CLT: Is a set of means normally distributed?

version 2b, 23-Feb-2012
author: Craig Shallahamer, craig@orapub.com / orapub.general@gmail.com

Introduction

I wanted to flexibly, using a small number of values, using a large number of values, using various statistical distributions demonstrate a central limit theorem claim: *If you divide up a population into sample sets and calculate the mean of each sample set, the distirbution of the means will be normal.* This is my first test.

Because the sample set of means is normal, we can make all the typical claim that normality provides. This is my second test.

This particular file is based on my February 2012 blog posting entitled, "Wanna Bet." For that particular test, a 900 value sample set is being used. I used an exponential distribution because it is obviously not normally distributed.

Sample values

I have included three possibilities below. Just make sure the sample set the sampleSetRaw variable to your sample set!

In[1]:=

```
cutoffMin = -9999.0;
cutoffMax = 9999.0;

SeedRandom[13]; (* 13 used for blog posting *)
ssReal = RandomReal[{0, 500}, 3000];
ssNormal = RandomReal[NormalDistribution[50, 5], 3000];
ssExpo = RandomReal[ExponentialDistribution[1 / 5], 900];
ssChi = RandomVariate[ChiSquareDistribution[35], 500];
ssBlog = RandomVariate[ChiSquareDistribution[40], 75];

sampleSetRaw = ssExpo;
```

Basic Numeric Statistics

The main basic numeric statistics shown are:

First five values simply lists the first five values of your sample set. Use this to check the data being used is what you think.
Number of samples is in fact the number of samples. Use this to check the data has been entered correctly.
Average is the statistical mean.
Median is the middle value after all the values are sorted. This is also the 50-percentile value.
Standard deviation is a measure of dispersion and is particularly valuable when the distribution is normal.
P-Value is a measure of contrast. In this instance, we are constrasting the sample set to the normal distribution. Loosely speaking, if the P-Value is greater than 0.05 then our sample set is likely to be normally distributed.

In[9]:=

```
sampleSetGood = {};
sampleSetBad = {};
Table[
  If[((sampleSetRaw[[i]] ≤ cutoffMax) && (sampleSetRaw[[i]] ≥ cutoffMin)),
   AppendTo[sampleSetGood, sampleSetRaw[[i]] ],
   AppendTo[sampleSetBad, sampleSetRaw[[i]] ]
  ]
  , {i, 1, Length[sampleSetRaw]}
 ];

countRaw = Length[sampleSetRaw];
countGood = Length[sampleSetGood];
countBad = Length[sampleSetBad];

firstFiveRaw = Take[sampleSetRaw, 5];
firstFiveGood = Take[sampleSetGood, 5];
firstFiveBad = Take[sampleSetBad, 5];

avgParent = Round[N[Mean[sampleSetGood]], 0.0000010];
medParent = Round[N[Median[sampleSetGood]], 0.0000010];
stdParent = Round[N[StandardDeviation[sampleSetGood]], 0.000010];
varParent = Variance[sampleSetGood];
pValue = Round[N[DistributionFitTest[sampleSetGood]], 0.0000010];
pct90 = Round[N[Quantile[sampleSetGood, 0.90]], 0.000010];
pct95 = Round[N[Quantile[sampleSetGood, 0.95]], 0.000010];
pct99 = Round[N[Quantile[sampleSetGood, 0.99]], 0.000010];
maxVParent = Max[sampleSetGood];

Grid[{
  {"Number of total samples", countRaw},
  {"Number of good samples", countGood},
  {"Number of bad samples", countBad},
  {"First five raw samples", firstFiveRaw},
  {"First five good samples", firstFiveGood},
  {"First five bad samples", firstFiveBad},
  {"Good Sample Details", "---"},
  {"  Mean", avgParent},
  {"  Median (50%-tile)", medParent},
  {"  Maximum", maxVParent},
  {"  Percentiles (90,95,99)", {pct90, pct95, pct99}},
  {"  Variance", Round[N[varParent], 0.00000010]},
  {"  Standard deviation", stdParent},
  {"  P-Value", pValue}
 },
 {Alignment → {Left},
  Frame → None}
]
```

Take::take : Cannot take positions 1 through 5 in {}. ≫

Out[27]=

```
Number of total samples   900
Number of good samples    900
Number of bad samples     0
First five raw samples    {1.65495, 2.12077, 3.98637, 1.50936, 12.9898}
First five good samples   {1.65495, 2.12077, 3.98637, 1.50936, 12.9898}
First five bad samples    Take[{}, 5]
Good Sample Details       ---
  Mean                    5.34305
  Median (50%-tile)       3.46265
  Maximum                 47.1627
  Percentiles (90,95,99)  {11.8099, 16.964, 23.8207}
  Variance                31.1043
  Standard deviation      5.57712
  P-Value                 0.
```

Basic Visual "Statistics"

Histograms are a fantastic way to get a quick grasp of a large number of samples. Below are a select number of histogram, each focusing on a specific numeric quality.

In[28]:=

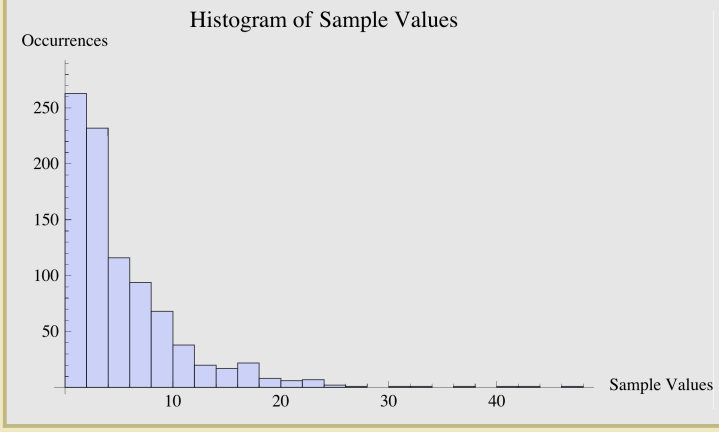
```
hLabel = "Sample Values";
vLabel = "Occurrences";

histStnd = Histogram[sampleSetGood,
    PlotLabel → "Histogram of Sample Values", AxesLabel → {hLabel, vLabel}];
histStndSmooth = SmoothHistogram[sampleSetGood, PlotLabel →
    "Smoothed Histogram of Sample Values\n(Probability Distribution Function)",
  AxesLabel → {hLabel, ""}]; histCC = Histogram[sampleSetGood, Automatic,
  "CumulativeCount", PlotLabel → "Histogram of Sample Values\nCumulative Count",
  AxesLabel → {hLabel, vLabel}];
histProb = Histogram[sampleSetGood, Automatic, "Probability", PlotLabel →
     "Histogram of Sample Values\nProbability", AxesLabel → {hLabel, "% Occurs"}];
histStndSmallBin = Histogram[sampleSetGood, {0.250}, PlotLabel →
     "Histogram of Sample Values\nbin size 0.250", AxesLabel → {hLabel, vLabel}];
Print[
  "
    "];
```

Below is a standard histogram, where each sample is shown as a single block placed on the vertical axis based on its value. Common sample values (i.e., blocks) show as high stacks.

In[35]:=

```
histStnd
```

Out[35]=

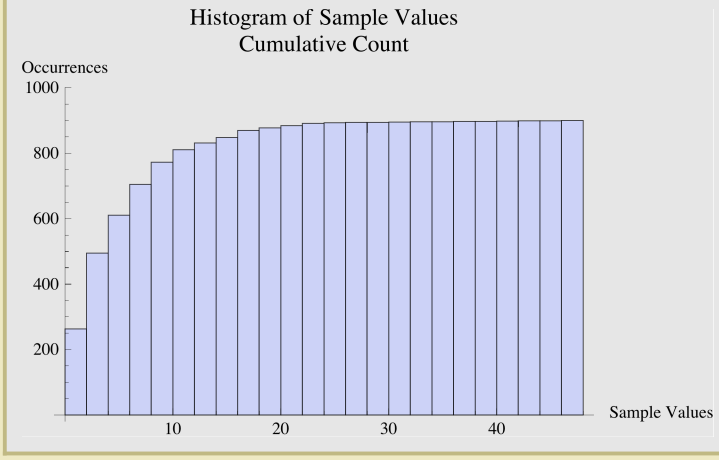


Below is known as a Cummlative Count histogram. Each vertical bar represents the total number of samples values that are less than or equal to the bin.

In[36]:=

```
histCC
```

Out[36]=

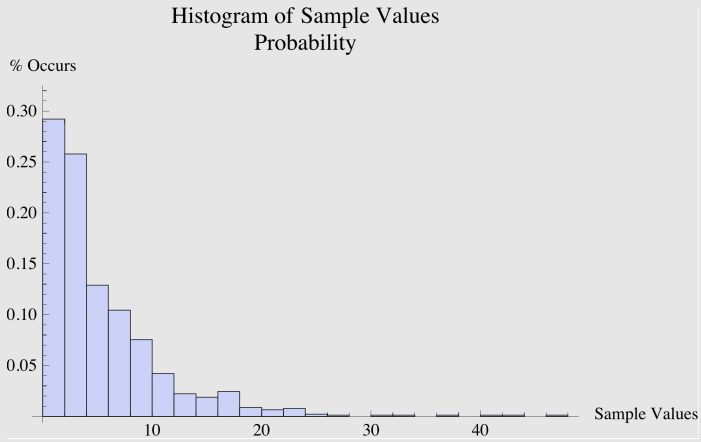


Below is a Probability Historgram. It will visually look exactly like the standard histogram but the vertical axis is a percentage value. Each vertical bar's height represents the percentage of values that is contains. In contrast, the standard histogram hight is the actual number of sample occurrences.

In[37]:=

```
histProb
```

Out[37]=

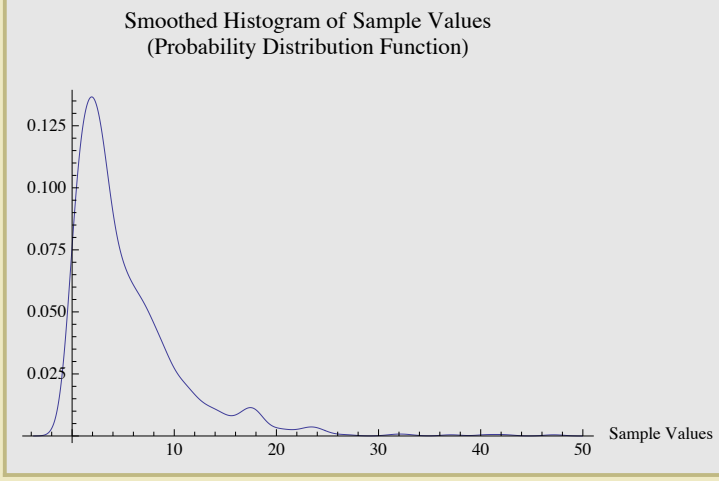


Below is a Smoothed Histogram. It will have a similar shape to the Standard Histogram, but will be mathematically smoothed. Sometimes this is a much more pleasent and informative visual, but not always. Remember, it is smoothed so it does not consist of the actual values. For example, you may see the line go negative, even though there are no negative values. In reality, the plot is the probability distribution function (PDF).

In[38]:=

```
histStndSmooth
```

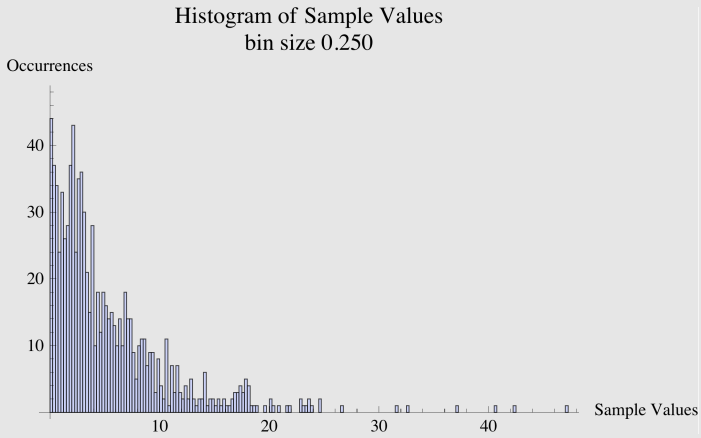
Out[38]=



Below is a standard histogram but with the bin size set to 0.250. This is only useful when the sample values range below 1.0, such as when sampling SQL statements (we all hope). Sample sets with large sample values will likely not result in a plot.

In[39]:=
```
histStndSmallBin
```

Out[39]=



Are child sample set MEANS normally distributed?

According to the central limit theorem, regardless of a parent sample set's distribution (e.g., normal), dividing the parent set into many smaller sample sets, and then creating a sapmle set based on their averages (the means of the child sample sets), the distribution of the child sample set's means is supposed to be normally distributed. In fact, this is supposed to be true regardless of the parent sample set's distribution! But is this really true? This is what this section is all about. Have fun!

Here's an example of how you can test this. Create a 900 value sample set based on a real life distribution, such as normal, exponential, or uniform. Randomlly create 30 sub sample sets from the original sample set, each with 30 values. Compute the mean for each of the 30 subsets and place these into a new sample set. Based on this new sample set, calculate the basic stats and plot the histogram... it's normally distributed! ...the larger our sample sets the more likely this is true.

Here is a good link: http://www.statisticalengineering.com/central_limit_theorem.htm

In[40]:=
```
subScript = 6; (* this is where I start pulling from the parent set,
actually the superSet, to create the child saple sets *)
subSSnumber = IntegerPart[(Length[sampleSetGood]^0.5)];

subSSLength = IntegerPart[Length[sampleSetGood] / subSSnumber];
superSet = Join[sampleSetGood, sampleSetGood];

Table[
  subSSValues[i] = {};
  Table[
   subScript = subScript + 1;
   AppendTo[subSSValues[i], superSet[[subScript]]];
   , {j, 1, subSSLength}
  ];
  , {i, 1, subSSnumber}
 ];
subSSValues[1];
subSSValues[2];
subSSMeans = {};
Table[
  AppendTo[subSSMeans, Mean[subSSValues[i]] ];
  , {i, 1, subSSnumber}
 ];
```

In[49]:=
```
subSSMeans
```
Out[49]=
```
{6.35511, 4.91329, 3.7737, 5.27418, 6.52658, 7.68304, 4.4319, 6.09553, 4.64664, 6.10894,
 4.422, 5.21073, 4.33155, 6.78821, 5.38437, 5.38456, 5.80179, 4.60562, 4.87516, 6.77315,
 5.38634, 5.55096, 5.43312, 4.48588, 4.75642, 5.06397, 4.43674, 4.72971, 5.00164, 6.06055}
```

In[50]:=
```
Length[subSSMeans]
```
Out[50]=
```
30
```

In[51]:=
```
Length[subSSValues[1]]
```
Out[51]=
```
30
```

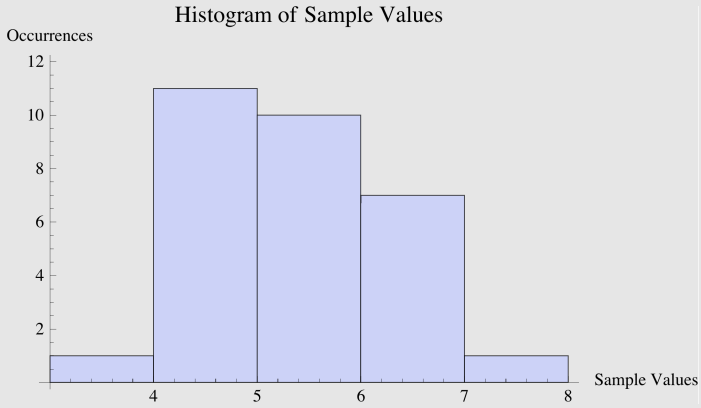
In[52]:=
```
DistributionFitTest[subSSMeans]
```
Out[52]=
```
0.184393
```

In[53]:=
```
Mean[subSSMeans]
```
Out[53]=
```
5.34305
```

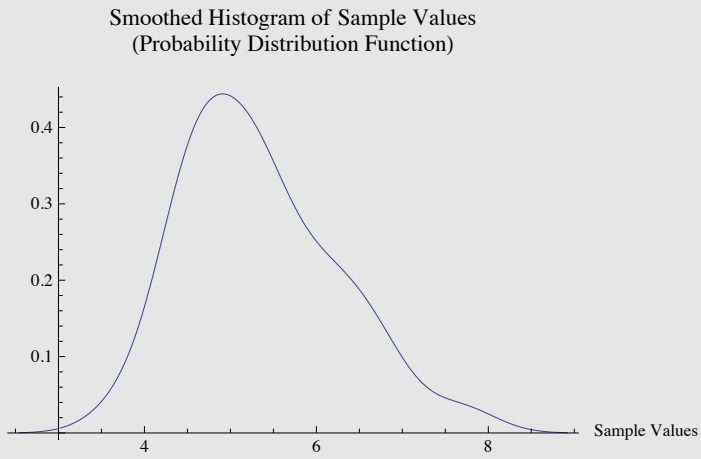
In[54]:=
```
StandardDeviation[subSSMeans]
```
Out[54]=
```
0.891705
```

In[55]:=
```
hLabel = "Sample Values";
vLabel = "Occurrences";

histStnd = Histogram[subSSMeans,
   PlotLabel → "Histogram of Sample Values", AxesLabel → {hLabel, vLabel}];
histStndSmooth = SmoothHistogram[subSSMeans, PlotLabel →
    "Smoothed Histogram of Sample Values\n(Probability Distribution Function)",
   AxesLabel → {hLabel, ""}];
histStnd
histStndSmooth
```

Out[59]=



Out[60]=



I want to demonstrate that 95% of the child sample set (of means) fall within 2*stdev of the population mean.

While this is kind of technical, the standard deviation is describing our samples, not an entire population. So what we are saying is only technically valid when related our 30 samples. If we are making inferences beyond that, we need to calculate the “population standard deviation” which has slightly different math. But we’re OK in this example because we are refering specifically to our sample set.

In[61]:=
```
avgSSMean = Mean[subSSMeans];
avgSSStd = StandardDeviation[subSSMeans]
avgSSHigh = avgSSMean + 2 * avgSSStd
avgSSLow = avgSSMean - 2 * avgSSStd
yesCntrStd = 0;
noCntrStd = 0;

Table[
  (*Print[i," ",subSSMeans[[i]]];*)
  If[(avgSSLow ≤ subSSMeans[[i]]) && (avgSSHigh ≥ subSSMeans[[i]]),
    yesCntrStd = yesCntrStd + 1,
    noCntrStd = noCntrStd + 1
   ];
  , {i, 1, subSSnumber}
 ];
Print["Yes=", yesCntrStd, " No=", noCntrStd,
  " Yes%=", N[yesCntrStd / (yesCntrStd + noCntrStd)]];
```

Out[62]=
```
0.891705
```

Out[63]=
```
7.12646
```

Out[64]=
```
3.55964
```

```
Yes=29 No=1 Yes%=0.966667
```