Lisa Over
Homework 11
April 14, 2015

CODE for Metropolis Function

#metro function receives nine parameters: two vectors "x.s" and “y.s” for the data values of interest, "a0" for the initial alpha, "b0" for the initial beta, "k" for the alpha interval, "c" for the beta interval, "N" for number of independent random normal realizations, "lag" for determining how many realizations to skip between saves, and "burnin" for determining how many realizations to skip before starting to save.

```
metro <- function(x.s,y.s,a0,b0,k,c,N,lag,burnin) {

        #Set N to be N*lag+burnin
        N <- N*lag + burnin

        #Initialize vectors to hold alpha (a.v) and beta (b.v) values
        a.s <- NULL
        b.s <- NULL

        #store the acceptance rate for alpha (a.cnt) and beta (b.cnt)
        a.cnt = 0
        b.cnt = 0

  for(i in 1:N) {

#Generate an alpha star 'a.star' from the proposal density (Normal) using a0 as mu and k as
std dev
        a.star = rnorm(1,a0,k)

#Compute probability for alpha using Normal priors with mean (0), variance (100) -- Use
full conditional (from joint posterior which is likelihood*prior(alpha)*prior(beta)) --
compute ratio of full conditional given a.star to the full conditional given a0
        numerator = sum(y.s*a.star) - sum(log(1 + exp(a.star + b0*x.s))) - a.star^2/200
        denominator = sum(y.s*a0) - sum(log(1 + exp(a0 + b0*x.s))) - a0^2/200
        target.ratio = exp(numerator - denominator)

        #Use target.ratio to determine if a.star should be accepted
        param = 0
        accept.code = 0
        if(target.ratio < 1) {
                if(runif(1,0,1) < target.ratio) {
                        param = a.star
                        accept.code = 1
                }
                else {
                        param = a0
                        accept.code = 0
                }
        }
        else {
                param = a.star
```

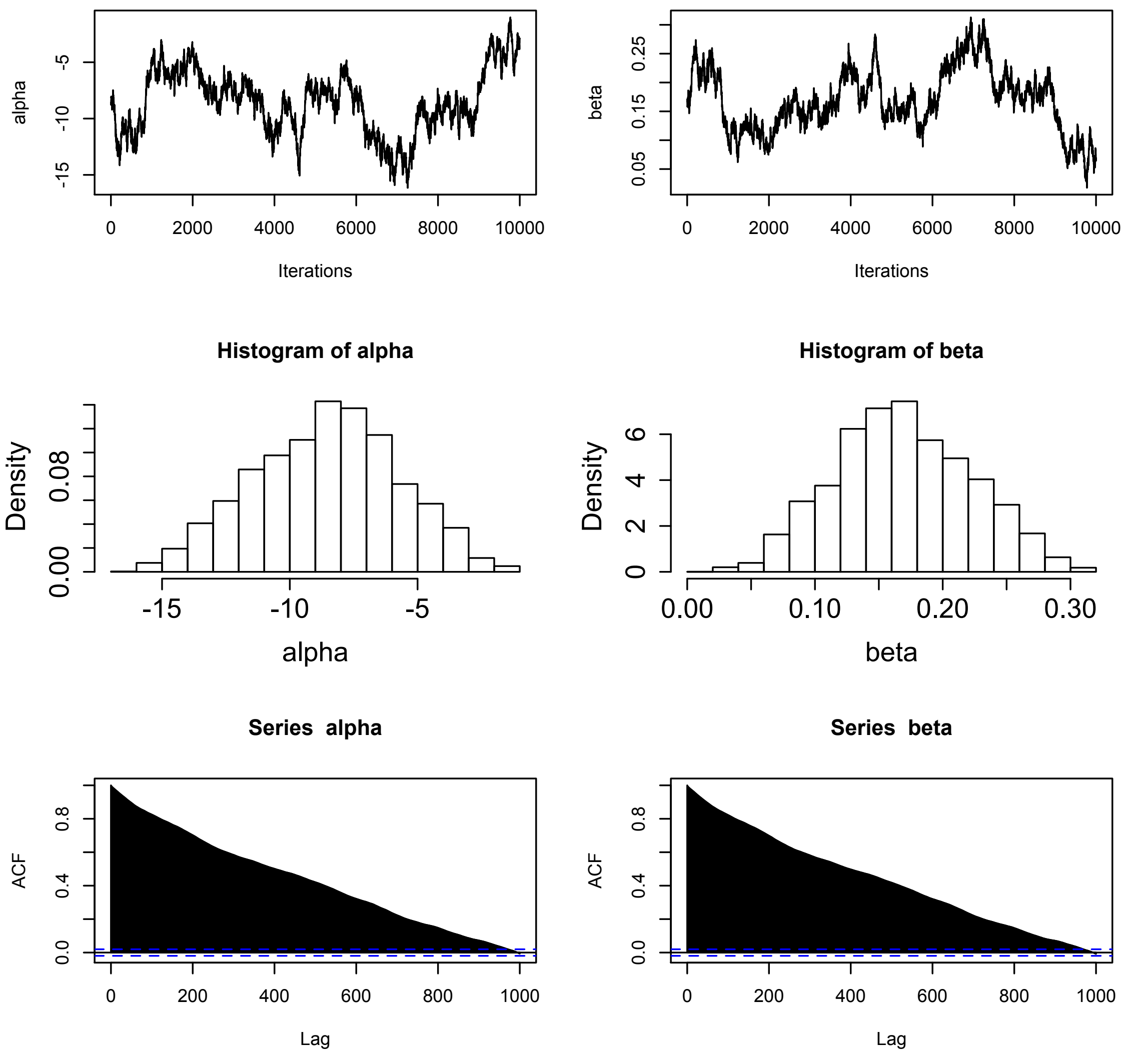
```
                accept.code = 1
        }

#if i is greater than burnin and if i is a multiple of the lag, store value
#add accept.code to total a.cnt - accept.code will be 1 if a.star was accepted and 0 ow
#add accepted value to vector - a0 or a.star - stored as param
        if(i > burnin) {
        if(i %% lag == 0) {
                a.cnt = a.cnt + accept.code
                a.s = c(a.s, param)
                #set a0 equal to param (parameter that was stored in vector -- a.star or a0)
for next iteration
                a0 = param
  }
 }

#Reset target.ratio, numerator, denominator
numerator = 0
denominator = 0
target.ratio = 0
#Generate a beta star 'b.star' from the proposal density (Normal) using b0 as mu and c as
std dev
        b.star = rnorm(1,b0,c)

#Compute probability for beta using Normal priors with mean (0), variance (100) -- Use full
conditional (from joint posterior which is likelihood*prior(alpha)*prior(beta)) -- compute
ratio of full conditional given b.star to the full conditional given b0
        numerator = sum(y.s*b.star*x.s) - sum(log(1 + exp(a.star + b.star*x.s))) -
b.star^2/200
        denominator = sum(y.s*b0*x.s) - sum(log(1 + exp(a.star + b0*x.s))) - b0^2/200
        target.ratio = exp(numerator - denominator)

        #Use target.ratio to determine if b.star should be accepted
        param = 0
        accept.code = 0
        if(target.ratio < 1) {
                if(runif(1,0,1) < target.ratio) {
                        param = b.star
                        accept.code = 1
                }
                else {
                        param = b0
                        accept.code = 0
                }
        }
        else {
                param = b.star
                accept.code = 1
        }
```

Lisa Over
Homework 11
April 14, 2015

```
#if i is greater than burnin and if i is a multiple of the lag, store value
#add accept.code to total b.cnt - accept.code will be 1 if b.star was accepted and 0 ow
#add accepted value to vector - b0 or b.star - stored as param
        if(i > burnin) {
        if(i %% lag == 0) {
                b.cnt = b.cnt + accept.code
                b.s = c(b.s, param)
                #set b0 equal to param (parameter that was stored in vector -- b.star or b0)
for next iteration
                b0 = param
   }
  }
 }
 vectors <- list("alpha" = a.s, "aCount" = a.cnt, "beta" = b.s, "bCount" = b.cnt)
 return(vectors)
}

#Convert to dataframe and attach
df = data.frame(state.x77)
attach(df)

#Prepare data for use in function metro
perCapita = 4445
y.s = as.numeric(Income > perCapita)
x.s = HS.Grad

#Set number or realizations, lag, and burnin
N = 10000
lag = 1000
burnin = 20

#Set k and c, interval limits for alpha and beta, respectively
k=.35
c=.007

#Obtain alpha and beta from data to use for a0 and b0
output = summary(glm(y.s~x.s,family=binomial))
a0 = output$coef[1,1]
b0 = output$coef[2,1]

vectors = metro(x.s,y.s,a0,b0,k,c,N,lag,burnin)
```

Lisa Over
Homework 11
April 14, 2015

RESULTS for Metropolis Method



Plots with lag=1000, k=0.35, and c=.007

Acceptance probabilities: alpha -> 0.6984; beta -> 0.6786

Mean and variance: alpha -> -8.5736, 8; beta -> 0.169, 0.003

95% credible interval for beta: (0.0692, 0.2720)

Lisa Over
Homework 11
April 14, 2015

CODE and RESULTS for Built-in R Function

```
#Convert to dataframe and attach
df = data.frame(state.x77)
attach(df)

perCapita = 4445
y.s = as.numeric(Income > perCapita)
x.s = HS.Grad

#Obtain alpha and beta from data to use for a0 and b0
output = summary(glm(y.s~x.s,family=binomial))


Call:
glm(formula = y.s ~ x.s, family = binomial)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3269  -0.6611   0.5067   0.9300   1.4732

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -8.02370    2.63976  -3.040  0.00237 **
x.s          0.15842    0.04976   3.184  0.00145 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 68.029  on 49  degrees of freedom
Residual deviance: 53.848  on 48  degrees of freedom
AIC: 57.848

Number of Fisher Scoring iterations: 4
```

Lisa Over
Homework 11
April 14, 2015

CODE and RESULTS for Built-in SAS Function

R code to export data

```
# export data frame to text file
 # write out text datafile and
 # a SAS program to read it
 library(foreign)
 write.foreign(thedata, "c:/Users/TA00/My Documents/thedata.txt",
"c:/Users/TA00/My Documents/thedata.sas",   package="SAS")
```

SAS code

```
DATA  rdata ;
INFILE  "c:/Users/TA00/My Documents/thedata.txt"
    DSD
    LRECL= 10 ;
INPUT
x_s
 y_s
;
LABEL  x_s = "x.s" ;
LABEL  y_s = "y.s" ;
RUN;

proc logistic data=y_s descending;
model y_s=x_s;
run;
```

Lisa Over
Homework 11
April 14, 2015

The SAS System

The LOGISTIC Procedure

Model Information		
Data Set	WORK.RDATA	
Response Variable	y_s	y.s
Number of Response Levels	2	
Model	binary logit	
Optimization Technique	Fisher's scoring	

Number of Observations Read	50
Number of Observations Used	50

Response Profile		
Ordered Value	y_s	Total Frequency
1	0	21
2	1	29

Probability modeled is y_s=0.

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	70.029	57.848
SC	71.941	61.672
-2 Log L	68.029	53.848

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	14.1811	1	0.0002
Score	12.9365	1	0.0003
Wald	10.1360	1	0.0015

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	8.0237	2.6398	9.2389	0.0024
x_s	1	-0.1584	0.0498	10.1360	0.0015

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
x_s	0.853	0.774	0.941

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	76.0	Somers' D	0.522
Percent Discordant	23.8	Gamma	0.523
Percent Tied	0.2	Tau-a	0.260
Pairs	609	c	0.761

Lisa Over
Homework 11
April 14, 2015

CONCLUSION

The results from four methods—metropolis function, built-in R, built-in SAS, and JMP—are consistent, and the coefficient beta is significant. This indicates that the graduation rate can be used to classify a region as having above or below the average per capita income.

	Beta	**Statistical significance**
Metropolis For log odds of above/below (per capita)	0.169	H_o: Beta = 0; Credible Interval: (0.0692, 0.2720) Does not contain 0
R For log odds of above/below (per capita)	0.15842	H_o: Beta = 0; P-value: 0.00145 Whole model ChiSq 14.181 (difference between null deviance and residual deviance)
SAS For log odds of below/above (per capita)	-0.1584	H_o: Beta = 0; P-value Wald ChiSq: 0.0015 Global null hypothesis ChiSq 14.181 with p-value 0.0002
JMP For log odds of above/below (per capita)	0.1584	P-value Wald ChiSq: 0.0015 -LogLikelihood difference: 7.09; Whole model ChiSq 14.181 with p-value 0.0002

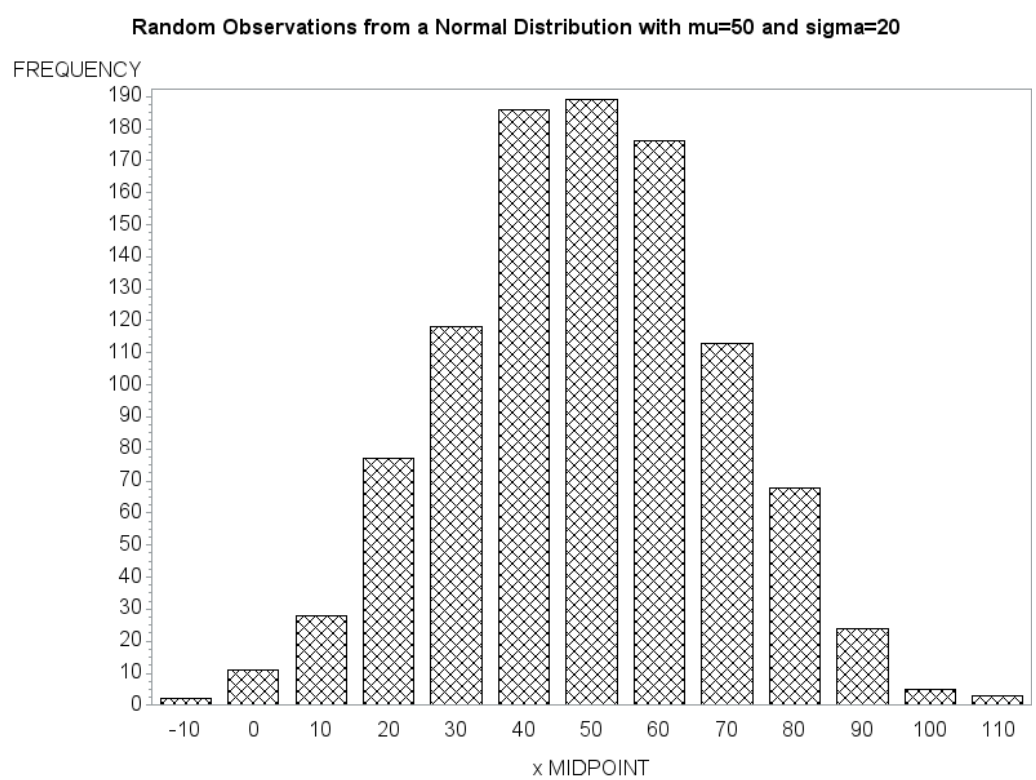
Lisa Over
Homework 11
April 14, 2015

SAS Random Variables

8.1

```
data one;
    mu = 50;
    sd = 20;
    seed = 46327;
    do i = 1 to 1000;
            z = rannor(seed);
            x = (z*sd) + mu;
            output;
    end;
run;

goptions cpattern = black htext = 1.5;

proc gchart;
vbar x/ space = 0 midpoints = -10 to 110 by 10 width = 8;
title 'Random Observations from a Normal Distribution with mu=50 and sigma=20';
run;
```

The distribution of the 1000 random Normal realizations is Normally distributed with mean 50 and standard deviation 20 as illustrated by the histogram below.



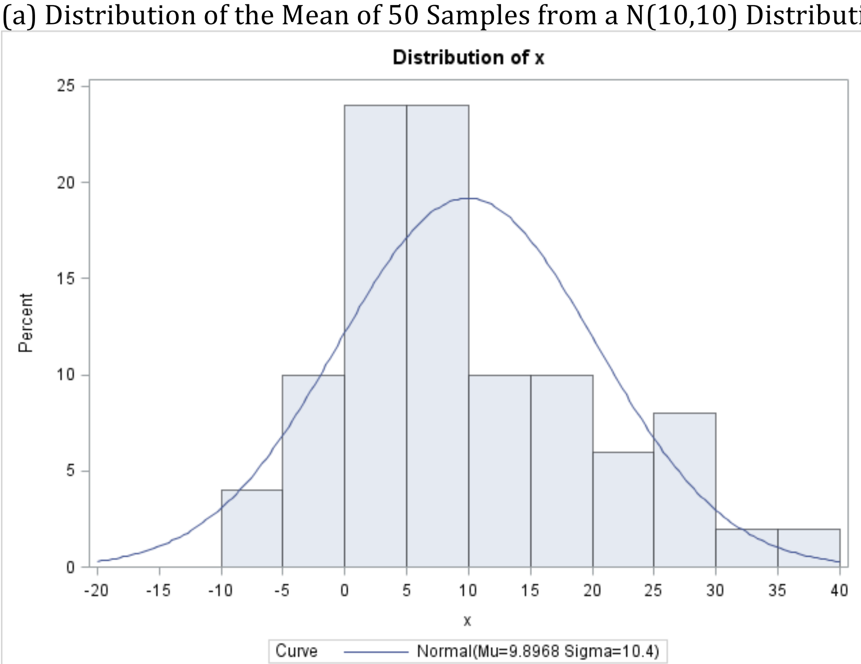
Lisa Over
Homework 11
April 14, 2015

8.2

```
data one;
    mu = 10;
    sd = 10;
    N = 5000;
    seed = 46327;
    do i = 1 to N;
        z = rannor(seed);
        x = (z*sd) + mu;
        output;
    end;
run;

goptions cpattern = black htext = 1.5;

proc univariate;
var x;
histogram x/ normal endpoints = -20 to 40 by 5;
title1 'Distribution of the Mean of 5000 Samples';
title2 'from a N(10,10) Distribution';
run;
```

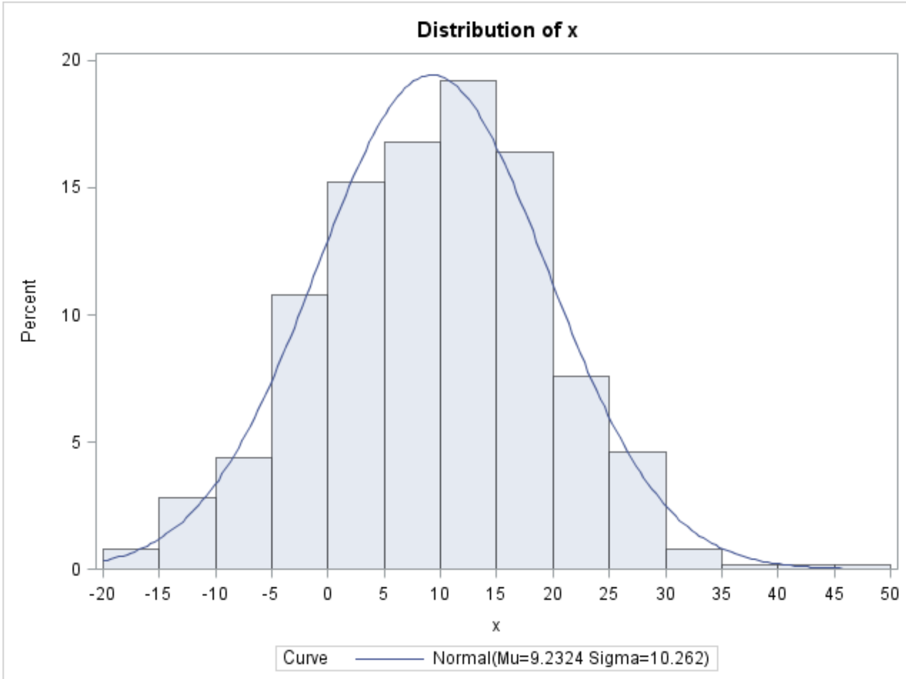
As the sample sizes increase from 50 to 500 to 5000, the distributions of the means move from not looking Normal at all (a) to looking very Normal (c). The distribution in (a) is significantly right skewed. The distribution in (b) is slightly left skewed. The distribution in (c) is symmetric.

(a) Distribution of the Mean of 50 Samples from a N(10,10) Distribution

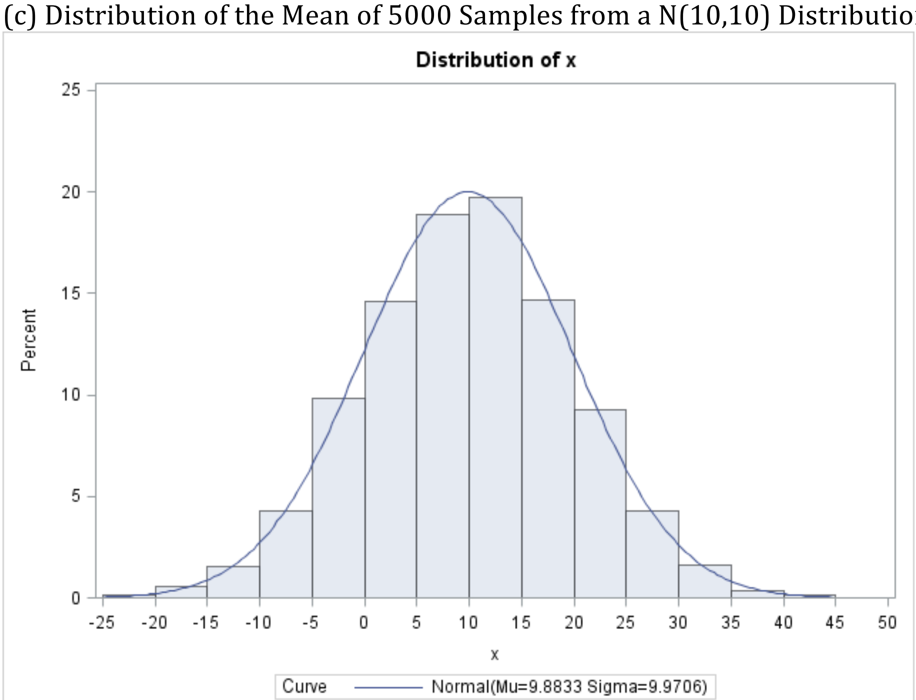


(b) Distribution of the Mean of 500 Samples from a N(10,10) Distribution

Lisa Over
Homework 11
April 14, 2015



(c) Distribution of the Mean of 5000 Samples from a N(10,10) Distribution

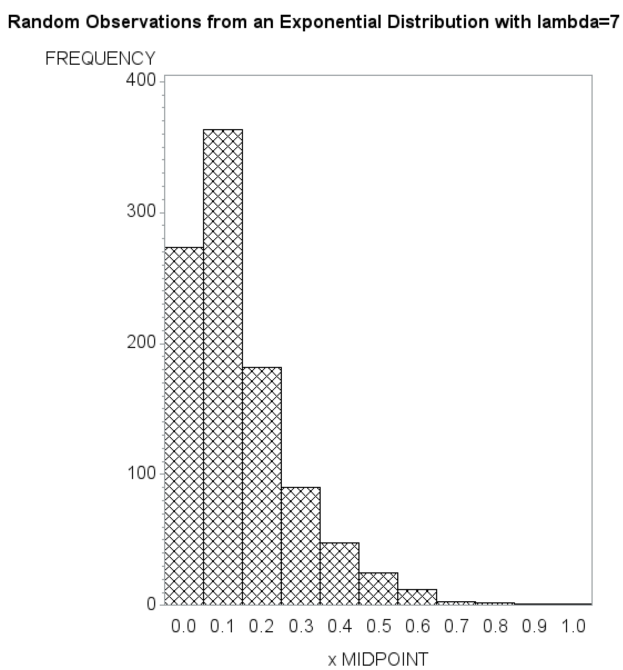


Lisa Over
Homework 11
April 14, 2015

8.3

```
data one;
    lambda = 7;
    N = 1000;
    seed = 46327;
    do i = 1 to N;
            z = ranexp(seed);
            x = z/lambda;
            output;
    end;
run;

goptions cpattern = black htext = 1.5;

proc gchart;
vbar x/ space = 0 midpoints = 0 to 1 by .1 width = 5;
title 'Random Observations from an Exponential Distribution with lambda=7';
run;
```

The 1000 exponential realizations have an exponential distribution with parameter lambda=7.

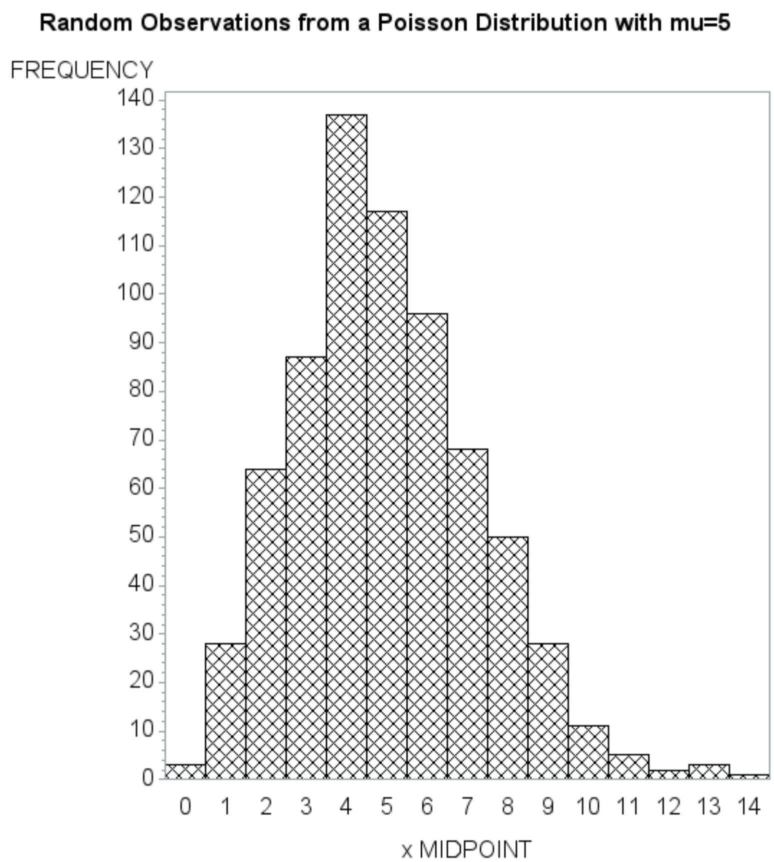


Lisa Over
Homework 11
April 14, 2015

8.4

```
data one;
    mu = 5;
    N = 700;
    do i = 1 to N;
            x = ranpoi(46327,mu);
            output;
    end;
run;

goptions cpattern = black htext = 1.5;

proc gchart;
vbar x/ space = 0;
title 'Random Observations from a Poisson Distribution with mu=5;
run;
```

The 700 Poisson realizations have a bell shaped distribution that is skewed slightly to the right with mean 5.

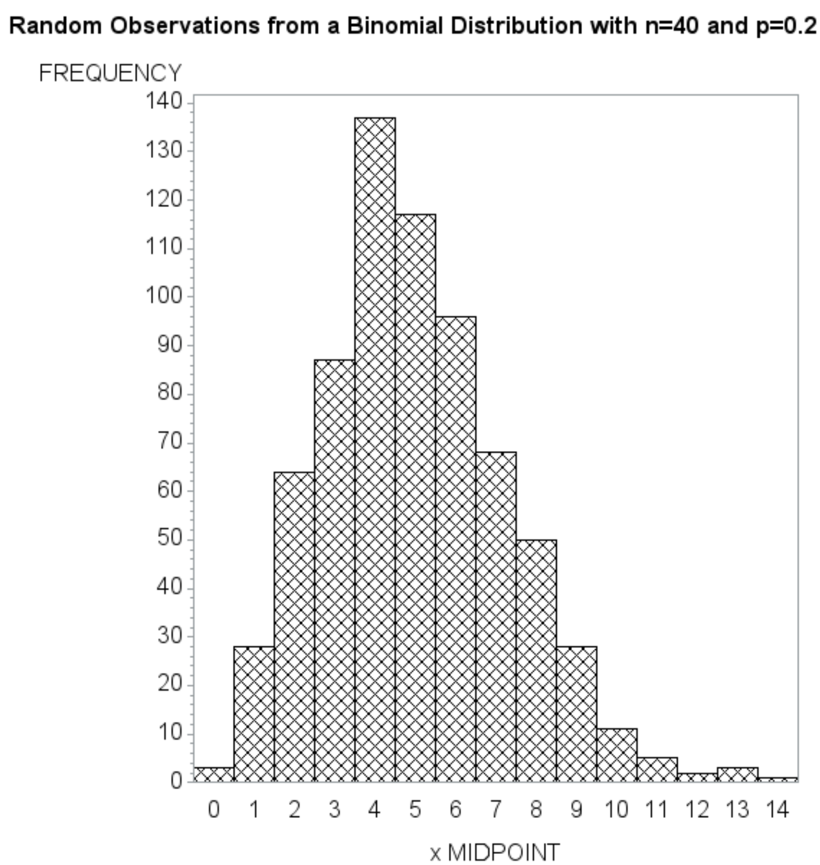


Lisa Over
Homework 11
April 14, 2015

8.5

```
data one;
    N = 500;
    n = 40;
    p = 0.2;
    seed = 4491;
    do i to N;
        x = ranbin(seed, n, p);
        output;
    end;
    run;

    goptions cpattern = black htext = 1.5;

proc gchart;
vbar x/ space = 0 midpoints = 0 to 1 by .1 width = 5;
title 'Random Observations from a Binomial Distribution with n=40 and p=0.2';
run;
```

The 500 binomial realizations have a bell shaped distribution that is skewed slightly to the right with mean 5.



Lisa Over
Homework 11
April 14, 2015

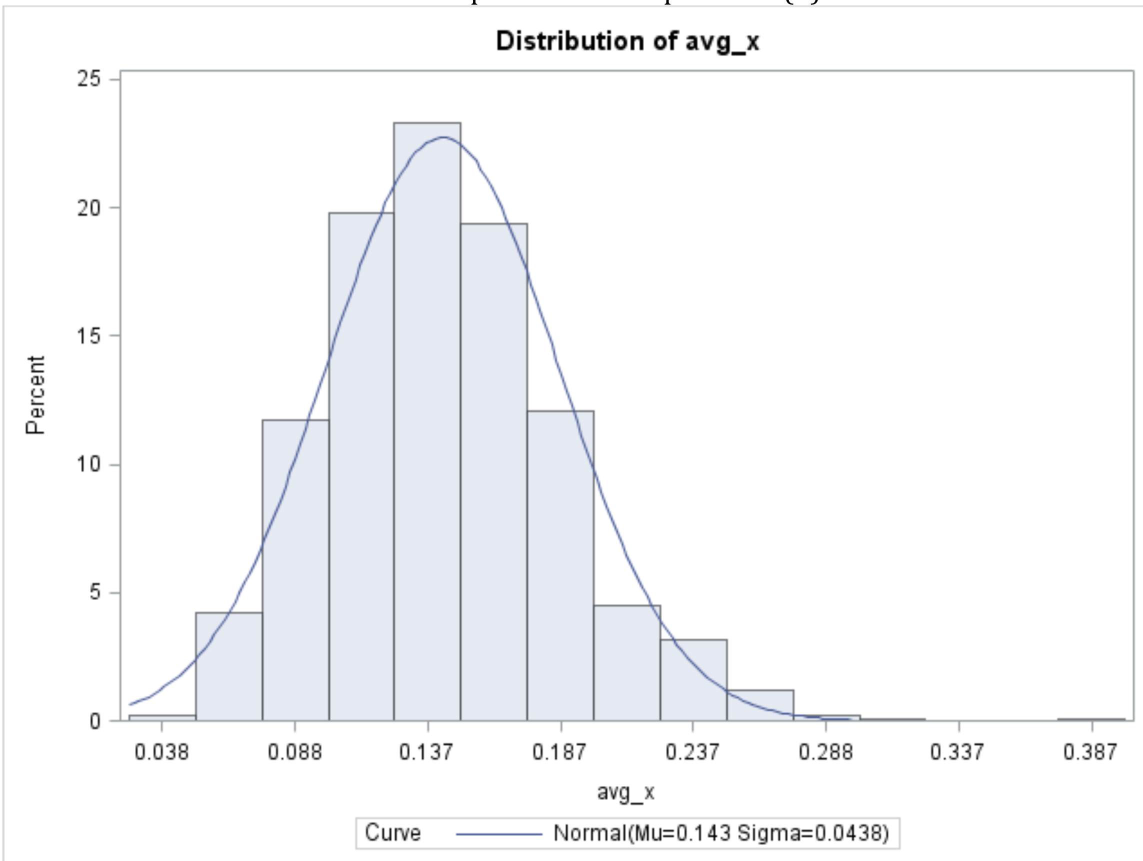
8.6

```
data one;
    lambda = 7;
    N = 1000;
    seed = 46327;
    do i = 1 to N;
            z1 = ranexp(seed);
            x1 = z1/lambda;
            z2 = ranexp(seed);
            x2 = z2/lambda;
            z3 = ranexp(seed);
            x3 = z3/lambda;
            z4 = ranexp(seed);
            x4 = z4/lambda;
            z5 = ranexp(seed);
            x5 = z5/lambda;
            z6 = ranexp(seed);
            x6 = z6/lambda;
            z7 = ranexp(seed);
            x7 = z7/lambda;
            z8 = ranexp(seed);
            x8 = z8/lambda;
            z9 = ranexp(seed);
            x9 = z9/lambda;
            z10 = ranexp(seed);
            x10 = z10/lambda;
            avg_x = (x1+x2+x3+x4+x5+x6+x7+x8+x9+x10)/10;
            output;
    end;
run;

goptions cpattern = black htext = 1.5;

proc univariate;
var avg_x;
histogram avg_x/ normal;
title1 'Distribution of the Mean of 1000 Samples';
title2 'from an Exponential(7) Distribution';
run;
```

The 1000 realizations of the mean of ten random exponential(7) realizations have a Normal distribution with mean 0.143 and standard deviation 0.0438.

Lisa Over
Homework 11
April 14, 2015

Distribution of the Mean of 1000 Samples from an Exponential(7) Distribution



Lisa Over
Homework 11
April 14, 2015

8.7

```
data one;
	N = 1000;
	a = 10;
	b = 20;
	seed = 4491;
	do i to N;
		z1 = ranuni(seed);
		x1 = a+(b-a)*z1;
		z2 = ranuni(seed);
		x2 = a+(b-a)*z2;
		z1 = ranuni(seed);
		x1 = a+(b-a)*z1;
		z3 = ranuni(seed);
		x3 = a+(b-a)*z3;
		z4 = ranuni(seed);
		x4 = a+(b-a)*z4;
		z5 = ranuni(seed);
		x5 = a+(b-a)*z5;
		z6 = ranuni(seed);
		x6 = a+(b-a)*z6;
		z7 = ranuni(seed);
		x7 = a+(b-a)*z7;
		z8 = ranuni(seed);
		x8 = a+(b-a)*z8;
		z9 = ranuni(seed);
		x9 = a+(b-a)*z9;
		z10 = ranuni(seed);
		x10 = a+(b-a)*z10;
		avg_x = (x1+x2+x3+x4+x5+x6+x7+x8+x9+x10)/10;
		output;
	end;
	run;

	goptions cpattern = black htext = 1.5;

proc univariate;
var avg_x;
histogram avg_x/ normal;
title1 'Distribution of the Mean of 1000 Samples';
title2 'from a Uniform(10,20) Distribution';
run;
```

The 1000 realizations of the mean of ten random uniform(10, 20) realizations have a Normal distribution with mean 0.143 and standard deviation 0.0438.

Lisa Over
Homework 11
April 14, 2015

Distribution of the Mean of 1000 Samples from a Uniform (10,20) Distribution