Caltech Department of Mathematics

Ma 3/103
Introduction to Probability and Statistics

KC Border
Winter 2017

Assignment 4

Due Tuesday, January 31 by 4:00 p.m. at 253 Sloan

Instructions:

When asked for a probability or an expectation, give both a formula and an explanation for why you used that formula, and also give a numerical value when available.

When asked to plot something, use informative labels (even if handwritten), so the TA knows what you are plotting, attach a copy of the plot, and, if appropriate, the commands that produced it.

No collaboration is allowed on optional exercises.

Exercise 1 The coin-tossing data can be used to illustrate the normal approximation to the binomial. By chopping the data into strings of length 64, each string represents 64 independent Bernoulli trials. The number of 1s in each string is then a Binomial$(64, p)$ random variable. Combining this year's and the previous years' data there are enough to generate 2124 such Binomial$(64, p)$ random variables. I have done this for you and put the results in the file at `http://www.math.caltech.edu/~2016-17/2term/ma003/Data/Binomial64.txt`, but feel free to do it yourself. (The raw results are at `http://www.math.caltech.edu/~2016-17/2term/ma003/Data/FlipsCombined.txt`.)

1. (30 pts) Plot the histogram against the Binomial$(64, p)$ probability mass function and plot the empirical cdf against the Binomial cdf. Note that both R and Mathematica have at least three ways to calculate the bins for use in a histogram, the Sturges method, Freedman–Diaconis or FD method, and the Scott method. Provide a histogram for each of the three methods. Use $p = 0.5$. Say something intelligent about the plots to show that you have looked at them.

Hint for R: Here is some undocumented sample R code for the empirical cdf. (It has been pasted from the console of RStudio, a nice free package for the major OSes, from RStudio.com.) R has a built-in function, `ecdf`, for empirical distributions. The `pbinom` and `dbinom` functions give the cdf and probability mass function for the binomial distribution. So here is how to plot the empirical distribution.

```
setwd("YOUR PATHNAME GOES HERE")
#
#  CDFs
raw = as.matrix(read.table("Binomial64.txt"))
n = 64
p = .5
plot(0:64,pbinom(0:64,n,p), col="blue", type="l")
plot.ecdf(raw,add=T)
#
# Histograms
plot(dbinom(0:64,n,p), col="blue", type="l", main="Binomial64 Data")
hist(raw, breaks = "FD", freq=FALSE , add=TRUE)
# or
plot(dbinom(0:64,n,p), col="blue", type="l", main="Binomial64 Data")
hist(raw, breaks = "Sturges", freq=FALSE , add=TRUE)
# or
plot(dbinom(0:64,n,p), col="blue", type="l", main="Binomial64 Data")
hist(raw, breaks = "scott", freq=FALSE , add=TRUE)
```

The `type="l"` draws a smooth curve through the data; or you could have used `type="p"`. If you are using a front end for R, such as RStudio, you can export the graphics via the menus.

Here is some Mathematica code that does something similar:

```
SetDirectory["/Your directory path goes here"]
a = Flatten[Import["Binomial64.txt", "Table"]];
g1 = Histogram[a, "FreedmanDiaconis", "PDF"];
n = 64;
p = 0.5;
g2 = DiscretePlot[PDF[BinomialDistribution[n, p], k], {k, 0, n},
   PlotMarkers -> Point,
   Joined -> True, PlotStyle -> {{AbsolutePointSize[5], Red}}
   ];
g3 = Show[g1, g2]
Export["Binomial64.pdf", g3]

g1 = Histogram[a, "Sturges", "CDF"];
g2 = DiscretePlot[CDF[BinomialDistribution[n, p], k], {k, 0, n},
   PlotMarkers -> Point,
   Joined -> True, PlotStyle -> {{AbsolutePointSize[5], Red}}
   ];
g3 = Show[g1, g2]
```

2. (20 pts) Assuming $p = 0.5$, *standardize* each variable. Plot the histograms and empirical distribution. Superimpose the standard normal density on the histogram, and superimpose the empirical distribution and cdf. Print and submit your plots. Using the "eyeball criterion," how good does this look? Which method, the histogram or the empirical distribution, seems to be better?

   Hint for R: The `curve` command plots functions and the `pnorm` and `dnorm` functions give the standard normal cdf and density.

   ```
   std = (raw - n*p)/sqrt(n*p*(1-p))
   emp = ecdf(std)
   plot(emp, main="Empirical CDF of Standardized Binomial64 Data")
   curve(pnorm,add=TRUE,col="red")
   ```

   For the histogram, try:

   ```
   hist(std, freq=FALSE, main="Histogram of Standardized Binomial64 Data")
   curve(dnorm,add=TRUE,col="red")
   ```

   Hint for Mathematica:

   ```
   raw = Flatten[Import["Binomial64", "Table"]];
   std = (raw - n p)/Sqrt[n p (1 - p)];

   g1 = Histogram[std, "Sturges", "CDF"];
   g2 = Plot[CDF[NormalDistribution[0, 1], x], {x, -4, 4}
      ];
   g3 = Show[g1, g2]

   g1 = Histogram[std, "Sturges", "PDF"];
   g2 = Plot[PDF[NormalDistribution[0, 1], x], {x, -4, 4}];
   g3 = Show[g1, g2]
   ```

3. (10 pts) There is another way to test how well the standardized data fit the standard normal, which may be even easier to visualize. It is called a **Normal QQ plot**. The **quantile function** q of a distribution with a continuous increasing cdf F is just the inverse of the cdf F^{-1}. That is, for $0 \leqslant p \leqslant 1$, $q(p)$ is the number x such that $p = F(x) = P(X \leqslant x)$. For distributions with jumps and flat spots, R's

quantile function uses interpolation. If you have enough data the empirical cdf is pretty close to being continuous, so the interpolation is not a serious issue.

A QQ plot plots the quantiles of one distribution against the quantiles of the other. If the distributions are the same, then the QQ plot will be a straight line of slope 1 through the origin. This is an easy condition to check visually. If the slope is 1 but not through the origin the the random variables differ by a constant. If the slope is not 1, the the variables are scaled. If the plot is not close to a straight line then the random variables probably do not have the same distribution.

A Normal QQ plot plots the quantiles of the empirical cdf against those of a standard Normal. R has a function for it, `qqnorm`.

Make a Normal QQ plot for your standardized data. Say something intelligent about the plot to show that you have looked at it.

Hint: If you've been using my R code, just type

```
qqnorm(std, main="Normal QQ Plot for Binomial64 Data")
qqline(std)
```

For Mathematica, try

```
QuantilePlot[std]
```

□

Exercise 2 (40 pts) Pitman [1, Exercise 3.Review.30, p. 255.]

Diagonal neighbor random walk. Let (S_n, T_n) denote the position after n steps of a random walk on the lattice of points in the plane with integer coordinates, starting from $(S_0, T_0) = (0, 0)$. Suppose that $S_{n+1} = S_n \pm 1$ and $T_{n+1} = T_n \pm 1$ where the signs are picked by two independent tosses of a fair coin, independently at each step.

a. For $c > 0$, find the limit as $n \to \infty$ of the probability that (S_n, T_n) is inside the square with corners at $(\pm c\sqrt{n}, \pm c\sqrt{n})$.

b. Let $R_n = \sqrt{S_n^2 + T_n^2}$, the distance from the origin. Find $\boldsymbol{E}(R_n^2)$.

c. Find b, as small as you can, such that $\boldsymbol{E}(R_n) \leqslant \sqrt{bn}$ for every n.

d. Let p_n denote the probability that the random walk is at $(0, 0)$ after n steps. Find the numerical value of p_3 (as a decimal).

e. Show that $p_{2m} \sim c/m$ as $m \to \infty$ for a constant c. What is c?

Hint: Think. □

Exercise 3 (10 pts) How much time did you spend on the previous exercises? □

Exercise 4 (Optional Exercise) (50 pts) Select n points on a circle independently according to a uniform distribution. What is the probability that all of them lie on the same semicircle?

Hint: Think before you calculate. □

References

[1] J. Pitman. 1993. *Probability*. New York, Berlin, and Heidelberg: Springer.