1. Computational Exercise 1

The R statistical package is a powerful open source program that provides many of the capabilities of commercial packages and is available in computer labs on campus. Unlike most commercial packages, you can also download and install R on your own computer. See the Technology section of the course web page,

(http://www.sandgquinn.org/stonehill/MTH395/Fall2012/index.html#tech)

for instructions on how to do this.

In this exercise we will use R to perform a numerical simulation, also known as a *monte carlo experiment*, to explore the properties of mean and variance for binomial random variables.

1.1. **Sample mean vs. Population mean.** In section 2.12, the author defines a *random sample* as a collection of n elements chosen from a population of size N in such a way that all

$$\binom{N}{n}$$

possible subsets have an equal probability of being selected.

Of course this applies to finite populations only. More generally, a random sample is a sample of size n chosen in a way that gives every possible sample the same chance of being selected.

In section 3.3, we defined the mean and variance for a *random variable* Y with probability function $P(Y = y) = p(y)$ as:

$$E(Y) = \mu = \sum_y y \cdot p(y)$$

and

$$V(Y) = \sigma^2 = \sum_y [y - E(Y)]^2 \cdot p(y) = E(Y^2) - [E(Y)]^2$$

In section 1.3, the author defines two related quantities computed from a random sample.

Definition (sample mean)**.** *The* ***sample mean*** *of a random sample* $y_1, y_2, \ldots, y_n$ *of size* n *is defined as:*

$$\overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

Definition (sample variance)**.** *The* ***sample variance*** *of a random sample* $y_1, y_2, \ldots, y_n$ *of size* n *is defined to be:*

$$s^2 = \frac{1}{n-1}\sum_{y=1}^{n}(y_i - \overline{y})^2$$

The symbols $\overline{y}$ and s^2 are used to distinguish the **sample** mean and variance from the **population** mean and variance, which are traditionally denoted by μ and σ^2, respectively.

Note that the divisor in the definition of s^2 is $n-1$, not n as you might expect. This is done because, as we will see, if we took a very large collection of random samples of size n and computed

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - \overline{y})^2$$

for each of them, the mean of these values would approach

$$\frac{n-1}{n}\sigma^2$$

as the size of the collection became larger and larger. The $n-1$ factor ensures that the mean approaches σ^2, the population variance, which makes s^2 an *unbiased* estimate of σ^2. This just says that it has no tendency to over or underestimate σ^2. In any case, as $n \to \infty$, the leading factor $(n-1)/n \to 1$.

It is my opinion that more emphasis than necessary is placed on the division by $n-1$, because for all but the smallest samples, it makes very little difference. Secondly, for a normally distributed population dividing by n produces what is known as a *maximum likelihood* estimate of σ^2. Maximum likelihood estimates have many desirable properties.

1.2. **Using R to simulate the binomial experiment.** R has facilities built in for generating artificial samples from common distributions. In particular, to generate a vector of values that simulate repeating the binomial experiment with n trials and probability of success p a total of N times, enter following at the command prompt: (after entering R at the command prompt to start R, or clicking the shortcut)

```
x<-rbinom(N,n,p)
```

To generate a sample of size $N = 10,000$ from a binomial experiment with 10 trials and probability of success $p = 0.5$, enter:

```
x<-rbinom(10000,10,0.5)
```

One thing people new to R find frustrating is that usually if the command you entered worked, you just get a command prompt with no indication that anything happened. If this is the case, it means R created an array of values with 10,000 entries and stored them in a vector called `x`. You can display the first 20 elements of `x` by typing

```
x[1:20]
```

which should produce something like:

```
 [1] 6 6 5 3 6 4 4 5 6 6 6 7 5 4 4 4 6 2 8 7
```

Note that all of the entries in `x` are positive integers between 0 and 10, inclusive, as you would expect because this is the range of a binomial random variable with $n = 10$. You can produce a histogram of the 10,000 values by typing

```
hist(x)
```

This should illustrate that the outcome of the binomial experiment with $n = 10$ and $p = 0.5$ that is most likely is 5, and the likelihood decreases as we move towards zero and ten. You are encouraged to experiment with different values of p and n by modifying the R statements accordingly.

You can use the up-arrow key to recall previously typed commands so you do not have to keep typing them. To exit R, type

```
quit()
```

You will be prompted to save the image of your R session for next time with something like:

```
Save workspace image?  [y/n/c]:
```

This is usually a good idea because if you respond with "y", the next time you start R all of your data and command history will be automatically restored, which prevents starting from scratch each time. To see what is in your workspace, type

```
ls()
```

Problem 1) Modify the probability function for the binomial distribution

$$p(x) = P(X = x) = \binom{10}{x} \left(\frac{1}{2}\right)^x \left(1 - \frac{1}{2}\right)^{10-x} \qquad x = 0, 1, 2, \ldots, 10$$

to give the probability function

$$p(y) = P(Y = y)$$

for the random variable $Y = 2X - 10$. What is $E(Y)$? (a numerical answer is acceptable)

Problem 2) Use R to generate a simulated sample of size 10,000 from a binomial population with $n = 10$ and $p = 0.5$ and store it in an array called `x`. Compute the sample mean $\overline{x}$ and the sample variance s_x^2 using:

```
mean(x)
```

and

```
var(x)
```

How do the results compare to the population mean and variance for a binomial distribution given in Theorem 3.7?

Problem 3) Compute a new array of values called y to simulate the values of the random variable for the random walk using the following R command:

```
y<-2*x-10
```

Compute the sample mean $\overline{y}$ and the sample variance s_y^2 using:

`mean(y)` and `var(y)`

How do the results compare with your answers to problem 1)?