar`; see [TS] **var**, [TS] **var svar**, [TS] **var ivsvar**, [TS] **vec**, [TS] **arima**, [TS] **arfima**, [TS] **lpirf**, [TS] **ivlpirf**, [DSGE] **dsge**, [DSGE] **dsgenl**, or [XT] **xtvar**.

Menu

Statistics > Postestimation

Syntax

```
irf cgraph (spec_1) [ (spec_2) ... [ (spec_N) ]] [ , options ]
```

where (*spec*$_k$) is

```
(irfname impulsevar responsevar stat [ , spec_options ])
```

irfname is the name of a set of IRF results in the active IRF file. *impulsevar* should be specified as an endogenous variable for all statistics except `dm` and `cdm`; for those, specify as an exogenous variable. *responsevar* is an endogenous variable name. *stat* is one or more statistics from the list below:

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
`csirf`	cumulative 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.

spec_options	Description
Main	
`noci`	suppress confidence bands
`level(#)`	set confidence level; default is `level(95)`
`lstep(#)`	use # for first step
`ustep(#)`	use # for maximum step
Plot	
`plot#opts(`*cline_options*`)`	affect rendition of the line plotting the # *stat*
CI plot	
`ci#opts(`*area_options*`)`	affect rendition of the confidence interval for the # *stat*

spec_options may be specified within a graph specification, globally, or in both. When specified in a graph specification, *spec_options* affect only the specification in which they are used. When supplied globally, *spec_options* affect all graph specifications. When supplied in both places, options in the graph specification take precedence.

options	Description
`set(`*filename*`)`	make *filename* active
Graph options	
`individual`	graph each combination individually
combine_options	affect appearance of combined graph
graph_options	any options other than `by()` documented in [G-3] ***twoway_options***

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

Options

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

spec_options affect the rendition of the specification and include the following:

Main

`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**. The value set of an overall `level()` can be overridden by the `level()` inside a (*spec*$_k$`)`.

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

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

Plot

`plot1opts(`*cline_options*`)`, ..., `plot4opts(`*cline_options*`)` affect the rendition of the plotted statistics. `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 plot

`ci1opts1(`*area_options*`)` and `ci2opts2(`*area_options*`)` affect the rendition of the confidence intervals for the first (`ci1opts()`) and second (`ci2opts()`) statistics. See [TS] **irf graph** for a description of this option and [G-3] ***area_options*** for the suboptions that change the look of the CI.

Graph options

`individual` specifies that each graph be displayed individually. By default, `irf cgraph` combines the subgraphs into one image.

combine_options affect the appearance of the combined graph; see [G-2] **graph combine**.

graph_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***).

Remarks and examples

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

The relationship between `irf cgraph` and `irf graph` is syntactically and conceptually the same as that between `irf ctable` and `irf table`; see [TS] **irf ctable** for a description of the syntax.

`irf cgraph` is much the same as using `irf graph` to make individual graphs and then using `graph combine` to put them together. If you cannot use `irf cgraph` to do what you want, consider the other approach.

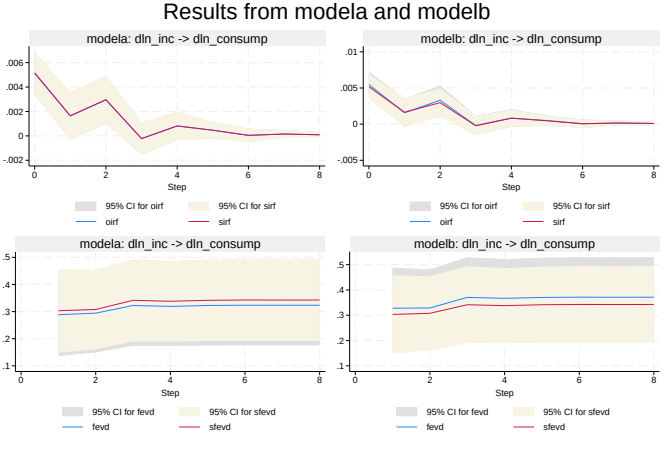
Example 1

You have previously issued the commands:

```
. use https://www.stata-press.com/data/r19/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)
```

You now type

```
. irf cgraph (modela dln_inc dln_consump oirf sirf)
>            (modelb dln_inc dln_consump oirf sirf)
>            (modela dln_inc dln_consump fevd sfevd, lstep(1))
>            (modelb dln_inc dln_consump fevd sfevd, lstep(1)),
>            title("Results from modela and modelb")
```



Stored results

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

Scalars

`r(k)`	number of specific graph commands

Macros

`r(individual)`	`individual`, if specified
`r(save)`	*filename*`, replace` from `saving()` option for combined graph
`r(name)`	*name*`, replace` from `name()` option for combined graph
`r(title)`	title of the combined graph
`r(save#)`	*filename*`, replace` from `saving()` option for individual graphs
`r(name#)`	*name*`, replace` from `name()` option for individual graphs
`r(title#)`	title for the #th graph
`r(ci#)`	level applied to the #th confidence interval or `noci`
`r(response#)`	response specified in the #th command
`r(impulse#)`	impulse specified in the #th command
`r(irfname#)`	IRF name specified in the #th command
`r(stats#)`	statistics specified in the #th command

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