Paper PK02

A Practical Introduction to the Bootstrap Using the SAS System

Nancy Barker, Oxford Pharmaceutical Sciences, Wallingford, UK

ABSTRACT

Discovering new medications is a field populated by many unknowns. Even when the human physiology is well understood it can be difficult to predict the way in which the body will react to a new drug in order to assess what the effects will be. Moreover, knowledge of the corresponding distributions for any derived measurements may be limited. In today's world of clinical trials it is not only important to know how a novel product performed, it is necessary to give some indication of the accuracy of any estimate of the performance. Without knowledge of the distribution, standard parametric techniques cannot be reliably executed and so an alternative is required.

Many conventional statistical methods of analysis make assumptions about normality, including correlation, regression, *t* tests, and analysis of variance. When these assumptions are violated, such methods may fail. Bootstrapping, a data-based simulation method, is steadily becoming more popular as a statistical methodology. It is intended to simplify the calculation of statistical inferences, sometimes in situations where no analytical answer can be obtained. As computer processors become faster and more powerful, the time and effort required for bootstrapping decreases to levels where it becomes a viable alternative to standard parametric techniques.

Although the SAS/STAT® software does not have any specific bootstrapping procedures, the SAS® system may be used to perform bootstrap methodology. Even with the speed of modern computers, careful use of efficient programming techniques is required in order to keep processing time to a minimum. The author will attempt to address programming techniques for bootstrapping methodology. This will include an introduction to the techniques of bootstrapping – including the calculation of standard errors and confidence intervals with the associated SAS code. This paper also compares and contrasts three different methods of calculating bootstrap confidence intervals.

INTRODUCTION

Discovering new medications is a field populated by many unknowns. Even when the human physiology is well understood it can be difficult to predict the way in which the body will react to a new drug in order to assess what the effects will be. Moreover, knowledge of the corresponding distributions for any derived measurements may be limited.

In the field of statistics, there are lots of methods that are practically guaranteed to work well if the data are approximately normally distributed and if all we are interested in are linear combinations of these normally distributed variables. In fact, if our sample sizes are large enough we can use the central limit theorem which tells us that we would expect means to converge on normality so we do not even need to have samples from a normal distribution as N increases. So if we have two groups of say 100 subjects each and we are interested in mean change from baseline of a variable then we have no need to worry and can apply standard statistical methods with only the most basic of checks for statistical validity.

However, what happens if this is not the case? Suppose we want to make inferences about the data when one of the following is true:

- Small sample sizes where the assumption of normality does not hold
- A non-linear combination of variables (e.g. a ratio)
- A location statistic other than the mean

Bootstrapping, a data-based simulation method for assigning measures of accuracy to statistical estimates, can be used to produce inferences such as confidence intervals without knowing the type of distribution from which a sample has been taken. The method is very computationally intensive, so it is only with the age of modern computers that it has been a viable technique.

The use of statistics in pharmaceutical research is becoming more and more sophisticated. It is increasingly common for proposed methodology to go beyond standard parametric analyses. In addition, cost data – with its extremely non-normal distribution is regularly collected. Bootstrapping methodology has become a recognised technique for dealing with these issues. In a recent CPMP guidance document "Guidance on Clinical Trials in Small Populations" (released for consultation in March 2005[1]), it is stated that "…some forms of Bootstrap methods make no assumptions about data distributions and so can be considered a 'safe' option when there are too few data to test or verify model assumptions … they prove particularly useful where very limited sample data are available…".

The aim of this paper is to introduce the reader to the process of bootstrapping and to look in a little more detail at two of the more common applications of bootstrapping: estimating standard error (SE) and estimating confidence intervals (CI) for statistics of interest. It should be noted that while all of the necessary calculations and SAS code has been included, a great deal of the statistical theory has been glossed over. Readers interested in understanding fully the statistical theory involved should read Efron and Tibshirani(1993)[2].

WHAT IS BOOTSTRAPPING?

The method of bootstrapping was first introduced by Efron as a method to derive the estimate of standard error of an arbitrary estimator. Finding the standard error of an estimate is an important activity for every statistician as it is rarely enough to find a point estimate; we always want to know how reasonable is the estimate – what is the *variability* of the estimator? In fact, sometimes statisticians are really greedy and not only want to know the point estimate and its standard error but also things like the bias or even the complete distribution of the estimator. If available these can be used to create confidence intervals or to test statistical hypotheses around our estimator.

The use of the term 'bootstrap' comes from the phrase "To pull oneself up by one's bootstraps" - generally interpreted as succeeding in spite of limited resources. This phrase comes from the adventures of Baron Muchausen - Raspe (1786)[3] In one of his many adventures, Baron Munchausen had fallen to the bottom of a lake and just as he was about to succumb to his fate he thought to pick himself up by his own bootstraps!

The method is extremely powerful and Efron once mentioned that he considered calling it *'The Shotgun'* since it can "… blow the head of any problem if the statistician can stand the resulting mess". The quotation relates to the bootstrap's wide applicability in combination with the large amount of data that results from its application together with the large volume of numerical computation that is required.

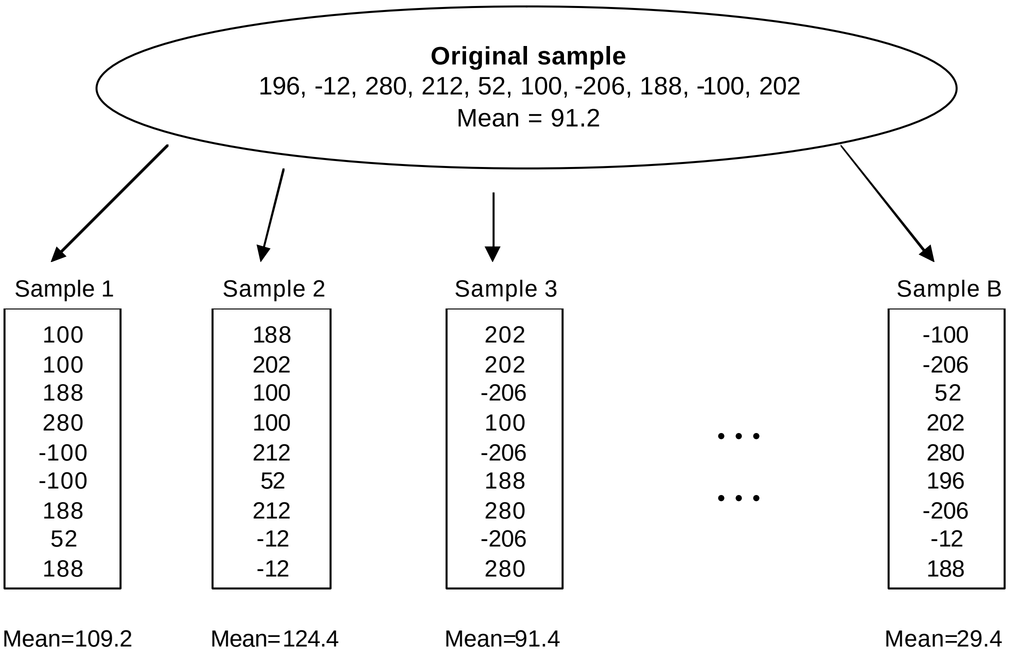

Figure 1 : Illustration of the Bootstrap Method

The basic idea behind a bootstrap is that for some reason we do not know how to calculate the theoretically appropriate significance test for a statistic: Some possible examples are that we want to do a t-test on a mean when the data is non-normal or perhaps we want to do a t-test on a median or maybe we want do a regression where the assumptions about the error term are violated.

Using the bootstrap approach assumes that the data are a random sample. The bootstrap simulates what would happen if repeated samples of the population could be taken by taking repeated samples of the data available. These repeated samples could each be smaller than the data available and could be done with or without replacement. Empirical research suggests that the best results are obtained with the repeated samples are the same size as the original sample and when it is done with replacement. Figure 1 illustrates this process.

Suppose we take a simple example, where we wish to estimate the standard error for a mean. We have 10 observations showing change from baseline for some variable X.

The original data set is randomly sampled with replacement B times with each sample containing exactly 10 observations (four of these samples are shown in Figure 1). Note that using random selection *with replacement* means that an individual value from the original data set can be included repeatedly within a bootstrap sample while other values may not be included at all. This is bootstrap replication.

The re-estimated statistics fall in a certain distribution; this may be viewed as a histogram (see Figure 2 below). This histogram show us an approximate estimate of the sampling distribution of the original statistic (in our case of the mean), and any function of that sampling distribution may be estimated by calculating the function on the basis of the histogram.

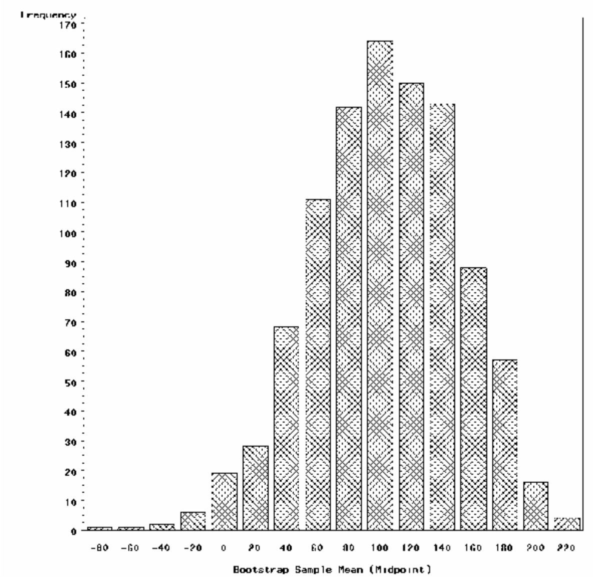

Figure 2: Histogram showing the estimated distribution of the mean for the sample data, based on 1000 replications

For example, the standard deviation (SD) of the re-estimated means is a reliable estimate of the standard error of the mean (that is, the average amount of scatter inherent in the statistic, or the average distance away from the mean at which other similarly constructed means would occur). So to find the standard error of the mean, we calculate the mean for each of the B bootstrap samples and the find the standard deviation of these means.

The more bootstrap replications we use, the more 'replicable' the result will be when a different set of samples is used. So if we re-ran the bootstrap analysis, we would be more likely to see the same results if we use a high number of bootstrap samples. To illustrate this, Table 1 shows the results of the bootstrap analyses, run 5 times each on B=20, 50, 100 and 1000.

From Table 1 below, we can see that the more replications we use the more likely we are to get similar results to previous analyses (i.e. reliability). In general I would recommend using at least 100 replications for reliability when calculating standard error.

Table 1: Results of bootstrap analyses with 20, 50, 100 and 1000 bootstrap replications

Number of Bootstrap Replications (B)	**Estimate of Standard Error**
Original sample	49.44
20	49.69
20	30.82
20	52.93
20	59.13
20	32.52
50	45.64
50	47.52
50	49.64
50	39.93
50	51.51
100	49.20
100	56.05
100	52.04
100	44.53
100	45.08
1000	47.39
1000	50.03
1000	48.03
1000	49.13
1000	48.47

Note that if we were interested in the median instead of the mean, we could easily create a standard error for the median but performing exactly the same process, but taking the standard deviation of the sample medians instead of the sample means. This is not something that could be done with standard parametric methodology.

BOOTSTRAPPING WITH THE SAS SYSTEM

Bootstrapping using SAS is relatively simple. The POINT= option in a SET command of a DATA STEP allows us to easily pick out the observations we require.

If the original data is contained within the data set ORIGINAL andwe can combine the RANUNI function with the POINT option to select our values as follows:

```
data bootsamp;
 do i = 1 to nobs;                /* Want same no. obs as in ORIGINAL */
   x = round(ranuni(0) * nobs);   /* x randomly selected from values 1 to NOBS */
   set original
       nobs = nobs
       point = x;             /* This selected the xth observation in ORIGINAL */
   output;                    /* Send the selected observation to the new data set */
 end;
 stop;                        /* Required when using the POINT= command */
run;
```

The resultant data set BOOTSAMP contains a single bootstrap replication with all variables from the original data set and with the same number of records. We can then go on to perform the analysis required. Note that it is important to use a different seed each time you create a sample in the RANUNI( ) function or you will create the same sample each time.

EFFICIENT PROGRAMMING

When performing bootstrapping operations it is extremely important to use efficient programming techniques. If we were to create each bootstrap sample individually, then perform the analysis required (PROC MEANS in the case of the example above), and append the results to the previous samples we require a lot longer for our programming.

Whenever possible, it is much better to use BY processing. If we were to create all our bootstrap samples simultaneously and then used PROC MEANS with a BY or CLASS statement, the difference in times can be quite startling. Creating the bootstrap samples simultaneously is simply a matter of using an additional DO statement in the DATA STEP as follows:

```
data bootsamp;
  do sampnum = 1 to 1000;             /* To create 1000 bootstrap replications */
    do i = 1 to nobs;
      x = round(ranuni(0) * nobs);
      set original
          nobs = nobs
          point = x;
      output;
   end;
 end;
 stop;
run;
```

This will create a data set with 1000*no. observations in ORIGINAL. Each bootstrap sample will be identified using the variable SAMPNUM which can be used to perform later BY processing.

Using the first method of creating each sample and finding its mean individually and appending the result to the data set containing the means from the previously calculated samples took the author’s home computer a total of *2 minutes and 41 seconds*. Using the second method of creating all samples simultaneously then using PROC MEANS with a CLASS statement took approximately *1 second*.

From this we can see that the optimal method of running code is to create all samples within a single data set and then use by processing (with the sample number as the BY variable). However, this is not always possible. You may be running an analysis that requires some form of coding that has no BY processing (e.g. SAS/IML) it may be that your initial data set is so large that your computer could not deal with running everything at once. In such a case the author would recommend closing your log before running (ODS LOG CLOSE;) as this gives a slight time saving. Running the code in this manner reduced the time from 2 minutes and 41 seconds to 1 minute and 50 seconds.

BOOTSTRAPPING FOR TREATMENT GROUP COMPARISONS

In the majority of clinical work, we are looking to compare two or more treatments. To do this we often use randomisation to assign the subjects to each treatment.

Suppose we have a trial with two treatment groups. If we were to use the bootstrapping methods to randomly select subjects we would have many bootstrap samples where the balance of subjects in each treatment group is different from the original sample. Instead, we need to create bootstrap samples which have the same number of observations in each group as are found in the original. To do this we re-sample independently for each treatment group and then reset the observations together.

Consider our original example: let us suppose that we had another 10 subjects in the study who were on a different treatment. Suppose that the first set of data came from subjects treated with ‘active’ (referred to as Active) medication while a further 10 subjects came from a group on ‘placebo’ (referred to as Placebo) medication (with values = 120, -80, -63, 200, 23, 54, -198, 165, -8, 19). This time, we wish to find the difference between the means of the two groups.

As discussed above, we create bootstrap samples which each have 10 Active values and 10 Placebo values, by independently sampling from each group. We then calculate the mean for each group within each sample and find the difference between the means (our statistic of interest) for each sample. We can then look at the distribution of this statistic in exactly the same way as we did for the ‘single group’ mean above.

To do this with SAS we need to split the data before re-sampling. Unfortunately, the POINT= option does not work when the WHERE= option is also used in the set statement, so two DATA steps are required: the first to divide the datasets for each treatment group and the second to perform the bootstrap sampling. If the original data set (ORIGINAL) contains two variables called CHANGE (for change from Baseline value) and TRT (treatment: 1=Active, 2=Placebo) – the following code will perform the desired analysis.

```
%macro bootse(b);
  data orig1 (where=(trt=1))
       orig2 (where=(trt=2));  /* Create one data set for each treatment */
   set original;
  run;

  data boot;
  %do t=1 %to 2;              /* Create independent sets of replications */
     do sample=1 to &b;
       do i = 1 to nobs;
         pt = round(ranuni(&t)*nobs) ;
         set orig&t nobs = nobs  point=pt; /*use treatment-specific data */
         output;
       end;
     end;
   %end;
   stop;
  run;

  proc means                   /* find the mean, by sample and treatment */
     data=boot
     noprint
     nway;
   class sample trt;
   var change;
   output out=x
         mean=mean;
  run;

  data diffmean;
     merge x (where=(trt=1) rename=(mean=mean1))
           x (where=(trt=2) rename=(mean=mean2));
     by sample;
   diffmean=mean1-mean2;    /* calculate the difference between the means */
  run;

  proc means
     data=diffmean
     std;
   var diffmean;
   output out=bootse
          std=bootse;
  run;
%mend;
```

CALCULATING CONFIDENCE INTERVALS USING THE BOOTSTRAP

So far we have looked at the computation of bootstrap standard errors. Standard errors are often used to assign approximate confidence intervals to a parameter q. For example, given an estimate $\hat{q}$ and an estimated standard error $s\hat{e}$, the usual 95% confidence interval is:

$$\text{Mean} \pm 1.96\text{*SE.} = \hat{q} \pm 1.96 * s\hat{e}$$

This is based on the assumption that:

$$Z = \frac{\hat{q} - q}{s\hat{e}} \sim N(0, 1)$$ (i.e. is approximately distributed to the Normal distribution with mean=0 and sd=1)

This is valid as $n \mapsto \infty$, but is only an approximation for finite samples. A better approximation would perhaps use the *t* distribution and assume that (for a sample size of n):

$$Z = \frac{\hat{q} - q}{s\hat{e}} \sim t_{n-1}$$ (i.e. is approximately distributed to the t-distribution with n-1 degrees of freedom)

For small samples this is a better approximation, but the use of the *t* distribution doesn't adjust the confidence interval to account for the skewness in the underlying population – or other errors that can result when $\hat{q}$ is not the sample mean.

Through the use of bootstrapping we can obtain accurate intervals without having to make normal theory assumptions. There are a variety of methods available which compute confidence intervals and we will go into a little detail on each.

In general, the variability between estimators for confidence intervals is greater than for standard errors and so it is better to have a larger number of bootstrap replications. The 100 replications recommended to calculate standard errors is not really acceptable for confidence limits. This author would generally recommend a set of 10,000 replications. The sampling error and number of replications required will not be discussed here. Efron and Tibshirani[1] (Chapter 19) gives an excellent discussion of this for those wishing for more detail.

For each of the following types of confidence intervals, we will demonstrate the SAS code required, assuming that we are trying to calculate a 95% confidence interval for the difference between the means for the example data above. In all cases, the %BOOTSE macro will be run and will create the data sets BOOT, DIFFMEANS and BOOTSE are in the section above.

THE BOOTSTRAP-T INTERVAL

This method allows us to estimate the distribution of our statistic $Z = \frac{\hat{q} - q}{s\hat{e}}$ directly from the data.

Suppose we create a set of bootstrap samples. For each, we estimate a value of Z as follows:

$$Z*(b) = \frac{\hat{q}*(b) - \hat{q}}{s\hat{e}*(b)}$$

where $\hat{q}*(b)$ is the estimate of $\hat{q}$ for the b^th bootstrap sample

and $s\hat{e}*(b)$ is the estimated standard error of $\hat{q}*$ for the b^th bootstrap sample.

The α^th percentile of $Z*(b)$ is estimated as the value $\hat{t}^{(a)}$ such that:

$$\frac{\#\{Z*(b) \le \hat{t}^{(a)}\}}{B} = a$$ where #{x} denotes the number of observations which meet condition x

The confidence interval is then calculated as:

$$\left(\hat{q} - \hat{t}^{(1-a)} \times s\hat{e}, \hat{q} - \hat{t}^{(a)} \times s\hat{e}\right)$$

This looks like really complicated statistics, but in practice is not difficult.

For example, if we want to calculate a 95% confidence interval for the mean and we plan to use 10,000 bootstrap samples, we estimate the value of the mean and its standard error for each sample. Then we calculate the value $Z*(b)$ as (mean of bootstrapped sample – mean of original sample)/SE of bootstrapped sample.

We then order the values of $Z*(b)$ from low to high and take $\hat{t}^{(0.025)}$= 250th value and $\hat{t}^{(0.975)}$=9750th value.

The confidence interval is then calculated using the mean from the original sample ($\hat{q}$) and the estimated standard error ($\hat{se}$ as calculated in the previous section).

So the lower limit =$\hat{q}$ -($\hat{t}^{(0.975)}$ x $\hat{se}$) and the upper limit is $\hat{q}$ -($\hat{t}^{(0.025)}$ x $\hat{se}$).

To do this in SAS we would use the following code:

```
%bootse(10000);

data bootorig;                      /* set original and sample data together so can */
   set original (in=a)              /* calculate mean values in one procedure       */
       boot;
 if a then sample=0;
run;

proc means
   data=bootorig
   noprint
   nway;
 class sample trt;
 var change;
 output out=x                            /* mean and standard error for each sample */
        mean=mean
        var=var
        n=n;
run;

data diff_z;
   merge x (where=(trt=1) rename=(mean=mean1 var=var1 n=n1))
         x (where=(trt=2) rename=(mean=mean2 var=var2 n=n2));
   by sample;

 diffmean=mean1-mean2;                /* calculate the difference between the means */
 diffse = sqrt((var1+var2)/(n1+n2));  /* estimate SE of difference                  */

 retain origdiff;                                 /* add value of original estimate */
 if sample=0 then origdiff=diffmean;              /* of difference to each record   */

 diff_z = (diffmean - origdiff)/diffse;
run;

proc sort
   data=diff_z;                                    /* order diff_z from low to high */
 by diff_z;
run;
```

```
data t_vals;                /* data set containing values for t(0.975) and t(0.025) */
   set diff_z end=eof;

  retain t_lo t_hi;
  if _n_=9750 then t_lo=diff_z;
  if _n_= 250 then t_hi=diff_z;

  if eof then output;
run;

data ci_t;
   merge diff_z (where=(sample=0))
         bootse (keep=bootse)
         t_vals (keep=t_:);

  conf_lo = origdiff - (t_lo * bootse);
  conf_hi = origdiff - (t_hi * bootse);

  keep origdiff bootse t_lo t_hi conf_lo conf_hi;
run;
```

A print out of the final data set ci_t would give us the following:

```
          origdiff     bootse      t_lo       t_hi       conf_lo     conf_hi
            103.2     60.2505    3.44251    -3.08735    -104.213    289.215
```

Giving us a confidence limit of (-104.2, 289.2) for the difference in the means between the groups.

It should be noted that in order to calculate our $Z^*(b)$ statistic for each bootstrap sample, we require the $\hat{se}^*(b)$, the standard error of $\hat{q}^*(b)$ for the bth sample. For this example, we were able to use the plug-in estimate:

$$\text{SE for difference between means} = \sqrt{\frac{VAR(A)+VAR(B)}{n_A+n_B}}$$

However, what happens if we want to look at a statistic for which there is no parametric approximation of the standard error? In order to approximate the standard error, we would need to calculate a bootstrap estimate *for each bootstrap sample*. In other words, two nested levels of bootstrapping. Since we are recommending a minimum of 100 replications in order to estimate a standard error and at least 1000 replications in order to calculate confidence intervals, this would give an overall number of bootstrap replications of 100*10000=1,000,000 – a formidable number, especially if our statistic of interest is costly to compute!

THE PERCENTILE INTERVAL

This is the simplest of the methods for calculating a bootstrap confidence interval. If we wish to calculate a 95% confidence interval we simply select the bootstrap estimates which lie on the 2.5th percentile and 97.5th percentile. So if we had calculated 10000 bootstrap estimates of our statistic q. We would order them from low to high and take the 250th value as the lower limit and the 9750th value as the upper limit.

Within SAS this is very straightforward:

```
%bootse(10000);

proc sort
   data=diffmean;                 /* sort estimates for difference from low to high */
  by diffmean;
run;
```

```
data ci_perc;
   set diffmean end=eof;      /* pull out 250th and 9750th values as confidence limits */

    retain conf_lo conf_hi;
    if _n_= 250 then conf_lo=diffmean;
    if _n_=9750 then conf_hi=diffmean;

    if eof then output;

    keep conf_lo conf_hi;
run;
```

A print out of the final data set ci_t would give us the following:

```
conf_lo    conf_hi
  -26       208.7
```

Giving us a confidence limit of (-26.0, 208.7) for the difference in the means between the groups.

BETTER CONFIDENCE INTERVALS

One of the most important goals of bootstrap methodology is that we produce confidence intervals which closely match 'exact' confidence intervals in the situations where statistical theory yields an exact answer. Unfortunately, neither the bootstrap-t method, nor the percentile method defined above meet this goal. The bootstrap-t intervals have good theoretical coverage – but tend to be erratic in practice. In addition, if no statistical formulae are available to estimate the standard error for the statistic of interest, a nested bootstrap procedure is required which can increase the required number of samples (and hence required computational time) exponentially. The percentile intervals are simpler to use and less erratic but comparison with exact statistical methods have demonstrated these to have less than satisfactory coverage in some cases.

Part of the problem with these confidence intervals is that they are unable to deal with the issues related to:

1. the bootstrap estimates are biased with respect to the original estimate
2. the standard error varies with the value of the estimate

An improved version of the percentile method has been formulated to help deal with these problems. The bias-corrected and accelerated (BCa) intervals have been shown to have a substantial improvement over the percentile method in both theory and practice. The method does however require a little knowledge of bias and jackknifing techniques before the computation can be performed.

BIAS

In statistics, bias is defined as "*any systematic failure of a sample to represent its population base*". In the case of bootstrapping, we want to see if there is a *median bias* – i.e. a systematic underestimation or overestimation of our sample statistic when calculated by the bootstrap replications. To do this we look at the proportion of samples which calculate a sample statistic which is below the statistic calculated by the original sample.

Let	$\hat{q}$	= the actual statistic calculated on the original records
	$\hat{q}^*(b)$	= the statistic calculated from the b th bootstrap sample
	B	= the number of bootstrap samples

Then the bias correction $\hat{z}_0 = \Phi^{-1}\left(\frac{\#\{\hat{q}^*(b) < \hat{q}\}}{B}\right)$

where $\#\{condition\}$ = number of samples to meet the specified condition and $\Phi^{-1}(x)$ indicates the inverse function of a standard normal distribution (PROBIT function in SAS) e.g. $\Phi^{-1}(0.975)$ = 1.96.

To calculate this in SAS for our example for difference between the means we would use the following code:

```
%bootse(10000);

data bootorig;            /* set original and sample data together so can calculate */
   set original (in=a)    /* mean values in one procedure                           */
       boot;
 if a then sample=0;
run;

proc means
   data=bootorig
   noprint
   nway;
 class sample trt;
 var change;
 output out=bootmean           /* mean for each sample */
        mean=mean;
run;

data diffmean (keep=sample diffmean)   /* data set containing bootstrap values      */
     bias     (keep=bias);             /* data set containing bias correction value */

   merge bootmean (where=(trt=1) rename=(mean=mean1))
         bootmean (where=(trt=2) rename=(mean=mean2)) end=eof;
   by sample;

 diffmean=mean1-mean2;                /* calculate the difference between the means */

 retain origdiff;                             /* add value of original estimate */
 if sample=0 then origdiff=diffmean;          /* difference to each record      */

 if diffmean lt origdiff then lessthan=1;    /*flag if bootstrap sample gives lower */
 else                         lessthan=0;    /*value than original sample           */

 retain nless 0;
 if sample gt 0 then nless=nless+lessthan;      /* count samples with flag lessthan */

 if sample ne 0 then output diffmean;    /* output only bootstrap sample statistics */

 if eof then do;                 /* for the last value calculate:                   */
    propless=nless/sample;       /* 1. proportion of values below original estimate */
    bias=probit(propless);       /* 2. inverse normal of that proportion            */
    output bias;                 /* 3. output only that record to new data set      */
 end;
run;
```

For our data, 54.5% of bootstrap samples gave differences between the means which were less than that found by the original sample. This gives us a value for the bias of 0.1135.

JACKKNIFING AND ACCELERATION

The acceleration statistic refers to the rate of change of the standard error of the estimate of our sample statistic with respect to the true value of the statistic. Efron and Tibshirani proposed that this be calculated using *jackknife values* of the estimate of the statistic.

Jackknifing looks at how much each individual record influences the estimate. To do this we recalculate the estimate of the LED with all but one record - removing each record in turn. For our example we would create a set of (10+10)=20 estimates for the difference between the mean which are each based on 19 of the records with a different record being removed each time.

Let $\hat{q}_{(i)}$ = the statistic of interest calculated on the jackknifed sample with the ith record removed

n = the number of jackknifed samples i.e. the number of records across all treatment groups

$\hat{q}_{(\bullet)}$ = the mean of the n jackknife samples = $\sum_{i=1}^{n} \hat{q}/n$

Then the acceleration $\hat{a}$ = $$\frac{\sum_{i=1}^{n}\left(\hat{q}_{(\bullet)} - \hat{q}_{(i)}\right)^3}{6\left\{\sum_{i=1}^{n}\left(\hat{q}_{(\bullet)} - \hat{q}_{(i)}\right)^2\right\}^{3/2}}$$

This statistic may look a little daunting, but is easy to calculate. We simply create our jackknife samples, find the difference between the mean of the samples and each sample and calculate both a squared value and a cubed value of our difference. Then we sum these squared and cubed differences and plug the values into the equation. In SAS, for our example, we would use the following code.

```
data origjack;                  /* create a new data set which contains observation */
     set original end=eof;      /* numbers 1 to &nobs (no. obs in data set)         */
 obsnum=_n_;
 if eof then call symput('nobs', put(obsnum, 2.));
run;

%macro jackdata;                            /* use macro for %do processing utility */
 data jackdata;
     set
      %do i=1 %to &nobs;                           /* do loop to create all samples */
         origjack (in = in&i
                   where=(obsnum ne &i))     /* remove a different value each time */
      %end;;

   %do i=1 %to &nobs;
      if in&i then repeat=&i;                /* add repeat number for each sample */
   %end;
 run;
%mend;
%jackdata;

proc means
   data=jackdata
   noprint
   nway;
 class repeat trt;
 var change;
 output out=jacksum                                          /* mean for each sample */
        mean=mean;
run;
```

```
data jacksum2;
   merge jacksum1 (where=(trt=1) rename=(mean=mean1))
         jacksum1 (where=(trt=2) rename=(mean=mean2));
   by repeat;
 diffmean=mean1-mean2;                /* calculate the difference between the means */
run;

proc sql
    noprint;
  select mean(diffmean)        /* put mean of jackknifed values into macro variable */
  into   :meanjack
  from   jacksum2;
quit;

data jacksum3;
    set jacksum2;
  cubed=(&meanjack - diffmean)**3;            /* create cubed value of difference   */
  squared=(&meanjack - diffmean)**2;          /* create squared value of difference */
run;

proc means
    data=jacksum3
    noprint;
  output out=jacksum4
         sum(cubed)=sumcube                            /* find sum of cubed values   */
         sum(squared)=sumsquar;                        /* find sum of squared values */
run;

data accel;
    set jacksum4;

  accel=sumcube / (6 * (sumsquar**1.5));           /* plug values into equation for */
  keep accel;                                      /* the acceleration statistic    */
run;
```

For our data, the acceleration statistic calculated is -0.0238.

CALCULATING THE BCA INTERVAL

Once we have calculated the bias correction $\hat{z}_0$ and acceleration statistic $\hat{a}$, we need to use these to select which of our B (B=10,000 in our examples) bootstrapped values to take as the end points in our 100*(1-2α) (α=0.025 for a 95% CI) confidence interval.

This is done in four simple steps:

1. Calculate $$a_1 = \Phi\left(\hat{z}_0 + \frac{\hat{z}_0 + \Phi^{-1}(a)}{1-\hat{a}\left(\hat{z}_0 + \Phi^{-1}(a)\right)}\right)$$

   and $$a_2 = \Phi\left(\hat{z}_0 + \frac{\hat{z}_0 + \Phi^{-1}(1-a)}{1-\hat{a}\left(\hat{z}_0 + \Phi^{-1}(1-a)\right)}\right)$$

   Where $\Phi(x)$ is the standard normal cumulative distribution function (PROBNORM function in SAS). and as before, $\Phi^{-1}(x)$ indicates the inverse function of a standard normal distribution (PROBIT function).

2. Calculate $N_1 = \mathrm{B} \times a_1$ rounded down to the nearest integer

and $N_2 = \mathrm{B} \times a_2$ rounded up to the nearest integer

3. Order the calculated statistic (difference between the means in our example) calculated from the B bootstrapped samples (as already created to calculate the bias-correction estimate) from lowest to highest.

4. Take the N_1th sample as the lower limit of the confidence interval and the N_2th sample upper limit of the confidence interval.

Again, the equations may look a little scary, but in practice this is not difficult to do. Assuming that we have already run the code above to create our bias correction and acceleration statistics (and so created the data sets DIFFMEAN, BIAS and ACCEL), the SAS code required is as follows:

Steps 1 and 2: Calculate values of a_1 and a_2 and use to calculate N_1 and N_2:

```
data ciends;
    merge accel
          bias;

  part1=(bias + probit(0.025)) / (1 - (accel*(bias + probit(0.025))));
  part2=(bias + probit(0.975)) / (1 - (accel*(bias + probit(0.975))));

  alpha1=probnorm(bias + part1);
  alpha2=probnorm(bias + part2);

  n1=alpha1*10000;
  n2=alpha2*10000;

  call symput('n1', put(floor(n1), 5.));      /* Create macro variables with values */
  call symput('n2', put(ceil(n2), 5.));       /* of N1 and N2 for later use         */

 run;
```

Step 3: order the bootstrap sample values of DIFFMEAN

```
proc sort
   data=diffmean;
  by diffmean;
run;
```

Step 4: select the ends of the confidence interval using the N_1 and N_2 values

```
data ci_bca;
    set diffmean end=eof;

  retain conf_lo conf_hi;
  if _n_=&n1 then conf_lo=diffmean;            /* select values for upper and lower */
  if _n_=&n2 then conf_hi=diffmean;            /* limits using N1 and N2 values     */

  if eof then output;

  keep conf_lo conf_hi;
run;
```

For our data, a bias correction of 0.1135 and an acceleration of -0.0238 gives us values of $\alpha_1 = 0.03455$ and $\alpha_2 = 0.98167$. So with 10,000 bootstrap samples we would take the 345th value as the lower CI limit and the 9817th value as the upper CI limit.

A print out of the final data set ci_bca would give us the following:

```
conf_lo    conf_hi
 -17.5      216.4
```

Giving us a confidence limit of (-17.5, 216.4) for the difference in the means between the groups.

COMPARISON OF CONFIDENCE INTERVALS

We now have three different ways of calculating bootstrap confidence intervals. But how do they compare? The author has stated that the BCa method is thought to address some of the identified issues with the bootstrap-t method and the percentile method – but how can we be sure that this is the better method?

To show the differences, the author created a simulated data set with 100 data points from a log-normal distribution, X. This distribution has the property that if we took the Y=log(X), then the values would come from a normal distribution.

In this case, we chose the distribution such that log(X) = Y ~N(2, 4) i.e. so that the log of the values has mean(μ)=2 and variance(σ^2)=4 (and so standard deviation(σ)=2). The X values take the variable name CHANGE (to simulate a change from baseline value) and so the Y values are LOGCHG. The simulation was performed using the RANNOR() function which is generates random values from a N(0, 1) distribution.

To do the random sampling within SAS you would use the following code:

```
data original;
   do pid=1 to 100;              /* create 100 observations                        */
      logchg=(rannor(0)*2 + 2);  /* simulate LOGCHG as normal with mean=2 and sd=2 */
      change = exp(logchg);      /* if LOG(CHANGE)=LOGCHG then EXP(LOGCHG)=CHANGE  */
      output;
   end;
run;
```

The log-normal distribution is a nice one to use as an example as it is very highly skewed and a confidence interval calculated with normal theory methods would be highly suspect. However, some statistical theory has been formulated to give a reliable estimate for the confidence interval for the mean (modified-Cox method - see Zhao and Gou (1997)[2]).

The modified-Cox method for calculating a confidence interval for the mean was then used to compare the three methods for bootstrap confidence intervals (10,000 iterations) with the following results:

Table 2: Comparison of bootstrap-confidence intervals for mean of log-normal data

Method	**Confidence Interval**
Modified-Cox	(25.5, 92.3)
Bootstrap-t	(23.9, 137.2)
Percentile	(18.8, 71.6)
BCa	(24.4, 101.7)

If we take the modified-Cox as the ‘true’ value for the confidence interval, we can see that the BCa method is by far the closest to this true value. The bootstrap-t method is relatively close on the lower limit of the confidence interval but is almost 50% larger at the upper limit than the modified-Cox value. The percentile interval gives limits which are too low at both ends – most importantly at the upper end showing that it does not cover the true interval at all. In comparison, the BCa method covers the interval given by the modified-Cox method and is only slightly more conservative at both ends.

A CAUTIONARY NOTE

In the non-bootstrap case, a parametric estimate accepts the possibility that a sample can be non-representative by random occurrence. A bootstrap depends on the sample being representative. If the sample is not representative, the conclusions will be completely inappropriate. It is very important to be confident that your sample is a good representation of the whole population from which you are sampling.

CONCLUSION

The bootstrap method can be generalised into these three steps

1. Create the bootstrap replications by re-sampling with replacement (replicating independently with two (or more) samples, such as for different treatment groups)
2. Calculate the statistic of interest for each sample
3. Use the distribution for the replicated statistic to make inferences.

Despite being a relatively new methodology, the bootstrap is quickly becoming a regular part of the clinical statistician's arsenal. It allows the statistician to assess the statistical accuracy of complicated procedures, by exploiting the power of the computer. Its aim is to carry out familiar statistical calculations such as standard errors, biases and confidence interval in an unfamiliar way – by purely computational means rather than through the use of statistical formulae.

The bootstrap relieves the statistician from having to do complex mathematical derivations, or in some instances, provides an answer where no analytical solution is possible. It allows us to assess the properties of a statistical procedure for the data at hand. The bootstrap is a fairly crude form of inference but it can be used when the statistician is unable or unwilling to carry out extensive modelling. For large samples, it gives accurate answers no matter what the underlying population.

Using bootstrapping techniques within the SAS system is relatively easy. The RANUNI function together with the POINT= option within the SET statement of a DATA STEP makes the creation of bootstrap samples a simple process. The BY processing allowed within most of the procedures available in SAS allows us to utilise efficient programming techniques, which can cut down on the computational time required.

Although we have only shown a brief introduction here, the world of the bootstrap is large and expanding. As well as calculating standard errors and confidence intervals, bootstrapping methodology is available for regression, hypothesis testing, adaptive estimation, calibration and bioequivalence techniques.

REFERENCES

1. Committee for Medicinal products for Human Use (CPMP), Guideline for Clinical Trials in Small Populations (Draft). Release for Consultation: March 2005. http://www.emea.eu.int/pdfs/human/ewp/8356105en.pdf

2. Efron and Tibshirani (1993), An Introduction to the Bootstrap, Chapman &Hall/CRC (IBSN: 0-412-04231-2).

3. Rudolf Erich Raspe (1786), The Singular Travels, Campaigns and Adventures of Baron Munchausen.

4. Zhou, X-H., and Gao, S. (1997), "Confidence intervals for the log-normal mean," *Statistics in Medicine*, 16, 783-790.

ACKNOWLEDGEMENTS

With many thanks to Katherine Hutchinson at OPS for time, encouragement and a thorough review,

RECOMMENDED READING

An Introduction to the Bootstrap: Efron and Tibshirani, Chapman &Hall/CRC 1993 (IBSN: 0-412-04231-2).

This book gives an excellent introduction to the methodology of the bootstrap. It is very readable and gives good examples throughout.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Nancy Barker
Oxford Pharmaceutical Sciences Ltd
The Stables, 114 Preston Crowmarsh,
Wallingford
OXON OX10 6SL
UK
Work Phone: +44 (0)1491 821680
Fax: +44 (0)8704 580729
Email: Nancy.Barker@ops-web.com