# Function for toss coin

library(ggplot2)

coinTossOutcomes <- function(N, p) {

# Define the sample space and probabilities

sample\_space <- c("H", "T")

probabilities <- c(p, 1 - p)

# Generate all possible outcomes for N coin tosses

all\_outcomes <- expand.grid(replicate(N, sample\_space, simplify = FALSE))

# Initialize a vector to store the probabilities of each outcome

outcome\_probabilities <- numeric(nrow(all\_outcomes))

# Calculate the probability of each outcome

for (i in 1:nrow(all\_outcomes)) {

outcome <- as.character(unlist(all\_outcomes[i, ]))

prob <- prod(ifelse(outcome == "H", probabilities[1], probabilities[2]))

outcome\_probabilities[i] <- prob

}

# Add the probabilities to the data frame

all\_outcomes$Probability <- outcome\_probabilities

# Combine the toss outcomes into a single string for easier viewing

all\_outcomes$X <- do.call(paste0, all\_outcomes[, 1:N])

# Create the final table

final\_table <- all\_outcomes[, c("X", "Probability")]

# Normalize table

return(final\_table)

}

# 1. N = 5

N <- 5

p <- 0.8

result <- coinTossOutcomes(N, p)

x\_axis = seq(0,1,0.001) ;print(x\_axis)

y = c()

for (x in x\_axis) {

events = sum(result$Probability > x) # >= (1-x)

y\_axis = log2(events)

y = c(y,y\_axis)

}

final\_result = data.frame(y,x\_axis)

final\_result$normal = final\_result$y/N

a = final\_result$normal

# 2. N = 10

rm(final\_table)

rm(result)

N <- 10

p <- 0.8

result <- coinTossOutcomes(N, p)

x\_axis = seq(0,1,0.001) ;print(x\_axis)

y = c()

for (x in x\_axis) {

events = sum(result$Probability > x) # >= (1-x)

y\_axis = log2(events)

y = c(y,y\_axis)

}

final\_result = data.frame(y,x\_axis)

final\_result$normal = final\_result$y/N

b = final\_result$normal

# 3. N = 15

rm(final\_table)

rm(result)

N <- 15

p <- 0.8

result <- coinTossOutcomes(N, p)

x\_axis = seq(0,1,0.001) ;print(x\_axis)

y = c()

for (x in x\_axis) {

events = sum(result$Probability > x) # >= (1-x)

y\_axis = log2(events)

y = c(y,y\_axis)

}

final\_result = data.frame(y,x\_axis)

final\_result$normal = final\_result$y/N

c = final\_result$normal

# 4. N = 20

rm(final\_table)

rm(result)

N <- 20

p <- 0.8

result <- coinTossOutcomes(N, p)

x\_axis = seq(0,1,0.001) ;print(x\_axis)

y = c()

for (x in x\_axis) {

events = sum(result$Probability > x) # >= (1-x)

y\_axis = log2(events)

y = c(y,y\_axis)

}

final\_result = data.frame(y,x\_axis)

final\_result$normal = final\_result$y/N

d = final\_result$normal

# 3. Combine all

df <- data.frame(x\_axis = final\_result$x\_axis, "5" = a, "10" = b, "20" = c, "40" = d)

# Reshape the data to long format

df\_long <- tidyr::gather(df, variable, value, -x\_axis)

# Generate the plot

ggplot(df\_long, aes(x = x\_axis, y = value, color = variable)) +

geom\_line() +

labs(title = "Q3.e) Essential Bits given delta",

x = "delta",

y = "Normalised Essential Bit Content") +

scale\_color\_discrete(name = "N")