Title

stata.com

meta data — Declare meta-analysis data

Description Remarks and examples References Also see

Description

This entry describes how to prepare your data for meta-analysis using the `meta` commands.

In a nutshell, do the following:

1. If you have access to summary data, use `meta esize` to compute and declare effect sizes such as an odds ratio or a Hedges's g.
2. Alternatively, if you have only precomputed (generic) effect sizes, use `meta set`.
3. To update some of your meta-analysis settings after the declaration, use `meta update`.
4. To check whether your data are already meta set or to see the current meta settings, use `meta query`.
5. If you want to perform multivariate meta-regression using `meta mvregress`, you do not need to `meta set` your data.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Overview

The declaration of your data to be `meta` data is the first step of your meta-analysis in Stata. `meta` data are your original data that also store key variables and characteristics about your specifications, which will be used by all `meta` commands during your meta-analysis session. The declaration step helps minimize mistakes and saves you time—you only need to specify the necessary information once.

You can use `meta set` or `meta esize` to declare your data to be `meta` data. If you have access only to precomputed effect sizes and their standard errors, use `meta set`. If you have access to summary data such as means and standard deviations from individual studies, use `meta esize` to compute the effect sizes and their standard errors and declare them. The latter is preferable because it provides access to more features such as the Mantel–Haenszel estimation method with binary data, which needs access to the actual 2×2 tables and not only the effect sizes for the computations.

For example, suppose that you have variables `es` and `se`, which contain the effect sizes and the corresponding standard errors. You can use

```
. meta set es se
```

to declare your data, and all subsequent `meta` commands will automatically use these variables in the meta-analysis analysis.

To review the current meta settings or to check whether the data are `meta` set, you can use `meta query`; see [META] **meta update**. After your data are declared, you can update some of the meta-analysis specifications by using `meta update`. If you wish to clear the meta settings after your meta-analysis analysis, you can use `meta clear`; see [META] **meta update**.

Declaring meta-analysis information

Two main components of meta-analysis are study-specific effect sizes and their precision. You must specify them during declaration. Other important components include the underlying meta-analysis model and an estimation method. You can specify them during declaration or later during analysis or use Stata's defaults. You can also specify options that affect the output of the `meta` commands. Below, we describe how you can declare various meta-analysis information.

Declaring effect sizes and their precision

As we mentioned above, you must declare study-specific effect sizes and their precision. This is done differently for `meta set` and `meta esize`.

`meta esize` computes effect sizes and their standard errors from summary data and then declares them. `meta set` declares already precomputed effect sizes and their standard errors. Thus, to use `meta set`, you do not need summary data from each study, but you need them for `meta esize`. Some analysis may not be available after `meta set` such as the Mantel–Haenszel estimation method and Harbord's test for the funnel-plot asymmetry because they require access to summary data.

Effect sizes and their precision using meta set. To use `meta set`, you must specify variables containing study-specific effect sizes and their precision. There are two ways to specify the precision of the effect sizes. You can either specify a variable containing the standard errors,

```
. meta set es se
```

Or, instead of the standard errors, specify the confidence intervals, and `meta set` will compute the corresponding standard errors based on them:

```
. meta set es cil ciu
```

In the above, the specified CI variables will be assumed to correspond to the 95% CIs. You can change this by specifying the `civarlevel()` option:

```
. meta set es cil ciu, civarlevel(90)
```

But do not confuse `civarlevel()` with `level()`. The former affects the confidence level only for the specified CI variables. The latter specifies the confidence level for the meta-analysis.

Effect sizes and their precision using meta esize. To use `meta esize`, you must specify summary data for each study. The type of summary data you specify depends on the effect size you wish to compute and consequently on the outcome of interest in the original studies.

`meta esize` computes and declares various effect sizes for two-group comparison of continuous outcomes and of binary outcomes. For continuous outcomes, you must specify the number of observations, means, and standard deviations for each treatment group (group 1) and control group (group 2).

```
. meta esize n1 m1 sd1 n2 m2 sd2
```

To compute effect sizes and their standard errors, `meta esize` also needs to know the type of the effect size. The above assumes Hedges's g standardized mean difference, but you can specify others in the `esize()` option; see effect sizes for continuous outcomes in *Syntax* of [META] **meta esize**.

For binary outcomes, you must specify 2×2 contingency tables for each study. You specify them as follows. Each of the four cells is represented by a variable such that each row represents a 2×2 table from a specific study. For instance,

```
. meta esize n11 n12 n21 n22
```

The order in which you specify the four variables is important: the top-left cell first, the top-right cell next, followed by the bottom-left cell, and finally the bottom-right cell. The above computes the log odds-ratio as an effect size, but you can select a different effect size; see effect sizes for binary outcomes in *Syntax* of [META] **meta esize**.

Options affecting effect-size and precision computations with meta esize. Depending on the chosen effect size, `meta esize` provides alternative ways of computing effect sizes and their standard errors.

For the Hedges's g effect size, there are two ways to compute the bias-correction factor used in its formula. For consistency with meta-analysis literature, `meta esize` uses an approximation, but you can specify the `exact` option within `esize()` to use the exact computation:

```
. meta esize n1 m1 sd1 n2 m2 sd2, esize(hedgesg, exact)
```

Note that the `esize` command uses the exact computation.

Both Hedges's g and Cohen's d effect sizes support standard error adjustment of Hedges and Olkin (1985) with `esize()`'s option `holkinse`:

```
. meta esize n1 m1 sd1 n2 m2 sd2, esize(cohend, holkinse)
```

For the (unstandardized) mean difference, you can choose to compute standard errors assuming unequal variance between the two groups:

```
. meta esize n1 m1 sd1 n2 m2 sd2, esize(mdiff, unequal)
```

For binary outcomes with log odds-ratios or log risk-ratios as effect sizes, `meta esize` automatically adjusts for zero cells when computing effect sizes. By default, it adds 0.5 to all cells of the 2×2 tables that contain at least one zero cell. You can specify other adjustments in the `zerocells(`*zcspec*`)` option. For example, with log odds-ratios, you can specify the treatment-arm continuity correction of Sweeting, Sutton, and Lambert (2004) as `zerocells(tacc)`, or you can request no zero-cell adjustment:

```
. meta esize n11 n12 n21 n22, zerocells(none)
```

See *Options* in [META] **meta esize**.

Declaring a meta-analysis model

Before you proceed with performing meta-analysis, we want you to think about the model underlying your meta-analysis. This decision is important because the selected meta-analysis model will determine the availability of some of the meta-analysis methods and, more importantly, how you interpret the obtained results; see *Comparison between the models and interpretation of the results* in [META] **Intro**. Also, most likely, you will want to use the chosen model during your entire meta-analysis session. Thus, we made the choices for the meta-analysis model and, consequently, the meta-analysis estimation method be part of the initial declaration step. But fear not! If desired, you can easily switch to a different meta-analysis model or method for the rest of your meta-analysis session or reset it temporarily for a particular analysis; see *Modifying default meta settings*.

We discuss the available models and the differences between them in detail in *Meta-analysis models* in [META] **Intro**.

Briefly, there are three models to choose from: a common-effect, fixed-effects, or random-effects model. They can be requested by specifying options `common`, `fixed`, or `random`. If you omit all of these options, the random-effects model will be assumed.

A common-effect model makes a strong assumption about the underlying true effect size being the same across (common to) all studies. When this assumption is true, this model is a reasonable choice. Most likely, you will want to verify the plausibility of this assumption for your data. So a model that allows the study effect sizes to be different may be a better choice during the initial analysis.

A fixed-effects model allows the effect sizes to be different across studies and assumes that they are fixed. You may ask: What does "fixed" mean? Different disciplines may have different definitions of a fixed effect. In the context of meta-analysis, you can think of fixed effects as effects of particular interest. In other words, your research questions and final inference are focused only on the specific studies that were selected in the meta-analysis.

Conversely, a random-effects model assumes that the study effect sizes are random, meaning that they represent a random sample from a larger population of similar studies. The results obtained from a random-effects model can be extended to the entire population of similar studies and not just the ones that were selected in the meta-analysis. The meta-analysis literature recommends to start with a random-effects model, which is also Stata's default for most `meta` commands.

So, which model should you choose? Our recommendation is to start with a random-effects model and explore the heterogeneity, publication bias, and other aspects of your meta-analysis data. If you are interested only in the inference about the particular studies in your data, a fixed-effects model may be a reasonable alternative. We suggest that you avoid using, or at least starting with, a common-effect model unless you verified that the underlying assumption of the common study effects is plausible for your data.

As we described in *Comparison between the models and interpretation of their results* in [META] **Intro**, a fixed-effects model and a common-effect model produce the same results in a meta-analysis. Although the final estimates are the same, their interpretation is different! In a common-effect model, the estimate of the overall effect size is an estimate of the true common effect size, whereas in a fixed-effects model, it is an estimate of the average of true, different study-specific effect sizes. Thus, the `meta` suite provides the two options `common` and `fixed` to emphasize the conceptual differences between the two models. Additionally, when you assume a common-effect model, you essentially imply that certain issues such as study heterogeneity are of no concern in your data. Therefore, when you specify the `common` option, certain commands such as `meta regression` will not be available. This is again our way of reminding you of the underlying assumption of a common-effect model. For other `meta` commands, specifying `common` versus `fixed` will merely change the reported title from, say, "Common-effect meta-analysis" to "Fixed-effects meta-analysis". Nevertheless, the title change is important because it encourages proper interpretation of the results.

Declaring a meta-analysis estimation method

Depending on a chosen meta-analysis model and effect size, there are a number of methods available to estimate the overall effect size. For a common-effect model and a fixed-effects model, the inverse-variance method, `common(invvariance)` and `fixed(invvariance)`, is used with generic effect sizes, which are declared by `meta set`, and with effect sizes for continuous data, which are declared by `meta esize`. With effect sizes for binary data (except Peto's log odds-ratio), which are also declared by `meta esize`, the Mantel–Haenszel method, `common(mhaenszel)` or `fixed(mhaenszel)`, is also available.

For a random-effects model, there are several different methods to estimate the between-study variance, which contributes to the weights used to estimate the overall effect size. The default method is REML, `random(reml)`, but other methods such as ML, `random(ml)`, and DerSimonian–Laird, `random(dlaird)`, are also available. See *Syntax* in [META] **meta set** for a full list.

When you specify `random`, the REML method is assumed. When you specify `common` or `fixed`, the inverse-variance method is assumed for all effect sizes except log odds-ratios, log risk-ratios, and risk differences, as specified with `meta esize`. For these effect sizes, the Mantel–Haenszel method is the default method.

See *Meta-analysis estimation methods* in [META] **Intro** for detailed descriptions of the methods.

Default meta-analysis model and method

During declaration, `meta set` and `meta esize` assume a random-effects model unless you specify one of options `fixed` or `common`. It also assumes the REML estimation method unless you specify some other method in option `random()`; see *Declaring a meta-analysis estimation method*.

The declared model will be used by all `meta` commands except `meta funnelplot` and `meta labbeplot`, which, for historical reasons, assume a common-effect model with the inverse-variance estimation method. But you can change the assumed model and method by specifying the corresponding options such as `random(dlaird)` with a `meta` command.

Also see *Modifying default meta settings* for details.

Declaring a confidence level for meta-analysis

By default, `meta set` and `meta esize` assume the 95% confidence level (or as set by `set level`) for the entire meta-analysis analysis. You can change this by specifying the `level()` option with these commands. You can also modify the confidence level after the declaration as we describe in *Modifying default meta settings*.

Declaring display settings for meta-analysis

`meta set` and `meta esize` also provide options to control the output of `meta` commands.

The `studylabel(`*varname*`)` option specifies a string variable that will be used by `meta` commands such as `meta summarize` and `meta forestplot` to label the studies in the output. By default, the generic labels—`Study 1`, `Study 2`, and so on—will be used.

The `eslabel(`*string*`)` option specifies a string that will be used by `meta` commands such as `meta summarize` and `meta forestplot` to label effect sizes in the output. The default label with `meta set` is `Effect size`. The default label with `meta esize` is specific to the chosen effect size. For instance, it is `Log Odds-Ratio` for log odds-ratios.

By default, all meta commands display a short summary about the declared meta settings such as the variables containing effect sizes and their standard errors. After the declaration, the meta commands do not require you to specify the effect-size variables and standard error variables again. They simply use the corresponding system variables (see *System variables*) created during declaration. The reported summary reminds you that those variables are part of your meta-analysis. You can suppress this summary from all meta commands by specifying the nometashow option with meta set or meta esize. You can also suppress this summary for a particular meta command by specifying the option with that command; see *Modifying default meta settings*.

Modifying default meta settings

You can modify the default meta settings both during and after the declaration. Some of the settings may even be modified (temporarily) for a particular meta command.

You can modify the default settings during the declaration by simply specifying the corresponding options with meta set or meta esize. For example, when we type

```
. meta set ...
```

a random-effects model with the REML estimation method is assumed. We can specify another estimation method, for example, ML, by using random(ml):

```
. meta set ... , random(ml)
```

Or we can specify a different meta-analysis model, for example, a fixed-effects model:

```
. meta set ... , fixed
```

After the declaration, you can use meta update to modify the current settings. For example, we can switch to a common-effect model for the rest of our meta-analysis by typing

```
. meta update, common
```

Now all subsequent meta commands will automatically assume a common-effect model.

In the above examples, we used meta set, but you can use the same specifications with meta esize. We also demonstrated only a few options, but the same principles apply to the other options supported by meta set and meta esize.

For options random(), common() (and common), fixed() (and fixed), level(), and nometashow, we can also modify the current setting temporarily while running a particular meta command. For example, suppose that we want to obtain the results assuming a 90% confidence level with meta summarize. We can type

```
. meta summarize, level(90)
```

If we wanted all relevant meta commands to use the 90% confidence level, we would have typed

```
. meta update, level(90)
```

Meta-analysis information

When you use meta set or meta esize, they record information about your study, effect sizes and their precision, and meta-analysis model and meta-analysis estimation method, among other things. This information will be used by subsequent meta commands. The summary information is mostly the same between the two commands, but meta esize records several additional settings.

Let's get familiar with the meta setting information by looking at examples.

Meta settings with meta set

Consider a fictional dataset, `metaset.dta`, containing generic effect sizes and their standard errors stored in the corresponding variables `es` and `se`.

```
. use https://www.stata-press.com/data/r17/metaset
(Generic effect sizes; fictional data)
. describe es se
Variable      Storage   Display    Value
    name         type    format    label      Variable label
------------------------------------------------------------------------------
es              double  %10.0g                Effect sizes
se              double  %10.0g                Std. err. for effect sizes
```

At the minimum, with `meta set`, we must specify the variables containing effect sizes and their standard errors. (For other uses of `meta set`, see *Remarks and examples* in [META] **meta set**.)

```
. meta set es se
Meta-analysis setting information

 Study information
    No. of studies: 10
       Study label: Generic
        Study size: N/A

       Effect size
             Type: <generic>
            Label: Effect size
         Variable: es

        Precision
        Std. err.: se
               CI: [_meta_cil, _meta_ciu]
         CI level: 95%

  Model and method
            Model: Random effects
           Method: REML
```

The summary is divided into four categories: information about the study, the specified effect sizes, their precision, and meta-analysis model and method. Below, we describe in detail each output category.

```
Study information
   No. of studies:  10
      Study label:  Generic
       Study size:  N/A
```

The study information consists of the number of studies (10 in our example), a study label (`Generic`), and a study size (`N/A`). If the `studylabel(`*varname*`)` option is specified, the `Study label:` will contain the name of the specified variable. Otherwise, a generic study label—`Study 1`, `Study 2`, and so on—will be used in the output of `meta` commands. If the `studysize(`*varname*`)` option is specified with `meta set`, the `Study size:` will contain the name of the specified variable.

```
       Effect size
             Type:  Generic
            Label:  Effect size
         Variable:  es
```

The effect-size information consists of the type of the effect size, its label, and the variable containing study-specific effect sizes. The effect-size `Type:` is always `Generic` with `meta set`. The effect-size `Label:` is either a generic `Effect size` or as specified in the `eslabel(`*string*`)` option. This label

will be used to label the effect sizes in the output of all meta commands. The effect-size Variable: displays the name of the declared variable containing effect sizes. After the declaration, both commands store study-specific effect sizes in the system variable _meta_es (see *System variables*). meta set simply copies them from the declared effect-size variable. Thus, Variable: will contain the name of the *esvar* variable, es in our example, with meta set.

```
Precision
Std. err.:  se
       CI:  [_meta_cil, _meta_ciu]
 CI level:  95%
```

The precision information consists of variables containing effect-size standard errors, confidence intervals, and the declared confidence level. As with the effect sizes, meta set uses the standard errors specified in the *sevar* variable (variable se here). The corresponding confidence intervals are computed using the effect sizes and their standard errors and stored in the system variables _meta_cil and _meta_ciu. With meta set, you can specify confidence intervals instead of the standard errors, in which case the standard errors will be computed from the effect sizes and confidence intervals and stored in _meta_se, in which case Std. err.: will contain _meta_se; see *Syntax* in [META] **meta set**. CI: always contains _meta_cil and _meta_ciu. The specified CI variables will be reported in User CI: with their corresponding confidence level reported in User CI level:, which is controlled by the civarlevel() option. The declared CI variables and the system CI variables will be the same only when civarlevel() is the same as level(), and the system variables are the ones that are used in the meta-analysis analysis.

CI level: reports the confidence level, controlled by the level() option, that will be used by all meta commands when computing confidence intervals for various meta-analyses such as the CIs of the overall effect size, regression coefficients, and so on. The default confidence level is 95% or as set by set level.

```
Model and method
          Model:  Random-effects
         Method:  REML
```

As we pointed out in *Declaring a meta-analysis model*, the meta-analysis model and, consequently, the meta-analysis estimation method are important aspects of your meta-analysis. As such, we made them be part of your declaration step too. By default, a random-effects model with the REML estimation method is assumed for most meta commands; see *Default meta-analysis model and method*. You can change the defaults as we describe in *Modifying default meta settings*.

Meta settings with meta esize

Consider metaescnt.dta, containing fictional study-specific summary data for continuous outcomes for group 1 and group 2.

```
. use https://www.stata-press.com/data/r17/metaescnt, clear
(Fictional summary data for continuous outcomes)
. describe n1 m1 sd1 n2 m2 sd2
```

Variable name	Storage type	Display format	Value label	Variable label
n1	byte	%9.0g		Study sizes of group 1
m1	float	%9.0g		Means of group 1
sd1	float	%9.0g		Std. dev. of group 1
n2	byte	%9.0g		Study sizes of group 2
m2	float	%9.0g		Means of group 2
sd2	float	%9.0g		Std. dev. of group 2

With meta esize, we must specify the summary data to compute an effect size. Let's focus on the studies comparing the mean differences between the two groups. Our summary data include the numbers of observations and the estimates of means and standard deviations for each group. We specify the variables containing these summaries following the command name.

```
. meta esize n1 m1 sd1 n2 m2 sd2
Meta-analysis setting information
 Study information
    No. of studies: 10
       Study label: Generic
        Study size: _meta_studysize
      Summary data: n1 m1 sd1 n2 m2 sd2
       Effect size
             Type: hedgesg
            Label: Hedges's g
         Variable: _meta_es
   Bias correction: Approximate
         Precision
         Std. err.: _meta_se
    Std. err. adj.: None
               CI: [_meta_cil, _meta_ciu]
         CI level: 95%
  Model and method
            Model: Random effects
           Method: REML
```

The meta setting information from meta esize is almost the same as the one produced by meta set, which we described in *Meta settings with meta set*, but has several additional settings. The summary-data variables are listed under Summary data:. As we mentioned earlier, meta esize computes the effect sizes and their standard errors from the specified summary data, so effect-size Variable: and Std. err.: contain the names of the corresponding system variables, _meta_es and _meta_se. The summary data also include the information about the study size, so Study size: displays the name of the system variable, _meta_studysize, that contains study size, which is equal to the sum of n1 and n2 in our example.

By default, meta esize computes the Hedges's g effect size for the two-group mean comparison. You can specify the esize(*esspec*) option to select a different effect size. For the Hedges's g effect size, there are two methods to compute the underlying bias-correction term: approximate or exact. For consistency with the meta-analysis literature, meta esize, by default, uses an approximation, as indicated in Bias correction: under Effect size. But you can change this by specifying the exact option within esize().

Another additional setting describes the type of adjustment applied when computing the standard errors of the effect sizes; see `Std. err. adj.:` under `Precision`. This adjustment is applicable only with the Hedges's g or Cohen's d effect size. No adjustment is made by default, but you can use the `holkinse` option within `esize()` to specify the adjustment of Hedges and Olkin (1985). For the mean-difference effect size, you can request the adjustment for unequal group variances by specifying `esize()`'s option `unequal`.

Finally, for log odds-ratios or log risk-ratios, `meta esize` additionally reports the type of adjustment made to the zero cells of contingency tables, which represent the summary data for binary outcomes. For these effect sizes, the type of adjustment will be listed in `Zero-cells adj.:` under `Effect size` (not applicable in our example). By default, 0.5 is added to each zero cell, but you can specify the `zerocells()` option with `meta esize` to apply a different adjustment or none.

System variables

`meta set` and `meta esize` store information about the meta-analysis settings in data characteristics ([P] **char**) and system variables.

`meta` system variables are the variables that begin with `_meta_`. There are four main variables that are stored by the two commands.

`_meta_es` stores study-specific effect sizes.

`_meta_se` stores the standard errors of study-specific effect sizes.

`_meta_cil` and `_meta_ciu` store the lower and upper limits of the CIs for study-specific effect sizes. These variables correspond to the confidence level declared for the meta-analysis, the value of which is stored in the data characteristic `_meta_level`.

Other system variables include integer study identifiers stored in `_meta_id`, study labels stored in a string variable `_meta_studylabel`, and study sizes stored in `_meta_studysize`. `_meta_studysize` is always stored with `meta esize`. With `meta set`, it is stored only when the variable containing study sizes is specified in the `studysize()` option.

Also see *Stored results* in [META] **meta set** and *Stored results* in [META] **meta esize**.

Examples of data declaration for meta-analysis

In this section, we demonstrate how to prepare data for meta-analysis in Stata for several case studies.

Declaring precomputed effect sizes using meta set

We will demonstrate how to use `meta set` to declare generic effect sizes.

▷ Example 1: Precomputed log hazard-ratios using meta set

We demonstrate how to declare the meta-analysis data from Steurer et al. (2006), who studied the effect of purine analogues for the treatment of chronic lymphocytic leukemia. Variables `loghr` and `seloghr` contain the log hazard-ratios and their standard errors.

```
. use https://www.stata-press.com/data/r17/leukemia2, clear
(Single-agent purine analogue treatment for leukemia)
. describe
Contains data from https://www.stata-press.com/data/r17/leukemia2.dta
 Observations:             4                  Single-agent purine analogue
                                                treatment for leukemia
    Variables:             6                  25 Apr 2020 12:09
                                              (_dta has notes)
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
author          str14   %14s                * Author
year            int     %8.0g                 Publication year
ntreat          int     %8.0g                 Treatment-group sample size
ncontrol        int     %8.0g                 Control-group sample size
loghr           float   %9.0g                 Log hazard-ratio
seloghr         float   %9.0g                 Standard error for loghr
                                            * indicated variables have notes
-------------------------------------------------------------------------------
Sorted by:
```

We use the `meta set` command to declare the effect sizes (log hazard-ratios) and their standard errors.

```
. meta set loghr seloghr
Meta-analysis setting information
 Study information
    No. of studies: 4
       Study label: Generic
        Study size: N/A
       Effect size
             Type: <generic>
            Label: Effect size
         Variable: loghr
        Precision
        Std. err.: seloghr
               CI: [_meta_cil, _meta_ciu]
         CI level: 95%
  Model and method
            Model: Random effects
           Method: REML
```

`meta set` reports that there are 4 studies in this dataset. The type of effect size is `Generic` because we used the precalculated effect size. The default label `Effect size` will be used in the output. The command also reports the variables that were used to declare the effect sizes, `loghr`, and their standard errors, `seloghr`. The other settings are as we described in *Meta settings with meta set*.

As we described in *System variables*, `meta set` created several system variables that will be used by other `meta` commands in the computations:

```
. describe _meta*
Variable      Storage   Display    Value
    name         type    format    label      Variable label
------------------------------------------------------------------------------
_meta_id        byte    %9.0g                 Study ID
_meta_studyla~l str7    %9s                   Study label
_meta_es        float   %9.0g                 Generic ES
_meta_se        float   %9.0g                 Std. err. for ES
_meta_cil       double  %10.0g                95% lower CI limit for ES
_meta_ciu       double  %10.0g                95% upper CI limit for ES
. list _meta*
     +-------------------------------------------------------------------+
     | _meta_id   _meta~el   _meta_es   _meta_se    _meta_cil   _meta_ciu |
     |-------------------------------------------------------------------|
  1. |        1    Study 1      -.592       .345   -1.2681876   .08418756 |
  2. |        2    Study 2     -.0791      .0787   -.23334916   .07514916 |
  3. |        3    Study 3      -.237       .144   -.51923481    .0452348 |
  4. |        4    Study 4       .163       .312   -.44850877   .77450878 |
     +-------------------------------------------------------------------+
```

`_meta_id` contains integers identifying the studies, and `_meta_studylabel` contains the study labels. `_meta_es` and `_meta_se` contain log hazard-ratios and their standard errors, and `_meta_cil` and `_meta_ciu` contain the corresponding lower and upper bounds of the 95% CIs for log hazard-ratios.

We did not specify the `studylabel()` option in this example, so generic labels will be used in the output of other `meta` commands such as `meta summarize`:

```
. meta summarize
  Effect-size label: Effect size
        Effect size: loghr
          Std. err.: seloghr
Meta-analysis summary                     Number of studies =      4
Random-effects model                      Heterogeneity:
Method: REML                                          tau2 =  0.0000
                                                   I2 (%) =    0.00
                                                       H2 =    1.00

------------------------------------------------------------------------------
       Study |  Effect size    [95% conf. interval]  % weight
-------------+----------------------------------------------------------------
     Study 1 |      -0.592      -1.268       0.084      3.68
     Study 2 |      -0.079      -0.233       0.075     70.70
     Study 3 |      -0.237      -0.519       0.045     21.12
     Study 4 |       0.163      -0.449       0.775      4.50
-------------+----------------------------------------------------------------
       theta |      -0.120      -0.250       0.009
------------------------------------------------------------------------------
Test of theta = 0: z = -1.82                     Prob > |z| = 0.0688
Test of homogeneity: Q = chi2(3) = 3.62            Prob > Q = 0.3049
```

Generic labels `Study 1`, `Study 2`, `Study 3`, and `Study 4` are used to label the studies. Also, the generic label `Effect size` is used to label the log hazard-ratios. See [META] **meta summarize** for details about `meta summarize`.

We can provide more descriptive labels for the studies and the effect sizes by specifying options `studylabel()` and `eslabel()`.

```
. generate studylbl = author + " (" + string(year) + ")"
. meta set loghr seloghr, studylabel(studylbl) eslabel("Ln(HR)")
Meta-analysis setting information
 Study information
    No. of studies: 4
       Study label: studylbl
        Study size: N/A
       Effect size
              Type: <generic>
             Label: Ln(HR)
          Variable: loghr
         Precision
         Std. err.: seloghr
                CI: [_meta_cil, _meta_ciu]
          CI level: 95%
  Model and method
             Model: Random effects
            Method: REML
```

We created a new variable, `studylbl`, that combines the author and year information of the published studies to use as our study labels. `meta set` reported that `studylbl` will be used to label the studies and `Ln(HR)` to label the effect sizes.

If we now rerun `meta summarize` (suppressing the table header), we see the new labels in the output.

```
. meta summarize, noheader
  Effect-size label: Ln(HR)
        Effect size: loghr
          Std. err.: seloghr
        Study label: studylbl
```

Study	Ln(HR)	[95% conf.	interval]	% weight
Johnson et al. (1996)	-0.592	-1.268	0.084	3.68
Leporrier (2001)	-0.079	-0.233	0.075	70.70
Rai (2000)	-0.237	-0.519	0.045	21.12
Robak (2000)	0.163	-0.449	0.775	4.50
theta	-0.120	-0.250	0.009	

```
Test of theta = 0: z = -1.82                     Prob > |z| = 0.0688
Test of homogeneity: Q = chi2(3) = 3.62            Prob > Q = 0.3049
```

After the original declaration, we can use `meta update` to update the meta settings instead of repeating `meta set`; see example 4.

◁

Also see *Remarks and examples* in [META] **meta set** for more examples of using `meta set`.

Computing and declaring effect sizes using meta esize

We demonstrate how to use `meta esize` to compute and declare effect sizes for continuous and binary outcomes.

Example 2: Mean differences for continuous data using meta esize

Consider the study of Gibson et al. (2002), who compared the performance of asthma-management programs for adults with asthma.

The asthma dataset contains the following summary-data variables:

```
use https://www.stata-press.com/data/r17/asthma, clear
(Education and medical review for asthma patients)
. describe ni meani sdi nc meanc sdc
```

Variable name	Storage type	Display format	Value label	Variable label
ni	int	%9.0g		Intervention-group sample size
meani	double	%9.0g		Average days off work/school for intervention group
sdi	double	%9.0g		Std. dev. of days off work/school for intervention group
nc	int	%9.0g		Control-group sample size
meanc	double	%9.0g		Average days off work/school for control group
sdc	double	%9.0g		Std. dev. of days off work/school for control group

Variables ni, meani, and sdi record the study-specific sample sizes, mean numbers of days off work/school, and standard deviations in the intervention group, and variables nc, meanc, and sdc record those items in the control group.

To illustrate, we will compute and declare a couple of effect sizes using meta esize. We will start with the default effect size—Hedges's g standardized mean. We use meta esize to compute this effect size for each study from the summary variables and declare them for further meta-analysis.

```
. meta esize ni meani sdi nc meanc sdc
(2 missing values generated)
Meta-analysis setting information
 Study information
    No. of studies: 13
       Study label: Generic
        Study size: _meta_studysize
      Summary data: ni meani sdi nc meanc sdc
       Effect size
             Type: hedgesg
            Label: Hedges's g
         Variable: _meta_es
   Bias correction: Approximate
         Precision
         Std. err.: _meta_se
    Std. err. adj.: None
               CI: [_meta_cil, _meta_ciu]
         CI level: 95%
  Model and method
            Model: Random effects
           Method: REML
```

There are missing values in the summary variables, so some of the generated system variables will also contain missing values as reported by the note.

meta esize reports that the computed effect size is Hedges's g. See *Meta settings with meta esize* for the explanation of other settings.

The above command is equivalent to

```
. meta esize ni meani sdi nc meanc sdc, esize(hedgesg)
  (output omitted)
```

With this effect size, we can specify that the adjustment of Hedges and Olkin (1985) be applied to the standard errors.

```
. meta esize ni meani sdi nc meanc sdc, esize(hedgesg, holkinse)
(2 missing values generated)

Meta-analysis setting information

 Study information
    No. of studies: 13
       Study label: Generic
        Study size: _meta_studysize
      Summary data: ni meani sdi nc meanc sdc

       Effect size
             Type: hedgesg
            Label: Hedges's g
         Variable: _meta_es
   Bias correction: Approximate

        Precision
        Std. err.: _meta_se
    Std. err. adj.: Hedges-Olkin
               CI: [_meta_cil, _meta_ciu]
         CI level: 95%

  Model and method
            Model: Random effects
           Method: REML
```

`meta esize` updates the adjustment in `Std. err. adj.:` under `Precision` to `Hedges-Olkin`.

Because all studies measured our outcome of interest on the same scale (number of days off work or school), we may consider the raw (unstandardized) mean difference as our effect size. We can compute it by specifying the `esize(mdiff)` option.

```
. meta esize ni meani sdi nc meanc sdc, esize(mdiff)
(2 missing values generated)

Meta-analysis setting information

 Study information
    No. of studies: 13
       Study label: Generic
        Study size: _meta_studysize
      Summary data: ni meani sdi nc meanc sdc

       Effect size
             Type: mdiff
            Label: Mean diff.
         Variable: _meta_es

        Precision
        Std. err.: _meta_se
    Std. err. adj.: None
               CI: [_meta_cil, _meta_ciu]
         CI level: 95%

  Model and method
            Model: Random effects
           Method: REML
```

The information about the type of the effect size and its label is updated to correspond to the mean differences.

As with `meta set`, we could have used `meta update` to update the meta settings after the initial declaration instead of using `meta esize`; see example 4.

◁

▷ Example 3: Log odds-ratios and log risk-ratios for binary data

Let's revisit the declaration we used in example 1 in [META] **meta** for the `bcg` dataset from the BCG vaccine study (Colditz et al. 1994). The summary data (contingency tables) are recorded in the following variables:

```
use https://www.stata-press.com/data/r17/bcg, clear
(Efficacy of BCG vaccine against tuberculosis)
. describe npost nnegt nposc nnegc
Variable      Storage   Display    Value
    name         type    format    label      Variable label
------------------------------------------------------------------------------
npost           int     %9.0g                 Number of TB positive cases in
                                                treated group
nnegt           long    %9.0g                 Number of TB negative cases in
                                                treated group
nposc           int     %9.0g                 Number of TB positive cases in
                                                control group
nnegc           long    %9.0g                 Number of TB negative cases in
                                                control group
```

The summary variables represent the cells of the 2×2 tables for each study.

As with continuous data, we specify the summary variables for binary data following `meta esize`:

```
. meta esize npost nnegt nposc nnegc
Meta-analysis setting information
 Study information
    No. of studies: 13
       Study label: Generic
        Study size: _meta_studysize
      Summary data: npost nnegt nposc nnegc
       Effect size
             Type: lnoratio
            Label: Log odds-ratio
         Variable: _meta_es
  Zero-cells adj.: None; no zero cells
        Precision
        Std. err.: _meta_se
               CI: [_meta_cil, _meta_ciu]
         CI level: 95%
  Model and method
            Model: Random effects
           Method: REML
```

The computed default effect sizes are log odds-ratios, whereas the effect of interest in this study is the risk ratio or, equivalently, the log risk-ratio.

To compute log risk-ratios, we specify `esize(lnrratio)`. We also specify the variable `studylbl` containing study labels in the `studylabel()` option.

```
. meta esize npost nnegt nposc nnegc, esize(lnrratio) studylabel(studylbl)
Meta-analysis setting information
 Study information
    No. of studies: 13
       Study label: studylbl
        Study size: _meta_studysize
      Summary data: npost nnegt nposc nnegc
       Effect size
             Type: lnrratio
            Label: Log risk-ratio
         Variable: _meta_es
   Zero-cells adj.: None; no zero cells
         Precision
         Std. err.: _meta_se
                CI: [_meta_cil, _meta_ciu]
          CI level: 95%
  Model and method
             Model: Random effects
            Method: REML
```

Notice that there are no zero cells in our data, so there is no zero-cells adjustment (see `Zero-cells adj.:` under `Effect size`).

◁

Also see example 4 for how to update the above `meta` settings without having to respecify the summary variables.

Displaying and updating meta settings

We show examples of how to display the current `meta` settings by using `meta query` and update them by using `meta update`.

▷ Example 4: Commands meta query and meta update

Recall example 3. Let's reload the dataset and use `meta query` to check whether the dataset is `meta` set.

```
. use https://www.stata-press.com/data/r17/bcg, clear
(Efficacy of BCG vaccine against tuberculosis)
. meta query
(data not meta set; use meta set or meta esize to declare as meta data)
```

The data are not `meta` set.

Let's again use `meta esize` to declare the data (quietly) and use `meta query` to display the current settings.

```
. quietly meta esize npost nnegt nposc nnegc
. meta query
-> meta esize npost nnegt nposc nnegc
Meta-analysis setting information from meta esize
 Study information
    No. of studies: 13
       Study label: Generic
        Study size: _meta_studysize
      Summary data: npost nnegt nposc nnegc
       Effect size
             Type: lnoratio
            Label: Log odds-ratio
         Variable: _meta_es
   Zero-cells adj.: None; no zero cells
         Precision
         Std. err.: _meta_se
                CI: [_meta_cil, _meta_ciu]
          CI level: 95%
  Model and method
              Model: Random effects
             Method: REML
```

In example 3, we redeclared the data to use the log risk-ratios as effect sizes. After the initial declaration, it is more convenient to use `meta update` to update the meta settings because we do not need to respecify the summary variables with `meta update`.

```
. meta update, esize(lnrratio) studylabel(studylbl)
-> meta esize npost nnegt nposc nnegc , esize(lnrratio) studylabel(studylbl)
Meta-analysis setting information from meta esize
 Study information
    No. of studies: 13
       Study label: studylbl
        Study size: _meta_studysize
      Summary data: npost nnegt nposc nnegc
       Effect size
             Type: lnrratio
            Label: Log risk-ratio
         Variable: _meta_es
   Zero-cells adj.: None; no zero cells
         Precision
         Std. err.: _meta_se
                CI: [_meta_cil, _meta_ciu]
          CI level: 95%
  Model and method
              Model: Random effects
             Method: REML
```

If your summary variables change, however, you must specify them with `meta esize`.

You can use `meta update` after either `meta esize` or `meta set`. `meta update` will respect the options of `meta esize` and `meta set`.

For example, recall the `meta set` declaration from example 1:

```
. use https://www.stata-press.com/data/r17/leukemia2, clear
(Single-agent purine analogue treatment for leukemia)
. quietly meta set loghr seloghr
```

Let's update the meta settings to include the variable containing study sizes.

```
. generate ssize = ntreat + ncontrol
. meta update, studysize(ssize)
-> meta set loghr seloghr , random(reml) studysize(ssize)
Meta-analysis setting information from meta set
 Study information
    No. of studies: 4
       Study label: Generic
        Study size: ssize
       Effect size
             Type: <generic>
            Label: Effect size
         Variable: loghr
        Precision
        Std. err.: seloghr
               CI: [_meta_cil, _meta_ciu]
         CI level: 95%
  Model and method
            Model: Random effects
           Method: REML
```

The `studysize()` option is supported only with `meta set`. If we tried to specify this option with `meta update` after `meta esize`, we would have received an error message.

◁

References

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. https://doi.org/10.1001/jama.1994.03510330076038.

Gibson, P., H. Powell, A. Wilson, M. J. Abramson, P. Haywood, A. Bauman, M. J. Hensley, E. H. Walters, and J. J. L. Roberts. 2002. Self-management education and regular practitioner review for adults with asthma. *Cochrane Database of Systematic Reviews* 3. https://www.cochranelibrary.com/cdsr/doi/10.1002/14651858.CD001117/full.

Hedges, L. V., and I. Olkin. 1985. *Statistical Methods for Meta-Analysis*. Orlando, FL: Academic Press.

Steurer, M., G. Pall, S. Richards, G. Schwarzer, J. Bohlius, and R. Greil. 2006. Single-agent purine analogues for the treatment of chronic lymphocytic leukaemia: A systematic review and meta-analysis. *Cancer Treatment Reviews* 32: 377–389. https://doi.org/10.1016/j.ctrv.2006.05.002.

Sweeting, M. J., A. J. Sutton, and P. C. Lambert. 2004. What to add to nothing? Use and avoidance of continuity corrections in meta-analysis of sparse data. *Statistics in Medicine* 23: 1351–1375. https://doi.org/10.1002/sim.1761.

Also see

[META] **meta esize** — Compute effect sizes and declare meta-analysis data

[META] **meta set** — Declare meta-analysis data using generic effect sizes

[META] **meta update** — Update, describe, and clear meta-analysis settings

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

[META] **Glossary**

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