Meta-analysis of test accuracy studies in Stata

A bivariate model approach

Yemisi Takwoingi

Version 2.0
August 2021

Please cite this version as: Takwoingi Y. Meta-analysis of test accuracy studies in Stata: a bivariate model approach. Version 2.0. August 2021. Available from: http://methods.cochrane.org/sdt/.

Contents

1 Introduction

Hierarchical or mixed models are recommended for meta-analysis of test accuracy studies (Leeflang et al. 2008;Macaskill et al. 2010). The aim of this practical tutorial is to guide both novice and experienced Stata users on how to perform meta-analysis of test accuracy studies by fitting the bivariate model (Chu and Cole 2006;Reitsma et al. 2005) using either the user written program **`metandi`** (Harbord and Whiting 2009;Harbord 2008) or the built in command **`xtmelogit`** or **`meqrlogit`**.

The mixed models estimation routine **`xtmelogit`** was introduced in Stata 10 and replaced by **`meqrlogit`** in Stata 13. Both commands have the same syntax and so can be used interchangeably without the need to modify the code presented in this tutorial beyond simply replacing occurrences of **`meqrlogit`** with **`xtmelogit`**. Prior to version 10, such modelling was possible with the user-written program **`gllamm`** (Generalized Linear Latent And Mixed Models) (Rabe-Hesketh et al. 2004). The **`gllamm`** manual is available for free download at http://www.bepress.com/ucbbiostat/paper160/. The code for fitting the bivariate model using **`gllamm`** is available in the appendix.

The example dataset used in this tutorial, schuetz.csv, is based on a published diagnostic test accuracy review (Schuetz et al. 2010). Schuetz and colleagues evaluated the diagnostic performance of multislice computed tomography (CT) and magnetic resonance imaging (MRI) for the diagnosis of coronary artery disease (CAD). Prospective studies that evaluated either CT or MRI (or both); used conventional coronary angiography (CAG) as the reference standard; and used the same threshold for clinically significant coronary artery stenosis (a diameter reduction of 50% or greater) were included in the review. A total of 103 studies provided a 2x2 table for one or both tests and were included in the meta-analysis: 84 studies evaluated only CT, 14 evaluated only MRI, and 5 studies evaluated both CT and MRI.

A do-file, "*Meta-analysis of test accuracy studies in Stata v2.0.do*", accompanies this tutorial. You can either run the commands from the file or you can create your own do-file as you step through the tutorial.

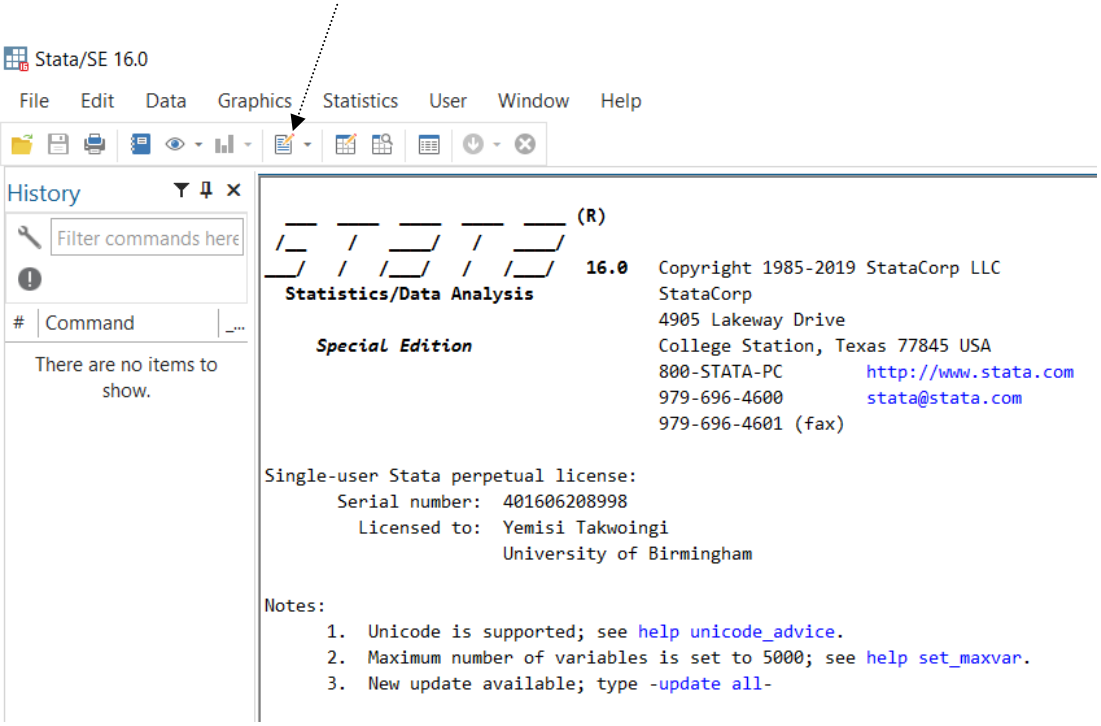
2 Getting started

If you are familiar with Stata you can skip this section.

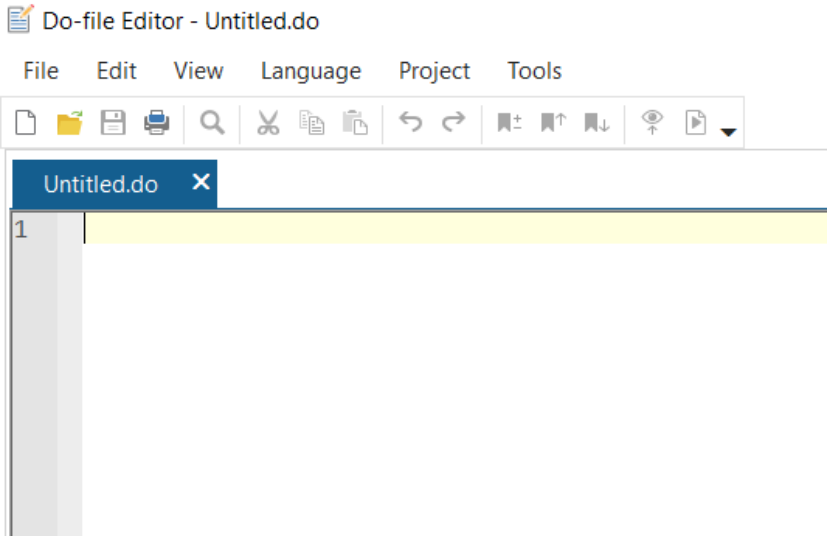
Although it is possible to use Stata interactively (i.e., you type the command in the command window, Stata performs it when you press enter, and any result produced is displayed in the results window, you enter another command, etc.), it is better to write Stata do-files. A do-file is a plain text file containing Stata commands and is created using an editor or word processor. The advantage of writing a do-file is that you do not have to type the same commands again and again before you get the correct sequence of commands. You can also keep a record of what you are doing and be able to reproduce it later.

To create a new do-file or to open an existing one do the following:

Click the *Do-file Editor* button to open the *Do-file Editor*.



You can save the do-file either via the *File* menu or by clicking the disk icon on the toolbar of the *Do-file Editor.*



Type the commands that you wish to submit to Stata in the Editor. You can add comments to the do-file to remind you later what's in the file and what each section or command is trying to accomplish. To add a comment begin with * or to enclose a block of text begin with /* and close with */.

To open the do-file "*Meta-analysis of test accuracy studies in Stata.do*", use the folder icon to browse to the location of the file.

NOTE: Stata is **case sensitive** so if you are not familiar with it beware when you create variables and type commands and program names. Commands are expected to be lowercase. Also be careful with

"=" and "==". For example, after the if command, Stata expects "==" for a test of equality; "=" produces an error in this case.

3 Reading data from a file

3.1 Set working directory

Set your working directory to the appropriate drive where you saved the file schuetz.csv.

Type the following in your do-file replacing "`U:\Handbook 2020`" with your own path:

```
cd "U:\Handbook 2020"
```

3.2 Read data into Stata

To read the comma delimited (Excel .csv) file containing the data you need to use the **`insheet`** command.

```
insheet using "schuetz.csv", comma clear
```

In Stata options for a command are specified after the comma.

The option `comma` above specifies the format of the file to read into Stata (.csv) and `clear` tells Stata that it is ok to replace data that is in memory. To ensure that you do not unintentionally lose data, **`insheet`** will not read new data if there is already data in memory.

To run the do-file highlight the lines you wish to run if not the whole file and then click the *Do Selected Lines* icon (last one on the toolbar). If you click the *Run Selected Lines* instead results will not be displayed in the results window.

Return to the Stata window to view results.

3.3 View the data

Click on the *Data Editor* (next to the *Do-file Editor* on the tool bar) to view the data you just read into Stata.

Alternatively type **`edit`** in the command window and press *Enter*.

If you are using version 10 or earlier, remember to close the *Data Editor* every time you wish to return to the Stata window or the *Do-file Editor.* If you fail to do this, you won't be able to type or execute a command.

To produce a summary of the dataset in memory, type and run

```
describe
```

The following will be shown in the output window.

```
. describe

Contains data
  obs:           108
 vars:             7
```

variable name	storage type	display format	value label	variable label
test	str3	%9s		**Test**
study_id	str18	%18s		**Study_ID**
tp	int	%8.0g		
fp	byte	%8.0g		
fn	byte	%8.0g		
tn	int	%8.0g		
indirect	byte	%8.0g		**Indirect**

A total of 103 studies provided a 2x2 table for one or both tests. Because five studies evaluated both CT and MRI, the total number of observations in the dataset is 108. The five studies can be identified using the variable indirect which is coded as 0 for comparative studies and as 1 for studies that only assessed CT or MRI.

4 Converting strings to numbers

The variable *test* in the dataset is a string variable. Use the command **encode** to generate a new numeric variable called *testtype*.

```
encode test, gen(testtype)
```

List the numeric value assigned to each test

```
label list testtype
```

The following will be shown in the result window

```
. label list testtype
testtype:
           1 CT
           2 MRI
```

From the above encoding, 1 represents CT and 2 represents MRI.

5. Meta-analysis with metandi

metandi performs bivariate meta-analysis of sensitivity and specificity using a generalized linear mixed model approach (Chu & Cole 2006). **metandi** requires 4 input variables: the number of true positives (tp), false positives (fp), false negatives (fn) and true negatives (tn) within each study.

metandi does not have an option that allows for inclusion of a covariate in the bivariate model (i.e. does not support meta-regression), and so **metandi** cannot be used to formally investigate heterogeneity or to compare the accuracy of two or more tests.

In Stata 10 and above, **metandi** fits the model using the command **xtmelogit** by default. In Stata 8 or 9 it uses **gllamm**. Both **gllamm** and **metandi** may not be installed on your machine or may not be up to date. If you are connected to the internet you can install the programs by running the following:

```
ssc install gllamm, replace

ssc install metandi, replace
```

Use **metandi** to meta-analyse studies that evaluated CT by using the **if** statement to restrict the data to only studies where the variable *testtype* is equal to 1.

```
metandi tp fp fn tn if testtype==1
```

NOTE: **metandi** fits **ONLY** the bivariate model. Stata does not have a command for fitting non-linear generalised mixed models and so it is not possible to fit the hierarchical summary ROC (HSROC) model (Rutter & Gatsonis 2001) in Stata. However, because of the close relationship between the HSROC model and the bivariate model, parameters for one model can be obtained from the other (Harbord et al. 2007). **metandi** uses the relationship between the models to output HSROC model parameters by using a function of the parameter estimates from the bivariate model. Summary test accuracy measures are also produced as shown below.

```
. metandi tp fp fn tn if testtype==1

Refining starting values:

Iteration 0:   log likelihood = -395.89917
Iteration 1:   log likelihood = -386.57627
Iteration 2:   log likelihood = -385.31091
Iteration 3:   log likelihood = -385.29942

Performing gradient-based optimization:

Iteration 0:   log likelihood = -385.29942
Iteration 1:   log likelihood = -385.29864
Iteration 2:   log likelihood = -385.29864

Meta-analysis of diagnostic accuracy

Log likelihood   = -385.29864                     Number of studies =       89
------------------------------------------------------------------------------
             |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
Bivariate    |
  E(logitSe) |   3.556827   .1742296                      3.215343    3.898311
  E(logitSp) |   1.932284   .1234624                      1.690302    2.174266
Var(logitSe) |   1.125341   .3199717                      .6445545    1.964758
Var(logitSp) |   .9009872   .1964317                      .5876802    1.381326
Corr(logits) |   .3183132   .1585522                     -.0160232    .5886271
-------------+----------------------------------------------------------------
HSROC        |
      Lambda |   5.407243   .2350024                      4.946647    5.867839
       Theta |   .6608847   .2175036                      .2345855    1.087184
        beta |  -.1111754   .1753129    -0.63   0.526    -.4547823    .2324315
     s2alpha |   2.654912   .6211695                      1.678396    4.199578
     s2theta |   .3432071   .0935866                      .2011169     .585685
-------------+----------------------------------------------------------------
Summary pt.  |
          Se |   .9722621   .0046987                      .9614076    .9801268
          Sp |    .873502   .0136421                      .8442639    .8979147
         DOR |    242.042   55.76057                      154.0972    380.1778
         LR+ |    7.68599   .8361463                      6.210106    9.512631
         LR- |   .0317548   .0054871                      .0226321    .0445547
       1/LR- |   31.49133   5.441572                      22.44434    44.18502
------------------------------------------------------------------------------
Covariance between estimates of E(logitSe) & E(logitSp)    .003737
```

Stata provides on-line help. For a menu of choices, type **`help`** in the command window and press Enter. You can obtain help on any command in Stata by typing **`help`** followed by the command's name. For example, to learn about **`metandi`** and to discover more options run

```
help metandi
```

5.1 Using metandi with RevMan

For those authoring a diagnostic test accuracy review in RevMan (Review Manager 5.3, 2014), the parameter estimates for the bivariate model can be copied from the Stata output and pasted into the relevant boxes in the"*Externally Calculated Parameters*" window to produce a SROC plot. See screenshots below.

```
. metandi tp fp fn tn if testtype==1

Refining starting values:

Iteration 0:   log likelihood = -395.89917
Iteration 1:   log likelihood = -386.57627
Iteration 2:   log likelihood = -385.31091
Iteration 3:   log likelihood = -385.29942

Performing gradient-based optimization:

Iteration 0:   log likelihood = -385.29942
Iteration 1:   log likelihood = -385.29864
Iteration 2:   log likelihood = -385.29864

Meta-analysis of diagnostic accuracy

Log likelihood   = -385.29864                     Number of studies =       89
```

	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
Bivariate						
E(logitSe)	3.556827	.1742296			3.215343	3.898311
E(logitSp)	1.932284	.1234624			1.690302	2.174266
Var(logitSe)	1.125341	.3199717			.6445545	1.964758
Var(logitSp)	.9009872	.1964317			.5876802	1.381326
Corr(logits)	.3183132	.1585522			-.0160232	.5886271
HSROC						
Lambda	5.407243	.2350024			4.946647	5.867839
Theta	.6608847	.2175036			.2345855	1.087184
beta	-.1111754	.1753129	-0.63	0.526	-.4547823	.2324315
s2alpha	2.654912	.6211695			1.678396	4.199578
s2theta	.3432071	.0935866			.2011169	.585685
Summary pt.						
Se	.9722621	.0046987			.9614076	.9801268
Sp	.873502	.0136421			.8442639	.8979147
DOR	242.042	55.76057			154.0972	380.1778
LR+	7.68599	.8361463			6.210106	9.512631
LR-	.0317548	.0054871			.0226321	.0445547
1/LR-	31.49133	5.441572			22.44434	44.18502

Covariance between estimates of E(logitSe) & E(logitSp) .003737

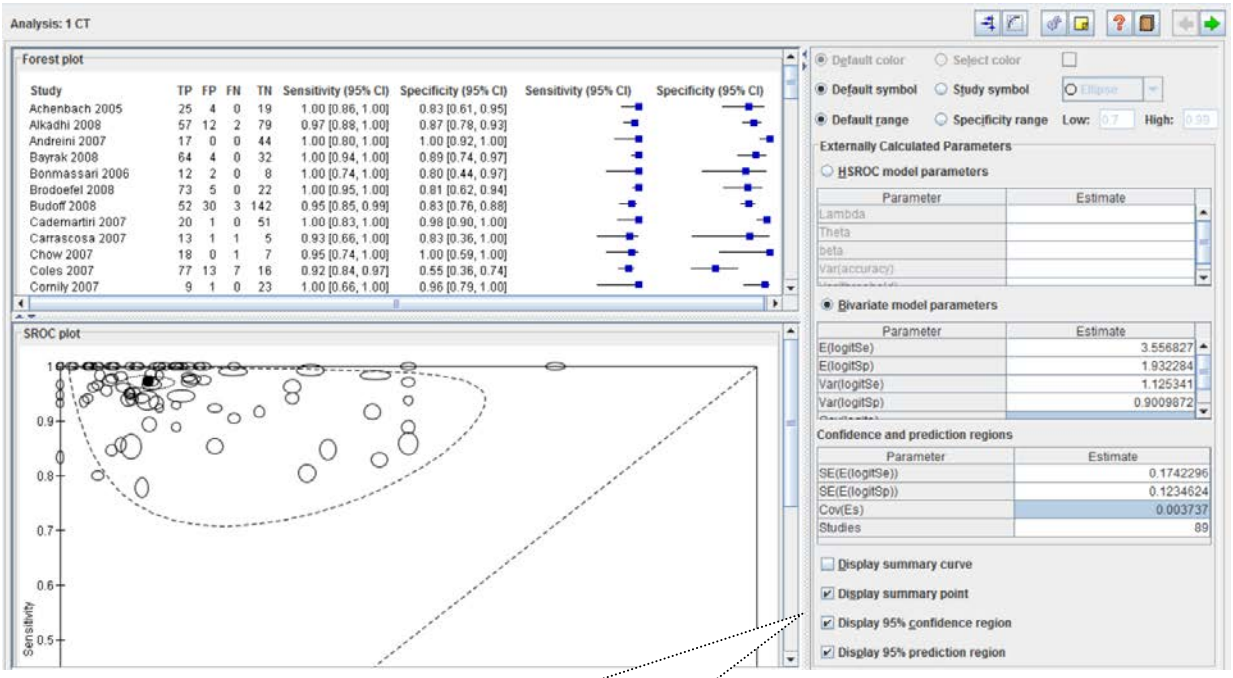
Bivariate model parameters

Parameter	Estimate
E(logitSp)	1.932284
Var(logitSe)	1.125341
Var(logitSp)	0.9009872
Cov(logits)	
Corr(logits)	0.3183132

Confidence and prediction regions

Parameter	Estimate
SE(E(logitSe))	0.1742296
SE(E(logitSp))	0.1234624
Cov(Es)	0.003737
Studies	89

You need either the correlation or the covariance between the variances of the random effects for logit sensitivity and logit specificity. RevMan will disable one of the textboxes when one of them has been filled in. **`metandi`** outputs the correlation of the logits (0.3183132 above) so scroll down to use *Corr(logits)* instead of using *Cov(logits).*



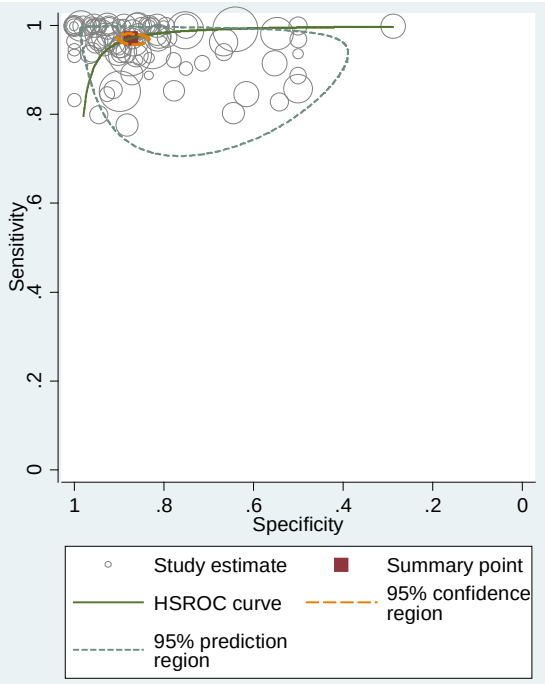
Check the display boxes to display summary points along with confidence and/or prediction regions.

5.2 Producing summary ROC plots with metandi

If you need to produce SROC plots outside RevMan, you can obtain a SROC plot as well as parameter estimates by adding the `plot` option to the **`metandi`** statement as follows

```
metandi tp fp fn tn if testtype==1, plot
```

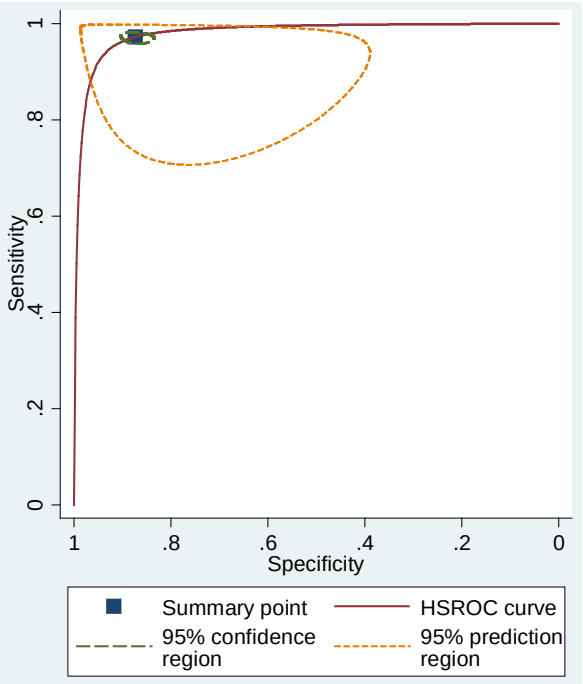
You may need to wait a few seconds for the graph to appear in the graphics editor.



There is no option with **`metandi`** to modify the plot but this can be done using **`metandiplot`**.

Run the following

```
metandiplot if testtype==1
```



A SROC plot without the study specific estimates of sensitivity and specificity will be produced as shown above.

If the optional variables *tp fp fn tn* are included in the command line, estimates of sensitivity and specificity from each study will also be shown on the plot. Try the following

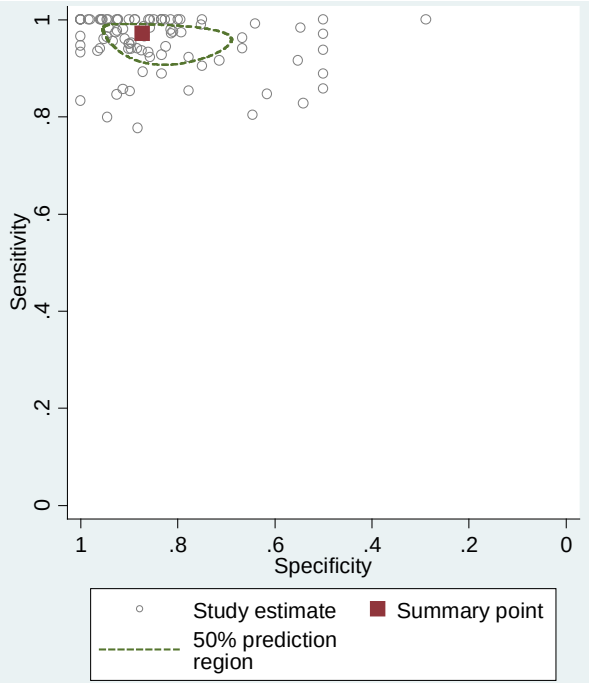
```
metandiplot tp fp fn tn if testtype==1
```

The default is to scale the plot symbol by the sample size of each study. To make the symbols all the same size, specify constant weights, e.g. `[aw=1]`. Try some other options too.

If a command line in the do-file is too long you can spread the command over two or more lines by using `///` to comment out a carriage return. Note there **MUST** be a space before the first of the 3 backslashes. For example,

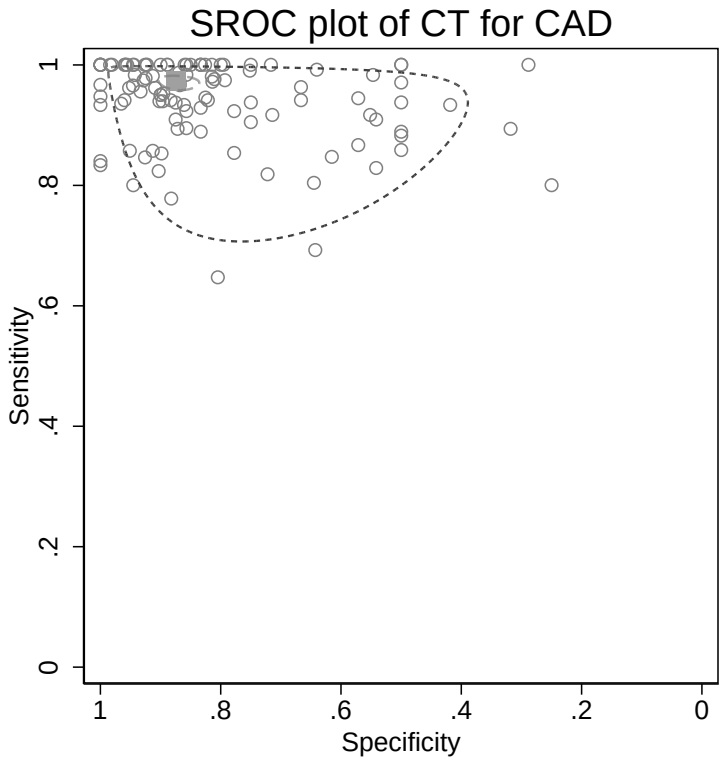
```
metandiplot tp fp fn tn if testtype==1 [aw=1], conf(off) curve(off)
predlevel(50)
```

The command above will produce a plot with constant weights for the symbol, remove the confidence region and SROC curve as well as draw a 50% prediction region on the plot. See plot below.



Here's another example including some twoway graph options.

```
metandiplot tp fp fn tn  [aw=1], curve(off) legend(off)
title(SROC plot of CT for CAD) scheme(s1mono)
```



Use **`help`** to find out more about **`metandiplot`**.

6 Meta-analysis with meqrlogit

As mentioned earlier **metandi** does not have an option for including a covariate in the model and you are also limited in what you can do when the model runs into problems. Therefore it is useful to know how to fit the model using the command **xtmelogit** or **meqrlogit** (replaced **xtmelogit** in Stata 13) directly—essentially doing what **metandi** does. As such the code described in this section is based entirely on Roger Harbord's **metandi** code (Harbord 2008).

6.1 Setting up the data

The data is currently in wide form with one record per study. The data needs to be reshaped into long form to give two records per study—one for the diseased group and one for the non-diseased.

Generate 5 new variables of type **long**. We need these before we can reshape the data.

- `n1` is number diseased
- `n0` is number without disease
- `true1` is number of true positives
- `true0` is the number of true negatives
- `recordid` is the unique identifier for each study (and test if a study in the dataset evaluated more than one test). `_n` will generate a sequence of numbers.

Type the following:

```
gen long n1=tp+fn
gen long n0=fp+tn
gen long true1=tp
gen long true0=tn
gen long recordid= _n
```

Convert data from wide form to long form

```
reshape long n true, i(recordid) j(sens)
```

Let's examine the **reshape** command.

- `long` tells **reshape** that we want to go from wide to long form
- `n` and `true` are the variables (with suffixes 0 and 1) to be converted from wide to long
- `i(recordid)` tells **reshape** that `recordid` uniquely identifies observations in the wide form
- `j(sens)` tells **reshape** that the suffix of `n` and `true` (0 and 1) should be used in creating the binary variable *sens* See the note in the output below.

Next sort the data by *study_id* and *sens*. Generate a new binary variable *spec* of type `byte` that takes the value 0 when *sens*=1 and vice versa.

```
sort study_id sens
gen byte spec=1-sens
```

Run all the lines above. The results are shown below

```
. gen long n1=tp+fn

. gen long n0=fp+tn

. gen long true1=tp

. gen long true0=tn

. gen long recordid= _n

.
. *** Reshape the data from wide to long format ***
. reshape long n true, i(recordid) j(sens)
(note: j = 0 1)

Data                               wide   ->   long
-----------------------------------------------------------------------------
Number of obs.                      108   ->      216
Number of variables                  12   ->       11
j variable (2 values)                     ->   sens
xij variables:
                                   n0 n1   ->   n
                              true0 true1   ->   true
-----------------------------------------------------------------------------
```

Look at the data after reshaping. You can run the command **`list`** or **`edit`** in the command window. Each study now has 2 records—when *sens* = 0, *spec* =1 and *true* = tn, and when *spec* = 0, *sens* = 1 and *true* = tp. Rows 31 to 38 (3 studies, one of which compared CT and MRI) of the data editor are shown below

	recordid	sens	test	study_id	tp	fp	fn	tn	indirect	n	true	spec
31	16	1	CT	Davin 2007	42	4	12	30	1	54	42	0
32	16	0	CT	Davin 2007	42	4	12	30	1	34	30	1
33	17	0	CT	Deetjen 2007	31	3	2	26	1	29	26	1
34	17	1	CT	Deetjen 2007	31	3	2	26	1	33	31	0
35	18	0	CT	Dewey 2006	62	5	4	46	0	51	46	1
36	18	1	CT	Dewey 2006	62	5	4	46	0	66	62	0
37	19	0	MRI	Dewey 2006	42	2	7	39	0	41	39	1
38	19	1	MRI	Dewey 2006	42	2	7	39	0	49	42	0

6.2 Modelling with meqrlogit

The data is now set up for running **`meqrlogit`**. Run the following:

```
meqrlogit true sens spec if testtype==1, nocons|| study_id: sens spec, ///
nocons cov(un) binomial(n) refineopts(iterate(3)) intpoints(5) variance
```

Let's examine the model specification and output:

- The variable `true` specifies the response while `sens` and `spec` are the fixed portions of the model similar to if we were using **`regress`** or some other Stata estimation command. Our fixed effects are coefficients on `sens` and `spec` without a constant term (`nocons`)
- With `|| study_id:` the random effects were specified at the level identified by the group variable `study_id.`
- `intpoints(5)` – the number of integration points for adaptive Gaussian quadrature

- `cov()` – covariance specifies the structure of the covariance matrix for the random effects. `cov(un)` specifies unstructured covariance allowing all variances and covariances to be distinct.
- `nocons` – suppresses the constant (intercept) term and is specified here for both the fixed effects and random-effects equations.
- `refineopts(iterate(3))` – controls the maximization process during the refinement of starting values. Two iterations is the default. Should the maximization fail because of instability in the Hessian calculations, one possible solution may be to increase the number of iterations here.
- `binomial(n)` – specifies the data are in binomial form and `n` as the binomial variable.
- `variance` – displays the random-effects parameter estimates as variances and covariances. To display them as standard deviations and correlations use option `stddev`.
- In the estimation log a set of iterations used to refine starting values are shown as well as a set of gradient-based iterations. By default, these are Newton-Raphson iterations but other methods are available by specifying the appropriate `maximize` options.
- The first estimation table reports the fixed effects. The second one shows the estimated variance components. The first section of this table is labelled `study: Unstructured` meaning these are random effects at the study level with unstructured covariance.
- The likelihood ratio test at the bottom compares this model to one using standard logistic regression. To know why the LR test is conservative click on the link in the output window to read the information. To avoid the LR test use option `nolr`.

```
. meqrlogit true sens spec if testtype==1, nocons|| study_id: sens spec, ///
> nocons cov(un) binomial(n) refineopts(iterate(3)) intpoints(5) variance

Refining starting values:

Iteration 0:   log likelihood = -395.89917
Iteration 1:   log likelihood = -386.57627
Iteration 2:   log likelihood = -385.31091
Iteration 3:   log likelihood = -385.29942

Performing gradient-based optimization:

Iteration 0:   log likelihood = -385.29942
Iteration 1:   log likelihood = -385.29864
Iteration 2:   log likelihood = -385.29864

Mixed-effects logistic regression               Number of obs     =        178
Binomial variable: n
Group variable: study_id                        Number of groups  =         89

                                                Obs per group:
                                                              min =          2
                                                              avg =        2.0
                                                              max =          2

Integration points =   5                        Wald chi2(2)      =     567.83
Log likelihood = -385.29864                     Prob > chi2       =     0.0000

------------------------------------------------------------------------------
        true |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
        sens |   3.556827   .1742296    20.41   0.000     3.215343    3.898311
        spec |   1.932284   .1234624    15.65   0.000     1.690302    2.174266
------------------------------------------------------------------------------

------------------------------------------------------------------------------
  Random-effects Parameters |   Estimate   Std. Err.      [95% Conf. Interval]
----------------------------+-------------------------------------------------
study_id: Unstructured      |
                  var(sens) |   1.125341   .3199718       .6445544    1.964758
                  var(spec) |   .9009872   .1964317       .5876802    1.381326
               cov(sens,spec) |   .3205208    .178406      -.0291485    .6701901
------------------------------------------------------------------------------
```

The five parameters of the bivariate model

Compare this result with the one obtained using **`metandi`**. The coefficient of `sens` and `spec` are the expected (mean) logit sensitivity and expected logit specificity (labeled as E(logitse) and E(logitsp) in the **`metandi`** output and in RevMan). We are also given estimates of the random effects parameters for logit sensitivity (var(sens)) and logit specificity (var(spec)), and their covariance (cov(sens, spec)). Note that **`metandi`** displays the correlation instead of this covariance. If instead of the covariance between the variances of the random effects you are interested in the correlation, the easiest thing to do is to run **`meqrlogit`** with the `stddev` option so that random-effects parameter estimates are displayed as standard deviations and correlations.

To find the covariance between the expected logit sensitivity and expected logit specificity, type the following to display contents of the variance-covariance matrix:

```
matrix list e(V)
```

(**Note: this is V and not v)**

```
. matrix list e(V)

symmetric e(V)[5,5]
                         eq1:         eq1:    lns1_1_1:    lns1_1_2:  atr1_1_1_2:
                        sens         spec        _cons        _cons        _cons
       eq1:sens     .03035595
       eq1:spec      .003737    .01524296
  lns1_1_1:_cons    .01115963    .00002506    .02021132
  lns1_1_2:_cons    .00016008    .00280439    .00067985       .011883
atr1_1_1_2:_cons    .00265386    .00027093    .00226926    .00273543    .03112703
```

The covariance between the estimated mean logit sensitivity and mean logit specificity is 0.003737. This covariance is required together with the other bivariate parameter estimates for construction of confidence or prediction regions around the summary point.

6.3 Display summary estimates

To transform all values automatically and display the transformed parameters with their standard errors and confidence intervals isn't quite straightforward but you can transform each value manually at a time. If you are content to do it manually and also do not require computation of additional summary measures such as diagnostic odds ratios (DOR) and likelihood ratios from the model parameters, then the rest of this section can be skipped.

Following estimation with **`metandi`** or **`gllamm,`** summary points with their confidence intervals can be displayed using the command **`_diparm`**. **`_diparm`** enables the display of ancillary parameters. Ancillary parameters are often estimated in a transformed metric; for example, rather than estimating sensitivity, logit(sensitivity) is estimated.

First rename the columns of the coefficient and variance-covariance matrices and also the rows of the latter because the command **`_diparm`** expects equations of the form `eqname_cons` (although you only provide `eqname` for the command).

To display the coefficient vector, run

```
matrix list e(b)
```

```
. matrix list e(b)

e(b)[1,5]
            eq1:         eq1:    lns1_1_1:    lns1_1_2:  atr1_1_1_2:
           sens         spec        _cons        _cons        _cons
y1    3.5568271    1.9322842    .05904326   -.05213213    .32976906
```

Unlike the other elements of the vector, we do not have the required form `eqname_cons` for `sens` and `spec`. Secondly, we must save the coefficient vector (`b`) and variance-covariance (`V`) matrix in Stata's system areas. To do this you need to write a short program.

Begin by dropping the program if it is already in Stata's memory. We know it isn't at this point but just in case you decide to rerun the program when it is already in memory. **`capture`** suppresses the error message if the program doesn't exist.

```
capture program drop renamematrix
```

Create the program by typing:

```
program define renamematrix, eclass
      matrix mb = e(b)
      matrix mv = e(V)
      matrix colnames mb = logitse:_cons logitsp:_cons
      matrix colnames mv = logitse:_cons logitsp:_cons
      matrix rownames mv = logitse:_cons logitsp:_cons
      ereturn post mb mv
end
```

The program `renamematrix` renames the matrices `b` and `V` as `mb` and `mv` and their columns/rows. Using the command **`ereturn post`**, the new coefficient vector (`mb`) and variance-covariance (`mv`) matrix are saved in Stata's system areas. `eclass` states that the program being defined returns results in `e()` or modifies already existing results in `e()`.This is done using the **`ereturn`** command. If the program is not explicitly declared to be `eclass`, it may not directly replace or change results in `e()`.

Run the program.

```
renamematrix
```

Finally, display the summary estimates for sensitivity, specificity, DOR and LRs. The DOR and LRs are derived using functions of the expected logit sensitivity and expected logit specificity.

```
_diparm logitse, label(Sensitivity) invlogit

_diparm logitsp, label(Specificity) invlogit

_diparm logitse logitsp, label(DOR) ci(log) function(exp(@1+@2))
derivative(exp(@1+@2) exp(@1+@2))

_diparm logitse logitsp, label(LR+) ci(log) function(invlogit(@1)/(1-
invlogit(@2))) derivative(exp(@2-@1)*invlogit(@1)^2/invlogit(@2)
exp(@2)*invlogit(@1))

_diparm logitse logitsp, label(LR-) ci(log) function((1-
invlogit(@1))/invlogit(@2)) derivative(exp(-
@1)*invlogit(@1)^2/invlogit(@2) exp(-@1-@2)*invlogit(@1))
```

Look up **`_diparm`** in help to understand the syntax. DOR, LR^+ and LR^- are derived from two parameters, logitse and logitsp. `function()` supplies one expression, but the `derivative()` must supply the derivatives with respect to each parameter. Whenever `function()` is specified, the `derivative()` option must also be specified. It is the derivative f'(x) of the function f(x).

Below are the summary estimates with 95% confidence intervals.

```
. capture program drop renamematrix

.
. program define renamematrix, eclass
  1. matrix mb = e(b)
  2. matrix mv = e(V)
  3. matrix colnames mb = logitse:_cons logitsp:_cons
  4. matrix colnames mv = logitse:_cons logitsp:_cons
  5. matrix rownames mv = logitse:_cons logitsp:_cons
  6. ereturn post mb mv
  7. end

.
. renamematrix

.
. _diparm logitse, label(Sensitivity) invlogit
 Sensitivity │    .9722621   .0046987                      .9614076    .9801268

. _diparm logitsp, label(Specificity) invlogit
 Specificity │     .873502   .0136421                      .8442639    .8979147

. _diparm logitse logitsp, label(DOR) ci(log) function(exp(@1+@2)) derivative(exp(@1+@2) exp(@1+@2))
         DOR │     242.042   55.76057                      154.0972    380.1778

. _diparm logitse logitsp, label(LR+) ci(log) function(invlogit(@1)/(1-invlogit(@2))) derivative(exp(@2-@1)*i
> nvlogit(@1)^2/invlogit(@2) exp(@2)*invlogit(@1))
         LR+ │     7.68599   .8361463                      6.210106    9.512631

. _diparm logitse logitsp, label(LR-) ci(log) function((1-invlogit(@1))/invlogit(@2)) derivative(exp(-@1)*inv
> logit(@1)^2/invlogit(@2) exp(-@1-@2)*invlogit(@1))
         LR- │    .0317548   .0054871                      .0226321    .0445547
```

The results in the output above are summarised in the table below.

Measure	***Summary estimate (95% CI)***
Sensitivity	0.97 (0.96, 0.98)
Specificity	0.87 (0.84, 0.90)
Diagnostic odds ratio	242 (154, 380)
Positive likelihood ratio	7.69 (6.21, 9.51)
Negative likelihood ratio	0.03 (0.02, 0.04)

Running these models and getting a neat summary at the end is not trivial and having **`metandi`** is a great help!

To make sure your do-file is correct run the entire file to recreate your analysis.

7 Meta-regression with meqrlogit

The bivariate model is flexible and can be extended to investigate sources of heterogeneity or to compare the accuracy of two or more tests. The same statistical modelling approach (by addition of a covariate to the model) is used for investigating heterogeneity and for making test comparisons. **`meqrlogit`** fits regression models and so it is fairly straightforward to add a covariate to the model. However, dummy variables must be created for the covariate. A likelihood ratio test can be used to compare models with or without a covariate term.

7.1 Create dummy variables for the covariate

To compare CT and MRI in our example, create dummy variables for the covariate *testtype* as follows:

```
gen seCT=0
gen spCT=0
gen seMRI=0
gen spMRI=0
replace seCT=1 if testtype==1 & sens==1
replace spCT=1 if testtype==1 & spec==1
replace seMRI=1 if testtype==2 & sens==1
replace spMRI=1 if testtype==2 & spec==1
```

Recall (see 6.1) that *sens* and *spec* are binary variables—each study has 2 records for each test such that when *sens* = 0, *spec* =1 and *true* = tn, and when *spec* = 0, *sens* = 1 and *true* = tp.

7.2 Separate meta-analysis for each test

The assumption of equal variances for the random effects of the logit sensitivities and the logit specificities of different subgroups may be reasonable in many situations when investigating heterogeneity in the accuracy of a single test, but less so when comparing the accuracy of tests. Macaskill and colleagues provide further guidance in Chapter 10 of the Cochrane Handbook for Systematic Reviews of Diagnostic Test Accuracy (Macaskill et al. 2010).

Since variances may differ between tests, begin by meta-analysing each test separately. This will enable assessment of the variances of the random effects for logit sensitivity and logit specificity for each test, and provide insight into whether or not assumption of equal variances for the tests is likely to be reasonable.

Meta-analysis of CT

```
meqrlogit true sens spec if testtype==1, nocons || ///
study_id: sens spec, nocons cov(un) binomial(n) ///
refineopts(iterate(3)) intpoints(5) variance
```

Meta-analysis of MRI

```
meqrlogit true sens spec if testtype==2, nocons || ///
study_id: sens spec, nocons cov(un) binomial(n) ///
refineopts(iterate(3)) intpoints(5) variance
```

Meta-analysis of CT

true	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
sens	3.556827	.1742296	20.41	0.000	3.215343	3.898311
spec	1.932284	.1234624	15.65	0.000	1.690302	2.174266

Random-effects Parameters	Estimate	Std. Err.	[95% Conf.	Interval]
study_id: Unstructured				
var(sens)	1.125342	.3199718	.6445545	1.964758
var(spec)	.9009872	.1964317	.5876802	1.381326
cov(sens,spec)	.3205208	.178406	-.0291485	.6701901

Meta-analysis of MRI

true	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
sens	1.966556	.162188	12.13	0.000	1.648673	2.284438
spec	.8405047	.2423615	3.47	0.001	.3654848	1.315525

Random-effects Parameters	Estimate	Std. Err.	[95% Conf.	Interval]
study_id: Unstructured				
var(sens)	.1116773	.1616462	.0065449	1.905583
var(spec)	.7268399	.3515484	.2816701	1.875585
cov(sens,spec)	-.1462588	.1381683	-.4170637	.1245461

Examine the variances of the random effects (var(sens) and var(spec)) of the logits and covariances cov(sens, spec) for the 2 tests.

Do you think that the variance for the random effects for logit sensitivity of CT is roughly the same as that of MRI?

Are the variances also similar for the logit specificities of the 2 tests?

What about the covariances?

7.3 Compare test accuracy

Question: Is there a difference in sensitivity and/or specificity between CT and MRI?

Fit the bivariate model without the covariate for *testtype* (model A)

```
meqrlogit true sens spec, nocons || study_id: sens spec, nocons
cov(un) binomial(n) refineopts(iterate(3)) intpoints(5) variance nolr
```

Use **`estimates store`** to store the estimates for the log likelihood from the model above.

```
estimates store A
```

Add covariate terms (the dummy variables) to the model for both logit sensitivity and logit specificity but not for the variance parameters (model B). This model assumes equal variances for the random effects for the logit sensitivities. The same assumption also applies to the variances for the random effects for the logit specificities.

```
meqrlogit true seCT seMRI spCT spMRI, nocons || study_id: sens spec,
nocons cov(un) binomial(n) refineopts(iterate(3)) intpoints(5)
variance nolr

estimates store B
```

Perform a likelihood ratio test comparing the model (A) without covariate with the model (B) that includes the covariate *testtype* and assumes equal variances. Use the stored values in A and B.

```
lrtest A B
```

```
. lrtest A B

Likelihood-ratio test                                 LR chi2(2)  =     41.98
(Assumption: A nested in B)                           Prob > chi2 =    0.0000
```

1. Is there a statistically significant difference in sensitivity and/or specificity between CT and MRI?

There is statistical evidence (chi-square=41.98, 2df, P<0.0001) that the expected sensitivity and/or specificity differs between CT and MRI. However, further analyses is needed to determine if the difference is in sensitivity, specificity, or both. These analyses can be done by dropping covariate terms from the model (i.e. allowing only sensitivity or only specificity to vary by *testtype*), and using likelihood ratio tests to compare the fit of alternative (nested) models.

2. Is there a statistically significant difference in sensitivity between CT and MRI?

Fit the model assuming sensitivity is the same for CT and MRI but allow specificity to vary with *testtype*.

```
meqrlogit true sens spCT spMRI, nocons || study_id: sens spec,
nocons cov(un) binomial(n) refineopts(iterate(3)) intpoints(5)
variance nolr

estimates store C
```

Perform a likelihood ratio test comparing model (C) with the model (B) that allows both sensitivity and specificity to vary with *testtype*. Use the stored values in B and C.

```
lrtest B C
```

```
. lrtest B C

Likelihood-ratio test                                 LR chi2(1)  =     23.50
(Assumption: C nested in B)                           Prob > chi2 =    0.0000
```

There is statistical evidence (chi-square=23.50, 1df, P<0.0001) that the expected sensitivity differs between CT and MRI.

3. Is there a statistically significant difference in specificity between CT and MRI?

Fit the model assuming specificity is the same for CT and MRI but allow sensitivity to vary with *testtype*.

```
meqrlogit true seCT seMRI spec, nocons || study_id: sens spec,
nocons cov(un) binomial(n) refineopts(iterate(3)) intpoints(5)
variance nolr

estimates store C
```

Perform a likelihood ratio test comparing model (D) with the model (B) that allows both sensitivity and specificity to vary with testtype. Use the stored values in B and D.

```
lrtest B D
```

```
. lrtest B D

Likelihood-ratio test                                 LR chi2(1)  =     23.01
(Assumption: D nested in B)                           Prob > chi2 =    0.0000
```

There is statistical evidence (chi-square=23.01, 1df, P<0.0001) that the expected specificity differs between CT and MRI.

Having examined the variances for the random effects for the logits in the separate meta-analysis for each test (also look at a SROC plot of both tests), assumption of equal variances may not be appropriate. There was a difference in the variances of the random effects especially for the logit sensitivities as observed from the meta-analysis of each test. Since there are many studies for each test, it should be possible to fit a model with separate variances for the logits of each test (model E). This may take a while to run...

```
meqrlogit true seCT seMRI spCT spMRI, nocons || study_id: seCT spCT,
nocons cov(un)|| study_id: seMRI spMRI, nocons cov(un) binomial(n)
refineopts(iterate(3)) intpoints(5) variance nolr

estimates store E
```

Perform a likelihood ratio test comparing the model (B) that includes the covariate *testtype* and assumes equal variances for each test with the model (E) that includes the covariate *testtype* but allows for separate variances for each test. Use the stored values in B and E.

```
lrtest B E
```

```
. lrtest B E

Likelihood-ratio test                                 LR chi2(3)  =      9.68
(Assumption: B nested in E)                           Prob > chi2 =    0.0215
```

4. Does model E fit better than model B?

There is statistical evidence (chi-square=9.68, 3df, P=0.02) to suggest that the assumption of equal variances may not be reasonable.

Finally, perform a likelihood ratio test comparing the model (A) without covariate with the model (E) that includes the covariate *testtype* and allows for separate variances for each test. Use the stored values in A and E.

```
lrtest A E
```

```
. lrtest A E

Likelihood-ratio test                                 LR chi2(5)  =     51.66
(Assumption: A nested in E)                           Prob > chi2 =    0.0000
```

There is statistical evidence (chi-square=51.66, 5df, P<0.0001) that the expected sensitivity and/or specificity differs between CT and MRI. You can repeat 2 and 3 above to check whether there is still a difference in sensitivity or specificity when separate variances for each test are allowed in the model. Alternatively, you can perform Wald tests by using the post estimation command **`test.`**

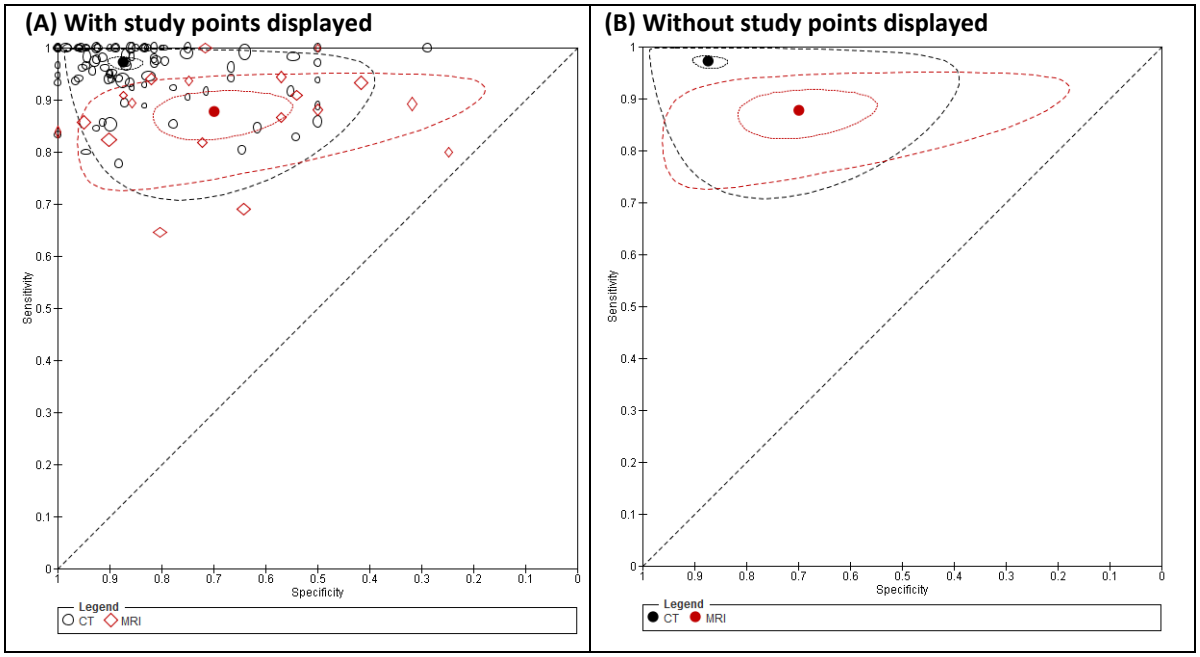
To find the covariance between the estimated mean logit sensitivity and mean logit specificity for each test, type the following to display contents of the variance-covariance matrix:

```
matrix list e(V)
```

```
. matrix list e(V)

symmetric e(V)[10,10]
                        eq1:          eq1:          eq1:          eq1:     lns1_1_1:     lns1_1_2:  atr1_1_1_2:
                       seCT         seMRI          spCT         spMRI         _cons         _cons         _cons
       eq1:seCT    .03035595
      eq1:seMRI    4.662e-10     .02630489
       eq1:spCT     .003737     2.110e-10     .01524296
      eq1:spMRI   -9.127e-10    -.00828709    -1.165e-09     .05873914
  lns1_1_1:_cons    .01115963    9.641e-10     .00002505    -8.795e-10     .02021132
  lns1_1_2:_cons    .00016007    2.657e-10     .00280438    -1.066e-09     .00067985      .011883
atr1_1_1_2:_cons    .00265386    6.156e-10     .00027092    -1.360e-09     .00226926     .00273543     .03112704
  lns1_2_1:_cons    1.001e-08    .02955113     8.255e-09     .00033684     1.078e-08     7.151e-09     8.250e-09
  lns1_2_2:_cons   -8.916e-10     .0001946    -1.505e-09     .00433213    -1.063e-09    -1.844e-09    -2.725e-09
atr1_2_1_2:_cons    5.250e-09     .01225547     3.470e-09     .00602231     4.835e-09     4.004e-09     3.291e-09
```

From the above, the covariance between the estimated mean logit sensitivity and mean logit specificity of CT is 0.003737 and that of MRI is –0.00828709.

The parameter estimates for CT and MRI from model E can be entered into the multiple tests analysis in RevMan to produce a SROC plot with summary operating points for CT and MRI, and their 95% confidence and 95% prediction regions as shown below in figures (A) and (B). Because there are many CT studies and some are clustered together, it is difficult to see the summary point and confidence region for CT in panel (A). Therefore, in panel (B), the SROC plot is shown with only the summary points and regions.



The finely dotted line around each summary point is the 95% confidence region and the dashed line around each point is the 95% prediction region.

Post estimation of a model, the `nlcom` command uses nonlinear combinations of parameter estimates to compute point estimates, and the delta method to compute their standard errors. After fitting model E, the `nlcom` command was used to compute absolute and relative differences in sensitivity and specificity as shown below. The choice of absolute or relative differences depends on review authors. From the output below, the absolute difference in the sensitivity and the specificity of CT compared to MRI are 0.10 (95% CI 0.06, 0.13), and 0.17 (95% CI 0.07, 0.28). The P values in the output tables are from Wald tests.

```
. nlcom diff_sensitivity: invlogit(_b[seCT])-invlogit(_b[seMRI])

diff_sensi~y:  invlogit(_b[seCT])-invlogit(_b[seMRI])

------------------------------------------------------------------------------
             true |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
------------------+-----------------------------------------------------------
diff_sensitivity |   .0950214   .0180869     5.25   0.000     .0595718    .1304711
------------------------------------------------------------------------------

. nlcom diff_specificity: invlogit(_b[spCT])-invlogit(_b[spMRI])

diff_speci~y:  invlogit(_b[spCT])-invlogit(_b[spMRI])

------------------------------------------------------------------------------
             true |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
------------------+-----------------------------------------------------------
diff_specificity |   .1749305   .0528258     3.31   0.001     .0713938    .2784673
------------------------------------------------------------------------------
```

The log of relative sensitivity and relative specificity were computed by taking the difference between the estimated summary sensitivities on the log scale, for example, [log(sensitivity of CT) − log(sensitivity of MRI)] to ensure appropriate estimation of standard errors using the delta method.

Exponents of the estimates in the output below were computed to obtain relative sensitivity and relative specificity of 1.11 (95% CI 1.06, 1.15) and 1.25 (95% CI 1.08, 1.445).

```
. nlcom log_relative_sensitivity: log(invlogit(_b[seCT]))-log(invlogit(_b[seMRI]))

log_relati~y:  log(invlogit(_b[seCT]))-log(invlogit(_b[seMRI]))

------------------------------------------------------------------------------
                        true |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-----------------------------+------------------------------------------------
log_relative_sensitivity     |   .1028441   .0204882     5.02   0.000     .0626879    .1430002
------------------------------------------------------------------------------

. nlcom log_relative_specificity: log(invlogit(_b[spCT]))-log(invlogit(_b[spMRI]))

log_relati~y:  log(invlogit(_b[spCT]))-log(invlogit(_b[spMRI]))

------------------------------------------------------------------------------
                        true |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-----------------------------+------------------------------------------------
log_relative_specificity     |   .2234729   .0747054     2.99   0.003      .077053    .3698929
------------------------------------------------------------------------------
```

7.4 Display summary estimates

Create and run a program similar to the program in 6.3 as follows:

```
capture program drop renamematrix
```

Rename the elements of the coefficient and variance matrices

```
program define renamematrix, eclass
      matrix mb = e(b)
      matrix mv = e(V)
      matrix colnames mb = logitseCT:_cons logitseMRI:_cons
                           logitspCT:_cons logitspMRI:_cons
      matrix colnames mv = logitseCT:_cons logitseMRI:_cons
                           logitspCT:_cons logitspMRI:_cons
      matrix rownames mv = logitseCT:_cons logitseMRI:_cons
                           logitspCT:_cons logitspMRI:_cons
      ereturn post mb mv
end
```

Run the program

```
renamematrix
```

Display summary points by taking the inverse logits of the mean logit sensitivity and mean logit specificity for each test

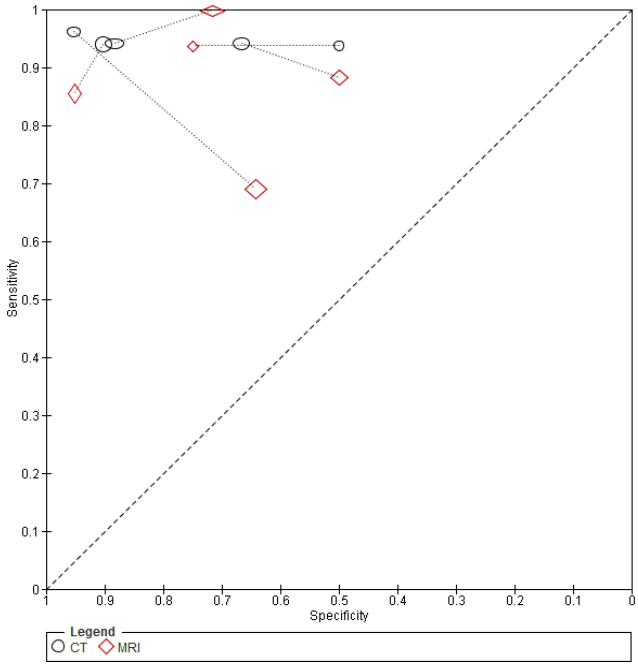
```
_diparm logitseCT, label(Sensitivity CT) invlogit
_diparm logitseMRI, label(Sensitivity MRI) invlogit
_diparm logitspCT, label(Specificity CT) invlogit
_diparm logitspMRI, label(Specificity MRI) invlogit
```

Display other summary estimates derived using functions of the mean logit sensitivities and mean logit specificities

```
_diparm logitseCT logitspCT, label(LR+ CT) ci(log)
function(invlogit(@1)/(1-invlogit(@2))) derivative(exp(@2-
@1)*invlogit(@1)^2/invlogit(@2) exp(@2)*invlogit(@1))

_diparm logitseMRI logitspMRI, label(LR+ MRI) ci(log)
function(invlogit(@1)/(1-invlogit(@2))) derivative(exp(@2-
@1)*invlogit(@1)^2/invlogit(@2) exp(@2)*invlogit(@1))

_diparm logitseCT logitspCT, label(LR- CT) ci(log) function((1-
invlogit(@1))/invlogit(@2)) derivative(exp(-
@1)*invlogit(@1)^2/invlogit(@2) exp(-@1-@2)*invlogit(@1))

_diparm logitseMRI logitspMRI, label(LR- MRI) ci(log) function((1-
invlogit(@1))/invlogit(@2)) derivative(exp(-
@1)*invlogit(@1)^2/invlogit(@2) exp(-@1-@2)*invlogit(@1))
```

The summary estimates for sensitivity are 0.97 (95% CI 0.96, 0.98) for CT and 0.88 (95%CI 0.84, 0.91) for MRI. The summary estimates for specificity are 0.87 (95%CI 0.84, 0.90) for CT and 0.70 (95%CI 0.59, 0.79) for MRI. There was strong evidence that CT has higher sensitivity and higher specificity than MRI for detecting clinically significant coronary artery stenosis (a diameter reduction of 50% or greater).

A limitation of this analysis is that most of the studies included in the meta-analysis were non-comparative; only five studies were direct head-to-head comparisons of CT and MRI (see SROC plot below). Thus, the difference in accuracy is prone to confounding due to differences in patient groups and study methodology.



The dotted line connects the results for CT and MRI within each study.

Summary estimates of test accuracy can differ between meta-analyses of non-comparative and comparative studies (Takwoingi et al. 2013). Therefore, if possible, analysis limited to comparative studies should also be conducted and reported along with the analysis of all studies.

Fitting the bivariate model to these data may be problematic because of the limited number of studies, and the model may need to be simplified (see Takwoingi 2017). See the Cochrane Handbook for Systematic Reviews of Diagnostic Test Accuracy for more on this analysis, and also if interested in other software packages.

What changed between version 1.2 and 2.0?

1. The **`xtmelogit`** command was changed to **`meqrlogit`** but syntax remained the same.
2. The variable study was changed to *recordid* to avoid confusion with *study_id* which is the group variable that unquely identifies each study in the Schuetz dataset. The *recordid* variable uniquely identifies each observation, for example a study that evaluated CT and MRI will have a different *recordid* for CT and MRI (see Dewey 2006 in the snip of the dataset in section 6.1). There would be no difference between using *recordid* and *study_id* in the random component (`||study_id:`) of the **`meqrlogit`** statement if there were no comparative studies of CT versus MRI in the dataset. However, because there are five comparative studies and to ensure both tests are clustered within each comparative study, the random effects need to be specified at the level identified by the group variable *study_id*. For an explanation of the impact of using the observation identifier rather than the study identifier, see Section 7.6.1 in Takwoingi 2016 (https://etheses.bham.ac.uk/id/eprint/6759/).
3. Added code for computing absolute and relative differences in sensitivity and specificity.
4. Minor edits were made to the text due to the above changes and also to improve clarity.

Please email DTA-ET@contacts.bham.ac.uk if you find errors or have suggestions for improvement. Thank you.

References

Chu H, Cole SR. Bivariate meta-analysis for sensitivity and specificity with sparse data: a generalized linear mixed model approach (letter to the Editor). J Clin Epidemiol. 2006;59(12):1331-2.

Harbord RM, Whiting P. metandi: Meta-analysis of diagnostic accuracy using hierarchical logistic regression. Stata Journal. 2009;9(2):211-29.

Harbord, R. metandi: Stata module for meta-analysis of diagnostic accuracy. Statistical Software Components, Boston College Department of Economics. Revised 15 Apr 2008.

Harbord RM, Deeks JJ, Egger M, Whiting P, Sterne JA. A unification of models for meta-analysis of diagnostic accuracy studies. Biostatistics. 2007;8(2):239-51.

Leeflang MM, Deeks JJ, Gatsonis C, Bossuyt PM; Cochrane Diagnostic Test Accuracy Working Group.. Systematic Reviews of Diagnostic Test Accuracy. Ann Intern Med. 2008;149(12):889-97.

Macaskill P, Gatsonis C, Deeks JJ, Harbord RM, Takwoingi Y. Chapter 10: Analysing and presenting results. In: Deeks JJ, Bossuyt PM, Gatsonis C, eds. Cochrane Handbook for Systematic Reviews of Diagnostic Test Accuracy Version 1.0. The Cochrane Collaboration; 2010. Accessed at http://srdta.cochrane.org/ on 31 October 2013.

Rabe-Hesketh S, Skrondal A, Pickles A. "GLLAMM Manual" (October 2004). U.C. Berkeley Division of Biostatistics Working Paper Series. Working Paper 160. Accessed at http://biostats.bepress.com/ucbbiostat/paper160 on 31 October 2013.

Reitsma JB, Glas AS, Rutjes AW, Scholten RJ, Bossuyt PM, Zwinderman AH. Bivariate analysis of sensitivity and specificity produces informative summary measures in diagnostic reviews. J Clin Epidemiol. 2005;58(10):982-90.

Review Manager (RevMan) [Computer program]. Version 5.2. Copenhagen: The Nordic Cochrane Centre, The Cochrane Collaboration, 2012.

Rutter CM, Gatsonis C. A hierarchical regression approach to meta-analysis of diagnostic test accuracy evaluations. Stat Med. 2001;20(19)2865-84.

Schuetz GM, Zacharopoulou NM, Schlattmann P, Dewey M. 2010. Meta-analysis: noninvasive coronary angiography using computed tomography versus magnetic resonance imaging. Ann Intern Med. 2010;152(3):167-77.

Takwoingi Y, Leeflang MM, Deeks JJ. Empirical evidence of the importance of comparative studies of diagnostic test accuracy. Ann Intern Med 2013;158(7):544-54.

Takwoingi Y. Meta-analytic approaches for summarising and comparing the accuracy of medical tests. University of Birmingham Research Archive. 2016. Available from https://etheses.bham.ac.uk/id/eprint/6759/.

Takwoingi Y, Guo B, Riley RD, Deeks JJ. Performance of methods for meta-analysis of diagnostic test accuracy with few studies or sparse data. Stat Methods Med Res 2017; 26: 1896-1911.

Appendix Meta-analysis with gllamm

The structure of the model with **gllamm** is similar to **meqrlogit** in some respects. The main difference in the execution of **gllamm** is that you must define equations for the linear predictors, multiplying the latent variables before running the command to fit the model the first time. `eqs(eq1 eq 0)` below specifies the equation names defined before running **gllamm**. A numeric variable is expected for option `i()` so encode *study_id* and generate a new numeric variable named id. The variables *recordid* and *id* are essentially the same for this analysis since it is the analysis of a single test. However, to avoid confusion if this code is later modified for a test comparison, the *id* variable was created and used instead of *recordid*.

gllamm runs slower than **meqrlogit**. In **metandi**, two univariate models are fitted to provide starting values for **gllamm**.

Type

```
encode study_id, gen(id)

eq eq1: sens
eq eq0: spec

gllamm true sens spec if testtype==1, nocons i(id) nrf(2) ///
eqs(eq1 eq0) family(binomial) link(logit) denom(n)  ip(g) nip(5) adapt
```

See the help file for a description of the available options for **gllamm**.

```
. gllamm true sens spec if testtype==1, nocons i(id) nrf(2) ///
> eqs(eq1 eq0) family(binomial) link(logit) denom(n)  ip(g) nip(5) adapt

Running adaptive quadrature
Iteration 0:    log likelihood = -420.75234
Iteration 1:    log likelihood = -385.35912
Iteration 2:    log likelihood = -385.25912
Iteration 3:    log likelihood = -385.25877


Adaptive quadrature has converged, running Newton-Raphson
Iteration 0:   log likelihood = -385.25877
Iteration 1:   log likelihood = -385.25877  (backed up)
Iteration 2:   log likelihood = -385.25862
Iteration 3:   log likelihood = -385.25862

number of level 1 units = 178
number of level 2 units = 89

Condition Number = 2.1439351

gllamm model

log likelihood = -385.25862

------------------------------------------------------------------------------
        true |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
        sens |   3.561943   .1756551    20.28   0.000     3.217665     3.90622
        spec |   1.935379   .1237949    15.63   0.000     1.692746    2.178013
------------------------------------------------------------------------------


Variances and covariances of random effects
------------------------------------------------------------------------------


***level 2 (id)

    var(1): 1.12793 (.31319328)
    cov(2,1): .31995036 (.1714329) cor(2,1): .31733135

    var(2): .9012745 (.19543816)
------------------------------------------------------------------------------
```

var(1) is the variance of the random effects for logit sensitivity
var(2) is the variance of the random effects for logit specificity
Use either the covariance of the logits, cov(2,1), or the correlation of the logits, cor(2,1) in RevMan.

There may be slight discrepancy in the results obtained compared to those of **`meqrlogit`** due to different starting values and options such as number of integration points. Compare your results with that produced by **`metandi`** when option `gllamm` is used. If you need to run **`metandi`** again remember you have modified the dataset so use the original data.