Coins, Trees, Bars and Bells: Simulation of the Binomial Process

The cointoss experiment can be visualized as a tree, generalized as the binomial distribution, and made continuous as the normal law of errors.

by Lee De Cola

In science, error is the difference between data and some model. For example, the geographic position, or {EAST, NORTH} data coordinates, of many objects on the earth can be determined quite precisely, say with global positioning systems (GPS). A map, on the other hand, is a model that tells the user where objects should be found, and an accurate map is one with small errors, or, more precisely, with a low probability of finding large differences between actual GPS and predicted map position.

The standard U.S. Geological Survey (USGS) topographic maps display the statement "This map complies with National Map Accuracy Standards." This standard states that "For maps on publication scales larger than 1:20,000, not more than 10 percent of the points tested shall be in error by more than 1/30 inch...."[Thompson 1988, page 104]. What exactly does this mean?

Consider the following experiment. The USGS headquarters are located in Reston Virginia, which is shown on the 1:24,000 scale quadrangle map of Vienna VA. Suppose that I have determined the map and GPS coordinates of 100 easily located points (road intersections, building corners, bench marks, etc.). The difference between GPS and map locations (say in meters) gives me a sample of 100 errors, and the map accuracy standards require among other things, that no more than 10 of these may deviate from their true locations by more than about 12 meters:

```
Needs["Miscellaneous`Units`"]

scale = 1/24000 ;

Convert[1/50 Inch, Meter] / scale
```

```
12.192 Meter
```

If this were an article about maps and projections, we could discuss how you could convert these map measurements into errors, and how--say using state-of-the-art GPS--you could determine the "true" locations of the points to within centimeters. But for the moment let's assume that the 100 errors are given. This is clearly a statistical problem, because whenever the discussion involves concepts like sampling, errors, and percentages, it is certain that probability and statistics are involved.

TOSSING A COIN

The toss of a coin has long been regarded as a ready illustration of the simplest probability distribution, the binomial process. The random function returns a random bit value that can be associated with heads or tails. Here is a toss function:

```
toss := If[ Random[Integer] == 1, H, T]
```

Every time the object is called a head or a tail is returned.

```
toss
toss
toss
```

```
H

T

H
```

When you toss a coin n times you get a random sequence of heads and tails. Here is the record of 20 tosses:

```
TableForm[ Table[ toss,  {20} ], TableDirections -> Row]
```

```
H T T T T T H H H H H T T H T T H T T T
```

A BINARY TREE

This particular outcome of the cointoss experiment can be regarded as a unique path along a complete binary tree, with individual upper branches signifying heads and the lower branches tails. Here is the first line of such a tree:

```
firstline = Line[ { {-1, 0}, { 0, 1} }] ;
```

Glynn and Grey [1991, ch. 6] present a technique for making trees that branch circularly outward from a vertical trunk. I have modified their approach to make branches by attaching to the end of each branch 2 new and smaller lines whose own ends are shifted vertically, the upward line for heads, the downward line for tails.

```
yshrink = 1/2 ;
xspace = 1 ;
branch[Line[{{x1_, y1_}, {x2_, y2_}}]] :=
  { Line[ { {x2, y2}, {x2 + xspace, y2 + yshrink * (y2 - y1)} } ],
    Line[ { {x2, y2},{x2 + xspace, y2 - yshrink * (y2 - y1)}  }  ] }
```

This is what happens to the first line:

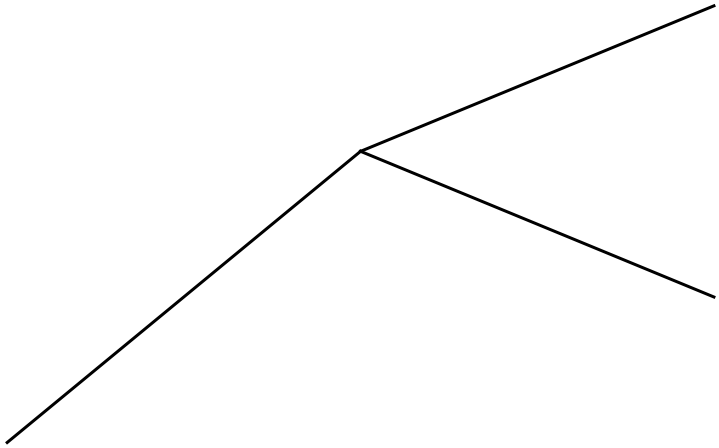
```
branch[firstline]
```

```
                   3                        1
{Line[{{0, 1}, {1, -}}], Line[{{0, 1}, {1, -}}]}
                   2                        2
```

In order to visualize successive branches they must be flattened:

```
Show[ Graphics[ Flatten[ { firstline,  branch[firstline]  } ]  ]]
```

FIGURE 1. First application of the function branch[] to the segment Line[{{-1, 0}, {0, 1}}].



```
-Graphics-
```

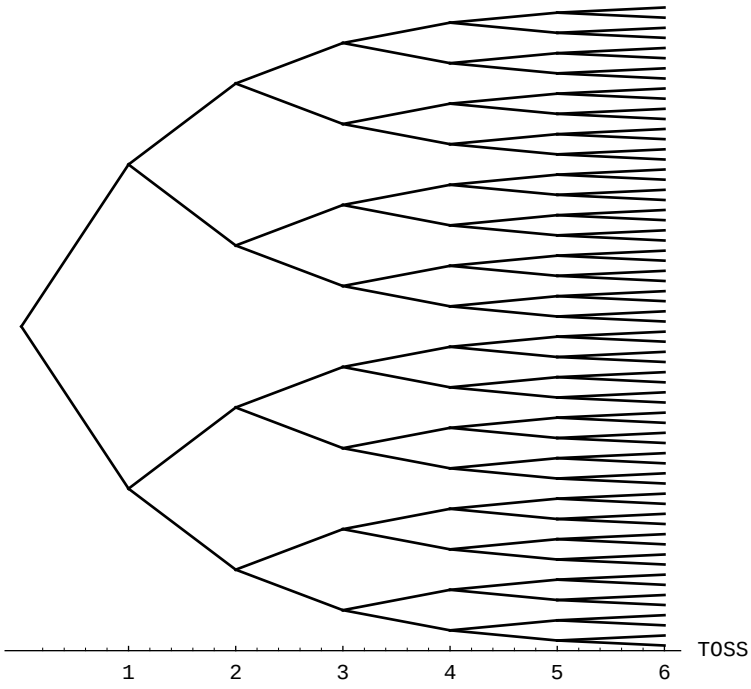
The following function carries this out automatically by mapping the branch operation on any lines given it and then flattening the result:

```
doBranch[lines_] := Flatten[ Map[ branch,  lines  ]]
```

This function is then nested recursively n = 6 times, the first (seed) line is dropped, and then visualized:

```
Show[ Graphics[  Drop[  NestList[ doBranch,  {firstline},  6   ],1  ] ],
 Axes -> {Automatic, None}, AxesLabel -> { "TOSS", None},
 AspectRatio -> 1]
```

FIGURE 2. Recursive application of the function branch[] to depth 6. (First segment is discarded).




```
-Graphics-
```

You can see that this tree results in 2^6 = 64 end nodes by toss = 6, as it should. Such a sideways tree is widely used in decision theory to illustrate such "games against nature" as choices made in response to prior choices. When the payoffs are shown at the end nodes it is possible to devise strategies of play.

A TABLE OF OUTCOMES

Each of the paths of the tree is a unique sequence of heads and tails, and we saw that there are 2^n such paths. In fact, each path can be viewed as the binary representation of the numbers i = 0...2^n - 1.

```
binrep[x_, n_] :=  Table[  Round[  Mod[x, 2^i]/(2^i - 1) ], {i, n, 1, -1} ]
```

This function can be used to show x to any number of binary places n:

```
binrep[10, 6]
```

```
{0, 0, 1, 0, 1, 0}
```

We can create a table of the binary expansion

```
bintable[n_] := Table[  binrep[i, n], {i, 0, 2^n - 1}] ;
bintable[3]
```

```
{{0, 0, 0}, {0, 0, 1}, {0, 1, 0}, {0, 1, 1},
 {1, 0, 0}, {1, 0, 1}, {1, 1, 0}, {1, 1, 1}}
```

Now find the sum of the lists, or in other words, the number of heads in each outcome:

```
Apply[Plus, %, {1}]
```

```
{0, 1, 1, 2, 1, 2, 2, 3}
```

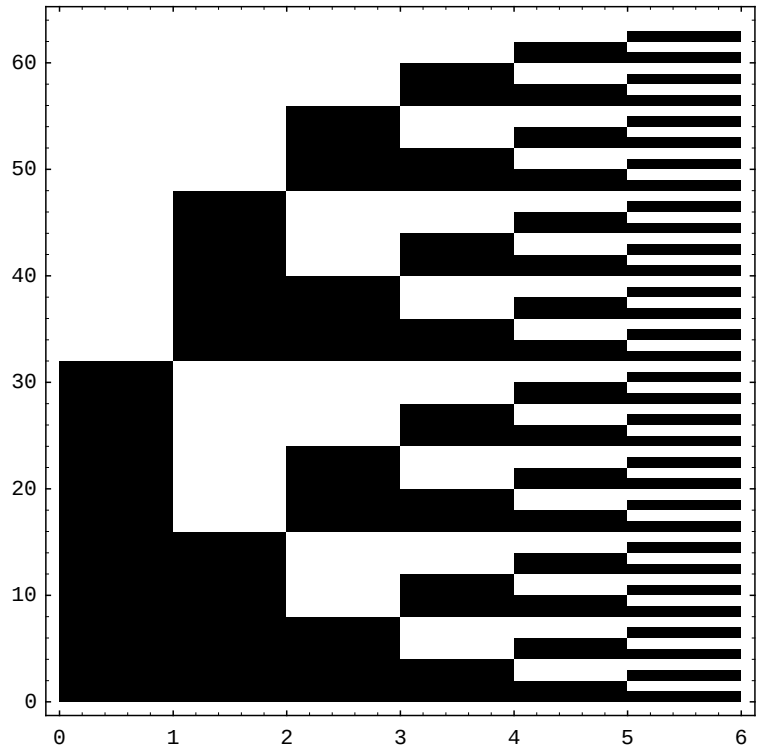
Let's compute the 6th binary expansion (or the representations up to 64:

```
n = 6 ;
binary6 = bintable[n] ;
```

An effective way to view this expansion is a ListDensityPlot, which is another visualization of the above tree.

```
ListDensityPlot[  binary6, Frame -> True,  Mesh -> False, AspectRatio -> 1]
```

FIGURE 3. Binary expansions of the numbers from 1 to 64 using the *Mathematica* function ListDensityPlot[].



Again, interpreting a 1 as head and a 0 as tail, here are the heads of each outcome:

```
numheads = Apply[Plus, binary6, {1}]
```

```
{0, 1, 1, 2, 1, 2, 2, 3, 1, 2, 2, 3, 2, 3, 3, 4, 1,

 2, 2, 3, 2, 3, 3, 4, 2, 3, 3, 4, 3, 4, 4, 5, 1, 2,

 2, 3, 2, 3, 3, 4, 2, 3, 3, 4, 3, 4, 4, 5, 2, 3, 3,

 4, 3, 4, 4, 5, 3, 4, 4, 5, 4, 5, 5, 6}
```

And here is a table showing the outcomes and the number of heads, separated by a space:

```
outcomes = Table[Append[ Append[ Table[
 If[  binary6[[i]][[j]] == 1, H, T ], {j, n} ], Space ],  numheads[[i]]  ], {i, 2^n}] ;
```

Which is displayed in TableForm with appropriate headings and alignment:

```
TableForm[  outcomes, TableDepth -> 2,  TableSpacing -> {0, 3},
 TableHeadings -> {  Automatic,   Append[ Append[ Range[n], Space ],
  "HEADS"  ] }, TableAlignments -> { Automatic,  Center }]
```

TABLE 1. An arrangement of possible outcomes for six tosses of a fair coin.

	1	2	3	4	5	6	HEADS
1	T	T	T	T	T	T	0
2	T	T	T	T	T	H	1
3	T	T	T	T	H	T	1
4	T	T	T	T	H	H	2
5	T	T	T	H	T	T	1
6	T	T	T	H	T	H	2
7	T	T	T	H	H	T	2
8	T	T	T	H	H	H	3
9	T	T	H	T	T	T	1
10	T	T	H	T	T	H	2
11	T	T	H	T	H	T	2
12	T	T	H	T	H	H	3
13	T	T	H	H	T	T	2
14	T	T	H	H	T	H	3
15	T	T	H	H	H	T	3
16	T	T	H	H	H	H	4
17	T	H	T	T	T	T	1
18	T	H	T	T	T	H	2
19	T	H	T	T	H	T	2
20	T	H	T	T	H	H	3
21	T	H	T	H	T	T	2
22	T	H	T	H	T	H	3
23	T	H	T	H	H	T	3
24	T	H	T	H	H	H	4

25	T	H	H	T	T	T	2
26	T	H	H	T	T	H	3
27	T	H	H	T	H	T	3
28	T	H	H	T	H	H	4
29	T	H	H	H	T	T	3
30	T	H	H	H	T	H	4
31	T	H	H	H	H	T	4
32	T	H	H	H	H	H	5
33	H	T	T	T	T	T	1
34	H	T	T	T	T	H	2
35	H	T	T	T	H	T	2
36	H	T	T	T	H	H	3
37	H	T	T	H	T	T	2
38	H	T	T	H	T	H	3
39	H	T	T	H	H	T	3
40	H	T	T	H	H	H	4
41	H	T	H	T	T	T	2
42	H	T	H	T	T	H	3
43	H	T	H	T	H	T	3
44	H	T	H	T	H	H	4
45	H	T	H	H	T	T	3
46	H	T	H	H	T	H	4
47	H	T	H	H	H	T	4
48	H	T	H	H	H	H	5
49	H	H	T	T	T	T	2
50	H	H	T	T	T	H	3
51	H	H	T	T	H	T	3
52	H	H	T	T	H	H	4
53	H	H	T	H	T	T	3
54	H	H	T	H	T	H	4
55	H	H	T	H	H	T	4
56	H	H	T	H	H	H	5
57	H	H	H	T	T	T	3
58	H	H	H	T	T	H	4
59	H	H	H	T	H	T	4
60	H	H	H	T	H	H	5
61	H	H	H	H	T	T	4
62	H	H	H	H	T	H	5
63	H	H	H	H	H	T	5
64	H	H	H	H	H	H	6

This table clearly shows that every outcome is equally likely, and indeed each has probability

```
2^-n

N[%]
```

```
1
--
64

0.015625
```

although the first and last outcomes are unusual looking.

SELF-SIMILARITY

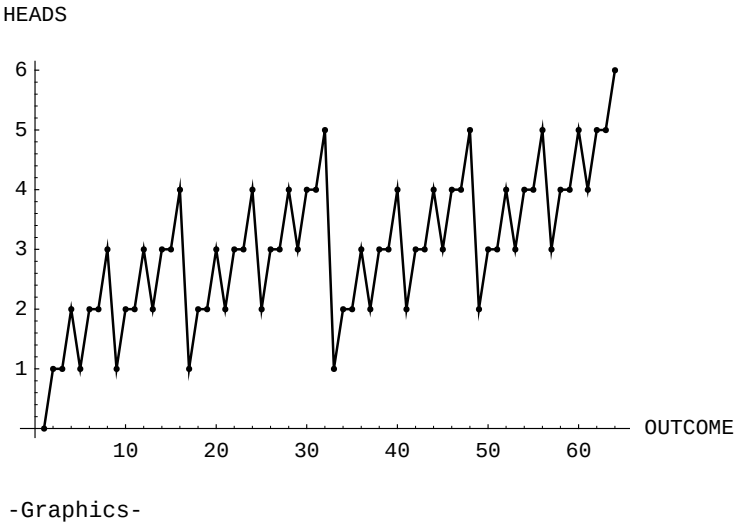
You may notice that the last column of the table, representing the number of heads, has a curious rhythm of increases and decreases. This pattern is clearer when the values are plotted:

```
plot1 = ListPlot[ numheads, AxesLabel -> { "OUTCOME", "HEADS"},
 PlotStyle -> {PointSize[.01]}, DisplayFunction -> Identity] ;

plot2 = ListPlot[ numheads, PlotJoined -> True, DisplayFunction -> Identity] ;

Show[ {  plot1,   plot2  }, DisplayFunction -> $DisplayFunction]
```

FIGURE 4. Plot of the number of heads in each outcome shown in TABLE 1.



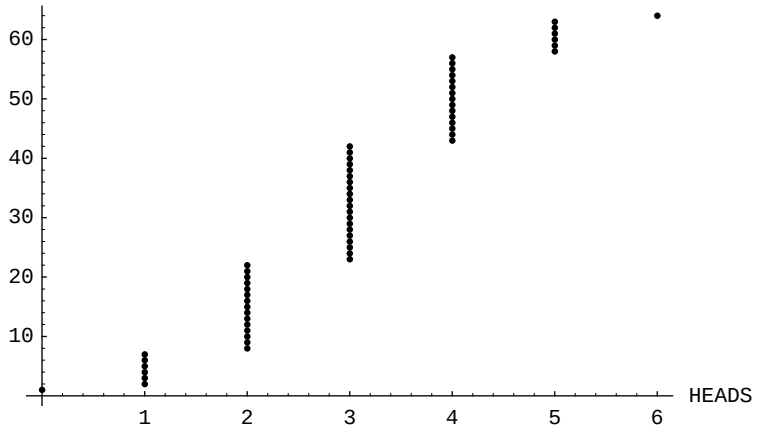
You may also have noted above that the tree of outcomes was exactly self-similar in that each branch was a scaled-down version of the one out of which it grew, so that the tree was a visualization of a perfectly fractal process. The head count pattern however is only roughly self-similar; pieces of it appear to be scaled-down versions of the whole. Viewed as a curve (when the points are joined), the above plot certainly has fractal dimension $D > 1$ (compare it to a line between (1, 0) and (64, 6)). But viewed as a "dust," the points have $D < 1$ in that the points between 0 and n become denser as n increases. For a more detailed discussion of these matters see Lam and De Cola [1993], as well, of course, as Mandelbrot [1988].

THE BINOMIAL NUMBERS

Another way of looking at the number of heads is to sort their values and examine their cumulative frequencies. This shows the concentration around n/2.

```
ListPlot[ Transpose[ { Sort[numheads],  Range[Length[numheads]]  }  ],
 AxesLabel -> {"HEADS", None},  PlotStyle -> PointSize[.01]]
```

FIGURE 5. Cumulative frequencies for the number of heads in the 64 possible outcomes of 6 coin tosses.



-Graphics-

The number of heads for each outcome has a frequency distribution that can be described by a binned count of the number of heads that appears in each outcome. We need a few packages.

```
Needs["Statistics`DescriptiveStatistics`"] ; (* Mean, etc. *)
Needs["Statistics`DataManipulation`"] ; (* BinCounts *)

Needs["Graphics`Graphics`"] ; (* BarChart *)

Needs["Graphics`FilledPlot`"] ; (* Filled Plot  *)
```

To count the number of heads we first need their range. The result is no surprise: in 6 tosses of a coin you can't get fewer than 0 or more than 6 heads.

```
{ Min[numheads],  Max[numheads]}
```

```
{0, 6}
```

```
BinCounts[ numheads,  % - {1, 0}]
```

```
{1, 6, 15, 20, 15, 6, 1}
```

But these are just the binomial coefficients

```
% == Table[Binomial[6, i], {i, 0, 6}]
```

```
True
```

So in order to find the distribution of heads in a cointoss, we can use the table of binomial coefficients. Here, for example, is a list of the head counts and their associated coefficients for 20 tosses. The order of the rows is reversed so that if you tilt the table 90 degrees clockwise you will see not only the counts but also (in the lengths of the numbers) the log to the base 10 of the counts.

```
n = 20 ;

toss20 = Table[  { Binomial[n, i],  i }, {i, 0, n}];

TableForm[ Reverse[%],TableSpacing -> {0, 3},TableAlignments -> {Top, Right}]
```

- **! JOE, I'M NOT SURE WHAT TO DO ABOUT THE ALIGNMENT OF THIS TABLE. MAYBE JUST RIGHT ALIGNMENT, UNTIL I SHOW PROOFS TO AUTHOR. DAVID**

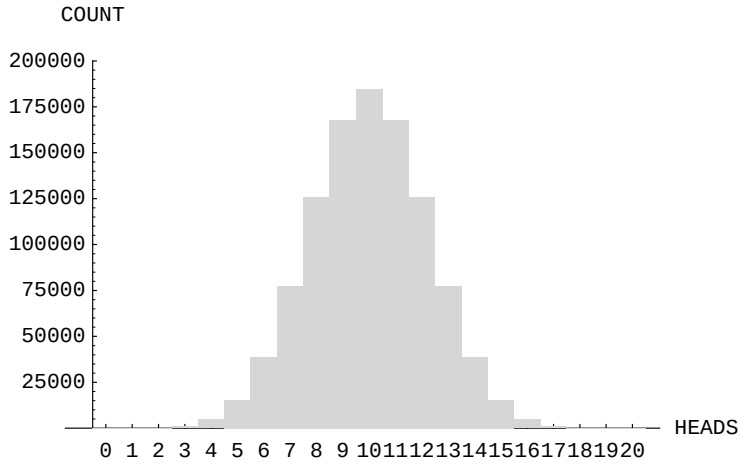
TABLE 2. Head counts and associated coefficients for 20 tosses of a fair coin.

1	20
20	19
190	18
1140	17
4845	16
15504	15
38760	14
77520	13
125970	12
167960	11
184756	10
167960	9
125970	8
77520	7
38760	6
15504	5
4845	4
1140	3
190	2
20	1
1	0

These coefficients can be visualized using a histogram, which can be constructed using the BarChart function with bars spaced close together.

```
histogram = BarChart[  toss20, BarGroupSpacing -> 0,
AxesLabel -> { "HEADS", "COUNT" },
 BarStyle -> {GrayLevel[.84]},
  PlotRange -> {Automatic, {0, 200000}}, BarEdges -> False]
```

FIGURE 6. Histogram corresponding to values in TABLE 2.




```
-Graphics-
```

BACK TO MAPS

Let's return to our original problem of sampling locations on a map. Recall that the National Map Accuracy Standard would allow that no more than 10% (or 10 of our 100 points) could be more than 12.2 meters in error. In order to visualize this, we ask what cointoss events correspond to this situation?

In the binomial/cointoss model we are interested in the two ends or "tails" of the above bar chart, and since we are already dealing with 20 tosses, we ask which events correspond to 10% of the "tails" of the bar chart above. (Perhaps one of the reasons students find statistics so confusing is that the language can sometimes be ambiguous--we call the ends of the distribution as well as the "bottom" of the coin both tails!) Here is a function that returns from the binomial coefficients the tail events:

```
Take[toss20, {10, 12}]
```

```
{{167960, 9}, {184756, 10}, {167960, 11}}
```

Note that these events (or sets of outcomes) constitute about 50% of the distribution, so a way of thinking about a "fifty-fifty" chance is 9, 10, or 11 heads in 20 tosses of a coin:

```
Plus @@ Transpose[%][[1]] / 2^n // N
```

```
0.496555
```

Similarly, 7 to 13 heads constitutes about 90% of the distribution:

```
prob7to13 = N[ Plus @@  Transpose[  Take[ toss20,  {8, 14} ]  ][[1]] / 2^n]
```

```
0.884682
```

So if you shade in the part of the above histogram corresponding both to 1 to 6 and to 13 to 20 heads, then these tails (of the distribution, not the coins!) correspond to the 10% likelihood of your locations on the map being more than 12 meters off.

THE GAUSSIAN DISTRIBUTION

There is an obvious similarity between the above histogram and the gaussian distribution. To generalize the binomial we need its mean and standard deviation. We could look up the formulas, use the mean and standard deviation functions, or, as here, work them out:

```
mu = Sum[  i * Binomial[n, i], {i, 0, n}]/2^n
```

```
10
```

```
sigma = Sqrt[  Sum[ (i - mu)^2 * Binomial[n, i],  {i,0,n} ]/2^n]//N
```

```
2.23607
```

The first result is not surprising; if you toss a coin n times you "expect" got get n/2 heads--this is a simple example of the powerful idea of symmetry. The mean and standard deviation terms can be used to parameterize the normal or gaussian function:

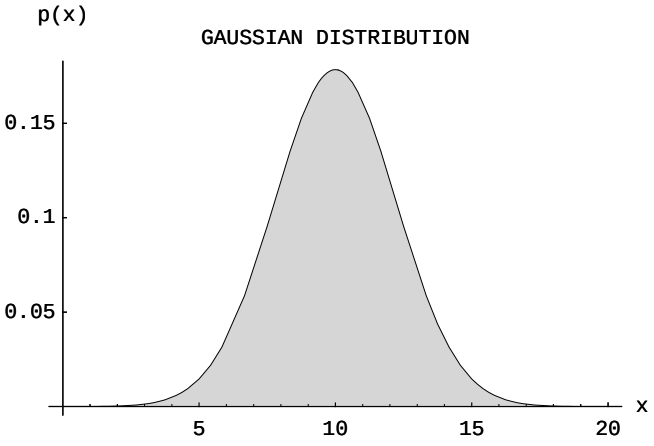
```
gauss[x_] :=
  (1/(Sqrt[2 Pi] sigma)) *
  Exp[-(1/2)((x-mu)/sigma)^2]
```

```
General::spell1:
   Possible spelling error: new symbol name "gauss"
     is similar to existing symbol "Gauss".
```

When this function is visualized over the range of possible heads we get the bell curve whose integration predicts the probability of getting that many heads.

```
FilledPlot[ gauss[x], {x, 0, 20},PlotStyle -> AbsoluteThickness[0],
  Fills -> {GrayLevel[.84]}, AxesLabel -> {"x", "p(x)"},
 Ticks -> {Automatic, Range[0, .2, .05]}, PlotLabel -> "GAUSSIAN DISTRIBUTION"]
```

FIGURE 7. Plot of the Gaussian distribution.



-Graphics-

For example, we can use this continuous function to estimate the probabilility of tossing from 7 to 13 heads. We do this by integrating from the midpoint between 6 and 7 heads to the midpoint between 13 and 14 heads, getting a result not very different from the exact number found above.

```
Integrate[gauss[x], {x, 6.5, 13.5}]//N
% - prob7to13

General::intinit: Loading integration packages.
0.882475
-0.00220657
```

In a sense we have come full circle, from an examination of a single coin toss, through the complex but self-similar behavior of the binomial process, to the continuous distribution whose picture generalizes what happens when a simple process is infinitely repeated to reveal a quite elegant picture.

A final visualization of this idea is that of W. J. Youden, recalled by Tufte [1983, p. 143]:

THE
NORMAL
LAW OF ERROR
STANDS OUT IN THE
EXPERIENCE OF MANKIND
AS ONE OF THE BROADEST
GENERALIZATIONS OF NATURAL
PHILOSOPHY ◊ IT SERVES AS THE
GUIDING INSTRUMENT IN RESEARCHES
IN THE PHYSICAL AND SOCIAL SCIENCES AND
IN MEDICINE, AGRICULTURE, AND ENGINEERING ◊
IT IS AN INDISPENSABLE TOOL FOR THE ANALYSIS AND THE
INTERPRETATION OF THE BASIC DATA OBTAINED BY OBSERVATION AND EXPERIMENT

- **!Joe, in Tufte's book, somehow line 5 "EXPERIENCE OF MANKIND"fits better into the bellcurve. David. Not sure how they did it.**

ACKNOWLEDGMENT

I thank Elizabeth Bradford for her generous assistance.

REFERENCES

Gray, Theodore W. and Jerry Glynn. *Exploring Mathematics with Mathematica*, Addison-Wesley, Redwood City CA (1991).

Lam, Nina and Lee De Cola. *Fractals in Geography*, Prentice-Hall, Englewood-Cliffs NJ: (1993).

Mandelbrot, Benoit B. *The Fractal Geometry of Nature*, W.H. Freeman, San Francisco (1983).

Thompson, Morris M. *Maps for America*. USGS, Washington DC (1988).

Tufte, Edward R. *The Visual Display of Quantitative Information*, The Graphics Press, Cheshire CT (1983).

ABOUT THE AUTHOR

Lee De Cola is a Research Physical Scientist with degrees in mathematics, regional planning, and geography. At the USGS he is currently using the scaling insights of fractal geography to map regional changes in the Baltimore-Washington megalopolis. He is also participating in the development of a participatory planning process for the sustainable development of the Greater Yellowstone Area. He has taught at several universities in the United States (currently at George Mason University) as well as the University of Ibadan, Nigeria. His hobbies are bicycling, the clarinet, and reading the *New York Times*.

Lee De Cola

U.S. Geological Survey

521 National Center

Reston VA 20192

ldecola@usgs.gov