Solo Group 10 - Project 3 - Karthik Ragunath Ananda Kumar

Solution 1 (a)

```
In order to compute the  Mean Squared Error (MSE), we first set a population parameter and then calculate the est
imator values by simulating values.
Then we calculate the estimated value (mean) of the square of the difference between the estimator(θ^) and the pa
rameter(θ) values.
```

Solution 1 (b)

```r
compute_estimates = function(n,theta){
  samples_generated = runif(n, min=0, max=theta)
  method_of_moments_estimator = 2 * mean(samples_generated)
  mle = max(samples_generated)
  mse1 <- (method_of_moments_estimator-theta)^2
  mse2 <- (mle-theta)^2
  return (c(mse1,mse2))
}

replicate_estimate_computation = function(n,theta){
  replicate_estimates = rowMeans(replicate(1000, compute_estimates(n, theta)))
  return (c(replicate_estimates[1], replicate_estimates[2]))
}
```

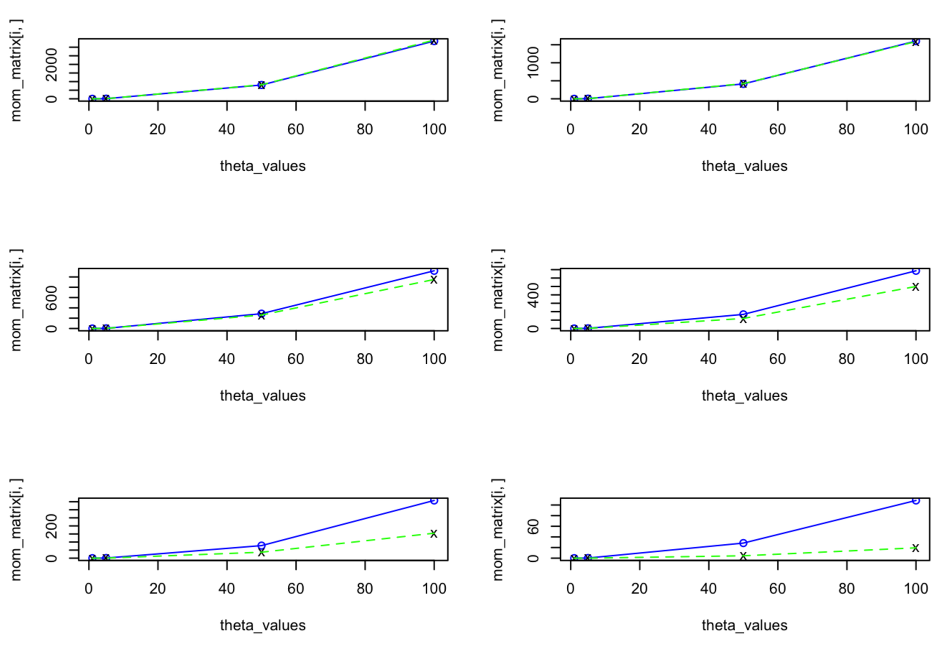
```r
replicate_estimate_computation(1, 1)
```

```
## [1] 0.3437179 0.3248368
```

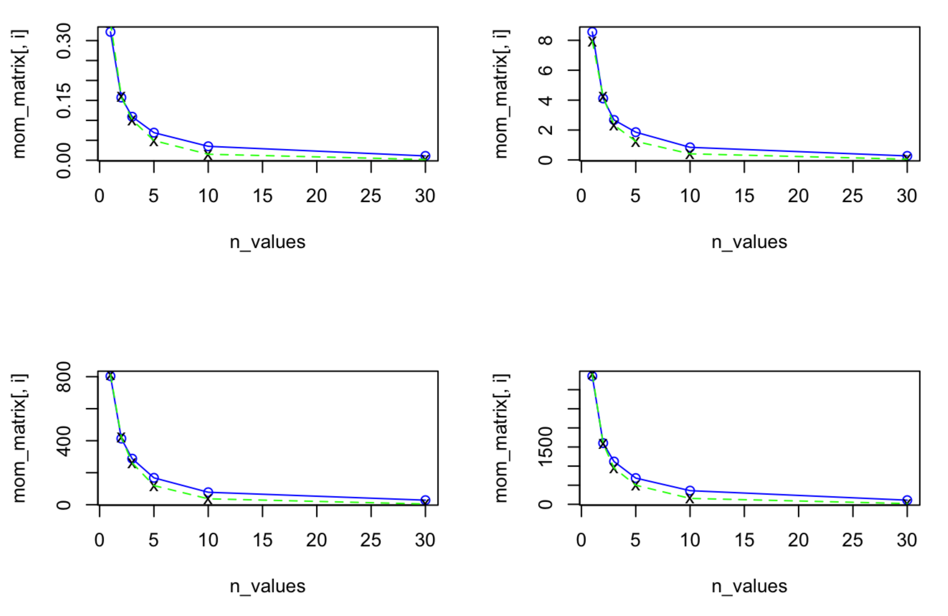
Solution 1(c)

```r
n_values = c(1,2,3,5,10,30)
theta_values = c(1,5,50,100)

mom_matrix = matrix(-1, length(n_values), length(theta_values),
                    dimnames = list(c("n1=1","n2=2","n3=3", "n4=5", "n5=10", "n6=30"),
                                    c("theta1=1", "theta=5", "theta=50", "theta=100")))
mle_matrix = matrix(-1, length(n_values), length(theta_values),
                    dimnames = list(c("n1=1","n2=2","n3=3", "n4=5", "n5=10", "n6=30"),
                                    c("theta1=1", "theta=5", "theta=50", "theta=100")))


for(i in 1:length(n_values)){
  for(j in 1:length(theta_values)){
    result = rowMeans(replicate(1000, compute_estimates(n_values[i],theta_values[j])))
    mom_matrix[i,j] = result[1]
    mle_matrix[i,j] = result[2]
  }
}
```

```r
par(mfrow=c(3,2))
for(i in 1:length(n_values)){
  plot(theta_values,mom_matrix[i,],col="blue",lty=1,type ='o')
  points(theta_values,mle_matrix[i,],pch='x')
  lines(theta_values,mle_matrix[i,],col="green",lty=2)
}
```




```r
par(mfrow=c(2,2))
for(i in 1:length(theta_values)){
  plot(n_values,mom_matrix[,i],col="blue",lty=1,type ='o')
  points(n_values,mle_matrix[,i],pch='x')
  lines(n_values,mle_matrix[,i],col="green",lty=2)
}
```



Solution 1(d)

```
From the first graph, for smaller values of n such as 1, 2 - Method of Moments estimator is slightly better than
Maximum Likelihood Estimator. But as n value increase, i.e. from n = 3, 5, 10, 30 - Maximum Likelihood Estimator
is better than Method of Moments Estimator. Therefore, in generic terms, Maximum Likelihood estimator is better t
han Method of Moments estimator.

From the second graph, we can't conclude anything conclusively on whether Maximum Likelihood estimator is better
than Method of Moments estimator.
```

Solution 2 (a) and 2 (b)

(2) (a)

$$f(x) = \begin{cases} \frac{\theta}{x^{\theta+1}}, & x \geq 1 \\ 0, & x < 1 \end{cases}$$

$$f(\theta) = \prod_{i=1}^{n} \left(\frac{\theta}{x_i^{\theta+1}} \right)$$

$$\ln(f(\theta)) = \ln \left(\prod_{i=1}^{n} \frac{\theta}{x_i^{\theta+1}} \right)$$

$$= \ln \left(\theta^n \cdot \prod_{i=1}^{n} \frac{1}{x_i^{\theta+1}} \right)$$

$$= n \ln \theta + \sum_{i=1}^{n} \ln(x_i^{-\theta-1})$$

$$= n \ln \theta + (-\theta-1) \sum_{i=1}^{n} \ln x_i$$

$$= n \ln \theta - \theta \sum_{i=1}^{n} \ln x_i - \sum_{i=1}^{n} \ln x_i$$

Inorder to find the min value, we differentiate with respect to θ and equate to zero

$$\frac{df(\theta)}{d\theta} = 0$$

$$\frac{d \ln(f(\theta))}{d\theta} = \frac{n}{\theta} - \sum_{i=1}^{n} \ln x_i - 0 = 0.$$

Scanned with CamScanner

Stats Question 2

$$\frac{n}{\theta} - \sum_{i=1}^{n} \ln(x_i) = 0.$$

$$\frac{n}{\theta} = \sum_{i=1}^{n} \ln(x_i)$$

$$\theta = \frac{n}{\sum_{i=1}^{n} \ln(x_i)}.$$

(b) From (a), we found that expression for MLE for $f(\theta)$ is given by the given

$$\theta = \frac{n}{\sum_{i=1}^{n} \ln(x_i)}$$

$$\theta = \frac{5}{\ln(21.42) + \ln(14.65) + \ln(50.42) + \ln(28.78) + \ln(11.23)}$$

$$= \frac{5}{3.07 + 2.68 + 3.92 + 3.35 + 2.41}$$

$$\theta = \frac{5}{15.43} = 0.32$$

Scanned with CamScanner

Stats Question 2

Solution 2 (c)

```r
mle_function = function(theta,dat){
  result=length(dat)*log(theta)-(theta+1)*sum(log(dat))
  return (-result)
}

x=c(21.42,14.65,50.42,28.78,11.23)
mle_function(1,x)
```

```
## [1] 30.89485
```

```r
theta = optim(par=1,fn=mle_function,method="L-BFGS-B",lower=0.01,hessian=TRUE,dat=x)
theta$par
```

```
## [1] 0.3236796
```

```
From the theta variable output, we can see that the theta value estimated using optim function is same as the the
ta value computed theorotically
```

Solution 2 (d)

```
qNorm computation:

1 - alpha = 0.95
alpha = 0.05
alpha / 2 = 0.025
1 - alpha = 0.975
```

```r
theta = optim(par=1,fn=mle_function,method="L-BFGS-B",lower=0.01,hessian=TRUE,dat=x)
theta$par
```

```
## [1] 0.3236796
```

```r
standard_error = (1/theta$hessian)^0.5
standard_error
```

```r
confidence_interval = theta$par+c(-1,1)*standard_error*qnorm(0.975)
```

```
## Warning in c(-1, 1) * standard_error: Recycling array of length 1 in vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```r
confidence_interval
```

```
## [1] 0.03996984 0.60738939
```