F79MA: Assessed Project 1

Amit Parekh | H00184445

Semester 1, 2017

1 Executive Summary

Since $\tilde{p}$ may be estimated to be 0, using an MLE is more beneficial provided the number of trials conducted in a set is large. With at least 20 trials conducted, the MLE is also closer to the MME.

Using my proposal also ensures that a likelihood of a bumper failing the safety test is always found. Provided the probability of a bumper failing a test is above 0.05, the average number of tests needed to find the exact probability will be close to the number of trials within a set of tests, so large quantities of repeated sets of tests are less likely to be necessary.

I recommend conducting 20 to 40 trials, per set of tests. This is not a substantially large number and ensure the costs incurred are not too great, compared to if sets of 100 trials were conducted.

2 Estimating likelihood of a bumper failing

By conducting a set of n safety tests, the probability that a bumper may fail is p. Let X denote the number of these n bumpers that fail the safety test. Define a success as a bumper failing the safety test. Therefore, X is binomially distributed as it calculates the number of successes of n independent trials with probability p. The probability can be estimated using the Method of Moments Estimator (MME) such that:

$$\tilde{p}(X) = \frac{X}{n}$$

However, if each bumper passes the test, then $p = 0$ and the number of safety tests will be to be repeated.

2.1 Calculating the Maximum Likelihood Estimator (MLE)

Alternatively, let Y denote the number of bumpers failing the safety test in the final set of n trials. Therefore, $Y \in [1, n]$ because in the final set conducted, at least one bumper always fails the test. The likelihood of each bumper failing is independent of any other bumper failing. As such, p cannot be 0.

Given that $Y = y$, let $\hat{p}$ denote the MLE $\hat{p}(y)$.

$$\begin{aligned} L(p; y) = \mathbf{P}(Y = y) &= \mathbf{P}(X = y | X > 0) \\ &= \frac{\mathbf{P}(X = y, X > 0)}{\mathbf{P}(X > 0)} \end{aligned}$$

Since $X \sim \text{Bin}(n, p)$,

$$\mathbf{P}(X > 0) = 1 - \mathbf{P}(X = 0) = 1 - (1 - p)^n$$

By substituting and simplifying, we can solve to find an equation dependent on y, n, and $\hat{p}$.

$$
\begin{aligned}
\therefore L(p) &= \frac{\mathbf{P}(X = y, X > 0)}{1 - (1-p)^n} \\
&= \frac{\binom{n}{y} p^y (1-p)^{n-y}}{1 - (1-p)^n} \\
l(p) &= \log(p^y) + \log((1-p)^{n-y}) - \log(1 - (1-p)^n) \\
&= y \log(p) + (n-y) \log(1-p) - \log(1 - (1-p)^n) \\
l'(p) &= \frac{y}{p} - \frac{n-y}{1-p} - \frac{n(1-p)^{n-1}}{1-(1-p)^n} \\
\therefore 0 &= \frac{y}{\hat{p}} - \frac{n-y}{1-\hat{p}} - \frac{n(1-\hat{p})^{n-1}}{1-(1-\hat{p})^n} \\
\frac{n(1-\hat{p})^{n-1}}{1-(1-\hat{p})^n} &= \frac{y}{\hat{p}} - \frac{n-y}{1-\hat{p}} \\
&= \frac{y(1-\hat{p}) - \hat{p}(n-y)}{\hat{p}} \\
n\hat{p}(1-\hat{p})^n &= (1-(1-\hat{p})^n)(y(1-\hat{p}) - \hat{p}(n-y)) \\
&= (1-(1-\hat{p})^n)(y - n\hat{p}) \\
&= y - y(1-\hat{p})^n - n\hat{p} + n\hat{p}(1-\hat{p})^n \\
\therefore y &= n\hat{p} + y(1-\hat{p})^n
\end{aligned}
$$

2.2 Comparing MME to MLE

Substituting j for y in both $\tilde{p}(y)$ and $\hat{p}(y)$

$$\tilde{p}(j) = \frac{j}{n} \Rightarrow j = n\tilde{p}$$

$$
\begin{aligned}
j &= n\hat{p} + j(1-\hat{p})^n \\
n\hat{p} &= j - j(1-\hat{p})^n \\
&= j(1-(1-\hat{p})^n) \\
\therefore j &= \frac{n\hat{p}}{1-(1-\hat{p})^n}
\end{aligned}
$$

Equating the values of j,

$$
\begin{aligned}
n\tilde{p} &= \frac{n\hat{p}}{1-(1-\hat{p})^n} \\
\tilde{p} &= \frac{\hat{p}}{1-(1-\hat{p})^n} \\
\therefore \tilde{p}(j) &\geq \frac{\hat{p}(j)}{1-(1-\hat{p}(j))^n}
\end{aligned}
$$

From the final line, it's clear that any value of $\tilde{p}$ is greater than or equal to the RHS because the RHS is always divided by a positive number since p is always going to be positive $\forall j = 1, 2, \ldots, n$.

2.3 Solving the MLE

The MLE can be solved using the Newton-Raphson method, which involves iterating through values of $\hat{p}(y)$ until it is within a given tolerance level. By setting the tolerance to 0.0000001, an extremely small number, it implies that the estimated guess of $\hat{p}(y)$ is more accurate.

Using the Newton-Raphson method and when $n = 10$ and $y = 2$, meaning that there are 2 bumpers that fail the test out of 10 trials, $\hat{p}(y) = 0.2173$.

Further, using the Newton-Raphson method to calculate the MLE and MME for different values of n and y can be illustrated using Figure 1 below.

Figure 1: Estimated probability for different numbers of trials within a set of safety tests

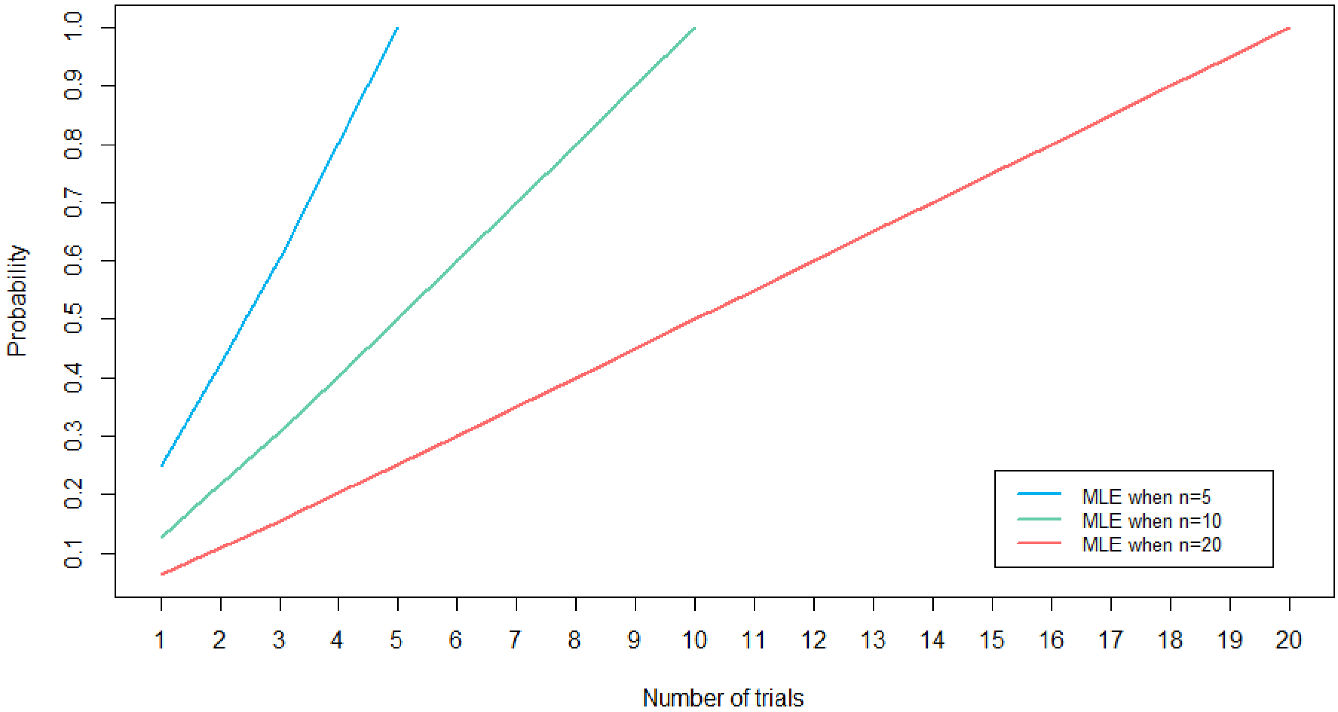

Figure 1 shows that for larger values of n, the MLE is closer related to the MME. This is because the MLE is unreliable for small samples. Therefore when estimating with small values of n, $\hat{p}(y)$ may not be an accurate estimate of p. A small sample size means that there are not enough sample points to accurately check the variability of the data.

2.4 Distribution of the number of high-impact safety tests carried out

Let Z be the distribution of the number of high-impact safety tests carried out. Using simulations in R with different values of n and p, Z can be illustrated by Figure's 2 to 7.

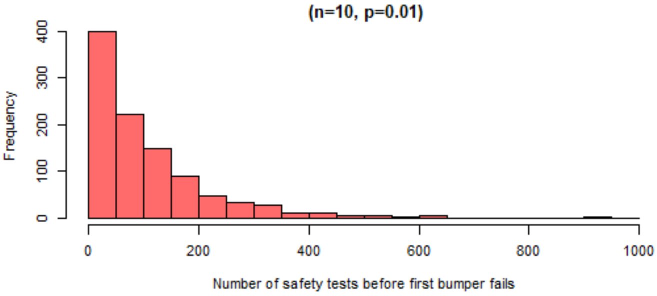

Figure 2: Distribution of the number of high-impact safety tests carried out until one failure (n=10, p=0.01)

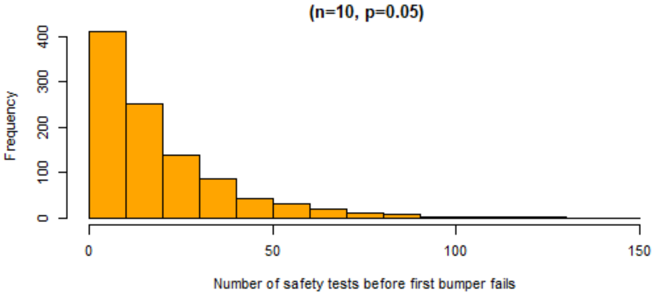

Figure 3: Distribution of the number of high-impact safety tests carried out until one failure (n=10, p=0.05)

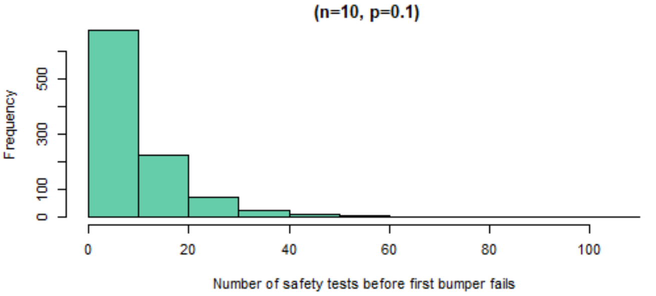

Figure 4: Distribution of the number of high-impact safety tests carried out until one failure (n=10, p=0.1)

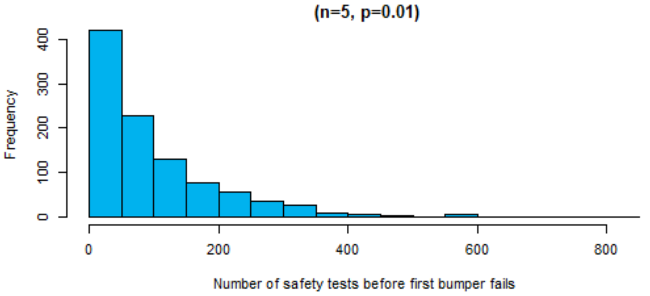

Figure 5: Distribution of the number of high-impact safety tests carried out until one failure (n=5, p=0.01)

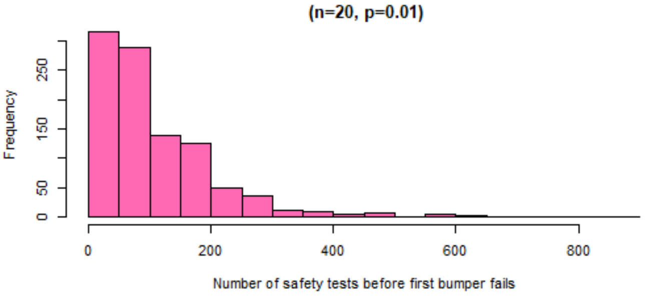

Figure 6: Distribution of the number of high-impact safety tests carried out until one failure (n=20, p=0.01)

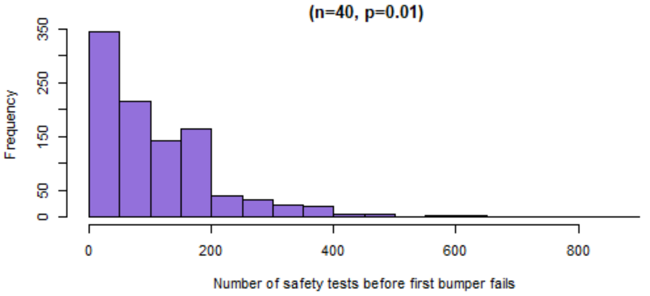

Figure 7: Distribution of the number of high-impact safety tests carried out until one failure (n=40, p=0.01)

The histograms indicate that Z is likely to be geometrically distributed.

Each histogram is positively skewed and only has one tail. This shows that for smaller values of p, the number of trials conducted is larger. This is because p indicates the probability of a bumper failing the tests.

As n gets larger, the frequency of trials conducted before one bumper fails the tests for any given n gets much smaller as a bumper is more likely to fail before repeating a set of n tests over again. This is prominent by comparing Figure 5 with Figure 7, as the number of tests conducted in total is also smaller.

The more trials carried out, the higher the cost incurred by the company as for every bumper tested, the bumper is then discarded wasted. Additionally, conducting the tests requires labour to oversee the tests and report on findings, and if any issues occur.

2.4.1 Calculating the average number of tests conducted

Let $\bar{z}$ be the sample mean of the Z distribution. Therefore, the average number of tests conducted is shown by the table below.

n values	p values			
	0.01	0.025	0.05	0.075
5	$\bar{Z} = 108.26$	$\bar{Z} = 41.48$	$\bar{Z} = 21.6$	$\bar{Z} = 15.65$
10	$\bar{Z} = 102.76$	$\bar{Z} = 45.25$	$\bar{Z} = 25.27$	$\bar{Z} = 18.23$
25	$\bar{Z} = 110.88$	$\bar{Z} = 51.1$	$\bar{Z} = 34.875$	$\bar{Z} = 29.52$
40	$\bar{Z} = 122.04$	$\bar{Z} = 62.48$	$\bar{Z} = 44.68$	$\bar{Z} = 42$
80	$\bar{Z} = 143.26$	$\bar{Z} = 92.32$	$\bar{Z} = 81.52$	$\bar{Z} = 80.16$
160	$\bar{Z} = 201.12$	$\bar{Z} = 162.72$	$\bar{Z} = 160.32$	$\bar{Z} = 160$

By running simulations in R, we can investigate the effect that different values of n has on the total average number of high-impact safety tests conducted. For smaller values of n, we can see that, on average, fewer tests are conducted. As p gets larger, on average, more tests are conducted.

Interestingly, as n gets larger, the average number of tests conducted tends to the value of n. As both n and p get larger, the sample mean seems to be more likely to be close to n. This shows that the expected number of trials that will be conducted before a bumper fails is consistent with the number of trials conducted.

3 Appendix

3.1 Notes about code

Each section of code should be run independently of another.

I have use magrittr in some scripts, so if it doesn't work, check to see if the magrittr package is installed. You can run the code below to install and use magrittr prior to running a script.

```
install.packages("magrittr")
library("magrittr")
```

3.2 Estimating $\hat{p}(y)$ for given values of n and y

```
f.mle <- function(x,n,p) { n*p - x*(1-p)^n - x }
df.mle <- function(x,n,p) { n + x*n*(1-p)^(n-1) }

pHat <- 0 #inital guess for p
n <- 10 # for n trials trials
y <- 2 # assuming y fails out of n trials

err <- f.mle(y,n,pHat) # calculate error

while (abs(err) > 0.0000001) {
    pHat <- pHat - err/df.mle(y,n,pHat)
    err <- f.mle(y,n,pHat)
}
```

3.3 Estimating $\hat{p}(y)$ for different values of n and y

```
# set functions of MLE and MME
f.mle <- function(x,n,p) { n*p - x*(1-p)^n - x }
df.mle <- function(x,n,p) { n + x*n*(1-p)^(n-1) }
f.mme <- function(x,n,p) { x/n }

pHat <- 0.2 # set inital guess for pHat
pTilde <- pHat # set initial guess for pTilde
T <- 0.000001 #set tolerance

no <- c(5,10,20) # number of trials
n <- no %>% length() %>% no[.] # set n as the largest number of trials

lst <- list(numeric(5), numeric(10), numeric(20)) # initialise objects in a list

# calculate mle for different values of n
mle <- 0
for (n in no) {
    for (y in 1:n) {
        err <- f.mle(y,n,pHat) # calculate error

        while (abs(err) > T) {
            pHat <- pHat - err/df.mle(y,n,pHat)
            err <- f.mle(y,n,pHat)
        }

        mle[y] <- pHat
    }
    index <- which(no == n)

    lst[[index]] <- mle
}

# plot lines on graph

lines.col <- c("deepskyblue2", "mediumaquamarine", "indianred1")

plot (1:n,
      lst[[3]],
      xaxt='n',
      yaxt='n',
      type='l',
      col= lines.col[3],
      lwd=2,
      main = paste('Figure 1: Estimated probability for different numbers',
                   'of trials within a set of safety tests'),
      xlab = 'Number of trials',
      ylab = 'Probability')

for (i in 1:(length(no)-1)) {
    lines (1:(no[i]), (lst[[i]]), col=lines.col[i], lwd=2)
}

legend( "bottomright",
        inset=.05,
        legend=c("MLE when n=5", "MLE when n=10", "MLE when n=20"),
        col= lines.col,
        lty=1,
        lwd=2,
        cex=0.8)

axis(1, at=seq(1, no[3],by=1)); axis(2, at=seq(0,1,by=0.1))
```

3.4 Illustrating the distribution of Z, for some values of p and n

```
findTested <- function(j,p,n) {

    tested <- 0 # initialise vector

    for (k in 1:j) {
        tests <- 0
        i <- 0

        while (i == 0) {
            x <- rbinom(n,1,p)
            if ( sum(x) >= 1 ) {
                i <- 1
            }
            tests <- tests + n
        }
        tested[k] <- tests
    }
    tested
}

k <- 1000 # number of total trials

var <- list( c(0.01,10),
             c(0.05, 10),
             c(0.1, 10),
             c(0.01, 5),
             c(0.01, 10),
             c(0.01, 20) )

a <- findTested(k, var[[1]][1], var[[1]][2])
b <- findTested(k, var[[2]][1], var[[2]][2])
c <- findTested(k, var[[3]][1], var[[3]][2])
d <- findTested(k, var[[4]][1], var[[4]][2])
e <- findTested(k, var[[5]][1], var[[5]][2])
f <- findTested(k, var[[6]][1], var[[6]][2])

list <- list(a,b,c,d,e,f)

par(mfrow=c(3,2))

for (i in 1:6) {
    tested <- list[[i]]

    dist.xlab <- "Number of sets of n safety tests carried out";
    dist.ylab <- "Frequency";
    dist.title <- "Distribution of the number of high-impact safety tests carried out
                   until one failure \n ";
    dist.n <- var[[i]][2]
    dist.p <- var[[i]][1]
    dist.main <- paste(dist.title, "(n=", dist.n, ", p=", dist.p, ")", sep = "")

    w <- seq(0, max(tested)+(1/dist.p), by=(1/dist.p))

    hist(tested, xlab=dist.xlab, ylab=dist.ylab, main=dist.main, breaks=w )
}
```

3.5 Calculating the sample mean with varied n and p values

```r
k <- 1000 # number of total trials
tested <- 0 # initialise vector
m <- 0

no <- c(5, 10, 25, 40, 80, 160)

p <- 0.075 # probabilty of bumper failing

for (b in 1:length(no)) {
    n <- no[b]

    for (k in 1:k) {
        tests <- 0

        i <- 0

        while (i == 0) {
            x <- rbinom(n,1,p)
            if ( sum(x) >= 1 ) {
                i <- 1
            }
            tests <- tests + n
        }
        tested[k] <- tests
    }

    m[b] <- mean(tested)
}; m
```