Title

stata.com

meta forestplot — Forest plots

Description

`meta forestplot` summarizes `meta` data in a graphical format. It reports individual effect sizes and the overall effect size (ES), their confidence intervals (CIs), heterogeneity statistics, and more. `meta forestplot` can perform random-effects (RE), common-effect (CE), and fixed-effects (FE) meta-analyses. It can also perform subgroup, cumulative, and sensitivity meta-analyses. For tabular display of meta-analysis summaries, see [META] **meta summarize**.

Quick start

Default forest plot after data are declared by using either `meta set` or `meta esize`

```
meta forestplot
```

As above, but apply the hyperbolic tangent transformation to effect sizes and their CIs

```
meta forestplot, transform(tanh)
```

Add vertical lines at the overall effect-size and no-effect values

```
meta forestplot, esrefline nullrefline
```

Customize the overall effect-size line, and annotate the sides of the plot, with respect to the no-effect line, favoring the treatment or control

```
meta forestplot, esrefline(lcolor(green))       ///
     nullrefline(favorsleft("Favors vaccine")  ///
     favorsright("Favors control"))
```

Add a custom diamond with a label for the overall effect-size ML estimate by specifying its value and CI limits

```
meta forestplot, customoverall(-.71 -1.05 -.37, label("{bf:ML Overall}"))
```

Forest plot based on subgroup meta-analysis

```
meta forestplot, subgroup(groupvar)
```

Forest plot based on cumulative meta-analysis

```
meta forestplot, cumulative(ordervar)
```

Default forest plot after data are declared with `meta set` but with the columns spelled out

```
meta forestplot _id _plot _esci _weight
```

Default forest plot after data are declared with `meta esize` but with the columns spelled out

```
meta forestplot _id _data _plot _esci _weight
```

As above, but with the weights omitted

```
meta forestplot _id _data _plot _esci
```

Same as above, but the columns are rearranged

```
meta forestplot _id _data _esci _plot
```

Same as above, but plot variables `x1` and `x2` as the second and last columns

```
meta forestplot _id x1 _data _esci _plot x2
```

Change the format of the `_esci` column

```
meta forestplot, columnopts(_esci, format(%7.4f))
```

Menu

Statistics > Meta-analysis

Syntax

```
meta forestplot [column_list] [if] [in] [, options]
```

column_list is a list of column names given by *col*. In the *Meta-Analysis Control Panel*, the columns can be specified on the **Forest plot** tab of the Forest plot pane.

options	Description
Main	
`random[(remethod)]`	random-effects meta-analysis
`common[(cefemethod)]`	common-effect meta-analysis
`fixed[(cefemethod)]`	fixed-effects meta-analysis
reopts	random-effects model options
`subgroup(varlist)`	subgroup meta-analysis for each variable in *varlist*
`cumulative(cumulspec)`	cumulative meta-analysis
`sort(varlist[, ...])`	sort studies according to *varlist*
Options	
`level(#)`	set confidence level; default is as declared for meta-analysis
eform_option	report exponentiated results
`transform(transfspec)`	report transformed results
`tdistribution`	report t test instead of z test
`[no]metashow`	display or suppress meta settings in the output
Maximization	
maximize_options	control the maximization process; seldom used
Forest plot	
`columnopts(col, [colopts])`	column options; can be repeated
`cibind(bind)`	change binding of CIs for columns `_esci` and `_ci`; default is `cibind(brackets)`
`sebind(bind)`	change binding of standard errors for column `_esse`; default is `sebind(parentheses)`
`nohrule`	suppress horizontal rule
`hruleopts(hrule_options)`	change look of horizontal rule
text_options	change looks of text options such as column titles, supertitles, and more
plot_options	change look or suppress markers, restrict range of CIs, and more
test_options	suppress information about heterogeneity statistics and tests
graph_options	change the lines, labels, ticks, titles, scheme, etc. on the forest plot
`nooverall`	suppress row corresponding to the overall effect size
`olabel(string)`	modify default overall effect-size label under the `_id` column; default label is `Overall`

`nooverall` and `olabel()` do not appear in the dialog box.

col	Description
Default columns and order	
`_id`	study label
`_data`	summary data; `_data1` and `_data2` (only after `meta esize`)
`_plot`	forest graph
`_esci`	effect size and its confidence interval
`_weight`	percentage of total weight given to each study
Summary-data columns and order	
Continuous outcomes	
Treatment group	
`_data1`	summary data for treatment group; `_n1`, `_mean1`, and `_sd1`
`_n1`	sample size in the treatment group
`_mean1`	mean in the treatment group
`_sd1`	standard deviation in the treatment group
Control group	
`_data2`	summary data for control group; `_n2`, `_mean2`, and `_sd2`
`_n2`	sample size in the control group
`_mean2`	mean in the control group
`_sd2`	standard deviation in the control group
Dichotomous outcomes	
Treatment group	
`_data1`	summary data for treatment group; `_a` and `_b`
`_a`	number of successes in the treatment group
`_b`	number of failures in the treatment group
Control group	
`_data2`	summary data for control group; `_c` and `_d`
`_c`	number of successes in the control group
`_d`	number of failures in the control group
Other columns	
`_es`	effect size
`_ci`	confidence interval for effect size
`_lb`	lower confidence limit for effect size
`_ub`	upper confidence limit for effect size
`_se`	standard error of effect size
`_esse`	effect size and its standard error
`_pvalue`	p-value for significance test with `subgroup()` or `cumulative()`
`_K`	number of studies with `subgroup()`
`_size`	within-group sample size with `subgroup()`
`_order`	order variable for cumulative meta-analysis with `cumulative()`
varname	variable in the dataset (except meta system variables)

Columns _data, _data1, _data2, and the other corresponding data columns are not available after the declaration by using meta set.

Columns _n1, _mean1, _sd1, _n2, _mean2, and _sd2 are available only after the declaration by using meta esize for continuous outcomes.

Columns _a, _b, _c, and _d are available only after the declaration by using meta esize for binary outcomes.

Column _pvalue is available only when option cumulative() or subgroup() with multiple variables is specified.

Columns _K and _size are available only when option subgroup() with multiple variables is specified.

Column *varname* is not available when option subgroup() with multiple variables is specified.

colopts	Description
supertitle(*string*)	super title specification
title(*string*)	title specification
format(%*fmt*)	numerical format for column items
mask(*mask*)	string mask for column items
plotregion(*region_options*)	attributes of plot region
textbox_options	appearance of textboxes

text_options	Description
coltitleopts(*textbox_options*)	change look of column titles and supertitles
itemopts(*textbox_options*)	change look of study rows
overallopts(*textbox_options*)	change look of the overall row
groupopts(*textbox_options*)	change look of subgroup rows
bodyopts(*textbox_options*)	change look of study, subgroup, and overall rows
nonotes	suppress notes about the meta-analysis model, method, and more

plot_options	Description
crop(# #)	restrict the range of CI lines
ciopts(*ci_options*)	change look of CI lines (size, color, etc.)
nowmarkers	suppress weighting of study markers
nomarkers	suppress study markers
markeropts(*marker_options*)	change look of study markers (size, color, etc.)
noomarker	suppress the overall marker
omarkeropts(*marker_options*)	change look of the overall marker (size, color, etc.)
nogmarkers	suppress subgroup markers
gmarkeropts(*marker_options*)	change look of subgroup markers (size, color, etc.)
insidemarker[(*marker_options*)]	add a marker at the center of the study marker
esrefline[(*line_options*)]	add a vertical line corresponding to the overall effect size
nullrefline[(*nullopts*)]	add a vertical line corresponding to no effect
customoverall(*customspec*)	add a custom diamond representing an overall effect; can be repeated

test_options	Description
noohetstats	suppress overall heterogeneity statistics
noohomtest	suppress overall homogeneity test
noosigtest	suppress test of significance of overall effect size
noghetstats	suppress subgroup heterogeneity statistics
nogwhomtests	suppress within-subgroup homogeneity tests
nogbhomtests	suppress between-subgroup homogeneity tests

graph_options	Description
xline(*linearg*)	add vertical lines at specified x values
xtitle(*axis_title*)	specify x-axis title
xlabel(*rule_or_values*)	major ticks plus labels
xtick(*rule_or_values*)	major ticks only
xmlabel(*rule_or_values*)	minor ticks plus labels
xmtick(*rule_or_values*)	minor ticks only
title(*tinfo*)	overall title
subtitle(*tinfo*)	subtitle of title
note(*tinfo*)	note about graph
caption(*tinfo*)	explanation of graph
t1title(*tinfo*) t2title(*tinfo*)	rarely used
b1title(*tinfo*) b2title(*tinfo*)	rarely used
l1title(*tinfo*) l2title(*tinfo*)	vertical text
r1title(*tinfo*) r2title(*tinfo*)	vertical text
scheme(*schemename*)	overall look
nodraw	suppress display of graph
name(*name*, ...)	specify name for graph
saving(*filename*, ...)	save graph in file

nullopts	Description
favorsleft(*string*[, *textbox_options*])	add a label to the left of the no-effect reference line
favorsright(*string*[, *textbox_options*])	add a label to the right of the no-effect reference line
line_options	affect the rendition of the no-effect reference line

Options

Main

random[(*remethod*)], common[(*cefemethod*)], fixed[(*cefemethod*)], subgroup(*varlist*), cumulative(*cumulspec*), and sort(*varlist*[, ...]); see *Options* in [META] **meta summarize**.

reopts are tau2(#), i2(#), predinterval, predinterval(#[, *line_options*]), and se(*seadj*). These options are used with random-effects meta-analysis. See *Options* in [META] **meta summarize**.

`predinterval` and `predinterval(#[, line_options])` draw whiskers extending from the overall effect marker and spanning the width of the prediction interval. *line_options* affect how the whiskers are rendered; see [G-3] ***line_options***.

Options

`level(#)`, *eform_option*, `transform()`, `tdistribution`, and `[no]metashow`; see *Options* in [META] **meta summarize**.

Maximization

maximize_options: `iterate(#)`, `tolerance(#)`, `nrtolerance(#)`, `nonrtolerance`, `from(#)`, and `showtrace`; see *Options* in [META] **meta summarize**.

Forest plot

`columnopts(`*col* `[, `*colopts*`])` changes the look of the column identified by *col*. This option can be repeated.

colopts are the following options:

`supertitle(`*string*`)` specifies that the column's supertitle is *string*.

`title(`*string*`)` specifies that the column's title is *string*.

`format(%`*fmt*`)` specifies the format for the column's numerical values.

`mask(`*mask*`)` specifies a string composed of formats for the column's statistics. For example, *mask* for column `_weight` that identifies the column of weight percentages may be specified as `"%6.2f %%"`.

`plotregion(`*region_options*`)` modifies attributes for the plot region. You can change the margins, background color, an outline, and so on; see [G-3] ***region_options***.

textbox_options affect how the column's items (study and group) are rendered. These options override what is specified in global options `bodyopts()`, `itemopts()`, and `groupopts()`. See [G-3] ***textbox_options***.

Options `format()`, `mask()`, and *textbox_options* are ignored by `_plot`.

`cibind(`*bind*`)` changes the binding of the CIs for columns `_esci` and `_ci`. *bind* is one of `brackets` or `parentheses`. By default, the CIs are bound by using brackets, `cibind(brackets)`. This option is relevant only when `_esci` or `_ci` appears in the plot.

`sebind(`*bind*`)` changes the binding of the standard errors for column `_esse`. *bind* is one of `parentheses` or `brackets`. By default, the standard errors are bound by using parentheses, `cibind(parentheses)`. This option is relevant only when `_esse` appears in the plot.

`nohrule` suppresses the horizontal rule.

`hruleopts(`*hrule_options*`)` affects the look of the horizontal rule.

hrule_options are the following options:

`lcolor(`*colorstyle*`)` specifies the color of the rule; see [G-4] ***colorstyle***.

`lwidth(`*linewidthstyle*`)` specifies the width of the rule; see [G-4] ***linewidthstyle***.

`lalign(`*linealignmentstyle*`)` specifies the alignment of the rule; see [G-4] ***linealignmentstyle***.

`lpattern(`*linepatternstyle*`)` specifies the line pattern of the rule; see [G-4] ***linepatternstyle***.

`lstyle(`*linestyle*`)` specifies the overall style of the rule; see [G-4] ***linestyle***.

margin(*marginstyle*) specifies the margin of the rule; see [G-4] ***marginstyle***.

text_options are the following options:

coltitleopts(*textbox_options*) affects the look of text for column titles and supertitles. See [G-3] ***textbox_options***.

itemopts(*textbox_options*) affects the look of text for study rows; see [G-3] ***textbox_options***. This option is ignored when option subgroup() is specified and contains multiple variables or when option cumulative() is specified.

overallopts(*textbox_options*) affects the look of text for the overall row. See [G-3] ***textbox_options***.

groupopts(*textbox_options*) (synonym subgroupopts()) affects the look of text for subgroup rows when option subgroup() is specified. See [G-3] ***textbox_options***.

bodyopts(*textbox_options*) affects the look of text for study, subgroup, and overall rows. See [G-3] ***textbox_options***.

nonotes suppresses the notes displayed on the graph about the specified meta-analysis model and method and the standard-error adjustment.

plot_options are the following options:

crop(#$_1$ #$_2$) restricts the range of the CI lines to be between #$_1$ and #$_2$. A missing value may be specified for any of the two values to indicate that the corresponding limit should not be cropped. Otherwise, lines that extend beyond the specified value range are cropped and adorned with arrows. This option is useful in the presence of small studies with large standard errors, which lead to confidence intervals that are too wide to be displayed nicely on the graph. Option crop() may be used to handle this case.

ciopts(*ci_options*) affects the look of the CI lines and, in the presence of cropped CIs (see option crop()), arrowheads.

ci_options are any options documented in [G-3] ***line_options*** and the following options of [G-2] **graph twoway pcarrow**: mstyle(), msize(), mangle(), barbsize(), mcolor(), mfcolor(), mlcolor(), mlwidth(), mlstyle(), and color().

nowmarkers suppresses weighting of the study markers.

nomarkers suppresses the study markers.

markeropts(*marker_options*) affects the look of the study markers.

marker_options: msymbol(), mcolor(), mfcolor(), mlcolor(), mlwidth(), mlalign(), mlstyle(), and mstyle(); see [G-3] ***marker_options***.

nowmarkers, nomarkers, and markeropts() are ignored when option subgroup() is specified and contains multiple variables or when option cumulative() is specified.

noomarker suppresses the overall marker.

omarkeropts(*marker_options*) affects the look of the overall marker.

marker_options: mcolor(), mfcolor(), mlcolor(), mlwidth(), mlalign(), mlstyle(), and mstyle(); see [G-3] ***marker_options***.

nogmarkers suppresses the subgroup markers.

gmarkeropts(*marker_options*) affects the look of the subgroup markers.

marker_options: mcolor(), mfcolor(), mlcolor(), mlwidth(), mlalign(), mlstyle(), and mstyle(); see [G-3] ***marker_options***.

nogmarkers and gmarkeropts() are ignored when option subgroup() is not specified.

insidemarker and insidemarker(*marker_options*) add markers at the center of study markers. *marker_options* control how the added markers are rendered.

marker_options: msymbol(), mcolor(), mfcolor(), mlcolor(), mlwidth(), mlalign(), mlstyle(), and mstyle(); see [G-3] ***marker_options***.

insidemarker() is not allowed when option subgroup() is specified and contains multiple variables or when option cumulative() is specified.

esrefline and esrefline(*line_options*) specify that a vertical line be drawn at the value corresponding to the overall effect size. The optional *line_options* control how the line is rendered; see [G-3] ***line_options***.

nullrefline and nullrefline(*nullopts*) specify that a vertical line be drawn at the value corresponding to no overall effect. *nullopts* are the following options:

favorsleft(*string* [, *textbox_options*]) adds a label, *string*, to the left side (with respect to the no-effect line) of the forest graph. *textbox_options* affect how *string* is rendered; see [G-3] ***textbox_options***.

favorsright(*string* [, *textbox_options*]) adds a label, *string*, to the right side (with respect to the no-effect line) of the forest graph. *textbox_options* affect how *string* is rendered; see [G-3] ***textbox_options***.

favorsleft() and favorsright() are typically used to annotate the sides of the forest graph (column _plot) favoring the treatment or control.

line_options affect the rendition of the vertical line; see [G-3] ***line_options***.

customoverall(*customspec*) draws a custom-defined diamond representing an overall effect size. This option can be repeated. *customspec* is *#es #lb #ub* [, *customopts*], where *#es*, *#lb*, and *#ub* correspond to an overall effect-size estimate and its lower and upper CI limits, respectively. *customopts* are the following options:

label(*string*) adds a label, *string*, under the _id column describing the custom diamond.

textbox_options affect how label(*string*) is rendered; see [G-3] ***textbox_options***.

marker_options affect how the custom diamond is rendered. *marker_options* are mcolor(), mfcolor(), mlcolor(), mlwidth(), mlalign(), mlstyle(), and mstyle(); see [G-3] ***marker_options***.

Option customoverall() may not be combined with option cumulative().

test_options are defined below. These options are not relevant with cumulative meta-analysis.

noohetstats suppresses overall heterogeneity statistics reported under the Overall row heading on the plot.

noohomtest suppresses the overall homogeneity test labeled as Test of θ_i=θ_j under the Overall row heading on the plot.

noosigtest suppresses the test of significance of the overall effect size labeled as Test of θ=0 under the Overall row heading on the plot. This test is not reported with subgroup analyses, and thus this option is implied when option subgroup() is specified.

noghetstats suppresses subgroup heterogeneity statistics reported when a single subgroup analysis is performed, that is, when option subgroup() is specified with one variable. These statistics are reported under the group-specific row headings.

nogwhomtests suppresses within-subgroup homogeneity tests. These tests investigate the differences between effect sizes of studies within each subgroup. These tests are reported when a single subgroup analysis is performed, that is, when option subgroup() is specified with one variable. The tests are labeled as Test of θ_i=θ_j under the group-specific row headings.

nogbhomtests suppresses between-subgroup homogeneity tests reported when any subgroup analysis is performed, that is, when option subgroup() is specified. These tests investigate the differences between the subgroup overall effect sizes and are labeled as Test of group differences on the plot.

graph_options: xline(), xtitle(), xlabel(), xtick(), xmlabel(), xmtick(), title(), subtitle(), note(), caption(), t1title(), t2title(), b1title(), b2title(), l1title(), l2title(), r1title(), r2title(), scheme(), nodraw, name(), and saving(); see [G-3] ***twoway_options*** for details.

The following options are available with meta forestplot but are not shown in the dialog box:

nooverall suppresses the row corresponding to the overall effect size in the forest plot.

olabel(*string*) modifies the default overall effect-size label under the _id column, which, by default, is Overall.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Overview

Meta-analysis results are often presented using a forest plot (for example, Lewis and Ellis [1982]). A forest plot shows effect-size estimates and their confidence intervals for each study and, usually, the overall effect size from the meta-analysis (for example, Lewis and Clarke [2001]; Harris et al. [2016]; and Fisher 2016). Each study is represented by a square with the size of the square being proportional to the study weight; that is, larger squares correspond to larger (more precise) studies. The weights depend on the chosen meta-analysis model and method. Studies' CIs are plotted as whiskers extending from each side of the square and spanning the width of the CI. Heterogeneity measures such as the I^2 and H^2 statistics, homogeneity test, and the significance test of the overall effect sizes are also commonly reported.

A subgroup meta-analysis forest plot also shows group-specific results. Additionally, it reports a test of the between-group differences among the overall effect sizes. A cumulative meta-analysis forest plot shows the overall effect sizes and their CIs by accumulating the results from adding one study at a time to each subsequent analysis. By convention, group-specific and overall effect sizes are represented by diamonds centered on their estimated values with the diamond width corresponding to the CI length.

For more details about forest plots, see, for instance, Anzures-Cabrera and Higgins (2010). Also see Schriger et al. (2010) for an overview of their use in practice.

Using meta forestplot

meta forestplot produces meta-analysis forest plots. It provides a graphical representation of the results produced by meta summarize and thus supports most of its options such as those specifying a meta-analysis model and estimation method; see [META] **meta summarize**.

The default look of the forest plot produced by meta forestplot depends on the type of analysis. For basic meta-analysis, meta forestplot plots the study labels, effect sizes and their confidence intervals, and percentages of total weight given to each study. If meta esize was used to declare meta data, summary data are also plotted. That is,

```
. meta forestplot
```

is equivalent to typing

```
. meta forestplot _id _plot _esci _weight
```

after declaration by using meta set and to typing

```
. meta forestplot _id _data _plot _esci _weight
```

after declaration by using meta esize.

If multiple variables are specified in the subgroup() option,

```
. meta forestplot, subgroup(varlist)
```

is equivalent to typing

```
. meta forestplot _id _K _plot _esci _pvalue, subgroup(varlist)
```

For cumulative meta-analysis,

```
. meta forestplot, cumulative(varname)
```

is equivalent to typing

```
. meta forestplot _id _plot _esci _pvalue _order, cumulative(varname)
```

You can also specify any of the supported columns with meta forestplot, including variables in your dataset. For example, you may include, say, variables x1 and x2, as columns in the forest plot by simply specifying them in the column list,

```
. meta forestplot _id x1 _plot _esci _weight x2
```

See *Plot columns* for details about the supported columns.

The CIs correspond to the confidence level as declared by meta set or meta esize. You can specify a different level in the level() option. Also, by default, the CIs are bound by using brackets. You can specify cibind(parentheses) to use parentheses instead.

As we mentioned earlier, you can produce forest plots for subgroup analysis by using the subgroup() option and for cumulative meta-analysis by using the cumulative() option.

You can modify the default column supertitles, titles, formats, and so on by specifying the columnopts() option. You can repeat this option to modify the look of particular columns. If you want to apply the same formatting to multiple columns, you can specify these columns within columnopts(). See *Options* for the list of supported column options.

Options esrefline() and nullrefline() are used to draw vertical lines at the overall effect-size and no-effect values, respectively. Suboptions favorsleft() and favorsright() of nullrefline() may be specified to annotate the sides (with respect to the no-effect line) of the plot favoring treatment or control.

Another option you may find useful is crop(#$_1$ #$_2$). Sometimes, some of the smaller studies may have large standard errors that lead to CIs that are too wide to be displayed on the plot. You can "crop" such CIs by restricting their range. The restricted range will be indicated by the arrowheads at the corresponding ends of the CIs. You can crop both limits or only one of the limits. You can modify the default look of the arrowheads or CI lines, in general, by specifying ciopts().

You may sometimes want to show the overall effect-size estimates from multiple meta-analysis models (for example, common versus random), from different estimation methods (REML versus DL), or for specific values of moderators from a meta-regression. This may be accomplished via the customoverall(*#es #lb #ub* [,*customopts*]) option. This option may be repeated to display multiple diamonds depicting multiple custom-defined overall effect sizes.

You can specify many more options to customize the look of your forest plot such as modifying the look of text for column titles in coltitleopts() or the column format in format(); see *Syntax* for details.

meta forestplot uses the following default convention when displaying the results. The results from individual studies—individual effects sizes—are plotted as navy squares with areas proportional to study weights. The overall effect size is plotted as a green (or, more precisely, forest green using Stata's color convention) diamond with the width corresponding to its CI. The results of a single subgroup analysis—subgroup effect sizes—are plotted as maroon diamonds with the widths determined by the respective CIs. The results of multiple subgroup analyses are plotted as maroon circles with the CI lines. The cumulative meta-analysis results—cumulative overall effect sizes—are displayed as green circles with CI lines.

Options itemopts(), nomarkers, and markeropts() control the look of study rows and markers, which represent individual effect sizes. These options are not relevant when individual studies are not reported such as with multiple subgroup analysis and cumulative meta-analysis.

Options groupopts(), nogmarkers, and gmarkeropts() control the look of subgroup rows and markers and are relevant only when subgroup analysis is performed by specifying the subgroup() option.

Options overallopts(), noomarker, and omarkeropts() control the look of overall rows and markers, which represent the overall effect sizes. These options are always applicable because the overall results are always displayed by default. With cumulative meta-analysis, these options affect the displayed cumulative overall effect sizes.

Graphs created by meta forestplot cannot be combined with other Stata graphs using graph combine.

Plot columns

meta forestplot supports many columns that you can include in your forest plot; see the list of supported columns in *Syntax*. The default columns plotted for various analyses were described in *Using meta forestplot* above. Here we provide more details about some of the supported columns.

meta forestplot provides individual columns such as _es and _se and column shortcuts such as _esse. Column shortcuts are typically shortcuts for specifying multiple columns. For instance, column _data is a shortcut for columns _data1 and _data2, which themselves are shortcuts to individual summary-data columns. For continuous data, _data1 is a shortcut for columns _n1, _mean1, and _sd1, and _data2 is a shortcut for _n2, _mean2, and _sd2. For binary data, _data1 corresponds to the treatment-group numbers of successes and failures, _a and _b, and _data2 to the respective numbers in the control group, _c and _d. Column _data and the corresponding summary-data columns are available only after declaration by using meta esize.

The other column shortcuts are _ci, _esci, and _esse. In addition to serving as shortcuts to the respective columns (_lb and _ub; _es, _lb, and _ub; and _es and _se), these shortcut columns have additional properties. For instance, when you specify _ci, the lower and upper CI bounds are separated with a comma, bounded in brackets, and share a title. That is,

```
. meta forestplot _ci
```

is similar to specifying

```
. meta forestplot _lb _ub,
> columnopts(_lb _ub, title(95% CI))
> columnopts(_lb, mask("[%6.2f,"))
> columnopts(_ub, mask("%6.2f]"))
```

Similarly, _esci additionally combines _es and _ci with the common column title, and _esse combines _es and _se and bounds the standard errors in parentheses. _ci, _esci, and _esse also apply other properties to improve the default look of the specified columns such as modifying the default column margins by specifying plotregion(margin()).

If you want to modify the individual columns of the shortcuts, you need to specify the corresponding column names in columnopts(). For instance, if we want to display the effect sizes of the _esci column with three decimal digits but continue using the default format for CIs, we can type

```
. meta forestplot _esci, columnopts(_es, format(%6.3f))
```

If we specify _esci instead of _es in columnopts(),

```
. meta forestplot _esci, columnopts(_esci, format(%6.3f))
```

both effect sizes and CIs will be displayed with three decimal digits. On the other hand, if we want to change the default title and supertitle for _esci, we should specify _esci in columnopts(),

```
. meta forestplot _esci, columnopts(_esci, supertitle("My ES") title("with my CI"))
```

Also see example 6 and example 7 for more examples of customizing the default look of columns.

Column _plot corresponds to the plot region that contains graphical representation of the effect sizes and their confidence intervals. You can modify the default look of the plot by specifying the *plot_options* in *Syntax*.

Column _es corresponds to the plotted effect sizes. For basic meta-analysis, this column displays the individual and overall effect sizes. For subgroup meta-analysis, it also displays subgroup-specific overall effect sizes. For cumulative meta-analysis, it displays the overall effect sizes corresponding to the accumulated studies.

Some of the columns such as _pvalue, _K, _size and _order are available only with specific meta-analyses. _pvalue is available only with multiple subgroup analyses or with cumulative analysis; it displays the p-values of the significant tests of effect sizes. _K and _size are available with multiple subgroup analyses and display the number of studies and total sample size within each subgroup, respectively. _order is available only with cumulative meta-analysis; it displays the values of the specified ordering variable.

You may also add variables in your dataset to the forest plot. For instance, if you want to display variables x1 and x2 in the second and last columns, you may type

```
. meta forestplot _id x1 _plot _esci _weight x2
```

Duplicate columns are ignored with meta forestplot. Also, column shortcuts take precedence. That is, if you specified both _es and _esci, the latter will be displayed.

Examples of using meta forestplot

In this section, we demonstrate some of the uses of `meta forestplot`. The examples are presented under the following headings:

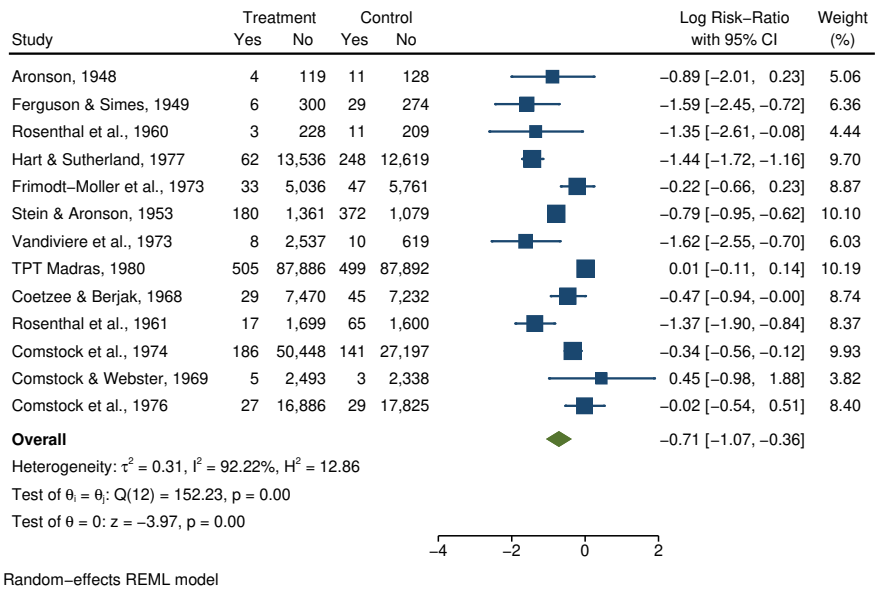
▷ Example 1: Forest plot for binary data

Consider the dataset from Colditz et al. (1994) of clinical trials that studies the efficacy of a Bacillus Calmette-Guérin (BCG) vaccine in the prevention of tuberculosis (TB). This dataset was introduced in *Efficacy of BCG vaccine against tuberculosis (bcg.dta)* of [META] **meta**. In this section, we use its declared version and focus on the demonstration of various options of `meta forest`.

```
. use https://www.stata-press.com/data/r16/bcgset
(Efficacy of BCG vaccine against tuberculosis; set with -meta esize-)
```

Let's construct a basic forest plot by simply typing

```
. meta forestplot
  Effect-size label:  Log Risk-Ratio
        Effect size:  _meta_es
          Std. Err.:  _meta_se
        Study label:  studylbl
```



By default, the basic forest plot displays the study labels (column _id), the summary data (_data), graphical representation of the individual and overall effect sizes and their CIs (_plot), the corresponding values of the effect sizes and CIs (_esci), and the percentages of total weight for each study (_weight). You can also customize the columns on the forest plot; see example 6 and example 11.

In the graph, each study corresponds to a navy square centered at the point estimate of the effect size with a horizontal line (whiskers) extending on either side of the square. The centers of the squares (the values of study effect sizes) may be highlighted via the insidemarker() option; see example 9. The horizontal line depicts the CI. The area of the square is proportional to the corresponding study weight.

The overall effect size corresponds to the green diamond centered at the estimate of the overall effect size. The width of the diamond corresponds to the width of the overall CI. Note that the height of the diamond is irrelevant. It is customary in meta-analysis forest plots to display an overall effect size as a diamond filled inside with color. This, however, may overemphasize the actual area of the diamond whereas only the width of it matters. If desired, you may suppress the fill color by specifying the omarkeropts(mfcolor(none)) option.

Under the diamond, three lines are reported. The first line contains heterogeneity measures I^2, H^2, and $\widehat{\tau}^2$. The second line displays the homogeneity test based on the Q statistic. The third line displays the test of the overall effect size being equal to zero. These lines may be suppressed by specifying options noohetstats, noohomtest, and noosigtest. See [META] **meta summarize** for a substantive interpretation of these results.

Some forest plots show vertical lines at the no-effect and overall effect-size values. These may be added to the plot via options nullrefline() and esrefline(), respectively; see example 4. Also, you may sometimes want to plot custom-defined overall effect sizes such as based on multiple meta-analysis models. This may be accomplished via the customoverall(); see example 11.

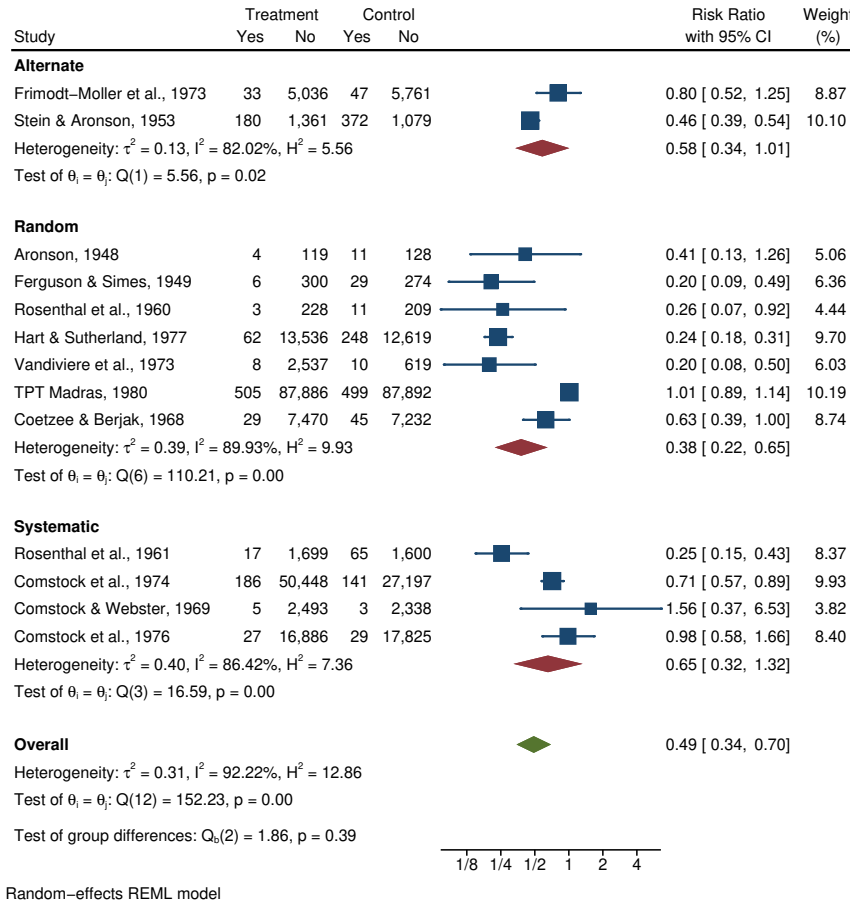
meta forestplot provides a quick way to assess between-study heterogeneity visually. In the absence of heterogeneity, we would expect to see that the middle points of the squares are close to the middle of the diamond and the CIs are overlapping. In these data, there is certainly evidence of some heterogeneity because the squares for some studies are far away from the diamond and there are studies with nonoverlapping CIs.

◁

▷ Example 2: Subgroup-analysis forest plot

Continuing with example 1, let's now perform a subgroup meta-analysis based on the method of treatment allocation recorded in variable alloc. We specify subgroup(alloc) and also use the eform option to display exponentiated results, risk ratios (RRs) instead of log risk-ratios in our example.

```
. meta forestplot, subgroup(alloc) eform
  Effect-size label:  Log Risk-Ratio
        Effect size:  _meta_es
          Std. Err.:  _meta_se
        Study label:  studylbl
```



In addition to the overall results, the forest plot shows the results of meta-analysis for each of the three groups. With subgroup meta-analysis, each group gets its own maroon diamond marker that represents the group-specific overall effect size. Just like with the overall diamond, only the widths (not the heights) of the group-specific diamonds are relevant on the plot. Similarly to the overall marker, you can specify the `gmarkeropts(mfcolor(none))` option to suppress the fill color for the group-specific diamonds.

Heterogeneity measures and homogeneity tests are reported at the bottom (below the group-specific diamond marker) within each group. These provide information regarding the heterogeneity among the studies within each group. They may be suppressed with options `noghetstats` and `nogwhomtests`, respectively. Notice that the significance test of the overall effect size equal to zero is no longer reported under Overall summary for subgroup analysis.

A test of between-group differences based on the Q_b statistic is reported at the bottom. This test investigates the difference between the group-specific overall effect sizes. It may be suppressed with `nogbhomtests`.

You may also specify multiple variables in `subgroup()`, in which case a separate subgroup analysis is performed for each variable; see example 5 for details.

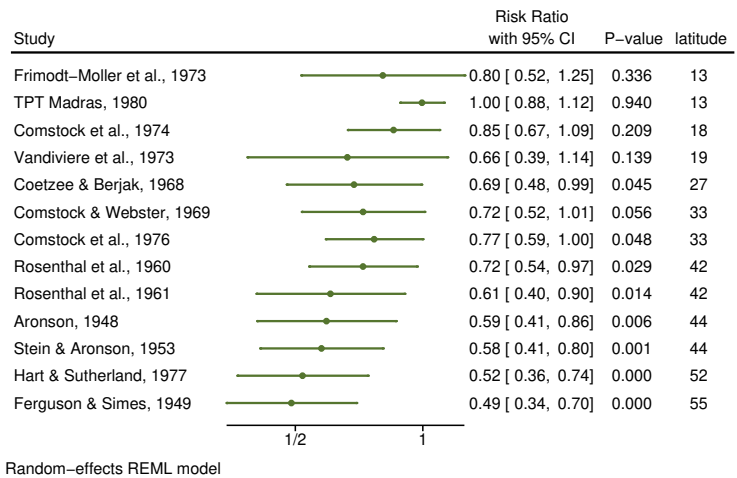
◁

▷ Example 3: Cumulative forest plot

Continuing with example 1, we now perform a cumulative meta-analysis in the ascending order of variable `latitude`. You can specify suboption `descending` within the `cumulative()` option to request a descending order.

We also specify `rr`, which is a synonym of the `eform` option we used in example 2, to display the RRs instead of the default log risk-ratios.

```
. meta forestplot, cumulative(latitude) rr
  Effect-size label:  Log Risk-Ratio
        Effect size:  _meta_es
          Std. Err.:  _meta_se
        Study label:  studylbl
```

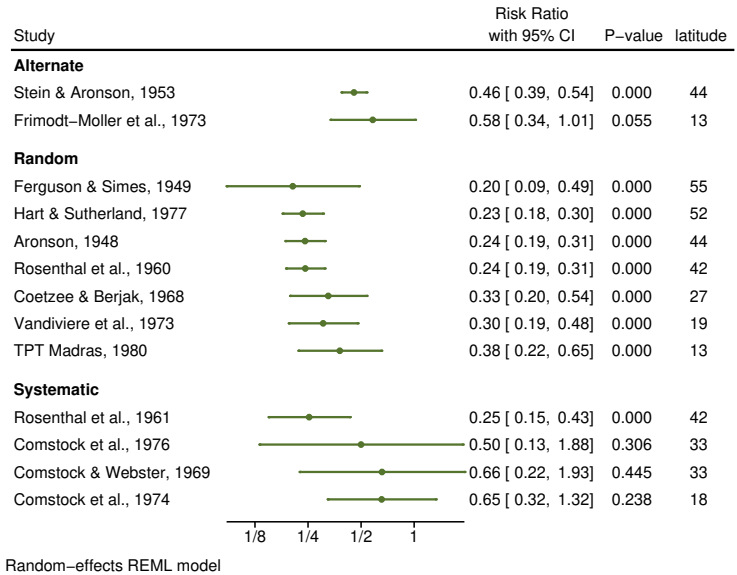


By default, the cumulative meta-analysis forest plot displays the study labels (`_id`), the plot of effect sizes and their CIs (`_plot`), the values of effect sizes and their CIs (`_esci`), the p-values (`_pvalue`) of the corresponding significance tests of the effect sizes, and the values of the order variable (`_order`).

The displayed effect sizes correspond to cumulative overall effect sizes or the overall effect sizes computed for each set of accumulated studies. To distinguish them from study-specific effect sizes, we plot them as unweighted circles using the same color, green, as the overall effect size in a standard meta-analysis forest plot. You can change the default style and color of the markers by specifying the `omarkeropts()` option. The corresponding CIs of the cumulative effect sizes are plotted as CI lines.

We may construct a cumulative forest plot stratified by a covariate by specifying `by()` within `cumulative()`. For example, let's stratify our cumulative analysis by the method of treatment allocation recorded in variable `alloc`.

```
. meta forestplot, cumulative(latitude, by(alloc) descending) rr
  Effect-size label:  Log Risk-Ratio
        Effect size:  _meta_es
          Std. Err.:  _meta_se
        Study label:  studylbl
```



We specified that the analysis be conducted in the descending order of the `latitude` variable. The stratified forest plot shows the same columns as before but the cumulative analysis is performed separately for each group of `alloc`. A consistent pattern is observed across all three groups—RRs tend to increase as latitude decreases.

◁

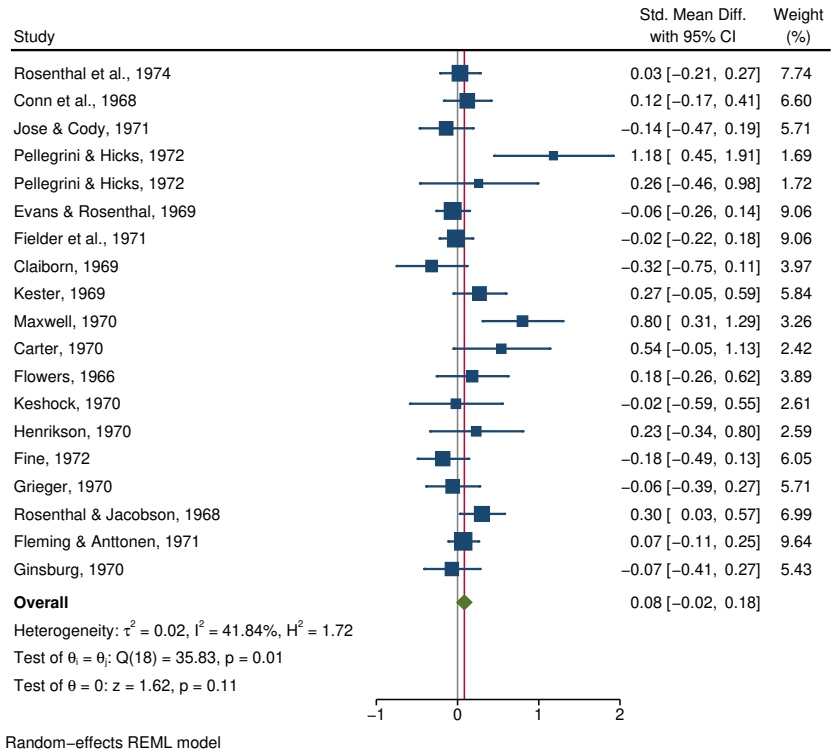
▷ Example 4: Forest plot for precomputed effect sizes

Recall the pupil IQ data (Raudenbush and Bryk 1985; Raudenbush 1984) described in *Effects of teacher expectancy on pupil IQ (pupiliq.dta)* of [META] **meta**. Here we will use its declared version (declared with `meta set`) to construct a forest plot of precomputed effect sizes.

```
. use https://www.stata-press.com/data/r16/pupiliqset, clear
(Effects of teacher expectancy on pupil IQ; set with -meta set-)
```

We specify the `nullrefline` option to show the no-effect line at 0. Effect sizes with corresponding CIs that cross this line are not statistically significant at the 5% level. We also specify the `esrefline` option to draw a vertical line at the overall effect-size value. The default look of both lines may be modified by specifying options `nullrefline(`*line_options*`)` and `esrefline(`*line_options*`)`. See [G-3] ***line_options***.

```
. meta forestplot, nullrefline esrefline
  Effect-size label:  Std. Mean Diff.
        Effect size:  stdmdiff
          Std. Err.:  se
        Study label:  studylbl
```



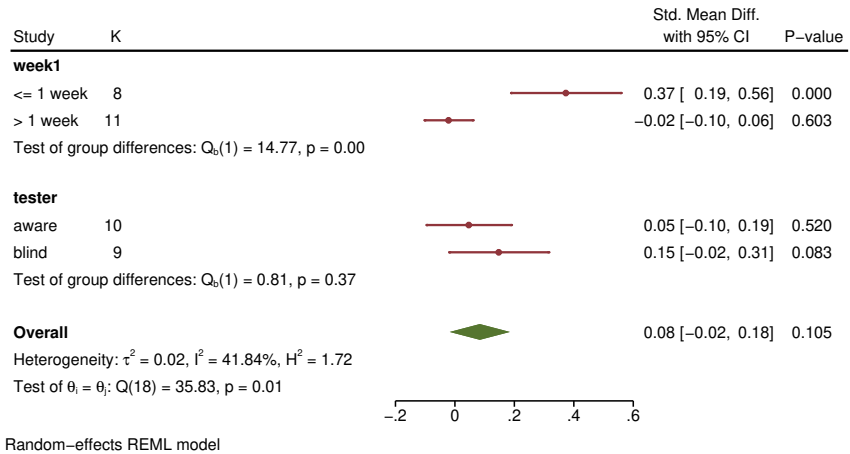
When `meta` data are declared by using `meta set` (that is, when we are working with precomputed effect sizes), the `_data` column is not available. If desired, you may plot the values of effect sizes and their standard errors by specifying the `_esse` column. Other components of the graph are interpreted as in example 1.

◁

Example 5: Multiple subgroup-analyses forest plot

Continuing with example 4, we will conduct multiple subgroup analyses and summarize their results on a forest plot. We specify variables `tester` and `week1` in `subgroup()` as follows:

```
. meta forestplot, subgroup(week1 tester)
  Effect-size label:  Std. Mean Diff.
        Effect size:  stdmdiff
          Std. Err.:  se
        Study label:  studylbl
```



By default, the forest plot displays the study labels (`_id`), the number of studies within each group (`_K`), the plot of effect sizes and their CIs (`_plot`), the values of effect sizes and their CIs (`_esci`), and the p-values (`_pvalue`) of the corresponding significance tests.

To keep the output compact, the forest plot does not report individual studies, only the number of studies in each group. The between-group homogeneity test based on the Q_b is reported for each subgroup analysis. For example, for subgroup analysis based on variable `week1`, there are two groups, `<= 1 week` and `> 1 week`. The test investigates whether the overall effect sizes corresponding to these two groups are the same. The results of this test are identical to those we would have obtained if we had specified `subgroup(week1)`.

Just like with cumulative meta-analysis in example 3, `meta forestplot` uses unweighted circles and CI lines to display the overall group-specific effect sizes and their CIs. But here the circles are displayed in maroon—the same color used to display the group-specific diamonds in a single-variable subgroup analysis (see example 2).

◁

Example 6: Modifying columns' order and cropping confidence intervals

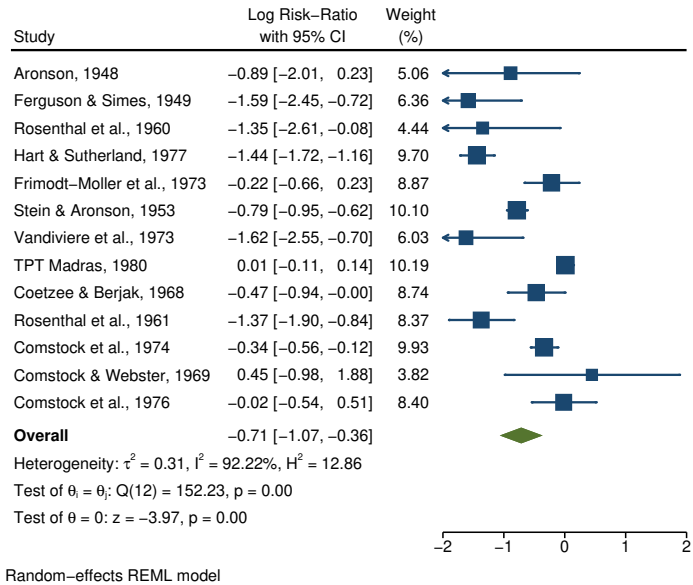
For this and the following examples, let's return to our BCG dataset from example 1.

```
. use https://www.stata-press.com/data/r16/bcgset, clear
(Efficacy of BCG vaccine against tuberculosis; set with -meta esize-)
. meta update, nometashow
```

We used `meta update` to suppress the meta setting information displayed by `meta forest` for the rest of our meta-analysis.

We can choose which columns to display and the order in which to display them in the forest plot by specifying the corresponding column names in the desired order. In the code below, we display the study labels first, followed by the effect sizes and their CIs, then weights, and finally the plot. We also use the `crop(-2 .)` option to restrict the range of the CIs at a lower limit of -2.

```
. meta forestplot _id _esci _weight _plot, crop(-2 .)
```



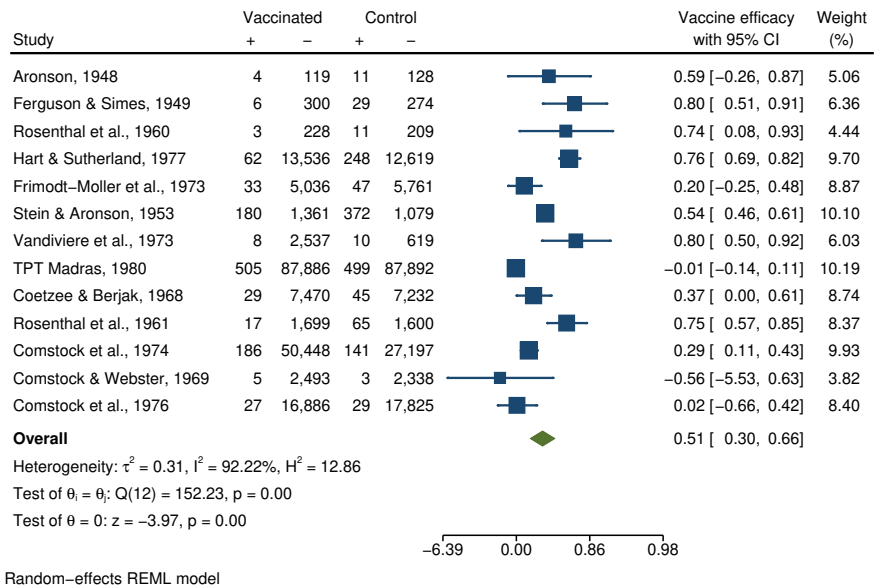
CIs that extend beyond the lower limit of -2 are identified with an arrow head at the cropped endpoint.

Example 7: Applying transformations and changing titles and supertitles

Continuing with the BCG dataset from example 6, we demonstrate how to override the default title and supertitle for the _data column. The binary data we have correspond to a 2×2 table with cells _a, _b, _c, and _d. Cells _a and _b may be referred to as _data1, and cells _c and _d may be referred to as _data2.

We override the supertitle for the _data1 column to display "Vaccinated" and the titles for each cell to display either a "+" or a "−" as follows. We also use the transform() option to report vaccine efficacies instead of log risk-ratios. Vaccine efficacy is defined as $1 - \text{RR}$ and may be requested by specifying transform(efficacy).

```
. meta forestplot, transform(Vaccine efficacy: efficacy)
> columnopts(_data1, supertitle(Vaccinated))
> columnopts(_a _c, title(+)) columnopts(_b _d, title(-))
```

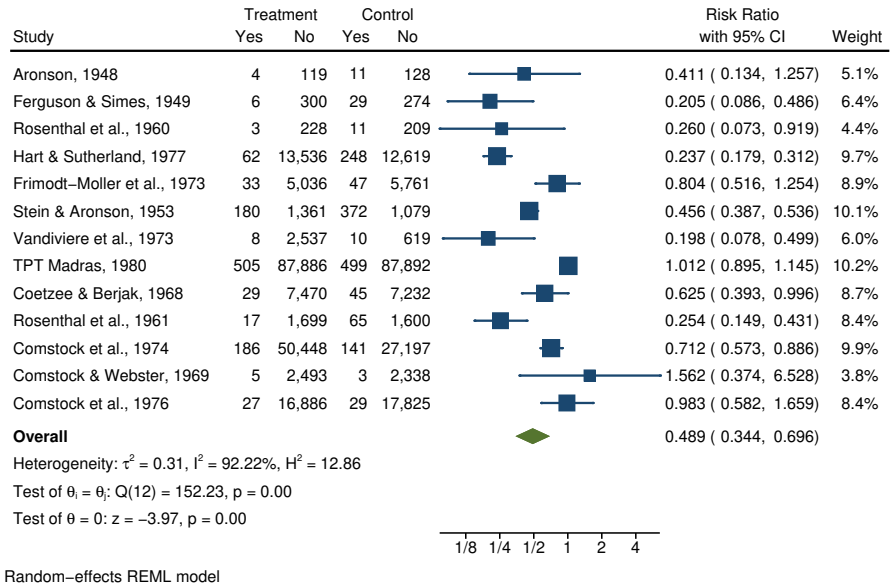


By specifying transform(Vaccine efficacy: efficacy), we also provided a meaningful label, "Vaccine efficacy", to be used for the transformed effect sizes. The overall vaccine efficacy is 0.51, which can be interpreted as a reduction of 51% in the risk of having tuberculosis among the vaccinated group.

Example 8: Changing columns' formatting

Following example 7, we now demonstrate how to override the default formats for the _esci and _weight columns. For the _esci column, we also specify that the CIs be displayed inside parentheses instead of the default brackets. For the _weight column, we specify that the plotted weights be adorned with a % sign and modify the title and supertitle accordingly.

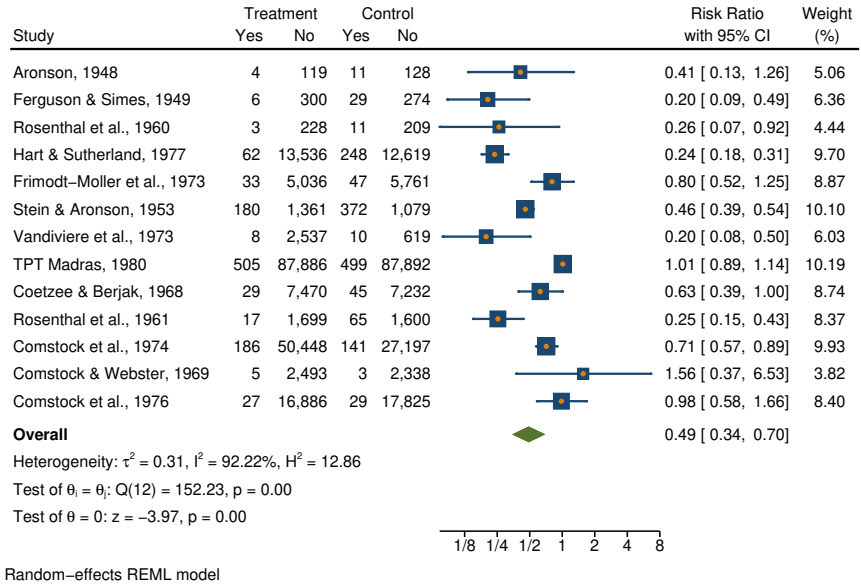
```
. meta forestplot, eform cibind(parentheses)
> columnopts(_esci, format(%6.3f))
> columnopts(_weight, mask(%6.1f%%) title(Weight) supertitle(""))
```



Example 9: Changing axis range and adding center study markers

In this example, we specify the `xscale(range(.125 8))` and `xlabel(#7)` options to specify that the x-axis range be symmetric (on the risk-ratio scale) about the no-effect value of 1 and that 7 tick marks be shown on the axis.

```
. meta forest, eform xscale(range(.125 8)) xlabel(#7) insidemarker
```

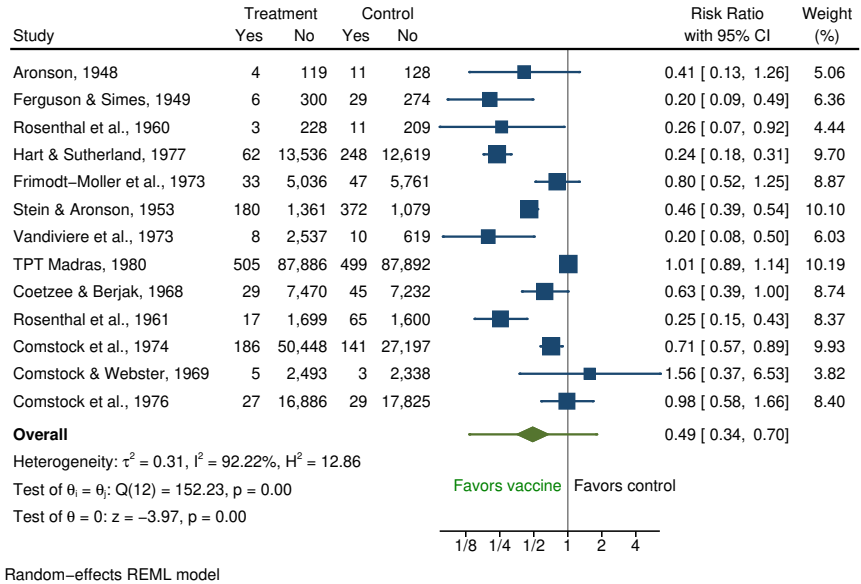


We also used the `insidemarker` option to insert a marker (orange circle) at the center of the study markers (navy-blue squares) to indicate the study-specific effect sizes. The default attributes of the inserted markers may be modified by specifying `insidemarker(`*marker_options*`)`; see [G-3] ***marker_options***.

Example 10: Prediction intervals and sides favoring control or treatment

Below, we specify the `favorsleft()` and `favorsright()` suboptions of the `nullrefline()` option to annotate the sides of the plot (with respect to the no-effect line) favoring the treatment or control. We will also specify the `predinterval` option to draw the prediction interval for the overall effect size.

```
. meta forest, eform predinterval
> nullrefline(
> favorsleft("Favors vaccine", color(green))
> favorsright("Favors control")
> )
```



In our example, the effect sizes that are falling on the "Favors vaccine" side (left side) reported that the treatment (vaccine) reduced the risk of tuberculosis. The default placement of the labels may be modified using the Graph Editor; see [G-1] **Graph Editor**.

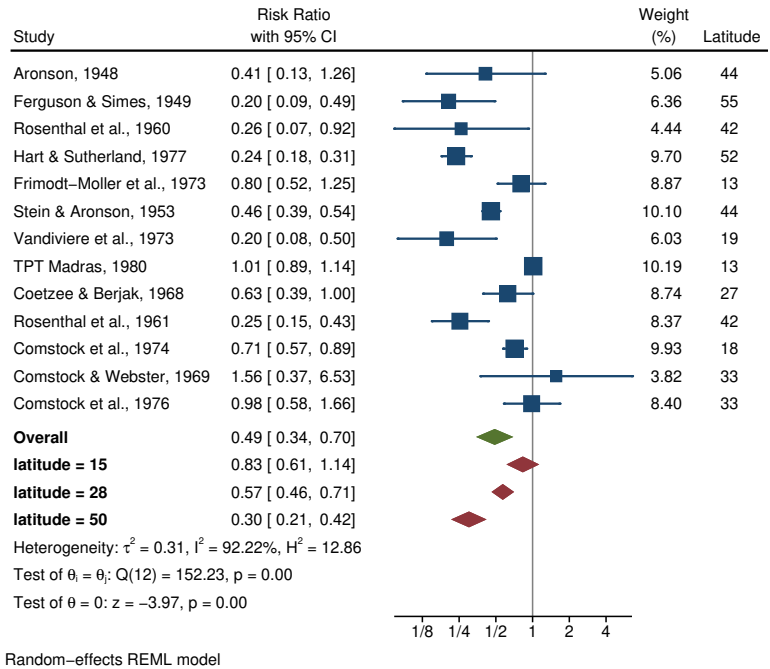
The prediction interval, represented by the green whiskers extending from the overall diamond, provides a plausible range for the effect size in a future, new study.

Example 11: Adding custom columns and overall effect sizes

Consider example 2 of [META] **meta regress postestimation**. We will use the results of the `margins` command from that example to display overall effect sizes at the specified latitudes on the forest plot. This may be done by specifying multiple `customoverall()` options, as we show below.

We also add the `latitude` variable to the forest plot (as the last column) to show study effect sizes as a function of that variable. And we swap the `_esci` and `_plot` columns compared with the default forest plot.

```
. local col mcolor("maroon")
. meta forest _id _esci _plot _weight latitude, nullrefline
> columnopts(latitude, title("Latitude"))
> customoverall(-.184 -.495  .127, label("{bf:latitude = 15}") 'col')
> customoverall(-.562 -.776 -.348, label("{bf:latitude = 28}") 'col')
> customoverall(-1.20 -1.54 -.867, label("{bf:latitude = 50}") 'col')
> rr
```



The latitude-specific overall effect sizes from the meta-regression model are shown as maroon diamonds. In the `customoverall()` options, we specified the values of log risk-ratios, effect sizes in the estimation metric. But because we used the `rr` option, `meta forestplot` displayed the overall diamonds as risk ratios. For example, the mean risk ratio for studies conducted at `latitude` $= 50$ is roughly 0.30 with a CI of $[0.2, 0.4]$.

Methods and formulas

Methods and formulas for the statistics reported by `meta forestplot` are given in *Methods and formulas* of [META] **meta summarize**.

References

Anzures-Cabrera, J., and J. P. T. Higgins. 2010. Graphical displays for meta-analysis: An overview with suggestions for practice. *Research Synthesis Methods* 1: 66–80.

Colditz, G. A., T. F. Brewer, C. S. Berkey, M. E. Wilson, E. Burdick, H. V. Fineberg, and F. Mosteller. 1994. Efficacy of BCG vaccine in the prevention of tuberculosis: Meta-analysis of the published literature. *Journal of the American Medical Association* 271: 698–702.

Fisher, D. J. 2016. Two-stage individual participant data meta-analysis and generalized forest plots. In *Meta-Analysis in Stata: An Updated Collection from the Stata Journal*, ed. T. M. Palmer and J. A. C. Sterne, 2nd ed., 280–307. College Station, TX: Stata Press.

Harris, R. J., M. J. Bradburn, J. J. Deeks, R. M. Harbord, D. G. Altman, and J. A. C. Sterne. 2016. metan: Fixed- and random-effects meta-analysis. In *Meta-Analysis in Stata: An Updated Collection from the Stata Journal*, ed. T. M. Palmer and J. A. C. Sterne, 2nd ed., 29–54. College Station, TX: Stata Press.

Lewis, J. A., and S. H. Ellis. 1982. A statistical appraisal of post-infarction beta-blocker trials. *Primary Cardiology* Suppl. 1: 31–37.

Lewis, S., and M. Clarke. 2001. Forest plots: Trying to see the wood and the trees. *British Medical Journal* 322: 1479–1480.

Raudenbush, S. W. 1984. Magnitude of teacher expectancy effects on pupil IQ as a function of the credibility of expectancy induction: A synthesis of findings from 18 experiments. *Journal of Educational Psychology* 76: 85–97.

Raudenbush, S. W., and A. S. Bryk. 1985. Empirical Bayes meta-analysis. *Journal of Educational Statistics* 10: 75–98.

Schriger, D. L., D. G. Altman, J. A. Vetter, T. Heafner, and D. Moher. 2010. Forest plots in reports of systematic reviews: A cross-sectional study reviewing current practice. *International Journal of Epidemiology* 39: 421–429.

Also see

[META] **meta data** — Declare meta-analysis data

[META] **meta summarize** — Summarize meta-analysis data

[META] **meta** — Introduction to meta

[META] **Glossary**

[META] **Intro** — Introduction to meta-analysis