

Meta-analysis of test accuracy studies in Stata

A bivariate model approach

Yemisi Takwoingi

Version 1.1

April 2016

Please cite this version as: Takwoingi Y. Meta-analysis of test accuracy studies in Stata: a bivariate model approach. Version 1.1. April 2016. Available from: <http://methods.cochrane.org/sdt/>.

Contents

1	Introduction	3
2	Getting started	3
3	Reading data from a file	5
3.1	Set working directory	5
3.2	Read data into Stata	5
3.3	View the data	6
4	Converting strings to numbers	6
5.	Meta-analysis with metandi	7
5.1	Using metandi with RevMan	8
5.2	Producing summary ROC plots with metandi	10
6	Meta-analysis with xtmelogit	14
6.1	Setting up the data	14
6.2	Modelling with xtmelogit	15
6.3	Display summary estimates	18
7	Meta-regression with xtmelogit	21
7.1	Create dummy variables for the covariate	21
7.2	Separate meta-analysis for each test	21
7.3	Compare test accuracy	23
7.4	Display summary estimates	26
	References	29
	Appendix Meta-analysis with gllamm	30

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**.

The mixed models estimation routines **xtmelogit** and **xtmepoisson** were introduced in Stata 10 for fitting two-way, multi-way, multilevel, and hierarchical random effects models on binary and count data respectively. 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.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.

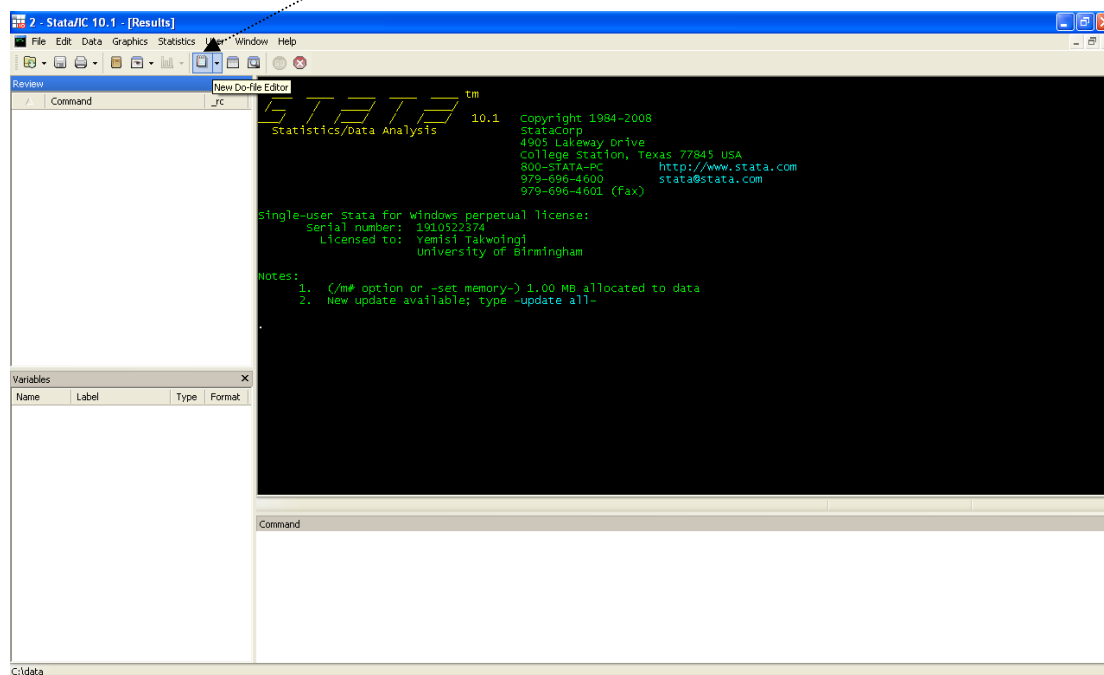
2 Getting started

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

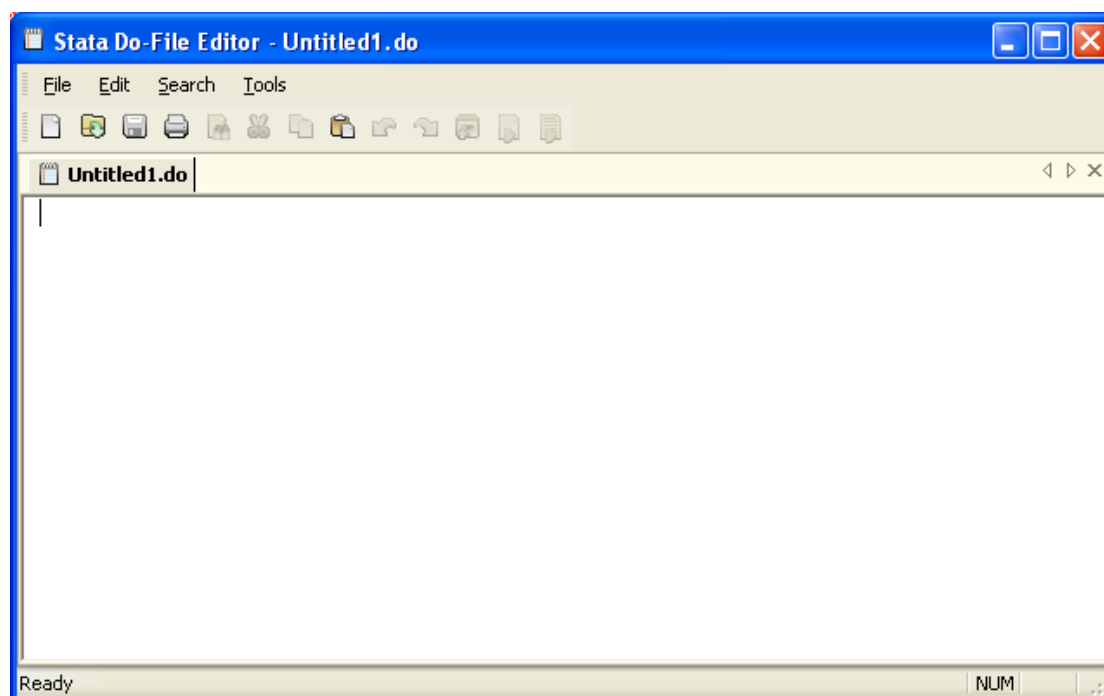
Although it is possible to use Stata interactively (i.e., you type the command in the command window, Stata performs it when you press enter, and any result produced is displayed in the results window, you enter another command, etc.), it is better to write Stata do-files. A do-file is a plain text file 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 `"C:\Users\takwoiny\Desktop\Stata DTA meta-analysis"` with your own path:

```
cd "C:\Users\takwoiny\Desktop\Stata DTA meta-analysis"
```

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 should be shown in the output window.

. describe					
Contains data					
obs:			108		
vars:			6		
size:			3,348	(99.9% of memory free)	
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			

A total of 103 studies provided a 2x2 table for one or both tests. Because 5 studies evaluated both CT and MRI, the total number of observations in the dataset is 108.

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 = -385.31052
Iteration 3: log likelihood = -385.29943
```

Performing gradient-based optimization:

```
Iteration 0: log likelihood = -385.29943
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)	1.125342	.3199719			.6445546 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	.6608845	.2175036			.2345853 1.087184
beta	-.1111755	.1753129	-0.63	0.526	-.4547825 .2324315
s2alpha	2.654912	.6211695			1.678397 4.199578
s2theta	.3432072	.0935866			.2011169 .585685
Summary pt.					
se	.9722621	.0046987			.9614076 .9801268
Sp	.873502	.0136421			.8442639 .8979147
DOR	242.042	55.76058			154.0972 380.1779
LR+	7.68599	.8361464			6.210105 9.512631
LR-	.0317548	.0054871			.0226321 .0445547
1/LR-	31.49133	5.441573			22.44434 44.18503
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 5.2, 2012), 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 of the corresponding analysis in RevMan 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 = -385.31052
Iteration 3: log likelihood = -385.29943
```

Performing gradient-based optimization:

```
Iteration 0: log likelihood = -385.29943
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)	1.125342	.3199719			.6445546 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	.6608845	.2175036			.2345853 1.087184
beta	-.1111755	.1753129	-0.63	0.526	-.4547825 .2324315
s2alpha	2.654912	.6211695			1.678397 4.199578
s2theta	.3432072	.0935866			.2011169 .585685
Summary pt.					
Se	.9722621	.0046987			.9614076 .9801268
Sp	.873502	.0136421			.8442639 .8979147
DOR	242.042	55.76058			154.0972 380.1779
LR+	7.68599	.8361464			6.210105 9.512631
LR-	.0317548	.0054871			.0226321 .0445547
1/LR-	31.49133	5.441573			22.44434 44.18503
Covariance between estimates of E(logitSe) & E(logitSp)					.003737

Externally Calculated Parameters

☐ HSROC model parameters

Parameter	Estimate
Lambda	
Theta	
beta	
Var(accuracy)	
Var(threshold)	

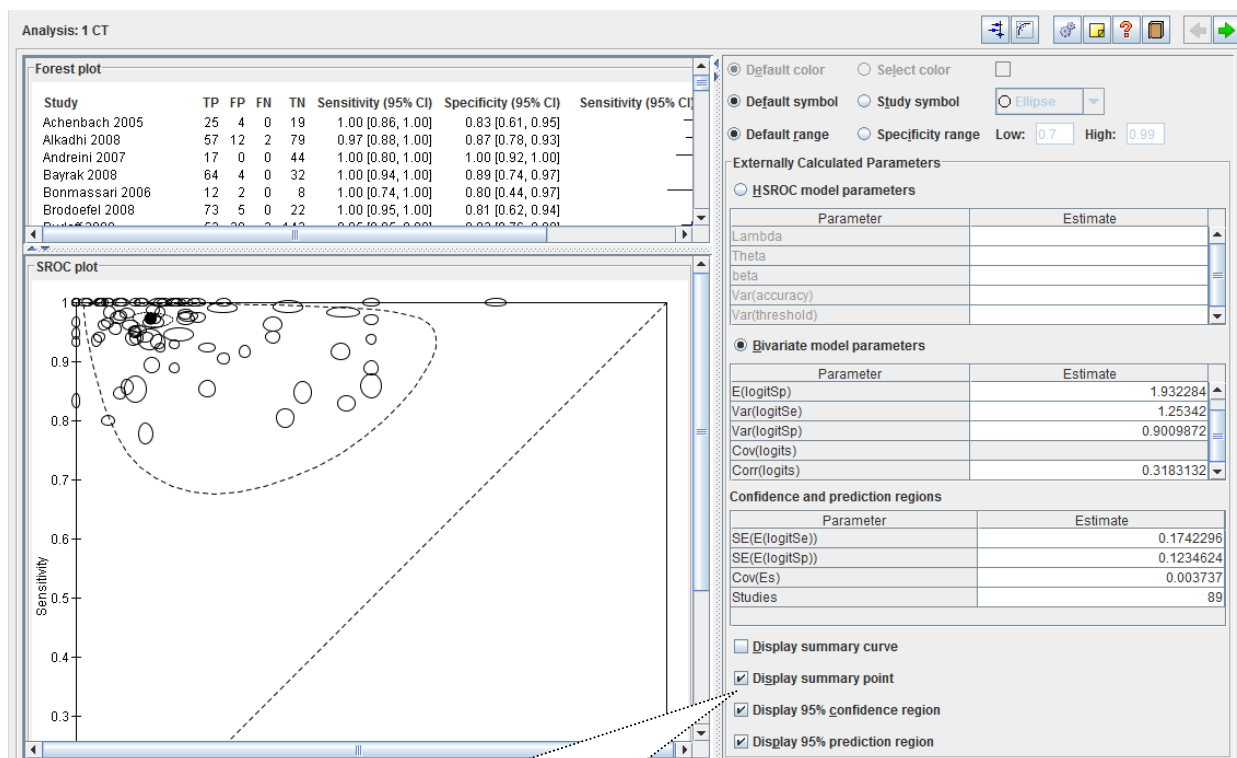
☒ Bivariate model parameters

Parameter	Estimate
E(logitSe)	3.556827
E(logitSp)	1.932284
Var(logitSe)	1.125342
Var(logitSp)	.9009872
Cov(logits)	

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)*. See screen shot below.



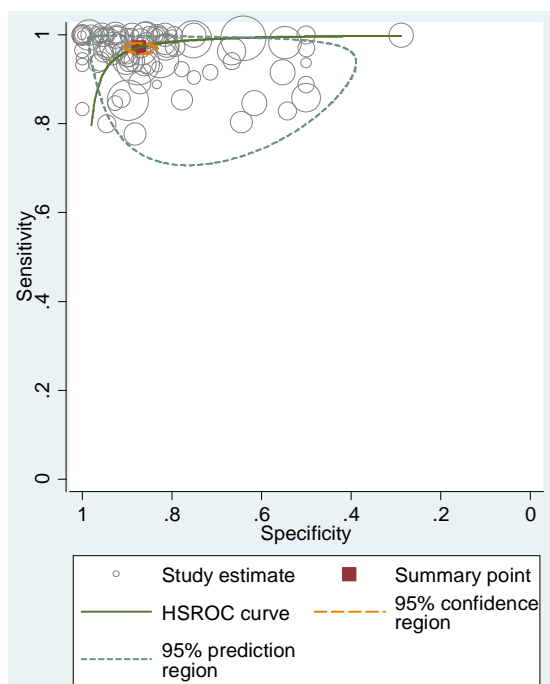
Check the display boxes to display summary points along with confidence and/or prediction regions.

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
```

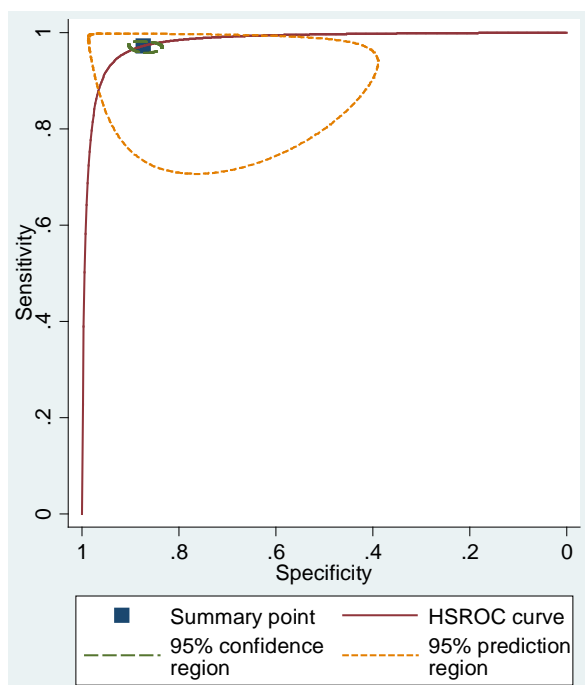
You may need to wait a few seconds for the graph to appear in the graphics editor.



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

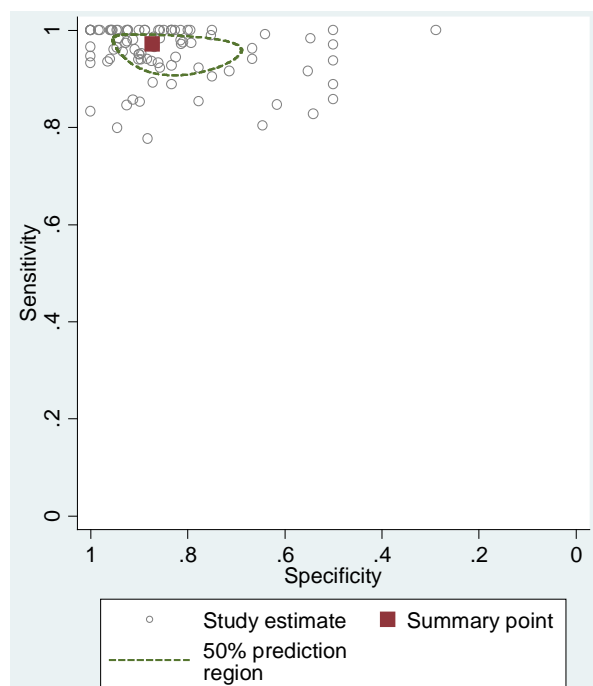
```
metandiplot tp fp fn tn if testtype==1
```

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,

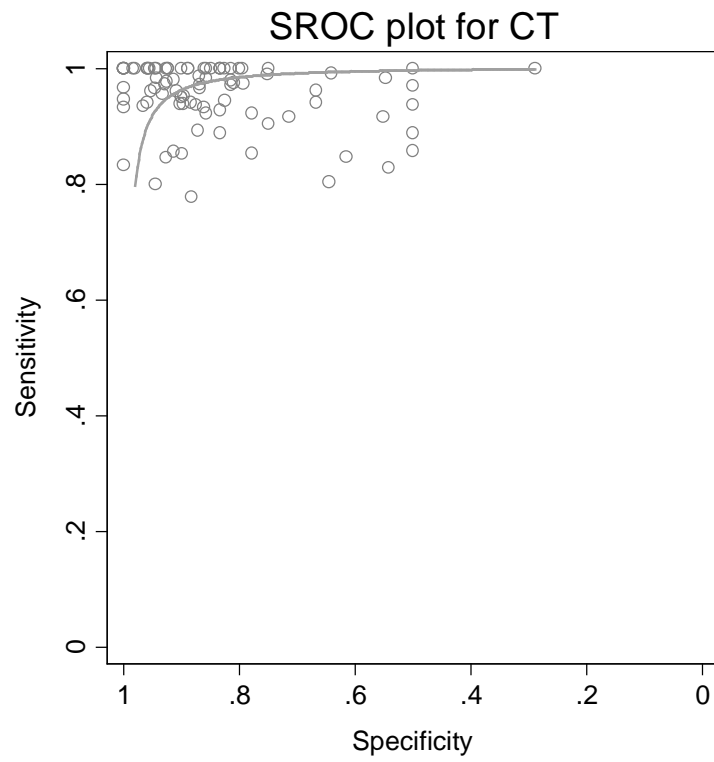
```
metandiplot tp fp fn tn if testtype==1 [aw=1], conf(off) ///
curve(off) predlevel(50)
```

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 if testtype==1 [aw=1], conf(off) ///
pred(off) summ(off) yscale(titlegap(3)) xscale(titlegap(3)) ///
legend(off) title(SROC plot for CT) scheme(slmono)
```



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

6 Meta-analysis with `xtnlogit`

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 `xtnlogit` 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
- `study` is the unique identifier for each study. `_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 study=_n
```

Convert data from wide form to long form

```
reshape long n true, i(study) 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(study)` tells **reshape** that `study` 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` and `sens`. Generate a new binary variable `spec` of type `byte` that takes the value 0 when `sens=1` and vice versa.

```
sort study 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 study=_n

. reshape long n true, i(study) 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

```

. sort study sens
. gen byte spec=1-sens

```

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. The first 10 rows (5 studies) of data in the data editor are shown below

	study	sens	test	study_id	tp	fp	fn	tn	testtype	n	true	spec
1	1	0	CT	Achenbach 2005	25	4	0	19	CT	23	19	1
2	1	1	CT	Achenbach 2005	25	4	0	19	CT	25	25	0
3	2	0	CT	Alkadhi 2008	57	12	2	79	CT	91	79	1
4	2	1	CT	Alkadhi 2008	57	12	2	79	CT	59	57	0
5	3	0	CT	Andreini 2007	17	0	0	44	CT	44	44	1
6	3	1	CT	Andreini 2007	17	0	0	44	CT	17	17	0
7	4	0	CT	Bayrak 2008	64	4	0	32	CT	36	32	1
8	4	1	CT	Bayrak 2008	64	4	0	32	CT	64	64	0
9	5	0	MRI	Bedaux 2002	7	1	0	1	MRI	2	1	1
10	5	1	MRI	Bedaux 2002	7	1	0	1	MRI	7	7	0

6.2 Modelling with xtmelogit

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

```

xtmelogit true sens spec if testtype==1, nocons|| study: 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:** the random effects were specified at the level identified by the group variable **study**.
- **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 because the default is to display them as standard deviations and correlations.
- 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`.


```
. xtmelogit true sens spec if testtype==1, nocons|| study: 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 = -385.31052
Iteration 3:  log likelihood = -385.29943
```

Performing gradient-based optimization:

```
Iteration 0:  log likelihood = -385.29943
Iteration 1:  log likelihood = -385.29864
Iteration 2:  log likelihood = -385.29864
```

Mixed-effects logistic regression

Binomial variable: **n**

Group variable: **study**

Number of obs = 178

Number of groups = 89

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

Integration points = 5

Log likelihood = -385.29864

wald chi2(2) = 567.83

Prob > chi2 = 0.0000

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
true						
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: Unstructured				
var(sens)	1.125342	.3199719	.6445546	1.964758
var(spec)	.9009872	.1964317	.5876802	1.381326
cov(sens,spec)	.3205208	.178406	-.0291485	.6701901

LR test vs. logistic regression: chi2(3) = 361.87 Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

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 **xtmelogit** without the **variance** 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      .03035596
eq1:spec      .003737
lns1_1_1:_cons      .01115963      .00002506      .02021132
lns1_1_2:_cons      .00016008      .00280439      .00067985      .011883
atr1_1_1_2:_cons      .00265386      .00027092      .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, $\text{logit}(\text{sensitivity})$ 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      3.5568272      1.9322842      .05904338      -.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 vlogitse:_cons
    vlogitsp:_cons covlogits:_cons

    matrix colnames mv = logitse:_cons logitsp:_cons vlogitse:_cons
    vlogitsp:_cons covlogits:_cons

    matrix rownames mv = logitse:_cons logitsp:_cons vlogitse:_cons
    vlogitsp:_cons covlogits:_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 LR_s. The DOR and LR_s 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 vlogitse:_cons vlogitsp:_cons covlogits:_cons
4. matrix colnames mv = logitse:_cons logitsp:_cons vlogitse:_cons vlogitsp:_cons covlogits:_cons
5. matrix rownames mv = logitse:_cons logitsp:_cons vlogitse:_cons vlogitsp:_cons covlogits:_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 55.76058 154.0972 380.1779
. _diparm logitse logitsp, label(LR+) ci(log) function(invlogit(@1)/(1-invlogit(@2))) derivative(ex
> p(@2-@1)*invlogit(@1)^2/invlogit(@2) exp(@2)*invlogit(@1))
LR+ | 7.68599 .8361464 6.210105 9.512631
. _diparm logitse logitsp, label(LR-) ci(log) function((1-invlogit(@1))/invlogit(@2)) derivative(ex
> p(-@1)*invlogit(@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 `xtnlogit`

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. `xtnlogit` 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

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

Meta-analysis of MRI

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

CT output

Mixed-effects logistic regression				Number of obs	=	178
Binomial variable: n				Number of groups	=	89
Group variable: study				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: Unstructured						
	var(sens)	1.125342	.3199719	.6445546	1.964758	
	var(spec)	.9009872	.1964317	.5876802	1.381326	
	cov(sens,spec)	.3205208	.178406	-.0291485	.6701901	

MRI output

Mixed-effects logistic regression				Number of obs	=	38
Binomial variable: n				Number of groups	=	19
Group variable: study				obs per group: min	=	2
				avg	=	2.0
				max	=	2
Integration points = 5				wald chi2(2)	=	185.00
Log likelihood = -85.92495				Prob > chi2	=	0.0000
true	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
sens	1.966556	.1621879	12.13	0.000	1.648673	2.284438
spec	.8405047	.2423616	3.47	0.001	.3654847	1.315525
Random-effects Parameters		Estimate	Std. Err.	[95% Conf. Interval]		
study: Unstructured						
	var(sens)	.111677	.161646	.0065448	1.905587	
	var(spec)	.7268402	.3515486	.2816702	1.875586	
	cov(sens,spec)	-.1462589	.1381684	-.417064	.1245462	

Examine the variances of the random effects ($\text{var}(\text{sens})$ and $\text{var}(\text{spec})$) of the logits and covariances $\text{cov}(\text{sens}, \text{spec})$ 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)

```
xtmelogit true sens spec, nocons || study: 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 that the variances for the random effects for the logit sensitivities are similar. The same assumption also applies to the variances for the random effects for the logit specificities.

```
xtmelogit true seCT seMRI spCT spMRI, ///
nocons || study: 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 for each test. Use the stored values in A and B.

```
lrtest A B
```

. lrtest A B			
Likelihood-ratio test	LR chi2(2)	=	27.38
(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=27.38, 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 assuming the same sensitivity or specificity for the tests and dropping the relevant 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*.

```
xtmelogit true sens spCT spMRI, nocons || study: 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) =	19.50
(Assumption: C nested in B)	Prob > chi2 =	0.0000

There is statistical evidence (chi-square=19.5, 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*.

```
xtmelogit true seCT seMRI spec, nocons || study: 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) =	13.24
(Assumption: D nested in B)	Prob > chi2 =	0.0003

There is statistical evidence (chi-square=13.24, 1df, $P = 0.0003$) 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...


```
xtmelogit true seCT seMRI spCT spMRI, nocons || study: seCT spCT, ///
nocons cov(un)|| study: 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
(Assumption: B nested in E)      LR chi2(3) =      8.45
                                Prob > chi2 =     0.0375
```

4. Does model E fit better than model B?

There is statistical evidence (chi-square=8.45, 3df, P=0.038) 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
(Assumption: A nested in E)      LR chi2(5) =     35.84
                                Prob > chi2 =     0.0000
```

There is statistical evidence (chi-square=35.84, 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.

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)
```

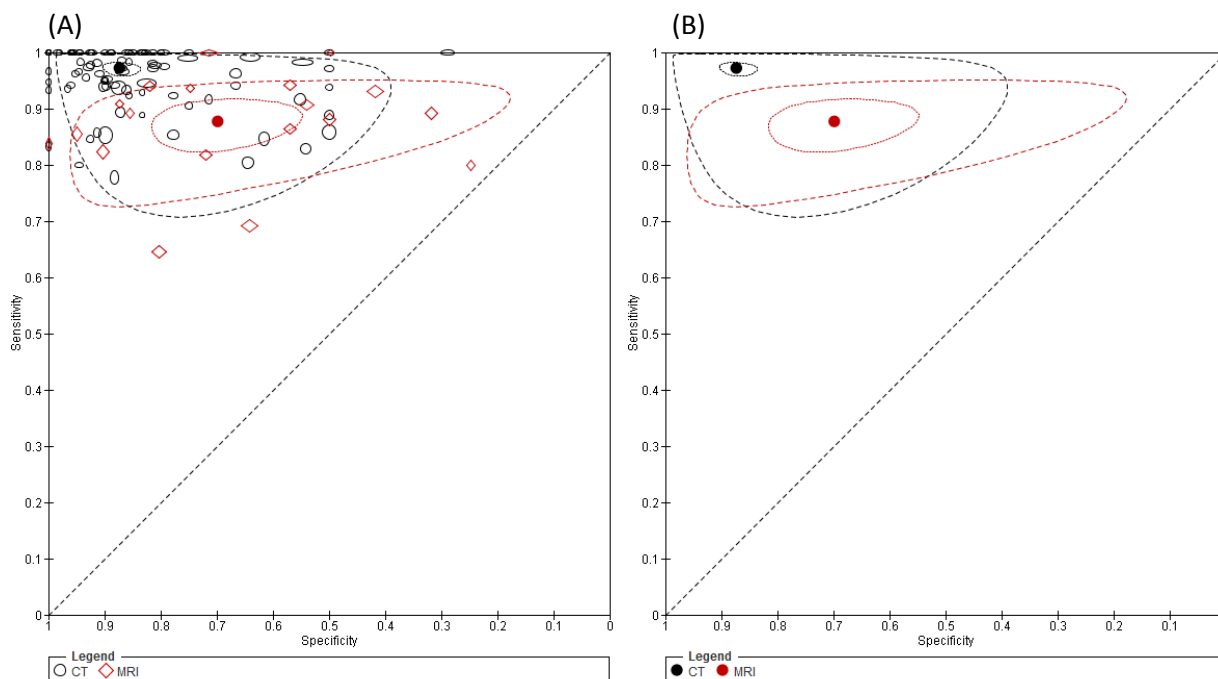
```
. matrix list e(V)

symmetric e(V)[10,10]
      logitseCT:  logitseMRI:  logitspCT:  logitspMRI:  logitspMRI:
      _cons      _cons      _cons      _cons      _cons
logitseCT:_cons  .03035595
logitseMRI:_cons -2.845e-10   .02630489
logitspCT:_cons  .003737    2.552e-11   .01524296
logitspMRI:_cons 2.209e-10   -.00828708  7.029e-11   .05873913
logitspMRI:_cons .01115963  -4.480e-10  .00002506   2.502e-10   .0202
logitspMRI:_cons .00016008  -1.976e-11  .00280438  -3.017e-10   .0006
logitspMRI:_cons .00265386  -7.279e-10  .00027093   2.871e-10   .0022
logitspMRI:_cons -1.438e-09   .02955129   6.985e-10   .00033689  -2.061
logitspMRI:_cons 8.310e-10   .0001946    8.392e-11   .00433208   8.779
logitspMRI:_cons 2.271e-09   .01225551  -2.037e-11   .00602231   4.149
```

Note: The output above was truncated on the right to fit the page.

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 -0.00828708 .

The parameter estimates for CT and MRI from model E can be entered into the corresponding 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 it is difficult to see the summary point and confidence region for CT in (A). Therefore, in (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.

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))

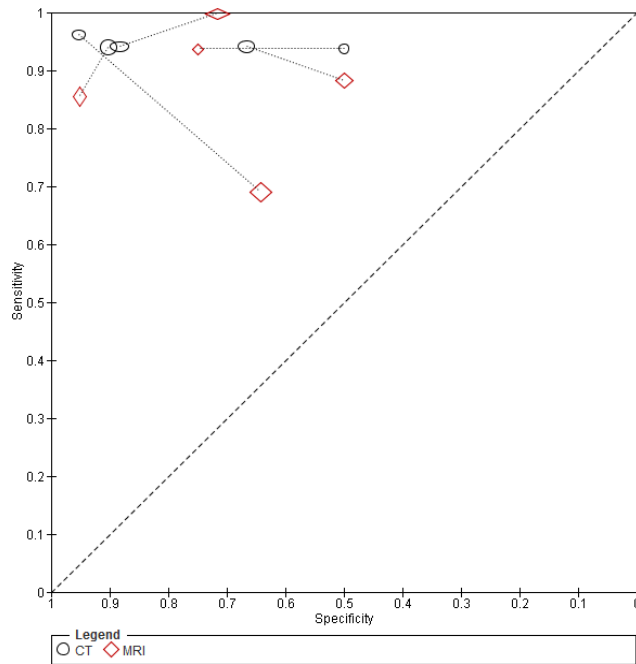
_diparm logitseMRI logitspMRI, label(LR+ MRI) ci(log)
function(invlogit(@1)/(1-invlogit(@2))) derivative(exp(@2-
@1)*invlogit(@1)^2/invlogit(@2) exp(@2)*invlogit(@1))

_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 5 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.



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 Chapter 10 of the Cochrane Handbook for Systematic Reviews of Diagnostic Test Accuracy (Macaskill et al. 2010) for more on this analysis, and also if interested in SAS code for fitting the bivariate or HSROC model.

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

Please email uksu@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.

Appendix Meta-analysis with gllamm

The structure of the model with **gllamm** is similar to **xtmelogit** 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**.

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

Type

```
eq eq1: sens
eq eq0: spec

gllamm true sens spec if testtype==1, nocons i(study) 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(study) 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 = 2.1439352

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 (study)

var(1): 1.12793 (.31319327)
cov(2,1): .31995032 (.17143291) cor(2,1): .31733131

var(2): .90127452 (.19543815)
-----

```

There may be slight discrepancy in the results obtained compared to those of **xtmelogit** 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.