# Structure

## Introduction

### Background

In cost-effectiveness models, it is highly recommended that probabilistic sensitivity analysis (PSA) is performed. (Claxton et al., 2005; NICE, 2008) This requires producing numerous samples of parameter values used in the model, whose variability is intended to reflect uncertainty in the true value of those parameters. In many cases, the values of two or more parameters are known to be related to each other in some way, and this relationship also needs to be reflected appropriately in the PSA.

For example, we may have a Markov model where a patient who at the start of a time cycle begins in a particular health state could move to any one of four mutually exclusive and exhaustive health states by the end of a fixed time period. PSA would therefore need to reflect uncertainty in the transition probabilities for each of these four states, but do so such that the sum of the four probabilities is 1. A modeller may adopt a relatively simple approach to doing this, such as estimating three of the transition probabilities independently, then defining the transition probability of the forth state as equal to one less than the sum of the other three transition probabilities. In theory, this approach could produce negative probabilities for this forth state, and so is not statistically sound. In this example, a more sophisticated and statistically sound approach would be to estimate all four transition probabilities jointly using a Dirichlet distribution. However, this approach could require more data or additional assumptions. [EXAMPLES, REFERENCES?!]

Another type of relationship that could exist is that two or more variables are monotonically related. By this we mean that for two variables X and Y, though we are uncertain about the true value of X, and uncertain about the true value of Y, we are certain that Y is greater than X. A common example of this would be where a disease has a less severe state, and a more severe state, and it would be clinically implausible to assume that the mean health-related quality of life (HRQoL) while in the less severe state is less than in the more severe state.

In this paper, we compare ten different methods for jointly simulating the PSA of two variables that we assume to be monotonically related. These ten methods fit broadly into one of four classes of method:

1) **Naïve methods** (methods 1 and 2), where the two variables are sampled independently;
2) **Resampling and replacement methods** (methods 3, 4, 5 and 6), where draws from independent distributions are either selectively resampled or replaced with other draws;
3) **Multivariate model methods** (methods 7, 8 and 9), where the variables used in the PSA are jointly from a multivariate model where a covariance between variables is explicitly specified;
4) **Difference model methods** (method 10), where PSA draws for all but one of the variables are produced by adding a draw from a positively bounded distribution onto a draw for another distribution

In this paper we compare the properties of PSA samples created by each of the ten methods. All methods use only summary statistics, sample means and standard errors, which are often the only data available to modellers. In our comparison, the summary means and standard errors are derived from simulated individual patient data reporting the HRQoL for thirty patients with a hypothetical disease which could either be in a moderate state or a severe state. For each patient, we have their HRQoL in the moderate state, and also their HRQoL in the severe state. From the IPD we produce 1,000 joint estimates of the mean HRQoL in the moderate state and in the severe state using a bootstrapping procedure. These bootstrapped estimates, based directly on the IPD, are the gold standard against which the estimates produced by each of the methods, which use only summary data, are compared. In general, we consider methods which produce PSA samples most similar to the bootstrapped estimates to be preferable to those which produce dissimilar PSA samples.

The inspiration for this paper was that we have observed authors of economic evaluations relying too heavily on the first two classes of method, which we believe are inadequate for handing monotonicity in this context. [REFERENCES, EXAMPLES]

# Method

## Simulated Data

Our data is of thirty hypothetical patients who progress from a moderate disease state (Stage 1) to a more severe disease state (Stage 2). Each individual's HRQoL in the less severe disease state (U1) and the mere severe disease state (U2) is reported. The individual patient data (IPD) are shown in the appendix in Table 3, and the corresponding scatter plot for these data are shown in Figure 1.

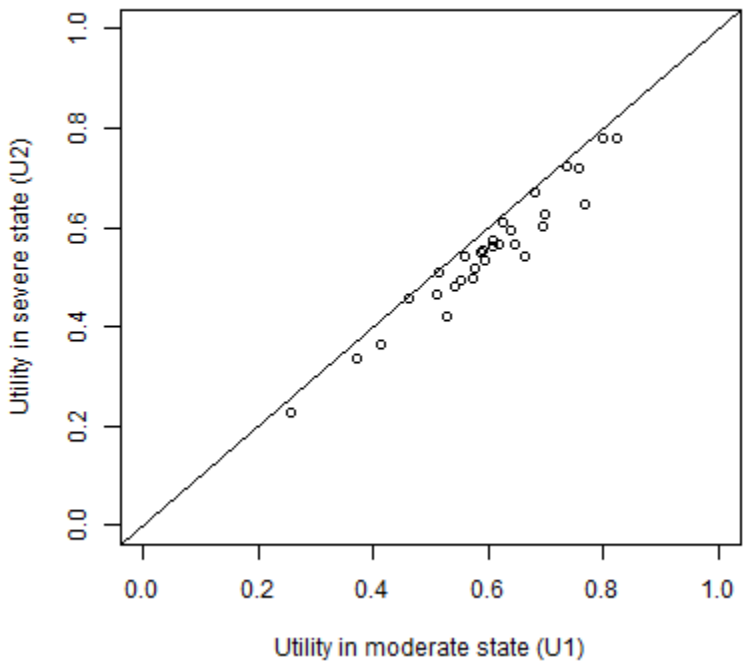



**Figure 1 A plot of the simulated individual patient data**

## Bootstrapped estimates of means

As modellers are typically interested in representing uncertainty in expected values (uncertainty in the means) rather than predicted values (uncertainty and variability in the range of values encountered), 'true' uncertainty in the mean values of U1 and U2 was estimated by repeatedly resampling the IPD, and for each resample calculating the mean values of U1 and U2 produced. Doing this 1,000 times produced the data shown in Figure 2. This approach illustrates what the modellers would be able to produce for the PSA if they had access to the IPD, and so represents the 'gold standard' against which the other methods, which have access only to aggregate level data, are compared.

We can see that the two parameters are monotonically related, as no estimate of U1 is less than the corresponding estimate of U2, and so no value crosses the diagonal line. We can also see that though the two means are strongly correlated (r= 0.97) but not perfectly correlated. Because of this, there is some variability in the differences between the two estimates, U1 – U2. This shows that simply adding E(U1) – E(U2), i.e. 0.60 – 0.55 = 0.05, onto the PSA estimates of U2 to producing corresponding PSA estimates of U1 would not be correct, as it would not accurately represent the uncertainty in the differences between U1 and U2.

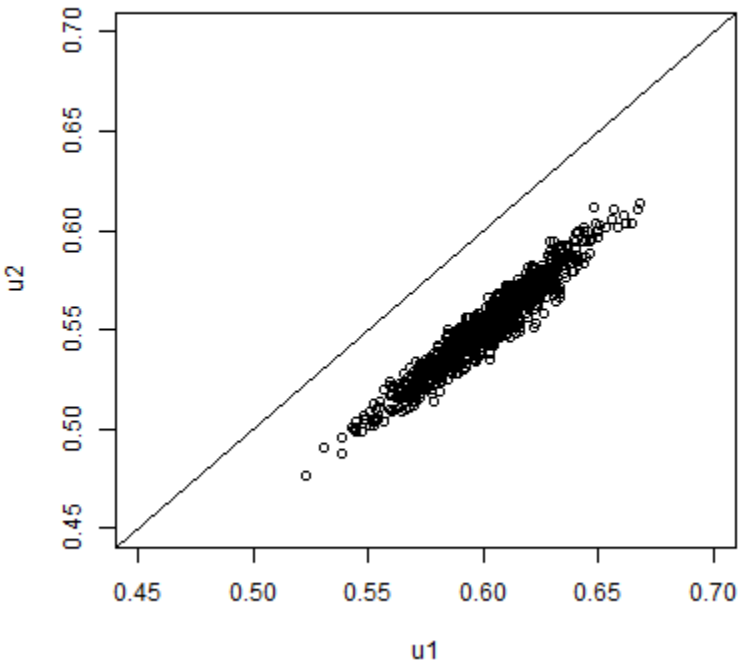



**Figure 2 Scatterplot of 1000 PSA draws of the joint means of U1 and U2 produced by bootstrapping the IPD in table 1**

## Summary statistics

In our hypothetical example, we assume the modeller will not have access to the IPD, but only to the summary information shown in Table 1. This summary information, together with the knowledge that U2 should be less than U1, is the only information used in each of the ten approaches described below.

| | U1 | U2 |
|---|---|---|
| Mean | 0.60 | 0.55 |
| 95% confidence interval of mean | 0.555 to 0.644 | 0.506 to 0.594 |

**Table 1 The assumed available summary data. This is assumed to be the only information available to the modeller**

## The Montonicity Assumption

When modellers are generating multiple estimates for use within PSA using these summary data, the key monotonicity condition that must hold is that an estimate of U2 should always be equal to or lower than a corresponding estimate for U1. More formally, if there are M runs within the PSA, and the subscript i defines predicted values from the ith run, then $U1_i \geq U2_i$ for all $i \in \mathbf{N}_1^M$; where M is the total number of PSA samples. If monotonicity were violated then some of the estimated values of U1 - U2 produced from the PSA would be negative.

## The Ten Methods

The ten methods considered are described in Table 2. Within all approaches we make the simplifying assumption that the summary statistics above relate to normal distributions, rather than other distributions which are constrained to produce utility values within a plausible range, such as 0 to 1 if assuming no worse-than-death health states. This seems reasonable given the extremely small probability that a sampled value of either U1 or U2 would be greater than 1 or less than 0.

All methods were implemented using the R programming language. (R Development Core Team, 2011) The R code, including annotations, is presented on the appendix below.

| Class | Method Number | Name | Method Description |
|---|---|---|---|
| Naïve Methods | 1 | Independent Sampling | For each of the M PSA runs, take one draw from U1 and one draw from U2 independently (i.e. assume no covariance between U1 and U2) |
| | 2 | Quantile Matching/ Number Seed Recycling | For each of the M PSA runs, use the same random number seed when drawing a sample from U2 and U1. (This is equivalent to selecting the same quantile from both distributions.) |
| Resampling and replacement methods | 3 | Upward Replacement | For each of the M PSA runs:<br>Stage 1: draw a sample from U2<br>Stage 2: draw a sample from U1<br>Stage 3: Check if the value of U1 drawn is less than the corresponding value of U2 drawn. If it is, then replace the value of U1 with the U2 value. |
| | 4 | Downward Replacement | For each of the M PSA runs:<br>Stage 1: draw a sample from U1<br>Stage 2: draw a sample from U2.<br>Stage 3: Check if the value of U2 drawn is greater than the corresponding value of U1 drawn. If it is, then replace the value of U2 with the U1 value. |
| | 5 | Upward Resampling | For each of the M PSA runs:<br>Stage 1: draw first from U1.<br>Stage 2: draw from U2.<br>Stage 3: Check if the value of U1 is less than U2. If it is, |

| | | | |
|---|---|---|---|
| | | | then go back to Stage 2 (i.e. resample). If not, then stop. |
| | 6 | Downward Resampling | For each of the M PSA runs:<br>Stage 1: draw first from U2.<br>Stage 2: draw from U1.<br>Stage 3: Check if the value of U2 is greater than U1. If it is, then go back to Stage 2 (i.e. resample). If not, then stop. |
| Multivariate model methods | 7 | AIVM Covariance | Assume that the covariance between U1 and U2 is equal to the average of the individual variances of the means (AIVM) of U1 and U2.<br>If assuming this covariance implies that the correlation between U1 and U2 is greater than 1, then instead select the covariance between U1 and U2 which implies a correlation of 1 between U1 and U2. |
| | 8 | Lower Bounded Covariance Retrofitting | Select the minimum value of a covariance between U1 and U2 such that the two following conditions are met:<br>Condition 1: U1 – U2 > 0 for all PSA runs.<br>Condition 2: The covariance between U1 and U2 is greater than AIVM.<br>If this implies that the correlation between U1 and U2 is greater than 1, then instead use the covariance value associated with a correlation of 1. |
| | 9 | Upper Bounded Covariance Retrofitting | Methodology 8 but where the second condition is that the covariance between U1 and U2 is less than AIVM. |
| Difference model methods | 10 | Beta Distribution Difference Modelling | Use a derived distribution of U1, called $U1^{(*)}$, rather than U1 itself.<br>$U1^{(*)}$ is defined as equal to U2 + Δ, where Δ is drawn from a Beta distribution. The parameters of the Beta distribution are selected so as to minimise the differences between $U1^{(*)}$ and U1. |

**Table 2 Summary of the ten approaches considered**

## Naïve methods

Methods one and two are both very simple. Method 1, independent sampling, is the simplest method of all, and does not take the monotonicity condition into account at all. Nevertheless, in cases where the means of U1 and U2 are far apart and the standard errors of both parameters are small, this method may still produce PSA values which do not violate the monotonicity assumption most of the time. With the data considered here, however, this is not the case, and so the approach is liable to produce erroneous samples. Method 2 is also frequently used in economic evaluations [references?], and simply involves using the same random number stream when drawing from both the U1 and U2 distributions. Method 2 is broadly equivalent to pairing the quantiles from the estimated distributions of U1 and U2 within PSA runs, matching the lowest estimate of U1 with the lowest estimate of U2, the second lowest estimate of U1 with the second lowest estimate of U2, and so on. For this reason, quantile-pairing was not considered as a separate strategy.

### Resampling and replacement methods

Methods 3, 4, 5 and 6 are also commonly used in economic models, as they are relatively simple to implement. [References!?] All four methods involve sampling one of the two paired values, $U1_i$ or $U2_i$, independently, before sampling the other value, $U2_i$ or $U1_i$. For methods 3 and 4, the second value is then replaced with the first value if it violates the monotonicity assumption. For methods 5 and 6, the second value is retained if it does not violate the monotonicity assumption, and resampled if it does violate the assumption. The second value is resampled until a value which does not violate the monotonicity assumption is drawn.

There are theoretical reasons to be concerned with both of these types of method. The replacement methods can be assumed to produce biased estimates of the mean value. This is because they effectively involve 'stacking' the probabilities of all values below (or above) threshold value onto that threshold, e.g. if the threshold is 0.2, then the probability of realising a value of 0.2 is equivalent to the probability of getting all values up to 0.2. Doing this shifts the centre of the probability density function, and so the expected value of the distribution. The resampling methods can also assumed to produce biased means, but with an opposite direction of effect. Like the replacement methods, it sets the probability of realising values below (or above) a threshold value to 0, but unlike the replacement method the probability of realising the threshold value is not increased. Instead, because the total probability of realising a value must equal 1, the probability of realising a value above (or below) the threshold must increase. This is somewhat analogous to comparing the distribution of distances from a starting point a ball lands on the ground after being thrown in an open space, compared with if the ball were thrown towards a vertical wall, which it will bounce against if it hits. [JM1]

### Multivariate model methods

Methods 7, 8 and 9 each involve selecting covariances on the basis either of the variances presented in the summary statistics for U1 and U2, or on whether monotonicity is maintained on all runs of the PSA. Method 7 involves setting the covariance between U1 and U2 to the average of the individual variances of the means (AIVM). Method 8 involves setting the covariance to such a value that no PSA draws violate the monotonicity assumption, subject to the constraint that the covariance is also greater than AIVM. For method 9, the covariance is also set such that no PSA draws violate the monotonicity assumption, but this time subject to the constraint that the covariance is also less than the AIVM.

Another logical constraint also applies to all three methods. This is that the covariances cannot imply a correlation of greater than 1. The correlation of two random variables X and Y is defined as follows:

$$\rho_{XY} := \frac{Cov\,(X,Y)}{\sigma_X \sigma_Y}$$

For this reason, the upper limit of the covariance must be $\sigma_X \sigma_Y$. For approach seven, this effectively states that the covariance selected is:

$$Cov\,(X,Y) = Min\left\{\sigma_X \sigma_Y \quad , \quad \frac{{\sigma_X}^2 + {\sigma_Y}^2}{2}\right\}$$

This constraint also places an upper limit on the range of covariances which may be considered in methods eight and nine.

The R code used to implement methods seven, eight and nine is presented in the appendix.

### Difference model methods

Within method 10, instead of the independent distribution U1 being used directly in the PSA, it is used indirectly, in order to produce an alternative sampling distribution U1*. This alternative distribution should produce a similar distribution of values to U1, without biased means or standard deviations, but also should ensure strict monotonicity. We do this by defining:

$$U1* \equiv U2 + \Delta$$

$$\Delta \sim Beta(a, b)$$

Because Δ is drawn from a Beta distribution, which can only produce values between 0 and 1 inclusive, we know that the monotonicity condition is satisfied because Δ would have to be negative in order for U1* to be less than U2.

In order to ensure that U1* has the same mean value as U1, we define:

$$E(\Delta) = E(U1 - U2) \equiv \mu$$

$$\mu = \frac{a}{a+b}$$

If we define $N \equiv a + b$, then $a = \mu N$ and $b = 1 - N$. We use a numerical optimisation algorithm which searches for the optimal value of N, $\widehat{N}$, subject to the above constraints, such that the root mean square (RMS) of the differences in means and standard deviation between U1 and U1* is minimised. I.e. selecting a value of N which satisfies the following condition:

$$\min(\sqrt{(\mu_{U1} - \mu_{U1*})^2 + (\sigma_{U1} - \sigma_{U1*})^2})$$

Where $\mu_{U1}$ is the sample mean of U1, $\sigma_{U1}$is the standard error of U1, and $\mu_{U1*}$ and $\sigma_{U1*}$ are the corresponding quantities for U1*. The R code used to perform this optimisation is presented in the appendix.

### Methods where monotonicity cannot be violated

For some of the methods, it is analytically impossible for monotonicity to be violated, and so they must satisfy the monotonicity condition. These methods are 3, 4, 5, 6 and 10. For methods 7, 8 and 9, which use algorithms to select covariances between parameters, it is possible that for some runs U2 values will occasionally exceed corresponding U1 values, although this may be a rare occurrence. Where violation of monotonicity is possible, modellers should be able to specify what level of monotonicity violation is tolerable. For example, monotonicity violation may be acceptable, so long as it occurs with a frequency of less than 1/10,000. For brevity, methods 3,4, 5, 6, and 10 will be described as satisfying 'strict monotonicity'; whereas methods 7, 8 and 9 will be described as satisfying 'relaxed monotonicity'.

### Comparing between methods

We use two visual approaches to compare the ten methods with each other, and with the bootstrapped estimates based on the IPD. In all cases, the closer the output from a method is to the

bootstrapped estimates, the better it is at accurately representing the relationship between U1 and U2 given only summary data.

Firstly, we produce scatterplots of 1,000 joint estimates of U1 and U2 for each of the ten methods. These are drawn on the same scale as the scatterplot shown in Figure 2, and so the joint patterns of scatter produced by each method can easily be compared with Figure 2.

Secondly, we use violin plots to compare the distribution of the quantities U1, U2, and U1 - U2 for each of the ten methods with the bootstrapped estimates. This comparison is facilitated by using violin plots, which are similar to box plots but also present kernel density estimates of distributions of the type presented in Figure 3.(Hintze & Nelson, 1998) An appropriate method for representing the monotonic relationship given only the summary data should produce distributions for these quantities which look similar to the bootstrapped values for each of these three quantities.

# Results

## Parameterisation of methods 7-9

For method seven, the variance associated with the standard errors of both U1 and U2 are 0.000504 to three significant figures, and so the AIVM is also 0.000504. The product of the standard errors of U1 and U2 is also 0.000504 to this many significant figures. This product defines the covariance which implies a correlation of 1, and so the upper bound of the covariance that can be set. This means that method seven is simply equivalent to setting the correlation between the means of U1 and U2 to 1. This also means that in this example method seven and method eight are identical.

For method eight, which forces a covariance of 0.00504, implying a correlation of 1, was also identified, as method eight uses the covariance value from method seven as its upper bound, and as this value already implies a correlation of 1 it cannot be any higher. For method nine, however, which uses a covariance of 0 as its lower bound, a covariance of 0.000380 was identified, implying a correlation of 0.754.

### Parameterisation of method ten

The optimisation routine selected an N value of 1925.72, producing Beta parameters a = 96.86, and b =1828.87. Figure 3 below shows the distribution of 1000 draws from U1* alongside 1000 draws of U1 and U2. We see that the distribution of U1* closely matches that of U1. The mean and standard deviation of U1 and U1* were both identical to two decimal places, with a mean of 0.60 and a standard deviation of 0.02.

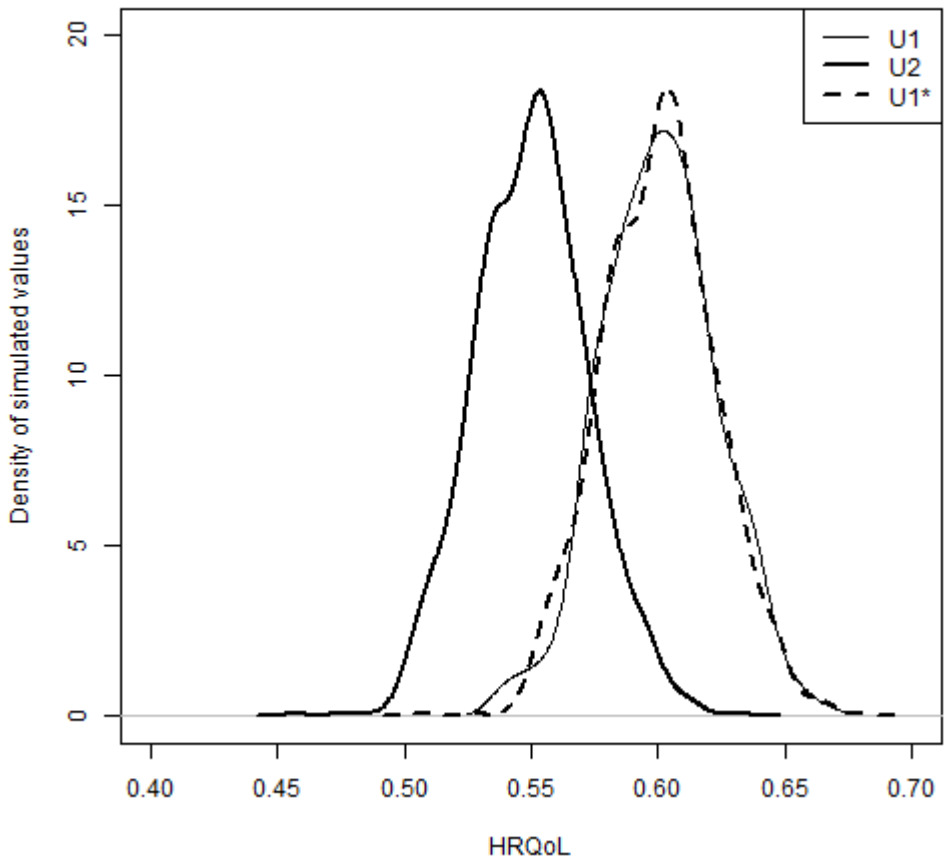



**Figure 3 Comparison of the distribution of estimates of U2, U1, and U1* produced using Method 10**

### Scatterplots

Figure 4 to Figure 13 present the scatter of the PSA produced by each of the methods one to ten.

[Because, in this example, methods seven and eight are directly equivalent, they are represented by a single scatterplot, Figure X]

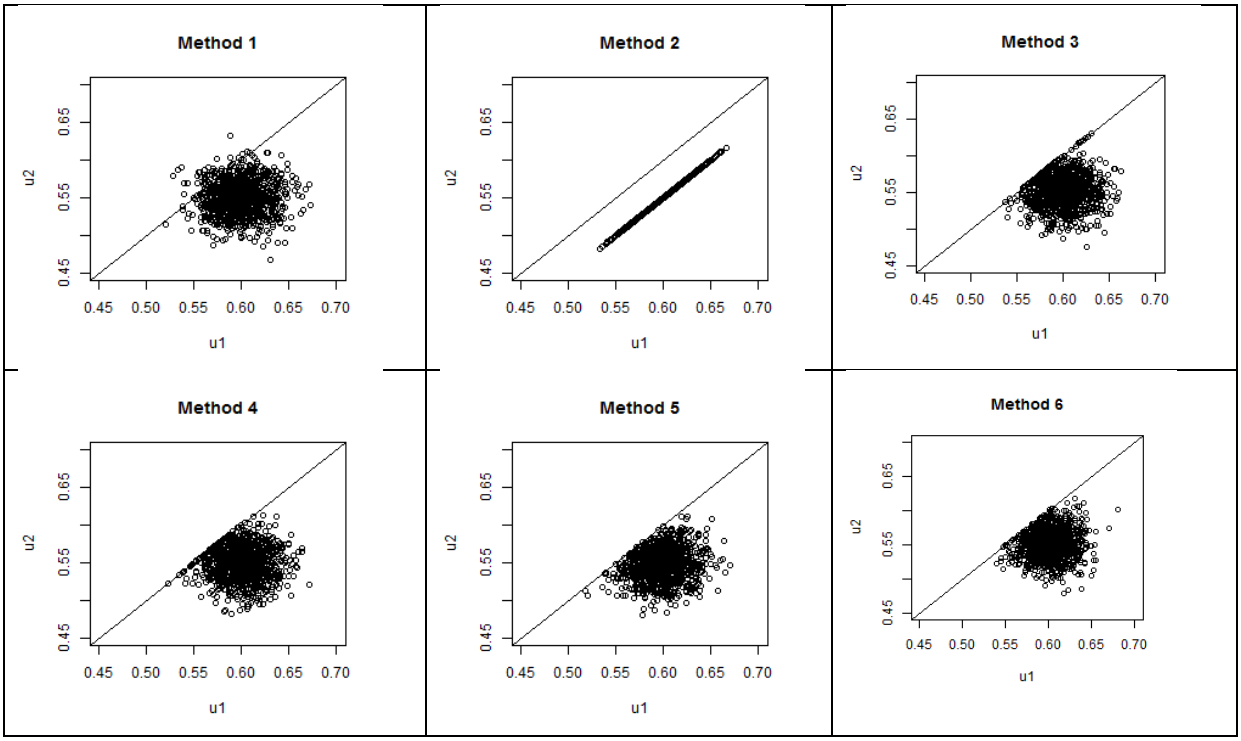
Figure 2 shows the corresponding scatter produced from the bootstrapped PSA. The diagonal line indicates parity between corresponding draws of U1 and U2. Scatter above this diagonal line shows that some proportion of the draws produced by the method violate the monotonicity assumption. A good method should be able to produce a similar pattern of scatter given the aggregate data as the bootstrapped method is able to produce using the IPD.

Figure 4 shows the scatterplot for method one. This shows some scatter above the diagonal line, showing that some proportion of the draws violates the monotonicity assumption, highlighting the inadequacy of the approach. All other approaches appear to produce no estimates which violate the monotonicity assumption.

Methods three (Figure 6), four (Figure 7), five (Figure 8) and six (Figure 9) all show nonlinearities in the scatter, with no values above the diagonal line but relatively high densities of values just below the diagonal line. These discontinuities suggest that the methods of ensuring monotonicity is liable to produce biases in the estimated mean values.

The majority of the approaches appear to produce patterns of variance in the scatter which are qualitatively dissimilar to the bootstrapped scatter. Methods one (Figure 4), three (Figure 6), four(Figure 7), five (Figure 8), and six (Figure 9) all produce uncorrelated scatter that is too wide, indicating the correlation of the PSA estimates is too low. By contrast, methods two (Figure 5), seven (Figure 10), and eight( Figure 11) all produce scatter which is too narrow, indicating the correlation estimates are too high.

We see from the scatter that method ten (Figure 9) and method nine (Figure 13) are both closest in appearance to the bootstrapped scatter, with method 10 exhibiting closer values than method 9.



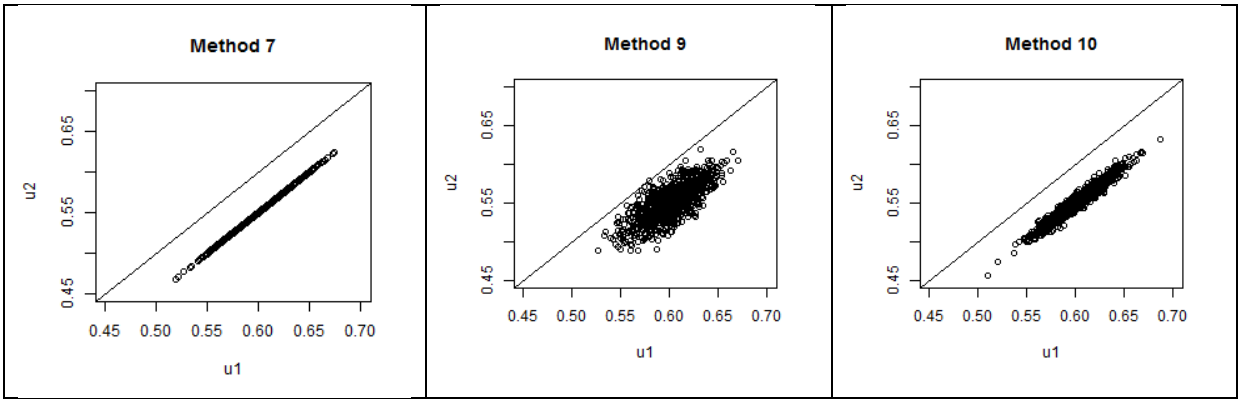



Figure 4 PSA Scatterplot: Independent sampling

Figure 5 PSA scatterplot: Same random number seed

Figure 6 Upward Replacement

Figure 7 Downward Replacement

Figure 8 Upward Resampling

Figure 9 Downwards Resampling

Figure 10 Setting covariance to AIVM

Figure 11 Upper bounded covariance fitting

Figure 12 Lower bounded covariance fitting

Figure 13 Beta difference modelling

### Monotonicity violation

The two replacement methods (3 and 4), the two resampling methods (5 and 6), and the Beta method (method 10) are all designed such that it is analytically impossible for them to violate the monotonicity assumption. The other methods could all potentially violate monotonicity. In this example, the only approach where monotonicity is violated in the 1000 PSA samples being compared is method 1, independent sampling, where 53 out of the 1000 PSA samples violated monotonicity. The precise proportion of samples violating monotonicity will differ slightly each time PSA is performed, due to stochastic uncertainty.

### Comparing U1, U2 and U1-U2

This section will compare the distribution of values produced for U1, U2, and the derived quantities U1 – U2, for each of the ten methods, compared with the gold standard, the bootstrapped data. Like the scatterplots shown in Figure 4 through to Figure 13, they therefore allow nuanced comparisons between the distributions to be made.

Figure 14 shows the distribution of values of U1 produced by each of the ten methods, compared with the bootstrapped distribution shown on the left. We see that all distributions are broadly similar, although many of the methods produce wider distributions than the bootstrapped distribution. Method 3 appears closest in shape to the bootstrapped distribution, and method 10 has the most excessively wide distribution. In all cases, however, the distributions are broadly similar.

Figure 15 shows that the distribution of estimates of U2 produced by each of the ten methods, compared with the bootstrapped estimates on the left. Again, most methods produce broadly similar distributions of these quantities, although the distributions produced by a number of methods, including methods 1, 7 and 10, appear too wide.

Figure 16 shows the distribution of U1 – U2, i.e. the differences in paired draws of U1 and U2, produced by each of the ten methods, compared with bootstrapped estimates of this quantity. As shown in Figure 4, we see clearly that method 1, independent sampling, producing some estimates where monotonicity is violated, because some of the distribution of values is below the 0. Two other types of problem are also observed.

Methods 2, 7 and 8 all severely underestimate the uncertainty in this quantity, as all estimates U1 – U2 are identical. This is because in this case methods seven and eight both explicitly assume a perfect correlation between U1 and U2, and because the standard errors of both U1 and U2 are the same method 2 effectively does the same as well.

Methods 3, 4, 5 and 6 all show the second type of problem, in that they introduce a discontinuity in the at the lower end (U1 – U2 = 0), while also having too wide a distribution at the upper end. Based on this measure, only methods 9 and 10 appear broadly appropriate in terms of representing this form of uncertainty. Of these two methods, method 10 produces the distribution of values which is closest to the bootstrapped distribution.

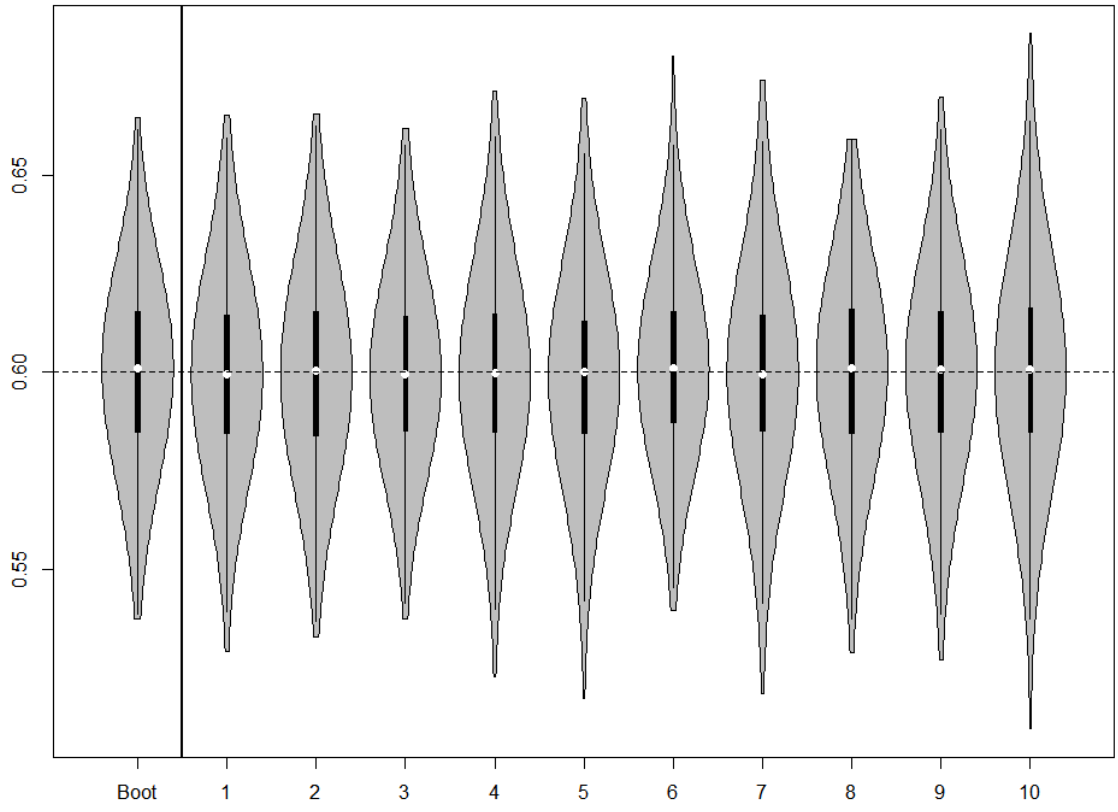



**Figure 14 Violin plot of distribution of U1 estimates produced by each of the 10 methods, compared with bootstrapped estimates**

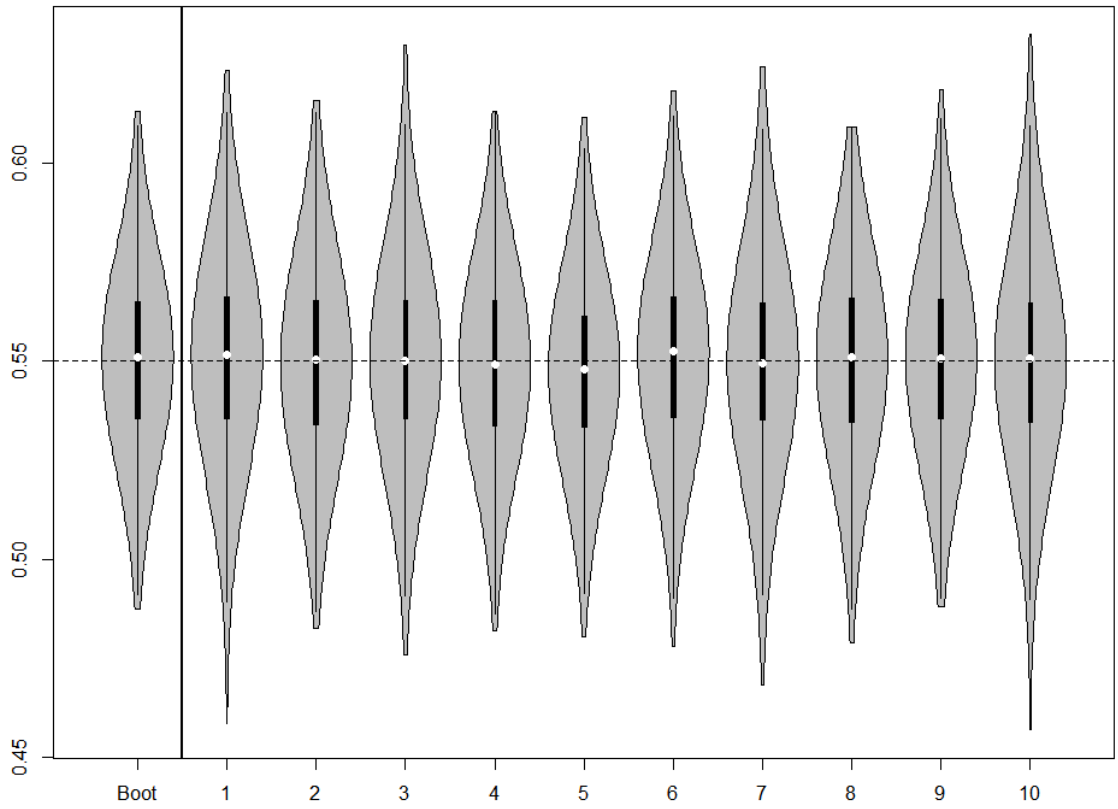



**Figure 15 Violin plot of distribution of U2 estimates produced by each of the 10 methods, compared with bootstrapped estimates**

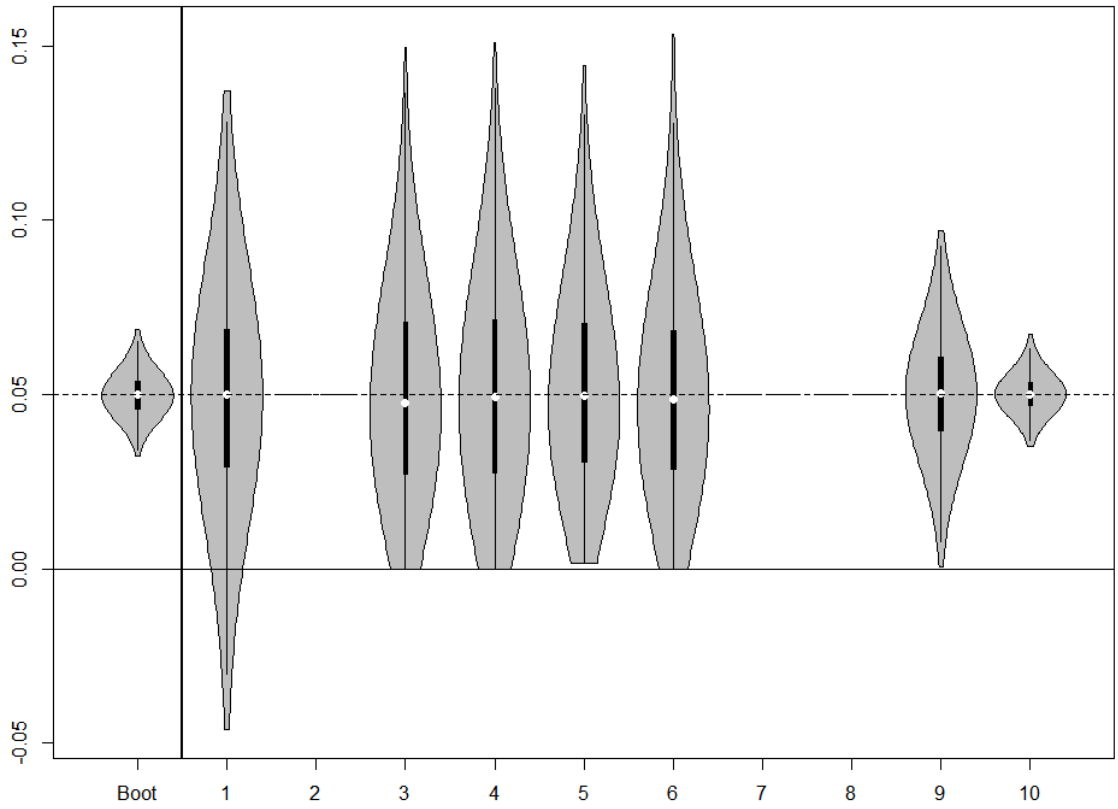



**Figure 16 Violin plot of distribution of U1 - U2 estimates, for each of the 10 methods, compared with bootstrapped estimates. The density estimates for methods 2, 7 and 8 are not missing, but all estimates of the quantity U1 – U2 are identical.**

# Discussion:

## What was found

This paper has compared ten methods which may be used to handle the monotonicity assumption within PSA, against a 'gold standard' of bootstrapped estimates of hypothetical IPD. It confirmed that independent sampling is liable to produce violations of the monotonicity assumption, and so should not be adopted where it is important to incorporate this assumption within the PSA estimates. It also found that a number of other commonly used methods for incorporating the monotonicity can effectively discard or misrepresent an important form of uncertainty: i.e. uncertainty about the difference between U1 and U2. Some of these methods (2, 7, and 8) effectively ignore this form of uncertainty, doing the equivalent of assuming perfect correlation between the two quantities, and so implicitly that the two parameters are really one parameter that is repeated as a linear transformation. Other methods (3, 4, 5, 6) introduce implausible discontinuities into the distribution of differences between values. There are also theoretical reasons to assume that some of the approaches (3, 4, 5, 6) will produce biased means, although such biases were not readily apparent in our results.

## Limitations

A limitation of the approach adopted is that the methods described do not …

Not using betas

Not using a range different U1, U2 examples.

Not presenting a 3 state (U1, U2, U3) example.

Within all approaches we make the simplifying assumption that the summary statistics above relate to normal distributions.

Another limitation with our approach is that we have not explored the dependence of the results on the number of PSA samples used. For methods 8 and 9 we should expect dependence between the covariance selected and the number of 'training' samples being compared. This is because the methods select a covariance such that none of the 'training' samples exhibit monotonicity. As the number of 'training' samples increases, the probability of extreme values, including pairs of values where monotonicity is violated, increases as well. To reduce this dependence on 'training' sample size some small tolerance for monotonicity violation, such as accepting one violation per 1,000 pairs of draws, should be considered. choice of the covariance selected will depend on the

## Implications for Research

Further research should explore how dependent our results and conclusions are on the simplifying assumptions we have made that the summary statistics relate to normal distributions. For example, if we assume that all HRQoL values have to be within the range 0 to 1, then the Beta distribution may be a more appropriate way of representing the distribution of values. Given summary statistics reporting a mean μ and a standard error σ, the Beta parameters a and b can be derived as

$$a = (\frac{1-\mu}{\sigma^2} - \frac{1}{\mu})\mu^2$$

$$b = a(\frac{1}{\mu} - 1)$$

Although using Beta distributions with this reparameterisation would have been trivial to adopt for some of the methods, such as Method 1, for other methods, such as those involving variance-covariance parameters, this approach would have been more difficult to adopt, and require further explanation to describe. Our aim within this paper in part to represent standard practice in which normal distributions are commonly used for this purpose [references?]

Further research should also look at the dependence of the results and conclusions on the data we have used. For example, the standard errors of the distribution of the two mean values U1 and U2 are the same in our data, and additional research we have carried out has shown that, when Method 2 is used, the variance of the quantity U1 – U2 varies nonlinearly as a function of the ratios of the variances of U1 and U2, with higher ratios of the variances leading to greater variance in this quantity. However, with Method 2 once the ratio of the variances is increased beyond a certain limit then a proportion of the distributions violate the monotonicity assumption, and so suffers the same limitation as independent sampling. Further details of this additional analysis is available from the corresponding author on request.

## Implications for practice

[Where mean values for U1 and U2 are far apart and standard errors are small, then even independent sampling is unlikely to produce violation of monotonicity. In these cases it may not be useful to adopt a more sophisticated method.]

[Conversely, where the means for U1 and U2 are very close together and the standard errors are large, we should be asking how confident we are about the validity of the monotonicity assumption, and how implausible it would be that the mean HRQoL in U1 is actually lower than the mean HRQoL in U2.]

[An important question to ask is how often we should believe in 'strict monotonicity'. In our dataset, we produced IPD where each patient's HRQoL in the less severe state was better than that patient's HRQoL in the more severe state. We could instead have IPD where one or more of the patients' HRQoL in the less severe state happened to be lower than their HRQoL in the more severe state. Bootstrapped estimates of the joint means of these HRQoL values could then produce some estimates where monotonicity has been violated. If we accepted that the bootstrapped approach is the 'gold standard' then a method where there is some degree of crossover (violation of strict monotonicity) should be appropriate, and conversely something like method ten, the beta difference approach, may be inappropriate. ]

[Depends how much of an issue one considers violation of monotonicity assumption to be.

Less of an issue when not looking at expected values.]

[Make sure to test whether monotonicity is a problem in practice if not in theory.

[Where SEs are small relative to differences between means of distributions, even a naïve approach assuming distributions are independent could produce no violation of monotonicity on average within 1000 PSA runs]

[Need to be sure that the monotonicity assumption is valid in the particular context considered.

For example, although perhaps one can assume increasing disease severity should definitely be associated with decreased patient utility, it might not make as much sense to assume costs are also monotonic. It may be that the most severe stages of a disease are cheaper to treat than less severe stages, as fewer effective treatment options exist.

[It could also be that many clinicians assume two formally defined disease states are actually very similar, and so it would not be contrary to their expectations if the utility value of a 'more severe' state were actually about the same or higher on average than that of the 'less severe' state.

[Also far less likely to always be true when considering predicted rather than expected values, as it's very probable that on some occasions at least some patients with a more severe disease will have higher utility than at least some patients with a less severe condition.]

When presented with summary data and with a belief that monotonicity must apply, a judicious selection of the covariance parameters or of the distributions for the differences appears appropriate. The former strategy is likely to be preferential if there are more than two parameters and there is some belief of correlation between the variables.

### Para 6

# Appendices

## Appendix 1: the hypothetical patient level data

| Patient number | U1 | U2 |
|---|---|---|
| 1 | 0.73624 | 0.72501 |
| 2 | 0.69819 | 0.62577 |
| 3 | 0.75643 | 0.71941 |
| 4 | 0.63822 | 0.59433 |
| 5 | 0.64629 | 0.56543 |
| 6 | 0.61907 | 0.56542 |
| 7 | 0.80013 | 0.77922 |
| 8 | 0.41191 | 0.36400 |
| 9 | 0.66461 | 0.54031 |
| 10 | 0.51380 | 0.50906 |
| 11 | 0.59403 | 0.53216 |
| 12 | 0.37144 | 0.33756 |
| 13 | 0.60832 | 0.57257 |
| 14 | 0.52807 | 0.42046 |
| 15 | 0.82390 | 0.77916 |
| 16 | 0.68240 | 0.66897 |
| 17 | 0.46268 | 0.45757 |
| 18 | 0.57651 | 0.51728 |
| 19 | 0.57256 | 0.49599 |
| 20 | 0.60720 | 0.56142 |
| 21 | 0.54251 | 0.48132 |
| 22 | 0.62520 | 0.61098 |
| 23 | 0.69423 | 0.60328 |
| 24 | 0.51200 | 0.46383 |
| 25 | 0.59166 | 0.55184 |
| 26 | 0.55963 | 0.54106 |
| 27 | 0.58825 | 0.55057 |
| 28 | 0.76697 | 0.64782 |
| 29 | 0.55125 | 0.49158 |
| 30 | 0.25630 | 0.22664 |

Table 3 Hypothetical individual patient data

## Appendix 2: R code

| R Code | Comments |
|---|---|
| rm(list=ls())<br><br>Data.2D <- data.frame(<br>U1=<br>c(<br>0.6981868, 0.7564343, 0.6382204, 0.6462851, 0.6190710,<br>0.8001344, 0.4119082, 0.6646116, 0.5137965, 0.5940299,<br>0.3714398, 0.6083170, 0.5280737, 0.8239041, 0.6823991,<br>0.4626827, 0.5765112, 0.5725570, 0.6071968, 0.5425066,<br>0.6251989, 0.6942350, 0.5120049, 0.5916603, 0.5596280,<br>0.5882450, 0.7669716, 0.5512535, 0.2562950<br>),<br>U2=<br>c(<br>0.6257671, 0.7194083, 0.5943290, 0.5654279, 0.5654237,<br>0.7792152, 0.3639981, 0.5403120, 0.5090605, 0.5321613,<br>0.3375571, 0.5725718, 0.4204609, 0.7791617, 0.6689688,<br>0.4575665, 0.5172808, 0.4959917, 0.5614181, 0.4813226,<br>0.6109787, 0.6032772, 0.4638334, 0.5518375, 0.5410590,<br>0.5505654, 0.6478170, 0.4915789, 0.2266444<br>)<br>)<br><br>require(MASS)<br><br>plot(U2 ~ U1, data=Data.2D, xlim=c(0,1), ylim=c(0,1),<br>xlab="Utility in moderate state (U1)", ylab="Utility in severe state (U2)")<br>abline(0,1)<br><br>cov(Data.2D)<br>cor(Data.2D)<br><br>U1.summary <- list(mu=0.60, sd=(0.644 - 0.600)/1.96)<br>U2.summary <- list(mu=0.55, sd=(0.594 - 0.550)/1.96) | Clear the R workspace<br><br>Load the data<br><br>Load a required library<br><br>Plot the data<br><br>Create a y=x line<br><br>Identifying the true variance-covariance of the scatter<br>Identifying the true correlation<br><br>The summary data for U1 and U2. This is the only data typically available to a modeller, and the only data used by the methods evaluated below. |
| **Methods** | |
| n.PSA <- 1000 | Set the number of PSA runs to use |
| ***Method 1: Independent Sampling*** | |

| | |
|---|---|
| PSA.method01 <- data.frame(u1=rnorm(n.PSA, mean=U1.summary$mu, sd=U1.summary$sd), u2=rnorm(n.PSA, mean=U2.summary$mu, sd=U2.summary$sd)) | Creates a dataframe containing 1000 draws of U1 and 1000 draws of U2, independently sampled |
| ***Method 2: Same Random Number Seed*** | |
| seed.value <- 20 | Set the value to use for the random number seed |
| set.seed(seed.value) | Set the random number seed to use the seed value |
| u1 <- rnorm(n.PSA, mean=U1.summary$mu, sd=U1.summary$sd) | Run the rnorm function using this seed value and u1 summaries |
| set.seed(seed.value) | Re-set the random number seed back to 20 |
| u2 <- rnorm(n.PSA, mean=U2.summary$mu, sd=U2.summary$sd) | Run the rnorm function using this seed value and u2 summaries |
| PSA.method02 <- data.frame(u1=u1, u2=u2) | Create a dataframe with the PSA values |
| rm(u1, u2) | Remove u1 and u2 objects stored outside the dataframe |
| ***Method 3: Upward Replacement*** | |
| u1 <- rnorm(n.PSA, mean=U1.summary$mu, sd=U1.summary$sd)<br>u2 <- rnorm(n.PSA, mean=U2.summary$mu, sd=U2.summary$sd) | Create values independently for U1 and U2 as per method 1 |
| u1[u1 < u2] <- u2[u1 < u2] | Identify the vector of values where monotonicity has been violated and replaces the violating u1 values with corresponding u2 values |
| PSA.method03 <- data.frame(u1=u1, u2=u2)<br>rm(u1, u2) | Create a dataframe with the PSA values and remove local copy of u1 and u2 |
| ***Method 4: Downward Replacement*** | |
| u1 <- rnorm(n.PSA, mean=U1.summary$mu, sd=U1.summary$sd)<br>u2 <- rnorm(n.PSA, mean=U2.summary$mu, sd=U2.summary$sd) | Create values independently for U1 and U2 as per method 1 |
| u2[u2 > u1] <- u1[u2 > u1] | Identify the vector of values where monotonicity has been valued and replace the violating u2 values with corresponding u1 values |
| PSA.method04 <- data.frame(u1=u1, u2=u2)<br>rm(u1, u2) | Create a dataframe with the PSA values and remove local copy of u1 and u2 |

| ***Method 5: Upwards Resampling*** | |
|---|---|
| u1 <- rnorm(n.PSA, mean=U1.summary$mu, sd=U1.summary$sd) | Sample the U1 values in the usual way. |
| u2 <- rep(NA, n.PSA) | Create an empty vector of the same length as the u1 vector to hold u2 estimates generated |
| for (i in 1:n.PSA){<br>  continue <- F<br>  while(continue==F){<br>  this.u2 <- rnorm(1, mean=U2.summary$mu, sd=U2.summary$sd)<br>  if (this.u2 < u1[i]){<br>  u2[i] <- this.u2<br>  continue <- T<br>  }<br>  }<br>} | A short routine that, for each element in the initially empty u2 vector, keeps resampling from the independent u2 distribution until a value is found which is less than the corresponding u1 value. |
| PSA.method05 <- data.frame(u1=u1, u2=u2)<br>rm(u1, u2) | Package u1 and u2 estimates in a data frame then deletes local copies of the u1 and u2 objects |
| ***Method 6: Downwards Resampling*** | |
| u1 <- rep(NA, n.PSA)<br>u2 <- rnorm(n.PSA, mean=U2.summary$mu, sd=U2.summary$sd) | Create an empty u1 vector<br>Sample u2 values in the usual way. |
| for (i in 1:n.PSA){<br>  continue <- F<br>  while(continue==F){<br>  this.u1 <- rnorm(1, mean=U1.summary$mu, sd=U1.summary$sd)<br>  if (this.u1 > u2[i]){<br>  u1[i] <- this.u1<br>  continue <- T<br>  }<br>  }<br>} | A short routine that, for each element in the initially empty u1 vector, keeps resampling from the independent u1 distribution until a value is found which is less than the corresponding u2 value. |
| PSA.method06 <- data.frame(u1=u1, u2=u2)<br>rm(u1, u2) | Package u1 and u2 estimates in a data frame then deletes local copies of the u1 and u2 objects |
| ***Method 7: Setting covariance to AIVM*** | |

| | |
|---|---|
| | Creates a short function which produces U1 and U2 values jointly which are correlated. The correlation is that associated with AIVM unless this would imply a correlation greater than 1, in which case a correlation of 1 is used instead. |
| MakeAIVMCov.2d <- function(mu.X, sd.X, mu.Y, sd.Y, n.psa=n.PSA){ | The function takes five inputs: the means and standard deviations of the two variables, and the number of PSA runs. |
| require(MASS) | Checks whether a library of functions has been loaded, this includes the function mvrnorm, which is required later. |
| varX <- sd.X^2<br>varY <- sd.Y^2 | Calculates variances given the standard deviations |
| aivm <- min(<br>mean(<br>c(varX, varY)<br>),<br>sd.X * sd.Y) | Produces a variable, called aivm, which is the minimum of the average individual variances of the means, or the covariance which would imply a correlation of 1. |
| sig <- matrix(data=c(varX, aivm, aivm, varY), nrow=2, byrow=T) | Produces a 2x2 covariance matrix using aivm as the off-diagonal values. |
| aivm.samples <- mvrnorm(n=n.psa, mu=c(mu.X, mu.Y), Sigma=sig )<br>colnames(aivm.samples) <- c("X.sampled", "Y.sampled")<br>aivm.samples <- as.data.frame(aivm.samples) | Produce correlated samples of the two variables<br>Formats aivm.samples to be consistent with those produced elsewhere. |
| return(aivm.samples)<br>} | Returns the labelled and formatted output as the function output |
| PSA.method07 <- MakeAIVMCov.2d(<br>mu.X=U1.summary$mu,<br>sd.X=U1.summary$sd,<br>mu.Y=U2.summary$mu,<br>sd.Y=U2.summary$sd<br>) | Uses the function created above with the summary values identified |
| names(PSA.method07) <- c("u1", "u2") | Renames the variables in the data frame created in the above line for consistency |
| ***# METHOD 8: Lower Bounded Covariance Retrofitting***<br>***# METHOD 9: Upper Bounded Covariance Retrofitting*** | |

| | |
|---|---|
| MakeBCVR.2d <- function(mu.X, sd.X, mu.Y, sd.Y, n.psa=n.PSA, incBy=0.00001, upper=T){<br>  require(MASS) | Another function, this time with seven inputs. These are the five inputs to the MakeAIVMCov.2d() function above, and the following two arguments:<br>incBy : the size of the increment in each guess for the appropriate covariance<br>upper: whether method 8 or method 9 should be calculated. Upper=T is method 9, and upper=F is method 8. Both incBy and upper have default values, which will be used if other values have not been specified. |
| varX <- sd.X^2<br>varY <- sd.Y^2 | Calculates variances for each variables as in previous function |
| if(upper==T){<br>  lowerbound <- 0<br>  } else {<br>  lowerbound <- mean(varX, varY)<br>  } | Set a value called lowerbound to 0 if method 9 is used (upper=T), oe the AIVM if method 8 is used  (upper=F) |
| upperbound <- min(sd.X * sd.Y,<br>  mean(varX, varY)<br>  ) # upper bounds are the minimum of the AIVM or the cov which implies a cor > 1 | |
| this.cov <- lowerbound<br>cat(varX, varY, lowerbound, upperbound, this.cov, "\n")<br>mus <- c(mu.X, mu.Y)<br>search <- T | |
| if(this.cov==upperbound){ # if the maximum value's been reached already<br><br>  cat("Upperbound already reached\n")<br>  search <- F # if the upper limit's already been reached, go no further<br>  testsig <- matrix(c(varX, this.cov, this.cov, varY), nrow=2, byrow=T)<br>  testsamples <- mvrnorm(n.psa, mu=mus, Sigma=testsig)<br>} else {<br>  cat("Upperbound not yet reached\n")<br>  this.cov <- lowerbound<br>  cat("This covariance: ", this.cov, "\n", sep="")<br>  testsig <- matrix(c(varX, this.cov, this.cov, varY), nrow=2, byrow=T)<br>  testsamples <- mvrnorm(n.psa, mu=mus, Sigma=testsig) | |

```
  }

  while(search==T){
    cat("trying ", this.cov, "\n")
    testsig <- matrix(c(varX, this.cov, this.cov, varY), nrow=2,
byrow=T)
    try.testsamples <- try(mvrnorm(n.psa, mu=mus,
Sigma=testsig))
    if(class(try.testsamples)=="try-error"){ # if mvrnorm has
been passed impossible values
      search <- F
      cat("Error picked up\n")

    } else {
      cat("No error in mvrnorm args\n")
      testsamples <- try.testsamples # if the attempted values
are correct, use them
      if (any(testsamples[,1] < testsamples[,2])){
        cat("Violation with ", this.cov, "\n")
        this.cov <- this.cov + incBy # increment the values by a
little bit
        cat("Trying ", this.cov, "\n")
      } else {
        cat("Found ", this.cov, "\n")
        search <- F
      }
    }
  }
  return(list(cov=this.cov, samples=testsamples))
}

tmp <- MakeBCVR.2d(
  mu.X=U1.summary$mu,
  sd.X=U1.summary$sd,
  mu.Y=U2.summary$mu,
  sd.Y=U2.summary$sd,
  upper=F
)

method08.cov <- tmp$cov

PSA.method08 <-data.frame(tmp$samples)

names(PSA.method08) <- c("u1", "u2")

tmp <- MakeBCVR.2d(
  mu.X=U1.summary$mu,
  sd.X=U1.summary$sd,
  mu.Y=U2.summary$mu,
  sd.Y=U2.summary$sd
)
```

```
method09.cov <- tmp$cov

PSA.method09 <- data.frame(tmp$samples)
names(PSA.method09) <- c("u1", "u2")


plot(u2 ~ u1, data=PSA.method08)
plot(u2 ~ u1, data=PSA.method09)
```

| *Method 10: Beta distribution difference fitting* | |
|---|---|
| rU1 <- rnorm(n.PSA, U1.summary$mu, U1.summary$sd)<br>rU2 <- rnorm(n.PSA, U2.summary$mu, U2.summary$sd) | Independent samples of U1 and U2 using normal distributions<br><br>**FUNCTIONS** |
| ShowImps <- function(U1, U2, a, b, n.PSA, main="", xlim=c(0,1), ylim=NA, ylab="", generate=F){<br>if (all(is.na(ylim))==T){<br>plot(density(U1), xlim=xlim, main=main, ylab=ylab)<br>} else {<br>plot(density(U1), xlim=xlim, main=main, ylim=ylim, ylab=ylab)<br>}<br>lines(density(U2), lty="dashed")<br><br>increment <- rbeta(n.PSA, a, b)<br>U1.subst <- U2 + increment<br><br>lines(density(U1.subst), lty="dashed", lwd=2)<br><br>legend("topright", legend=c("U1", "U2", "U1*"), lty=c("solid", "dashed", "dashed"), lwd=c(1,1,2))<br><br>if(generate==F){<br>output <- list(<br>mean.u1 = mean(U1),<br>sd.u1 = sd(U1),<br>mean.u2 = mean(U2),<br>sd.u2 = sd(U2),<br>mean.u1s = mean(U1.subst),<br>sd.u1s = sd(U1.subst)<br>)<br>} else {<br>output <- list(<br>mean.u1 = mean(U1),<br>sd.u1 = sd(U1),<br>mean.u2 = mean(U2),<br>sd.u2 = sd(U2),<br>mean.u1s = mean(U1.subst),<br>sd.u1s = sd(U1.subst),<br>U1.subst = U1.subst<br>)<br>}<br>return(output)<br>} | ShowImps function which shows, with a kernel density plot, the implications of assuming different a and b parameter values for the Beta distribution, in terms of how similar U1* is to U1. |

| | |
|---|---|
| CalcImps <- function(U1, U2, log.a, log.b, n.PSA){<br>  increment <- rbeta(n.PSA, exp(log.a), exp(log.b))<br>  U1.subst <- U2 + increment<br><br>  mean.u1 <- mean(U1)<br>  sd.u1 <- sd(U1)<br>  mean.u2 <- mean(U2)<br>  sd.u2 <- sd(U2)<br>  mean.u1s <- mean(U1.subst)<br>  sd.u1s <- sd(U1.subst)<br><br>  dif.mean <- mean.u1 - mean.u1s<br>  dif.sd <- sd.u1 - sd.u1s<br>  dif.rms <- (dif.mean^2 + dif.sd^2)^0.5<br><br>  output <- list(<br>  mean.u1 = mean.u1,<br>  sd.u1 = sd.u1,<br>  mean.u2 = mean.u2,<br>  sd.u2 = sd.u2,<br>  mean.u1s = mean.u1s,<br>  sd.u1s = sd.u1s,<br>  dif.mean = dif.mean,<br>  dif.sd = dif.sd,<br>  dif.rms = dif.rms<br>  )<br><br>  return(output)<br>} | CalcImps function which calculates the implications of different a and b parameters for the Beta distribution on the resulting root mean squared difference between the sample mean and sample standard deviation of U1 compared with U1*.<br><br>Root mean squared difference |
| MinRms.N <- function(par, U1, U2, n.PSA){<br>  this.N <- exp(par)<br>  sample.mu <- mean(U1 - U2)<br><br>  a <- this.N * sample.mu<br>  b <- this.N - a<br>  this.obj <- CalcImps(U1, U2, log.a=log(a), log.b=log(b), n.PSA)<br><br>  return(this.obj$dif.rms)<br>} | # Function for calculating the a and b parameters for the Beta function which are logically implied by different N values, and which returns the root mean squared difference |
| | **OPTIMISATION** |
| init.log.N <- 0<br>init.par <- init.log.N | Starting value of log(N) for numerical optimisation. Log(N) rather than N is used to allow unconstrained optimisation. |
| optim.out <- optim(init.par, MinRms.N, U1=rU1, U2=rU2, n.PSA=n.PSA, method="BFGS", hessian=T) | Optimisation function, which repeatedly calls the MinRms.N |

| | |
|---|---|
| | function with different values of N, converging on N values which minimise the root mean square of the difference. |
| n.bfgs <- exp(optim.out$par) | Calculates the N value identified by the optim routine |
| sample.mu <- mean(rU1 - rU2) | Calculates the sample mu |
| a.n <- sample.mu * n.bfgs<br>b.n <- n.bfgs - a.n | Calculates the a and b parameters implied by the N value and sample mu |
| Imps.output <- ShowImps(rU1, rU2, a.n, b.n, n.PSA, main="", xlim=c(0.4, 0.8), ylim=c(0, 20),<br>ylab="Density of estimated values",<br>generate=T) | Draw the densities of U2, U1 and U1* values produced by using the optimised a and b parameters for the Beta distribution |
| rU1s <- Imps.output$U1.subst | Loads 1,000 U1* samples |
| PSA.method10 <- data.frame(u1 = rU1s, u2=rU2) | Saves PSA produced by this method |
| Bootstrapped estimates for comparison | |
| methodBoot.PSA <- matrix(NA, ncol=2, nrow=n.PSA) | Create an empty matrix 2 columns wide |
| for (i in 1:n.PSA){<br>draws <- 1: dim(Data.2D)[1]<br>size=dim(Data.2D)[1]<br>tmp <- Data.2D[sample(draws, size, T),]<br>methodBoot.PSA[i,] <- c(mean(tmp[,1]), mean(tmp[,2]))<br>} | Loop 1000 times<br><br><br>Sample with replacement<br>Calculate bootstrapped means of each sample with replacement |
| methodBoot.PSA <- data.frame(methodBoot.PSA)<br>names(methodBoot.PSA) <- c("u1","u2") | Load output from above loop into a dataframe |
| Packaging results together | |

| | |
|---|---|
| MethodsBlock <- list(<br>methodboot=methodBoot.PSA,<br>method01=PSA.method01,<br>method02=PSA.method02,<br>method03=PSA.method03,<br>method04=PSA.method04,<br>method05=PSA.method05,<br>method06=PSA.method06,<br>method07=PSA.method07,<br>method08=PSA.method08,<br>method09=PSA.method09,<br>method10=PSA.method10) | # packaging results together in list to make them easier to automate |