# 1 Introduction

## 1.1 Abstract

Re-sampling based bootstrap methods are well-documented in statistics literature. However relatively little is written on the fully enumerated bootstrap. Fully enumerated bootstraps are in effect a bootstrap operation where the bootstrap is carried on an infinite amount of time. In a fully enumerated bootstrap the result is a completely enumerated distribution of all possible outcomes with associated probabilities. There are many limiting factors that keep one from the fully enumerating a bootstrap. However in the cases were full enumeration is possible the advantage of the elimination of re-sampling error presents itself. In this paper we will use A Probability Programming Language site 2017 to fully enumerate bootstraps of certain random variable algebraic operations.

This paper will explore the current computational limitations of fully enumerated bootstraps for convolutions of random variables as well as some other random variable algebraic operations. Furthermore we will investigate I know some called the pure fully enumerated bootstrap (PFEB) in which there are no coincidental repeat values in the algebra. We will also give applications where a fully enumerated bootstrap discrete random variable can Model a true but unknown continuous random variable. These these discrete models for continuous phenomenon have the advantage of not needing a parametric assumption for distribution fitting. Also these discrete models will eliminate re-sampling error.

## 1.2 Dice Example

Consider a game of Crabs, a very well known gambling game, if we wanted to analyze possible strategies to help us win we could begin by investing possible outcomes of rolling a pair of dice. First, consider a single standard die consisting of six sides, 1, 2, 3, 4, 5 and 6. The probability of rolling the die and getting any of these six numbers is $\frac{1}{6}$. This is a simple discrete distribution which contains one distinct probability and six possible outcomes. If we were to sum two dice, also know as rolling two dice, the possible outcomes and probabilities change significantly. There are now 11 possible outcomes, they are 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, and 12. In addition, there are six distinct probabilities of getting these outcomes. They are

$\frac{1}{36}, \frac{1}{18}, \frac{1}{12}, \frac{1}{9}, \frac{5}{36}, \frac{1}{6}$. These probabilities differ since some values can be summed by various combination of two dice. For example, 6 can be the result of adding 2 from the first die with a 4 from the second dice, as well as adding up two 3s. Whereas a 4 is only produced from rolling two 2s. Therefore, the probability of the dice summing to a six will be higher than the probability of it summing to a four. It can be seen that certain numbers are more likely to be rolled, while others are less likely. This idea is extremely applicable for predicting the outcome of a gambling game involving two dice.

If we created a new set of dice which still had six sides, but instead of having the numbers 1, 2, 3, 4, 5 and 6, the dice had the numbers, 1.1, 2.01, 3.001, 4.0001, 5.00001, and 6.000001. The values are altered by a tenth or less than the original dice, yet if two dice of this set are summed they produce very different results. When these dice are summed they produce distinct sums that don't overlap to be the same number such as before. Instead we get a total of 21 different outcomes and only 2 distinct probabilities. The two distinct probabilities are $\frac{1}{36}$ and $\frac{1}{18}$. The first would be if we rolled the same value for both dice, such as 1.1 with 1.1. This would give us 2.2, which will occur a single time a single time for our fully enumerated bootstrap. Thus, the probability of rolling a 2.2 is 1 out of 36. The other outcome is if two different numbers are rolled such as 1.1 and 2.01 which would give us 3.01. We could get this sum by rolling at 1.1 fist and a 2.01 second or vice versa, thus the probability of getting a sum of 3.11 is 2 out of 36.

If we look at our two sets of dice and their results, the set of standard dice produced 11 outcomes and 6 probabilities. This distribution has a mean of 7 and a standard deviation of $\approx 2.415$. The altered dice produces nearly double the amount of outcomes, 21 and only two probabilities. The mean of the distribution is 7.037 and the standard deviation is $\approx 2.379$. Therefore we can see that these samples have less than fourth hundredths of a difference in their means and less than a tenth difference in their standard deviations. Yet, we can clearly see that the small difference between the two sets of dice changes our outcome elements and their probabilities significantly. Thus, we classify the altered set of die as a Pure Fully Enumerated Bootstrap. Due to the fact no elements are repeat values and no summations of the elements accidentally sum to the same value. This pure fully enumerated bootstrap produced 21 outcomes and 2 distinct probabilities. Both of which are the correct number of outcomes and probabilities we have calculated for a set of 12 elements and 2 convolutions. Those numbers will be explained further on in the paper. We can discern that the standard

set of dice is a non-pure fully enumerated bootstrap, or rather just simply a fully enumerated bootstrap. The accidentally repeat summations causes the number of outcome elements to be significantly less and thus the number of distinct probabilities has increased.

From this example we have showed the initial start of discerning between pure and non-pure fully enumerated bootstraps and the effect various attributes of our sample have on its outcome element and distinct probabilities. We also can consider the importance of applying fully enumerated bootstrapping on convolutions and averages. The rolling of dice is just one example of the application. We could also apply this to the idea of back-up generators. If we consider a generator with a certain life span and we were able to use it until it ran out of energy, and then use another one of the same model until it ran out and so on for a certain number of generators. These methods would help us investigate when our generators would run out of electricity, using the distribution of time the generators on average run for. We not only will discuss convolutions but also reliability block diagrams and simply discuss fully enumerated bootstrapping for discrete random variables.

# 2 Literature Review

## 2.1 What Has Been Done Before

## 2.2 What Is New

# 3 Fully Enumerated Bootstrapping

Define Fully Enumerated Bootstrap: A fully enumerated bootstrap is when a sample is bootstrapped until it produces a completely enumerated distribution of possible outcomes and their associated probabilities to eliminate re-sampling error. Define Pure Fully Enumerated Bootstrap for Convolutions: A sample convoluted n number of times becomes a pure fully enumerated bootstrap if the operation produces the complete set of possible outcome elements from the sample set and the correct number of distinct probabilities of those outcomes. The sample set will need to have a significant number of digits and not repeat digits in order to achieve this.

# 4 Pure vs. Non-Pure Fully Enumerated Bootstrapping

## 4.1 Sum of Digits

As we continued to explore how to achieve a pure fully enumerated bootstrap for a convolution, we saw our results varied based on the number significant digits our sample elements contained. To explore this idea further, we created a random sample set representing an exponential function with a mean of $\frac{1}{10}$. The set contained 25 total elements and we convoluted this set four times. We began with each element of our set containing 13 significant digits after the decimal. We conducted a fully enumerated bootstrapping on the sample and plotted the PDF of the enumerated distribution. We then determined the total number of outcome elements and the number of distinct probabilities. We hoped to find a pattern as we decreased the number of significant digits, eventually rounding our elements to only two significant digits after the decimal. Below we have the 12 graphs, which range anywhere from 2-13 significant digits, and are labeled below with the number of outcome elements and distinct probabilities of each.

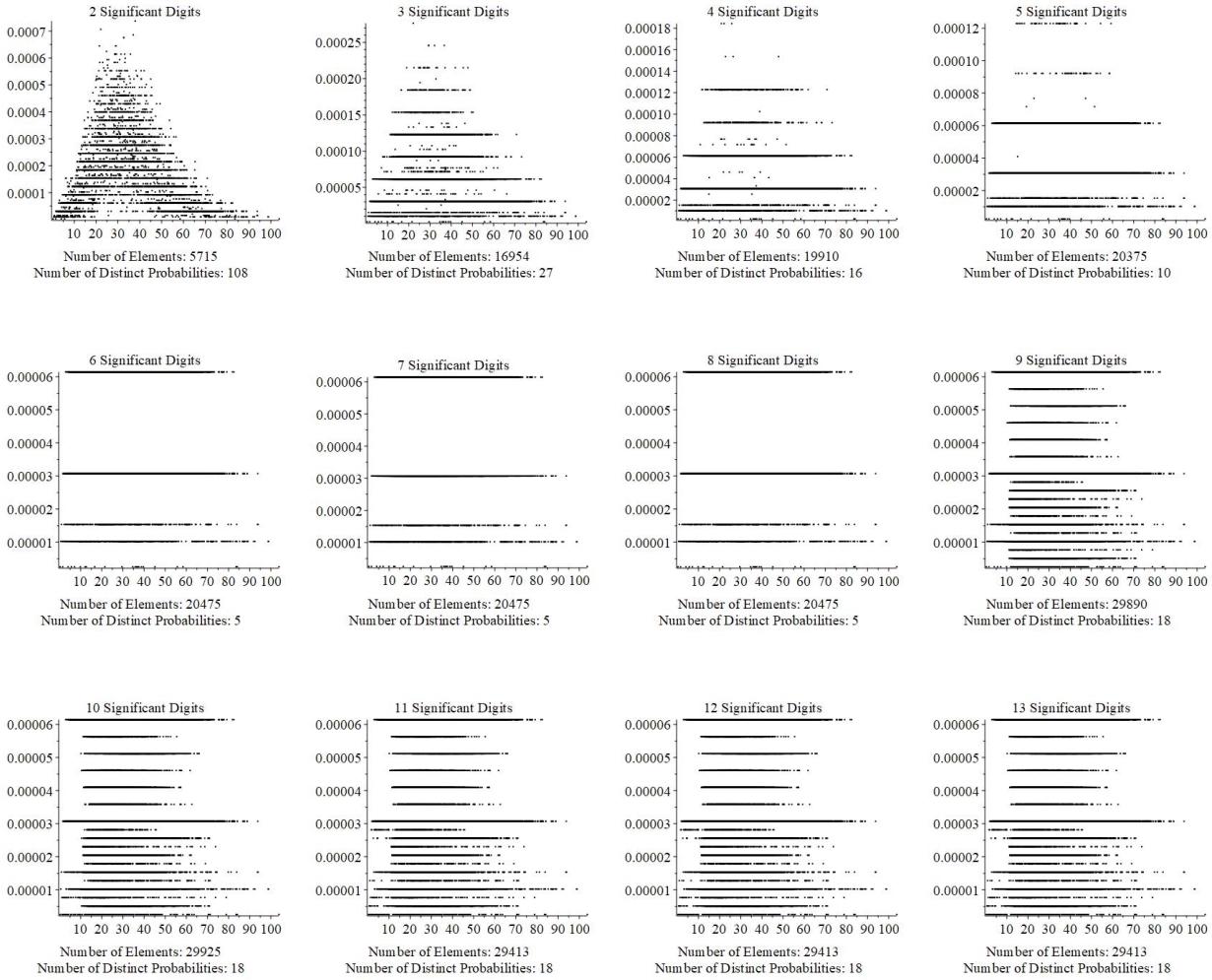


Figure 1: PDFs of the fully enumerated bootstraps of our distribution with the set varying in significant digits

At first glance you can see as we move through the graphs there is a flow from less concentration of dots, to very clear lines, back to less concentration of dots. If there are only two or three significant digits the set isn't precise enough to denote when two numbers actually sum to different things and instead recognizes them as the same sum. Thus, we get a lot less outcome elements, for example with only two significant digits the sample had only 5715 outcome elements. While based on the formula we will prove below the set with 25 elements and four convolutions is a pure fully enumerated bootstrap when it produces 20475 outcome elements. The number of probabilities also is significantly greater with only two significant digits. The fully enumerated bootstrap has 108 distinct probabilities and it should only have 5 if it was pure.

There is also a trend that as the significant digits become too great such as twelve or thirteen we see that numbers that should be summing to the same number are being evaluated as two separate sums, thus creating many more outcome elements than expected. Now, the set has 29413 sample outcome elements which is significantly higher than the number of outcome elements expected, 20475. However, we do see that the fully enumerated bootstrap with thirteen significant digits has higher number of distinct probabilities, much like the sample with two few digits, in this case it is only 18 rather than 5.

Finally, we can see there is a range of significant digits that produced pure fully enumerated bootstraps. Those are the PDFs for digits six, seven and eight, where there are five lines visible on the graphs representing the five distinct probabilities. In addition, the graph shows a lack of outliers which means they also had the correct number of outcomes. These significant digits display the difficulty it is to acquire a pure fully enumerated bootstrap.

## 4.2 Do we want Pure Fully Enumerated Bootstraps?

As we mentioned earlier, the variation of significant digits has an influence on producing pure or non-pure fully enumerated bootstrapping. In this section, we will take a look at the difference between pure and non-pure fully enumerated bootstrap. Also, we will further explore if pure fully enumerated bootstrap is better than the non-pure fully enumerated bootstrap and do we really need to perform pure fully enumerated bootstrap all the time.

x1= Z~Exponential(1/10,0), x1=x2=x3
x1+x2+x3= Z~Erlang(1/10,3)

| Digits | Mean (μ) | Stdv.(σ) | % of error (μ) | % of error (σ) |
|---|---|---|---|---|
| 1 | 34.00000009 | 13.31453319 | 0.000735813963326 | 0.00032578584024 |
| 2 | 33.97439982 | 13.31743181 | 0.000017689182801 | 0.00010815330072 |
| 3 | 33.97504007 | 13.31873898 | 0.000001155555528 | 0.00001000910566 |
| 4 | 33.97491375 | 13.31886910 | 0.000002562472345 | 0.00000023950977 |
| 5 | 33.97499829 | 13.31886853 | 0.000000074172184 | 0.00000028230618 |
| 6* | 33.97500081 | 13.31887229 | 0 | 0 |
| 7 | 33.97500123 | 13.31887210 | 0.000000012362031 | 0.00000001426547 |
| 8 | 33.97500123 | 13.31887206 | 0.000000012362031 | 0.00000001726873 |
| 9 | 33.97500167 | 13.31887142 | 0.000000025312729 | 0.00000004805212 |
| 10 | 33.97500165 | 13.31887146 | 0.000000024724061 | 0.00000006231759 |
| 11 | 33.97500197 | 13.31887049 | 0.000000034142751 | 0.00000013514658 |
| 12 | 33.97500197 | 13.31887049 | 0.000000034142751 | 0.00000013514658 |
| 13 | 33.97500197 | 13.31887049 | 0.000000034142751 | 0.00000013514658 |

**Table 1. Fully enumerated bootstrapping μ and σ for exponential distribution with different digits when the sample size n = 25 and the number of convolution m = 3 as shown. The fourth column shows the percentage of difference of mean between Digit 6 and other digits. The fifth column indicates the percentage of difference between Digits 6 and other digits in case of standard deviation.**

Table 1 illustrates a comparison between pure and non-pure fully enumerated bootstrap. We chose the digits six to compare with other digits since it produces the pure fully enumerated bootstrap which proven by having the correct number of total number of elements and the distinct probabilities. Rest of the significant digits produce non-pure fully enumerated bootstrap. Comparing between pure and non-pure fully enumerated bootstrap, the percentage of error will first decrease with the increasing of the digits. Then, after pass a certain digit like digit 6, the percentage of error start to increase again. However, the percentage of error on both the estimate of mean and the standard deviation is fairly small. Most of the percentage of the error is around 10 to the negative 8th , which are small enough to ignore. We can conclude that variation of digits would not induce a significant error on estimate.

x1= Z~Exponential(1/10,0), x1=x2=x3
x1+x2+x3= Z~Erlang(1/10,3)
True value: μ=30, σ= 17.32050808

| Digits | Mean (μ) | Stdv.(σ) | % of error (μ) | % of error(σ) |
|---|---|---|---|---|
| 1 | 34.00000009 | 13.31453319 | 0.1333333363 | 0.2312850679 |
| 2 | 33.97439982 | 13.31743181 | 0.132479994 | 0.231117716 |
| 3 | 33.97504007 | 13.31873898 | 0.1325013357 | 0.2310422465 |
| 4 | 33.97491375 | 13.31886910 | 0.132497125 | 0.231034734 |
| 5 | 33.97499829 | 13.31886853 | 0.132499943 | 0.2310347669 |
| 6* | 33.97500081 | 13.31887229 | 0.132500027 | 0.2310345498 |
| 7 | 33.97500123 | 13.31887210 | 0.132500041 | 0.2310345608 |
| 8 | 33.97500123 | 13.31887206 | 0.132500041 | 0.2310345631 |
| 9 | 33.97500167 | 13.31887142 | 0.1325000557 | 0.2310346 |
| 10 | 33.97500165 | 13.31887146 | 0.132500055 | 0.2310345977 |
| 11 | 33.97500197 | 13.31887049 | 0.1325000657 | 0.2310346537 |
| 12 | 33.97500197 | 13.31887049 | 0.1325000657 | 0.2310346537 |
| 13 | 33.97500197 | 13.31887049 | 0.1325000657 | 0.2310346537 |

Table 2. The comparison between fully enumerated bootstrapping estimate of μ and σ and the true value for exponential distribution with different digits when the sample size n = 25 and the number of convolution m = 3 as shown.

Referring to Table 2, there is a difference between fully enumerated bootstrap estimate of mean and the true mean, which possibly introduces by the sampling error. Same thing happens for the standard deviation. However, the main point of this table is to investigate if pure fully enumerated bootstrap will have a smaller error rate in comparison to non-pure fully enumerated bootstrap. It is too see if pure fully enumerated bootstrap is a better estimate, which does not induce measurement error. Looking at the error rate between pure and non-pure fully enumerated bootstrap, the difference is small enough to ignore. Combine with Table 1, the observation implies that the error between pure and non-pure fully enumerated bootstrap is not significant in terms of the estimate of mean and standard deviation. The induced measurement error while using the non-pure fully enumerated bootstrap would not hurt the accuracy of the result.

Hence, in real life application, we can choose to use either pure or non-pure fully enumerated bootstrap depends on the sample data. We do not have to force the sample data has certain significant digits and

other elements to reach the requirement for pure fully enumerated bootstrap. A sample data has only one or two significant digits will fit in the non-pure fully enumerated category. If we use non-pure fully enumerated bootstrap, there usually are fewer digits to process in order to generate the estimate. Fewer digits would allow the computer to calculate the estimate easier because it take significant less time to run through the bootstrap. This finding overall implies a wider and bigger application for the fully enumerated bootstrap.

## 4.3 Proof

**Theorem:** In a fully enumerated bootstrap, the number of ways of choosing $m$ elements from a set of $n$ elements, with replacement, is given by:

$$\binom{n+m-1}{m}. \tag{4.1}$$

**Proof**: Let the number of ways of distributing $m$ elements from a set of $n$ elements be denoted by $w(n, m)$. Let's start with the simple case of $w(n, 1)$. We have the set of $n$ elements given by $(x_1, x_2, ...., x_n)$ . There is only one place out of the set of these $n$ elements. Therefore, there are $n$ ways we could do this. This is because we could put any of the $n$ elements from this set into this one place. Now consider $w(n, 2)$. Let's say, then, we have a set of $n$ elements and only two places to put them, with replacement. How many ways can this be done? The situation is shown below:

$$(x_1, x_2, ...., x_n) \rightarrow (), ().$$

The possibilities are: put same elements in both places: $(x_1, x_1), (x_2, x_2), (x_3, x_3), ...., (x_n, x_n)$. Or, put different combinations $(x_1, x_2)$, $(x_1, x_3)....,(x_1, x_n)$ or $(x_2, x_3), ....., (x_2, x_n)$ or $(x_{n-2}, x_{n-1}), (x_{n-2}, x_n)$ or $(x_{n-1}, x_n)$. Therefore, the problem has been divided into two parts. What are the number of ways of putting in same elements, and what are the ways of putting in two different elements. The number of ways of putting in two same elements is clearly $n$. Choose any one element, put it in one place. Pick this same element again. And we can do this for all the $n$ elements in the list. Hence, there are $\binom{n}{1}$ ways. The number of ways of choosing two different elements is also straightforward. This is basically choosing in any two elements out of a list of $n$ elements without replacement. Therefore, the number of ways to do this is $\binom{n}{2}$. Therefore, the total number of ways is given by:

$$\binom{n}{1} + \binom{n}{2} = \binom{n+1}{2}. \tag{4.2}$$

Let's look at the last case of $w(n, 3)$ and then prove our result without any loss of generality. The number of different arrangements are outlined below:

(1) Put the same element in all the three places. Number of ways: $\binom{n}{1}$.

(2) Put all three different elements in the three different places. Number of ways: $\binom{n}{3}$.

(3) Put two same elements, one different element. Number of ways: $n \times \binom{n-1}{1} = 2 \times \binom{n}{2}$.

Therefore, the total number of ways is given by:

$$\binom{n}{1} + \binom{n}{2} + \binom{n}{2} + \binom{n}{3} = \binom{n+1}{3} + \binom{n+1}{2} = \binom{n+2}{3}. \tag{4.3}$$

By a similar logic, one can construct $w(n, 4)$. However, a pattern has started to emerge. The pattern is shown below:

$$\begin{aligned} w(n,1) &= \binom{n}{1} \\ w(n,2) &= \binom{n+1}{2} \\ w(n,3) &= \binom{n+2}{3} \\ w(n,4) &= \binom{n+3}{4} \end{aligned} \tag{4.4}$$

However, we are not sure whether this general pattern will continue, and that it will actually converge to $\binom{n+m-1}{m}$. We prove this using induction. Having already proven the base step $w(n, 1) = n$, we proceed top the induction step. Assume

$$w(n,m) = \binom{n+m-1}{m} \tag{4.5}$$

Is true. Prove:

$$w(n, m+1) = \binom{n+m}{m+1}. \tag{4.6}$$

Note that, using Pascal's identity:

$$\begin{aligned} w(n,2) &= \binom{n+1}{2} = \binom{n}{1} + \binom{n}{2} = w(n,1) + \binom{n}{2}. \\ w(n,3) &= \binom{n+2}{3} = \binom{n+1}{2} + \binom{n+1}{3} = w(n,2) + \binom{n+1}{3}. \\ w(n,4) &= \binom{n+3}{4} = \binom{n+2}{3} + \binom{n+2}{4} = w(n,3) + \binom{n+2}{4}. \end{aligned} \tag{4.7}$$

Therefore:

$$w(n, m+1) = w(n,m) + \binom{n+m-1}{m} = \binom{n+m-1}{m} + \binom{n+m-1}{m+1} = \binom{n+m}{m+1}. \tag{4.8}$$

Therefore, by Principle of Mathematical Induction, we have proved that:

$$w(n,m) = \binom{n+m-1}{m}. \tag{4.9}$$

## 4.4 Alternative Proof (most of this is verbatim lift, so cite carefully): https://artofproblemsc and-urn

This problem is equivalent to the common ball and urns problem. How many ways can one distribute $k$ indistinguishable objects into $n$ bins? We can imagine this as finding the number of ways to drop $k$ balls into $n$ urns, or equivalently to drop $k$ balls amongst $n-1$ dividers.

If you could only put one ball in each urn, then there would be $\binom{n}{k}$ possibilities; the problem is that you can repeat urns, so this does not work. You can, however, reframe the problem as so: imagine that you have the

$n$ urns (numbered 1 through $n$) and then you also have $k-1$ urns labeled "repeat 1st", "repeat 2nd", ..., and "repeat $k-1$-th". After the balls are in urns you can imagine that any balls in the "repeat" urns are moved on top of the correct balls in the first $n$ urns, moving from left to right. There is a one-to-one correspondence between the non-repeating arrangements in these new urns and the repeats-allowed arrangements in the original urns.

For a simple example, consider 4 balls and 5 urns. The one to one correspondence between several of the possibilities and the "repeated urns" version is shown. Since there are 4 balls, these examples will have three possible "repeat" urns. For simplicity, I am listing the numbers of the urns with balls in them, so "1,1,2,4" means 2 balls in urn $1, 1$ in urn 2, and 1 in urn 4. The same is true for the "repeat" urns options but I use the notation $r_1$ etc.

1,2,3,4 $\longleftrightarrow$ 1,2,3,4 (no repeats). 1,1,2,4 $\longleftrightarrow$ 1,2,4, $r_1$. 1,2,2,2 $\longleftrightarrow$ 1,2, $r_2$, $r_3$. 4,4,5,5 $\longleftrightarrow$ 4,5, $r_1$, $r_2$. Since the re-framed version of the problem has $n+k-1$ urns, and $k$ balls that can each only go in one urn, the number of possible scenarios is simply $\binom{n+k-1}{k}$.

## 4.5 Probability Chart

As we have mentioned throughout this chapter, when classifying pure vs. non-pure fully enumerated bootstrap there are two distinct qualities to consider, the first is the number of outcomes and the second is the number of distinct probabilities. The formula proved above, will calculate that amount of outcomes for any sample size with any number of convolution. Unfortunately, we were unable to find the exact combinatoric formula for determining the number of distinct probabilities for any sample size with any number of convolution. However, we were able to discern some patterns in these values and produce code that helped us determine the values for up to 10 convolutions.

Let's start with a simple example to help illustrate the methods we used to understand th3 values for distinct probabilities. If we consider a single sample set of four that is convoluted four times. Let's consider this set to be made up of **a**, **b**, **c**, and **d**. Each letter has a distinct value that doesn't lead to any accidental repeat sums. Now we will consider the five different combinations of elements which will generate the five

total probabilities for a set of four elements, convoluted four times.

We start out by classifying the possible combinations of our four elements. For many of these combinations there are a variety of letters (elements) we could have chosen to illustrate the particular probability. However, it is important to note that these probabilities will be produced by a variety of outcomes, what differs from each probability is the formula it takes to build it. Thus producing the distinct probability, that many different values can have.

First, it is possible for us to sum **a**, **b**, **c** and **d**. This would mean no matter what order we place our elements in, it will equal the sum of $a+b+c+d$. To illustrate this idea better we could add $a+b+c+d$ or we could add $b+c+d+a$ and both would be considered separate possible convolutions out of our 256 possible convolutions but they would add up to the same summation. Thus to get all the possible combination of the four elements added together we would consider 4! which is 24. Thus the probability of getting four different elements added together would be $\frac{24}{256}$. The second distinct probability comes from the addition of one element added twice and then two other elements. Thus the formula would look something like this $a+a+b+c$. This specific formula can be produced by 12 different combinations of those letter. Thus it produces the probability $\frac{12}{256}$. We could have chosen **d**, **d**, **c**, and **a** and got the same probability as it isn't the letters each time that are important but the template of how to build the combinations of the letters. The third distinct probability comes from the addition of three of the same element and one different element. This one can only be produced by four different convolutions. For example if we add **a**, **a**, **a** and **b**, we can write it four different ways by changing the spot the **b** is placed. Therefore, the probability of the outcome being one element added together three times along with another different element is $\frac{4}{256}$. The fourth distinct probability is that we add four of the same element. This is a pretty simple probability as it can only happen once thus it has a probability of $\frac{1}{256}$. The final probability is produced when we have two sets of doubles, more specifically we add the same element twice and then add another element twice. For example **a**, **a**, **b** and **b**. This formula can be produced in 6 different ways, thus the probability of summing two sets of double elements is $\frac{6}{256}$.

These are the only combinations for a convolution of four elements. Thus a convolution of four with at least four elements in the sample set, has five distinct probabilities. We checked the result with the code

that will be shown below. As you can see we had a general idea of how to build the possible probabilities based on a formula of building letter combinations. We will discuss more of the general patterns for these probabilities after we have introduced the code and table.

Below is a picture of the code used in Maple with the APPL code extended to generate an ideal sample to test the number of distinct probabilities. This code also used the determinable value of number of outcomes for a convolution to check our answer. This sample helped us produce the five probabilities we discussed above but it also helped us to see a general pattern for the number of convolutions not only to produce a pure fully enumerated bootstrap but also the number of probabilities when we have less than the desired number of elements or the ones with extra elements. Here is the code:

```
for ii from 13 to 13 do
  for jj from 5 to 5 do
  Xf:=[];
  Digits := 5;
    for kk from 1 to ii do
      Xf:= [op(Xf), IDF(X,RNG())];
    od;
    Xfs:=BootstrapRV(Xf);
    Digits :=15;
    Xconv :=ConvolutionIID(Xfs, jj):
    if(binomial(ii+jj-1,jj) <> nops(Xconv[1])
) then
      print(errorat, ii,jj)
    fi;
    #appendto("c:/glenstuff/output/setstuff7.
txt");
    print(ii,jj,nops(Xconv[2]), nops(convert
(Xconv[1],set)));
    appendto(terminal);  od;  od;
```

Figure 2: Code Used to generate the table of distinct probabilities

This code was used since it created a simple yet effective sample that would be most likely to produce the correct number of probabilities by limiting accidental sums. This code also checked the number of probabilities produced with the number of outcome elements that were generated as a way to check if the distinct probabilities were correct. Next, here is the table of probabilities we have generated for the number of distinct probabilities for up to 10 convolutions with at most 10 sample elements, using the code from above.

| Number of Convolutions | Number of Elements | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 3 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 4 | 3 | 4 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 5 | 3 | 5 | 6 | 7 | 7 | 7 | 7 | 7 | 7 |
| 6 | 4 | 7 | 9 | 10 | 11 | 11 | 11 | 11 | 11 |
| 7 | 4 | 8 | 10 | 12 | 13 | 14 | 14 | 14 | 14 |
| 8 | 5 | 9 | 14 | 16 | 18 | 19 | 20 | 20 | 20 |
| 9 | 5 | 12 | 17 | 21 | 23 | 25 | 26 | 27 | 27 |
| 10 | 6 | 13 | 21 | 26 | 30 | 32 | 34 | 35 | 36 |

Figure 3: Table of distinct probabilites for pure fully enumerated bootstraps

Many of the patterns we found in regards to the number of distinct convolutions can be seen by the distinct colors on the table. The immediate pattern that we have found and discerned is that for a fully enumerated bootstrap to be pure it must have as many numbers in its set as the number of convolutions. Those numbers are denoted in the blue boxes on the table. If the distribution doesn't contain enough elements than not all of these probabilities can be produced. For example, if a convolution of six doesn't have at least six elements in the sample set, then there is no way to generate the probability of six different elements added together since we don't have six separate elements to add. If we refer to the example above this was the value that generated a probability of $\frac{24}{256}$ for a convolution of four.

This idea brings us to the next two patterns we were able to find in the table. The first is denoted by the green boxes, which are to the left of the blue boxes in each row. We found that if you have one less element than the number of convolution in your sample set, then the pure fully enumerated bootstrap will have one less distinct probability. This is due to the fact that without the same number of sample elements as convolutions we cannot generate the probability for a the convolution of all different elements. With one less element than convolutions we will always have at least one double in the summation. The next pattern is denoted by the purple and it is to the left of the green boxes, also two to the left of the blue boxes. This pattern is very similar to the green boxes. However it is caused when a sample set has two less elements then the number of convolution. This is due to the set not being able to produce a convolution of all different elements and a convolution with two double elements and all random elements. The set must have three or more of the same element or some combination of doubles, triples, quadruples, etc. Thus these boxes have

two less distinct probabilities than the amount for a pure fully enumerated of that number of convolutions.

The final pattern that can be seen in this table is in the first row which is denoted by the yellow boxes. As you follow the row of two convolutions from a sample set of two elements all the way to ten elements the value of distinct probabilities remains at 2. Although, it is not shown in this table using the code above we were able to follow this out all the way to 15 elements and it still produced only two distinct probabilities. We were not able to calculate these values for the rest of the table as the code was unable to produce adequate results. We conjecture that the number of distinct probabilities for a specific convolution's pure fully enumerated bootstrap does not change once the set has the same or more elements than the number of convolutions.

We were unable to make more than conjectures about the combinatoric formula to calculate the value of distinct probabilities for a number of convolutions. We found that for each convolution there was a pattern of how distinct probabilities were built. It began with a probability for all different elements, then one for two of the same elements and the rest random. Then one with three of the same elements and the rest random. Then one with four of the same elements and the rest random and so on. Until, the formula summed all the same element for each convolution. Then the rest was any possible combination of doubles, triples, quadruples, etc. that are possible for that number of convolutions. As this pattern emerged we thought we had found the correct pattern to build a formula off of. Unfortunately we discovered that the probabilities started to overlap one another leading to unpredictable repeats that decreased the number of distinct probabilities. We were unable to predict when the probabilities overlapped for each value of convolutions. Thus it made it difficult to find the combinatoric formula for these distinct probabilities. We hope to do further research to find this formula in the future.

## 4.6 Non-Repeat Measures in X

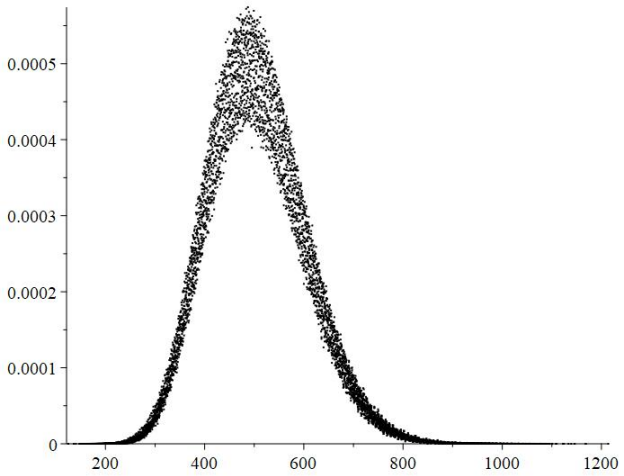


Figure 4: PDF of the fully enumerated bootstrap using ball bearing dataset.

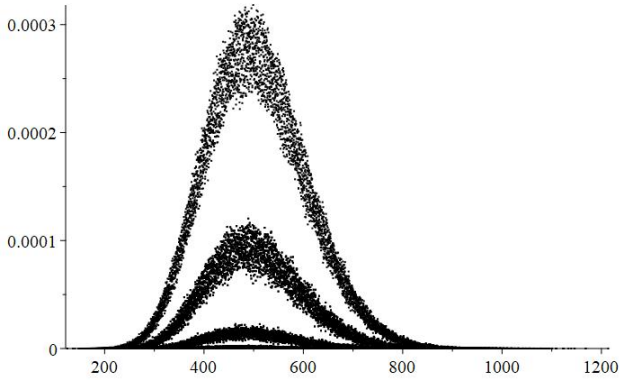


Figure 5: PDF of the fully enumerated bootstrap using one-element-modified ball bearing dataset.

From Figure 4 and Figure 5, we saw two very different distributions. However, both graphs are using the ball bearing dataset that automatically generate by maple. The dataset is "[17.88, 28.92, 33.00, 41.52, 42.12, 45.60, 48.48, 51.84, 51.96, 54.12, 55.56, 67.80, 68.64, 68.88, 84.12, 93.12, 98.64, 105.12, 105.84, 127.92, 128.04, 173.40." After changing a single element in the dataset from 68.64 to 68.63. The PDF of the bootstrap modifies drastically and having significantly more outcome elements. These two graphs illustrate how small changes will result dramatic change in which also supported by the sum of digits section.

# 5 Estimating Unknown Convolution Distributions with Discrete Distributions

We know that the summation of exponential distributions are equivalent to a Erlang distributions. Thus we used this knowledge to compare the summation of the fully enumerated bootstrap of a random exponential distribution of different number of elements and convolutions. We used an random exponential function with a mean of $\frac{1}{10}$ and compared it the an Erlang with that same mean and the number of convolutions that were used for that particular graph. The value of sample elements were $5, 10, 15$ and $20$. Along with $2, 3, 4$ and $5$ convolutions. Below are the graphs that we found using Maple and APPL, **n** denotes the number of sample elements and **m** denotes the number of convolutions.

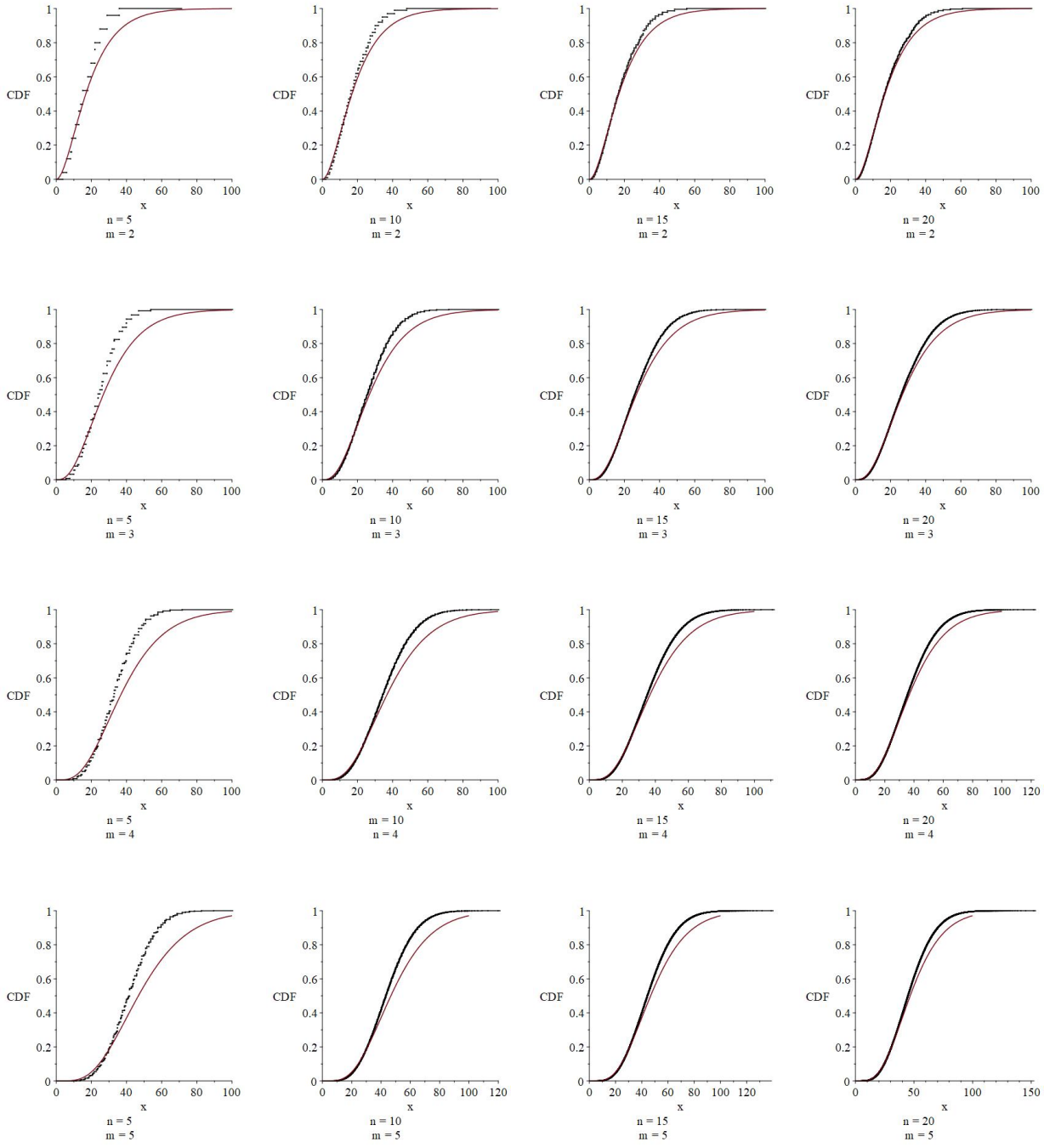


Figure 6: Exponential and Erlang Distribution Comparisons

The graphs display a couple patterns in the data. As you go from left to right and increase the number of elements the graphs become very close to identical and have a much smaller gap between the Erlang PDF's curve and the curve generated by the fully enumerated bootstrap convolution. We also can see that as you

follow the graphs down, keeping the number of elements the same but increasing the number of convolutions, the gap between the two curves grows. This makes sense that as the number of convolutions increase the error increases producing a graph with greater re-sampling error.

# 6 Fully Enumerated Bootstrapping as applied to reliability block diagrams.

Fully Enumerated Bootstrapping technique can be applied to problems related to system reliability. The advantage of this is that an empirical system lifetime distribution of a coherent system can be modeled, using by a reliability block diagram. Another advantage is that, since we will have the failure data for individual components, the system lifetime distribution derived here will be free of re-sampling error. However, a critical topic of discussion will be the presence of bias in the bootstrapping technique: depending on system configuration, fully enumerated bootstrap can underestimate or overestimate the mean system lifetime.

Let's consider a system of two components in series. We are interested in the time after which the system will fail. This system, in principle, can be anything: a power supply system in series with some computation hardware. The figure below illustrates this:

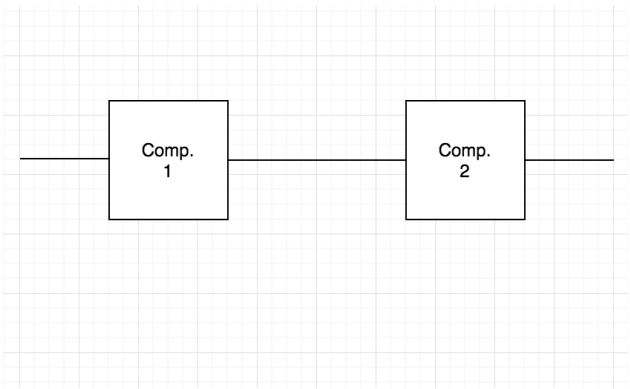


Figure 7: Two components in series

We also know the distribution of the failure time of this system in consideration. So, let's suppose the failure times of the components are probability density functions of the Normal function, with the distributions

given as follows:

$$C1 = N(\mu, \sigma) = N\,(25, 3)\,; C2 = N(\mu, \sigma) = N\,(20, 5)\,. \tag{6.1}$$

The algorithm for fully enumerated bootstrap to estimate the mean lifetime is given as follows:

**(1)** Create a list of $n$ random failure times for each component.

**(2)** Fully Enumerate Bootstrap this list. What this does is that it takes this list of $n$ numbers, and creates a discrete PDF with probability of each number in the list having a probability of $\frac{1}{n}$. This yields two discrete PDF's–call it $B1$ and $B2$–each with $n$ number of components.

**(3)** Take the minimum of $B1$ and $B2$, which is basically the computation of failure rate of the components in series.

**(4)** Repeat the experiment 2000 times, and average all the minimums, to be compared against the true mean of the system, which is 20.

We repeated this experiment for each $n = 10, 15, 20, 25$. The table below shows the results:

| | True Mean | $n = 10$ | $n = 15$ | $n = 20$ |
|---|---|---|---|---|
| Mean | 20 | 21.580 | 20.514 | 18.70 |
| Difference | 0 | 1.580 | 0.514 | 1.30 |

Table 1: Bias in a series system of components

As can be seen, there is some bias in the system. The following Probability Density Functions will also help to illustrate this point.The PDF's do not match, not even coming close to lining up.

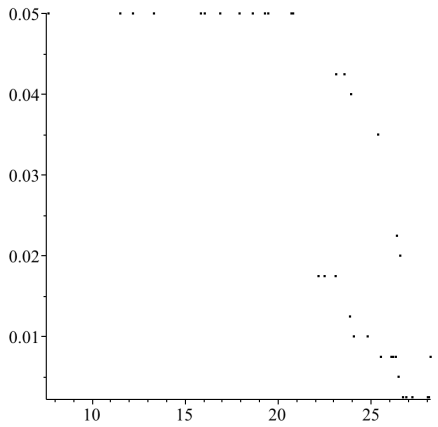


Figure 8: PDF for Bootstrapped, $n = 20$

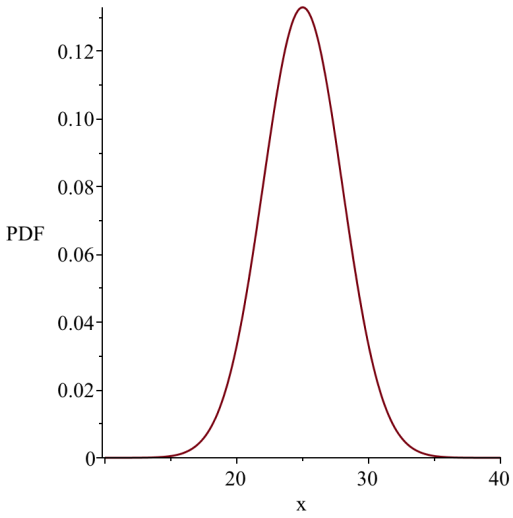


Figure 9: PDF of True Distribution.

In much the similar way, we can construct a system for the parallel and a combination of series-parallel systems. In general, it is found that the bias in the parallel system is the least, followed by the combination system of series-parallel, and series system having the greatest bias. We proceed with the calculation of with the same components as before, but now connected in parallel. Such a system is shown below:

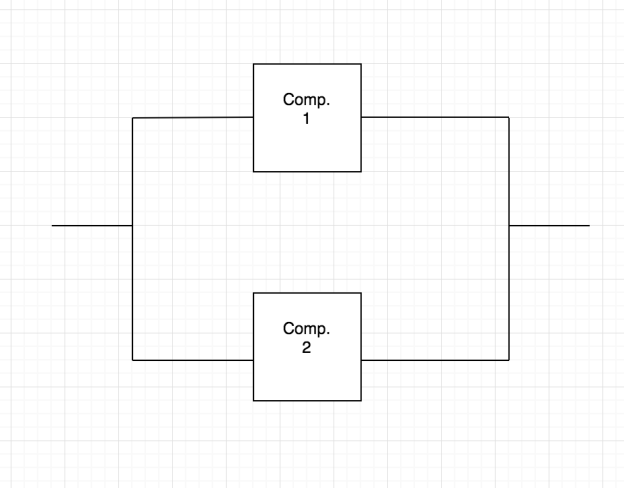


Figure 10: Parallel System.

The advantage of systems connected in parallel is that if one component of the system fails, the other component still works. As a result, the true mean life time is the maximum of the two means of the normally distributed PDF's. Therefore, the true mean is given by 25.

c

The bias in this system is less than that of the series system. The PDF's of the true mean distribution as well as the Bootstrapped system are shown below:

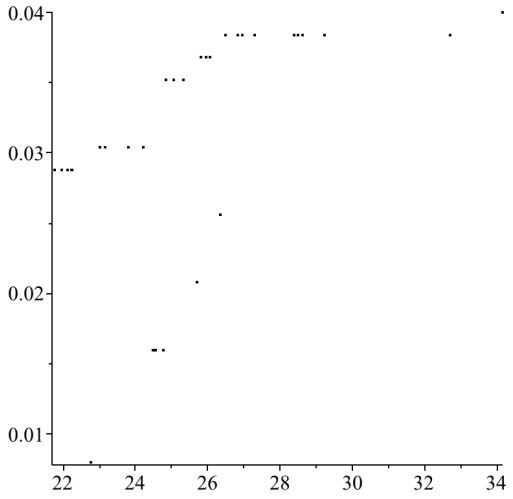


Figure 11: Parallel System infinitely Bootstrapped PDF

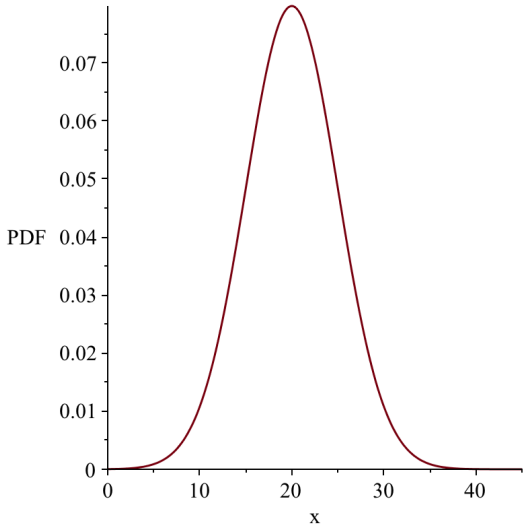


Figure 12: Parallel System true PDF

We see that the PDF's still do a poor job of lining up. It is conjectured that increasing $n$ will make the PDF's more aligned to each other.

**Closing remarks:** There exists inherent bias when a system in fully enumerated bootstrapped. It is, however, possible to reduce the inherent bias in the system by evaluating different configurations of systems: a parallel system will inherently have less bias than a series one. A combination of series and parallel systems can also be used to achieve improvement in the system. We do one final example as a series and parallel combination to see the system bias in "WheatStone Bridge" system, a common electrical engineering system configuration. The system is shown below:

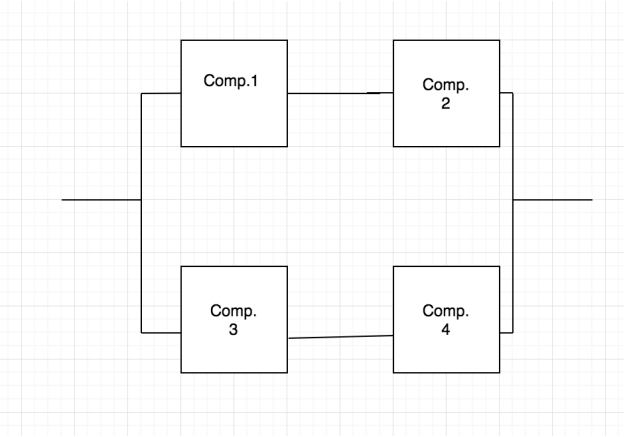


Figure 13: Wheatstone Bridge

The algorithm that will produce this system configuration is as follows:

(1) We take the minimum of Component 1 and Component 2, since they are in series. Let's call this $M1 = Min(Comp1, Comp2)$

(2) Similarly, we take the minimum of Component 3 and Component 4, $M2 = Min(Comp3, Comp4)$.

(3) Finally, we take the maximum, $Max(M1, M2)$, since they both are in parallel to each other.

Let's suppose all the components are PDFs of the normally distributed function.

$$C1 = N(20, 5); C2 = N(25, 3) \tag{6.2}$$

$$C3 = N(23, 6); N = (30, 2) \tag{6.3}$$

The table of results for $n = 10, 15, 20$ is shown below:

| | True Mean | $n = 10$ | $n = 15$ | $n = 20$ |
|---|---|---|---|---|
| Mean | 23 | 21.46 | 21.427 | 21.940 |
| Difference | 0 | 1.540 | 1.573 | 1.06 |

Table 2: Bias in a series-parallel system

Conclusion: A series-parallel combination results in more bias than a purely parallel system, but less bias than that of a purely series system.

# 7 Measuring Re-sampling Error

x1= Z~Exponential(1/10,0), x1=x2=x3

x1+x2+x3= Z~Erlang(1/10,3)

B= 10,000

| Digits | % of error (μ) | % of error (σ) |
|---|---|---|
| 1 | 0.00322382087382274 | 0.00177189733184745 |
| 2 | 0.00321401350953861 | 0.00176953962895474 |
| 3 | 0.00321460194822331 | 0.00176644443691160 |
| 4 | 0.00321457696121598 | 0.00176635097434083 |
| 5 | 0.00321458114781029 | 0.00176634479275024 |
| 6* | 0.00321458889466517 | 0.00176630220335647 |
| 7 | 0.00321458896235703 | 0.00176630444550384 |
| 8 | 0.00321458857489783 | 0.00176630817407100 |
| 9 | 0.00321457624981290 | 0.00176635593797896 |
| 10 | 0.00321457684037024 | 0.00176635292943071 |
| 11 | 0.00321456738686710 | 0.00176642589000867 |
| 12 | 0.00321456738686713 | 0.00176642589000867 |
| 13 | 0.00321456738686713 | 0.00176642589000867 |

Table 3. The difference between normal bootstrap's and fully enumerated bootstrap's estimate of μ and σ for exponential distribution with different digits as shown. The sample size n = 25 and the number of convolution for fully enumerated bootstrap is m = 3. The normal bootstrap has been resample 10,000 times.

From the previous section, we knew that fully enumerated bootstrap gives you an accurate estimate. Hence, the difference between the fully enumerated bootstrap's and re-sample based bootstrap's estimate is induced by the re-sampling error. To compare the value between them, we find a way to measure the re-sampling error. Referring to Table 3, the error rate of mean is keep decreasing when there are more digits. As for the standard deviation, the error rate first decreases and then increases again with the growing significant digits.