**Upstrapping to Determine Futility: Predicting Future Outcomes from Past Data**

Jessica Wild, MS1, Alexander M. Kaizer, PhD1

1 Department of Biostatistics and Informatics, University of Colorado Anschutz Medical Campus, Aurora, Colorado

**Abstract**

**Background:** Clinical trials often involve futility monitoring to reassess the chances of a successful trial based on data already collected. Since the trial is ongoing, this available dataset includes only a fraction of the planned sample size. Since the trial’s power and sample size calculations only ensure that the treatment effect can be reliably tested given the full sample size, estimating trial futility from incomplete data is challenging.

**Methods:** Upstrapping involves resampling the available data (with replacement) to supplement data already collected until a new dataset is generated that matches the desired total sample size of the trial. The resampling is done within each treatment group to preserve a 1:1 allocation ratio.

**Simulations:** Simulations included a binary outcome measured once per subject with subjects assigned to treatment or control. Various simulation settings were considered based on sample size (100, 300, or 1000 per group), relative risk between treatment groups (1 or 0.8 for treatment compared to control), and interim stopping point (25%, 50%, or 75% of subjects enrolled). Upstrapping was then applied to the interim data. Logistic regression was used to estimate treatment effect in every upstrapped dataset. Each simulation was repeated 1000 times.

**Results & Conclusions:** Based on simulation results, we can conclude that upstrapping performs generally well for futility monitoring. For a given sample size, the likelihood of finding a significant result was comparable across interim stopping points. Similarly, the likelihood remained roughly equivalent across sample sizes at the same interim stopping point (although this is also impacted by power in the RR = 0.8 case). These results assume a constant treatment efficacy rate, meaning the next step in this research will be to explore the effects of time varying trends in disease condition over time.

**Introduction**

**Methods**

**Upstrap Algorithm**

**Simulation Settings**

**Statistical Analysis**

**Results**

**Null Setting Results**

**Alternative Setting Results**

**Discussion**

**Limitations**

This research is meant primarily as an explorative study of whether upstrapping is a reasonable approach for futility monitoring. Accordingly, many simplifying assumptions were made at both the simulation and modeling stages of analysis. We considered simulation settings based only on sample size, interim stopping point, and relative risk. Sample size in particular was chosen a priori without power calculations, meaning that of the three sample sizes two were found to be underpowered to detect the treatment difference in the alternative scenario (β = 0.52 for n = 600 and β = 0.23 for n = 200). It may be of interest to consider a sample size calibrated to achieve 80% power with a 5% type I error rate, as this is commonly considered ideal. All simulations assumed uniform subject accrual over time, and a constant treatment efficacy rate. To further test performance of the upstrap algorithm more complex simulations that account for time varying trends in disease condition over time will be necessary. It may also be worth considering more complicated modeling strategies (potentially with covariate information included) that more closely mirror common clinical trial settings in practice.

**Conclusions**

Based on the results of upstrapped futility monitoring in both null and alternative settings, we can conclude that applying the algorithm in this context appears to produce reasonable results, as long as calibration is performed beforehand to select appropriate p value and proportion thresholds to define trial success. This holds true across a wide variety of sample sizes and interim stopping points, and comparing results between differing relative risk scenarios indicates that the algorithm is able to successfully return higher likelihood of stopping for futility under the null case (RR = 1.0) compared to the alternative case (RR = 0.8).