# BOOSTING A MODELLER'S ARMOURY:
# PURSUING REALISTIC AND EFFICIENT COMPETING RISK HEALTH ECONOMIC MODELS

Stephen Kay

Model Outcomes Ltd, Cheshire, UK

## Objectives

Competing risks (CRs) are endemic in health economic models (HEMs). Techniques for estimation, using cause specific hazard models (CSHMs) are well understood. However, many teaching aids use the same parametric family to model each separate risk. This should never be assumed, particularly when the competing events are disparate such as progression, remission or side effect induced treatment switching. Visualisation and diagnostic/fit tools can be crucial in choosing between parametric forms.

Combining different parametric families together to make predictions requires flexible analytics and programming. This poster seeks to provide useful background information for the less technically inclined reader and more specific advice to HE analysts/programmers engaged in competing risk analysis.

## Methods and Underlying Principles

Hypothetical results of a randomised clinical trial, RCT, were generated for 4000 patients using Monte Carlo sampling techniques. All patients started in the Stable state and from it could remain there or move to either Progression or Adverse Events states or drop out (censored). Progression and Adverse Events are the competing events. Baseline patient characteristics that were programmed to affect transition probabilities were treatment (control or active), sex and age.

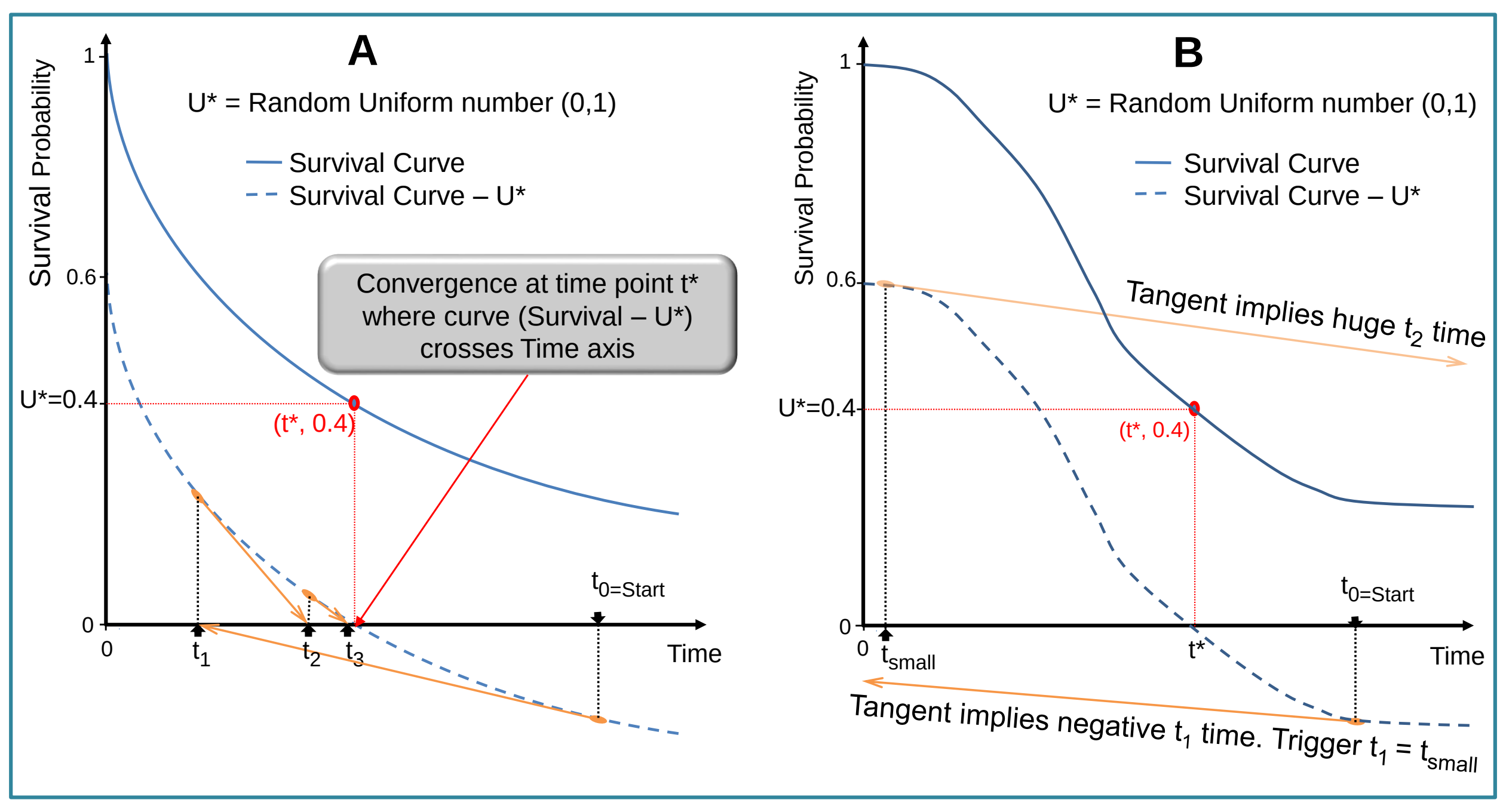
The “true” population CSHM for moving from state Stable to Progression was generated from a Weibull distribution with the scale parameter linked to all three patient characteristics , whilst the shape parameter was influenced by treatment alone. For moving from Stable to Adverse Events, the “true” population CSHM was generated from a Log-logistic model where only the scale parameter was influenced by the explanatory variables treatment and sex. The only difference between a “standard” parametric survival model and one that is “cause specific” is that in the “cause specific” when a competing event occurs it is treated as censored at time of occurrence.

The principles for sampling a “failure” time and which event occurs at failure are straightforward. The survival probability for remaining in the initial state at any time point is simply calculated as the multiplication of all the relevant individual cause-specific survival probabilities that arise from it. Once such a “combined survival” curve has been assembled for many consecutive time points a failure time can be generated using the popular general inverse-transform method: Simply sample a number between 0 and 1 from the uniform distribution and find the time point that produces a survival probability that is very close to it.

To estimate which competing event occurs at this failure time point, sample from the multinomial distribution (or the binomial if there only two competing events). The probabilities to enter into this multinomial/binomial distribution for each competing event are based on the relative size of their calculated hazard rates at the failure time.

Unfortunately, the application of all the above is often far from simple. Usually there is no analytical (closed form) solution to finding a time point from the combined survival curve for any inputted survival probability However, such a solution can be found using iterative methods that search for solutions within a small tolerance. The most widely known is Newton-Raphson (NR) and a visual demonstration of the process working successfully is shown in Figure 1A

**Figure 1 Demonstration of Successful (A) and Unsuccessful (B) Convergence of Newton – Raphson Method For Generating a Random Survival Time**


A
U* = Random Uniform number (0,1)
Survival Curve
Survival Curve – U*
Survival Probability
1
0.6
U*=0.4
0
Convergence at time point t* where curve (Survival – U*) crosses Time axis
(t*, 0.4)
t0=Start
0
t1
t2
t3
Time
B
U* = Random Uniform number (0,1)
Survival Curve
Survival Curve – U*
Survival Probability
1
0.6
U*=0.4
0
Tangent implies huge t2 time
(t*, 0.4)
t0=Start
0
tsmall
t*
Time
Tangent implies negative t1 time. Trigger t1 = tsmall

Within Figure 1A, the uniform random number between 0 and 1 that is drawn is represented by U* and equals 0.4. This value is subtracted from the survival curve – moving it down to the dashed curve. What the NR method does is to find the root (where it crossed the time axis) of this shifted curve (which by inspecting the graph is the solution to the problem). It finds this point by starting at time point $t_0$ (inputted by the user – any reasonable value). It moves to the shifted curve value at this time point and calculates its tangent line – in particular it calculates where this tangent line cuts the time axis (time point $t_1$). It then simply repeats this process and from the movements shown clearly converges to the correct result.

However the process can break down as shown in Figure 1B. Here the initial tangent line is very shallow implying a negative time point crossing the time axis. Clearly this is infeasible. A potential solution and one that sometimes works is to stipulate a small positive time point that is chosen for the next round ($t_{small}$ ) when this occurs. However if that too has a shallow gradient then it generates a huge next round time point and an endless cycle is formed. This problem is caused by the reverse sigmoidal (S) shape corresponding to many survival curves.

One possible solution is to employ a one-to-one transformation to this survival curve that straightens out much of the curvature. Then simply apply the same transformation to the drawn uniform random number and follow the procedures just outlined. An obvious candidate is the Cumulative Hazard, H(t), curve which is defined as H(t) = –log( S(t) ). In fact for many distributions H(t) has a less complex definition than S(t) and similarly for the definition of its derivative (the tangent slope = hazard rate) saving computing time.

When the NR method breaks down, an algorithm that can guarantee convergence is Brent’s method which combines root bracketing, bisection, and inverse quadratic interpolation techniques. Its drawback is that it is relatively slow.

## Methods and Underlying Principles (continued)

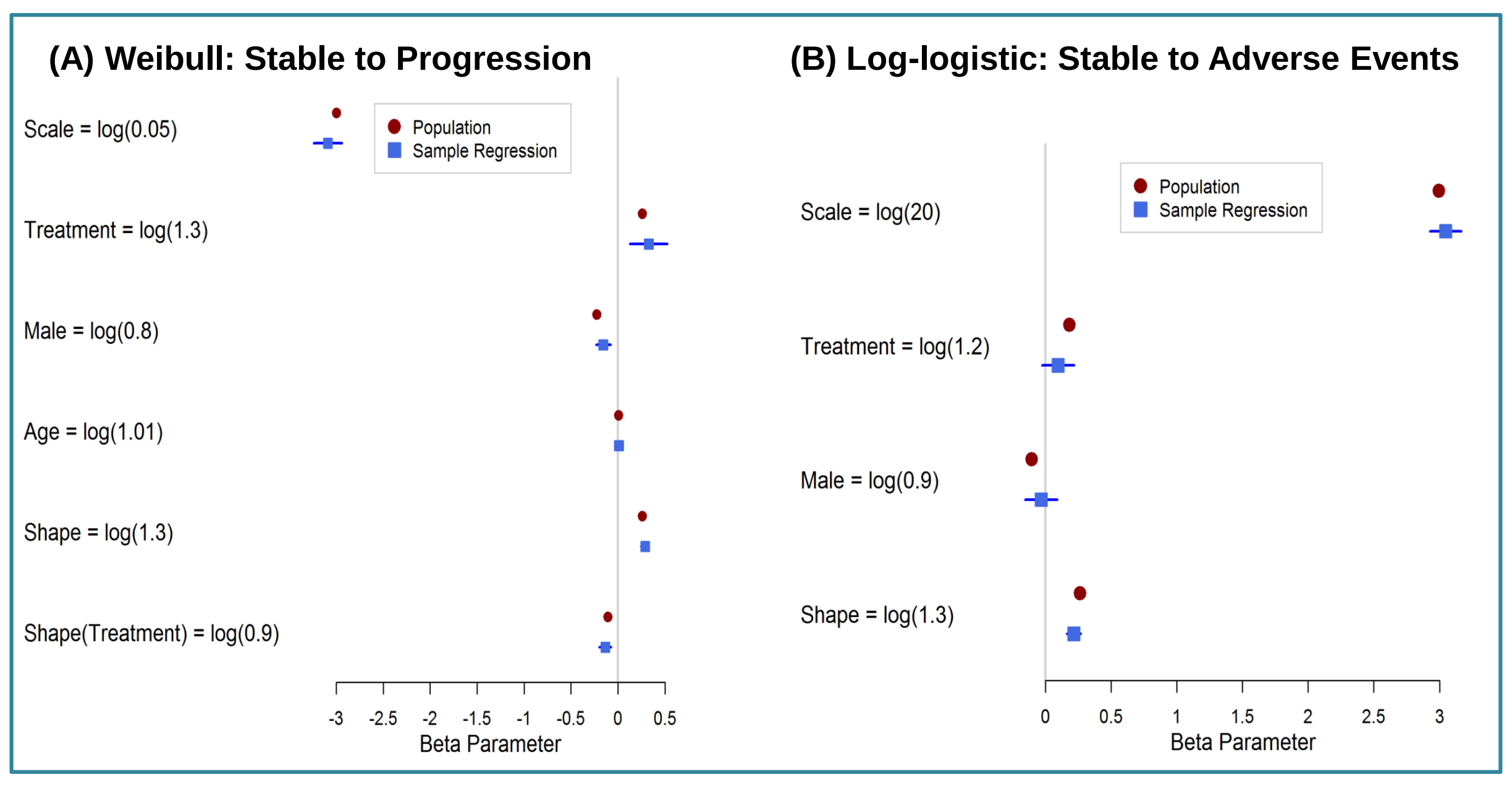
Realistic censoring was incorporated into the RCT by implementing a constant risk drop-out across time (exponential distribution) and a fixed study time end point of 18 months (all events simulated above this value were set to it and censored).

The RCT patient simulated times were then analysed using various CSHMs. Seven different distributions were applied to each of the two competing risks: Weibull, Loglogistic, Lognormal, Gompertz, Gamma, Generalised Gamma and Generalised F. Within each distribution, two versions were tried - one that put treatment effect only on the scale/location parameter (along with others) and one that also put it on all remaining parameters. Various plots and diagnostics were run to indicate the best fitting model, as detailed in the results section. Finally patient simulations were made from the best fitting models and timed for efficiency.

## Results

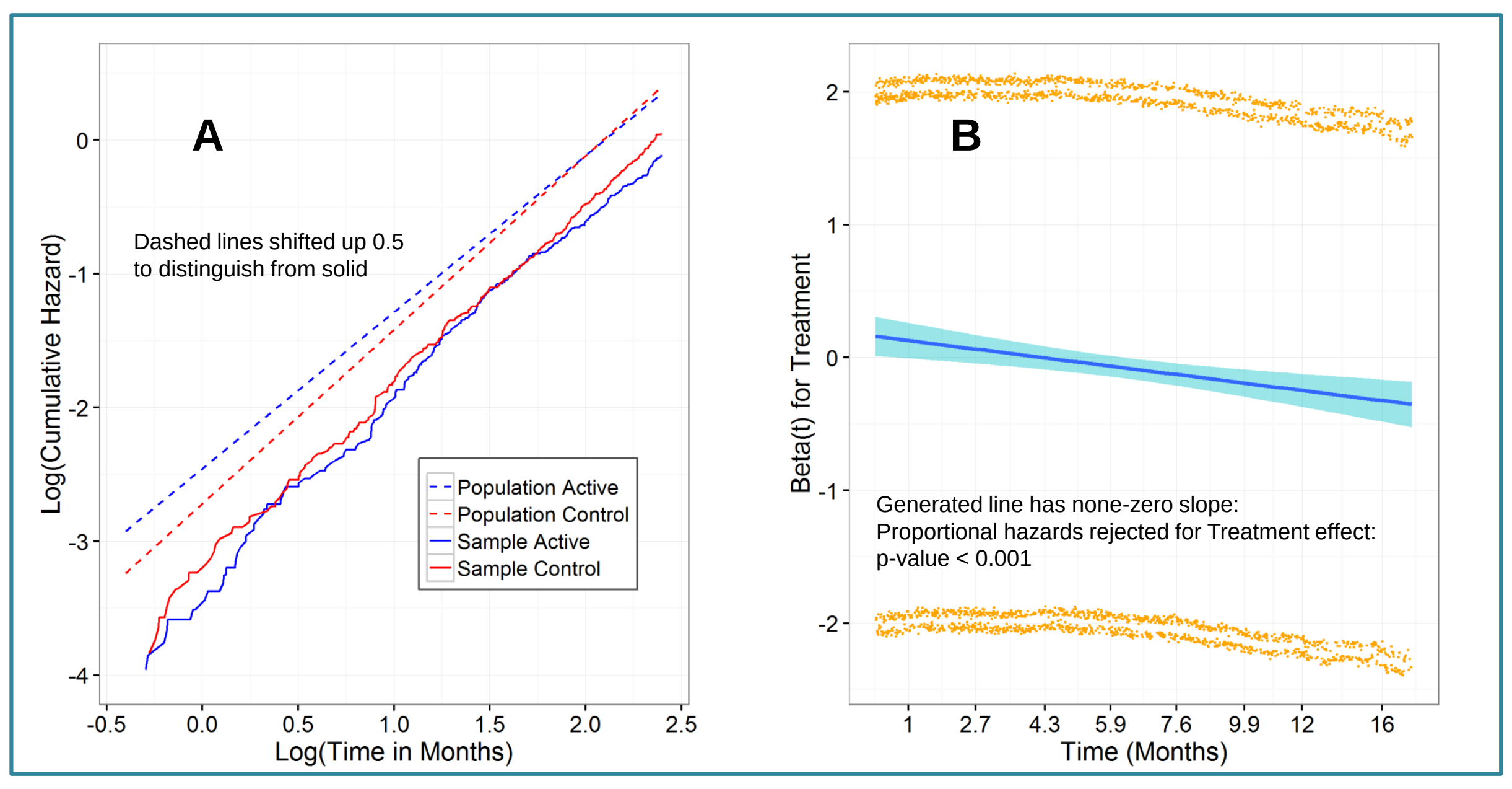
The actual “true” population parameter (Beta) values that generated the artificial RCT are shown in the left hand columns in each plot in Figure 2) for the respective competing events. Validation of this Monte Carlo generation process is supplied in these figures – applying the appropriate CSHM to this simulated data produced the parameter values shown in the sample survival regression results: the two sets are very close with all population values falling within the sample 95% confidence intervals.

**Figure 2 Population Parameters that Generated the Clinical Trial Data Shown Against Sample Survival Regression Estimates Applied to this Generated Data**



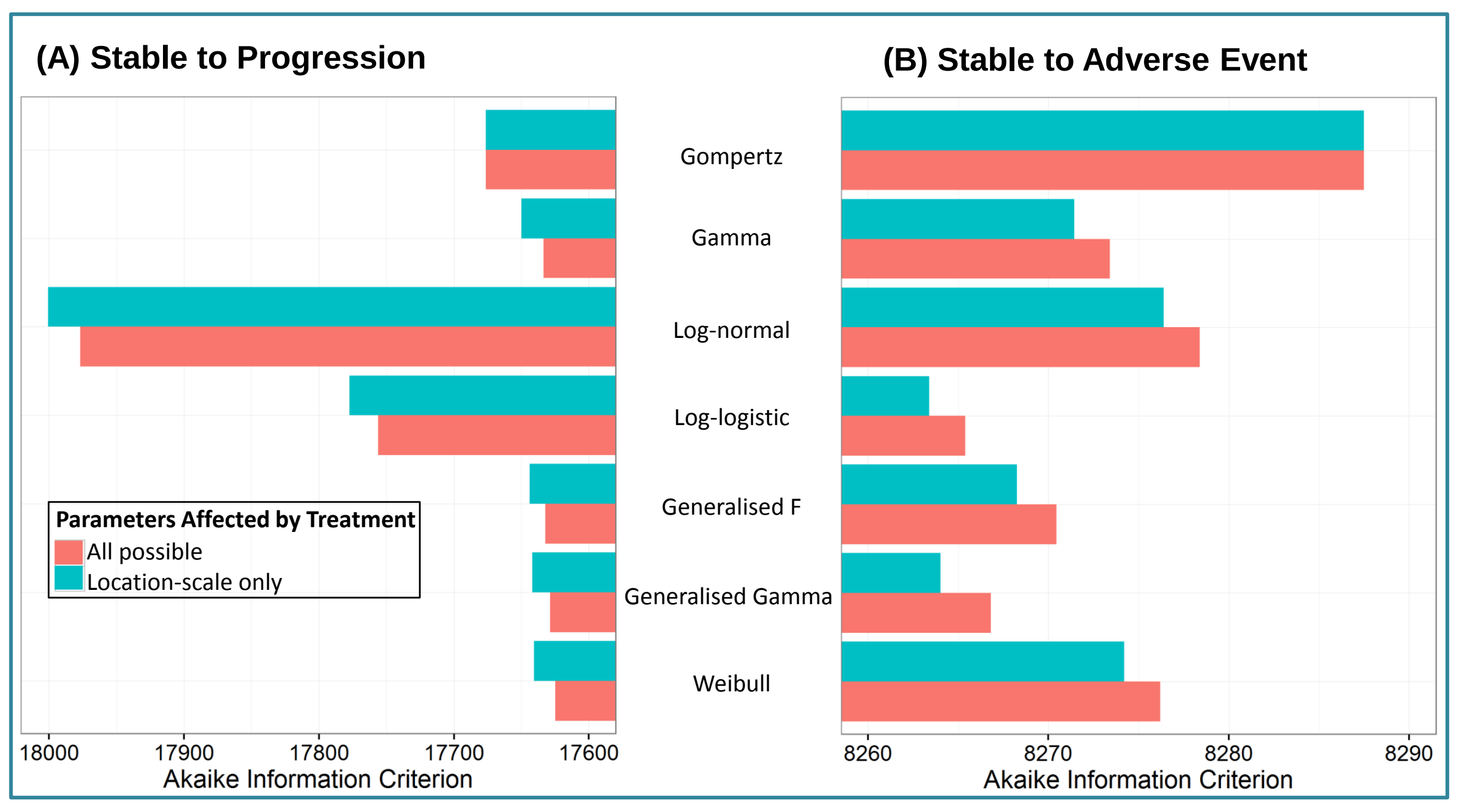
A log cumulative hazard plot against log time, which should be linear for Weibull models, is shown in figure 3A for transition Stable to Progression within males of five years of the mean age. It shows the true underlying population relationship for active versus control as dashed lines (both shifted 0.5 up). Both lines are linear but not parallel (indicating Weibull non-proportional hazards). The solid lines represent sample estimates produced from transformations to the appropriate Kaplan-Meier survival curves. These suggest linearity (confirming Weibull model) but fail to convey enough information to judge the proportional hazards assumption. This is done in Figure 3B plotting Schoenfeld residuals (in orange) for treatment effect and showing the best fitting linear line against time – it clearly has a non-zero slope thus rejecting proportional hazards for the treatment effect.

**Figure 3 Checking for (A) Weibull Functional Form (Linearity) and (B) Proportional Hazards Assumption For Treatment Effect via Schoenfeld Residuals**



Other diagnostic plots which were implemented for both competing events analysis included percentile-percentile plots between treatment groups (assessing accelerated failure time assumption), plots of logit(Kaplan-Meier survival) against log(time) to test log-logistic proportional odds assumption, Brier score predictive accuracy plots, model predicted hazard rates versus kernel density estimates plots and various residual (Standardised, Cox-Snell, Martingale-Deviance) plots. All these plus the Akaike information criterion (AIC) showed the Weibull and Log-logistic to be the most appropriate in matching the relevant population generated distributions. Results were very close however compared to the Generalised Gamma and Generalised F distributions. The latter subsumes both Weibull and Log-logistic distributions and hence this result is unsurprising. The AIC results are shown below.

**Figure 4 Akaike Information Criterion Scores for Various Competing Risk Survival Models: Parameters Affected by Treatment Allowed to Vary**

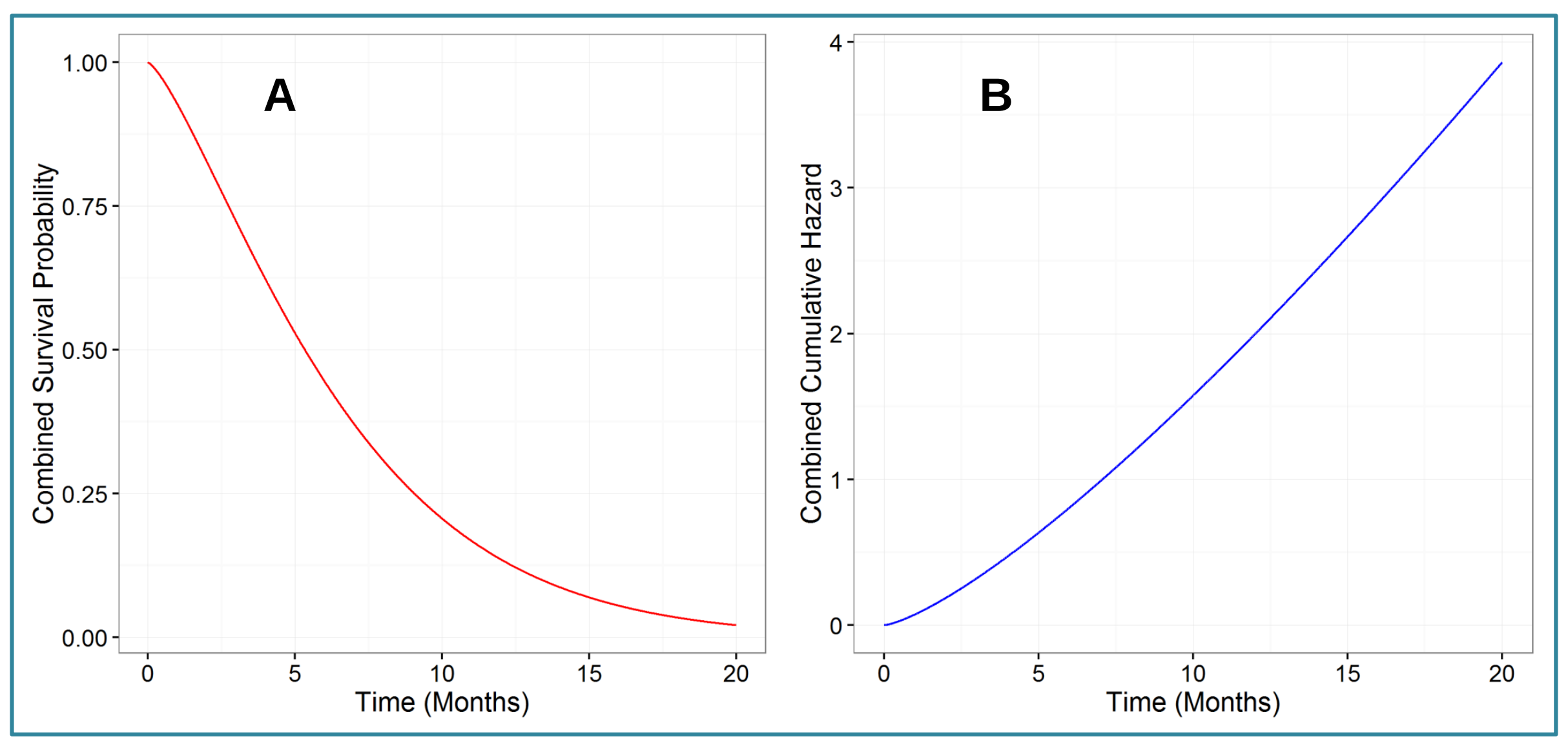


## Results (continued)

In terms of simulating the competing risks from the best fitting CSHMs (utilising models’ parameter and covariance matrices for PSA purposes) convergence was an issue for the NR method employing the combined survival probability curve. It converged for most but not all PSA parameter draws × patient profiles.

Figure 5 shows the combined survival and cumulative hazard curves for such a profile where convergence failed: the specific patient profile and PSA draw resulted in distributions of Weibull(shape=1.36, scale=0.05) and Log-logistic(shape=1.27,scale=20.9). The heavy curvature (reverse sigmoidal) in the survival curve prevented convergence (attempted to match a random uniform draw of 0.79); such a problem did not exist with the cumulative hazard curve which is clearly closer to being linear (and no inflection point).

**Figure 5 Patient Profile Combined Survival Curve (A) and Associated Cumulative Hazard Function (B) Where Simulation Fails Using (A) But Succeeds With (B)**



The time taken to achieve convergence for any given simulation varied greatly between routines. By far the most efficient was a method that used vectorised commands to generate all PSA and patient draws at the same time. This would only be possible if the competing risks occurred at the start of the modelling process. It took just 97 seconds within R to generate 10,000 PSA draws with 1000 patients within each (their survival time in State Stable computed together with which competing event was moved to). The technique adopted the NR method using cumulative hazards. Applying the same technique but using a more traditional “nested for loop” approach where results are simulated one at time took over 32 minutes (20 times slower). Switching to Brent’s method and simulating one at a time took 50 minutes (31 times slower) on an i7-4790 intel processor using one core (R3.3.1 on Windows).

Using benchmarking and profiling tools were crucial in reducing simulation times – it is not always obvious which commands are the fastest and it can vary dependent on context. For example there was a huge boost (12 times faster) when simulating results one at a time (“for loops”) from switching from equation (1) to (2) for calculating the Cumulative hazard at the current step time value t:

H = HweibullPH(t, Weib.shape , Weib.scale) + Hllogis(t, Loglog.shape,Loglog.scale) (1)

H = Weib.scale × t^Weib.shape - log(1/(1 + (t/Loglog.scale)^Loglog.shape)) (2)

Equation 1 uses commands from the excellent flexsurv package. These commands run numerous user entered error checks (commendable for a public available package) which are not too time costly within R’s vectorised operations for which they were designed. However, within R’s “for loops” they are extremely inefficient (R has poor memory management compared to other languages).

## Conclusion

Models should always be made to fit the data – it should never be the other way round. Simulating competing risk data (transition times and competing event moved to) with competing events being drawn from different distribution functions is in principle no harder than drawing them all from the same distribution. Therefore it is imperative, researchers do not choose the distribution that appears best fitting on the “most important competing event” and impose this on all others. Such a misguided approach has possibly been boosted by textbooks and online materials that show how separate survival models (one for each competing risk within the same distribution) can actually be generated from just one survival software command – readers may mistakenly believe this is always the correct way to proceed.

The NR procedure outlined in the text appears well suited to the sampling task. This appears particularly the case when utilising the cumulative hazard function: we have yet to find an example where the method has not achieved convergence (2/3/4 competing risks attempted across numerous different distribution permutations). Utilising the Survival distribution as an alternative appears more prone to convergence problems - owing to its potential reverse sigmoidal shape. However for distributions such as the Generalised F, it may be more efficient (less complex equation in terms of the survival function and its derivative), and thus both alternatives should be attempted and compared for efficiency.

If the NR fails to converge then Brent’s method can be applied (it will converge based on the fact that survival or cumulative hazard curves are monotonic). However this method is slower in achieving convergence as demonstrated in the presented benchmarking results. This is significant on realisation that the 50 minutes recorded only relates to one aspect of the DES model.

The R programming language has been used here as it is free and familiar to many. It also allows complete freedom to incorporate any statistical feature and is totally transparent – unlike standalone commercial DES modelling packages. Validation of code is often easier to achieve as well: Tasks can down into small functions and separately validated (often against academic released public R packages such as flexsurv). If run-times in R are too slow, programmers are encouraged to incorporate C/C++ code within R using the excellent Rcpp package or alternatively switch completely to “Julia” (author’s preferred choice).

Finally it has to be stated that “cause specific” are not the only way to estimate competing risk models – “sub-distribution hazard” models pursue a different approach and possess certain desirable properties. However the advantage of cause specific is that any existing survival command in software packages such as SAS, Stata or R can be used without modification (simply treating competing events as censored), whilst “sub-distribution” require specialist programming.

**For further information contact Stephen at steve@modeloutcomes.com (www.modeloutcomes.com)**

**Presented at 19th Annual European International Society for Pharmacoeconomics and Outcomes Research (ISPOR) Congress; Vienna, Austria; 31 October – 2 November 2016**