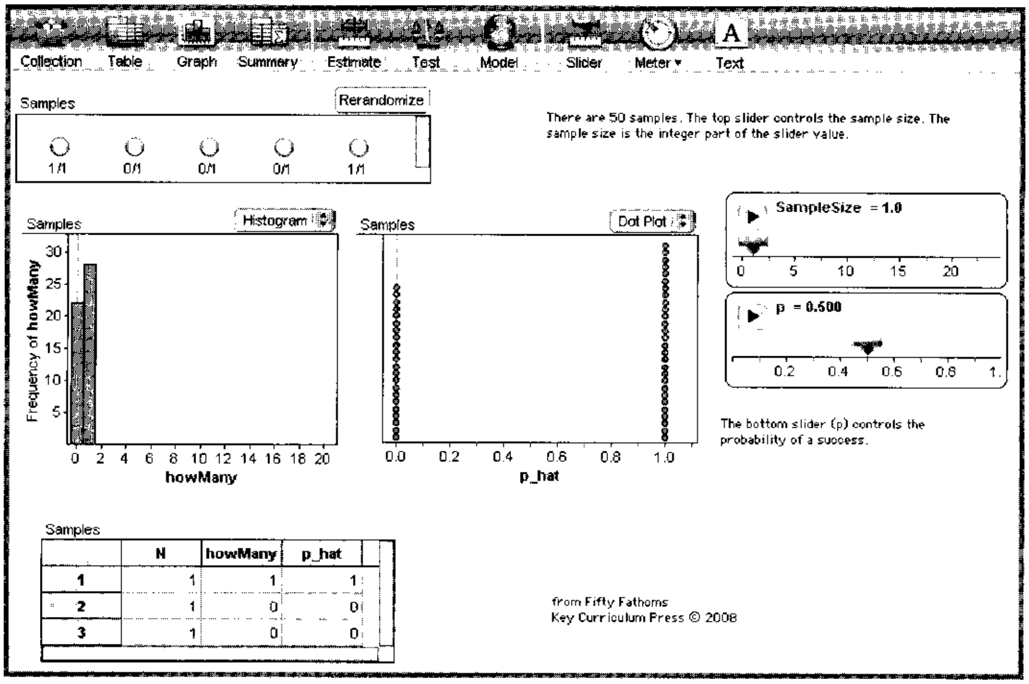
Demo 14: More Binomial

How the binomial distribution depends on sample size for small N • The relationship between the distribution of sample proportions and the distribution of sample counts

In this demo, we're not going to make the samples and then count how many successes there are in each set of draws. Instead, we'll use Fathom's built-in **randomBinomial( )** function to skip all that (therefore, if you have any doubts about what's going on, you should look at Demo 13, "Building the Binomial Distribution").

In this way we'll take a different look at what's confusing: How is it that we can take this yes/no kind of event—a binary event—and turn it into a distribution that's pretty smooth and has a hump in the middle? This time, we'll use Fathom's dynamic sliders to control the *sample size.*



What To Do

▷ Open **Binomial Small N.ftm.** It will look something like the illustration.

Each case in the collection represents one sample; the caption under each case—each gold ball—tells how many "successes" there were in how many trials. This could be heads in flips of the coin, black cards in a set of draws, or whatever. The graph at left in the illustration shows a histogram of the counts. At right are two sliders with explanatory text. The integer part of **SampleSize** is the sample size[1]. Below the graph you can see the first three cases in the data collection. There are 50 cases altogether, that is, 50 samples of size 1.

In the picture, our one object has a 50% chance of success. So, in our 50 cycles, we see about 25 cases with zero successes and about 25 with one success, which is what you'd expect. In the graph to the right of the histogram you see the same data represented as proportions instead of absolute numbers. The possible values with one card are **p_hat = 0** and **p_hat = 1.**

▷ Click the **Rerandomize** button on the **Samples** collection in the upper left. (Or choose **Rerandomize** from the **Collection** menu.) Observe what happens.

[1]We would have used **N**, but we're using that for the attribute in the case table. So the *slider* is **SampleSize**, the *attribute* is **N**. They mean the same thing conceptually.

- ▹ Move the slider named **p**, which controls the probability of a success. See what that does to the two graphs. Reset that probability to about 0.5.
- ▹ Finally, gently move the **SampleSize** slider. Watch what happens to the two graphs. Try to understand what happens when the sample sizes are 1, 2, 3, and 4—and then how the graph gets to look as it does at around $n = 20$.
- ▹ Try changing sample size for different values of **p**. See what happens.

It's easy to look at the **p_hat** graph for $n = 1$ and the one for $n = 20$ and think that they are two completely different situations. One is grossly bimodal, just two spikes at opposite ends. The other is a classic distribution, practically normal, with *nothing* at the ends. If you stare at that **p_hat** graph as you move the **SampleSize** slider, it can be hard to make sense of it.

On the other hand, looking at the **howMany** graph—the one in absolute number instead of in proportion—may help clear it up. After all, when $n = 1$, it's just a big spike, over on the left. As you add sample size, the hump gets less steep and moves to the right. The process is continuous, not a strange transformation from one kind of graph to something completely different. This is, I suppose, an advertisement for looking at a variety of representations in order to understand something.

Note: The **SampleSize** slider has been restricted so that it shows only integers. You do that in the slider's inspector.

Questions

1 In that first step, when you clicked **Rerandomize,** what happened? How much variation was there in the number of "0" and "1" cases?

2 Consider the case where **SampleSize** = 1 (as at the beginning) but where **p** has been changed. How could you estimate **p** using only the graphs?

3 What happens to the graphs when you add one more to the sample size?

4 How can you predict where the peak will be in the **howMany** graph? How about the **p_hat** graph? **Sol**

5 How do the two graphs relate to one another?

6 What is the algebraic relationship among **howMany, N,** and **p_hat**?

Extension

You can put the theoretical binomial function on the **howMany** graph like this:

- ▹ Click on the graph to select it.
- ▹ Choose **Plot Function** from the **Graph** menu.
- ▹ Enter

  binomialProbability(round(x), last(N), p) * count( )

  Press **OK** to close the formula editor. The function appears; the data should roughly match it.

Why those unusual arguments? That function, **binomialProbability,** takes three (or more) arguments: the number you're interested in x, the sample size, and the probability. So **binomialProbability(3, 4, .5)** is the chance that you get exactly 3 successes in 4 tries with probability 0.5. But the first two have to be whole numbers; otherwise, it doesn't tell you anything. So we have to round and do other things to avoid decimals.