

Chapter 3

Sampling Distributions

The “ing” in “sampling distribution” is very important. Without the “ing,” we have a “sample distribution,” which is a different thing. If you go out and measure the IQs of 50 randomly selected students in the Lake Wobegon Independent School District, you have a sample distribution (no “ing”). Chances are that not all of the students will be above average, but you might be curious to see if the students are – well – above average on average. So you would compute the mean of your sample distribution to see if it were above 100. Now, if you were repeat this process (don’t worry about why just yet) – collecting a sample of 50, computing the mean, and jotting it down – 29 more times, then you would have a *sampling* distribution of 30 means. A sample distribution is a collection of measurements. A sampling distribution is collection of summary statistics (often the mean), each of which was computed from a different sample. If it were up to me, they would have completely different names, like “data distribution” and “summary distribution” so students would get less confused. But we are stuck with what we have, so just play close attention to the presence or absence of that “ing.”

Why would we want a sampling distribution? Let’s start over at Lake Wobegon. You measure the IQs of 50 randomly selected students and compute their average IQ, which comes out to be 101.5. At first you might think “Aha, students are really above average here.” But then you might start to wonder: 101.5 is not that different from 100, what if the students are really average, and my number is just a fluke? What if students are really well above average (115, say), and my number is just a fluke in the other direction?

Perhaps the most obvious (if impractical) thing to do is have someone check your work. So, you have a friend go to Lake Wobegon, do 50 measurements and compute a mean. Intuitively, if she also gets 101.5, then this is probably a good estimate of the true mean. But what if she gets 105, or 90.5? You could get into an argument over whose mean is better, or you might decide to use the average of the two numbers but, if you think about it, the best way to figure out the value of the “right” mean is just to collect a bunch of them. So you might enlist 28 (or 48, or 98) more people and have each of them do the experiment and compute a mean. Then, you can plot the sampling distribution of the mean and see what is really going on.

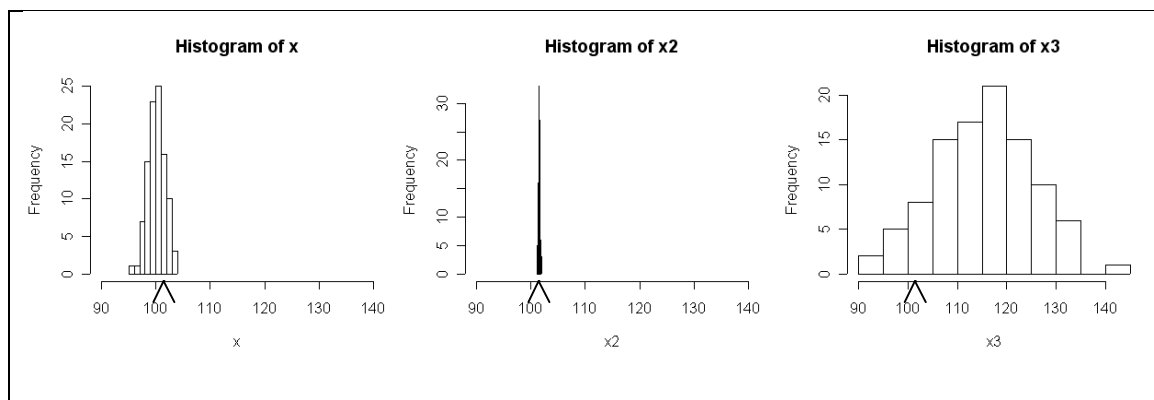


Figure 3.1 – Three possible sampling distributions given a mean from a single sample

of Lake Wobegon children (shown by small black arrow on the x-axis).

Figure 3.1 shows 3 possible sampling distributions of 100 means. On the left, our original estimate of the mean was a little on the high side, and it looks like Lake Wobegon children are average after all. In the middle, our original estimate was right on and, moreover, repeated sampling produced 100 nearly identical means, all above average. We would thus conclude that Lake Wobegon children are almost certainly above average, but not by an amount that is likely to be very important. On the right, we see that our original estimate was a bit low – about 93% of the means were above our original one. Only about 94% of the estimates are above 100, however. So, after all that work and what looks like pretty clear evidence that the average Lake Wobegon IQ is well above 100, we would not be able to conclude, statistically, that this was so.*

As a slightly more complicated example, consider a project in which a student wishes to see if cells divide more rapidly on one medium than another. Forty cell cultures are randomly assigned to the two groups. After a fixed amount of time, the areas of the cultures are measured, resulting in two sets of 20 numbers. The question is “Did medium A result in more growth than medium B?” The data are shown in Figure 3.1. Clearly, the mean of group B is higher, and it is higher by an amount that the student would consider important. The problem, however, is if we had simply divided the 40 cultures into two arbitrary groups, both using exactly the same medium, we still would have gotten some difference in the means, so how can we be sure that the difference we are seeing is a real difference.

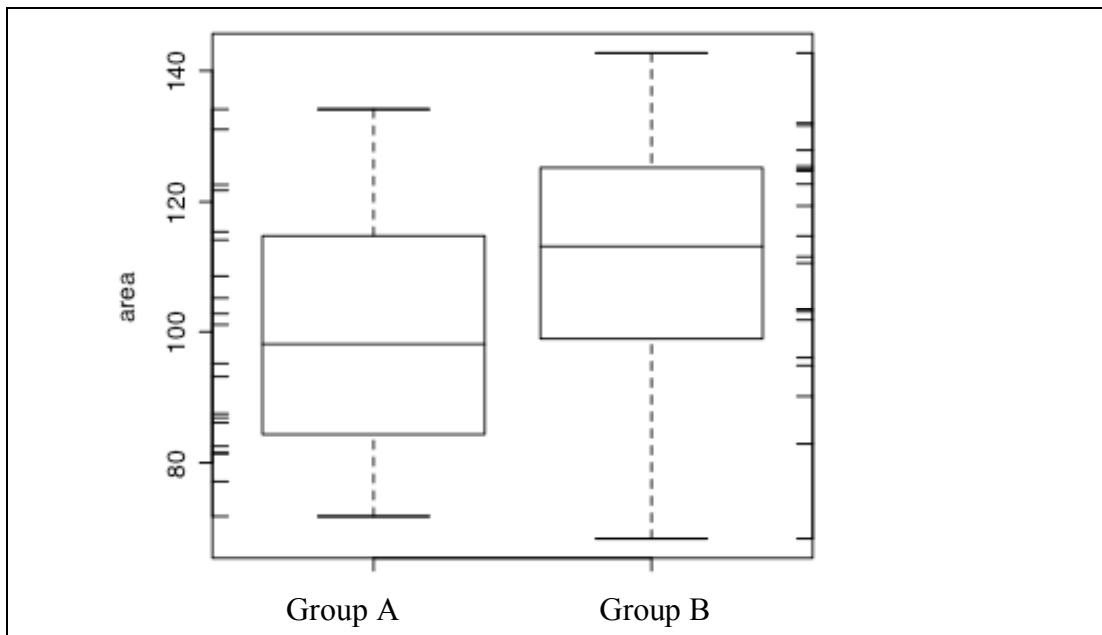


Figure 3.2 – A boxplot of the distributions of the two groups. The center lines are the medians, the box ends show the upper and lower quartiles (25th and 75th percentiles), and the whiskers show the maximum and minimum. The inner ticks

* The rightmost sampling distribution is much too broad to be realistic for these circumstances; the width of the one in the center is more realistic.

on the left and right show the locations of the individual data points.

In principle, this is an easy problem to solve – the student could simply run the experiment again. She might get a smaller difference, making her a little nervous, or a bigger difference, making her curious just how big the difference is. If the student had an inordinate amount of patience, she might decide to just buckle down in repeat the experiment many, many times. After having done so, the student will have a distribution of means for the two groups. In other words, the student will not only have estimates of the locations of the true means, but she will also have an estimate of how stable the means are given a sample size of 20. With this knowledge, she can make a much more informed decision concerning whether her original difference was real or not: if the two distributions of means overlap a lot, then the difference between any two means was likely due to chance. If the distributions do not overlap, however, then the difference is likely real. This can be simply thought of as a simple signal-to-noise problem, and the question is whether the signal – the difference between the means – is large relative to the noise – the precision with which the individual means are known. Obviously, however, this has come at a cost because our poor student has had to run her experiment 30 times (say) rather than once.

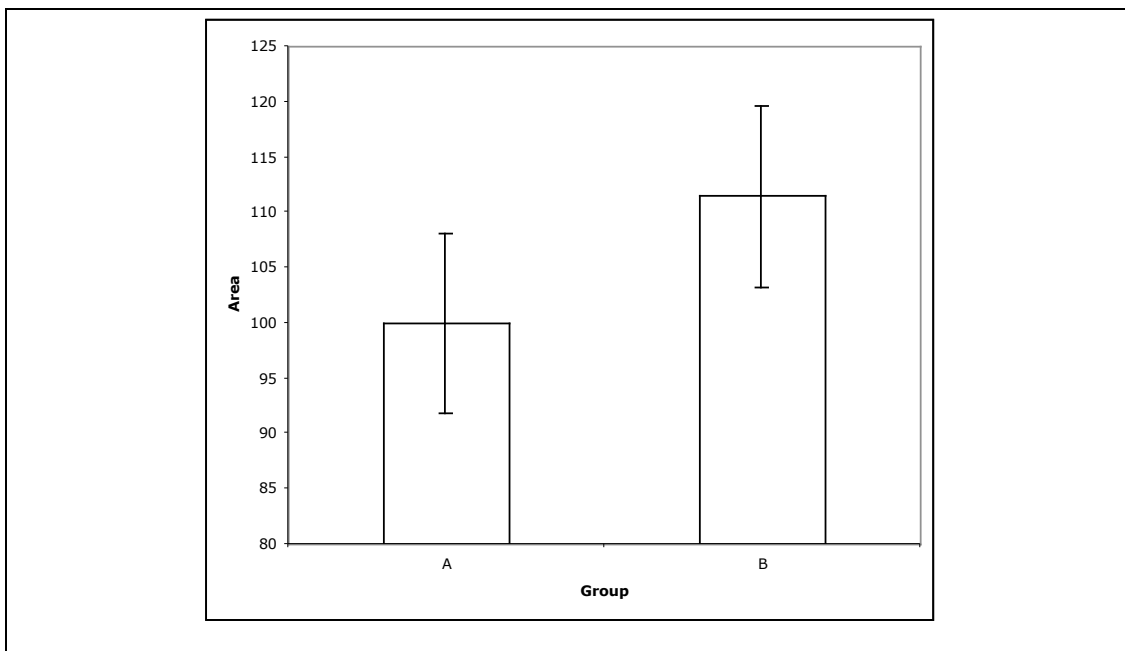


Figure 3.2 – Plot of the means of the two groups. The error bars are “95% confidence intervals; they enclose 95% of the sampling distribution of means. A comparison with Fig. 3.1` reveals that these distributions are quite a bit more narrow than the distributions of the data.

Distributions of means or any other summary statistic computed from data (variance, median, etc.) are known as *sampling distributions*, and they underlie all inferential statistics. As you are no doubt aware, however, it is not common for researchers to repeat and experiment many times over in order to estimate them. In fact, much of the

history of statistics concerns finding ways to calculate or estimate how a summary statistic would have behaved had the experiment been run many times over. In other words, much of statistics has concerned how to estimate some sampling distribution given only a single experiment's worth of data. As you might guess, this is not generally easy to do, which is why most people (almost automatically) concern themselves with averages of their data – the arithmetic mean has some nice properties that has allowed its sampling distribution, or derivatives of it, to be worked out. This, in turn, is why the vast majority of data analysis in the behavioral sciences is done using a relatively small number of statistical tests, the t-test and the ANOVA chief among them.

Here is some good news: the ubiquity of computers and numerical computing environments (such as MATLAB) has made it possible to compute the sampling distribution for almost anything given almost any set of data quickly and easily. This means that not only can we consider aspects of our data other than the mean, but we handle many situations in which the assumptions of the standard statistical tests have been violated. The key is to let the computer repeat our experiment for us while we just sit back and let sampling distribution(s) emerge before us. At this point you may be thinking “Wait, my computer can't run off and collect data, so how is it going to repeat the experiment?” What we actually do is to assume that the sample is sufficiently representative of the population that we can create artificial data sets based upon known properties of the sample, and treat them as results of repeated experiments. A common objection when people hear this is “You can't just make up data sets based on the sample – what if the sample is bad?” The answer is that, if the sample is bad, your analysis will be wrong but – and I can not stress this enough – so will *any* analysis, including the traditional statistical tests. GIGO – garbage in, garbage out – applies universally.

The utility of sampling distributions.

As mentioned above, sampling distributions underlie every inferential statistical test. If you have had a statistics course, then you will remember (at least the names of!) the t-test, ANOVA, chi-square, etc. All of these, and other, tests work in the same basic way – a number that you compute from your data is compared to a sampling distribution to see how likely or unlikely your number is, given some assumptions about the state of the world.

In the above examples, we determined out sampling distributions by brute force, actually repeating our entire experiment many times over, which does not seem like a very good deal. Consider, however, that if I were to go out and measure the IQs of 50 students at a large school not unlike Lake Wobegon, I could quickly tell how likely it is that the students at this school were smarter (or not as smart) as the Lake Wobegon students because, I could compare my mean with the Lake Wobegon sampling distribution. Not such a great deal for you, perhaps, but a great deal for me.

The key to using sampling distributions, then, is to not have to derive them yourself by repeating your experiments over and over. This is why most statistics books have tables of numbers in the back; the majority of these are tables of sampling distributions. Today, however, we have computers that allow us to not only do calculations more precisely than can be done using tables but, more importantly, we can compute sampling distributions for a variety of situations not covered by the tables.

Three paths to a sampling distribution

There are actually three ways to estimate a sampling distribution, although standard textbooks usually cover only one. These are 1) theoretical calculation, 2) Monte Carlo simulation, and 3) Bootstrapping. The important differences among them lies in how much you know or assume about the population under study, and how much you know about the behavior (across many experiments) of a summary statistic computed from the data.

Theoretical calculations and the Central Limit Theorem

Historically, the most common path to the sampling distribution in a particular situation is to stand on the shoulders of giants. That is to say, to do an experiment and compute a summary number, such as the mean, for which the sampling distribution has already been worked out.

To return to our high school IQ example, another way to approach the problem is to consider each of our original 50 IQ measurements as either “above average” (>100) or “not above average” (≤ 100). We have now converted our data into binomial (two outcome) form, and others before us have determined that repeated experiments with this kind of outcome will yield a binomial distribution. Intuitively, if the students at Lake Wobegon are actually about average, then we would expect roughly 25 (half) of them to be “above average.” Note that your intuition has already given you part of the sampling distribution under the “Lake Wobegon is average” theory – it should be centered around 25. Your intuition should also tell you that 27 or 28 above average scores wouldn’t be that unusual, whereas 40 or 50 would be a pretty strong indication that the school was, in fact above average. So without any math, we already have some sense of the location and width of the sampling distribution.

The binomial sampling distribution for $n=50$ and $p=0.5$ (chance of “above average”) is shown in Figure 3.3. This was generated by the MATLAB commands:

```
>> x = 0:50;
>> y=binopdf(x, 50, .5);
>> plot(x,y, 'o')
```

The first line makes an x-axis that is a sequence from our lowest to highest values. The second computes the binomial probabilities of each x-axis value. The last, obviously, produces the plot.

As our intuition (hopefully) indicated, the most likely outcomes cluster around 25, and extreme values are very rare. Say we had gotten, 33; inspection of the figure should convince you that a value this large or larger

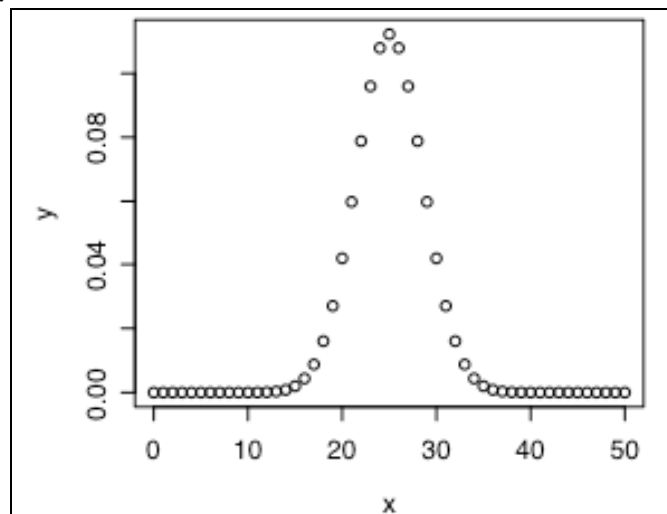


Figure 3.3 – The binomial sampling distribution for a sample size of 50 and $p = 0.5$.

would be extremely rare, assuming that LWH students are actually average. In fact, this would occur under 2% of the time.

In a vast majority of cases, people will concern themselves primarily with questions about the mean, and rely on the Central Limit Theorem to provide the sampling distribution.

The Central Limit Theorem states that:

A random variable that is the sum of many random variables, regardless of their individual distributions, will be distributed as a Gaussian random variable.

A simpler though much less precise version that I tell my students is “If it’s complicated, it’s probably Gaussian.” Let’s say you were to sit by a stop sign on a driver’s education training course, and note the position that the front wheels came to a stop for each of several hundred crossings of the intersection over a period of several days. Your intuition should tell you that the majority of stops will be clustered right around the position of the stop sign, with the occasional student stopping well short, and the occasion student running a few feet out into the intersection. Consider the numerous random variables that go into determining exactly where the car stops: the student’s personality, depth perception, dexterity, experience, etc., the instructor’s attentiveness, strictness, etc., the condition of the brakes and the tires, the wetness of the pavement, etc. What the Central Limit Theorem tells us is that, because the final stopping distance reflects the combined influence of all of these variables, then this final stopping distance will be distributed as a Gaussian, even though we have no idea how these individual variables are distributed! We don’t need to know the shape of the distribution describing how strict the various instructors are, say, because it is just one small component in the final outcome – if it’s complicated, it’s probably Gaussian.

The standard method of estimating the sampling distribution is particularly convenient if the population is Gaussian because, if this is the case, the CLT further says that the relationship between the standard deviation of the population (as estimated by the standard deviation of the sample) and the standard deviation of the sampling distribution (called the standard error), is

$$\sigma_{\bar{x}} = \sigma / \sqrt{n}$$

This says that the standard error, denoted as the standard deviation of sample means, is equal to the standard deviation of the population divided by the square root of your sample size (remember that, by convention, the true population parameters – which are usually unknown - are denoted by Greek letters, whereas estimated parameters are denoted by their Roman – English - counterparts). In practice, σ is estimated by s , the standard deviation of your sample. So now you can see why the standard methods were so popular before we had computers. In the above examples, all we really had to do was 1) assume our populations were roughly normal, 2) collect a sample and compute the standard deviation, and 3) divide by the square root of our sample size and – poof! – we would have had our sampling distribution. No need to enlist many (so-to-be-former) friends to repeat the experiment many times over or. Figure 3.4 summarizes this process.

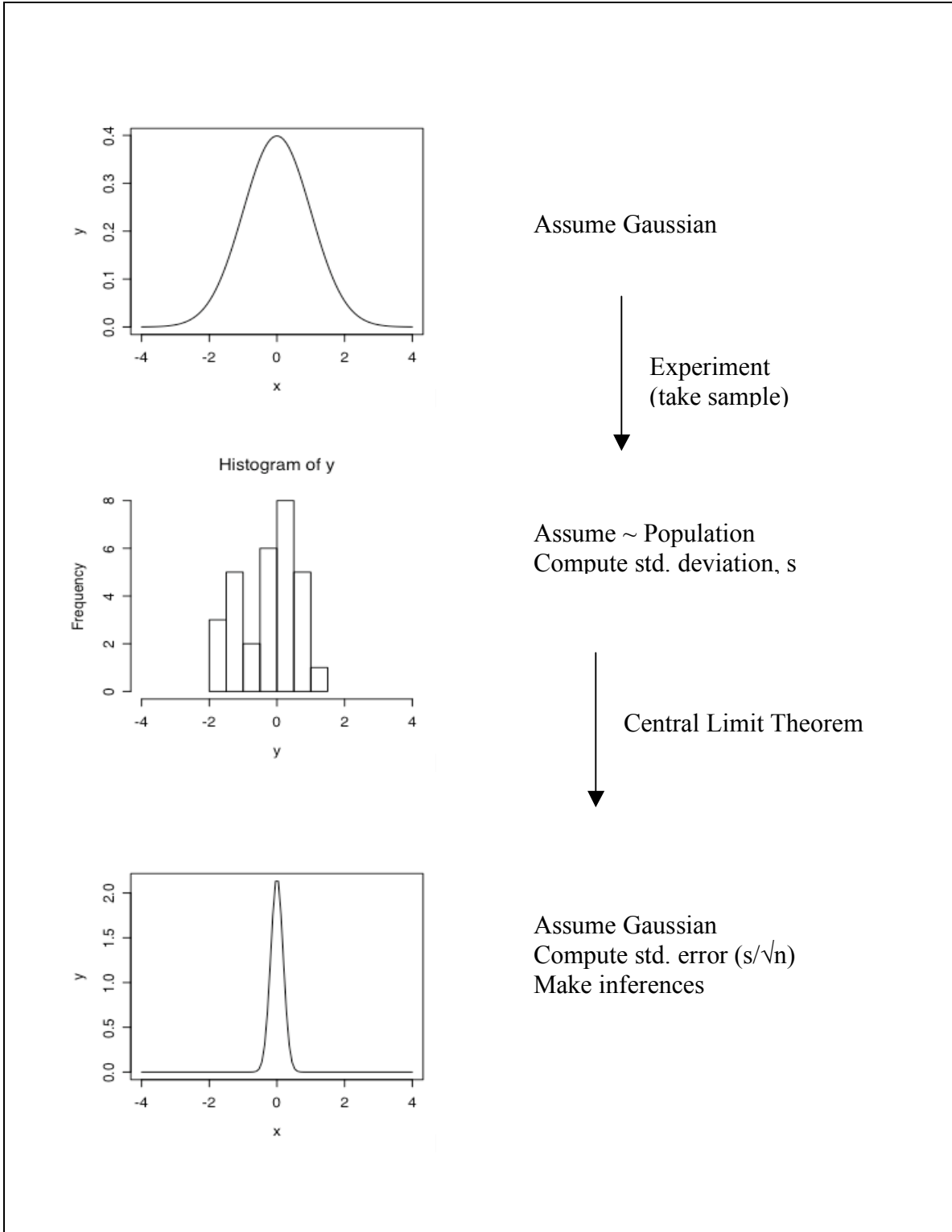


Figure 3.4 – The standard path to the sampling distribution of the mean, assuming a Gaussian population and CLT. The top row is the population, the middle row is the (experimentally determined) sample, and the bottom row is the sampling distribution of the mean.

Monte Carlo Simulation (a.k.a. the Parametric Bootstrap)

The above methods work well when the mean (or some other sum) is the summary statistic of interest, and when the sample size is sufficiently large that the sampling distribution is “Gaussian enough,” and when the population is Gaussian so that the formula for standard error applies. But what if we wish to use something other than the mean, such as the median? Or what if we are unwilling to assume that the population is Gaussian, and are thus uncertain as to the precise width of the sampling distribution?

If we know, or can hazard a good guess at, the shape of the underlying distribution of the data, then we can use this information to our advantage by doing a Monte Carlo simulation of the sampling distribution of our statistic of interest. This procedure, so named because it relies heavily on random numbers as does a casino, is diagrammed in Figure 3.5. From our data, we compute the parameters of the assumed population distribution, and then we use our computer to generate a large number of synthetic data sets whose members are random numbers drawn from this distribution. From each data set, we simply compute our statistic of interest as we did from our original data. Because each simulated data set was composed of random numbers, the value of any statistic computed from them will vary from data set to data set, just as it would if we replicated our experiment. Thus, if our assumption about the shape of the population distribution was valid, then the distribution of the statistic we calculated from the simulated data sets will be the sampling distribution for that statistic.

Once we have this estimated sampling distribution, we can, in principle, proceed just as in traditional statistics. The end result of the Monte Carlo simulation is simply a sampling distribution of a statistic of interest, which is exactly what the t-distribution, F-distribution, and all other distributions tabled in the back of statistics books are. In fact, one way to think of a sampling distribution generated by Monte Carlo simulation is as a home-made statistical table, one custom computed for your assumed population distribution and the sample size of your experiment.

Some people are initially uncomfortable with the first step of the Monte Carlo method, assuming the shape of the population distribution. Every time I have introduced this topic to a class, someone has asked something like “How can you just say that your data are distributed according to some distribution? Isn’t that cheating?” There are a couple of points that need to be made in response. First, when you use the traditional methods, you are assuming a shape of the population distribution; you are asserting that it is Gaussian. The universality of these tests and the ease of performing them with commercial statistical software has, I think, masked this assumption from many students, but it is a very real assumption nonetheless.

Second, we often have good theoretical reason for assuming that data are distributed a certain way. Counts of discreet, rare events for example are Poisson. In days gone by, they were often treated as Gaussian so traditional data analysis methods could be used, but with the ubiquity of computers, there is no reason assume things are Gaussian when they are not.

Finally, if you take a sample of data and it looks extremely skewed like an exponential distribution, then you will get much better results by assuming that the population is exponential, even if it is actually not, than if you assume it is Gaussian. Just because a lot of things are Gaussian does not mean that your data are, especially if they

look very non-Gaussian. If it does not walk like a duck, and does not talk like a duck, it probably is not a duck, even when there are a lot of ducks around. You may be wondering “But what if the population really is Gaussian and I just got a bad sample?” There is an easy if unnerving answer to this: if your sample is bad, your analysis will be bad, no matter what methodology you employ.

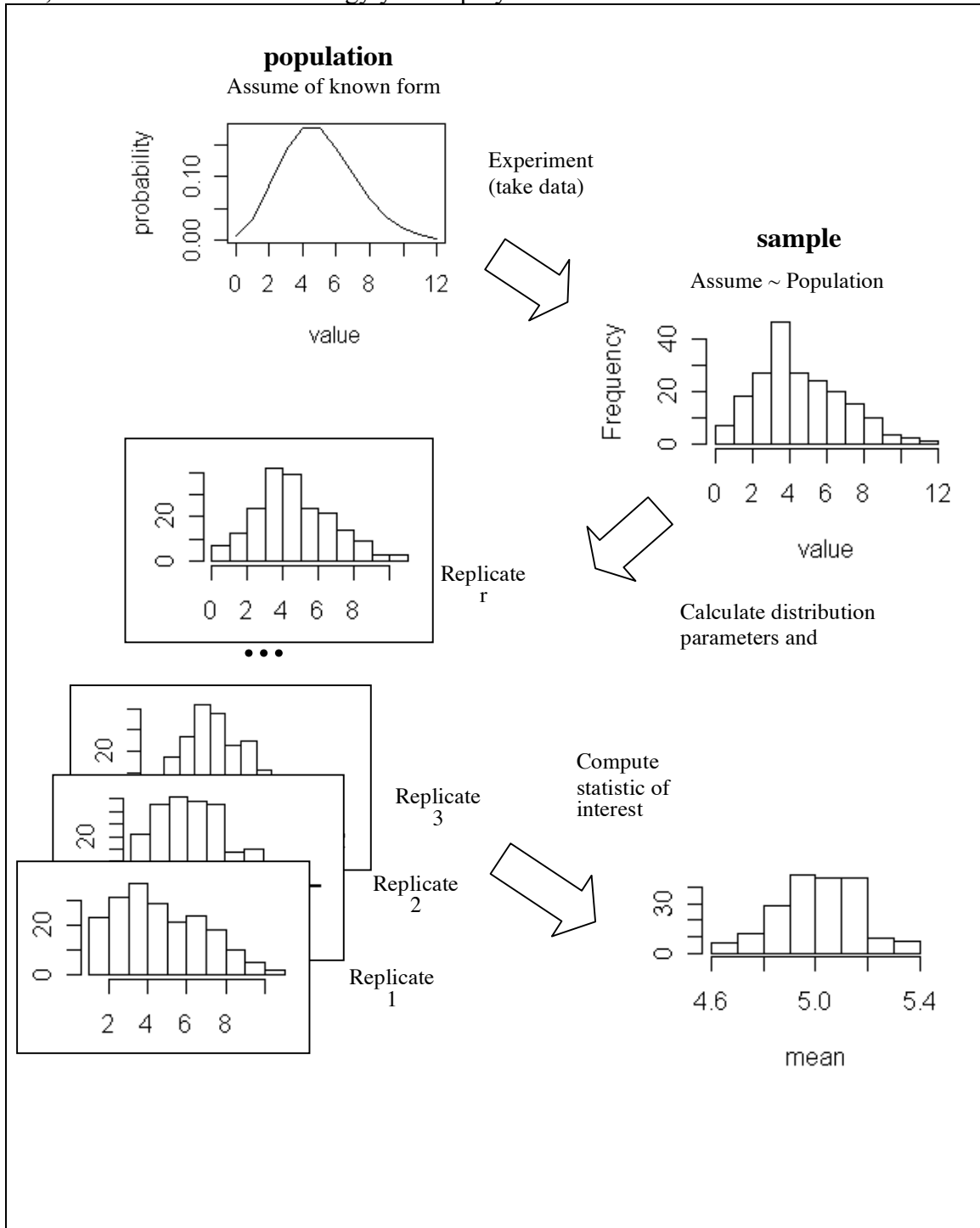


Figure 3.5 - The Monte Carlo path to the sampling distribution.

The Bootstrap (a.k.a. the Non-Parametric Bootstrap)

A consideration of the above two methods begs the question “What if we really don’t know the form of the population distribution?” That is, what if we are not only unwilling to assume that the data come from a Gaussian population, but what if we are unwilling to assume that the population is of *any* particular shape? In this case, we use the Bootstrap, and need to assume only that our sample is representative of the population. Note that this assumption is common to the other two techniques as well. As I just said in the previous section, if the sample is bad, any analysis going to produce bad results; garbage in, garbage out.

The Bootstrap procedure is outlined in Figure 3.6. As with the other techniques, we begin by collecting a sample of data (of course). The next step is the one that always give people trouble, so get ready. We now generate a large number of simulated data sets from our original data set, and each having the same sample size as the original data set. The key is that when we draw from our original data set to create a simulated data set, we sample *with replacement*. Consider the data set

[4, 9, 8, 1, 5]

To create one bootstrap sample by hand, we could write the five numbers on slips of paper and put them in a hat. We would then randomly choose one of the slips, write down the number on it, and put it back, and then repeat the procedure four more times. In all likelihood, we would not get an exact copy of the original data set (in fact, there would be a $(1/5)^5/5!$ chance of that – less than one in 3000). Because the replacement, the number of different meaningful simulated data sets is very large for even modest sized samples. To do this in MATLAB is easy:

```
>> sillydat = [4, 9, 8, 1, 5]
sillydat =

     4     9     8     1     5
>> randsample(sillydat, 5, true)
ans =

     9     8     1     4     8
>> randsample(sillydat, 5, true)
ans =

     1     1     5     1     8
>> randsample(sillydat, 5, true)
ans =

     9     9     1     4     9
```

Note that we got a different data set each time, each very different from the original (the similarity of the first and third is sheer chance – I was even tempted to replace the third one with one that *seemed* more random).

An equivalent way of looking at it is to imagine taking the data set, making an infinite number of copies of it, and sampling from it several times as though it were a population.

In terms of the above example, imagine making a huge number of copies of each slip of paper, enough to fill a football stadium instead of a hat. We could then draw slips of paper five at a time all day long and get a different data set virtually every time.

Hereafter, the technique is the same as the Monte Carlo. For each sample, we compute our statistic of interest – the median, the variance, whatever – and the resulting set of numbers composes our sampling distribution. Once we have the sampling distribution for a statistic, we are in a position to say what values of that statistic would be considered extreme and what values would be considered typical (which is all any statistical test is ultimately doing).

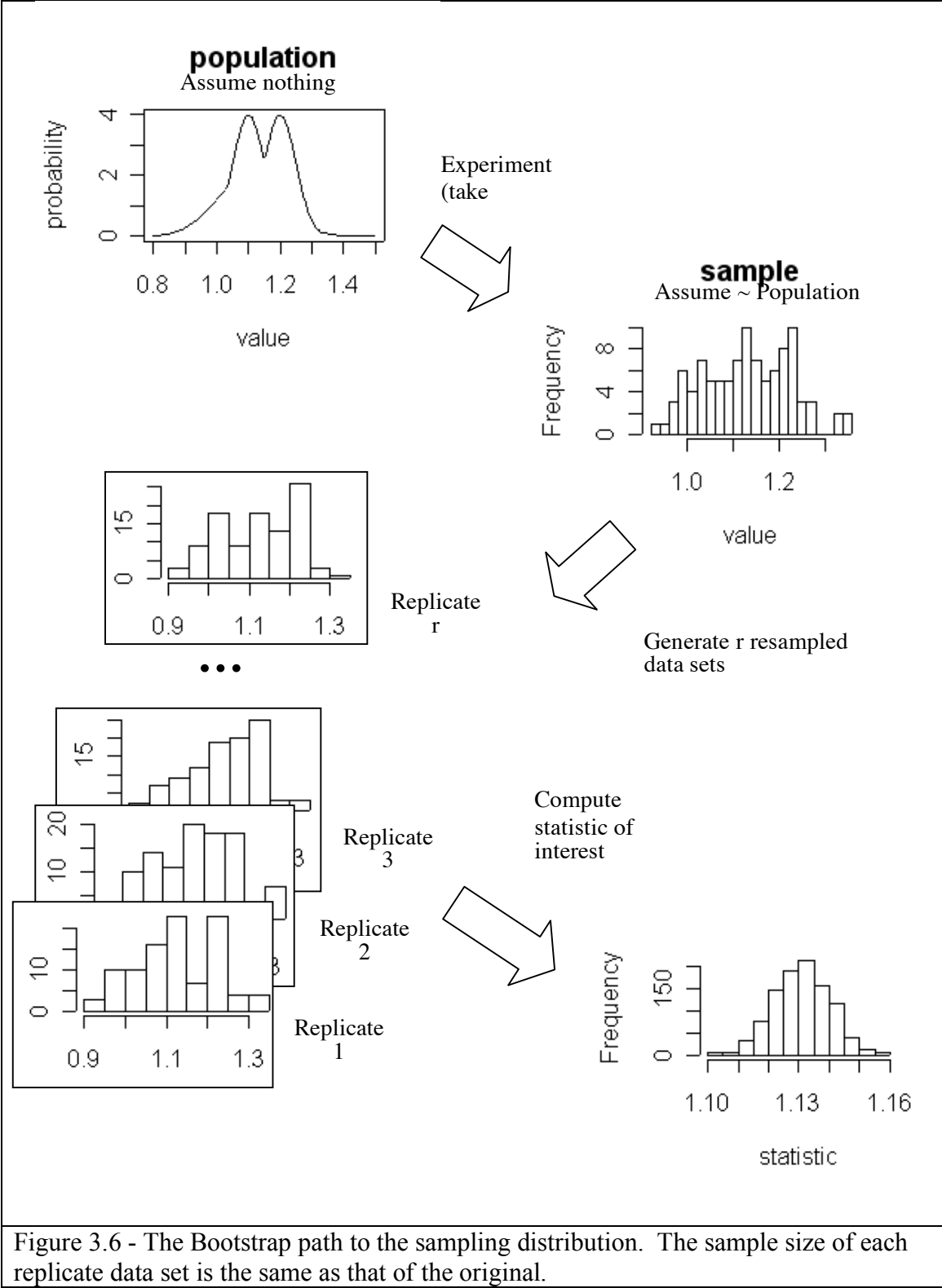


Figure 3.6 - The Bootstrap path to the sampling distribution. The sample size of each replicate data set is the same as that of the original.

As a final example illustrating how we use sampling distributions, consider the five number data set above. Note that our original numbers sum to 27. Suppose I were to propose a little betting game: I keep drawing samples of 5 numbers from the same unknown (to you) place that I got the first five, and every time the sum is more than 18, you give me a dollar, but every time the sum is less than or equal to 18, I have to give you 10 dollars. How would you go about determining if this game was a good bet for you?

The situation is basically this. We are going to do several experiments. The sample size is 5, and the statistic of interest is the sum. What you need to know is the sampling distribution for the sum of 5 numbers. Obviously, I have to win at least 10 times, on average, for every one time you win. If I win fewer than this, you will make money. In other words, if 10% or more of the sampling distribution of our statistic is less than or equal to 18, you are okay, because I will be giving you a \$10 bill more often than you are handing over 10 \$1 bills. But if less than 10% of the tail is cut off by 18, then the situation is reversed and I will make money in the long run.

So what you need to figure out is just where that 10% cutoff is – if it is less than 18, you will make money at our game! But there is a small catch: you don't know how I am getting the numbers. You *could* assume that I am getting the numbers from some Gaussian process (rounded, obviously), and press ahead analyzing the situation using the standard methods. The standard error for these data is 1.46 (rounded). Since the sum is just the mean scaled by 5 (the sample size), the standard error of the sum is just 5 times the standard error of the mean, or 7.18 (rounded). Since the sample size is so small and you don't know the “true” standard deviation, it is imperative that you use the Student's t distribution instead of the standard normal. First, find the 10% quantile:

```
>> tinv(.1, 4)
ans =
-1.5332
```

This tells us the 10% cutoff of a standard t distribution; in this example, it is 1.53 standard errors below mean. Next, find the boundary cutting of the lower 10% of our expected distribution of sums using same formula used to compute confidence intervals:

```
>> 27 - 1.53*7.18 % mean minus critical value times standard error
ans =
16.0146
```

Aha! The point cutting of the lower 10% of this distribution is lower than 18, so you should make money. The situation you just predicted will happen is shown in Figure 3.7. The smooth curve shows the predicted distribution of sums under standard theory – note that this allows for non-integer sums like 28.7, so you become suspicious of this approach already. Nevertheless, it looks qualitatively correct in that it predicts that the most frequently occurring value should be around 27, and that values larger or smaller than this get increasingly rare. The shaded region shows the lower 10% of the area under the distribution, and its upper border is 16. The cutoff I proposed for our game is 18, and more than 10% of the distribution lies to its left, as shown by the hatched region. This, in

turn, means that I should end up paying you \$10 more that 10% of the time, and you make money in the long run.

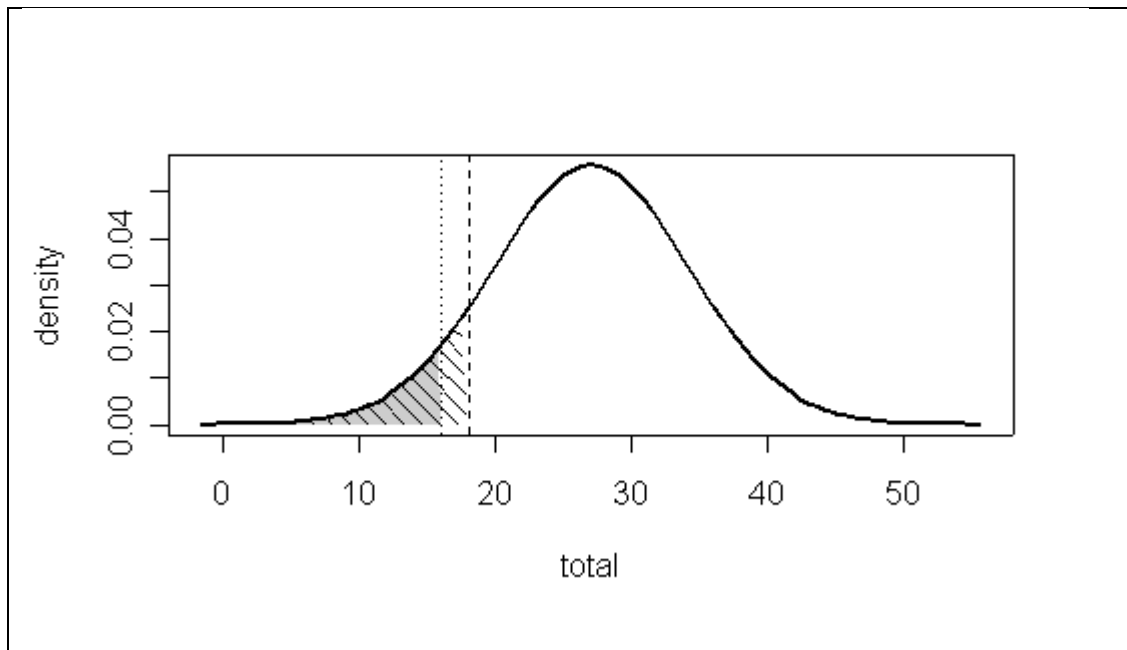


Figure 3.7 – The sum game as evaluated by traditional methods. The gray shaded region is 10%, and its boundary, ~16, is the predicted break even point. By the rules of our game, I will thus be giving you \$10 more than 10% of the time, as shown by the hatched region.

Note, however, that your analysis was predicated on the assumption that I was drawing the numbers from a Gaussian distribution that had a mean of 5.4 ($27/5$ – the mean of the original data), and a standard deviation of 3.21 (the standard deviation observed in the original data). This could be true, but the assumption that you know the distribution from which I am drawing the numbers seems like a pretty strong one in this case.

A safer way to tackle the problem might be to use the bootstrap, in which we assume only that the data at hand are representative of the population. To get our sampling distribution, all we do is sample from our original distribution as illustrated above many times over, saving the sum of sample each case (29, 16, 32, ...and so on). To do this in MATLAB, we would use a loop, just like we did near the end the tutorial chapter, which automates the process of generating new samples, computing the sum, and saving it.:

```
>> sillydat = [4, 9, 8, 1, 5];
>> nrep = 1000
>> samplesum = zeros(nrep, 1);
>> for(i=1:nrep)
onebootsample = randsample(sillydat, 5, true);
samplesum(i) = sum(onebootsample);
end
```

We can now simply count how many times you would win ten dollars and how many times you would lose a dollar if you played this game with me 1000 times:

```
>> nwins = sum(samplesum<=18)
```

```
nwins =  
      77
```

```
>> nloses = sum(samplesum>18)
```

```
nloses =  
      923
```

Although if we plan another 1000 games, these numbers would change a little, it is obvious that the game is not such a good deal for you after all. You would actually lose $923 - 10 \cdot 77 = 153$ dollars if these 1000 games were played.

The resulting 1000 sums are shown by the light gray histogram in Figure 3.8. The smooth black curve is the Gaussian distribution that best approximates the bootstrapped distribution. It is shown only for visual comparison with figure 3.7. Its standard deviation is:

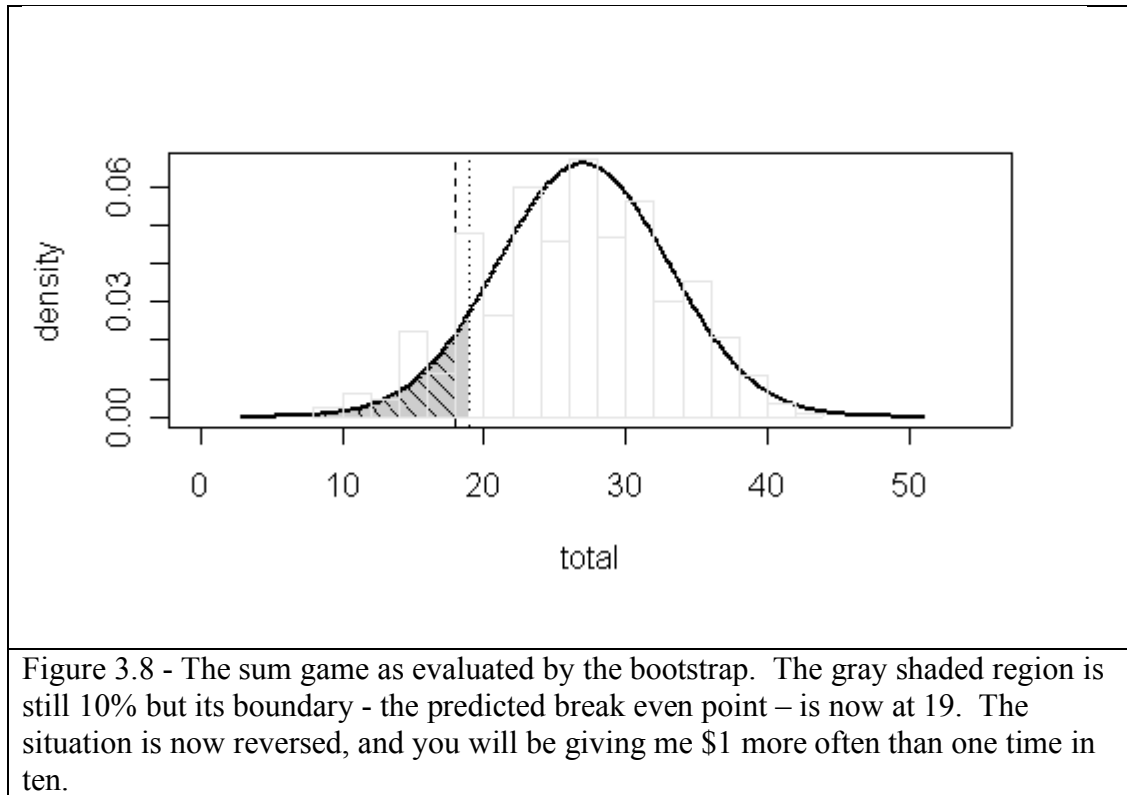
```
>> bootsterr = std(samplesum)
```

```
bootsterr =  
      6.2844
```

To see where the even cutoff point is, we can find the 10% point of our empirical bootstrapped distribution of sums using the `quantile()` function:

```
>> quantile(mysum, .1)
```

```
ans =  
      19
```



Hm. This says that the lower 10% of the distribution will be cut off by a value of 19 (rather than 16), shown by the gray region, meaning that less than 10% of the distribution is below 18, as shown by the hatched region. This analysis confirms that I would win money in the long run because you would be giving me \$1 more than one time in ten. Exactly the opposite of what we concluded using the traditional analysis!

A word of caution is in order here. This was an illustrative example only. Both analyses were based on a sample of 5 numbers, which is very small. Any analysis based on such a small sample is likely to be suspect. The fact that the analyses gave different answers, though, is quite interesting, and should give us good reason to carefully consider the method of analysis we use on a particular problem based on what we actually know about the population, as opposed to blindly proceeding with traditional methods regardless.

Summary

In this chapter, we learned about the most important concept in statistics: the sampling distribution. Like evolution by natural selection in biology, the sampling distribution is the concept that underlies and unifies all inferential statistics. Sampling distribution is what allows us to take a single value estimated from an experiment (like a median), and make statements about the population at large, or to evaluate theories that make predictions about what the value should be. Sampling distribution allows us to compute the probability that we might have obtained some value other than the one we

actually got, given the variability of the data at hand. Put another way, we can make an educated guess about what would happen if we actually repeated our experiment over and over and over again.

We also learned that, given a sample of experimental data, there are three paths to the sampling distribution: the traditional computational path, the Monte Carlo path, and the Bootstrapping path. As summarized in Figure 3.9, the selection of which path to use is based upon how much you know (or are willing to assume) about the actual distribution of the population of interest, and what parameter(s) you wish estimate.

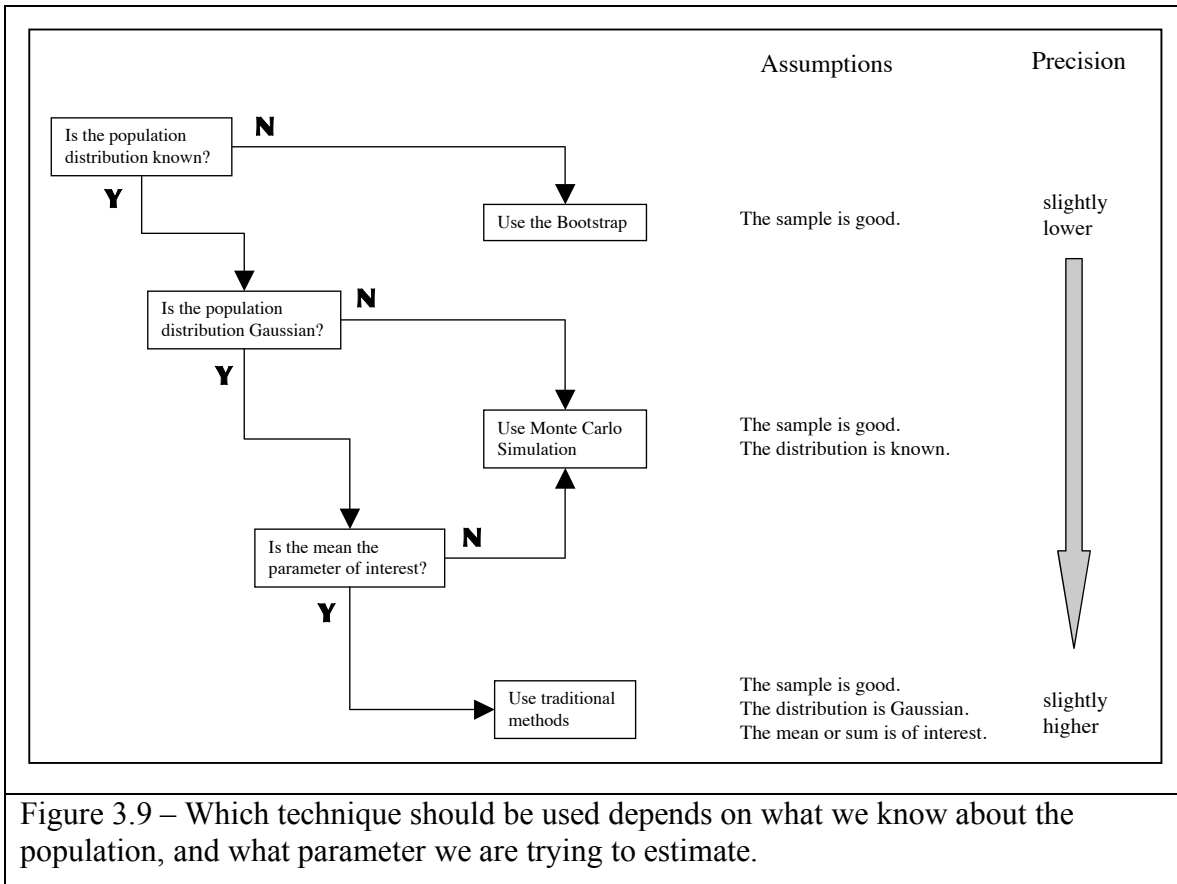


Figure 3.9 – Which technique should be used depends on what we know about the population, and what parameter we are trying to estimate.

If you do not know anything about the distribution, then the only option is to assume that your data are a good sample, and proceed with bootstrap methods (either that, or take more samples). At the other extreme, if you know that your sample comes from a Gaussian distribution and the statistic of interest is the mean, then you can use traditional methods. Alternatively, if you know the form of the population distribution, even if it is not Gaussian, or you wish to compute something other than the mean, or both, then you can proceed with Monte Carlo methods.

You may wonder, since the bootstrap requires only the minimal assumption that your sample is good, why use it all the time? A general answer will suffice for now: the more information you bring you bear on a problem, the better your answer is going to be. For example, when you use (validly) the traditional methods, then the sampling distribution is known exactly, and your answers will be precise. When you use the bootstrap, you are

bringing no information to bear on the problem except the data at hand, and a single sample of data can obviously tell us only so much about the population parameters. Of course, this is by no means a reason for using traditional methods when the assumptions are not met – applying bad information to problem will yield bad answers.

Excercises:

Problem Set 1

Many scientists have a rule of thumb concerning the minimum number of observations needed for each data point they wish to report. This is equivalent to asking how many observations yield a good enough picture of the underlying distribution.

- Determine the sample size of Gaussian random numbers that gives you a satisfactory Gaussian distribution (call this number N_s). Plot a typical sample (at your satisfactory sample size) and overlay an actual normal curve.

Programming notes:

The function `normpdf()` will give you standard normal curve, `line()` will let you overlay this curve on your graph. Alternatively, you can play with the distribution fitting tool, “`dffitool`”.

To get help on any function, type the function name “help” in front at the command line, e.g.

```
>> help normpdf
```

Do a quantile-quantile plot (a.k.a. "QQ plot" or “probability plot”) of your typical sample against a normal distribution - if your sample is in a variable "x", just type "`qqplot(x)`". Explain what a QQ plot is assuming the reader has never heard of one.

Find a distribution that is very un-Gaussian. (For a list of distributions that MATLAB can generate, see the distribution reference in MATLAB’s help)

For sample sizes of 1, 2, 20, and 200, plot a histogram of N_s means of each of the above two distributions (Gaussian and un-Gaussian). Describe what you find, and show QQ plots to illustrate the normality (or lack thereof) of these distributions of means. Programming note: you could accumulate each distribution of means "by hand" - drawing a sample, computing the mean and jotting it down - but you should try using a `for()` loop.

Problem Set 2

In this problem set, we are going analyze a situation in which the conventional assumptions are met in order to verify that Monte Carlo and Bootstrap methods actually work.

Download the file “`pop.mat`” and put it in whatever working directory you use for MATLAB. Load the data set into MATLAB (use “open” from the file menu). Your workspace should now contain a variable named “`mypop`”.

Explore the actual population (the only chance you will every get to do so!)

Do a histogram and overlay a standard normal curve (as we did in the previous problem) to convince yourself that the data are Gaussian.

Compute the mean and standard deviation. We are going to be considering this to be the population, so your computed values are the True (with a capital “T”) values – they should be denoted by Greek letters μ and σ and everything - but they should be pretty close to 0 and 1.

Take a sample and explore it.

Take a sample of data of size 30 (say) from this population (`mysamp = randsample(mypop, 30)`). Make a histogram from you sample and compute the mean and standard deviation. How does your sample compare with the population? (You may wish to make some graphical comparisons).

Evaluate the mean of this sample using the traditional method.

Sketch or plot the expected sampling distribution of your mean under CLT, and indicate the position of the true mean. Would you conclude that the mean of your sample is different from the true mean? Why or why not? (You should try and estimate the relevant area under the sampling distribution.)

What would you conclude about the mean of your sample using a one-sample t-test? (In MATLAB, use `ttest(mysamp, pmu)`, where `mysamp` is your sample and `pmu` is your population mean.

Evaluate the mean of this sample using the Monte Carlo method.

In terms of programming, you already know how to do this. Use `randn()` in a loop. Plot the sampling distribution. What is its width (standard error)? Is this what you expected? Where is the true mean relative to this sampling distribution?

Evaluate the mean of this sample using the Bootstrap method.

In terms of programming, this is almost identical to doing Monte Carlo. The only difference is that instead of generating replicate samples using a random number generator, you will draw them from your own original sample using `randsample(x, size, true)`, where `x` is your original sample and `size` is your original sample size.

As above, plot the sampling distribution, etc.