Project Report #2

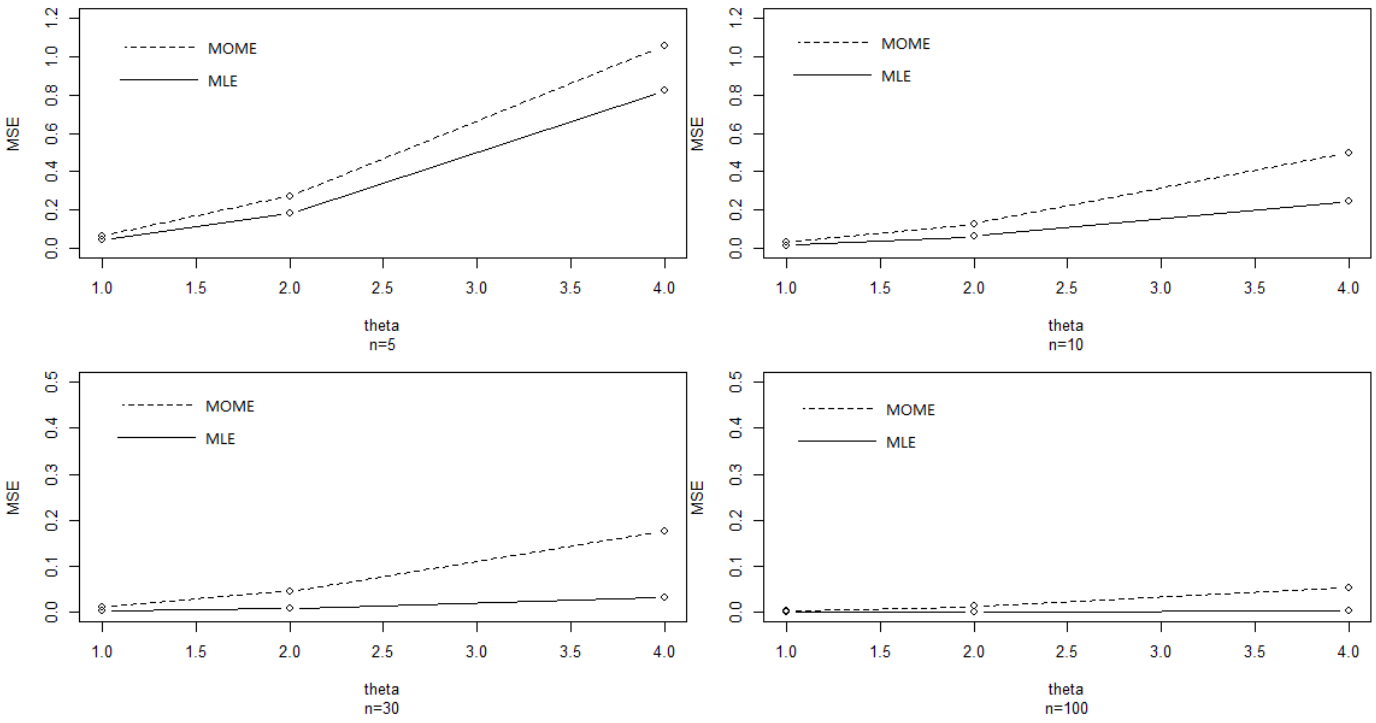
Section 1:

Exercise 1:
In the simulation, I am going to do 1000 experiments for each n and theta, then get the average of the MSE(mean squared errors) and make a graph for each n which contains both MSE for MLE and MOME to compare them. For Uniform distribution (0, theta) population, the MLE is the maximum value of data, and MOME is 2* mean(data). Then MSE is calculated by mean(estimator of theta – theta)^2). By using different n and theta and doing it 1000 times for each pair of n and theta and get the average of MSE, the comparison can be seen once the result is shown as graph.

Exercise 2:
Firstly, I get the lower bound and upper bound of confidence interval by using the following formula:
lower bound = estimator of p – $t_{n-1,\ alpha/2}$*(square root of variance of p)
upper bound = estimator of p + $t_{n-1,\ alpha/2}$ *(square root of variance of p)
Here, the estimator of p is calculated by (number of 1's / n). Variance of p is calculated by (estimator of p * (1 – estimator of p) /n . alpha = 0.05 given the confidence level is 95%. Then do this 1000 times for each pair of n and p, then see how many times the confidence interval covers the true p, which means p is less than the upper bound and larger than the lower bound. The coverage probability is the portion of numbers of experiments that covers true p in 1000 experiments.

Section 2:

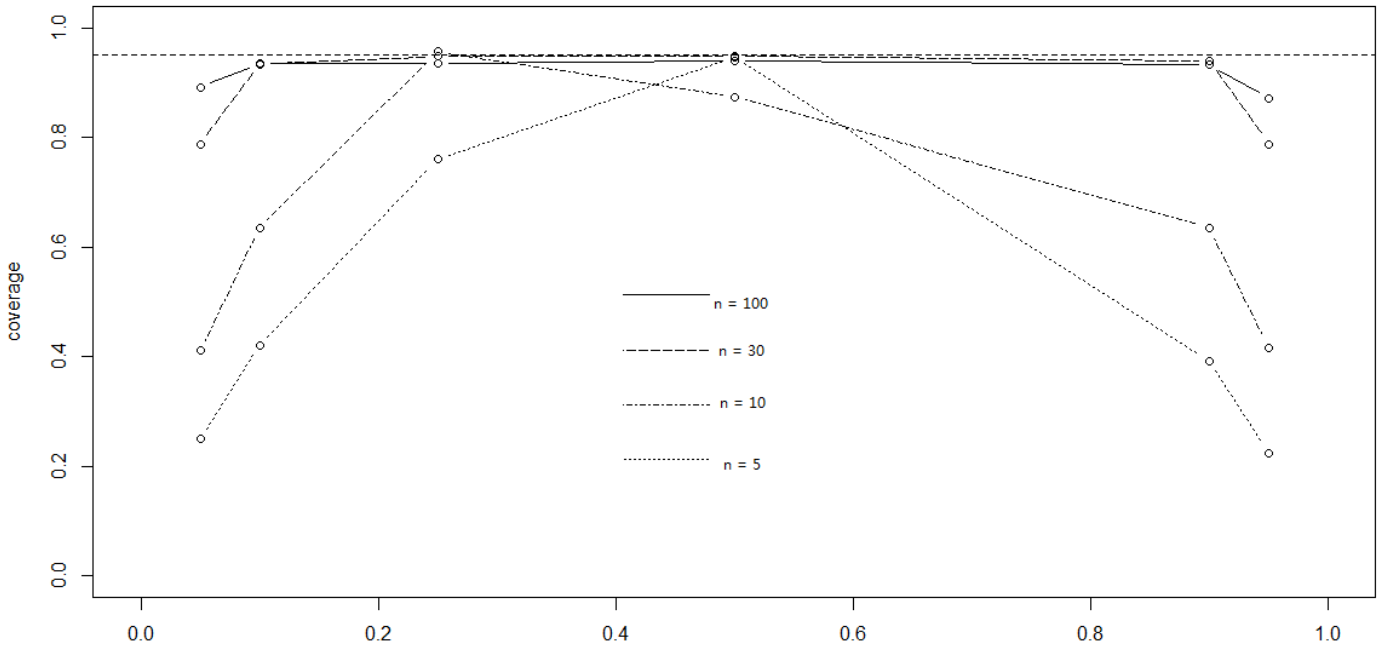
Exercise 1:
I made 4 graphs for exercise 1 as followed:



NOTE: the scales are not necessarily the same
from the graph, since the MOME cure is higher than the MLE cure, which means it is further from the true theta. I can conclude that generally speaking, MLE performs better in estimating theta of uniform distribution(0, theta). In addition, as n increases, the MSE of both MLE and MOME decreases. Beside, as theta increases, the MSE of both MLE and MOME decreases.

Exercise 2:

I made a graph for exercise 2 as followed:



from the graph, I can conclude that as n increases, the coverage rate is closer to the desired confidence level 95%. However, as the value of p increase, the coverage rate firstly increases and then decreases while the peak resides almost in p = 0.5. In addition, when n is small(less than 30), this pattern is easier to see, which mean the curve is more like a “mountain” while when n is large(greater than 30), the curve is flatter.

Section 3:

Exercise 1:

```
n = 1000 # number of experiments

generateData = function( num, par){ # generate Data
   result = runif( num, max = par)
   return( result )
}

MLE = function( data ){ # generate MLE, which is the max of data
   result = max( data )
   return( result )
}

MOME = function( data ){ # generate MOME, which is the 2 * mean( data )
   result = 2 * mean( data )
   return( result )
}


MSE = function( estPar, realPar ){ # calculate the mean squared error
```

```
  result = mean( ( estPar - realPar )^2 )
  return(result)
}

meanMSE.MLE = function( n, num, par ){ # get the mean of MSE for 1000 experiments of MLE
  mse = rep( 0, times = n ) # a vector to record result of experiments
  for( i in 1:n ){
    data = generateData( num, par ) # generate data
    mle = MLE( data )# get mle
    mse[i] = MSE( mle, par) # get mse for mle
  }
  result = mean( mse )
  return (result)
}

meanMSE.MOME = function( n, num, par ){ # get the mean of MSE for 1000 experiments of MOME
  mse = rep( 0, times = n ) # a vector to record result of experiments
  for( i in 1:n ){
    data = generateData( num, par ) # generate data
    mome = MOME( data ) # get mome
    mse[i] = MSE( mome, par) # get mse for mome
  }
  result = mean( mse )
  return (result)
}

#n = 5
MSE.MLE.n5.theta1 = meanMSE.MLE( n, 5, 1 ) # theta = 1, MLE
MSE.MOME.n5.theta1 = meanMSE.MOME( n, 5, 1) # theta = 1, MOME

MSE.MLE.n5.theta2 = meanMSE.MLE( n, 5, 2 ) # theta = 2, MLE
MSE.MOME.n5.theta2 = meanMSE.MOME( n, 5, 2) # theta = 2, MOME

MSE.MLE.n5.theta4 = meanMSE.MLE( n, 5, 4 ) # theta = 4, MLE
MSE.MOME.n5.theta4 = meanMSE.MOME( n, 5, 4) # theta = 4, MOME

#n = 10
MSE.MLE.n10.theta1 = meanMSE.MLE( n, 10, 1 ) # theta = 1, MLE
MSE.MOME.n10.theta1 = meanMSE.MOME( n, 10, 1) # theta = 1, MOME

MSE.MLE.n10.theta2 = meanMSE.MLE( n, 10, 2 )# theta = 2, MLE
MSE.MOME.n10.theta2 = meanMSE.MOME( n, 10, 2)# theta = 2, MOME

MSE.MLE.n10.theta4 = meanMSE.MLE( n, 10, 4 ) # theta = 4, MLE
MSE.MOME.n10.theta4 = meanMSE.MOME( n, 10, 4) # theta = 4, MOME

#n = 30
MSE.MLE.n30.theta1 = meanMSE.MLE( n, 30, 1 ) # theta = 1, MLE
MSE.MOME.n30.theta1 = meanMSE.MOME( n, 30, 1) # theta = 1, MOME
```

```
MSE.MLE.n30.theta2 = meanMSE.MLE( n, 30, 2 ) # theta = 2, MLE
MSE.MOME.n30.theta2 = meanMSE.MOME( n, 30, 2) # theta = 2, MOME

MSE.MLE.n30.theta4 = meanMSE.MLE( n, 30, 4 ) # theta = 4, MLE
MSE.MOME.n30.theta4 = meanMSE.MOME( n, 30, 4) # theta = 4, MOME

#n = 100
MSE.MLE.n100.theta1 = meanMSE.MLE( n, 100, 1 ) # theta = 1, MLE
MSE.MOME.n100.theta1 = meanMSE.MOME( n, 100, 1) # theta = 1, MOME

MSE.MLE.n100.theta2 = meanMSE.MLE( n, 100, 2 ) # theta = 2, MLE
MSE.MOME.n100.theta2 = meanMSE.MOME( n, 100, 2)# theta = 2,MOME

MSE.MLE.n100.theta4 = meanMSE.MLE( n, 100, 4 ) # theta = 4, MLE
MSE.MOME.n100.theta4 = meanMSE.MOME( n, 100, 4) # theta = 4, MOME

theta = c( 1, 2, 4) # x axis
MSE.MLE.n5 = c( MSE.MLE.n5.theta1, MSE.MLE.n5.theta2, MSE.MLE.n5.theta4)# y axis for n= 5, MLE
MSE.MOME.n5 = c( MSE.MOME.n5.theta1, MSE.MOME.n5.theta2, MSE.MOME.n5.theta4 )# y axis for n = 5, MOME
MSE.MLE.n10 = c( MSE.MLE.n10.theta1, MSE.MLE.n10.theta2, MSE.MLE.n10.theta4)# y axis for n= 10, MLE
MSE.MOME.n10 = c( MSE.MOME.n10.theta1, MSE.MOME.n10.theta2, MSE.MOME.n10.theta4 )# y axis for n = 10, MOME

MSE.MLE.n30 = c( MSE.MLE.n30.theta1, MSE.MLE.n30.theta2, MSE.MLE.n30.theta4) # y axis for n = 30, MLE
MSE.MOME.n30 = c( MSE.MOME.n30.theta1, MSE.MOME.n30.theta2, MSE.MOME.n30.theta4 )#y axis for n = 30, MOME

MSE.MLE.n100 = c( MSE.MLE.n100.theta1, MSE.MLE.n100.theta2, MSE.MLE.n100.theta4)# y axis for n = 100, MLE
MSE.MOME.n100 = c( MSE.MOME.n100.theta1, MSE.MOME.n100.theta2, MSE.MOME.n100.theta4 )# y axis for n = 100,
MOME

par(mfrow=c(2,2),mar=c(5,4,1,0))# 4 graph in 2 rows, 2 colums
plot( x = theta, y = MSE.MLE.n5,sub = "n=5",type = "b", xlab = "theta",ylab = "MSE",xlim = c(1,4), ylim = c(0,1.2))
lines( x = theta, y = MSE.MOME.n5, type = "b",lty = 2)

plot( x = theta, y = MSE.MLE.n10,sub = "n=10", type = "b", xlab = "theta",ylab = "MSE",xlim = c(1,4), ylim = c(0,1.2))
lines( x = theta, y = MSE.MOME.n10, type = "b",lty = 2)

plot( x = theta, y = MSE.MLE.n30,sub = "n=30", type = "b", xlab = "theta",ylab = "MSE",xlim = c(1,4), ylim = c(0,0.5))
lines( x = theta, y = MSE.MOME.n30, type = "b",lty = 2)

plot( x = theta, y = MSE.MLE.n100,sub = "n=100",type = "b", xlab = "theta",ylab = "MSE",xlim = c(1,4), ylim = c(0,0.5))
lines( x = theta, y = MSE.MOME.n100, type = "b",lty = 2)
```

Exercise 2:

```
n = 1000 # number of experiments
alpha = 0.05 # given confidence level 95%, alpha = 0.05

generateData = function( n, par ){ # gives n data from bernoulli distribution. par = p
    result = rbinom( n, 1, par )
```

```
    return( result )
}

CI = function( data, alpha ){ # get the CI from the data and alpha
    n = length( data) # get n
    p = sum( data ) / n # the estimator of p
    var = p*( 1 - p )/ n # get variance of p
    s = sqrt( var ) # get the square root of variance, which is the standard error of sample data
    t = qt( 1 - ( alpha / 2 ), n-1 ) # get pivet t
    result = p + c( -1, 1 ) * t * s # get the upper and lower bound
    return( result )
}

coverage = function( n, num, par, alpha ){ #get the coverage probability
    record = rep( 0, n ) # use a vector to store result of each experiments

    for ( i in 1:n ){ # for each experiment
        data = generateData( num, par ) # generate data
        ci = CI( data, alpha) # get the CI
        if( par >= ci[1] && par <= ci[2]){ # check if it covers the true value
            record[ i ] = 1 # if covers, set the value in record to 1
        }
    }
    result = sum( record ) / n # get the portion of experiments that covers real p
    return ( result )
}

# the following part is to get coverage probability from different pair of n and p
coverage.n5.p0.05 = coverage( n, 5, 0.05, alpha ) # n = 5, p = 0.05
coverage.n5.p0.1 = coverage( n, 5, 0.1, alpha ) # n = 5, p = 0.1
coverage.n5.p0.25 = coverage( n, 5, 0.25, alpha ) # n = 5, p = 0.25
coverage.n5.p0.5 =    coverage( n, 5, 0.5, alpha ) # n = 5, p = 0.5
coverage.n5.p0.9 = coverage( n, 5, 0.9, alpha ) # n = 5, p = 0.9
coverage.n5.p0.95 = coverage( n, 5, 0.95, alpha ) # n = 5, p = 0.95

coverage.n10.p0.05 = coverage( n, 10, 0.05, alpha ) # n = 10, p = 0.05
coverage.n10.p0.1 = coverage( n, 10 , 0.1, alpha )# n= 10, p = 0.1
coverage.n10.p0.25 = coverage( n, 10, 0.25, alpha )#n = 10, p = 0.25
coverage.n10.p0.5 = coverage( n, 10, 0.5, alpha ) #n = 10, p = 0.5
coverage.n10.p0.9 = coverage( n, 10, 0.9, alpha ) # n = 10, p = 0.9
coverage.n10.p0.95 = coverage( n, 10, 0.95, alpha ) # n = 10, p = 0.95

coverage.n30.p0.05 =    coverage( n, 30, 0.05, alpha ) # n = 30, p = 0.05
coverage.n30.p0.1 = coverage( n, 30, 0.1, alpha ) # n = 30, p = 0.1
coverage.n30.p0.25 = coverage( n, 30, 0.25, alpha ) # n = 30, p = 0.25
coverage.n30.p0.5 = coverage( n, 30, 0.5, alpha ) # n = 30, p = 0.5
coverage.n30.p0.9 = coverage( n, 30, 0.9, alpha ) # n = 30, p = 0.9
coverage.n30.p0.95 = coverage( n, 30, 0.95, alpha ) # n = 30, p = 0.95
```

```
coverage.n100.p0.05 = coverage( n, 100, 0.05, alpha ) # n = 100, p = 0.05
coverage.n100.p0.1 = coverage( n, 100, 0.1, alpha ) # n = 100, p = 0.1
coverage.n100.p0.25 = coverage( n, 100, 0.25, alpha ) # n = 100, p = 0.25
coverage.n100.p0.5 = coverage( n, 100, 0.5, alpha ) # n = 100, p = 0.5
coverage.n100.p0.9 = coverage( n, 100, 0.9, alpha ) # n = 100, p = 0.9
coverage.n100.p0.95 = coverage( n, 100, 0.95, alpha ) # n = 100, p = 0.95


# get the coverage probability for each n, which will serve as y value in the graph
coverage.n5 = c( coverage.n5.p0.05, coverage.n5.p0.1, coverage.n5.p0.25, coverage.n5.p0.5, coverage.n5.p0.9, coverage.n5.p0.95 )
coverage.n10 = c( coverage.n10.p0.05, coverage.n10.p0.1, coverage.n10.p0.25, coverage.n10.p0.5, coverage.n10.p0.9, coverage.n10.p0.95 )
coverage.n30 = c( coverage.n30.p0.05, coverage.n30.p0.1, coverage.n30.p0.25, coverage.n30.p0.5, coverage.n30.p0.9, coverage.n30.p0.95 )
coverage.n100 = c( coverage.n100.p0.05, coverage.n100.p0.1, coverage.n100.p0.25, coverage.n100.p0.5, coverage.n100.p0.9, coverage.n100.p0.95 )

# different p value serves as x value in the graph
p = c(0.05,0.1,0.25,0.5,0.9,0.95 )

par(mfrow=c(1,1),mar=c(5,4,1,0))# one graph
plot( x = p, y = coverage.n5, type = "b", xlab = "p",ylab = "coverage",xlim = c(0,1), ylim = c(0,1),lty = 3)# draw n = 5 curve
lines( x = p, y = coverage.n10, type = "b",lty = 4) # add n = 10 curve
lines( x = p, y = coverage.n30, type = "b",lty = 5) # add n = 30 curve
lines( x = p, y = coverage.n100, type = "b",lty = 1) # add n = 100 curve
abline( h = 0.95,lty = 2 ) # add 0.95 horizon line
```