Learnings from the design of a pathway model in R

Dawn Lee and Darren Burns

Peninsula Technology Assessment Group (PenTAG)
University of Exeter, St Luke's Campus, Exeter, EX1 2LU
PenTAG Health Technology Assessment | ESMI | University of Exeter
pentag@exeter.ac.uk

Why did we want an independent open-source reference model?

- Recommendations and inputs inconsistent in previous TAs
- None of the previous TAs considered subsequent therapy appropriately
- Treatments have been recommended which in hindsight were not cost-effective
- The process for NICE and EAGs was inefficient

The NICE RCC Pathways Pilot Open-Source Model

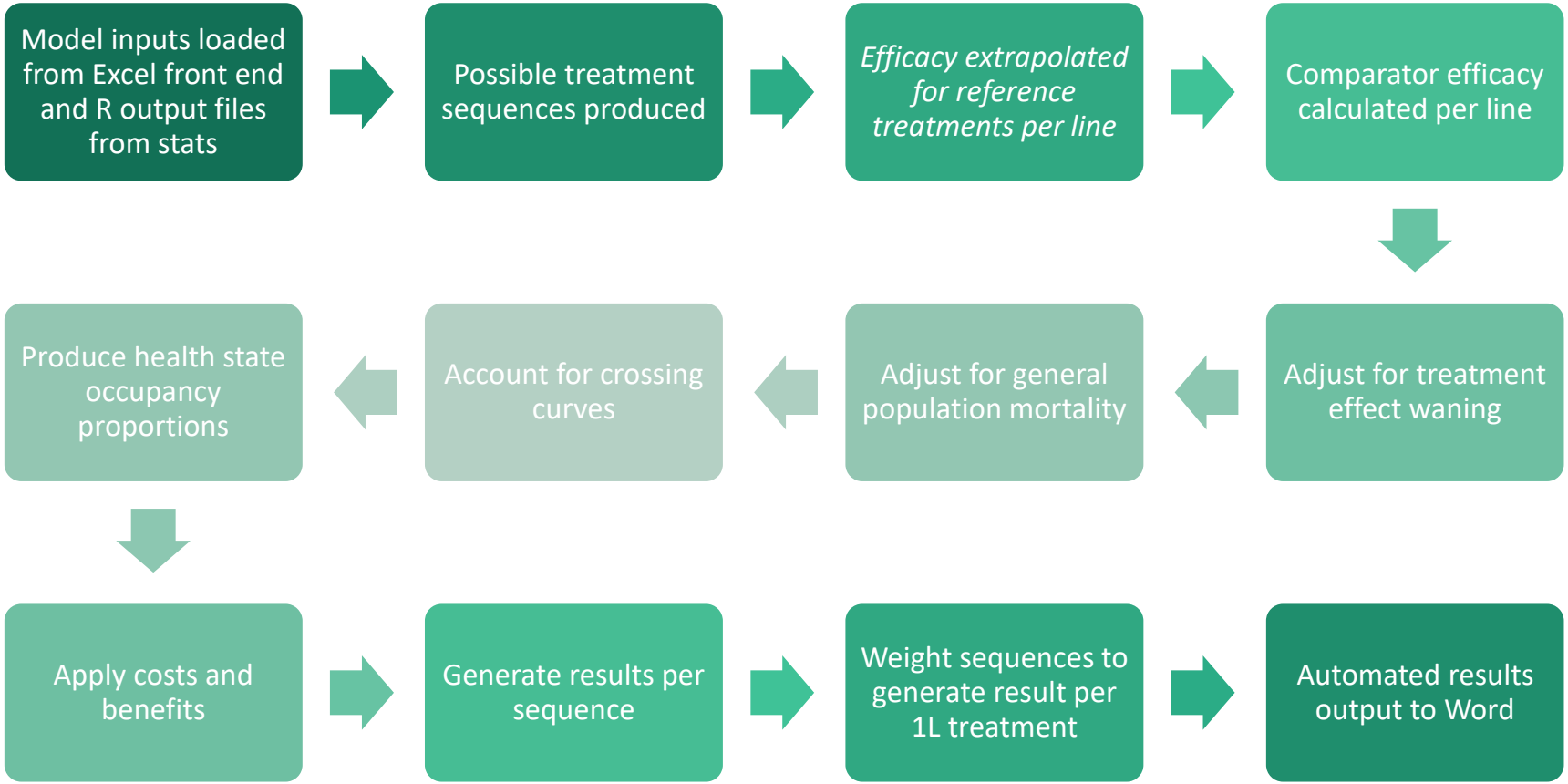
- Developed on behalf of NICE to support a live appraisal (nivo+cabo) – tight timelines!
- Ability to look at sequences for future initiatives
- PartSA and state transition structures
- Time varying hazard-ratios and hazards
- Data provided by intervention company and comparators ranging from time to event data inputs to aggregate level data only
- Use of RWE for baseline risk
- 744 possible sequences, up to 14,000 states per sequence, 2080 cycles, 3 populations
- No template models to follow

- **Given complexity and computational requirements R was the logical choice**

Lee, D., Burns, D. & Wilson, E. NICE's Pathways Pilot: Pursuing Good Decision Making in Difficult Circumstances. *PharmacoEconomics Open* (2024). https://doi.org/10.1007/s41669-024-00490-x

https://github.com/nice-digital/NICE-model-repo

How does it work?



R, sparse matrices and connecting evidence with outcomes

Using Matrix(), list() and Reduce() and a little-known R trick to build an efficient model engine, even in the face of more than 14,000 health states for 2080 cycles

Prepared by: Darren Burns

Delta Hat

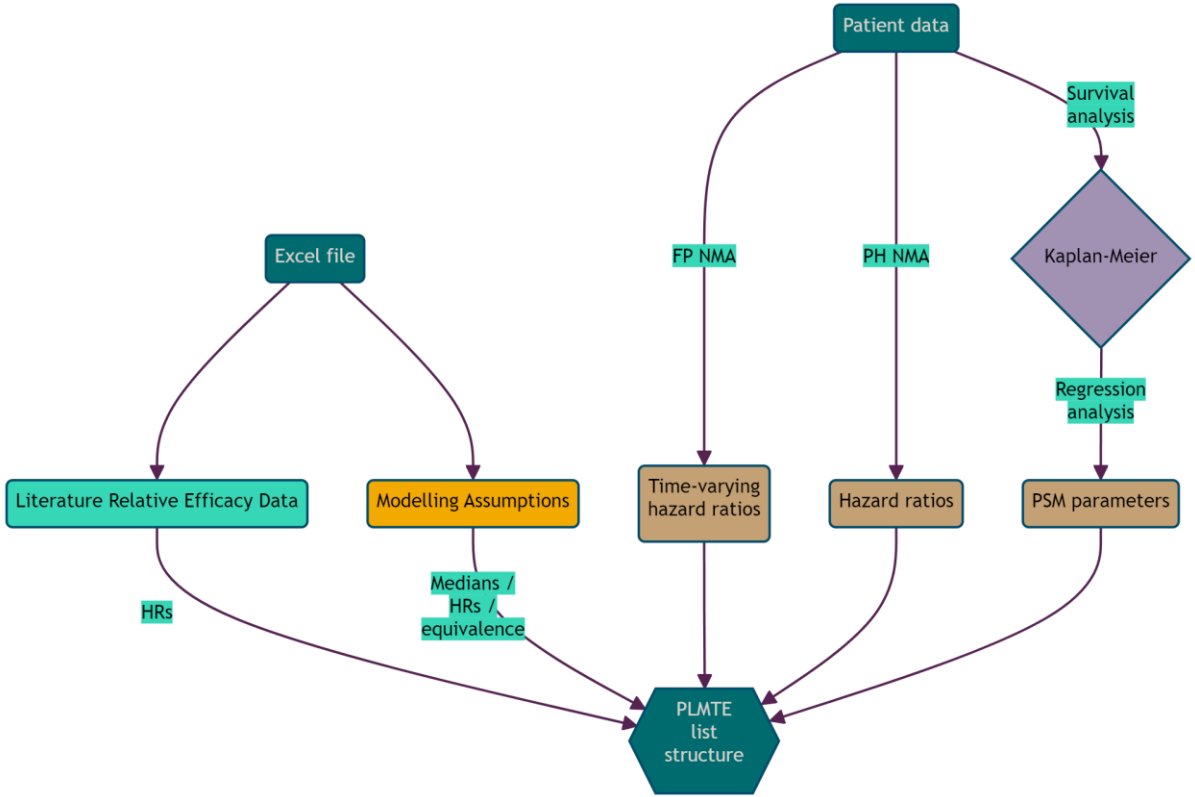
Generalized CE model extrapolation in PATT: information collation

Myriad processes for generating extrapolations for each PLMTE, considering:

- Direct extrapolations at any PLMTE
- Recursive application of relative efficacy across PLMTEs

Therefore, structure requires potential to link *any* PLMTE with *any* other PLMTE for any degree of separation (DoS), *without* changing any R code.

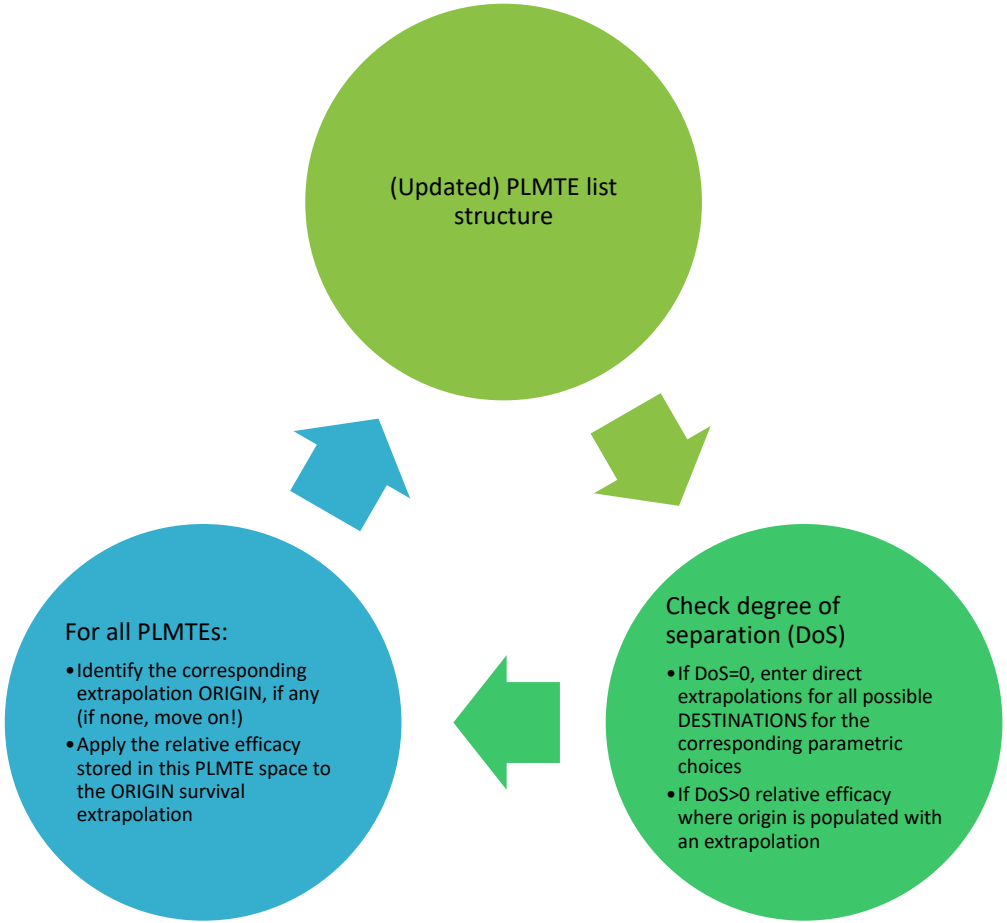
Issues: Excel is still ESSENTIAL for storing and managing inputs! Shiny interface could help and *could* provide essential additional validation which we had to do manually, but we had to move fast and change the entire UI repeatedly (minutes not hours, hours not days).



Key: P: Population; T: Trial; M: Molecule; T: Trial; E: Endpoint; HR: Hazard ratio; FP: Fractional polynomial; PH: Proportional hazards; PSM, parametric survival analysis

Generalized CE model propagation in PATT:
`f_releff_PropNetwork`

```
extraps <- Reduce(
  x          = 1:dos,
  init       = network,
  accumulate = FALSE,
  f          = function(prev_network, deg_of_sep) {
    if (deg_of_sep == 1) {
      prev_network_this <- Reduce(
        x          = 1:nrow(ref_curves),
        init       = prev_network,
        accumulate = FALSE,
        f          = function(prev, ref_curve_row) {
          ➢ Go row by row down reference curve table
          ➢ Assign ORIG = DEST for each ref curve
          ➢ User settings and parameters → extrapolation
          ➢ Add e.g. stopping rules etc
        }
        return(prev_network_this)
    } else {
      prev_network_this <- Reduce(
        x          = 1:nrow(releff),
        init       = prev_network,
        accumulate = FALSE,
        f          = function(prev, releff_row) {
          ➢ Go row by row down relative efficacy table
          ➢ For each DEST, check in ORIG for an extrap
          ➢ If ORIG ☑ + DEST ☒ + releff ☑, then ORIG → releff
            = DEST
          ➢ If not, do nothing
        }
        return(prev_network_this)
    }
})
```



Key: P: Population; T: Trial; M: Molecule; T: Trial; E: Endpoint; HR: Hazard ratio; FP: Fractional polynomial; PH: Proportional hazards

From hundreds of extrapolations to a Markov trace (1)

Starting point: every possible extrapolation from input set and layers of relative efficacy

We need to model every possible treatment pathway as efficiently as possible

We know that all patients start in state 1 (on 1L treatment), so we know the initial population vector **p** = $[P_1 \quad \cdots \quad p_D]$, where $p_1 = 1, ; p_{2\ldots D} = 0$ at baseline

We use tunnels. Lots and lots of tunnels…

Patients flow through successive lines of therapy, and we have the extrapolations

We can apply $p_t = 1 - \frac{s(t)}{s(t-1)}$ as state exit probabilities, and apply some assumptions to get all required TPs

We can put the TPs into matrix **M.** Note that for 5-line PATT, **M has over 14,000 health states (rows & cols)…M is very large!**

Factors steering approach:

1. R is very good at matrix algebra, and is fast when carefully vectorised
2. The `Matrix` package is designed specifically for efficiency and sparse matrix multiplication
3. There is an R "trick" which performs replace-in-place in RAM (i.e., `M[coord[,1], coord[,2]] <- coord[,3]`), **drastically** (orders of magnitude) improving performance, especially with 14,000+ square matrices!
4. Only 2 rows of **M** ever change once compiled (the first two)

Delta Hat

From hundreds of extrapolations to a Markov trace (2)

Now we have a (very large) transition probability matrix, but it's a bit fiddly!

`p%*%M` is at least 85% 0 × 0, so both **p** and **M** are **sparse**. `p%*%M` is a VERY bad idea in base R, as is `M[1,1] <- p_11; M[1,2] <- p_12`. Under the hood R copies the WHOLE matrix every time you do this i.e., extremely slow ☹

M has some key features:

- **Top two rows change every cycle**, but the **rest of the matrix is static**
- Rows 3+ of **M** are block-diagonal but offset by 1 column starting at cell 3,4. Blocks go in both directions like DT nodes!

Solution: Compiler function, updater function, extrapolator function

Compiler populates sparse matrix M using the PLMTE list (for cycle 0), then splits off the top two rows into a smaller matrix **M1** and the "rest" **M2**

Updater function generates a list of **M1** each model cycle

Extrapolator uses Reduce() to repeatedly perform **p** = **p1M1** + **p2M2** (after splitting **p**)

Factors steering approach:

1. R is very good at matrix algebra, and is fast when carefully vectorised
2. The `Matrix` package is designed specifically for efficiency and sparse matrix multiplication
3. There is an R "trick" which performs replace-in-place in RAM (i.e., `M[coord[,1], coord[,2]] <- coord[,3]`), **drastically** (orders of magnitude) improving performance, especially with 14,000+ square matrices!
4. Only 2 rows of **M** ever change once compiled (the first two)

Delta Hat

Is the model everything it could be?

Regularly updated?

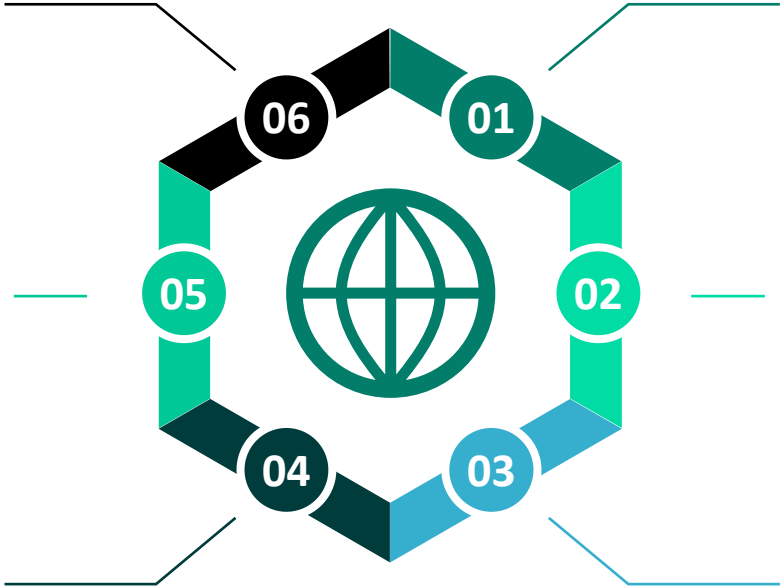
- There is no resource in place to maintain the model
- Unclear how / if future appraisals might use the model

Maintained?

- There is no resource in place to maintain the model now it is live (yet)

Flexible?

- Model created allows addition / removal of treatments, up to 4 lines of active treatment, PartSA and state transition structures, evaluation of treatments or sequences
- BUT there are other functionalities which would be useful to increase flexibility (flexible survival analysis, different utility and resource use application, allowing use of TPP, more generic appearance etc…)



Accessible?

- Code supplied with video explanations, written instructions and calls to walk through for manufacturers
- Shiny-front end in the original analysis plan was not funded
- Manufacturers struggled to engage with a model in R
- Developers struggled with version control in Git

Transparent?

- Considerably more detail available than for a usual STA model
- Intermediate calculations fully presented
- Report tables and figures generated from the code
- Full documentation for all functions not available but generally well commented

Quality controlled?

- Quality control conducted internally and by the DSU
- DSU found no major errors
- Unit testing not in place for the majority of functions

Where next?

HEverse: Forking package structure with resources packages

- Each individual model used for a HTA could be rolled into its own R package (contained within a meta-package) that takes resources from previous models during construction.
- For the example, if we had a meta-package called NICE, we would have packages like this:
  - NICEResources – common functions for modelling, plus NICE methods guidance on adapting and building one's own functions specific to a NICE HTA model
  - NICETATemplates – a package containing functions which generate general folder structures, layouts etc for modelling
  - NICEReports – a package containing templates of the rmarkdown/bookdown templates to be used for automated reporting.
  - NICETAXXX – CE model for NICE TA XXX
  - NICETAXXX – CE model for NICE TA XXX

↘ **Next step to get there:** joint application for grant funding with Bristol, Sheffield, York and UCL including template gold-standard model implementations in a set of key disease areas. **Get involved!**