Project 2 - Alon Goodman & Ran Hassid

Q1

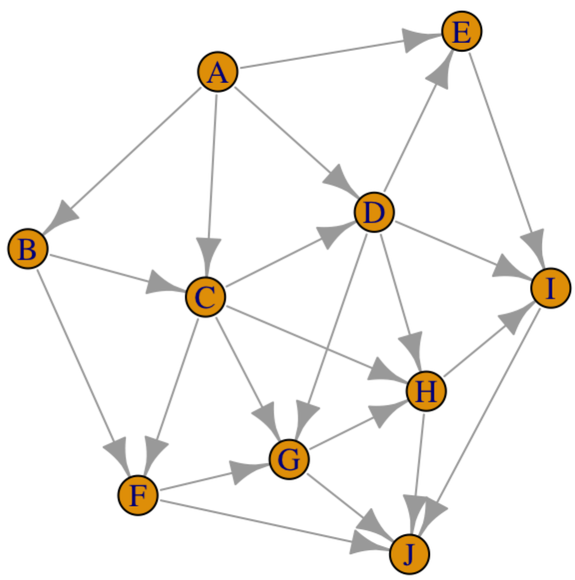
```r
#install.packages("igraph")
library(igraph)
```

```
##
## Attaching package: 'igraph'
```

```
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
```

```
## The following object is masked from 'package:base':
##
##     union
```

The System



a

The distribution of the time of until an edge will not working is exp(theta).

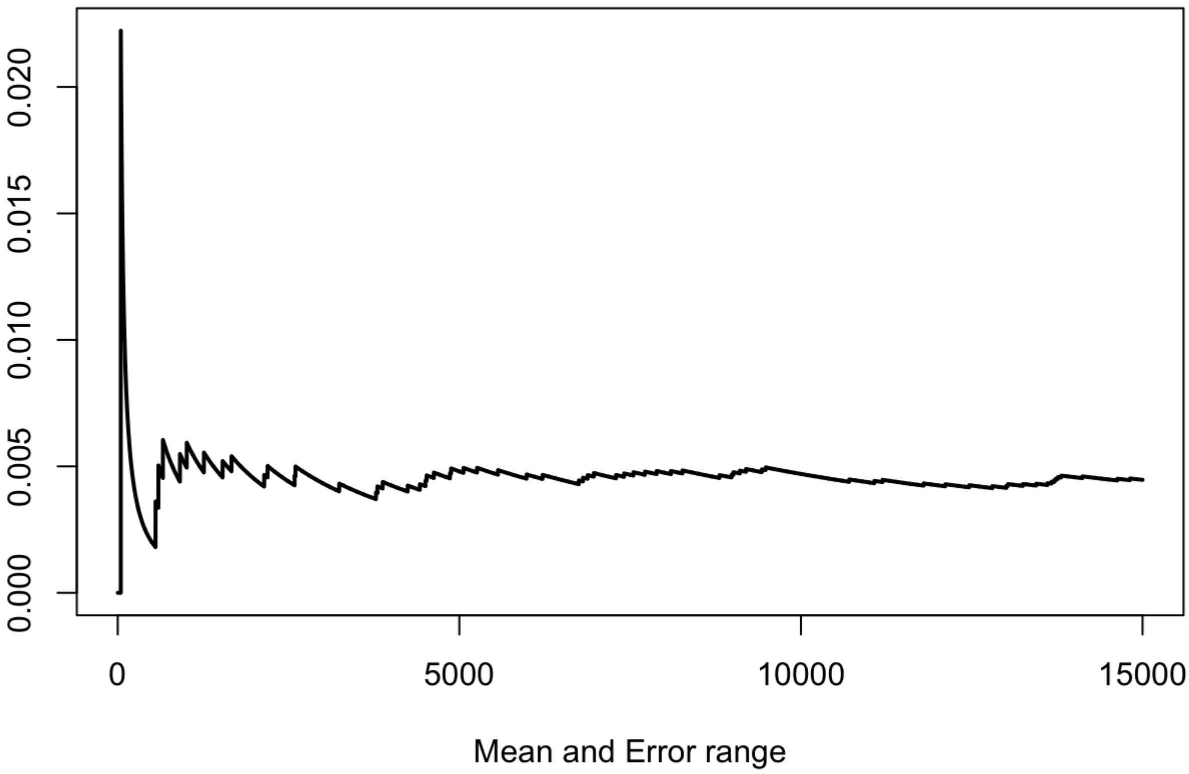
So, the probability that an edge will not work after 10 days:

P(exp(theta)<10) = 1 - e^(-10 * theta)

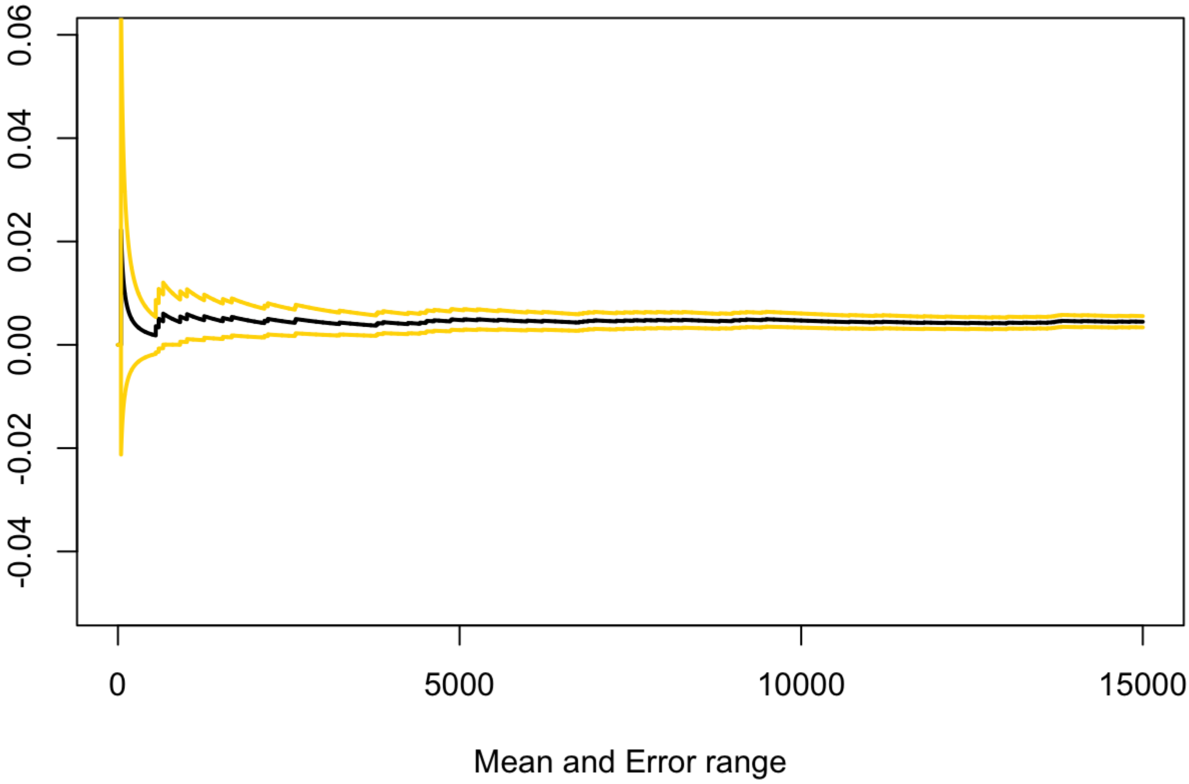
```
theta <- 0.02

h_x.result_vec.a <- numeric(15000)

for (i in 1:15000) {
  graph.zero_day.i <- make_graph(c("A", "B", "A", "C","A", "D","A", "E",
                              "B","C","B","F",
                              "C","F","C","G","C","D","C","H",
                              "D","E","D","H","D","I","D","G",
                              "E","I",
                              "F","G","F","J",
                              "G","H","G","J",
                              "H","I","H","J",
                              "I","J"), directed = TRUE)
  after_10_days.i <- rbinom(n = 22,size = 1,prob = 1-exp(-10*theta))
  graph_after_10_days.i <- delete.edges(graph = graph.zero_day.i,edges = which(after_
10_days.i==1))
  paths_A_J_after_10_days.i <- all_simple_paths(graph = graph_after_10_days.i,from =
"A","J")
  if(identical(paths_A_J_after_10_days.i,list())){
    h_x.result_vec.a[i] <- 1
  }

}
```

```
## Expected value estimator = 0.0045
## Varinace estimator = 0.0044
```

```
est.int <- cumsum(h_x.result_vec.a)/(1:15000)
est.err <- sqrt( cumsum( (h_x.result_vec.a-est.int)^2 )) / (1:15000)

plot(est.int, xlab = "Mean and Error range", ylab = "", type = "l", lwd = 2)
```




```
plot(est.int, xlab = "Mean and Error range", ylab = "", type = "l", lwd = 2, ylim = m
ean(h_x.result_vec.a)+100*est.err[15000]*c(-1,1))
lines(est.int + 2*est.err, col = "gold", lwd = 2)
lines(est.int - 2*est.err, col = "gold", lwd = 2)
```

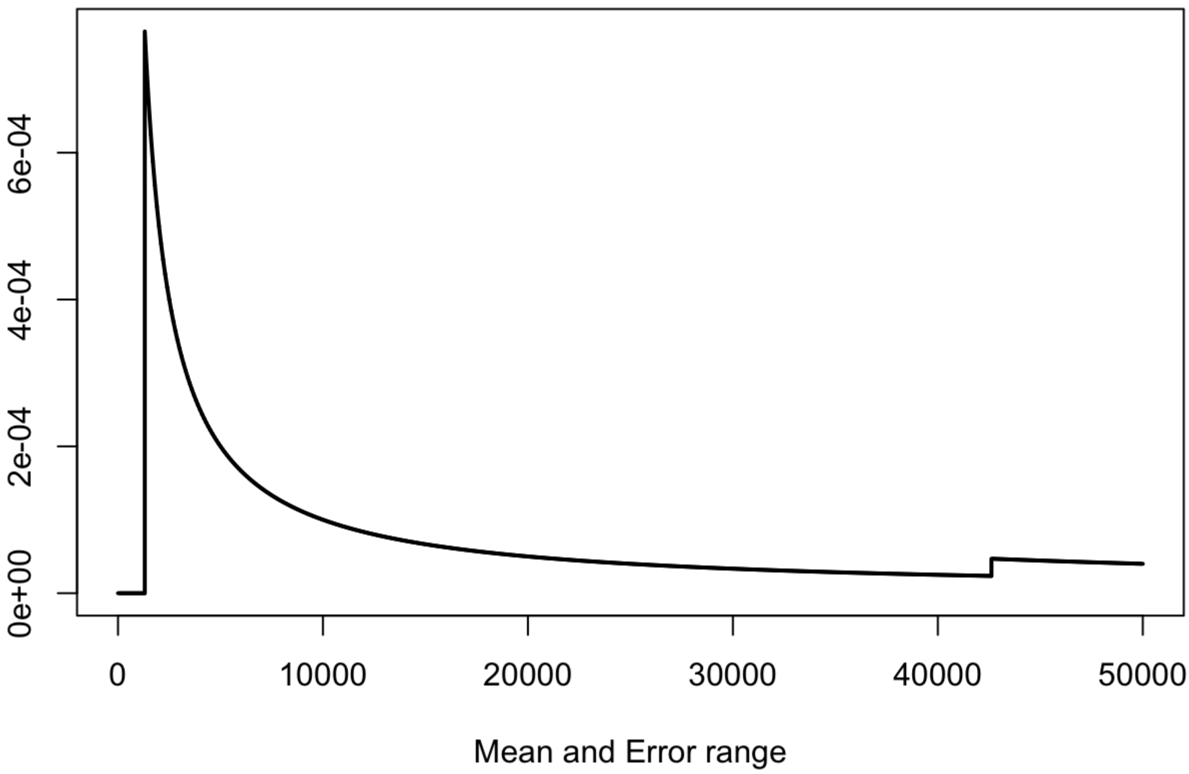


b

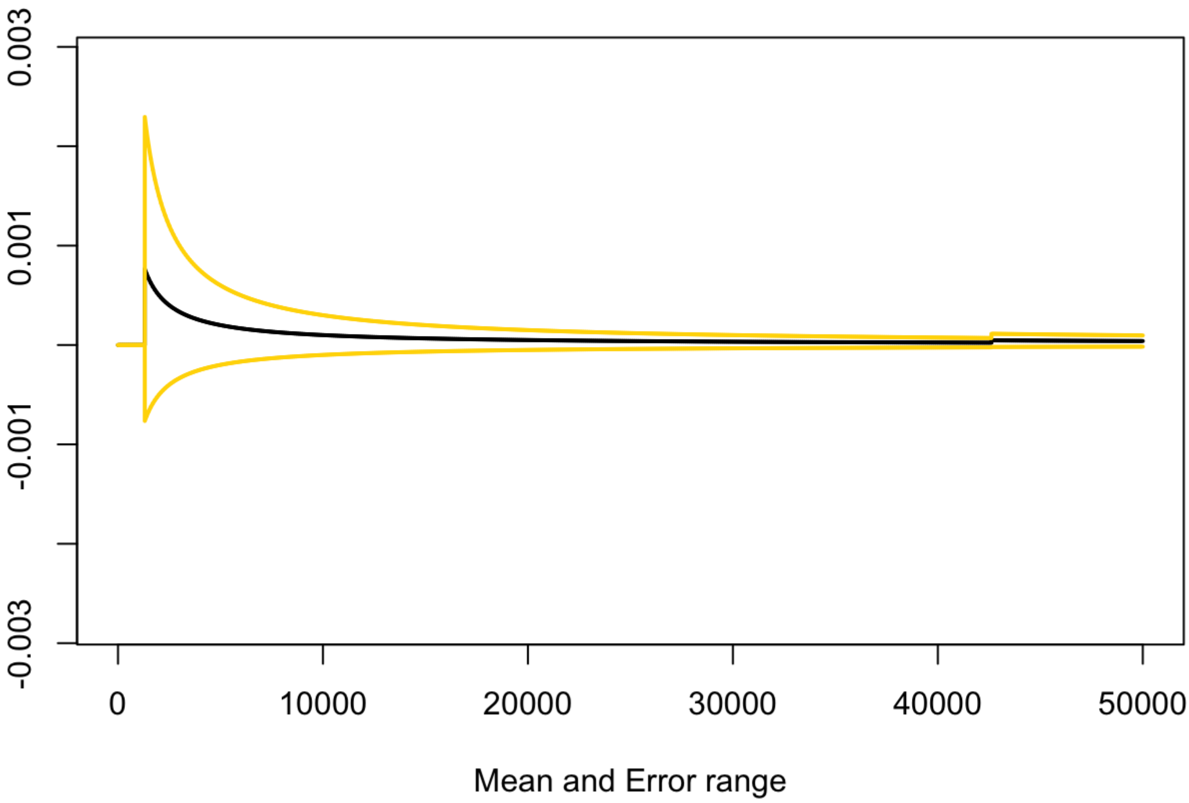
```
theta <- 0.005

h_x.result_vec.b <- numeric(50000)

for (i in 1:50000) {
  graph.zero_day.i <- make_graph(c("A", "B", "A", "C","A", "D","A", "E",
                            "B","C","B","F",
                            "C","F","C","G","C","D","C","H",
                            "D","E","D","H","D","I","D","G",
                            "E","I",
                            "F","G","F","J",
                            "G","H","G","J",
                            "H","I","H","J",
                            "I","J"), directed = TRUE)
  after_10_days.i <- rbinom(n = 22,size = 1,prob = 1-exp(-10*theta))
  graph_after_10_days.i <- delete.edges(graph = graph.zero_day.i,edges = which(after_
10_days.i==1))
  paths_A_J_after_10_days.i <- all_simple_paths(graph = graph_after_10_days.i,from =
"A","J")
  if(identical(paths_A_J_after_10_days.i,list())){
    h_x.result_vec.b[i] <- 1
  }

}
```

```
## Expected value estimator = 4e-05
## Varinace estimator = 3.99991999839997e-05
## Out of 50,000 test, the no. of failures = 2
```

```
est.int <- cumsum(h_x.result_vec.b)/(1:50000)
est.err <- sqrt( cumsum( (h_x.result_vec.b-est.int)^2 )) / (1:50000)

plot(est.int, xlab = "Mean and Error range", ylab = "", type = "l", lwd = 2)
```




```
plot(est.int, xlab = "Mean and Error range", ylab = "", type = "l", lwd = 2, ylim = m
ean(h_x.result_vec.b)+100*est.err[50000]*c(-1,1))
lines(est.int + 2*est.err, col = "gold", lwd = 2)
lines(est.int - 2*est.err, col = "gold", lwd = 2)
```

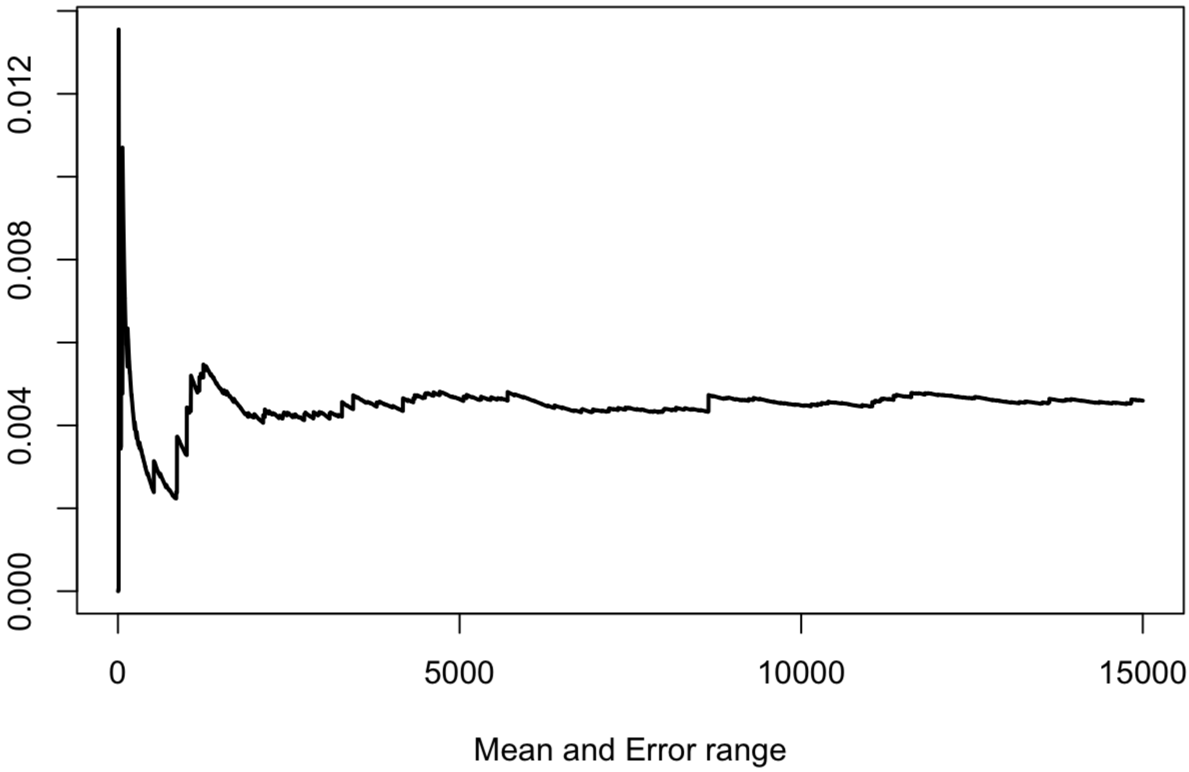


C

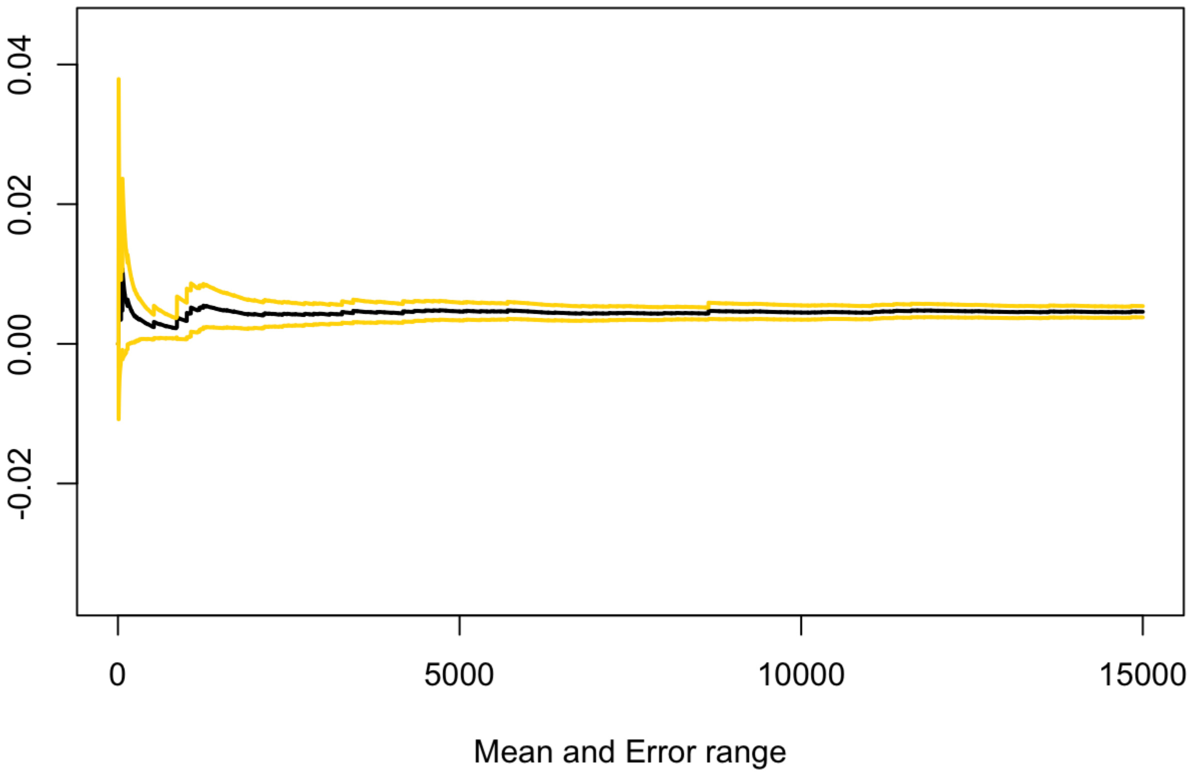
```
theta.f <- 0.02
theta.g <- 0.05

h_x.result_vec.c <- numeric(15000)

for (i in 1:15000) {
  graph.zero_day.i <- make_graph(c("A", "B", "A", "C","A", "D","A", "E",
                              "B","C","B","F",
                              "C","F","C","G","C","D","C","H",
                              "D","E","D","H","D","I","D","G",
                              "E","I",
                              "F","G","F","J",
                              "G","H","G","J",
                              "H","I","H","J",
                              "I","J"), directed = TRUE)
  after_10_days.i <- rbinom(n = 22,size = 1,prob = 1-exp(-10*theta.g))
  graph_after_10_days.i <- delete.edges(graph = graph.zero_day.i,edges = which(after_
10_days.i==1))
  paths_A_J_after_10_days.i <- all_simple_paths(graph = graph_after_10_days.i,from =
"A","J")
  if(identical(paths_A_J_after_10_days.i,list())){
    h_x.result_vec.c[i] <- 1*prod( dbinom(x = after_10_days.i,size = 1,prob = 1-exp(-
10*theta.f)) / dbinom(x = after_10_days.i,size = 1,prob = 1-exp(-10*theta.g)))
  }

}
```

```
## Expected value estimator = 0.0046
## Varinace estimator = 0.0024
```

```
est.int <- cumsum(h_x.result_vec.c)/(1:15000)
est.err <- sqrt( cumsum( (h_x.result_vec.c-est.int)^2 )) / (1:15000)

plot(est.int, xlab = "Mean and Error range", ylab = "", type = "l", lwd = 2)
```




```
plot(est.int, xlab = "Mean and Error range", ylab = "", type = "l", lwd = 2, ylim = m
ean(h_x.result_vec.c)+100*est.err[15000]*c(-1,1))
lines(est.int + 2*est.err, col = "gold", lwd = 2)
lines(est.int - 2*est.err, col = "gold", lwd = 2)
```

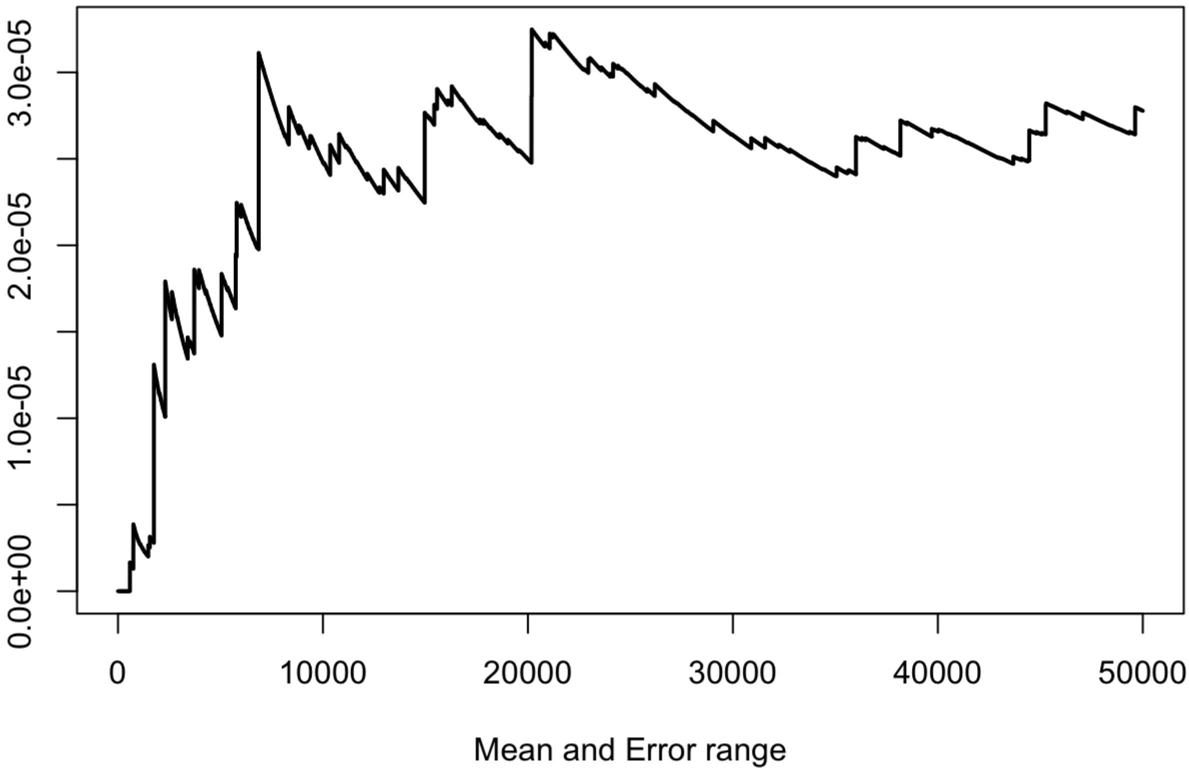


d

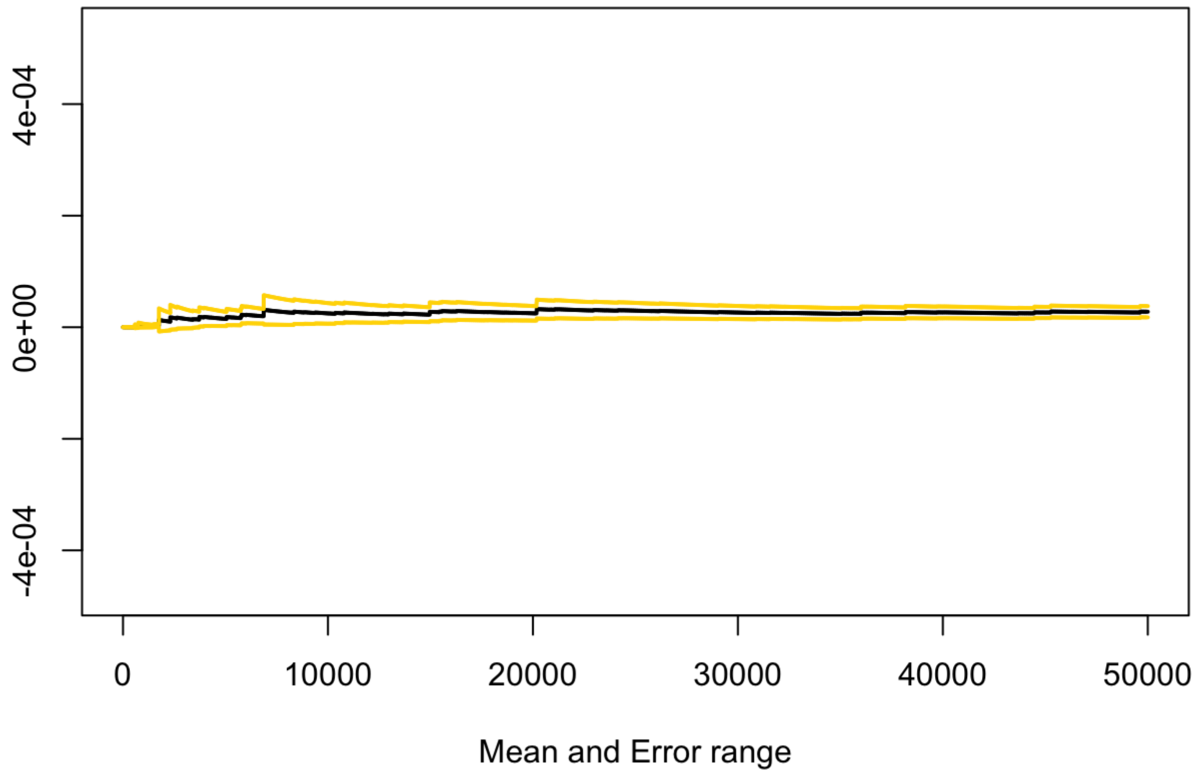
```
theta.f <- 0.005
theta.g <- 0.02

h_x.result_vec.d <- numeric(50000)

for (i in 1:50000) {
  graph.zero_day.i <- make_graph(c("A", "B", "A", "C","A", "D","A", "E",
                           "B","C","B","F",
                           "C","F","C","G","C","D","C","H",
                           "D","E","D","H","D","I","D","G",
                           "E","I",
                           "F","G","F","J",
                           "G","H","G","J",
                           "H","I","H","J",
                           "I","J"), directed = TRUE)
  after_10_days.i <- rbinom(n = 22,size = 1,prob = 1-exp(-10*theta.g))
  graph_after_10_days.i <- delete.edges(graph = graph.zero_day.i,edges = which(after_
10_days.i==1))
  paths_A_J_after_10_days.i <- all_simple_paths(graph = graph_after_10_days.i,from =
"A","J")
  if(identical(paths_A_J_after_10_days.i,list())){
    h_x.result_vec.d[i] <- 1*prod( dbinom(x = after_10_days.i,size = 1,prob = 1-exp(-
10*theta.f)) / dbinom(x = after_10_days.i,size = 1,prob = 1-exp(-10*theta.g)))
  }

}
```

```
## Expected value estimator = 2.7776796716378e-05
## Varinace estimator = 1.27083955599586e-06
```

```
est.int <- cumsum(h_x.result_vec.d)/(1:50000)
est.err <- sqrt( cumsum( (h_x.result_vec.d-est.int)^2 )) / (1:50000)

plot(est.int, xlab = "Mean and Error range", ylab = "", type = "l", lwd = 2)
```




```
plot(est.int, xlab = "Mean and Error range", ylab = "", type = "l", lwd = 2, ylim = m
ean(h_x.result_vec.d)+100*est.err[50000]*c(-1,1))
lines(est.int + 2*est.err, col = "gold", lwd = 2)
lines(est.int - 2*est.err, col = "gold", lwd = 2)
```




```
##                                        A           B           C           D
## Expected value estimator 0.004466667 4.00000e-05 0.004596683 2.77768e-05
## Varinac value estimator  0.004447012 3.99992e-05 0.002433420 1.27084e-06
```

It can be seen that the difference between section A and section B is that when the probability of system failure is so low, it is more “expensive” to sample and get a “good” estimation for expectation. When in sections C and D we used “IS” method, we actually sampled from a similar distribution but with a larger parameter. That is, we have created a situation where there is a greater chance of seeing a system failure. As a result, the samples were “cheaper”. This can be seen by looking at the variance estimations: IS’s variance estimations are significantly smaller than the non-IS variance estimations.

Q2

a

1c

$$Var(\hat{\mu}_1) = \frac{\sigma^2}{m}$$

$$Var(\hat{\mu}_2) = \frac{\sigma^2}{m}$$

$$corr(\hat{\mu}_1, \hat{\mu}_2) = \rho$$

$$\tilde{\mu} = \frac{1}{2}(\hat{\mu}_1 + \hat{\mu}_2)$$

$$Var(\tilde{\mu}) = Var\left[\frac{1}{2}(\hat{\mu}_1 + \hat{\mu}_2)\right] = \frac{1}{4} Var(\hat{\mu}_1 + \hat{\mu}_2) =$$

$$= \frac{1}{4}\left[Var(\hat{\mu}_1) + Var(\hat{\mu}_2) + 2Cov(\hat{\mu}_1, \hat{\mu}_2)\right] =$$

$$= \frac{1}{4}\left[\frac{\sigma^2}{m} + \frac{\sigma^2}{m} + 2\rho \cdot \sqrt{Var(\hat{\mu}_1)\cdot Var(\hat{\mu}_2)}\right] =$$

$$\rho(X,Y) = \frac{Cov(X,Y)}{\sqrt{Var(X)\,Var(Y)}}$$

$$= \frac{1}{4}\left[\frac{2\sigma^2}{m} + 2\rho\sqrt{\frac{\sigma^2}{m}\cdot\frac{\sigma^2}{m}}\right] =$$

$$= \frac{1}{4}\left[\frac{2\sigma^2}{m} + \frac{2\sigma^2}{m}\rho\right] = \frac{\sigma^2 + \sigma^2\rho}{2m} = \frac{(1+\rho)\,\sigma^2}{2m}$$

b

ג.

טענה: $\underline{X} = X_1, .., X_n$ מ"מ ב"ת

$h_1, h_2 : \mathbb{R}^n \to \mathbb{R}$ כולן מונו' עולות

אזי:

$$Cov\left(h_1(\underline{X}), h_2(\underline{X})\right) = E\left[h_1(\underline{X}) \cdot h_2(\underline{X})\right] - E\left[h_1(\underline{X})\right] \cdot E\left[h_2(\underline{X})\right] \geq 0$$

הוכחה: נוכיח בדרך אינדוקציה:

1) בסיס $n=1$:

נניח ש- h_1, h_2 כולן מונו' עולות. כלומר $\forall\, x > y : h_1(x) > h_1(y)$

$\forall\, x > y : h_2(x) > h_2(y)$

לכן,

$$h_1(x) - h_1(y) > 0 \to h_1(x) > h_1(y) \to x > y \to h_2(x) > h_2(y) \to h_2(x) - h_2(y) > 0$$

$$h_2(x) - h_2(y) < 0 \to h_1(x) < h_1(y) \to x < y \to h_2(x) < h_2(y) \to h_2(x) - h_2(y) < 0$$

לכן, אלא אם $h_2(x) - h_2(y) = 0$ or $h_1(x) - h_1(y) = 0$, הביטויים יהיו שווי סימן ולכן, המכפלה חיובית.

קיבלנו שמתקיים: $[h_1(x) - h_1(y)][h_2(x) - h_2(y)] \geq 0$

2) כאשר X, Y מ"מ ב"ת ש"ה, מתקיים

$$E\left[[h_1(x) - h_1(y)][h_2(x) - h_2(y)]\right] \geq 0$$

3) ידוע לנו ש- X, Y ש"ה. לכן, מתקיים:

$$\begin{cases} E[h_1(x)] = E[h_1(y)] \\ E[h_2(x)] = E[h_2(y)] \\ E[h_1(x)\, h_2(x)] = E[h_1(y)\, h_2(y)] \end{cases}$$

$$E[h_1(x) \cdot h_2(x)] - \underbrace{E[h_1(x)\, h_2(y)]}_{\text{ב"ת } X,Y} - \underbrace{E[h_1(y)\, h_2(x)]}_{\text{ב"ת } X,Y} + E[h_1(y)\, h_2(y)] \geq 0$$

$$E[h_1(x) \cdot h_2(x)] - E(h_1(x))\, E(h_2(y)) - E[h_1(y)]\, E(h_2(x)) +$$

$$+ E[h_1(y)\, h_2(y)] \geq 0$$

$\downarrow$

$$E\left[h_1(X)\cdot h_2(x)\right]-E\left(h_1(x)\right)E\left(h_2(x)\right)-E\left[h_1(x)\right]E\left[h_2(x)\right]+$$

$$+E\left[h_1(x)h_2(x)\right]=2E\left[h_1(X)\cdot h_2(x)\right]-2E\left[h_1(x)\right]E\left(h_2(x)\right)\geq 0$$

$$E\left[h_1(X)\cdot h_2(x)\right]-E\left[h_1(x)\right]E\left(h_2(x)\right)\geq 0$$

$$\boxed{Cov\left[h_1(X),h_2(X)\right]\geq 0}$$

4) נניח כי עבור n-1 מתקיים:

$$Cov\left(h_1(\underline{x}),h_2(\underline{x})\right)\geq 0$$

נוכיח את נכונות הטענה עבור n:

כלומר, נראה:

$$E\left[h_1(\underline{x})h_2(\underline{x})\mid X_n=x_n\right]-E\left[h_1(\underline{x})\mid X_n=x_n\right]E\left[h_2(\underline{x})\mid X_n=x_n\right]\geq 0$$

כאשר אנו מתנים ב $X_n=x_n$, התוחלת מתפרקת על שאר

הרכיבים $(x_1,..,x_{n-1})$.

נסמן: $(X_1,..,X_{n-1})=\underline{X}_{(n)}$ ונקבל:

$$E\left[h_1(\underline{x})h_2(\underline{x})\mid X_n=x_n\right]=E\left[h_1(\underline{x}_{(n)})h_2(\underline{x}_{(n)})\right]$$

$$E\left(h_1(\underline{x})\mid X_n=x_n\right)=E\left(h_1(\underline{X}_{(n)})\right)$$

$$E\left(h_2(\underline{x})\mid X_n=x_n\right)=E\left(h_2(\underline{X}_{(n)})\right)$$

$$E\left[h_1(\underline{x})h_2(\underline{x})\mid X_n=x_n\right]-E\left[h_1(\underline{x})\mid X_n=x_n\right]E\left[h_2(\underline{x})\mid X_n=x_n\right]=$$

$$=E\left[h_1(\underline{x}_{(n)})h_2(\underline{x}_{(n)})\right]-E\left(h_1(\underline{X}_{(n)})\right)E\left(h_2(\underline{X}_{(n)})\right)=$$

$$=Cov\left(h_1(\underline{X}_{(n)}),h_2(\underline{X}_{(n)})\right)\geq 0$$

הנחת האינדוקציה

לכן, הטענה נכונה.

C

ג'

תהי h פונ' מונ' בכל רכיב.

יהיו $U_1,...,U_n$ מ"מ ב"ת מהתפלגות $U(0,1)$.

נניח h מונ' עולה:

אם $h(u_1,..u_n)$ עולה בכל רכיב, $h((1-u_1),..,(1-u_n))$
יורדת בכל רכיב ולכן $-h((1-u_1),..,(1-u_n))$ עולה.

$$cov(h(u_1,..,u_n), -h((1-u_1),..,(1-u_n))) \geq 0$$

הוכחנו בסעיף ב' כי שהטענה נכונה עבור פונ' מונ' עולות.

$$-1 \cdot cov(h(u_1,..,u_n), h((1-u_1),..,(1-u_n))) \geq 0$$

$$cov(h(u_1,..,u_n), h((1-u_1),..,(1-u_n))) \leq 0$$

כנדרש.

עבור h מונ' יורדת:

אם $h(u_1,..u_n)$ יורדת בכל רכיב, $h((1-u_1),..,(1-u_n))$
עולה בכל רכיב, נשים לב שמתקיים $-h(u_1,..u_n)$ עולה.

$$cov(-h(u_1,..,u_n), h((1-u_1),..,(1-u_n))) \geq 0$$

הוכחנו בסעיף ב' כי שהטענה נכונה עבור פונ' מונ' עולות.

$$-1 \cdot cov(h(u_1,..,u_n), h((1-u_1),..,(1-u_n))) \geq 0$$

$$cov(h(u_1,..,u_n), h((1-u_1),..,(1-u_n))) \leq 0$$

כנדרש.

d

d.1

h(u) = -log(1-u)/theta

u gets values between 0 and 1.

So, (1-u) gets smaller values as u increases.

Giving a small value to a log function gets a smaller value and multiplying by minus 1/theta gives us a larger value (we know theta is positive).

Therefore, our function is a monotonically increasing function in u values.

```r
theta <- 0.02

h_x.result_vec.2.d <- numeric(15000)

for (i in 1:15000) {
  graph.zero_day.i <- make_graph(c("A", "B", "A", "C","A", "D","A", "E",
                           "B","C","B","F",
                           "C","F","C","G","C","D","C","H",
                           "D","E","D","H","D","I","D","G",
                           "E","I",
                           "F","G","F","J",
                           "G","H","G","J",
                           "H","I","H","J",
                           "I","J"), directed = TRUE)
  u_vec <- runif(22)
  after_10_days.i <- -log(1-u_vec)/theta # we sample from the exp(theta) distribution
using the inverse function
  graph_after_10_days.i <- delete.edges(graph = graph.zero_day.i,edges = which(after_
10_days.i<=10))
  paths_A_J_after_10_days.i <- all_simple_paths(graph = graph_after_10_days.i,from =
"A","J")
  if(identical(paths_A_J_after_10_days.i,list())){
    h_x.result_vec.2.d[i] <- 1
  }

}
```

d.2

we will use antithetic sampling by using the following h functions:

- h(u) = -log(1-u)/theta - Monotonically increasing function in u values.
- h(1-u) = -log(1-(1-u))/theta = -log(u)/theta - Monotonically decreasing function in u values.

```
theta <- 0.02

h_x.result_vec.2.d.inc <- numeric(15000)
h_x.result_vec.2.d.dec <- numeric(15000)

for (i in 1:15000) {
  graph.zero_day.i <- make_graph(c("A", "B", "A", "C","A", "D","A", "E",
                           "B","C","B","F",
                           "C","F","C","G","C","D","C","H",
                           "D","E","D","H","D","I","D","G",
                           "E","I",
                           "F","G","F","J",
                           "G","H","G","J",
                           "H","I","H","J",
                           "I","J"), directed = TRUE)
  u_vec <- runif(22)

  after_10_days.i.inc <- -log(1-u_vec)/theta
  after_10_days.i.dec <- -log(u_vec)/theta

  graph_after_10_days.i.inc <- delete.edges(graph = graph.zero_day.i,edges = which(af
ter_10_days.i.inc<=10))
  graph_after_10_days.i.dec <- delete.edges(graph = graph.zero_day.i,edges = which(af
ter_10_days.i.dec<=10))

  paths_A_J_after_10_days.i.inc <- all_simple_paths(graph = graph_after_10_days.i.in
c,from = "A","J")
  paths_A_J_after_10_days.i.dec <- all_simple_paths(graph = graph_after_10_days.i.de
c,from = "A","J")


  if(identical(paths_A_J_after_10_days.i.inc,list())){
    h_x.result_vec.2.d.inc[i] <- 1
  }
  if(identical(paths_A_J_after_10_days.i.dec,list())){
    h_x.result_vec.2.d.dec[i] <- 1
  }

}
```

```
A_S_expected_value_estimator <- 0.5*(mean(h_x.result_vec.2.d.inc) + mean(h_x.result_v
ec.2.d.dec))
A_S_varinace_estimator <- (var(h_x.result_vec.2.d.inc) + var(h_x.result_vec.2.d.dec)
 + 2*cov(h_x.result_vec.2.d.inc,h_x.result_vec.2.d.dec))/4
```

```
## Antithetic sampling expected value estimator = 0.0058
## Antithetic sampling varinace estimator = 0.0029
```

Comparison between Antithetic sampling estimator and Importance sampling:

```
## Antithetic sampling expected value estimator = 0.0058
## Antithetic sampling varinace estimator = 0.0029
## Importance sampling expected value estimator = 0.0045
## Importance sampling varinace estimator = 0.0044
```

We can see we got similar expected value estimator, but smaller varince for the Antithetic sampling estimator.

d.3

Antithetic sampling estimator

```
x_norm_vec <- rnorm(10000)
h_x_plus <- (exp(x_norm_vec) + 7)^(x_norm_vec/3)
h_x_minus <- (exp(-x_norm_vec) + 7)^(-x_norm_vec/3)

A_S_expected_value_estimator_d3 <- 0.5*(mean(h_x_plus)+mean(h_x_minus))
A_S_varinace_estimator_d3 <- (var(h_x_plus) + var(h_x_minus) + 2*cov(h_x_plus,h_x_min
us)) / 4
```

Monte Carlo estimator

```
h_x_monte_carlo <- (exp(x_norm_vec) + 7)^(x_norm_vec/3)

MC_expected_value_estimator_d3 <- mean(h_x_monte_carlo)
MC_varinace_estimator_d3 <- var(h_x_monte_carlo)
```

```
## Antithetic sampling expected value estimator = 1.574
## Antithetic sampling varinace estimator = 11.4832
## Monte Carlo expected value estimator = 1.5725
## Monte Carlo varinace estimator = 24.5819
```

```
## We got that the variance of antithetic sampling estimator is = 2.1 times smaller t
han the monte carlo estimator
```