ISTA 116: Lab Assignment #6 (50 pts)

SOLUTION

Due Monday, November 14th, 11:59 P.M.

Problem 1:

(32 pts)

A robot steps back and forth randomly along a line, but never ventures more than two steps away from the center point. After ten minutes of wandering, its position measured in number of steps from the center is given by the random variable X. Negative numbers indicate that the robot is to the left of the center point, and positive numbers indicate that it is to the right.

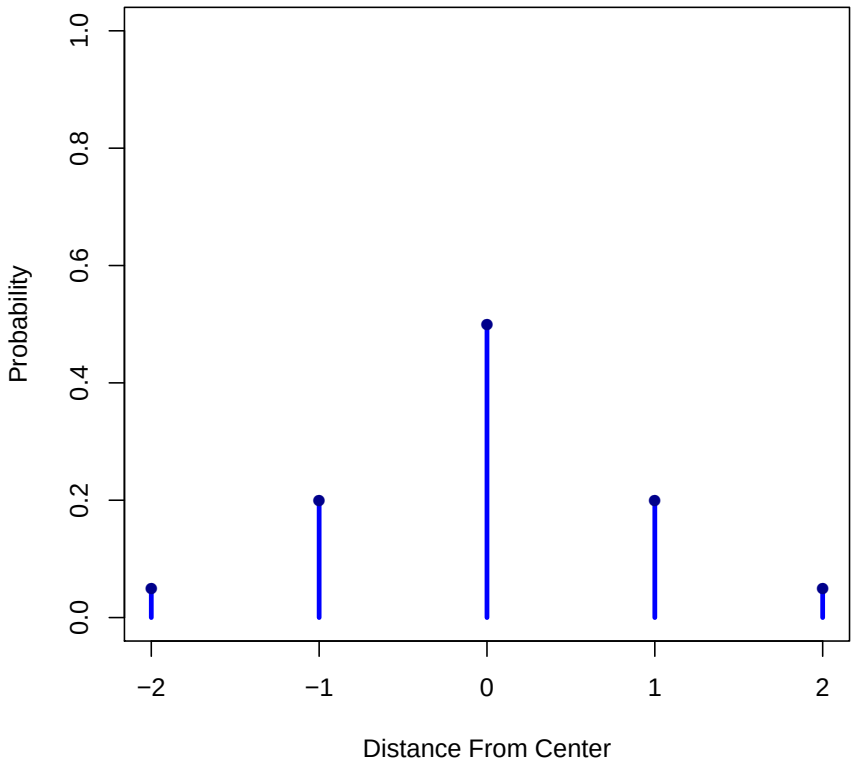
The distribution of X is characterized by the following probability mass function:

x	-2	-1	0	1	2
$P(X = x)$	0.05	0.2	0.5	0.2	0.05

a. (4 pts) Produce a "spike plot" based on this distribution.

We're looking for a spike for each column in the table with height equal to the value in the second row, so:

```
> values = -2:2
> probabilities = c(0.05, 0.2, 0.5, 0.2, 0.05)
> plot(values, probabilities, type = "h", ylim = c(0, 1), col = "blue",
+     lwd = 3, xlab = "Distance From Center", ylab = "Probability")
> points(values, probabilities, pch = 16, col = "dark blue")
```



b. (4 pts) Create a table showing the CDF for the distribution.

You can do this in R with functions we've already seen:

```
> (cdf = c(sum(probabilities[1]), sum(probabilities[1:2]), sum(probabilities[1:3]),
+     sum(probabilities[1:4]), sum(probabilities[1:5])))
```

```
[1] 0.05 0.25 0.75 0.95 1.00
```

There's an R function that does the same thing (*cumsum*):

```
> (cdf = cumsum(probabilities))
```

```
[1] 0.05 0.25 0.75 0.95 1.00
```

Or, for a small problem like this one it's easy to do by hand.

Here's the table we were asked for:

x	-2	-1	0	1	2
$P(values <= x)$	0.05	0.25	0.75	0.95	1

c. (6 pts) Compute the mean and standard deviation of this distribution.

Remember, the mean is the sum of all of the possible outcomes, each weighted by its probability. For the variance, you take each value's difference from the mean and square that, then weight the squared difference by it's probability, and sum them all up. Finally, don't forget to take the square root since we were asked for the *sd*, not the variance.

```
> (mn = sum(values * probabilities))

[1] 0

> (sd = sqrt(sum((values - mn)^2 * probabilities)))

[1] 0.8944272
```

d. (6 pts) Produce 3 random samples (*with replacement*) from this distribution: one of size 20, one of size 200, and one of size 2000. Use `sample()` in `R`, and store the result of each sample as a variable.

Remember, when using the *sample* function, you must pass in a vector of probabilities if you do not want uniform (i.e. each outcome has an equal chance of being selected) sampling!

```
> (sample20 = sample(values, 20, replace = TRUE, prob = probabilities))

 [1]  1  0 -1  1 -1  0 -1  0  1  0  0  0  1  0  0  0 -2  0  0  1

> (sample200 = sample(values, 200, replace = TRUE, prob = probabilities))

  [1]  0  0 -1  2  0  1  2  0  0  0  1  1 -1  0  1  0  0  0  1  0 -1  0 -1  0  0
 [26] -1 -1  2  0  0 -2  0 -2  0  0  0 -2  0  1  2  0  0 -1 -1  0  0 -1  2  0  1
 [51]  1  0 -2  0  0 -1 -1 -1 -1  0 -1  0 -1  0  0  0 -1  0  1  0  1  0  0 -1  1
 [76] -1  0 -1  0  0 -1  1  1  0  1 -1  1  0 -1  0  0 -1  2  0  0  0  2  0  0  0
[101] -1 -1  0 -1  0  0 -1 -1 -1 -1  1  0 -1  0  2  0 -2 -1  0 -1  0  0 -1 -1  0
[126]  0  1  1  2  0  1  0  0  0  1  1  0  1 -1  1  0  0  0  0  0  0 -1  0  1 -2
[151]  1  2  0  0 -1  1  1  0 -2  1  0 -1  0 -1  0  0  1 -1  0  0  1  1  1  2  0
[176]  0  0  2  0  1 -1  0  1 -1 -1  0  0 -1  1  0 -1 -1  0 -1  0  1  1  1  0  2

> sample2000 = sample(values, 2000, replace = TRUE, prob = probabilities)
```

e. (6 pts) For each sample, produce a table and a spike plot showing the proportion of draws at each value (you'll need to use `table()` followed by `prop.table()` for the table, followed by `plot()` for the plot). If you should end up with a sample where not

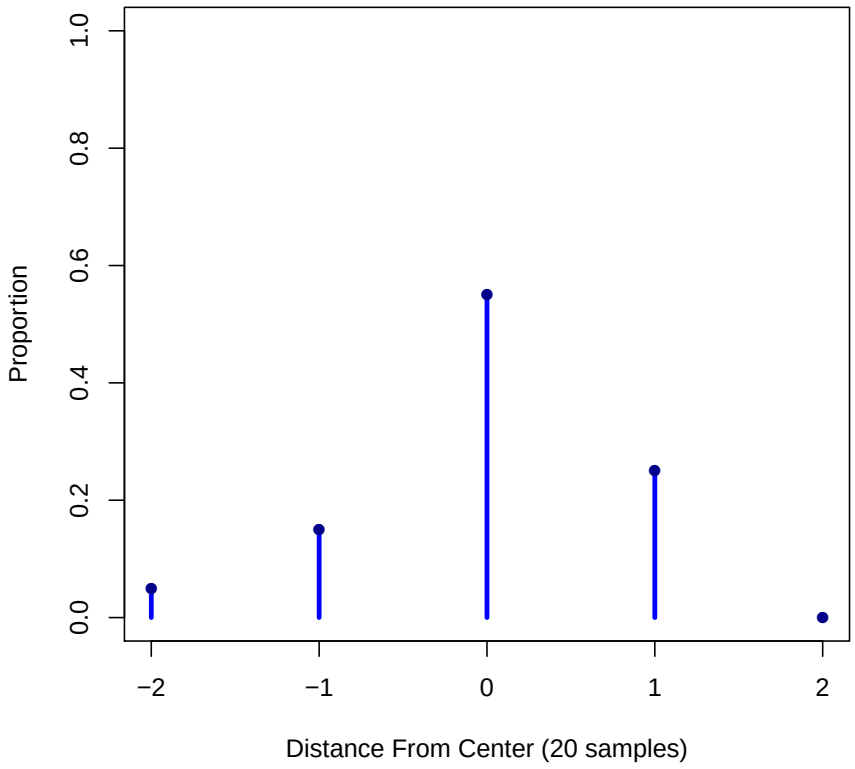
every possible value shows up, you may need to use the following trick after creating your sample and before creating your table to get 0s to appear in the table, rather than leaving out cells entirely:

```
mySample <- sample(some code here)
mySample <- factor(mySample, levels = x)
```

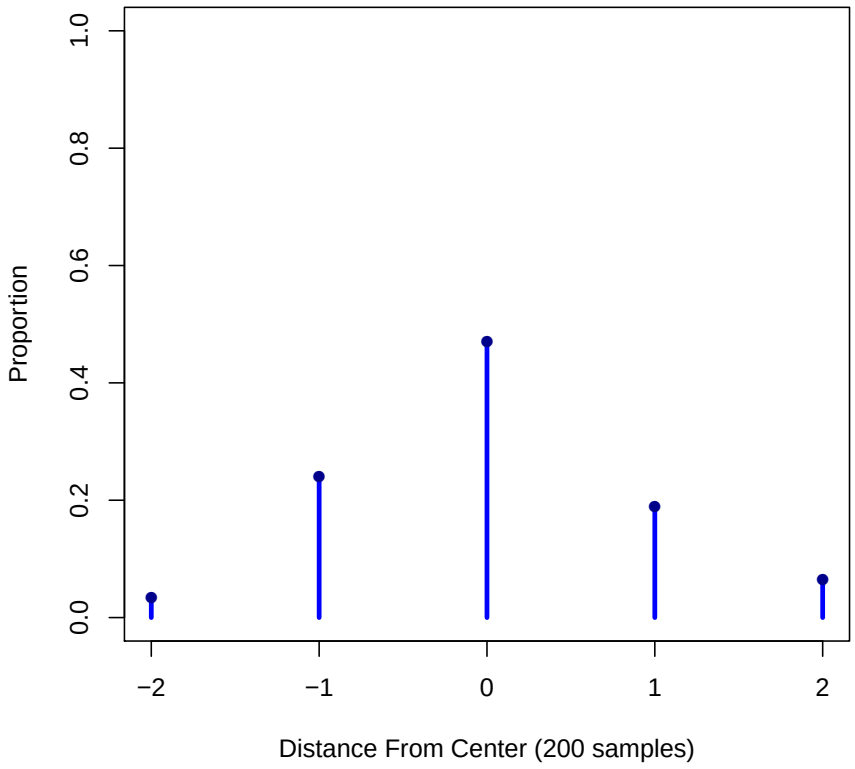
where `x` is the vector of the possible values.

Show both the tables and plots, and clearly label which corresponds to which sample size.

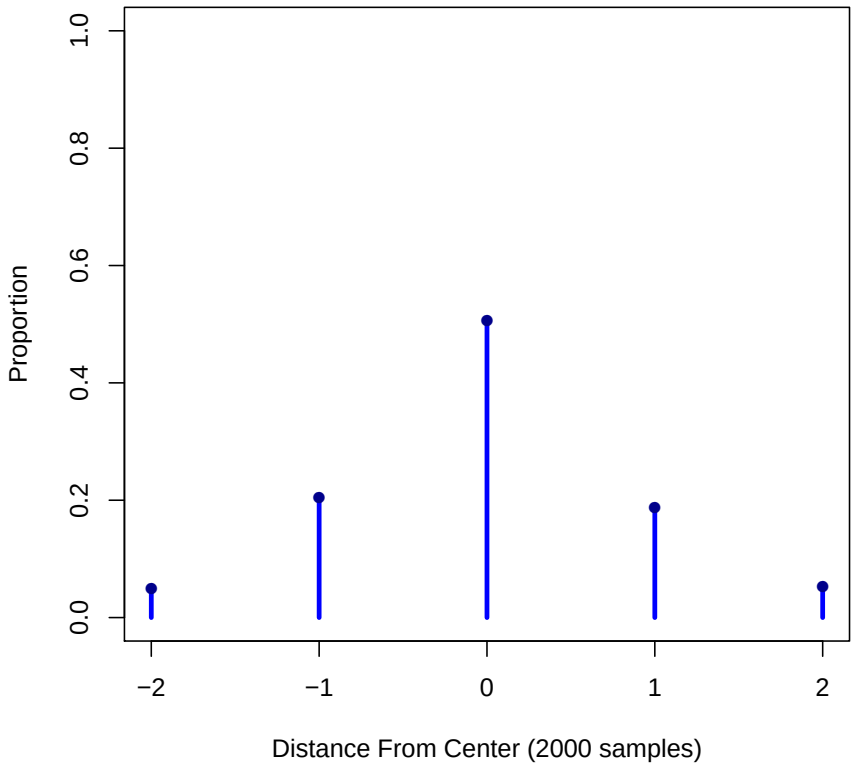
```
> sample20 = factor(sample20, levels = values)

> (tab20 = prop.table(table(sample20)))

sample20
  -2   -1    0    1    2
0.05 0.15 0.55 0.25 0.00

> plot(values, tab20[], type = "h", ylim = c(0, 1), col = "blue",
+     lwd = 3, xlab = "Distance From Center (20 samples)", ylab = "Proportion")
> points(values, tab20, pch = 16, col = "dark blue")
```




```
> (tab200 = prop.table(table(sample200)))

sample200
   -2    -1     0     1     2
0.035 0.240 0.470 0.190 0.065

> plot(values, tab200[], type = "h", ylim = c(0, 1), col = "blue",
+     lwd = 3, xlab = "Distance From Center (200 samples)", ylab = "Proportion")
> points(values, tab200, pch = 16, col = "dark blue")
```




```
> (tab2000 = prop.table(table(sample2000)))

sample2000
    -2     -1      0      1      2
0.0500 0.2040 0.5055 0.1880 0.0525

> plot(values, tab2000[], type = "h", ylim = c(0, 1), col = "blue",
+     lwd = 3, xlab = "Distance From Center (2000 samples)", ylab = "Proportion")
> points(values, tab2000, pch = 16, col = "dark blue")
```



f. (4 pts) Compare the sample spike plots in (e) to the distribution spike plot in (a). Comment on the similarities and differences.

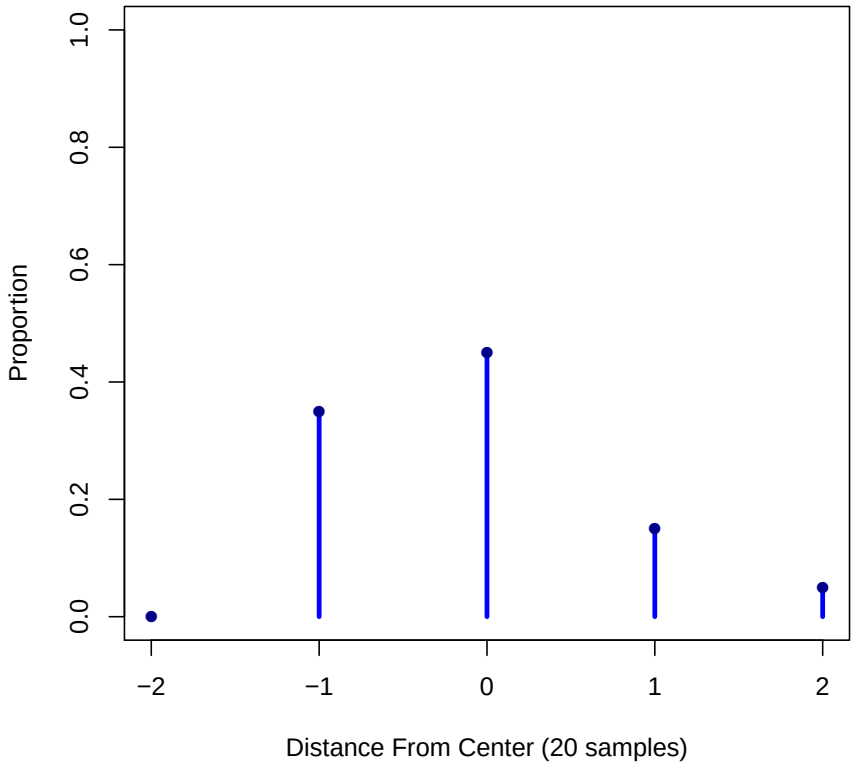
At smaller sample sizes the spike plots differ more from the PMF spike plot. As the sample size is increased, the distribution of the samples becomes closer to the probabilities in the probability mass function (and the plots become more similar).

g. (4 pts) Repeat parts (d) and (e), generating new samples. Which sample has changed the most?

```
> sample20 = factor(sample(values, 20, replace = TRUE, prob = probabilities),
+     levels = values)
> sample200 = sample(values, 200, replace = TRUE, prob = probabilities)
> sample2000 = sample(values, 2000, replace = TRUE, prob = probabilities)

> (tab20 = prop.table(table(sample20)))
```

```
sample20
  -2   -1    0    1    2
0.00 0.35 0.45 0.15 0.05
```

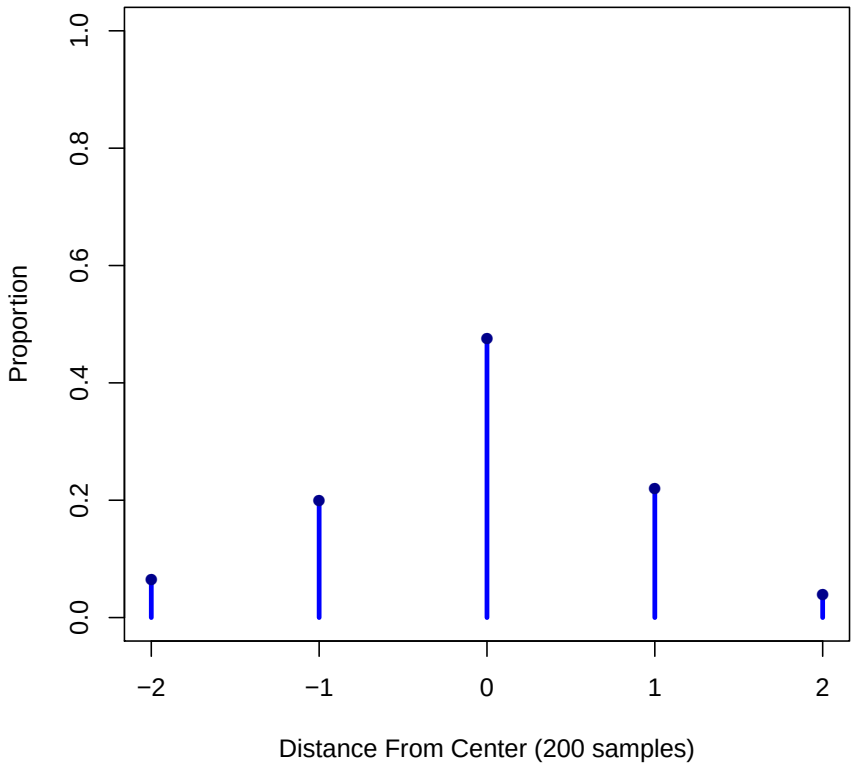
```
> plot(values, tab20[], type = "h", ylim = c(0, 1), col = "blue",
+     lwd = 3, xlab = "Distance From Center (20 samples)", ylab = "Proportion")
> points(values, tab20, pch = 16, col = "dark blue")
```



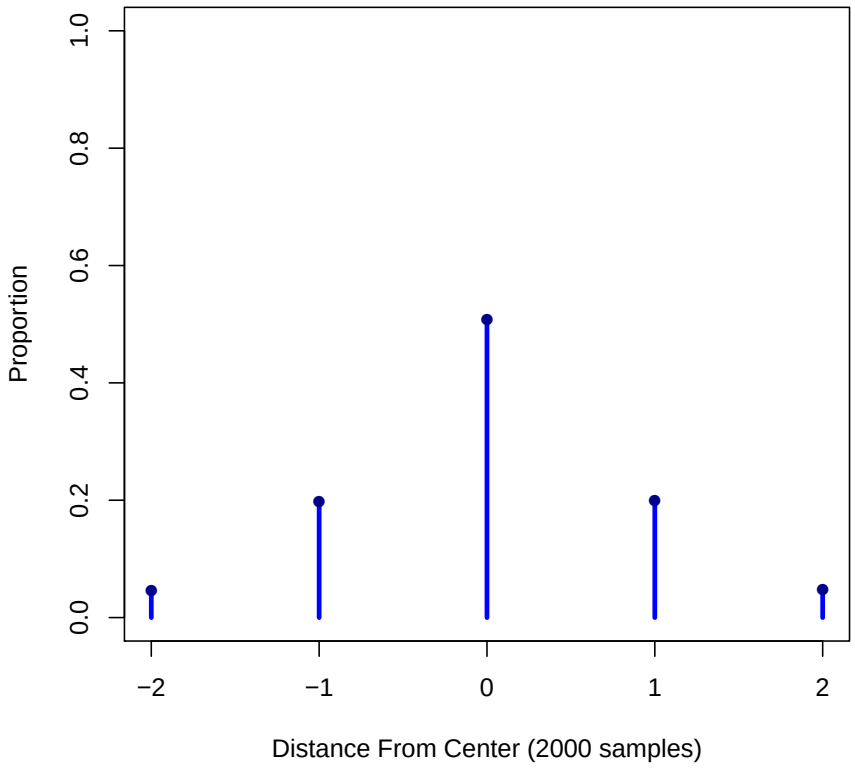

```
> (tab200 = prop.table(table(sample200)))
```

```
sample200
   -2    -1     0     1     2
0.065 0.200 0.475 0.220 0.040
```

```
> plot(values, tab200[], type = "h", ylim = c(0, 1), col = "blue",
+     lwd = 3, xlab = "Distance From Center (200 samples)", ylab = "Proportion")
> points(values, tab200, pch = 16, col = "dark blue")
```




```
> (tab2000 = prop.table(table(sample2000)))

sample2000
    -2     -1      0      1      2
0.0460 0.1985 0.5075 0.2000 0.0480

> plot(values, tab2000[], type = "h", ylim = c(0, 1), col = "blue",
+     lwd = 3, xlab = "Distance From Center (2000 samples)", ylab = "Proportion")
> points(values, tab2000, pch = 16, col = "dark blue")
```



The sample of size 20 has changed the most.

Problem 2:

(18 pts)

The following can all be answered either with simple counting or with one or two line computations in `R`. (Hint: for each one, consider a random variable that captures the quantity of interest, and think about whether it is characterized by a named distribution)

a. (3 pts) What's the probability of requiring all 7 flips to decide the winner in a "best of 7" coin toss game, assuming the coin is fair?

This is the same as the probability of getting exactly 3 Heads (or Tails) in 6 coin tosses. That question is a perfect match for *dbinom* which needs to know the number

of successes we're interested in seeing, the number of trials, and the probability of success on a single trial:

```
> dbinom(3, size = 6, 1/2)

[1] 0.3125
```

b. (3 pts) Which is more likely: rolling two dice 24 times and getting at least one double-sixes, or rolling one die four times and getting at least one six?

The probability of getting a least one double-sixes on 24 rolls of 2 dice means we're interested in seeing either 1 double-sixes, or 2 double-sixes, or 3, or ... We can use *dbinom* on the vector $1:24$ to get each one of those probabilities and then sum them:

```
> sum(dbinom(1:24, size = 24, 1/36))

[1] 0.4914039
```

Another way to look at the question is that the *only* thing we do *not* want to see is *no* double-sixes. So we could also use *pbinom* to find the probability of 0 (or fewer) successes and subtract that probability from 1:

```
> 1 - pbinom(0, size = 24, 1/36)

[1] 0.4914039
```

The probability of getting at least one six from 4 die rolls is answered similarly. We can either calculate individual probabilities for the outcomes we are interested in seeing and sum them (since we are looking at a *discrete* distribution):

```
> sum(dbinom(1:4, size = 4, 1/6))

[1] 0.5177469
```

As before, we can rephrase the question. Instead of asking for the probability of at least one six (i.e. number of sixes ≥ 1), we can ask for the probability of no sixes at all (i.e. number of sixes < 1) and then subtract from 1:

```
> 1 - pbinom(0, size = 4, 1/6)

[1] 0.5177469
```

So, rolling one die four times and getting at least one six is more likely.

c. (3 pts) A multiple choice test has 20 questions, each with 3 answers, exactly one of which is correct. What's the probability of scoring 65% or better by completely random guessing?

A 65% on a 20 question test means we need to get 0.65 * 20 questions right.

```
> (min = 0.65 * 20)
```

```
[1] 13
```

We can use *pbinom* to find the probability of getting any number smaller than that correct, and then subtract the result from 1.

```
> 1 - pbinom(min - 1, size = 20, 1/3)
```

```
[1] 0.003724569
```

d. (3 pts) A town has 1000 births in a particular year. If the probability of a boy for any particular birth is 51.3%, what is the probability that between 49.3% and 53.3% (inclusive) of the town's births are male, assuming all births are independent?

First we'll rephrase the question. What's the probability that between 493 and 533 of the town's births are male?

Using *dbinom* we'll need to sum the probability of each each outcome from 493 up to 533:

```
> sum(dbinom(493:533, size = 1000, 0.513))
```

```
[1] 0.8053847
```

As in the previous question, we can use *pbinom* to find the probability of an outcome falling within a given range $[bottom, top]$ by finding the probability of (outcome $\leq top$) and subtracting the probability of (outcome $\leq bottom$).

```
> pbinom(533, size = 1000, 0.513) - pbinom(493, size = 1000, 0.513)
```

```
[1] 0.7940515
```

But wait, the answers are *different*! The problem is that *pbinom* gives the probability of an outcome that is $\leq$ the given value. So, while our probability for the top value in the range is okay, our probability for the bottom value is off by 1. We're subtracting the probability of an outcome ≤ 493 but we wanted to accept 493 as a success. We needed to do this:

```
> pbinom(533, size = 1000, 0.513) - pbinom(492, size = 1000, 0.513)
```

```
[1] 0.8053847
```

e. (3 pts) A city has 10000 births in a particular year. If the probability of a boy for any particular birth is 51.3%, what is the probability that between 49.3% and 53.3% (inclusive) of the city's births are male, assuming all births are independent?

Let's use *pbinom*, but we won't repeat our last mistake:

```
> pbinom(5330, size = 10000, 0.513) - pbinom(4929, size = 10000,
+     0.513)

[1] 0.9999398
```

f. (3 pts) Suppose you roll a ten-sided die, numbered 0 through 9 to select digits for a random number. The first roll is the ones digit, and subsequent rolls generate successive decimal places. What's the probability that after 10 rolls, you have an integer? After 100 rolls? 1000?

As always, it's helpful to rephrase the question in terms we're already familiar with. It seems like the question is asking about the probability of some number of successes out of 10 rolls. What's a success? For the number to remain an integer we need to continue to roll 0's. Okay, so a success is rolling a 0 on a 10-sided die... but what about that first roll? On the first roll any number is an integer, right? Success guaranteed. So we don't care about that first roll. So we're really talking about 9 rolls. To use *dbinom* or *pbinom* we need to know the number of successes, the total number of trials, and the probability of success on an individual trial. We want 9 successes out of 9 trials where an individual success has probability of $\frac{1}{10}$. Here we want *exactly* 9 successes, so *dbinom* is the right function:

```
> dbinom(9, size = 9, 1/10)

[1] 1e-09

> dbinom(99, size = 99, 1/10)

[1] 1e-99

> dbinom(999, size = 999, 1/10)

[1] 0
```

(i.e. 1e-999)