University of BRISTOL

for Health Technology Assessment

Howard Thom

With Gianluca Baio, Nathan Green, Anthony Hatswell, Claire Williams, Nicky Welton, Pedro Saramago, Marta Soares, Padraig Dixon, James O'Mahony

(Scientific Committee of the R for HTA organization)

bristol.ac.uk

MRC Hubs for Trials Methodology Research

ConDuCT-II Hub

Background

- Majority of cost-effectiveness analysis (CEA) conducted in Excel
  - Computationally slow
  - Inflexible so limited to simple models
  - Calculations are opaque
  - Limited statistical analysis
- R statistical programming language overcomes these issues
- In this presentation, I'll give a tour of the advantages of R
- I'll briefly show how to implement a decision tree in R
- I'm keen for your feedback on advantages and disadvantages of R

What software do you mostly use for cost-effectiveness analysis?

CHOICE	VOTES	
Excel or other spreadsheet	54	87%
R	4	6%
MATLAB	1	2%
STATA	3	5%
BUGS (WinBUGS, OpenBUGS, JAGS, STAN)	0	0%

ANSWER CHOICES	RESPONSES
Excel or other spredsheet	60.61%
R	18.18%
Matlab	0.00%
STATA	18.18%
BUGS (WinBUGS, OpenBUGS, JAGS, STAN)	3.03%
TOTAL	

bristol.ac.uk

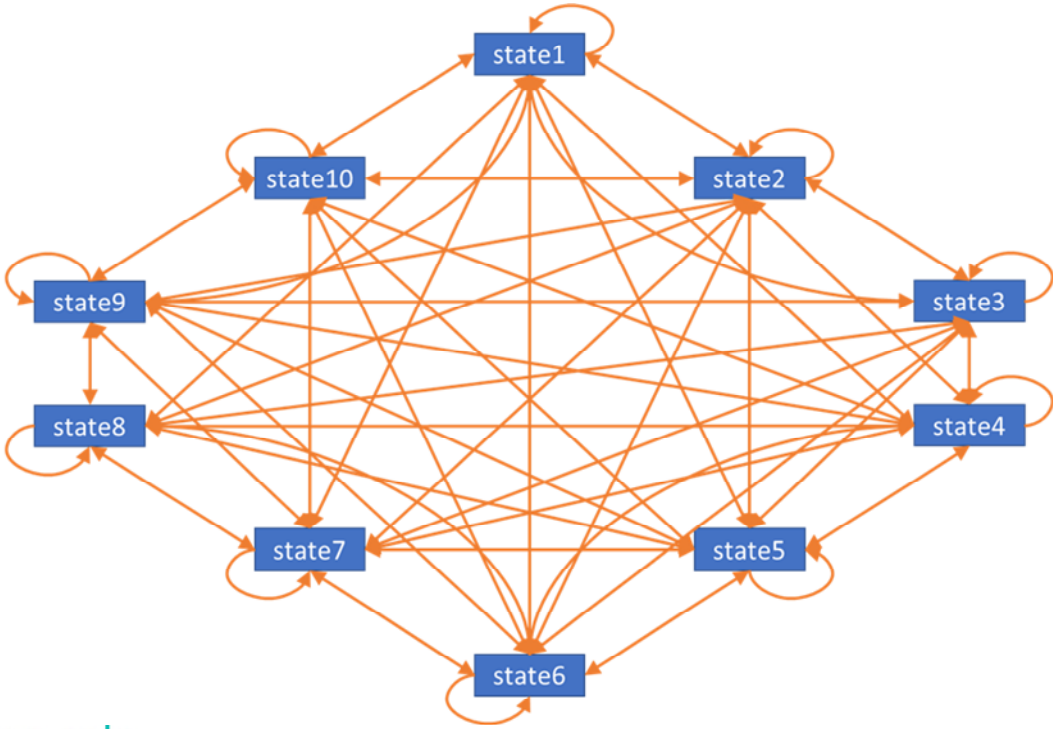
R for Health Technology Assessment

Why R?

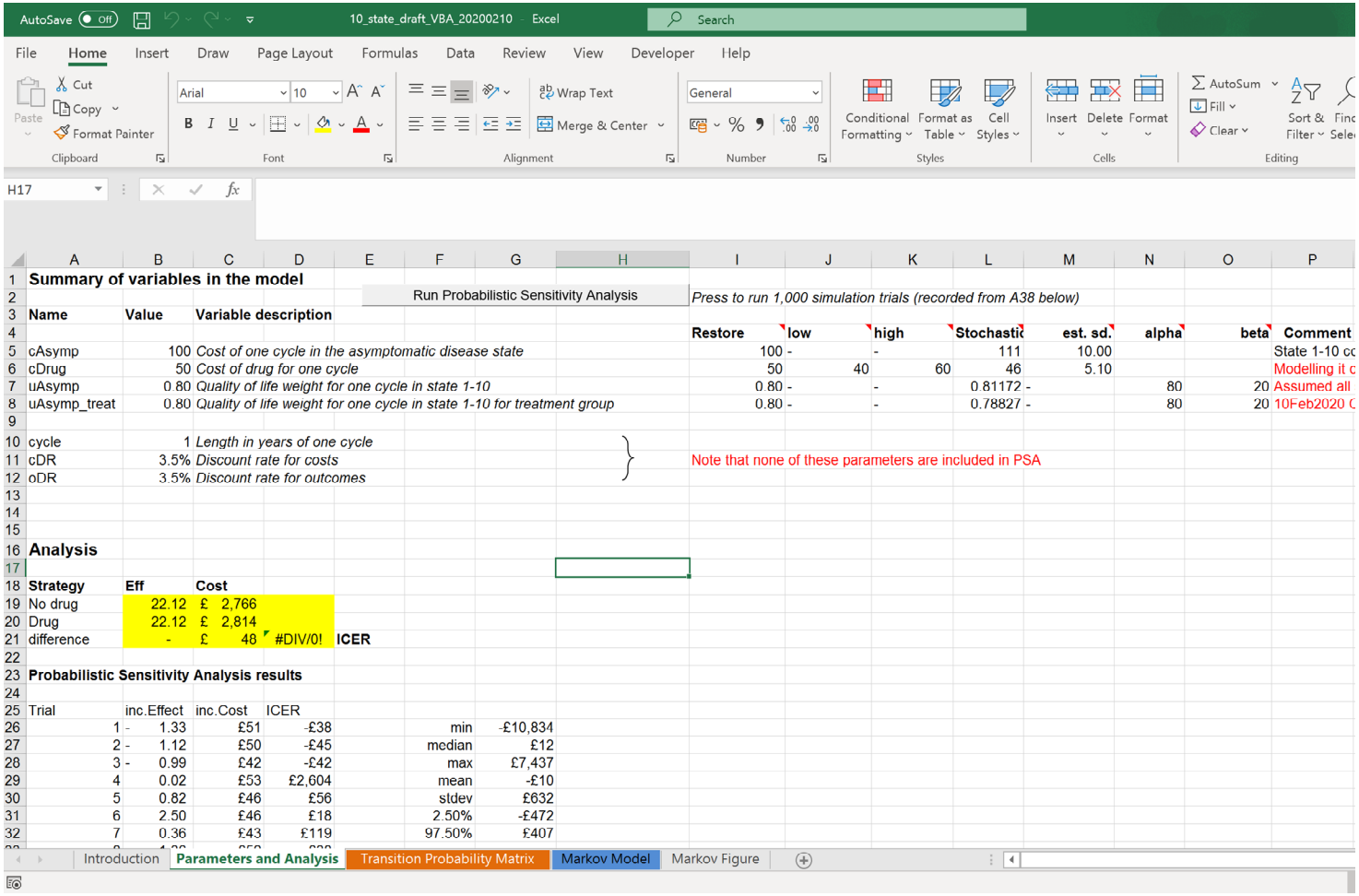
- Very fast, particularly with matrix operations common in cost-effectiveness analysis (CEA) modelling
- One-step analysis: all data analysis and economic modelling can be conducted in the same script.
- Packages: R is hugely extensible with thousands of packages.
  - These include many for HTA purposes, which I'll briefly review.
- All analyses, including sensitivities and scenario analyses, can be run from a single script.
  - This ensures reproducibility.
- Transparent: Excel formulae are hidden and tracing the web of dependencies is difficult.
- Flexible: R is a programming language (syntax is similar to C/C++) with functions, program flow, and object oriented features.
  - This enables much more ambitious economic models to be implemented, such as individual level simulations, discrete event simulation, and value of information analysis

Speed comparisons

- We'll start with the obvious advantage: speed.
- We'll show the same CEA model implemented in Excel and R
- This is a live demonstration
- 10-state Markov model, two treatments, 1000 samples.
  - Costs, utilities, and transition probabilities are random



Excel version



AutoSave Off | 10_state_draft_VBA_20200210 - Excel | Search

File Home Insert Draw Page Layout Formulas Data Review View Developer Help

Cut Copy Format Painter | Paste | Arial 10 | Wrap Text | Merge & Center | General | Conditional Formatting | Format as Table | Cell Styles | Insert | Delete | Format | AutoSum | Fill | Clear | Sort & Filter

Clipboard Font Alignment Number Styles Cells Editing

H17

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	**Summary of variables in the model**															
2					Run Probabilistic Sensitivity Analysis				*Press to run 1,000 simulation trials (recorded from A38 below)*							
3	**Name**	**Value**	**Variable description**													
4									**Restore**	**low**	**high**	**Stochastic**	**est. sd.**	**alpha**	**beta**	**Comment**
5	cAsymp	100	*Cost of one cycle in the asymptomatic disease state*						100	-	-	111	10.00			State 1-10 co
6	cDrug	50	*Cost of drug for one cycle*						50	40	60	46	5.10			Modelling it c
7	uAsymp	0.80	*Quality of life weight for one cycle in state 1-10*						0.80	-	-	0.81172	-	80	20	Assumed all
8	uAsymp_treat	0.80	*Quality of life weight for one cycle in state 1-10 for treatment group*						0.80	-	-	0.78827	-	80	20	10Feb2020 C
9																
10	cycle	1	*Length in years of one cycle*													
11	cDR	3.5%	*Discount rate for costs*						Note that none of these parameters are included in PSA							
12	oDR	3.5%	*Discount rate for outcomes*													
13																
14																
15																
16	**Analysis**															
17																
18	**Strategy**	**Eff**	**Cost**													
19	No drug	22.12	£ 2,766													
20	Drug	22.12	£ 2,814													
21	difference	-	£ 48	#DIV/0!	**ICER**											
22																
23	**Probabilistic Sensitivity Analysis results**															
24																
25	Trial	inc.Effect	inc.Cost	ICER												
26	1	- 1.33	£51	-£38		min	-£10,834									
27	2	- 1.12	£50	-£45		median	£12									
28	3	- 0.99	£42	-£42		max	£7,437									
29	4	0.02	£53	£2,604		mean	-£10									
30	5	0.82	£46	£56		stdev	£632									
31	6	2.50	£46	£18		2.50%	-£472									
32	7	0.36	£43	£119		97.50%	£407									

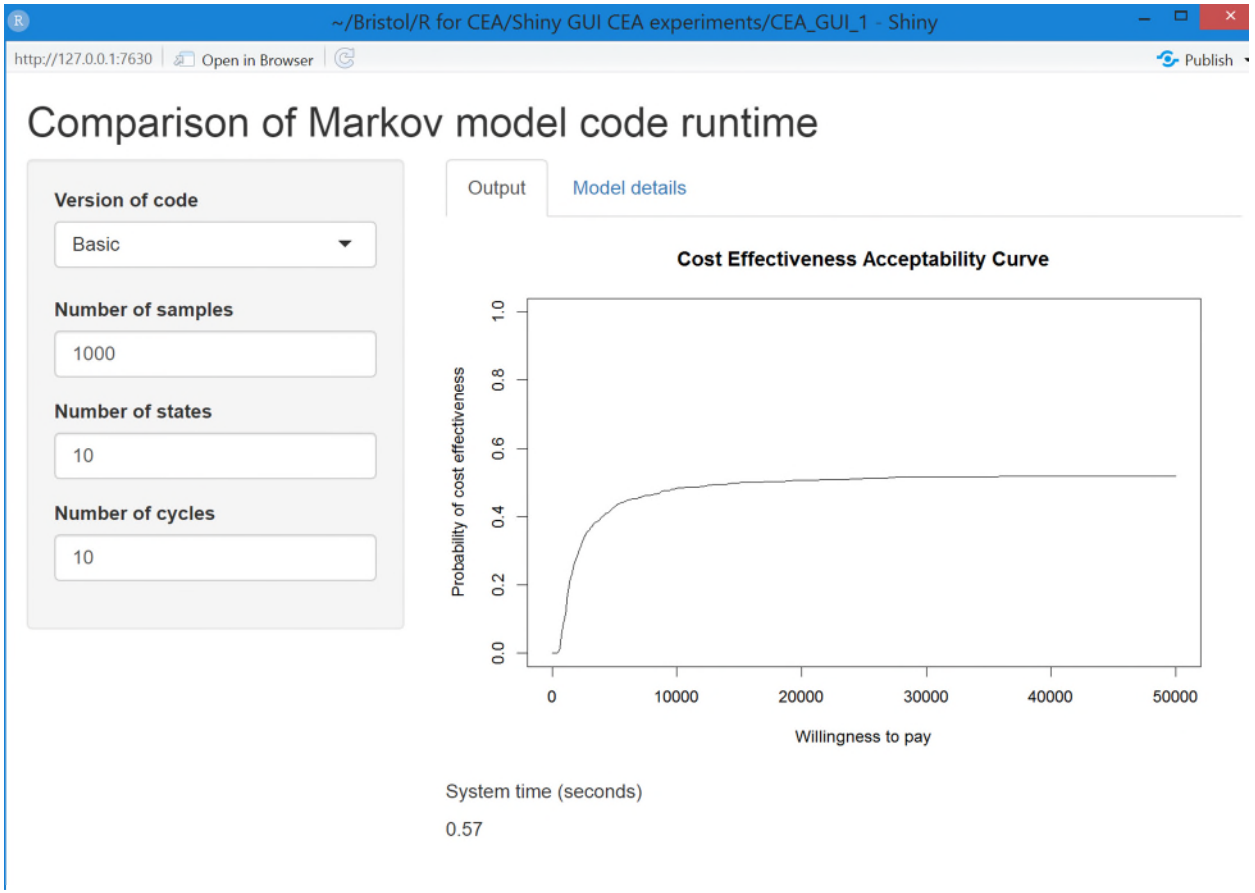
Introduction | Parameters and Analysis | Transition Probability Matrix | Markov Model | Markov Figure

bristol.ac.uk

R for Health Technology Assessment

R version

- The interface was coded in R Shiny (more on this later)

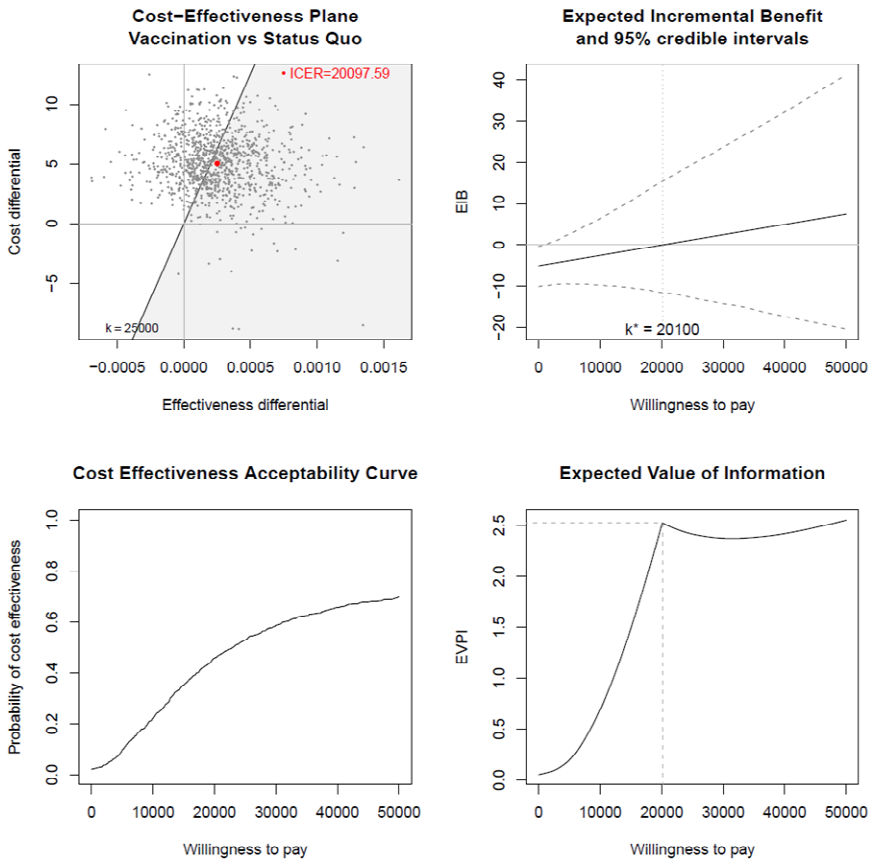


BCEA – Gianluca Baio, UCL

- As noted, the CEAC in the R Shiny version of the model was generated by the BCEA package.
- “Bayesian Cost-Effectiveness Analysis”
- BCEA is a suite of tools for post-processing results of a probabilistic sensitivity analysis
- Does not build CEA models for you. You must give it simulated costs and effects
  - Although these can be from any software, including Excel.

BCEA – Gianluca Baio, UCL

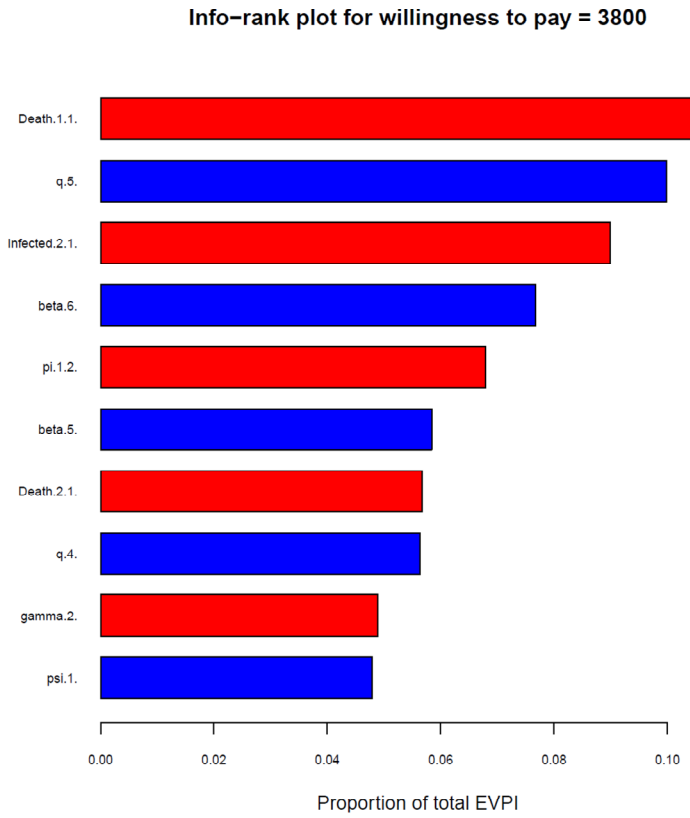
```
> m = bcea(e,c,ref=2,interventions=c("Status Quo","Vaccination"))
> plot(m)
```



BCEA – Gianluca Baio, UCL

- Includes automatic functions to estimate EVPPI.
- And what the proportion of total EVPI…
  - This is a more sophisticated analysis than one-by-one deterministic sensitivity analysis as accounts for probability that a parameter value will change

```
> ir = info.rank(inp$parameters,inp$mat,mm,howManyPars=10)
```



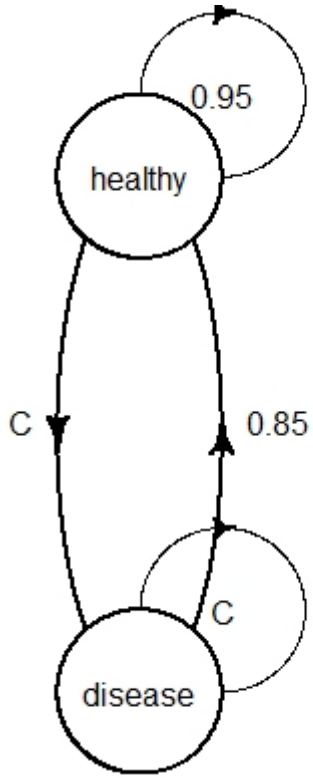
Heemod – created by Kevin Zarca, Urc Eco (France).

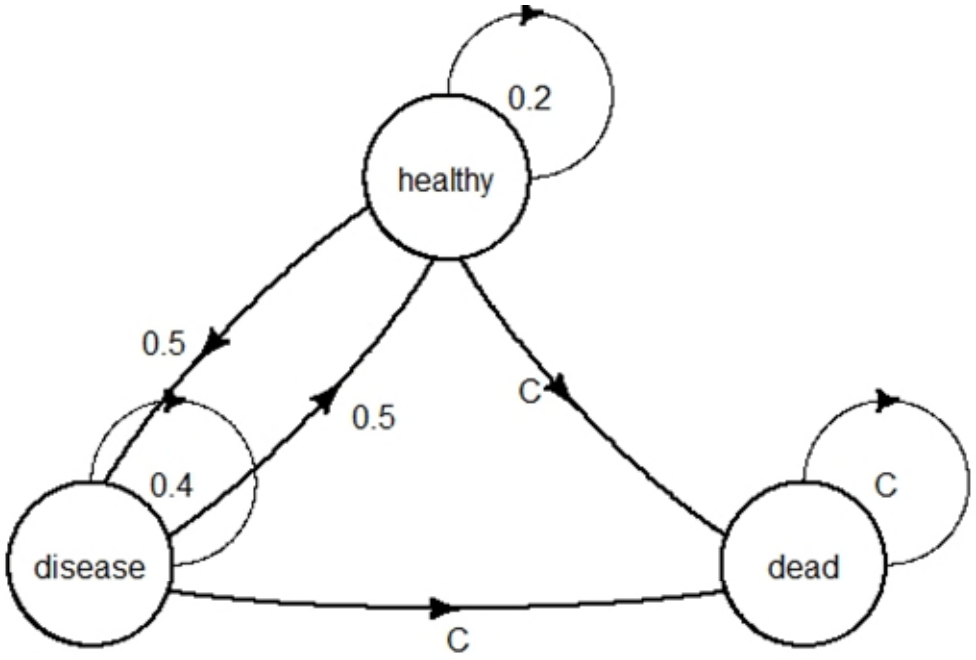
- Easy-to-use package for cohort multistate CEA modelling in R
- Implements modelling and reporting of Briggs textbook Decision Modelling for Health Economic Evaluation textbook.
- Deterministic and probabilistic sensitivity analysis
- Includes semi-Markov and non-homogeneous Markov models
- Has own plotting functions but can link to BCEA if preferred.

- Conversely, I've found it slow and not always flexible enough for complex modelling.
- However, has one particularly useful feature…

Heemod – created by Kevin Zarca, Urc Eco (France).

- Based on user inputs, generates a model diagram
- Not the prettiest, but useful for validation
- Examples based on Nathan Green (Imperial College) code.





R2OpenBUGS – Andrew Gelman, Columbia University

- Run Bayesian MCMC estimation through OpenBUGS from R
  - Similar packages exist for WinBUGS, JAGS, and STAN
- First load the library

library(R2OpenBUGS)

- Then some BUGS model file (e.g. the NICE DSU TSD NMA code)

source("fixed_effects_binary.R")

- Set simulation parameters

*n.chains<-2; num.sims<-10000*n.chains; burn.in<-50000*n.chains*

- Call R2OpenBUGS key function

bugs_object<-bugs(data = bugs_data_recovery, inits = NA, model = fixed_effects_binary…)

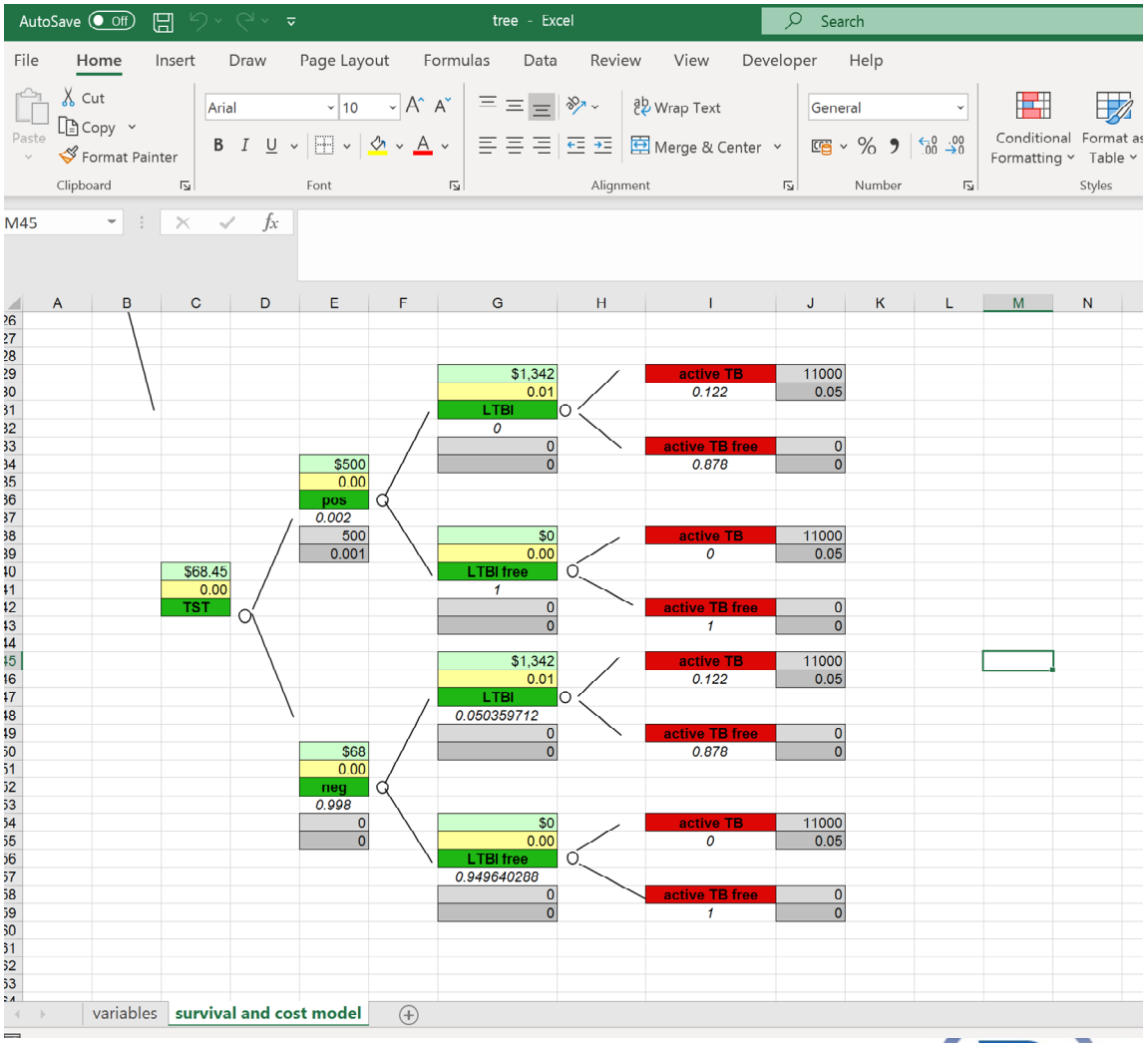
- Then get parameter samples from bugs_object$sims.array or statistics using bugs_object$summary

R2OpenBUGS – network meta-analysis

- I conduct all my NMAs in R.
- Automate formatting of systematic literature review data
  - Using read.xlsx() or read.csv()
- Automate model running (base case, sensitivities, and subgroups)
  - R2OpenBUGS
- Automate formatting of results (results tables, network plots, forest plots, rankograms)
  - Using write.xlsx() to link to Excel
  - Real example of this formatting will be shown

Xlerate – Richard Fitzjohn, Imperial College

- Proof-of-concept package to automatically convert Excel to R.
- Takes as input an Excel decision tree



bristol.ac.uk

Health Technology Assessment

Xlerate – Richard Fitzjohn, Imperial College

- You then specify the range of input and output cells

```
inputs <- xlerate::xlerate_ref( c("D3:D13", "D15:D16", "D18:D21"), sheet = 1, label = list(col = -1))
```

```
outputs <- xlerate::xlerate_ref( c("C40", "E34", "E50", "G29", "G38", "G45", "G54"), sheet = 2, label = list(row = 2))
```

- Using these, the package recursively searches through the Excel sheet and creates equivalent R code.
- R version is much faster and thus useful for sensitivity analyses of value of information analysis.
- Package under development and may be extended to Markov models.
- Code is available on GitHub:

https://github.com/HealthEconomicsHackathon/xlerate#xlerate

- And a blog entry describing the package is here:

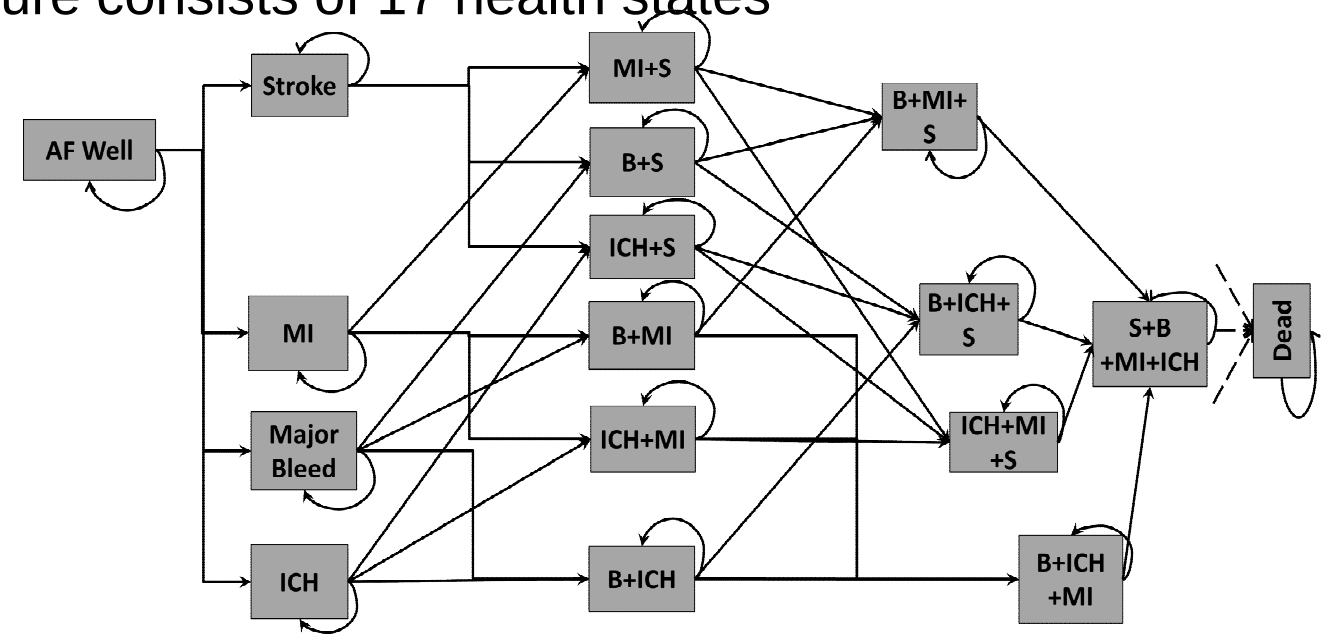
https://reside-ic.github.io/blog/experiments-in-transforming-excel-into-r/

Other packages

- Hesim by Devin Incerti
  - Individual level Markov and semi-Markov continuous-time multistate modelling
  - Specifically designed for CEA modelling
  - Intro available here: https://devinincerti.com/2019/01/01/sim-mstate.html
- Mstate by Hein Putter et al.
  - Continuous-time multistate modelling of panel data
  - Claire Williams paper on CEA modelling using mstate: https://journals.sagepub.com/doi/full/10.1177/0272989X16651869
- Flexsurv – Chris Jackson
- Msm – Chris Jackson
  - Continuous-time Markov modelling of panel data
  - Includes functionality for hidden Markov models

Complex models – DOACs for AF

- The added speed and flexibility of R enables much more complex and realistic models to be implemented
- Key example is directly acting oral anticoagulants (DOACs) for atrial fibrillation (AF).
- Structure consists of 17 health states



S=Ischemic stroke, B=Major extracranial bleed, MI=Myocardial infarction, ICH=Intracranial haemorrhage

TIA and SE are transient events. Patients can switch treatment following events.

Complex models – DOACs for AF

- Structure consists of 17 health states – memory states that overcome Markov limitation
- Duplicated for 6 treatment options: coumarin, 4 DOACs, and no treatment
  - 97 states in total
  - Treatment switching included, plus transient events
- Uses 98 input parameters – including link to competing risks NMA implemented through R2OpenBUGS
  - Fully probabilistic
- Conducted EVPPI analysis using multilevel Monte Carlo modelling
  - SAVI and model regression techniques could not manage the complexity and number of input parameters
- This model will now be demonstrated live…
- Version of code available on GitHub

https://github.com/Bogdasayen/DOACs-AF-Economic-model

GitHub

- Version control key to good programming practice
- GitHub is not just an online repository – it is a tool to ease collaborative working on large projects
- Updates are pulled from the repository, changes pushed back and committed.
- You can duplicate repositories (fork) or experiment with minor duplicates (branch).
- Can assign a team lead to review and accept commits.
- Interfaces easily with RStudio (more on this shortly)

Showcase – RMarkdown

bristol.ac.uk

R for Health Technology Assessment

Showcase – RShiny

- Package and extension to R that allows creation of html/JavaScript web apps.
- These can be hosted on Shiny server and made publicly available
  - For example, the NICE AF guidelines could include a link to the actual model.
  - Huge transparency benefit
- You've now seen two examples of RShiny user interfaces
  - I'm very much a novice but knocked these GUIs up in a couple of hours.
- You may also have seen the SAVI tool for value of information analysis – this is implemented in RShiny.
- Let's take a look at a fully fledged modelling example
- The Innovation and Value Initiative Rheumatoid Arthritis model (IVI-RA)
  - Devin Incerti (Genentech, formerly IVI), also created 'hesim' package. https://innovationandvalueinitiative.shinyapps.io/ivi-ra-expert/

RStudio

- sd

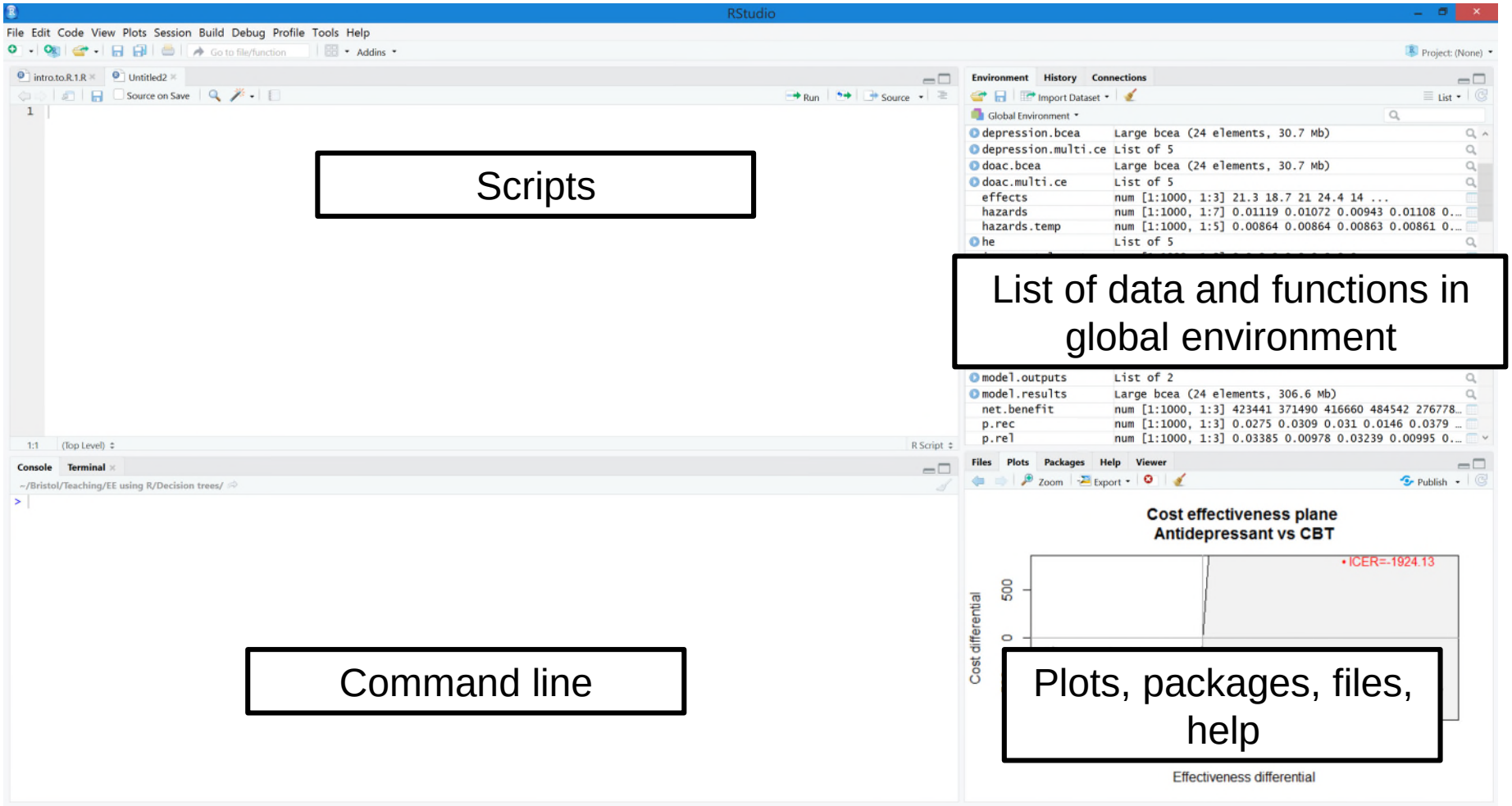
bristol.ac.uk

R for Health Technology Assessment

Introduction to building decision trees in R

- Heard the wonderful things you can do in R.
- Itching to get your hands dirty?
- We'll now talk through the specifics of programming a (probabilistic) decision tree in R.

Rstudio interface

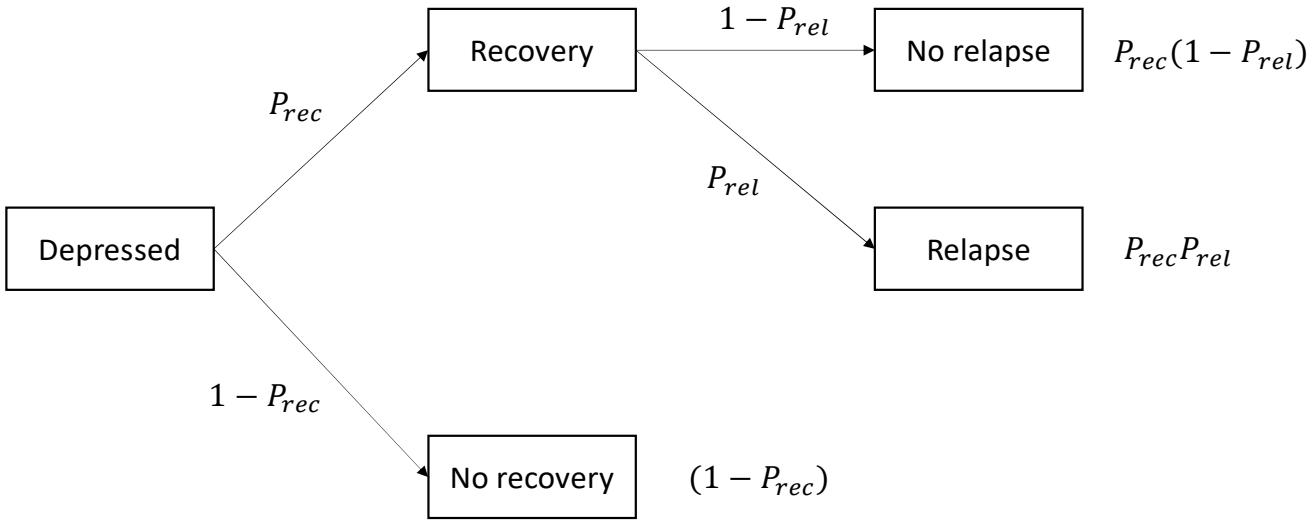


Simple functions – logit and its inverse

```
# Logistic link function
logit <- function(x) {
        return(log(x / (1 - x)))
}
# Inverse of logit
expit<-function(x) {
        return(1 / (1 + exp(-x)))
}
```

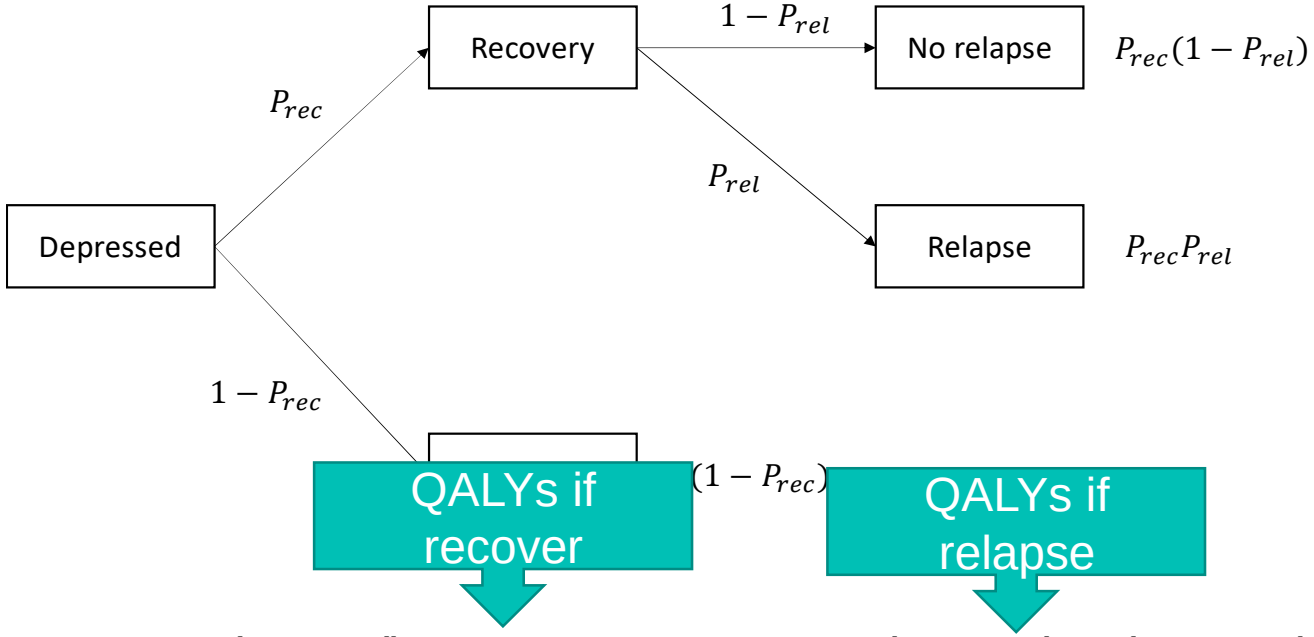
```
> logit(0)
[1] -Inf
> logit(0.5)
[1] 0
> logit(1)
[1] Inf
> expit(0)
[1] 0.5
>
```

Simple decision tree in R



- Consider this simple decision tree with artificial input parameters.
- Probabilities of recovery and relapse for no treatment (option 1), cognitive behavioural therapy (option 2), and antidepressants (option 3).
- This toy model is available on GitHub: https://github.com/Bogdasayen/Depression-toy-decision-tree-in-R

Implementing a decision tree in R



*effects <- p_rec * (1-p_rel) * q_rec + p_rec * p_rel * q_rel + (1-p_rec) * q_norec*

Cost of treatment

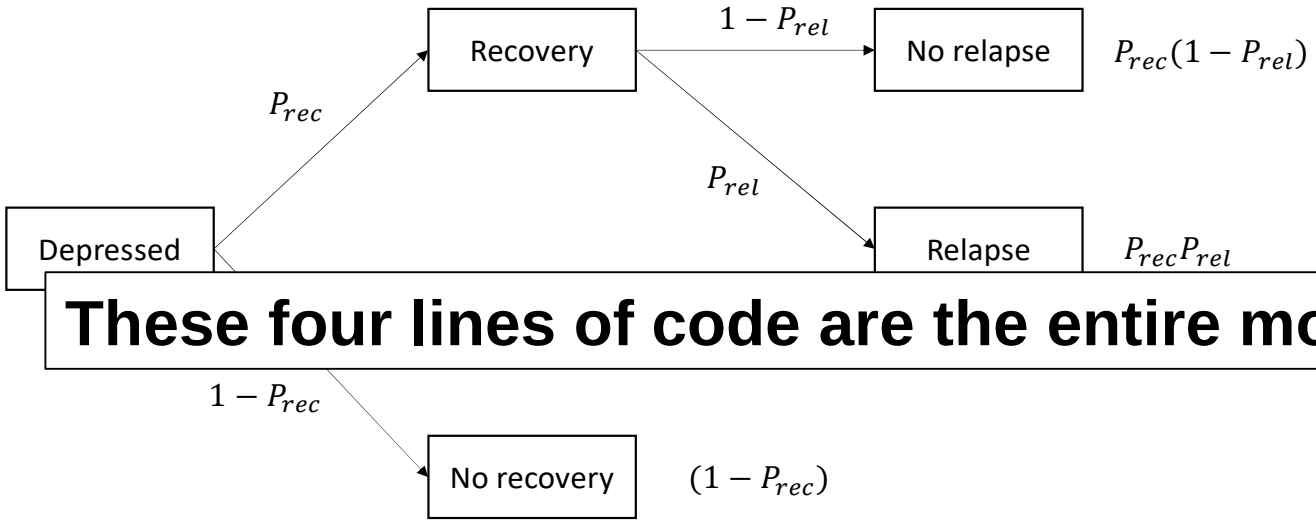
QALYs if not recover

*costs <- c_treat + p_rec * (1-p_rel) * c_rec + p_rec * p_rel * c_rel + (1-p_rec) * c_norec*

*net_benefit <- lambda_target * effects-cost*

incremental_nb <- net_benefit – net_benefit[, 1]

Implementing a decision tree in R



These four lines of code are the entire model!

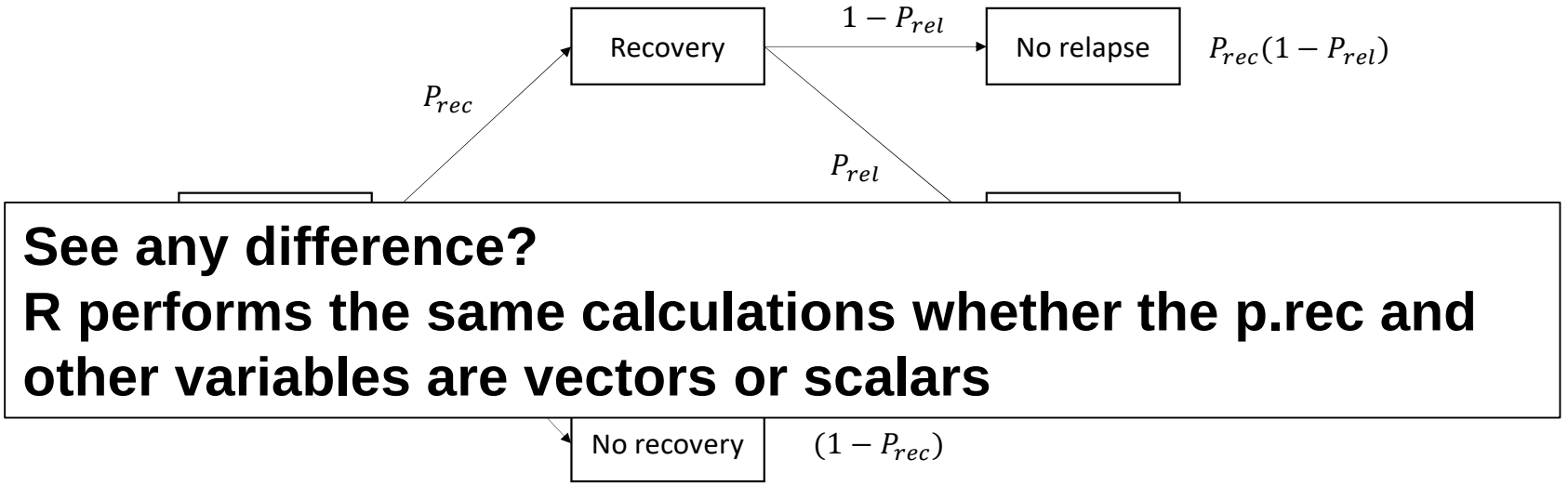
```
effects <- p_rec * (1-p_rel) * q_rec + p_rec * p_rel * q_rel + (1-p_rec) * q_norec
costs <- c_treat + p_rec * (1-p_rel) * c_rec + p_rec * p_rel * c_rel + (1-p_rec) *
c_norec
net_benefit <- lambda_target * effects-costs
incremental_nb <- net_benefit – net_benefit[, 1]
```

And now we make it probabilistic…

bristol.ac.uk

R for Health Technology Assessment

Making it probabilistic (model code)



See any difference?
R performs the same calculations whether the p.rec and other variables are vectors or scalars

*effects <- p_rec * (1-p_rel) * q_rec + p_rec * p_rel * q_rel + (1-p_rec) * q_norec*

*costs <- c_treat + p_rec * (1-p_rel) * c_rec + p_rec * p_rel * c_rel + (1-p_rec) * c_norec*

*net_benefit <- lambda_target * effects-costs*

incremental_nb <- net_benefit – net_benefit[, 1]

bristol.ac.uk

R for Health Technology Assessment

Making it probabilistic (Costs, Utilities)

Outcome	Costs	QALYS
Recovery, no relapse	$C_{rec} = N(\mu = 1000, \sigma = 50)$	$Q_{rec} = N(\mu = 26, \sigma = 2)$
Recovery, relapse	$C_{rel} = N(\mu = 2000, \sigma = 100)$	$Q_{rel} = N(\mu = 23, \sigma = 3)$
No recovery	$C_{no\ rec} = N(\mu = 2500, \sigma = 125)$	$Q_{no\ rec} = N(\mu = 20, \sigma = 4)$

```
# Costs for recovery, relapse, and no recovery over 30 year horizon
c_rec <- rnorm(n = n_samples, mean = 1000, sd = 50)
c_rel <- rnorm(n = n_samples, mean=2000, sd = 100)
c_norec<-rnorm(n = n_samples, mean=2500, sd = 125)

# QALYs for recovery, relapse, and no recovery over 30 year horizon
q_rec <- rnorm(n = n_samples, mean = 26, sd = 2)
q_rel <- rnorm(n = n_samples, mean = 23, sd = 3)
q_norec <- rnorm(n = n_samples, mean = 20, sd = 4)
```

Making it probabilistic (Treatment effects)

- Log odds ratios follow multivariate normal

Recovery: $\begin{pmatrix} lor_{2,rec} \\ lor_{3,rec} \end{pmatrix} \sim MVN\left(\begin{pmatrix} 0.99 \\ 1.33 \end{pmatrix}, \begin{pmatrix} 0.22 & 0.15 \\ 0.15 & 0.20 \end{pmatrix}\right)$

Relapse: $\begin{pmatrix} lor_{2,rel} \\ lor_{3,rel} \end{pmatrix} \sim MVN\left(\begin{pmatrix} -1.48 \\ -0.40 \end{pmatrix}, \begin{pmatrix} 0.14 & 0.05 \\ 0.05 & 0.11 \end{pmatrix}\right)$

- As it's a statistical language, the multivariate normal is implemented simply in R:

```
lor_rec <- mvrnorm(n = n_samples, mu = c(0.99, 1.33),
                   Sigma = matrix(c(0.22, 0.15, 0.15, 0.20), nrow = 2))
lor_rel <- mvrnorm(n = n_samples, mu = c(-1.48, -0.40),
                   Sigma = matrix(c(0.14, 0.05, 0.05, 0.11), nrow = 2))
```

Instead MCMC via R2OpenBUGS

- Can run the NMA each time your un the model
- Or (more likely) load precalculated log odds ratios for recovery (similarly for relapse)

mcmc_recovery <- read.csv(file = "lor.recovery.bugs.csv")

- Can use just the first n_samples of the matrix

lor_rec <- mcmc_recovery[1:n_samples,]

Making it probabilistic (Reference probabilities)

Parameter	No Treatment (Option 1)
P_{rec}	$P_{1,rec} = Beta(\alpha = 6, \beta = 200)$
P_{rel}	$P_{1,rel} = Beta(\alpha = 2, \beta = 100)$

- The beta distribution is another of many implemented in base R.
- Note however the idiosyncratic naming convention of the parameters.
- α is *shape1* and β is *shape2*.

```
p_rec[, 1] <- rbeta(n = n_samples,  shape1 = 6,  shape2 = 200)
p_rel[, 1] <- rbeta(n = n_samples,  shape1 = 2,  shape2 = 100)
```

Making it probabilistic (Comparator probabilities)

Parameter	CBT (Option 2)	Antidepressant (Option 3)
P_{rec}	$P_{2,rec} = expit(\text{logit}(P_{1,rec}) + lor_{2,rec})$	$P_{3,rec} = expit(\text{logit}(P_{1,rec}) + lor_{3,rec})$
P_{rel}	$P_{2,rel} = expit(\text{logit}(P_{1,rel}) + lor_{2,rel})$	$P_{3,rel} = expit(\text{logit}(P_{1,rel}) + lor_{3,rel})$

- We can use a loop over the number of treatments *n.treat*

```
for(i in 2:3)
{
  p_rec[, i] <- expit(logit(p_rec[, 1]) + lor_rec[, i - 1])
  p_rel[, i] <- expit(logit(p_rel[, 1]) + lor_rel[, i - 1])
}
```

Making it probabilistic - vectorise

Parameter	CBT (Option 2)	Antidepressant (Option 3)
P_{rec}	$P_{2,rec} = expit(\text{logit}(P_{1,rec}) + lor_{2,rec})$	$P_{3,rec} = expit(\text{logit}(P_{1,rec}) + lor_{3,rec})$
P_{rel}	$P_{2,rel} = expit(\text{logit}(P_{1,rel}) + lor_{2,rel})$	$P_{3,rel} = expit(\text{logit}(P_{1,rel}) + lor_{3,rel})$

- Or we can vectorise, which is much faster than a loop

p_rec[, c(2:n_treat)] <- expit(logit(p_rec[, 1]) + lor_rec[, c(2:n_treat) - 1])

p_rel[, c(2:n_treat)] <- expit(logit(p_rel[, 1]) + lor_rel[, c(2:n_treat) - 1])

The expit and logit functions work on vectors and matrices.

- Can set n.treat to any number without having to duplicate code.

Formatting results

- Use *paste("string1", "string2")* function for string concatenation
- Use *round(x,digits=3)* for numeric formatting

```
format_results<-function(x, n_digits = 2) {
        paste(round(mean(x), digits = n_digits),
              " (", round(quantile(x, probs = 0.025), digits = n_digits), ",
", round(quantile(x, probs = 0.975), digits = n_digits), ")", sep= "")
}
```

```
> paste("string1", "string2", sep = "+")
[1] "string1+string2"
> round(3.141592, digits = 3)
[1] 3.142
> format_results(c_rec)
[1] "999.46 (907.56, 1094.33)"
>
```

Decision tree results

- Build a results matrix

results_matrix <- matrix(NA, nrow = 6, ncol = n_treat)

- Name the rows and columns

rownames(results_matrix) <- c("Total costs", "Total QALYs", "Incremental costs", "Incremental QALYs", "Net Benefit", "Incremental NB")

colnames(results_matrix) <- t_names

- Then calculate summaries

```
for(i_treat in 1:n_treat)
{
  results_matrix["Total costs", i_treat] <- format_results(x = costs[, i_treat])
  results_matrix["Total QALYs", i_treat] <- format_results(x = effects[, i_treat])
  results_matrix["Incremental costs", i_treat] <- format_results(x = incremental_costs[, i_treat])
  results_matrix["Incremental QALYs", i_treat] <- format_results(x = incremental_effects[, i_treat])
  results_matrix["Net Benefit", i_treat] <- format_results(x = net_benefit[, i_treat])
  results_matrix["Incremental NB", i_treat] <- format_results(x = incremental_nb[, i_treat])
}
```

Exporting the results matrix to Excel

- Export as a csv

write.csv(results_matrix, file = "depression_results.csv")

- Or as an Excel file

library(xlsx)

write.xlsx(results_matrix, file = "depression_results.xlsx", sheetName = "CEA results")

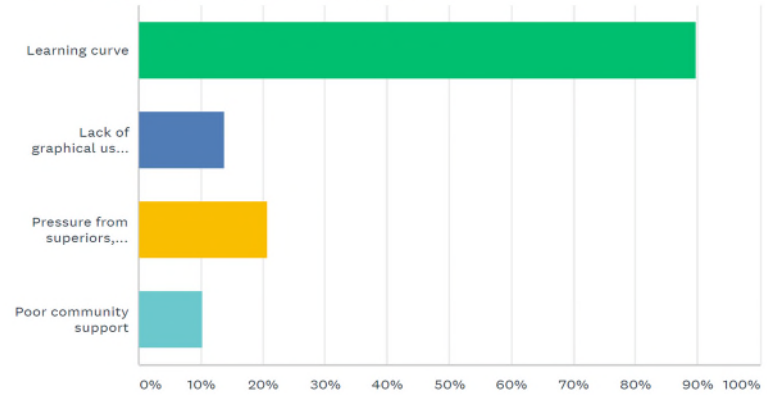
	No treatment	CBT	Antidepressant
Total costs	2458.08 (2216.38, 2692.91)	2678.9 (2424.37, 2937.03)	2366.58 (2087.97, 2621.49)
Total QALYs	20.09 (12.87, 27.59)	20.41 (13.54, 27.56)	20.59 (14.02, 27.52)
Net Benefit	399358.81 (254931.73, 549117.1)	405521.49 (267937.64, 548404.61)	409355.48 (277952.91, 548049.21)
Incremental NB	0 (0, 0)	6162.68 (-1978.38, 26095.58)	9996.67 (-2660.86, 36001.2)

What are the barriers to using R

- ISPOR and R for HTA surveys

Learning curve is main barrier to adopting R

SurveyMonkey Analyze - R for CEA pre-workshop survey

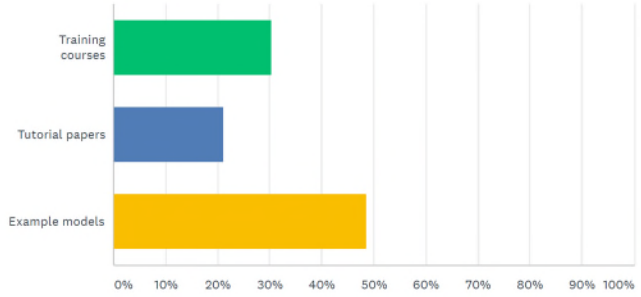


ANSWER CHOICES	RESPONSES
Learning curve	89.66%
Lack of graphical user interface	13.79%
Pressure from superiors, clients or assessors	20.69%
Poor community support	10.34%

Total Respondents: 29

What would be most helpful to ease transition to R?

Answered: 33 Skipped: 0



ANSWER CHOICES	RESPONSES
Training courses	30.30%
Tutorial papers	21.21%
Example models	48.48%
TOTAL	

bristol.ac.uk

R for Health Technology Assessment

Training opportunities

- EE modelling in R (Bristol)
- R for HTA training day
- Erik in Netherlands DES
- Tutorial papers and websites
- Intro to R (various, including DARTH)

Invitation to panel discussion

- 30th June 2020, London.