Software for network meta-analysis

Gert van Valkenhoef

Taipei, Taiwan, 6 October 2013

umcg

university of groningen

Workshop structure

- Introduction
  - Network meta-analysis; software
  - GeMTC R package
- Install & Run GeMTC (if needed)
- Worked example
  - All the code provided
  - Most results are not on the slides
  - To get results, run the code yourself!
- Free example
  - Apply what you've learned
- Discussion

Network meta-analysis

- is an extension of pair-wise meta-analysis
  - to handle > 2 interventions simultaneously
- is a *must have* for decision making
  - since it *guarantees* consistency of results
- enforces consistency (i.e. transitivity)
  - $d_{AC} = d_{AB} + d_{BC}$
  - which is *not* an additional assumption
  - but a consequence of *exchangeability*
- exchangeability, also assumed in pair-wise meta-analysis
  - all studies are similar...
  - in terms of effect-modifying covariates
  - such as: population, study design, follow-up

Software for network meta-analysis

General purpose software:

- WinBUGS, OpenBUGS, JAGS (Bayesian)
  - BY FAR most used, most flexible

Software for network meta-analysis

General purpose software:

- WinBUGS, OpenBUGS, JAGS (Bayesian)
  - BY FAR most used, most flexible
- Meta-regression software (frequentist)

Software for network meta-analysis

General purpose software:

- WinBUGS, OpenBUGS, JAGS (Bayesian)
  - BY FAR most used, most flexible
- Meta-regression software (frequentist)
- Multi-variate meta-analysis software (frequentist)
  - E.g. mvmeta (for Stata) can handle network meta-analysis

Software for network meta-analysis

General purpose software:

- WinBUGS, OpenBUGS, JAGS (Bayesian)
  - BY FAR most used, most flexible
- Meta-regression software (frequentist)
- Multi-variate meta-analysis software (frequentist)
  - E.g. mvmeta (for Stata) can handle network meta-analysis

Specialized:

Software for network meta-analysis

General purpose software:

- WinBUGS, OpenBUGS, JAGS (Bayesian)
  - BY FAR most used, most flexible
- Meta-regression software (frequentist)
- Multi-variate meta-analysis software (frequentist)
  - E.g. mvmeta (for Stata) can handle network meta-analysis

Specialized:

- GeMTC R package (Bayesian)

Software for network meta-analysis

General purpose software:

- WinBUGS, OpenBUGS, JAGS (Bayesian)
  - BY FAR most used, most flexible
- Meta-regression software (frequentist)
- Multi-variate meta-analysis software (frequentist)
  - E.g. mvmeta (for Stata) can handle network meta-analysis

Specialized:

- GeMTC R package (Bayesian)
- netmeta R package (frequentist)

Software for network meta-analysis

General purpose software:

- WinBUGS, OpenBUGS, JAGS (Bayesian)
  - BY FAR most used, most flexible
- Meta-regression software (frequentist)
- Multi-variate meta-analysis software (frequentist)
  - E.g. mvmeta (for Stata) can handle network meta-analysis

Specialized:

- GeMTC R package (Bayesian)
- netmeta R package (frequentist)

All require some experience with statistical software

Software for network meta-analysis

General purpose software:

- WinBUGS, OpenBUGS, JAGS (Bayesian)
  - BY FAR most used, most flexible
- Meta-regression software (frequentist)
- Multi-variate meta-analysis software (frequentist)
  - E.g. mvmeta (for Stata) can handle network meta-analysis

Specialized:

- GeMTC R package (Bayesian)
- netmeta R package (frequentist)

All require some experience with statistical software

- Not (yet?) available in dedicated MA software

Using BUGS/JAGS for network meta-analysis

- Advantages
  - Example code available for many data types
  - No need for 'correction' or 'imputation' of data
  - Can fit exact likelihood (unlike frequentist MA)
  - Modeling in BUGS is *very* flexible
- Disadvantages
  - Requires knowledge of MCMC methods, convergence
  - Some models can take a long time to run
  - Modifying existing code by hand sometimes error-prone
  - Working with BUGS is not always easy / intuitive

GeMTC: an R package

- GeMTC R package: interface around BUGS/JAGS
- Designed especially for network meta-analysis
- Writes the BUGS/JAGS code based on given data
- Much easier to work with than BUGS/JAGS directly
- Gives the full power of R for output analysis
- But... is a lot less flexible than coding models yourself!

GeMTC features

- Fixed effect and random effects models
- Various likelihoods:
  - Dichotomous (count) data: binom/logit
  - Survival (rate) data: binom/cloglog, poisson/log
  - Continuous data: normal/identity
- Visualizations:
  - Network graphs
  - Forest plots
  - Rank probability plots
  - Posterior distributions, time series, etc. (through CODA)
- Assessment of heterogeneity / inconsistency
  - Node-splitting
  - ANOHE
- Model fit: DIC (JAGS only)

GeMTC documentation

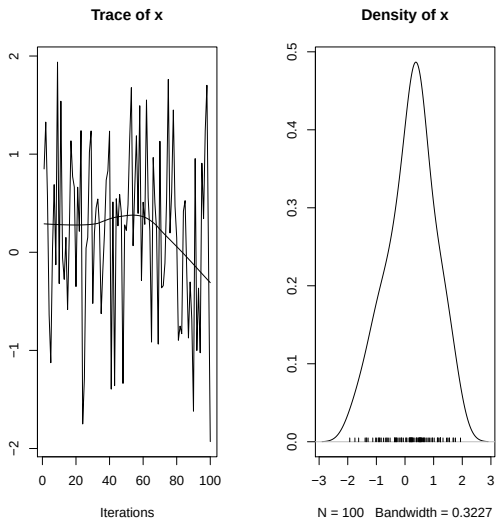
- `http://drugis.org/gemtc`
- `http://cran.r-project.org/web/packages/gemtc/`
- `http://github.com/gertvv/gemtc/`
- `?gemtc` in R

Installing the required software

- Everyone should have:
  - R
  - JAGS
  - The `rjags` and `gemtc` packages for R
- If not, installation instructions are in the handout

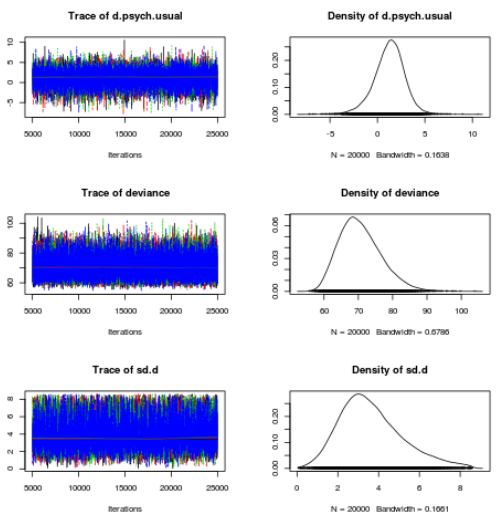
Testing rjags (optional)

```
> library(rjags)
  ...
> model <- jags.model(textConnection('model { x ~ dnorm(0, 1) }'))
  ...
> samples <- coda.samples(model, variable.names='x', n.iter=100)
  ...
> plot(samples)
```



Testing GeMTC

```
> library(gemtc)
  ...
> fileName <- system.file('extdata/welton-systolic.gemtc',
+                         package='gemtc')
> network <- read.mtc.network(fileName)
> model <- mtc.model(network)
> result <- mtc.run(model)
  ...
> plot(result)
```



Dataset: smoking cessation

Interventions to stop smoking:

- A: No treatment
- B: Self-help
- C: Individual counseling
- D: Group counseling

Outcome:

- Number of participants that stopped smoking

Dataset:

- 24 trials, including 2 three-arm trials

Provided to you as `smoking.xls`

Exporting the data (Excel)

Open the file in Excel:

	A	B	C	D
1	study	treatment	stopped smoking	participants
2	1	A	9	140
3	1	C	23	140
4	1	D	10	138
5	2	B	11	78
6	2	C	12	85
7	2	D	29	170
8	3	A	79	702
9	3	B	77	694
10	4	A	18	671
11	4	B	21	535

Save the data for GeMTC:

- Change ‘stopped smoking’ to ‘responders’
- Change ‘participants’ to ‘sampleSize’
- Save as ’smoking.csv’ (CSV file)

Importing the data

Make sure R is running in the correct directory...

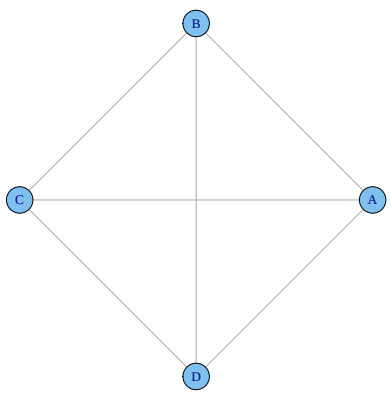
```
> getwd()
[1] "C:/Documents and Settings/Gert/"
> setwd("My Documents")
```

Then import the data:

```
> data <- read.table('smoking.csv', header=TRUE, sep=',')
> data
   study treatment responders sampleSize
1      1         A          9        140
2      1         C         23        140
...
```

Build the network

```
> network <- mtc.network(data.ab=data,
+                        description='Hasselblad (1998) smoking data')
> summary(network)
$Description
[1] "MTC dataset: Hasselblad (1998) smoking data"
...
> plot(network)
```



Also see `?mtc.network` for help / information

Build a model

We start with a fixed effect model:

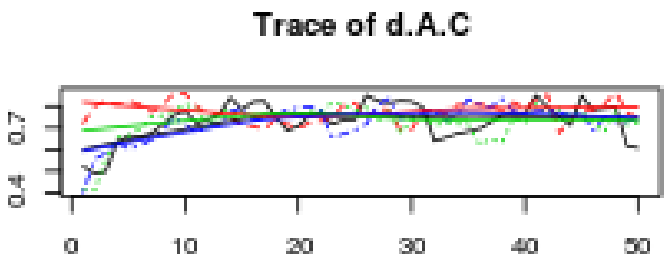
```
> model.fe <- mtc.model(network, linearModel='fixed')
```

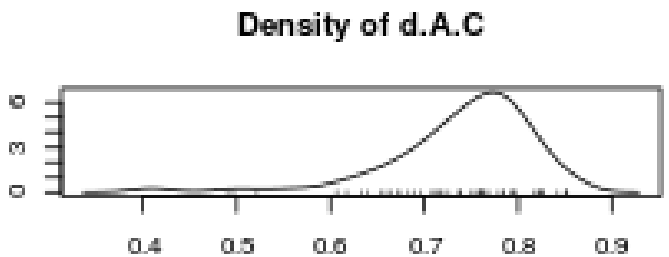
Try the following:

- `?mtc.model`
- `plot(model.fe)`
- `summary(model.fe)`
- `cat(model.fe$code)`
- `model.fe$data`

Run the model (1/3)

```
> result.fe <- mtc.run(model.fe, n.adapt=0, n.iter=50)
> plot(result.fe)
> gelman.diag(result.fe)
```





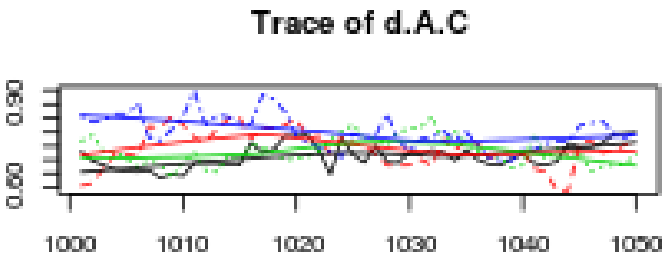
The simulation did not converge

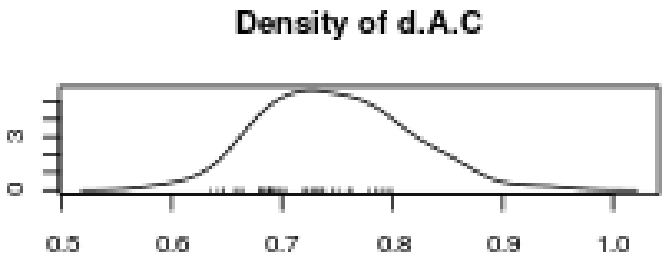
- 'Plume' in first start of the chain
- Visible movement of individual chains
- Gelman-Rubin diagnostics $>> 1$

We need a longer `n.adapt`

Run the model (2/3)

```
> result.fe <- mtc.run(model.fe, n.adapt=100, n.iter=50)
> plot(result.fe)
> gelman.diag(result.fe)
```





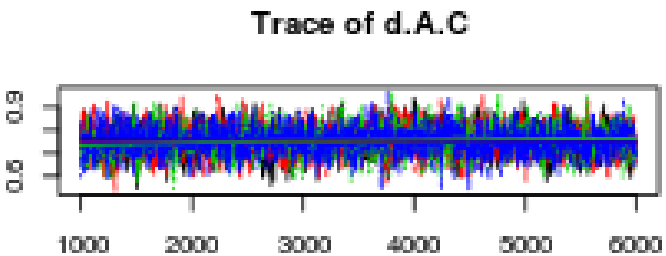
This is better, but...

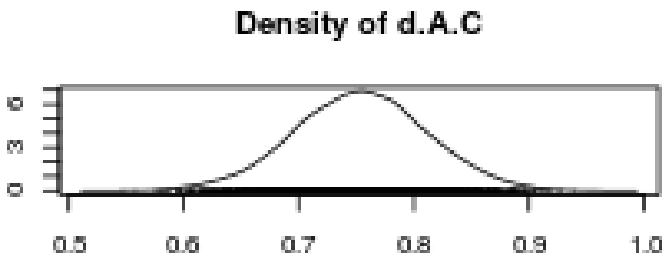
- Visible movement of individual chains
- Gelman-Rubin diagnostics $>> 1$

We need a longer `n.iter`

Run the model (3/3)

```
> result.fe <- mtc.run(model.fe, n.adapt=1000, n.iter=5000)
> plot(result.fe)
> gelman.diag(result.fe)
```





This looks very good!

Look at the results

```
> summary(result.fe)
> forest(result.fe)
> forest(relative.effect(result.fe, t1="C"))
```

How would you interpret these results?

Heterogeneity: is fixed effects OK?

```
> model.re <- mtc.model(network, linearModel='random')
> result.re <- mtc.run(model.re, n.adapt=1000, n.iter=5000)
> plot(result.re, ask=TRUE)
> gelman.diag(result.re)
> summary(result.re)
```

Let's compare model fit (DIC):

```
> result.fe$dic
> result.re$dic
```

- RE model more complex (high pD)
- FE model worse fit (high deviance)
- **RE model overall better (lower DIC)**
- Also note the `sd.d` is quite high!

Heterogeneity: visual methods

Can't we just look at a Forest plot?

- Forest plots are not so easy to draw for networks
- Multi-arm trials make everything more complicated
- But... we'll get to it later on!

You can use Forest plots for pair-wise comparisons

- Use e.g. the `meta` package
- Generally a very good idea to do this!

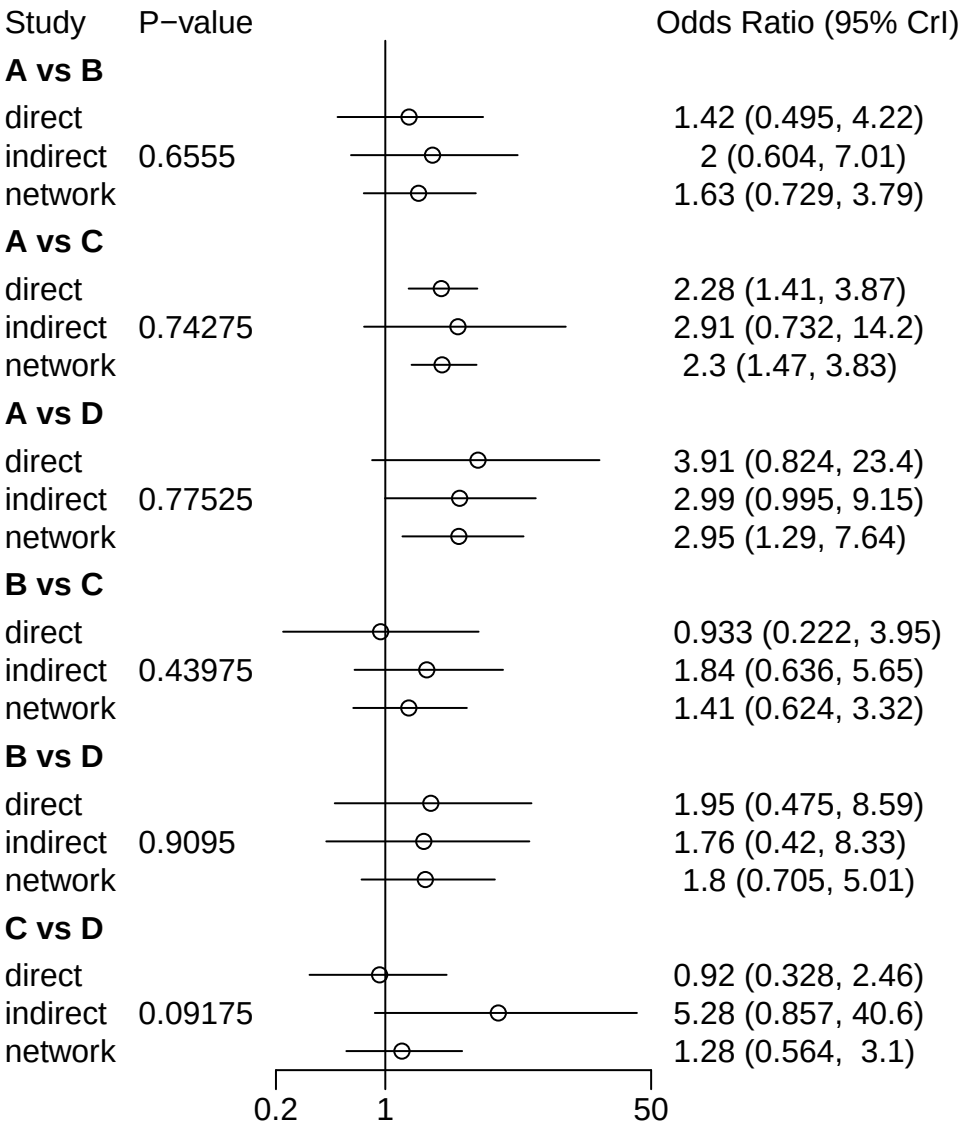
Inconsistency: node-splitting

GeMTC can do a full node-splitting analysis:

```
> result.ns <- mtc.nodesplit(network,
+                            linearModel='random',
+                            n.adapt=500,
+                            n.iter=2000)
> plot(result.ns)
> summary.ns <- summary(result.ns)
> plot(summary.ns)
> summary.ns
```

Very time consuming!

Inconsistency: node-splitting results



Inconsistency: node-splitting results

- Could be inconsistency in C vs D?
- However indirect very uncertain...
- ... and node-splitting 'maximizes' inconsistency
- (i.e. favors finding inconsistency over heterogeneity)

ANOHE: inconsistency / heterogeneity (EXPERIMENTAL)

```
> result.anohe <- mtc.anohe(network, n.adapt=1000, n.iter=5000)
> plot(result.anohe)
> summary.anohe <- summary(result.anohe)
> plot(summary.anohe, xlim=log(c(0.2, 5)))
> summary.anohe
```

- Looks like there is lots of heterogeneity...
- ... but probably no inconsistency

WARNINGS:

- New method (still to be published...)
- Only random effects works

Final results

Conclusions:

- Random effects model (high heterogeneity)
- Network appears to be consistent

Now we can:

- `forest(result.re)`
- `summary(result.re)`
- `plot(rank.probability(result.re))`

Example: thrombolytics data

Apply what you've learned...

```
> fileName <- system.file('extdata/luades-thrombolytic.gemtc',
+                         package='gemtc')
> network <- read.mtc.network(fileName)
```

- What studies and interventions do we have?
- Fixed effect or random effects?
- Are there any inconsistencies?
  - If so, how would you deal with it?
- How would you rank the treatments?

Doing a network meta-analysis

- Decide on the research question
  - Population, Intervention, (Control,) Intervention
- Identify the relevant studies
  - An art & science of itself
- Assess the studies & network
  - Are they similar enough to be combined?
  - Are there covariates / confounders / biases?
- Decide on basic statistical model
  - What are the appropriate likelihood and link?
  - Account for covariates / confounders / biases?
- Run models, assess convergence, compare model fit
  - Level of heterogeneity, inconsistency
  - Fixed or random effects?
- Decide on the model, interpret results

Take-away

- GeMTC makes network meta-analysis easier
  - No need to write BUGS/JAGS code
  - Visualizations specifically for network meta-analysis
  - Methods to assess heterogeneity / inconsistency
- But, network meta-analysis is still not easy
  - Checking convergence still necessary
  - Choosing the right statistical model is hard
  - Checking assumptions is tricky
  - ... what to do when they fail is even harder!
- R / GeMTC can do much more than what we saw today
  - I would recommend to invest in learning R!

Thank you!

- I hope you enjoyed this workshop
- And that you learned a lot
- Feel free to come to me with questions / suggestions