Tarea 1

library (tidyr)  
library(ggplot2)  
library(ggthemes)  
require(gridExtra)

## Loading required package: gridExtra

source("MM1\_queue.R")

1. Sea el número de ’unos’ obtenido en lanzamientos de un dado honesto. Entonces tiene una distribución binomial. Calcular una tabla con los valores de la función de distribución para por dos métodos: usando la función cumsum y usando la función . También determinar cuánto vale .
   * En este caso tenemos que , se quiere obtener la distribución de dicha variable aleatoria, comencemos por obtener la función de densidad y a partir de ella obtengamos la distribución:
   * bin\_density = dbinom(0:12, 12, 1/6)  
      bin\_distribution\_1 = cumsum(bin\_density)  
      distribution <- data.frame(bin\_distribution\_1, row.names = 0:12)  
      knitr::kable(distribution)

|  |  |
| --- | --- |
|  | * + bin\_distribution\_1 |
| * + 0 | * + 0.1121567 |
| * + 1 | * + 0.3813326 |
| * + 2 | * + 0.6774262 |
| * + 3 | * + 0.8748219 |
| * + 4 | * + 0.9636500 |
| * + 5 | * + 0.9920750 |
| * + 6 | * + 0.9987075 |
| * + 7 | * + 0.9998445 |
| * + 8 | * + 0.9999866 |
| * + 9 | * + 0.9999992 |
| * + 10 | * + 1.0000000 |
| * + 11 | * + 1.0000000 |
| * + 12 | * + 1.0000000 |
|  |  |

* + Ahora, utilizando el comando de R para obtener la distribución
  + bin\_distribution\_2 = pbinom(0:12,12,1/6)  
     distribution <- data.frame(bin\_distribution\_2, row.names = 0:12)  
     knitr::kable(distribution)

|  |  |
| --- | --- |
|  | * + bin\_distribution\_2 |
| * + 0 | * + 0.1121567 |
| * + 1 | * + 0.3813326 |
| * + 2 | * + 0.6774262 |
| * + 3 | * + 0.8748219 |
| * + 4 | * + 0.9636500 |
| * + 5 | * + 0.9920750 |
| * + 6 | * + 0.9987075 |
| * + 7 | * + 0.9998445 |
| * + 8 | * + 0.9999866 |
| * + 9 | * + 0.9999992 |
| * + 10 | * + 1.0000000 |
| * + 11 | * + 1.0000000 |
| * + 12 | * + 1.0000000 |

* + Finalmente, determinamos :
  + 1 - pbinom(7, 12, 1/6)
  + ## [1] 0.0001555443
  + sum(dbinom(8:12, 12, 1/6))
  + ## [1] 0.0001555443

1. (Estaturas de presidentes gringos). En un artículo de Wikipedia, se reportan las estaturas de los Presidentes de los Estados Unidos y los de sus oponentes en elecciones. Se ha notado que mientras más alto sea el presidente típicamente gana la elección.
   * Comencemos por leer el Dataframe de el archivo csv obtenido de Wikipedia:

* presidents\_heights <- read.csv('presidents\_heights.csv')  
   knitr::kable(presidents\_heights)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| * year | * winner | * winner\_height | * opponent | * opponent\_height |
| * 2008 | * Barack Obama | * 185 | * John McCain | * 175 |
| * 2004 | * George W. Bush | * 182 | * John Kerry | * 193 |
| * 2000 | * George W. Bush | * 182 | * Al Gore | * 185 |
| * 1996 | * Bill Clinton | * 188 | * Bob Dole | * 187 |
| * 1992 | * Bill Clinton | * 188 | * George H.W. Bush | * 188 |
| * 1988 | * George H.W. Bush | * 188 | * Michael Dukakis | * 173 |
| * 1984 | * Ronald Reagan | * 185 | * Walter Mondale | * 180 |
| * 1980 | * Ronald Reagan | * 185 | * Jimmy Carter | * 177 |
| * 1976 | * Jimmy Carter | * 177 | * Gerald Ford | * 183 |
| * 1972 | * Richard Nixon | * 182 | * George McGovern | * 185 |
| * 1968 | * Richard Nixon | * 182 | * Hubert Humphrey | * 180 |
| * 1964 | * Lyndon B. Johnson | * 192 | * Barry Goldwater | * 180 |
| * 1960 | * John F. Kennedy | * 185 | * Richard Nixon | * 182 |
| * 1956 | * Dwight D. Eisenhower | * 179 | * Adlai Stevenson II | * 178 |
| * 1952 | * Dwight D. Eisenhower | * 179 | * Adlai Stevenson II | * 178 |
| * 1948 | * Harry S. Truman | * 175 | * Thomas Dewey | * 173 |
|  |  |  |  |  |

* + Y realizamos la gráfica:
* ggplot(presidents\_heights, aes(opponent\_height, winner\_height)) +  
   scale\_x\_continuous(breaks = presidents\_heights$opponent\_height) +  
   scale\_y\_continuous(breaks = presidents\_heights$winner\_height) +  
   geom\_point(aes(opponent\_height, winner\_height) , color = "blue", shape = 5, size=3) +  
   geom\_text(aes(label=year),nudge\_y = -1, size=3)
* + Y otro tipo de gráfica:
* ggplot(presidents\_heights, aes(year, height)) +  
   scale\_x\_continuous(breaks = presidents\_heights$year) +  
   geom\_point(aes(year, winner\_height) , color = "blue", shape = 5, size=3) +  
   geom\_point(aes(year, opponent\_height), color = "red", shape = 6, size=3) +  
   theme\_hc()

3. La función genera observaciones aleatorias de una distribución Poisson. Usen la función para simular un número grande de muestras Poisson con parámetro Encuentren la función de masa de probabilidad, media, y varianza para las muestras. Comparen con los valores teóricos de la densidad Poisson.

* Comencemos por obtener la muestra con
* lambda = .61  
   writeLines(paste('Valores teóricos:\nMedia : ', lambda, ' \nVarianza :', lambda))
* ## Valores teóricos:  
  ## Media : 0.61   
  ## Varianza : 0.61
* pois = rpois(1000, lambda)  
   writeLines(paste('Con n = 1000:\nMedia muestral : ', mean(pois), ' \nVarianza muestral :', var(pois)))
* ## Con n = 1000:  
  ## Media muestral : 0.549   
  ## Varianza muestral : 0.540139139139139
* Donde la tabla de densidad de la muestra contra la densidad teórica de la distribución es:
* count <- data.frame(table(pois))  
   count$Freq = count$Freq/1000  
   m = max(pois)  
   count$teorica = dpois(0:m, .61)  
   count$error\_absoluto = abs(count$teorica - count$Freq)  
    
   knitr::kable(count)

|  |  |  |  |
| --- | --- | --- | --- |
| * pois | * Freq | * teorica | * error\_absoluto |
| * 0 | * 0.579 | * 0.5433509 | * 0.0356491 |
| * 1 | * 0.307 | * 0.3314440 | * 0.0244440 |
| * 2 | * 0.103 | * 0.1010904 | * 0.0019096 |
| * 3 | * 0.009 | * 0.0205551 | * 0.0115551 |
| * 4 | * 0.001 | * 0.0031346 | * 0.0021346 |
| * 5 | * 0.001 | * 0.0003824 | * 0.0006176 |
|  |  |  |  |

* Finalmente, graficamos para tener una comparación visual
* expetimental = ggplot(count, aes(pois, Freq)) +   
   labs(title="Frecuencias en la muestra", y="Número de ocurrencias", x="x") +  
   geom\_bar(stat="identity") + theme\_hc()  
    
   teorica = ggplot(data.frame(x = c(0:m) ), aes(x = x)) +  
   labs(title="Función de densidad Poisson", y="y", x="x") +  
   stat\_function( geom="point", n=m+1, fun=dpois, args=list(lambda)) +  
   theme\_hc()  
    
   grid.arrange(expetimental,teorica,ncol=2)
* lambda = .61  
   writeLines(paste('Valores teóricos:\nMedia : ', lambda, ' \nVarianza :', lambda))
* ## Valores teóricos:  
  ## Media : 0.61   
  ## Varianza : 0.61
* pois = rpois(10000, lambda)  
   writeLines(paste('Con n = 10000:\nMedia muestral : ', mean(pois), ' \nVarianza muestral :', var(pois)))
* ## Con n = 10000:  
  ## Media muestral : 0.6084   
  ## Varianza muestral : 0.602509690969097
* Donde la tabla de densidad de la muestra contra la densidad teórica de la distribución es:
* count <- data.frame(table(pois))  
   count$Freq = count$Freq/10000  
   m = max(pois)  
   count
* ## pois Freq  
  ## 1 0 0.5449  
  ## 2 1 0.3285  
  ## 3 2 0.1020  
  ## 4 3 0.0225  
  ## 5 4 0.0021
* count$teorica = dpois(0:m, .61)  
   count$error\_absoluto = abs(count$teorica - count$Freq)  
    
   knitr::kable(count)

|  |  |  |  |
| --- | --- | --- | --- |
| * pois | * Freq | * teorica | * error\_absoluto |
| * 0 | * 0.5449 | * 0.5433509 | * 0.0015491 |
| * 1 | * 0.3285 | * 0.3314440 | * 0.0029440 |
| * 2 | * 0.1020 | * 0.1010904 | * 0.0009096 |
| * 3 | * 0.0225 | * 0.0205551 | * 0.0019449 |
| * 4 | * 0.0021 | * 0.0031346 | * 0.0010346 |
|  |  |  |  |

* Finalmente, graficamos para tener una comparación visual
* expetimental = ggplot(count, aes(pois, Freq)) +   
   labs(title="Frecuencias en la muestra", y="Número de ocurrencias", x="x") +  
   geom\_bar(stat="identity") + theme\_hc()  
    
   teorica = ggplot(data.frame(x = c(0:m) ), aes(x = x)) +  
   labs(title="Función de densidad Poisson", y="y", x="x") +  
   stat\_function( geom="point", n=m+1, fun=dpois, args=list(lambda)) +  
   theme\_hc()  
    
   grid.arrange(expetimental,teorica,ncol=2)

1. Escriban una función en R llamada que regrese el valor estimado de de una muestra de tamaño , utilizando la fórmula del estimado máximo verosímil de la varianza.
   * Dado que

* sd.n <- function(X, insesgado = FALSE){  
    
   media = mean(X)  
   if (insesgado == TRUE){  
   var = sum( (X - media )^2 )/ (length(X)-1)  
   }  
   else{  
   var = sum( (X - media )^2 )/ (length(X))  
   }  
    
   return (var)  
   }
* Y comparando dicha función con la de R:
* var(pois)
* ## [1] 0.6025097
* sd.n(pois, insesgado=TRUE)
* ## [1] 0.6025097
* sd.n(pois)
* ## [1] 0.6024494

1. Escriban una función norma que calcule la norma Euclideana de un vector numerico de longitud . Evaluar la norma de los vectores , y

* norm\_2 <- function(x){  
   sqrt(sum(x^2))  
   }  
    
   v1 = c(0,0,0,1)  
   v2 = c(2, 5, 2, 4)  
   v3 = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)  
    
   norm\_2( v1 )
* ## [1] 1
* norm\_2( v2 )
* ## [1] 7
* norm\_2( v3 )
* ## [1] 19.62142

6. Usar la función curve para graficar la función en el intervalo . Luego usar la función integrate para calcular el valor de la integral:

El límite superior se especifica usando el argumento upper=Inf en la función integrate.

f <- function(x){  
 exp(-(x^2))/(1 + x^2)  
 }  
   
 curve(f, from = 0, to = 10)

integrate(f, lower = 0, upper = Inf)

## 0.6716467 with absolute error < 8.3e-05

1. Construir una matriz con 10 renglones y 2 columnas que contienen datos provenientes de una normal estándar: matrix(rnorm(20),10,2) Esta es una muestra de 10 observaciones de una distribución normal bivariada. Usen la función apply y la función norma que crearon en un ejercicio anterior para calcular las normas euclideanas para cada una de las 10 observaciones.

* x <- matrix(rnorm(20),10,2)  
  x
* ## [,1] [,2]  
  ## [1,] -1.09872495 0.6202522  
  ## [2,] -1.56066739 -0.7500999  
  ## [3,] 0.17937825 1.0126854  
  ## [4,] -0.61747718 -0.2485773  
  ## [5,] 1.91880964 -0.6863750  
  ## [6,] -0.82791842 -1.7737716  
  ## [7,] -0.05347176 1.9989214  
  ## [8,] 0.20889275 -1.2407624  
  ## [9,] -1.47802434 2.2568685  
  ## [10,] 0.39404314 -0.7418746
* apply(x, 1, norm\_2)
* ## [1] 1.2617089 1.7315694 1.0284495 0.6656341 2.0378766 1.9574766 1.9996364  
  ## [8] 1.2582240 2.6977789 0.8400286

1. Los siguientes datos describen el factor de desgaste de papel manufacturado bajo diferentes presiones durante el prensado. Cuatro hojas de papel fueron seleccionadas y probadas para cada uno de los cinco lotes manufacturados, Hacer un boxplot para comparar los diferentes factores de resistencia para cada presión.

* a = c(35.0,35.0,35.0,35.0,49.5,49.5,49.5,49.5,70.0,70.0,70.0,  
   70.0,99.0,99.0,99.0,99.0,140.0,140.0,140.0,140.0)  
   b = c(112,119,117,113,108,99,112,118,120,106,102,109,110,101,99,104,100,102,96,101)  
   desgaste <- data.frame("Presion" = a, "Factor de resistencia" = b)  
   knitr::kable(head(desgaste))

|  |  |
| --- | --- |
| * Presion | * Factor.de.resistencia |
| * 35.0 | * 112 |
| * 35.0 | * 119 |
| * 35.0 | * 117 |
| * 35.0 | * 113 |
| * 49.5 | * 108 |
| * 49.5 | * 99 |
|  |  |

* Y creamos la boxplot:
* ggplot(desgaste, aes(group = Presion, x = Presion, y = Factor.de.resistencia)) +   
   geom\_boxplot() +  
   scale\_x\_continuous(breaks = pretty(desgaste$Presion, n = 10)) +  
   scale\_y\_continuous(breaks = pretty(desgaste$Factor.de.resistencia, n = 10)) +  
   theme\_hc()

1. Relacionado con el modelo de línea de espera. Incorporar las variables:
   * El tiempo total promedio de los n clientes en el sistema (‘promedio\_tiempo\_en\_sistema’)
   * La longitud máxima de la cola (‘long\_max\_cola’)
   * La máxima espera en cola (‘max\_espera’)

* Y ejecutar el modelo 100 veces con , y y hacer un histograma para cada una de las estadísticas de desempeño, y calcular estadísticas descriptivas (media, varianza y coeficiente de variación, min, max, etc) para cada una de ellas.
* Obteniendo los resultados de las medidas descriptivas para las 100 simulaciones:
* df <- data.frame(0,0,0,0,0,0)  
   names(df)<-c("Prom\_espera","Utilizacion","Prom\_clientes",   
   "Promedio\_en\_sistema", 'Long\_max', 'Max\_espera')  
    
   for (i in 1:1000){  
   ans = mm1(lambdaA = 5, lambdaS = 4, n = 1000)  
   df[nrow(df) + 1,] = c(ans[1:6])  
   }  
   df <- data.frame(df[-1, ])  
   row.names(df) <- NULL  
   knitr::kable(head(df))

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| * Prom\_espera | * Utilizacion | * Prom\_clientes | * Promedio\_en\_sistema | * Long\_max | * Max\_espera |
| * 15.81188 | * 78.90583 | * 3.160683 | * 19.75928 | * 19 | * 113.36489 |
| * 10.26788 | * 77.62288 | * 2.083028 | * 14.09415 | * 11 | * 50.89060 |
| * 13.93689 | * 77.43624 | * 2.751316 | * 17.85945 | * 18 | * 72.85723 |
| * 32.82902 | * 84.47701 | * 6.648527 | * 37.00031 | * 29 | * 136.16922 |
| * 14.52488 | * 80.47354 | * 2.928778 | * 18.51585 | * 15 | * 57.17471 |
| * 14.39678 | * 77.41390 | * 2.870328 | * 18.27965 | * 21 | * 90.25467 |
|  |  |  |  |  |  |

* Mostrando estadísticas descriptivas
* summary(df)
* ## Prom\_espera Utilizacion Prom\_clientes Promedio\_en\_sistema  
  ## Min. : 7.181 Min. :69.36 Min. : 1.365 Min. :10.91   
  ## 1st Qu.:12.190 1st Qu.:77.42 1st Qu.: 2.407 1st Qu.:16.13   
  ## Median :14.464 Median :79.94 Median : 2.881 Median :18.44   
  ## Mean :15.573 Mean :79.95 Mean : 3.124 Mean :19.57   
  ## 3rd Qu.:18.050 3rd Qu.:82.24 3rd Qu.: 3.643 3rd Qu.:22.12   
  ## Max. :50.329 Max. :89.44 Max. :10.402 Max. :54.59   
  ## Long\_max Max\_espera   
  ## Min. :10.00 Min. : 40.40   
  ## 1st Qu.:15.00 1st Qu.: 65.30   
  ## Median :18.00 Median : 78.32   
  ## Mean :18.81 Mean : 81.99   
  ## 3rd Qu.:21.00 3rd Qu.: 93.27   
  ## Max. :47.00 Max. :238.58
* Y finalmente, graficando:
* Promedio\_espera\_hist = ggplot(df, aes(Prom\_espera)) +  
   geom\_histogram()  
  Utilizacion\_hist = ggplot(df, aes(Utilizacion)) +  
   geom\_histogram()  
  Promedio\_clientes\_hist = ggplot(df, aes(Prom\_clientes)) +  
   geom\_histogram()  
  tiempo\_en\_sistema\_hist = ggplot(df, aes(Promedio\_en\_sistema)) +  
   geom\_histogram()  
  Long\_max\_cola\_hist = ggplot(df, aes(Long\_max)) +  
   geom\_histogram()  
  Max\_espera\_en\_cola\_hist = ggplot(df, aes(Max\_espera)) +  
   geom\_histogram()  
    
  grid.arrange(Promedio\_espera\_hist, Utilizacion\_hist,   
   Promedio\_clientes\_hist, tiempo\_en\_sistema\_hist,  
   Long\_max\_cola\_hist, Max\_espera\_en\_cola\_hist,  
   ncol=3)
* ## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
  ## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
  ## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
  ## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
  ## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.  
  ## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.