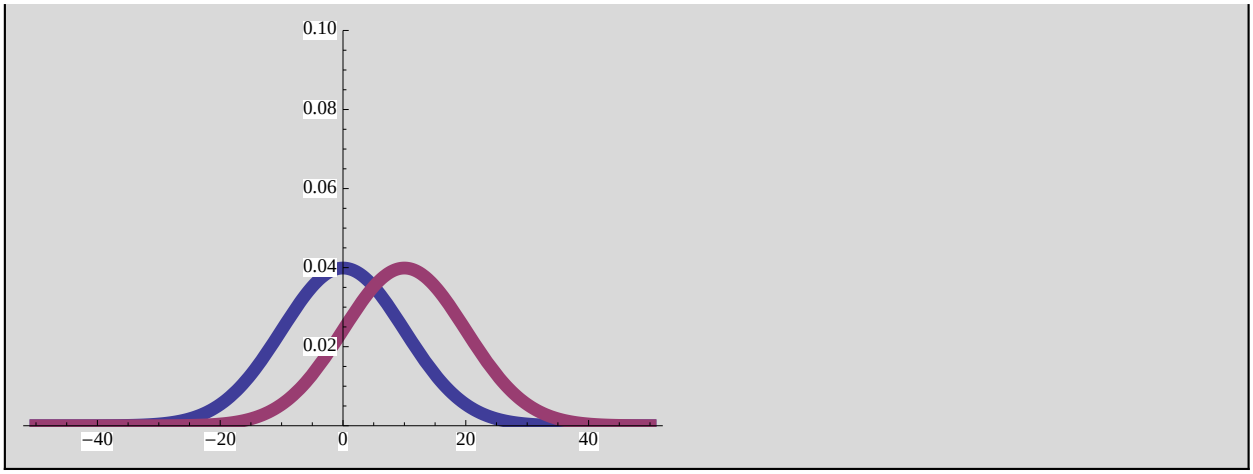
Imagine we do an experiment with two groups (an experimental and a control). In this experiment, we have 20 people in each condition.

```
Needs["Histograms`"];
SetDirectoryToNotebookLocation[];
```

```
ndist = NormalDistribution[0, 10];
ndist2 = NormalDistribution[10, 10];
```

```
f = PDF[ndist, x];
g = PDF[ndist2, x];
myplot2 =
 Plot[{f, g}, {x, -50, 50}, {PlotRange → {0, 0.1}, PlotStyle → {{Thickness[0.02`]}}}]
```




```
groupa = Table[Random[ndist], {20}];
groupb = Table[Random[ndist2], {20}];
```

Let's look at our data:

```
mydata = Transpose[{groupa, groupb}];
mydata // TableForm
```

```
-12.9379   -1.69164
-2.67007   12.7272
-4.18235   11.1443
10.4073    10.2906
-1.6696    7.68877
5.36348    16.7193
-7.0213    12.9419
-16.5087   12.0499
-11.673    22.5242
6.93744    8.93215
-0.884679  22.714
-9.59991   2.94287
5.67319    7.00858
-0.903916  14.5102
-4.08608   13.4046
2.56344    -3.17537
-0.078747  16.8591
-13.3851   0.529633
2.74922    12.6752
2.68488    14.4848
```

Are these groups different? How can we tell?

One way is to begin to describe the data. How about if we consider the maximum value and minimum value in each group?

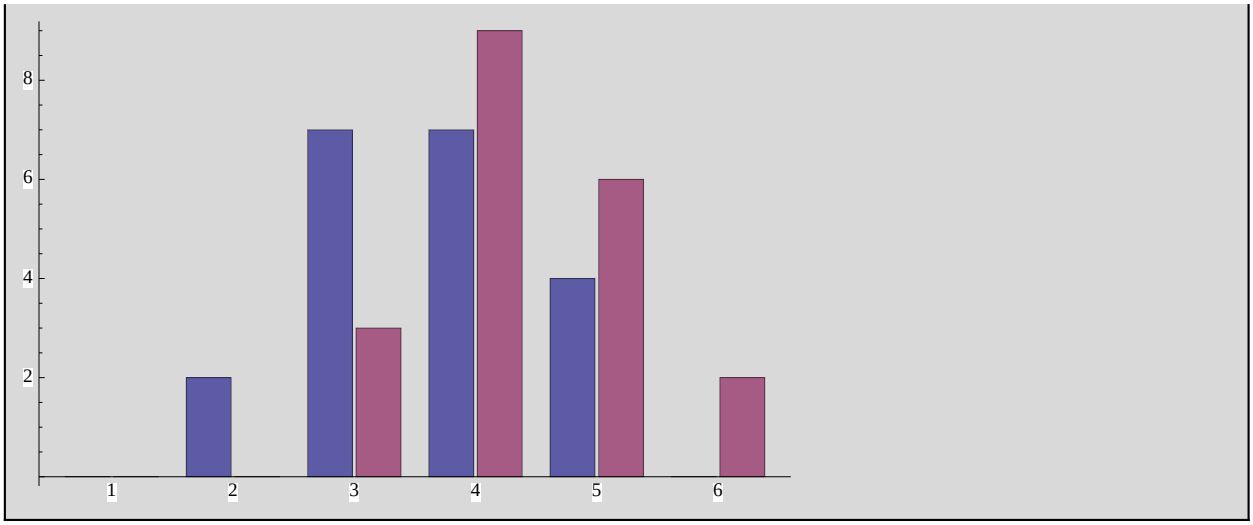
```
mydata // TableForm
```

```
-7.99444  1.91138
9.99132   7.44785
-13.5882  -2.90085
-4.41587  7.29401
6.34063   17.9343
5.78802   9.11708
17.6973   5.3681
-4.75167  20.8968
11.325    7.73589
8.6631    7.54058
16.6555   20.2841
16.4221   17.3804
-2.71555  7.97203
9.26033   10.3984
1.6923    18.3188
-7.06275  -0.306467
-12.4256  10.2488
-0.554747 12.3763
-0.304643 -1.93493
3.46768   4.36072
```

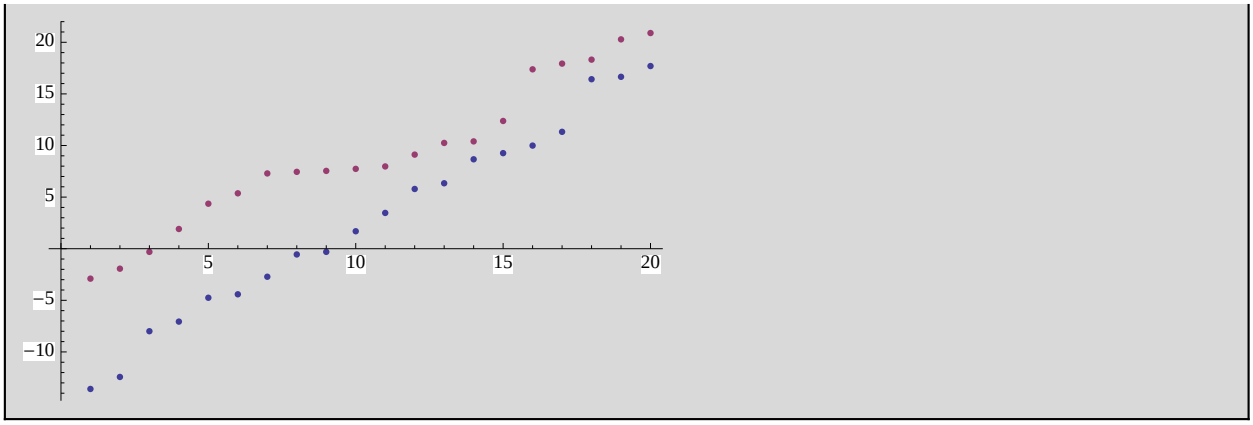
```
Transpose[{Sort[groupa], Sort[groupb]}] // TableForm
```

```
-13.5882  -2.90085
-12.4256  -1.93493
-7.99444  -0.306467
-7.06275  1.91138
-4.75167  4.36072
-4.41587  5.3681
-2.71555  7.29401
-0.554747 7.44785
-0.304643 7.54058
1.6923    7.73589
3.46768   7.97203
5.78802   9.11708
6.34063   10.2488
8.6631    10.3984
9.26033   12.3763
9.99132   17.3804
11.325    17.9343
16.4221   18.3188
16.6555   20.2841
17.6973   20.8968
```

```
BarChart[BinCounts[groupa, {-30, 30, 10}], BinCounts[groupb, {-30, 30, 10}]]
```




```
ListPlot[{Sort[groupa], Sort[groupb]}]
```




```
{Max[groupa], Min[groupa]}
{Max[groupb], Min[groupb]}
```

```
{17.6973, -13.5882}
```

```
{20.8968, -2.90085}
```

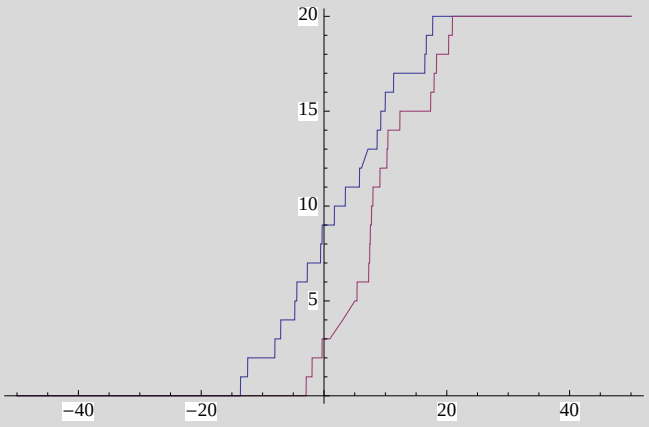
How about the RANGE of the data (max-min)?

```
Max[groupa] - Min[groupa]
Max[groupb] - Min[groupb]
```

```
31.2855
```

```
23.7977
```

```
Plot[
 {Length[Select[Sort[groupa], # < x &]],
  Length[Select[Sort[groupb], # < x &]]}, {x, -50, 50}]
```



Measures of CENTRAL TENDENCY

We begin our discussion with the desire to describe a collection of numbers. It just so happens that there is two main ways in which to describe a collection of numbers. One is called the measure of central tendency and another is the measure of dispersion or scatter. We will first look at a number of different measure of each of these.

Mean

In more concrete terms, the mean is quite simply defined as the the sum of all the scores in a sample divided by the number of scores in the sample.

$$\overline{X} = \frac{\sum_i x_i}{n}$$

There are a number of key points about the mean, First is that the mean is the score around which the deviation scores sum to zero. We can take a look at this informally by just making up a random sample, finding the mean of this sample, and finding the sum of the deviation scores around that mean.

Here we generate 10 random numbers between 0 and 100.

```
samples = RandomReal[{0, 100}, 10]
```

```
{72.1008, 69.6197, 18.1251, 86.605, 31.2848, 70.0553, 70.0837, 90.7437, 88.6898, 98.5136}
```

We can find the mean quite easily using the equation (1). We can check this with *Mathematica's* built in Mean[ ] function. First, we can find the sum of all the samples.

```
sumsamples = Apply[Plus, samples]
meanbyhandsample = sumsamples / Length[samples]
```

```
695.821
```

```
69.5821
```

Now, we can check this with *Mathematica*.

```
meansamples = Mean[samples]
```

```
69.5821
```

```
meansamples == meanbyhandsample
```

```
True
```

Now that we know the mean, lets find the deviations from the mean for each sample. What we to know is how far each indivdual score is from the mean. Fortunately, this is very simple to express in *Mathematica*.

```
deviations = samples - meansamples
```

```
{2.51867, 0.0375583, -51.457, 17.0229,
 -38.2974, 0.473181, 0.501508, 21.1615, 19.1077, 28.9314}
```

Now lets add up all the deviations to see if the do indeed sum to zero.

```
Apply[Plus, deviations]
```

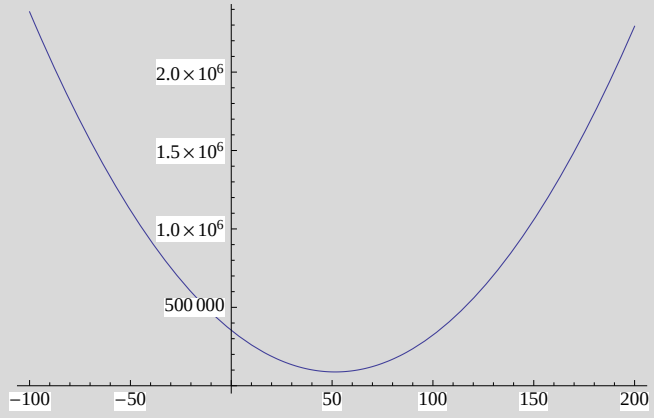
$$4.26326 \times 10^{-14}$$

Well, it is not quite zero, but that is because *Mathematica* has quite a bit of precision, but anything to the power of -14 is close enough to zero for us.

A second important property of the mean is that it is the score for which the squared deviations of the sample is a minimum. Imagine we were to find the deviations for each sample from the mean, and square it, then add these numbers up. What this property of the mean "means" is that the mean will minimize the squared deviations value.

We can take a look at this also. Lets take another random sample of 100 scores this time and find the sum of squared deviations for a range of values. We will see that if we choose the mean this value will be at a minimum.

```
Clear[samples]
Clear[f]
Clear[func]
samples = RandomReal[{0, 100}, 100];
f[x_] := Plus @@ (samples - x)^2;
func = Plot[f[x], {x, -100, 200}]
```



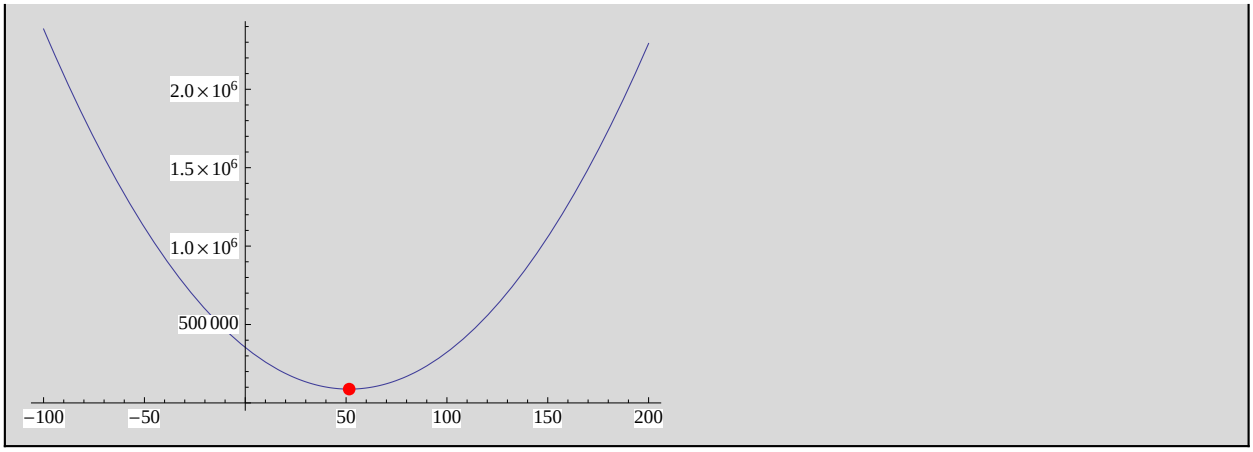
So we can see that the sum of squared deviation looks like a regular parabola which a minimum value somewhere around 50. Well, we can figure out exactly what the minimum of this function is by finding the mean of our sample.

```
Clear[samplemean]
samplemean = Mean[samples]
```

```
51.5358
```

If we plot this point on the graph we can see that it is in fact the minimum.

```
Clear[point]
point = ListPlot[{{samplemean, f[samplemean]}},
    PlotStyle → {PointSize[0.02`], Red}, DisplayFunction → Identity];
Show[{func, point}, DisplayFunction → $DisplayFunction]
```



The big black dot is the mean superimposed on the function of the sum of squared deviaions.

A final note about the mean is that it is sensitive to outliers in the data. We should all be aware of how one exceptional student can blow the curve for the whole class. It is important in statistical analysis to take onliers out under only extremely justified reasons. Lets take a quick look at an example of this. We are going to repeat this experiment with the median in the mode in the next section so we can compare the relavtive sensitivity of the mean, median, and mode to outliers.

```
Clear[samples];
samples = RandomReal[{0, 100}, 100];
Mean[samples]
```

```
51.1359
```

First we find the mean of 100 random numbers. Then we can add one really crazy outlier, 1000 and see how it affects the mean.

```
AppendTo[samples, 1000];
Mean[samples]
```

```
69.7411
```

Wow, if shot up quite a bit! In the next sections we can see how sensitive the median and mode are to this type of phenomena in our data.

Median

The median is the "middle" score. Half the scores in your sample will be above the median and half will be below it.

Lets take a look a 10 random number and find the median score. Five of the numbers should be above the median and 5 should be below it.

```
samples = RandomReal[{0, 100}, 10];
mymedian = Median[samples]
sorted = Sort[samples]
```

As we can see from the sorted version of the list this is true. The only thing to remember about the median is that if there is a even number of scores in your sample, then the median is defined as the average of the two middle scores. In this case, the median is the score half way between the 5th and 6th sorted score. We can check to verify.

```
median = (sorted[[5]] + sorted[[6]]) / 2
median == mymedian
```

The median minimizes the sum of the absolute deviations around itself. In the same way as we showed that the mean minimizes the sum of squared deviations of the sample, we can show that the median does the same for the absolute deviations. (the absolution deviations are the absolute value of the deviation around a score.)

```
Clear[samples]
Clear[f]
Clear[func]
samples = RandomReal[{0, 100}, 100];
f[x_] := Plus @@ Abs[samples - x];
func = Plot[f[x], {x, -100, 200}]
```

```
samplemedian = Median[samples]
```

```
Clear[point]
point = ListPlot[{{samplemedian, f[samplemedian]}},
   PlotStyle → {PointSize[0.02`], Red}, DisplayFunction → Identity];
Show[{func, point}, DisplayFunction → $DisplayFunction]
```

As we can see, once again, the median is the minimum value of the function devided by the sum of squared deviations.

So how sensitive is the median to outliers. Lets perform our ourlier test and see.

```
Clear[samples];
samples = RandomReal[{0, 100}, 100];
Median[samples]
```

```
45.6962
```

```
AppendTo[samples, 1000];
Median[samples]
```

```
46.8705
```

So the median went up by just about 1 whereas the mean went up by 20 to a outlier at 1000!! The median is thus less affected by extreme scores than the mean.

Mode

The mode is simple the most frequent score. If you wanted to guess an individual score and wanted to be exactly right the most, guess the mode. Otherwise the mode isn't that interesting.

Measures of DISPERSION OR SCATTER

Generally, we consider the measure of central tendency to be the overall magnitude of socres in the group of scores. This might be something like the DC level in a sinusoid. Individual scores in our sample will be distributed around this measure of central tendency. Thus, we can use the average distance of scores away from the mean as our measure of dispersion. If we use the mean as our measure of central tendency then this average will obviously be zero. Instead we can square each deviation score and then take the average. This value is called the variance or mean squared deviation score.

Variance

$$\sigma^2 = \frac{\sum_i (x_i - \overline{x})^2}{n - 1}$$

$$s^2 = \frac{\sum_i (x_i - \overline{x})^2}{n - 1}$$

```
randomlist = RandomReal[{0, 10}, 10]
```

```
{7.24961, 6.49333, 9.07067, 0.0578498, 7.62916, 9.07341, 6.82078, 1.47683, 9.2045, 8.17422}
```

```
Apply[Plus, (randomlist - Mean[randomlist])^2]
-----------------------------------------------
            Length[randomlist] - 1
Variance[randomlist]
```

```
10.2239
```

```
10.2239
```

Standard Deviation

The variance is great but the units of it are the squared values of our observations in the sample. For example, if we had a collection of lengths (in meters, m), then the variance would be expressed as m^2. Instead, we can take the square root of the variance to get what is called the standard deviation.

$$\sigma = \sqrt{\sigma^2} = \sqrt{\frac{\sum_i (x_i - \overline{x})^2}{n-1}}$$

$$s = \sqrt{s^2} = \sqrt{\frac{\sum_i (x_i - \overline{x})^2}{n-1}}$$

```
Sqrt[Apply[Plus, (randomlist - Mean[randomlist])^2]/(Length[randomlist] - 1)]
StandardDeviation[randomlist]
```

```
3.19749
```

```
3.19749
```

Another measure is the average deviation. To calculate this value, we take the absolute value of each deviation score. This is the true average "distance" of each point in our sample from the mean.

```
Apply[Plus, Abs[randomlist - Mean[randomlist]]]/Length[randomlist]
```

```
2.30942
```

Mathematica provides a nice function called the DispersionReport[] which calculates a number of dispersion measures at once.

```
DispersionReport[randomlist]
```

```
{Variance → 10.2239, StandardDeviation → 3.19749, SampleRange → 9.14665,
 MeanDeviation → 2.30942, MedianDeviation → 1.28867, QuartileDeviation → 1.28867}
```

A final measure of dispersion is the standard error of the mean which is calculated by dividing the standard deviation by the square root of the number of values in the sample.

```
StandardDeviation[randomlist] / Sqrt[Length[randomlist]]
StandardErrorOfSampleMean[randomlist]
```

```
1.01113
```

```
1.01113
```

Normal distribution and standard errors

Almost all of the mathematics behind inferential statistics depends on properties of the normal distribution. Why is the normal distribution so important? Where did people come up with this stuff? One reason is due to the central limit theorem. The central limit theorm holds that the sum of a large number of random variables will end up approximately normally distributed. Since the mean is just a sum, it means that inferences made with means will also be normally distributed irreguardless of the underlying process that generated the data. Thus, no matter what our "actual" data is if we use the mean as our summary, and we are interested in testing differences between the mean, we will end up using normal distributions. To get a sense of this consider the following example:

Central Limit Theorem

To get a more hands-on understanding of the Central Limit Theorem, lets look at how successive samples from the uniform distribution can be used to approximate the normal distribution.

First, lets create a function that will generate a list of random numbers sampled from the uniform distribution between a specified range. The function sampleUniform[ ] accomplishes this for us.

```
sampleUniform[max_, min_, n_] := data = RandomReal[{max, min}, n];
ndist = NormalDistribution[0, 10];
sampleNorm[n_] := Table[Random[ndist], {n}];
```

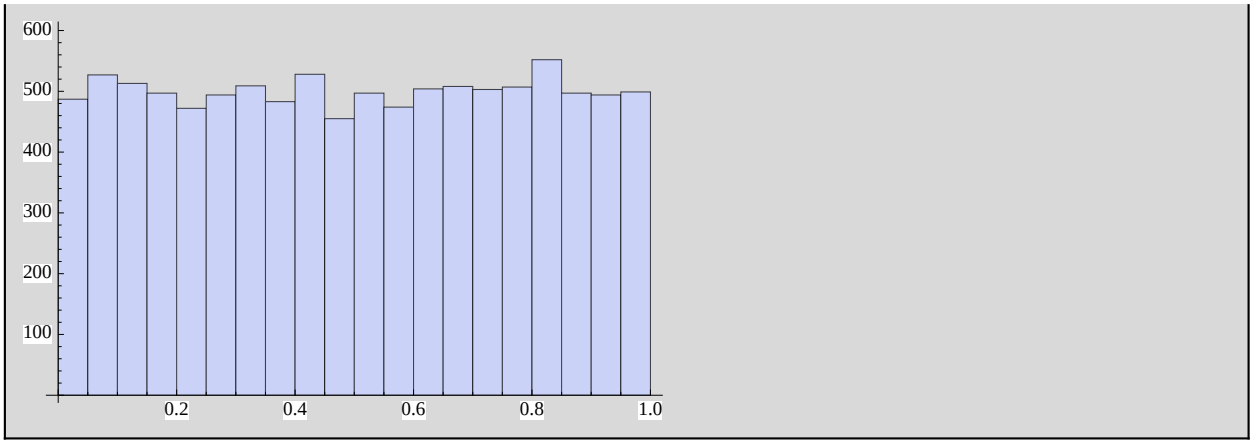
So lets look at what this function does with an example.

```
sampleUniform[0, 1, 5]
```

```
{0.262365, 0.250348, 0.557311, 0.526948, 0.890513}
```

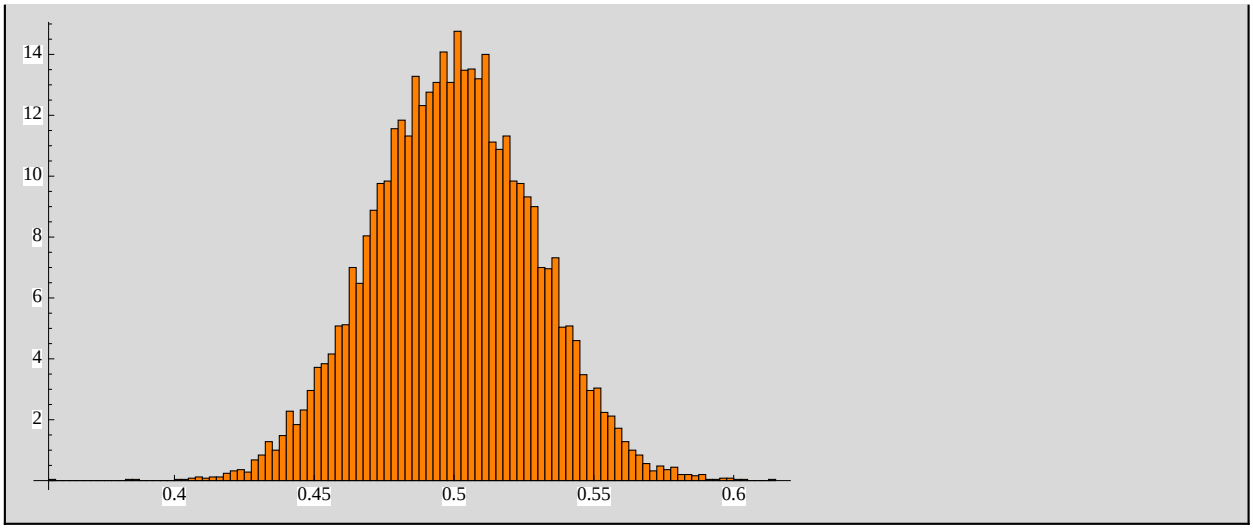
As you can see we get a list of 5 random number in the range 0->1. Now we can take a look at the PDF, or probability density function of the uniform distribution graphically. To do this we simply create a histogram of a large sample from the uniform distribution. Here we create a list with 10,000 random numbers in the random 0->1!! As you can see, the distribution is more or less uniform on the interval 0->1. If we increased the size of our sample we would expect a even better approximation to a square box, but since 10,000 gives us what we expect we can leave that for when we have a few hours to kill.

```
Histogram[sampleUniform[0, 1, 10 000]]
```



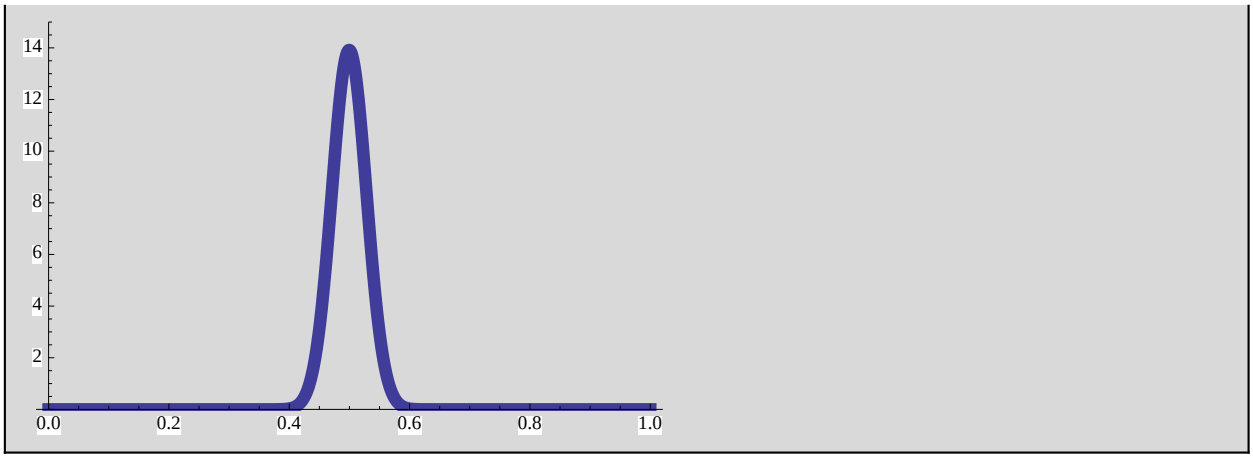
So now lets look at our sampling distribution. To create a sampling distribution, we take a large number of samples from our population (in this case our population is the uniform distribution). Each time we take a group of samples we take the mean of these observations and create a new distribution made up of these means. It is simple to extend our previous example to create the sampling distribution. We simple create a list of random numbers sampled from the uniform, take the mean of this list, and repeat and large number of times. We can then look at the distribution of the means of our samples. Lets do it 10,000 times.

```
data = Table[Mean[sampleUniform[0, 1, 100]], {10 000}];
myplot = Histogram[data, HistogramScale → 1, BarStyle → {Orange}]
```

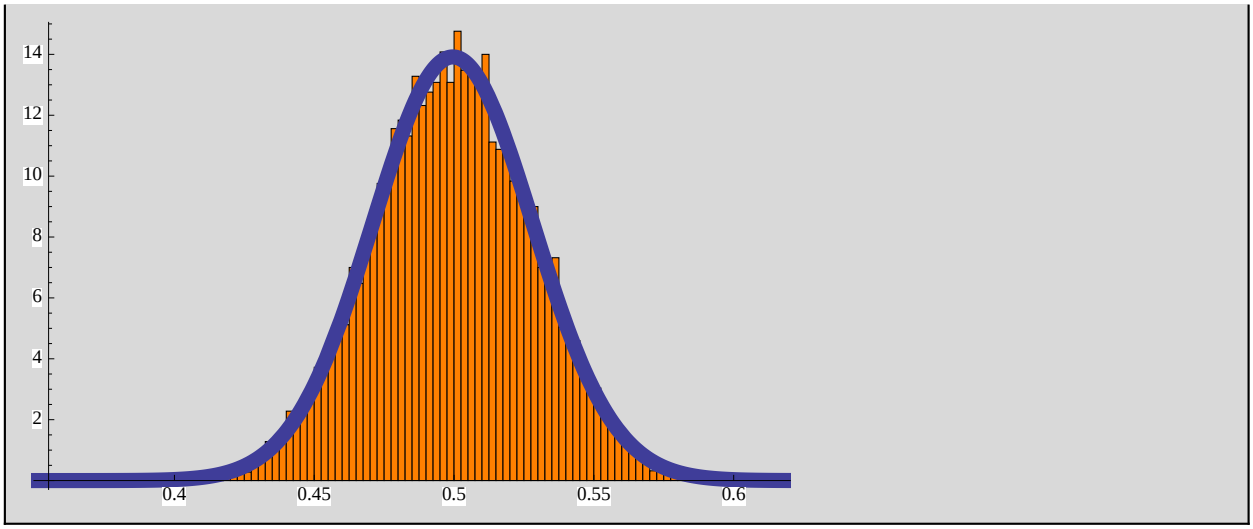


Wow! What a nice approximation to the Gaussian. If we forgot what the gaussian curve looks like, no problem, we can just plot the Gaussian distribution on the same plot so we can easily compare them. *Mathematica* has a built in function that is the closed form of the gaussian distribution. We just plug in the mean and standard deviation of our sampling distribution and we can compare how close to the normal our distribution really is.!

```
ndist = NormalDistribution[Mean[data], √Variance[data]];
Clear[f];
f = PDF[ndist, x];
myplot2 = Plot[f, {x, 0, 1},
  {PlotRange → {0, 15}, DisplayFunction → Identity, PlotStyle → {{Thickness[0.02`]}}}]
```

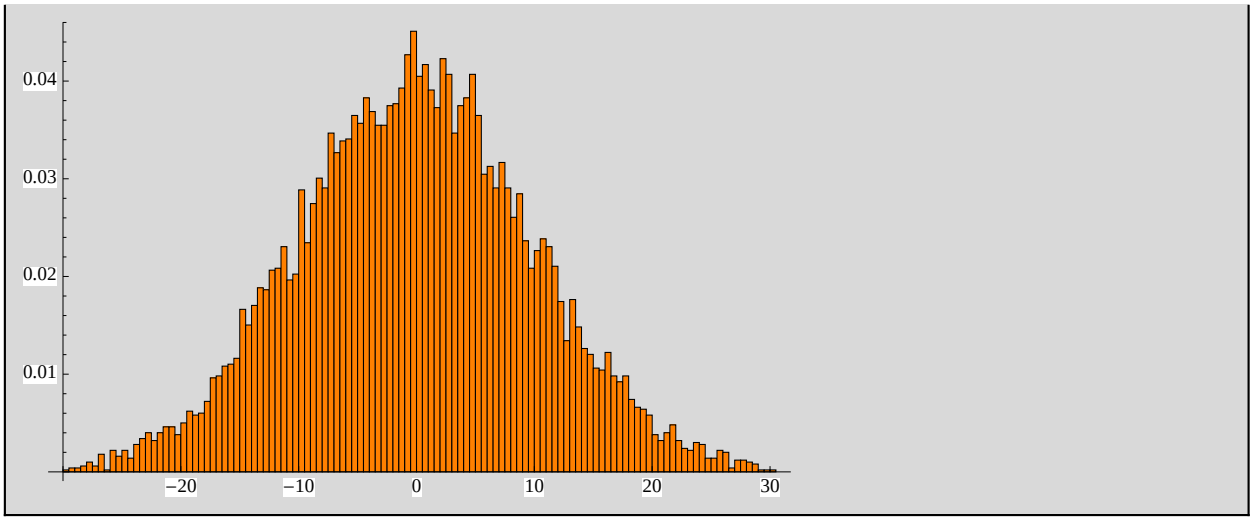



```
Show[myplot, myplot2]
```



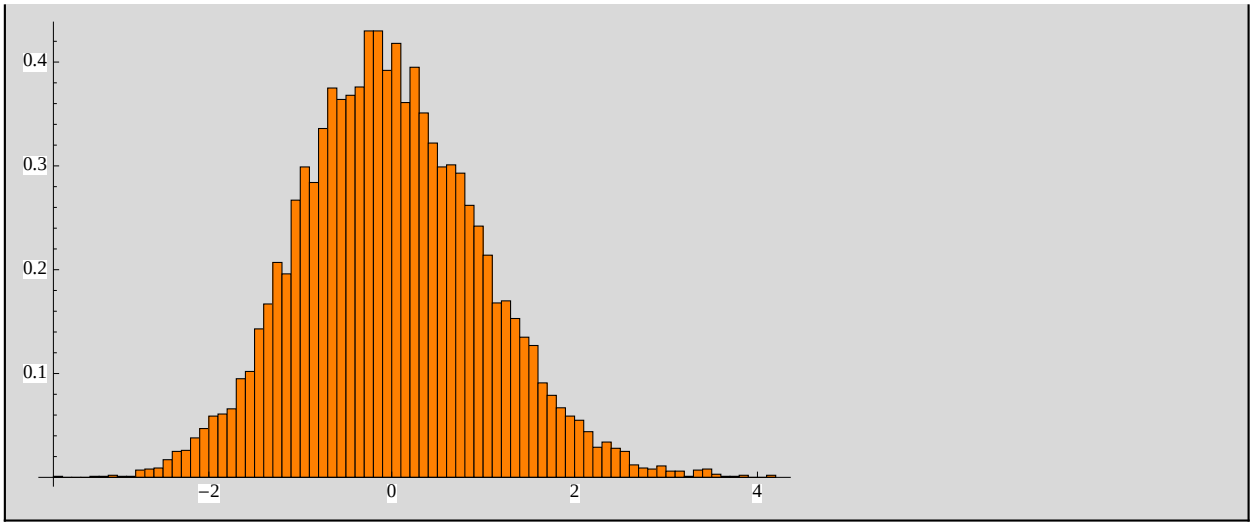
We can try this again with taking our original data from a normal

```
Histogram[sampleNorm[10 000], HistogramRange -> {-30, 30},
 HistogramScale → 1, BarStyle -> {Orange}]
```



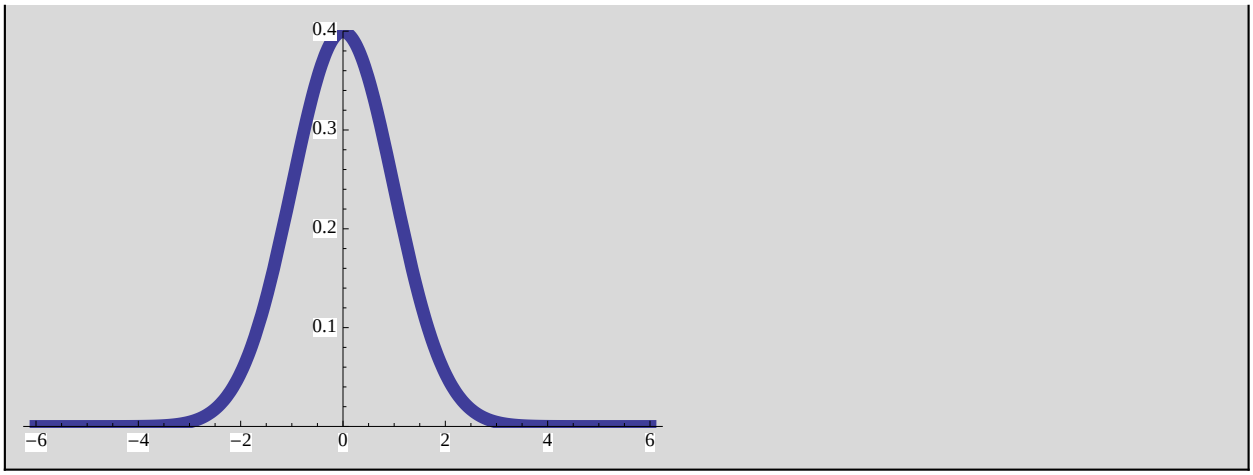
Now we can build the sampling distribution

```
data = Table[Mean[sampleNorm[100]], {10 000}];
myplot = Histogram[data, HistogramScale → 1, BarStyle → {Orange}]
```

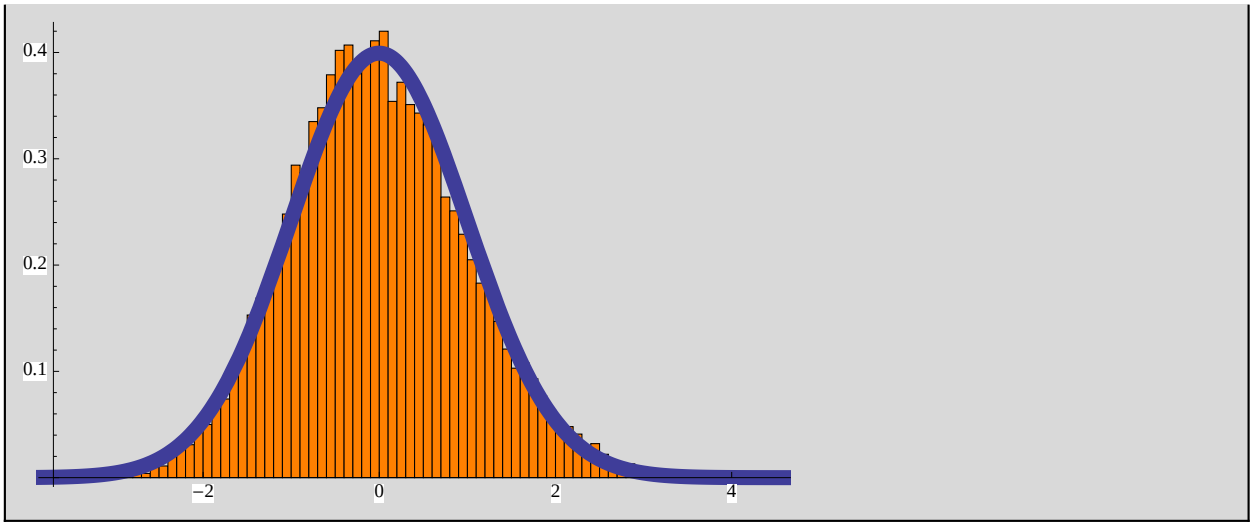


Finally we can compare this to a standard normal

```
ndist = NormalDistribution[Mean[data], √Variance[data]];
Clear[f];
f = PDF[ndist, x];
myplot2 = Plot[f, {x, -6, 6},
  {PlotRange → {0, 0.4}, DisplayFunction → Identity, PlotStyle → {{Thickness[0.02`]}}}]
```




```
Show[myplot, myplot2]
```



Thus, we have shown in at least two cases (one where the data is actually uniformly distributed, and one where it was actually normally distribted), the resulting distribution of means will be distributed according to the normal. the central limit theorm shows that no matter what our original distribution it, the mean generally will be normally distributed. Thus, if we observe a particular mean, say $\bar{x}$, we can assess how likely it by comparing it to this normal distribution.

z-Score

The next step is to use the normal distribution in order to make inferences. The first step in this is to measure how far the data is away from the mean of the distribution. One natural measure of this with normals it to measure the deviation of the observation away from the true mean (μ), divided by the standard deviation. In this case, our standard deviation acts as a "ruler" measuring how far away something has to be to be "far" or "close".

$$z = \frac{x - \mu}{\sigma}$$

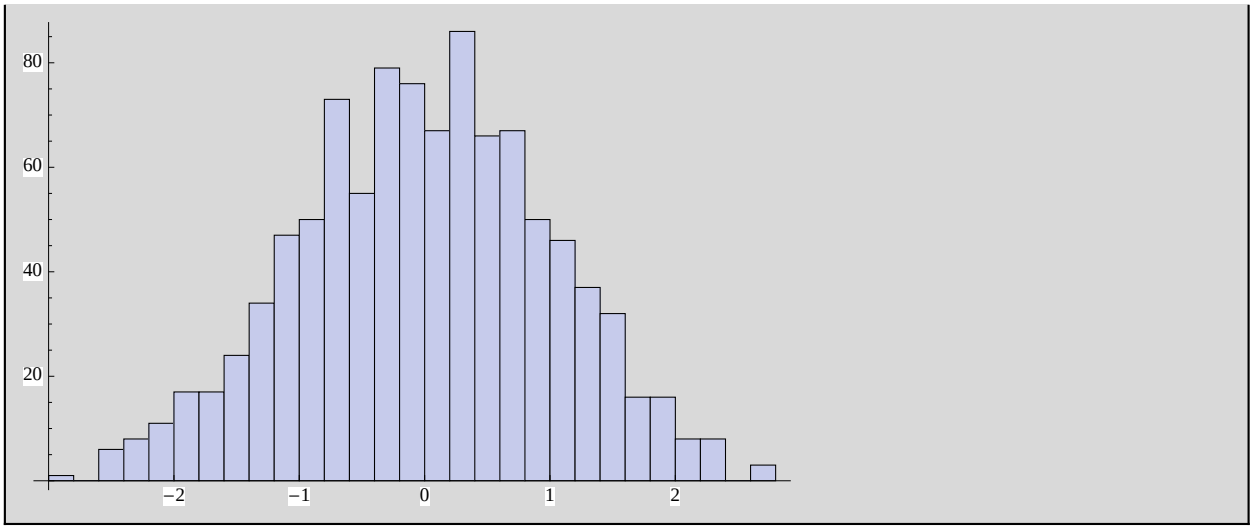
Basically this tells us how far away from the true mean (μ) our observation (x) is in units of the standard deviation (σ).

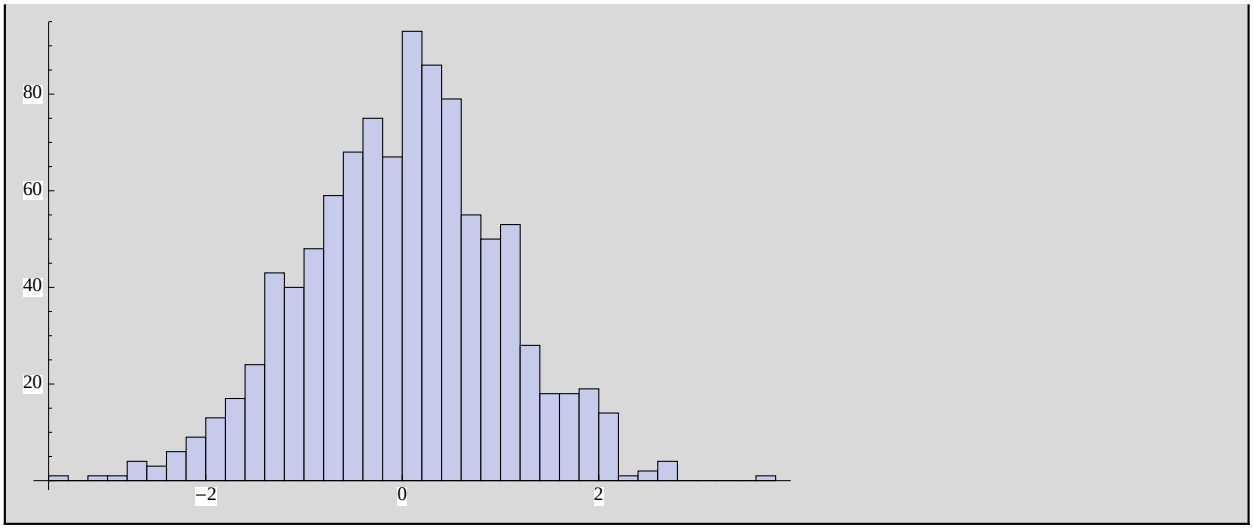
To make this clear, lets generate data from two normal distributions, each with different mean and sd.

```
ndist = NormalDistribution[0, 10];
sampleNorm1[n_] := Table[Random[ndist], {n}];
ndist2 = NormalDistribution[5, 20];
sampleNorm2[n_] := Table[Random[ndist2], {n}];
```

```
data1 = sampleNorm1[1000];
data2 = sampleNorm2[1000];
```

```
myplot1 = Histogram[(data1 - Mean[data1]) / StandardDeviation[data1]]
Histogram[(data2 - Mean[data2]) / StandardDeviation[data2]]
```





Thus, we have "standardized" our samples so they correspond to the same normal distribution (namely one with mean μ=0 and σ=1). We can then use the standard normal to compute probabilities of getting observations.

hide this

Computing probabilities from a z-score

test type: $H_a: \mu \geq \mu_0$ / $H_0: \mu = \mu_0$ | $H_a: \mu \leq \mu_0$ / $H_0: \mu = \mu_0$ | $H_a: \mu \neq \mu_0$ / $H_0: \mu = \mu_0$

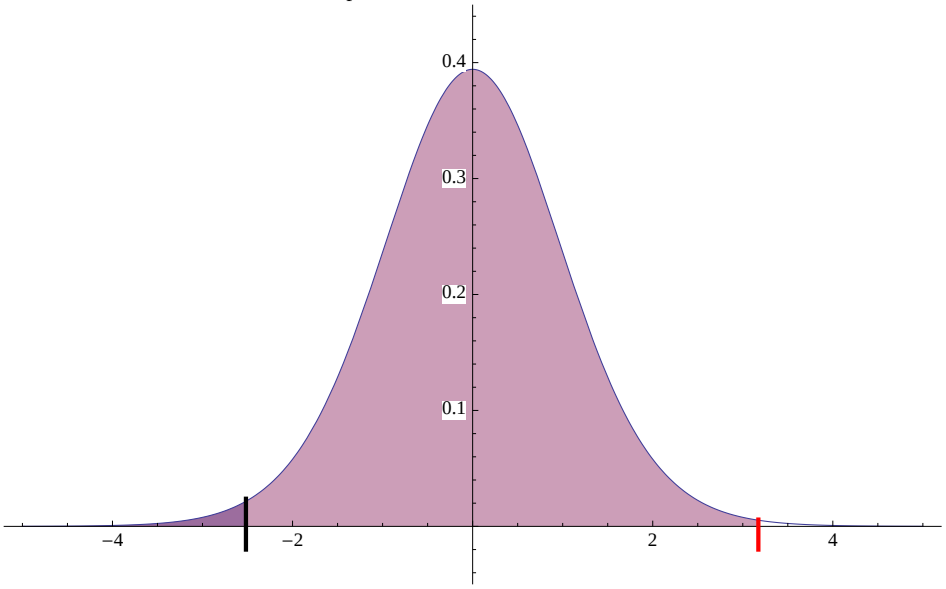
sample size μ_0

α population variance

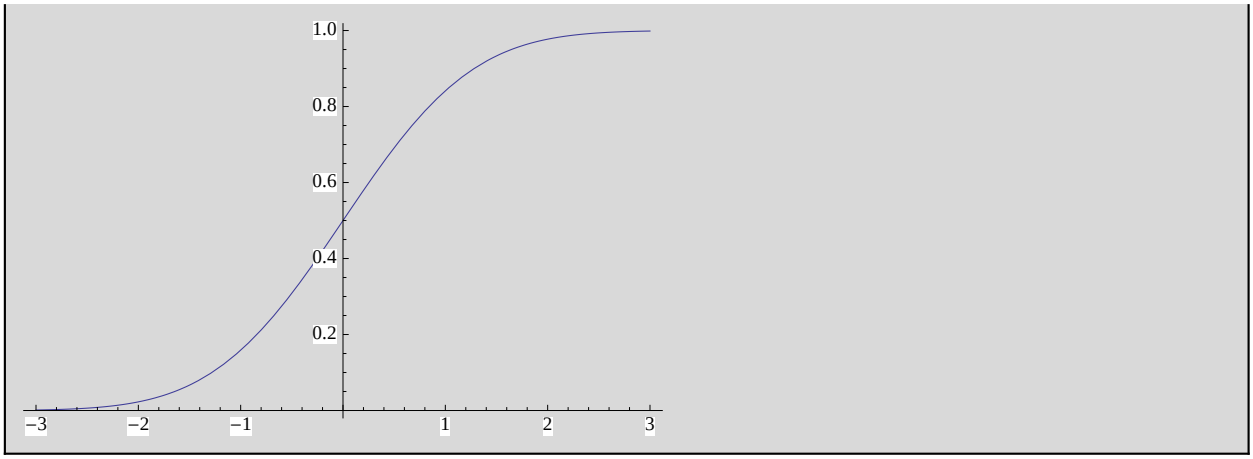
random seed

conclusion: do not reject H_0

$p-$value: 0.99771



```
Plot[CDF[NormalDistribution[0, 1], x], {x, -3, 3}]
```




```
CDF[NormalDistribution[0, 1], 3.5] (* prob of t less than 3.5 *)
1 - CDF[NormalDistribution[0, 1], 3.5] (* prob of z greater than 3.5 *)
```

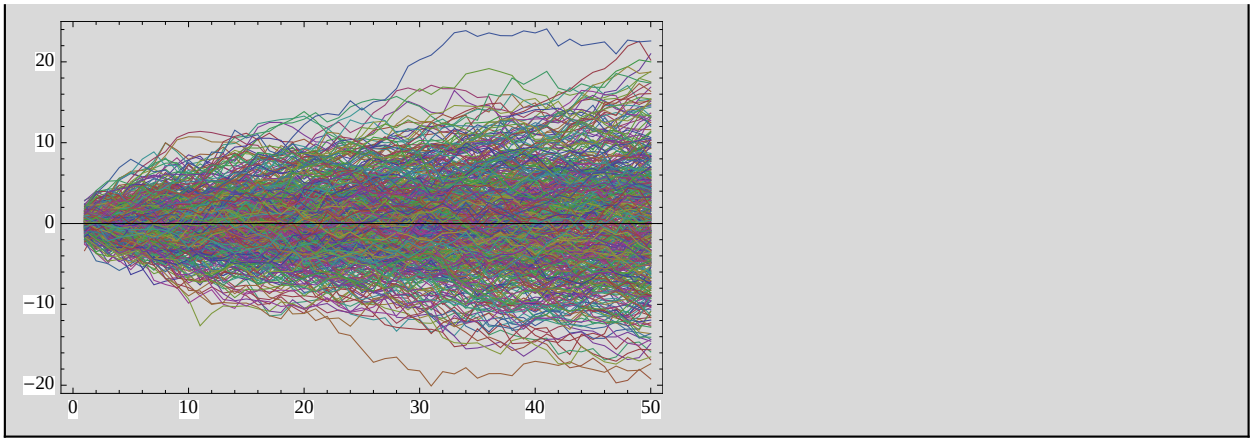
```
0.999767
```

```
0.000232629
```

Ok, so now we have some sense of why the normal distribution is used. However, one question you might be wondering is how the sampling distribution of the mean changes as the number of observations go into the mean. For example, does the sampling distribution of the mean change if we have only made two observations or if we have made 1000?

The following graph shows just that situations. Data are being generated according to the normal distribution. This plots the sum of the data experienced as a function of the number of trials. This is akin to a random walk. The sum starts out at zero, then goes up or down. The more observations we get the further this "random walk" gets from the starting point.

```
ndist = NormalDistribution[0, 1];
p2 = ListPlot[
  Table[Accumulate[Table[Random[ndist], {50}]], {500}], Joined → True, Frame → True]
```

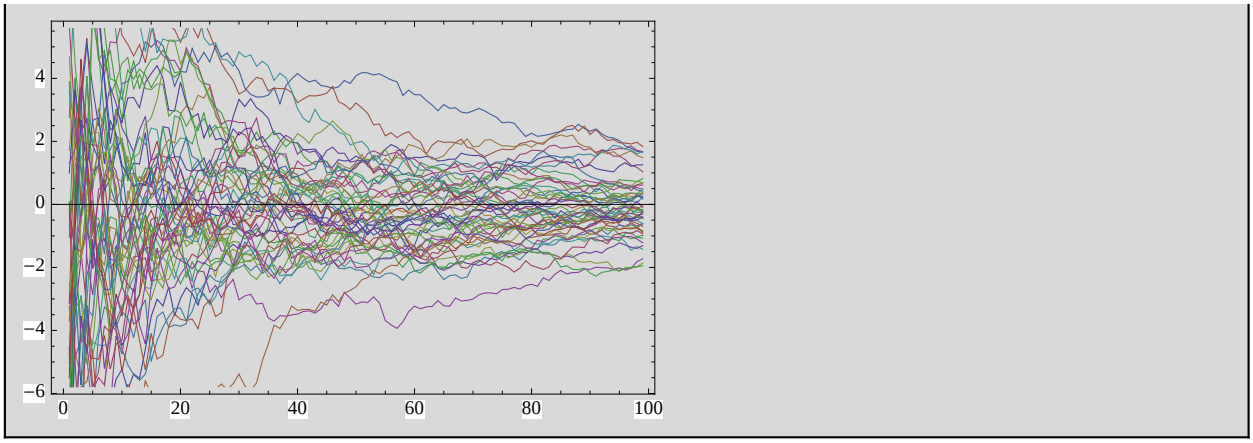


Since the sum is realated to the mean by a constant factor (divide by N) we can look at how additional observation change our estimate of the mean. The next graph does the exact same thing, but calculates the mean each time a new observation is added. Each path is a different random experiment with different observation (think of each line as a different lab running the same experiment in a different place with different subjects). As you can see the variability in the mean is actually going down as the number of observations goes up.

```
incmean[x_] := Map[Mean[Take[x, #]] &, Range[Length[x]]]

walk[n_] := Module[
    {data},
    data = Table[Random[ndist], {n}];
    Drop[incmean[data], 1]
   ];
```

```
p1 = ListPlot[
    Table[walk[100], {50}], Joined → True, Frame → True];

p2 = ListPlot[
    Table[walk[50], {50}], Joined → True, Frame → True];

p3 = Plot[10 + (10 / Sqrt[n]), {n, 0, 100}, PlotStyle → {{Thickness[0.02`]}}];
```

```
Show[p1]
```

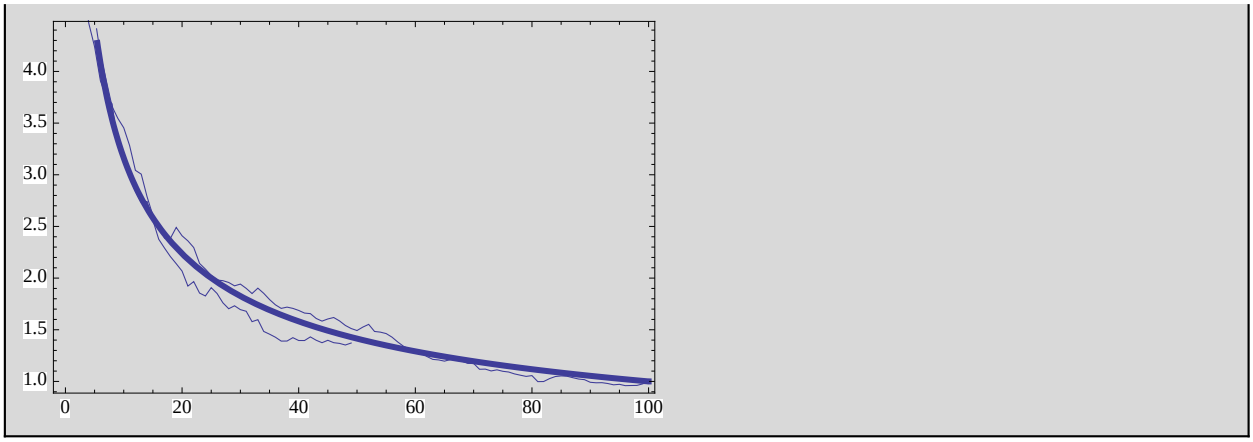


In fact, we can calcualte the expected standard deviation as the number of observations goes up in this simulations by just calculating the standard deviation of the means with N=1, N=2, N=4, ... N=100.

```
p1 = ListPlot[
    Map[StandardDeviation, Transpose[Table[walk[100], {50}]]], Joined → True, Frame → True];

p2 = ListPlot[
    Map[StandardDeviation, Transpose[Table[walk[50], {50}]]], Joined → True, Frame → True];

p3 = Plot[10 / Sqrt[n], {n, 0, 100}, PlotStyle → {{Thickness[0.01`]}}];
```

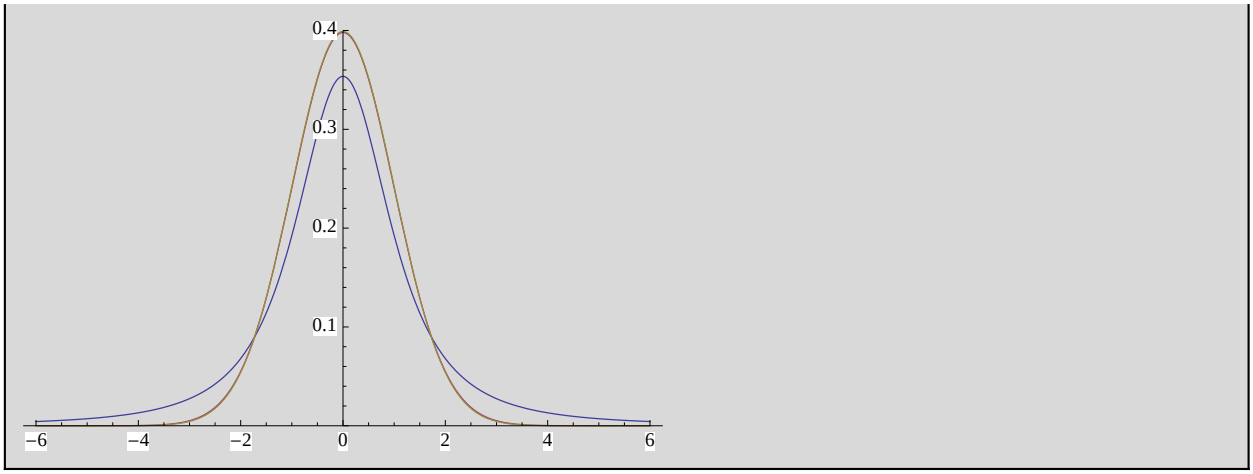
What we see is that the standard deviation of the sampling distribution of the mean is scaling with $\frac{1}{\sqrt{N}}$. In fact, the dark blue line plotted below is exactly that (it shows $10/\sqrt{N}$). Thus, our estimate of the standard deviation of the mean distribution should change with N, in fact by a factor $\sqrt{N}$. This is why the formula for standard error of the mean is SE=$\frac{s}{\sqrt{N}}$where s is the standard deviation of the sample. As N gets bigger, the $\sqrt{N}$ approches infinity, and then $1/\sqrt{N}$ approaches zero. Thus, as the sample gets really big, the standard deviation of the sampling distribution of the mean becomes simply the standard deviation of our sample. At intermediate sample sizes though, we need a correction. This correction is formalize by the Student's T distribution.

```
Show[p1, p2, p3]
```



The Student's T-distribution and the T-Score

```
ndist = StudentTDistribution[2];
f = PDF[ndist, x];
g = PDF[StudentTDistribution[100], x];
h = PDF[NormalDistribution[0, 1], x];
myplot2 =
 Plot[{f, g, h}, {x, -6, 6}, {PlotRange → {0, 0.4}, PlotStyle → {{Thickness[0.002`]}}}]
```



The students T-distribution is approximately the same as the normal, but the sample standard deviation is scaled by a factor $\sqrt{N}$ (rather than the population value of the standard deviation). Above the blue line is the student t-distribution for N=2, and the yellow line is a normal. As N gets bigger the blue line will get close and close to the yellow line (showing how the T-distribution converges to the normal for a big N).

Given this fact, and given that we are making inferences about unknown distributions (i.e, we don't know the population μ or σ, we have to use this "corrected" distribution in making our inferences. Thus, we use a t-score (related to the z-score) to measure how "surprising" our observed mean $\overline{x}$ is compared to the null hypothesis (that the true mean is μ_{H_0}).

$$t = \frac{\overline{x} - \mu_{H_0}}{s \Big/ \sqrt{n}}$$

Given, a particular t value we can compute the probability of observing that value according to the student T-distribution with degrees of freedom N-1.