Excel Front-End, R Back-End: A Gateway To Faster Individual Patient Simulations and Probabilistic Sensitivity Analyses in Cost-Effectiveness Models for HTA

MSR101

Clark LJ,[1] Mclean A,[2] van Hest N[2]

[1]Costello Medical, Manchester, UK; [2]Costello Medical, Bristol, UK

Objective

To adapt an individual patient simulation (IPS) Excel cost-effectiveness model (CEM), retaining the familiar Excel interface while running the model simulations in R, to improve model run speed using R's ability to perform element-wise operations on vectors and using multiple computing cores for iterative calculations, improving the user-experience of the CEM.

Background

- CEMs are commonly developed in Microsoft Excel® for health technology assessments (HTAs) due to familiarity, usability and perceived greater transparency.
- However, the computing speed of Excel is limited as it cannot perform element-wise operations on vectors or conduct parallel calculations using multiple computing cores.
- Therefore, Excel can be slow for complex models, especially IPS which track events for each patient over time, and probabilistic sensitivity analyses (PSA) which require hundreds or thousands of iterations to parameterise model uncertainty.

Methods

- Using four surrogate endpoints, the CEM modelled eight clinical events for two treatment arms for 1,000 patients.
- The Excel CEM was adapted to run the IPS in R while retaining the Excel interface.[1] Visual basic for applications (VBA) was used to export Excel inputs to R, where the model simulations were run, and results were imported to Excel for calculations and interpretation (**Figure 1**).
- VBA code originally used for the CEM was translated to iterative (per patient) R code, before vectorising (all patients as one matrix) where possible.
- To enhance computational efficiency, most sensitivity analyses (seed convergence, deterministic sensitivity analysis [DSA] and PSA) simulations in R were parallelised using the parallel package, enabling distribution of tasks across multiple processor cores (7 of 8 cores utilised in the reported analyses).
- Run time for the deterministic IPS was measured for one scenario on the same Windows machine, as an average of five runs for the deterministic IPS, and one run for the sensitivity analyses. Run times of model components for the deterministic IPS and the PSA were also measured.

Results

Overall Run Time (Figure 2)

- Despite the run time for the R deterministic analysis (78.37 s) being slightly longer than the VBA deterministic analysis (52.68 s), there were large improvements for run time in all sensitivity analyses, the majority of which used parallel computing.

Component Run Time (Figure 3)

- Run times of the deterministic IPS showed that a substantial percentage of run time, 28.77%, was spent exporting data, loading R, and loading the packages and data required to run the model. This was a much smaller percentage of the run time for the PSA (1.44%).
- As expected, the model simulations for the PSA took the greatest percentage of run time (86.56%, 72.88 minutes). However, a substantial percentage of the run time (10.90%, 9.18 minutes) was spent importing results into Excel and performing the calculations required to display the results.
- Although the establishment of the parallelisation process took some time (1.11%, 0.93 minutes), this still saved a substantial amount of time in the PSA, as the R PSA with parallelisation (1.40 hours) was 5.02 hours faster than the R PSA without parallelisation (6.42 hours).

Conclusion

Although the R model was marginally slower than the VBA model for the deterministic analysis, R demonstrated substantial efficiencies over Excel for the sensitivity analyses when utilising parallel computing.

This shows R's potential to improve CEM run-time while retaining a familiar Excel interface, which may increase the acceptance of programming languages other than VBA for CEM development by HTA agencies.

The inefficiencies of data transfer between R and Excel, and the time taken for Excel to perform calculations based on the results, may motivate the development of a CEM which runs entirely in R, or a different programming language, to increase speed even further, particularly for complex models.

FIGURE 1

Model architecture

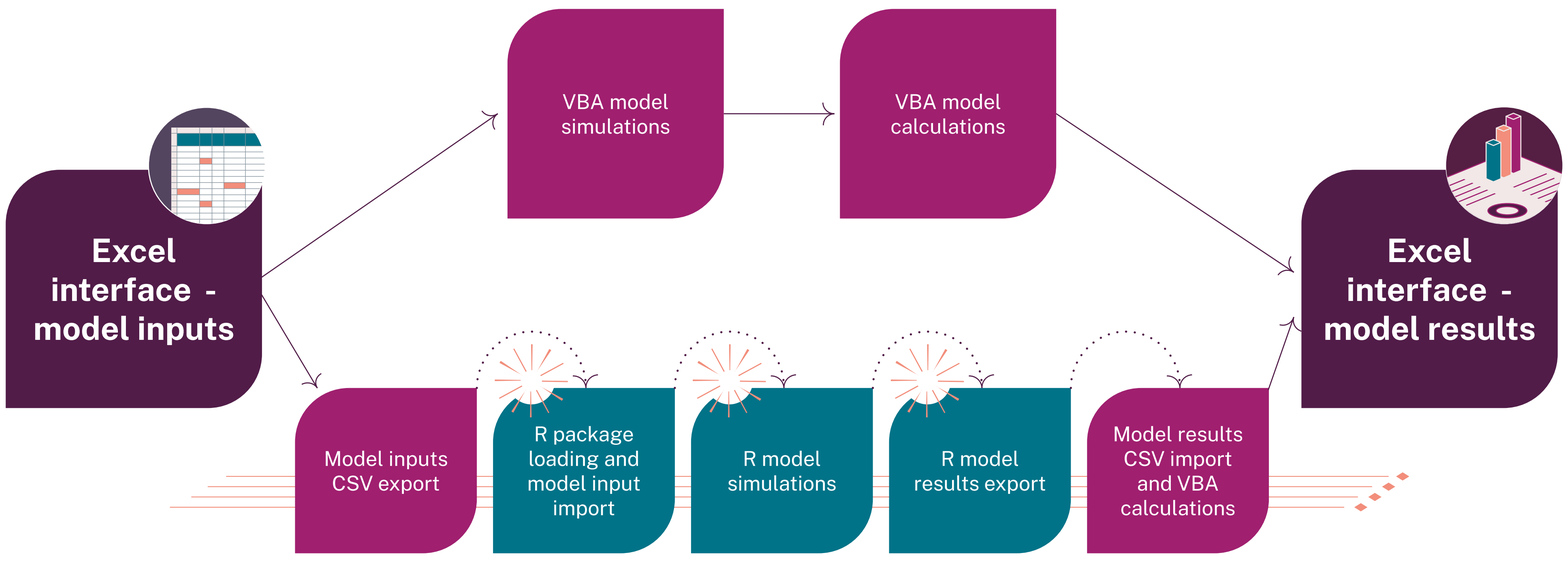
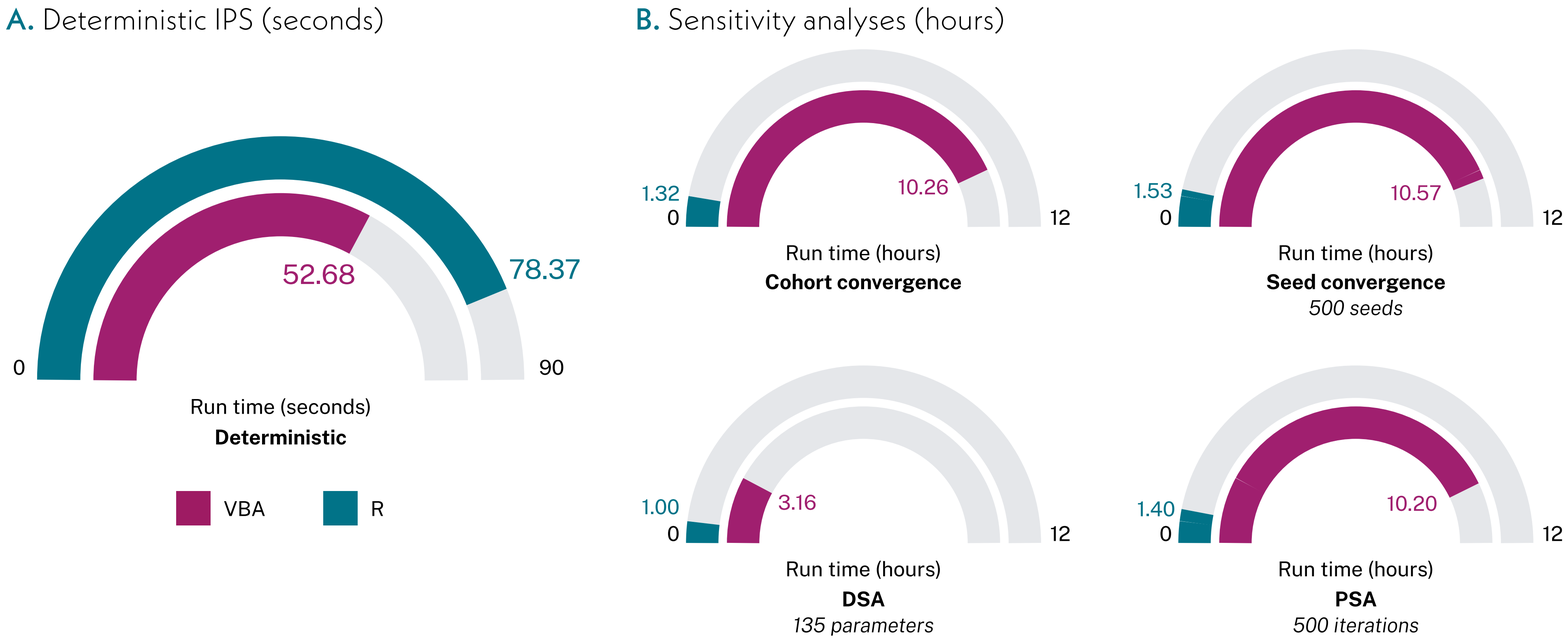


FIGURE 2

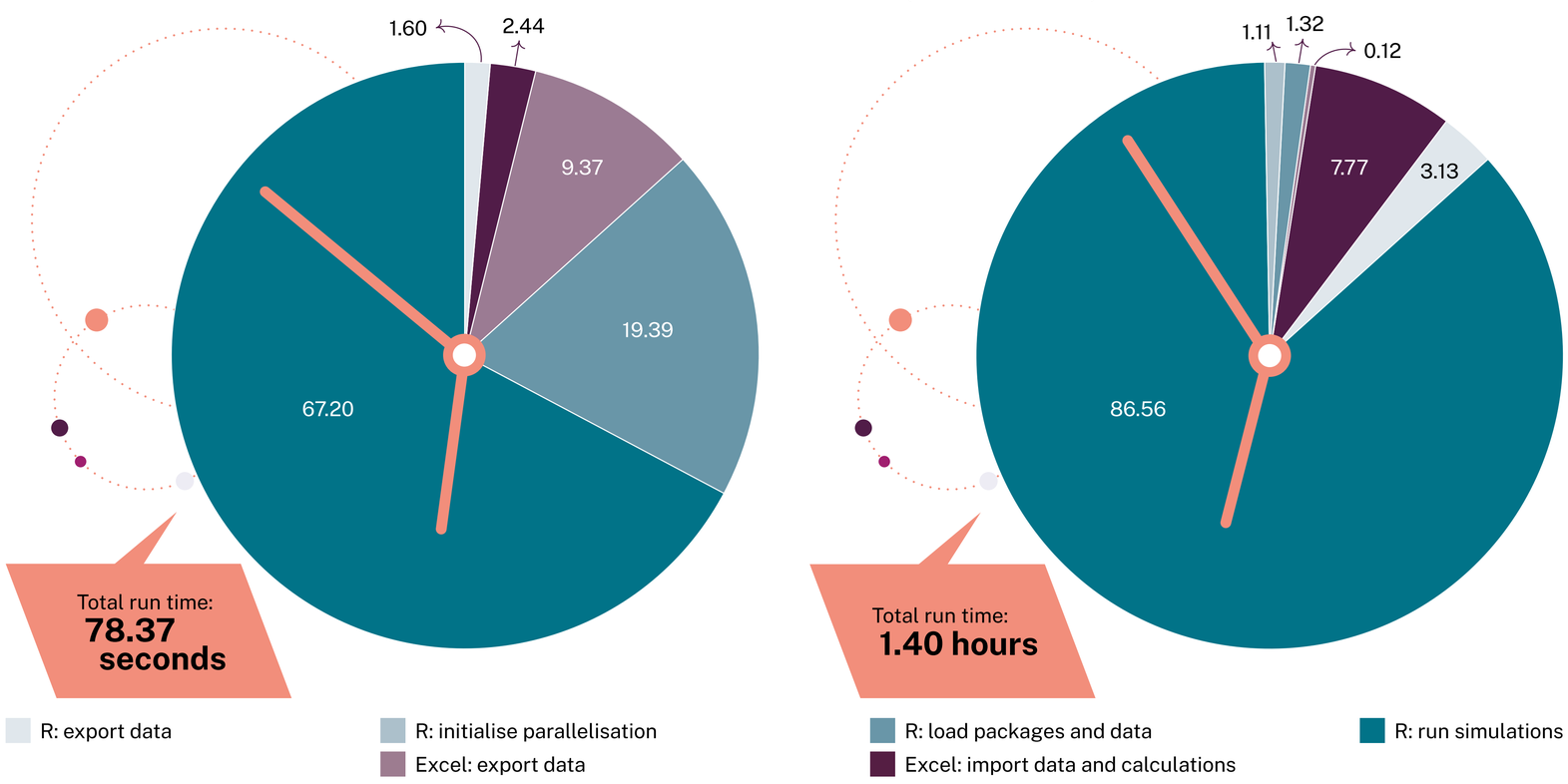
Run time comparison for the VBA and R models



All analyses run with 1,000 patients for a two-treatment comparison. In R, the cohort convergence model structure meant the simulation was run once for all 1,000 patients, with cumulative results recorded sequentially, instead of rerunning the simulation separately for each patient count as was done in VBA.

FIGURE 3

Run time comparison of different components of the R model (% of total run time)



All analyses run with 1,000 patients for a two-treatment comparison.

Abbreviations: CEM: cost-effectiveness model; **CSV:** comma seperated value; **DSA:** deterministic sensitivity analysis; **HTA:** health technology assessment; **IPS:** individual patient simulation; **PSA:** probabilistic sensitivity analysis **VBA:** visual basic for applications.

References: [1]R Core Team (2025). R: A Language and Environment for Statistical Computing. Available at: https://www.R-project.org/. [Last accessed 12 Sep 25].

Acknowledgements: The authors thank Martina Tasende, Costello Medical, for graphic design assistance. We also thank Tristan Curteis and Andrei Karlsson for their contributions and Alex Porteus for their review and editorial assistance in the preparation of this poster.

Presented at ISPOR Europe 2025 | Glasgow, UK | 9–12 November