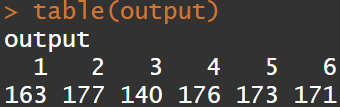
1. 首先請用runif 產生0-1之間的隨機數值,接著寫出一個自訂函數,其功能為模擬一個公正的六面骰子輸出函數的輸出結果為1-6的數值,且機率皆相同。請用此函數模擬丟骰子1000次, 並紀錄1-6出現的次數





Code:

## 1

#首先請用runif 產生0-1之間的隨機數值

runif(1,0,1)

# 接著寫出一個自訂函數,其功能為模擬一個公正的六面骰子輸出

# 函數的輸出結果為1-6的數值,且機率皆相同。

dice <- function(random){

if (random < 1 / 6) return(1)

else if (random < 2 / 6) return(2)

else if (random < 3 / 6) return(3)

else if (random < 4 / 6) return(4)

else if (random < 5 / 6) return(5)

else return(6)

}

# 請用此函數模擬丟骰子1000次

output <- c()

for(i in 1:1000){

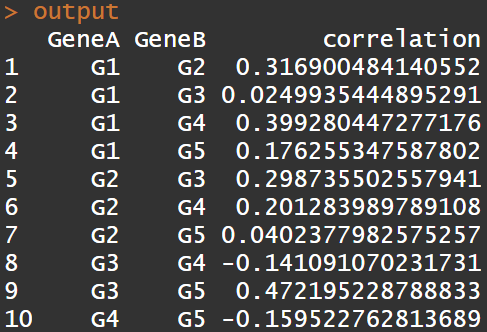
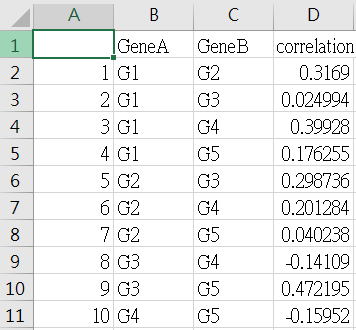
output[i] <- dice(runif(1,0,1))

}

# 並紀錄1-6出現的次數

table(output)

1. 請讀入partA\_II.csv檔案。請撰寫for 迴圈計算G1-G5兩兩間的Pearson correlation數值, 建立一個名稱為output的data.frame結構,第一個column 為Gene A的名字, 第二個column為Gene B的名字, 第三個column 為Pearson correlation 數值, 將其儲存為output.csv

Code:

## 2

# 請讀入partA\_II.csv檔案

a2 <- read.csv(file.choose())

# 請撰寫for 迴圈計算G1-G5兩兩間的Pearson correlation數值

m <- c()

for(i in 5:8){

for(j in (i+1):9){

r <- cor(a2[, i],a2[, j])

namex <- paste0("G", i-4)

namey <- paste0("G", j-4)

m <- rbind(m,c(namex,namey, r))

}

}

# 建立一個名稱為output的data.frame結構

output <- as.data.frame(m)

#第一個column 為Gene A的名字, 第二個column為Gene B的名字, 第三個column 為Pearson correlation 數值

colnames(output) <- c("GeneA", "GeneB", "correlation")

# 將其儲存為output.csv

write.csv(output, "output.csv")

1. 請讀入partB\_II.csv檔案 (檔案內部為minor allele的數目)◦ 請將樣本根據Case與Control 分群後計算 G1-G10分別在Additive model, Dominant model, Recessive model 下哪個Gene的Fisher exact test p-value 最顯著

Additive model



Dominant model



Recessive model



Code:

## 3

# 請讀入partB\_II.csv檔案 (檔案內部為minor allele的數目)

b2 <- read.csv(file.choose())

# 請將樣本根據Case與Control 分群後計算 G1-G10分別在Additive model, Dominant model,Recessive model 下哪個Gene的Fisher exact testp-value 最顯著

case <- subset(b2, b2$Type=="Case")

control <- subset(b2, b2$Type=="Control")

# Additive model

pvalue\_add <- c()

for(i in 3:12){

gcase <- table(case[,i])

gcontrol <- table(control[,i])

gmatrix <- rbind(gcase, gcontrol)

pvalue\_add[i-2] <- fisher.test(gmatrix)$p.value

}

paste0("G",which(pvalue\_add==min(pvalue\_add)))

# Dominant model

pvalue\_dom <- c()

casetemp <- case

casetemp[casetemp==1] <- 0

controltemp <- control

controltemp[controltemp==1] <- 0

for(i in 3:12){

gcase <- table(casetemp[,i])

gcontrol <- table(controltemp[,i])

gmatrix <- rbind(gcase, gcontrol)

pvalue\_dom[i-2] <- fisher.test(gmatrix)$p.value

}

paste0("G",which(pvalue\_dom==min(pvalue\_dom)))

# Recessive model

pvalue\_rec <- c()

casetemp <- case

casetemp[casetemp==1] <- 2

controltemp <- control

controltemp[controltemp==1] <- 2

for(i in 3:12){

gcase <- table(casetemp[,i])

gcontrol <- table(controltemp[,i])

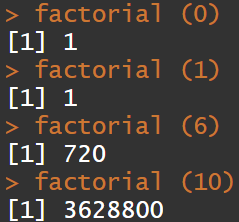
gmatrix <- rbind(gcase, gcontrol)

pvalue\_rec[i-2] <- fisher.test(gmatrix)$p.value

}

paste0("G",which(pvalue\_rec==min(pvalue\_rec)))

1. 請撰寫一個函數透過迴圈來計算數字階層



Code:

## 4 請撰寫一個函數透過迴圈來計算數字階層

factorial <- function(input){

temp <- 1

for(i in 1:input){

temp <- temp \* i

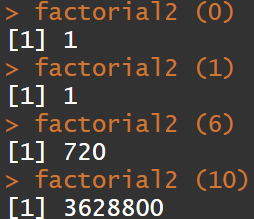
}

if (input == 0) return(1)

else return(temp)

}

1. 請撰寫一個函數透過遞迴方法來計算數字階層



Code:

## 5 請撰寫一個函數透過遞迴方法來計算數字階層

factorial2 <- function(input){

if (input == 0) return(1)

else return(input \* factorial2(input - 1))

}