Ejemplo de simulación de Monte Carlo

Equipo Monitoría 2022-I

2022-I

1. Simulación de Monte Carlo: lanzamiento de dos dados

1.1 Desarrollo conceptual/teórico

A continuación, se busca explicarle al estudiante como realizar una *simulación de Monte Carlo* basándose en un ejemplo del texto *Applied time series econometrics* de Enders (2008) [1].

El ejercicio del texto dice:

> Example of the Monte Carlo Method.
>
> Suppose you did not know the probability distribution for the sum of the roll of two dice. One way to calculate the probability distribution would be to buy a pair of dice and roll them several thousand times. If the dice were fair, you would find that a sum on your rolls would approximate this result:

Sum	2	3	4	5	6	7	8	9	10	11	12
Percentage	1/36	2/36	3/36	4/36	5/36	6/36	5/36	4/36	3/36	2/36	1/36

> Instead of actually rolling the dice, you can easily replicate the experiment on a computer. You could draw a random number from a uniform [0, 1] distribution to replicate the roll of the first die. If the computer-generated number falls within the interval [0, 1/6], set the variable $r_1 = 1$. Similarly, if the number falls within the interval [1/6, 2/6], set $r_1 = 2$, and so on. In this way, r_1 will be some integer 1 through 6, each with a probability 1/6. Next, draw a second number from the same uniform [0, 1] distribution to represent the roll of die 2 (r_2). You complete your first Monte Carlo replication by computing the sum $r_1 + r_2$. If you compute several thousand such sums, the sample distribution of the sums will approximate the true distribution.
>
> Of course, more complicated experiments are possible. It is interesting to note that this method was used to reform a standard recommendation at the blackjack tables. Of course, more complicated experiments are possible. It is interesting to note that this method was used to reform a standard recommendation at the blackjack tables.

De la descripción anterior, se observa que claramente lo que se quiere es *simular la distribución de la suma del lanzamiento de dos dados justos*. Como los valores que puede tomar el dado 1 van entre 1 a 6, y los valores que puede tomar el dado 2 van entre 1 a 6, los valores de la suma del lanzamiento de los dos dados deberían

[1] Los interesados pueden revisar la sección 4.4 del libro. En específico, el ejemplo se encuentra en la página 204 del texto

estar entre 2 a 12. Por consiguiente, el ejercicio lo que busca es conocer la probabilidad de cada posible resultado que resulte de la suma de los dos dados lanzados. Por ejemplo, encontrar cuál es la probabilidad de obtener que la suma de los dos dados lanzados sea un 3 o un 5.

La manera más fácil de modelar la situación anterior, es asumir que los resultados del lanzamiento del dado 1 es una *variable aleatoria discreta* llamada X_1 que puede tomar los valores de 1 a 6, mientras que los resultados del lanzamiento del dado 2 es una *variable aleatoria discreta* llamada X_2 que puede tomar los valores de 1 a 6. Por lo tanto, se busca encontrar la distribución de la variable aleatoria $X = X_1 + X_2$.

Encontrar esa distribución es muy sencilla, dado que se limita a un problema de contar[2], es decir, las posibles veces de que un resultado de la suma de los dos dados se obtenga sobre el total de posibles resultados. La tabla siguiente tabla muestra la suma que se obtiene por cada valor posible de los dos lanzamiento de los dados:

```
M=matrix(c(2,3,4,5,6,7,3,4,5,6,7,8,4,5,6,7,8,9,5,6,7,8,
   9,10,6,7,8,9,10,11,7,8,9,10,11,12),nrow=6)
M
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    2    3    4    5    6    7
## [2,]    3    4    5    6    7    8
## [3,]    4    5    6    7    8    9
## [4,]    5    6    7    8    9   10
## [5,]    6    7    8    9   10   11
## [6,]    7    8    9   10   11   12
```

El resultado 2 aparece una sola vez en la tabla anterior, por lo que su probabilidad de ocurrir es $\frac{1}{36}$, mientras que el número 3 aparece dos veces. Como resultado, su probabilidad de ocurrir es $\frac{2}{36}$, y así sucesivamente con cada número que aparece en la tabla hasta llegar al número 12.

Luego de encontrar dicha probabilidad teóricamente, cuyos valores son los que aparecen en la descripción del ejercicio, se procede a resolver el problema computacionalmente utilizando la *simulación de Monte Carlo*.

1.2 Desarrollo computacional por simulación de Monte Carlo.

La simulación de Monte Carlo es un procedimiento muy habitual para encontrar numéricamente resultados probabilísticos que muy difícilmente se podrían calcular analíticamente. En una simulación de Monte Carlo lo que busca es *replicar un proceso generador de datos por computadora*. Para ser más específicos, se simula los datos con las características de la muestra en cuestión (Enders 2008).

La simulación de Monte Carlo, a veces conocida en la literatura bajo el nombre de "experimento de Monte Carlo," requiere los siguientes pasos:

1. Una simulación genera una muestra aleatoria de tamaño T y los parámetros o estadísticos de interés son calculados.
2. El anterior procedimiento se repite N veces (donde N es un número grande), de tal forma que la distribución del parámetro de interés o el estadístico muestral de interés pueda ser tabulada. Esta distribución simulada empíricamente es usada como un *estimativo de la distribución teórica*.

[2]Es un problema de conteo porque el espacio de probabilidad en el que se define la variable aleatoria X es finito.

Finalmente, se resalta que en una simulación de Monte Carlo, la simulación se basa en una *distribución teórica conocida*, por ejemplo, una distribución normal. Es decir, se necesita especificar una distribución teórica, a la hora de efectuar el procedimiento de simulación por Monte Carlo[3].

Lo interesante de este procedimiento es que por métodos computacionales podemos obtener exactamente los mismos resultados que obtuvimos en el desarrollo analítico anterior. Si bien, este ejemplo es muy sencillo y solo ilustrativo, como ya se mencionó anteriormente, hay procesos aleatorios y distribuciones probabilísticas tan complejas, como es el caso cuando se trabaja, por ejemplo, en Econometría Bayesiana. Allí, es necesario utilizar métodos de simulación para conocer la distribución de interés[4].

Ahora bien, vamos a utilizar la simulación de Monte Carlo, para conocer las probabilidades de que ocurra un resultado en particular asociado a la suma del lanzamiento de los dos dados justos. En términos generales, el script consiste en una función general que me realiza la simulación del lanzamiento de los dos dados, y calcula la suma del resultado de dichos lanzamientos para cada dado. Además, realiza la simulación de que dichos lanzamientos se realizan 100000 veces. Es decir, en una sola función se simula el lanzamiento de dos dados (y se calcula la suma de los lanzamientos) 100.000 veces. Posteriormente, se genera un histograma que recopila gráficamente los resultados de la simulación, y que muestra claramente que los resultados obtenidos analíticamente son iguales a los obtenidos por la simulación.[5]

El procedimiento a seguir es el siguiente:

1. Se construye la función que me simula un número muy grande de veces el lanzamiento de dos dados y calcula la suma que resulte del lanzamiento de los dos dados[6].

- Se simula el lanzamiento de un dado. Para ello se utiliza un *if condicional* que me permite construir un dado justo. Como es un dado justo, cualquier de las 6 caras debería tener la misma probabilidad de salir. El condicional lo que busca es que se permita obtener, con la misma probabilidad, cualquiera de las 6 caras del dado. Lo que me garantiza que sea un dado justo, es que a la hora de seleccionar la cara del dado, se parte de que dicha cara del dado se obtiene una distribución uniforme discreta que va de 1 a 6, y que tiene la misma probabilidad para cada número entero que está entre 1 y 6[7].
- Luego de simular, como sería el lanzamiento de un solo dado, se genera una estructura de control cíclica, en este caso un *for*, de tal forma que pueda realizar el lanzamiento de los dados muchas veces. En realidad, lo que se hace es que en cada iteración del *for* se lanzan dos dados, y eso se realiza 100000 veces.
- Se almacenan en un *dataframe* los resultados de la simulación.

2. Se ejecuta la función descrita anteriormente, de tal forma que se genera una muestra que resulta de la simulación de monte carlo. Se especifica un número de repeticiones de 100000, pero en la práctica podrían hacerse más o menos repeticiones, ajustando el parámetro de la función, dependiendo de las necesidades del investigador[8].

[3]Esto en contraste por una simulación por *bootstrapping* donde la simulación se basa en un remuestreo con reemplazo de la misma muestra que se esté estudiando y, por lo tanto, no se basa en suponer una distribución teórica, como sí ocurre en la simulación por Monte Carlo. La simulación por bootstrapping es habitual emplearla a la hora de hacer inferencia estadística y encontrar intervalos de confianza sobre muestras que no se distribuyen de manera normal

[4]Para los/as interesados/as, para conocer muchas de las distribuciones que se emplean a la hora de hacer estimaciones en Econometría Bayesiana, es necesario hacer una sofisticación del procedimiento visto acá que se conoce como *Simulación de Monte Carlo por Cadenas de Markov: Markov Chain Monte Carlo (MCMC)*

[5]Recuerden que un histograma, entre otras cosas, me permite conocer la distribución de probabilidad de una variable aleatorio discreta, que es el uso que se le da en este ejercicio.

[6]En el script se simula 100.000 veces, pero eso es un parámetro que se puede ajustar por diseño de la función, de tal forma que se pueden efectuar más o menos simulaciones. Se recomienda un mínimo de 10.000 simulaciones para que los resultados computacionales se empiecen a acercar a los resultados teóricos.

[7]En realidad, lo que se hace en el código es un poco más complejo. Se parte de una distribución unifrome continua que va de 0 a 1 y se genera un número de esa distribución. Como dicho número está entre 0 y 1, lo que se hace es partir el intervalo $[0, 1]$ en 6 partes iguales y ver en que parte del intervalo cae el número generado por la distribución uniforme continua. Sabiendo en que parte cayó, es posible asignarle un número entero asociado a dicho grupo. Por ejemplo, si el valor que produce la distribución uniforme continua es 0.987434 ese número pertenece al grupo 6 de la partición del intervalo $[0, 1]$ y, por ende, se le asocia el número asociado al último grupo de dicha partición, a saber 6

[8]Se aconseja al lector que haga la simulación con un bajo número de repeticiones, y luego vaya aumentando el número de repeticiones para ver cómo cambian los resultados

3. Se gráfica la distribución de probabilidad asociado a la suma del lanzamiento de los dos dados, utilizando el dataframe generado en la ejecución de la simulación. La razón de utilizar un histograma es que es una forma gráfica de visualizar la distribución de probabilidad de una variable aleatoria discreta, en este caso la suma del resultado del lanzamiento de dos dados.

1.3 Código en R de la simulación de Monte Carlo para la suma del resultado del lanzamiento de dos dados justos.

A continuación, se expondrá la función anteriormente descrita y su histograma. Obsérvese que el resultado de la gráfica de este último muestra que las probabilidades que me muestra esta simulación, son exactamente iguales a la probabilides teóricas que se encuentran descritas en la descripción del ejercicio. Por ejemplo, se ve que en 7 se obtiene un probabilidad de 0.1666667 aproximadamente y se sabe que teóricamente la probabilidad de que la suma del lanzamiento de los dos dado sea 7 es $\frac{6}{36} = 0.166667$. Lo cual muestra que los dos resultados son iguales.

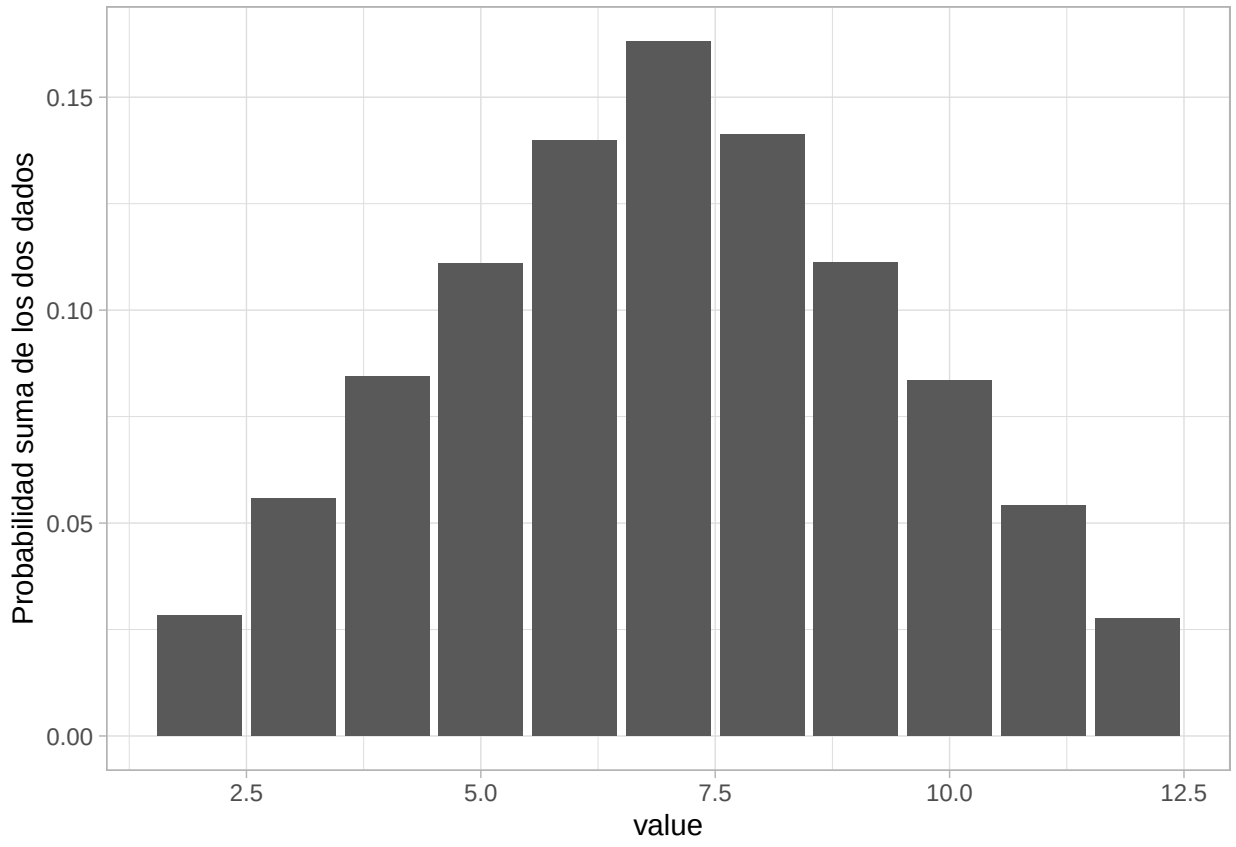
```
library(tidyverse)
```

```
# función que premite realizar la simulación de Monte Carlo
monte_carlo = function(rep){
  # set_number_dado simular el lanzamiento de un dado
  set_number_dado = function(random){
    number = 0
    if (random <= 1/6){
      number = 1
    }else if(1/6 < random &  random <= 2/6){
      number = 2
    }else if(2/6 < random & random <= 3/6){
      number = 3
    }else if(3/6 < random & random <= 4/6){
      number = 4
    }else if(4/6 < random & random <= 5/6){
      number = 5
    }else{
      number = 6
    }
    return(number)
  }
  # vector que almacena los resultados de la simulación de monte carlo
  vect = c()
  # simula rep veces el lanzamiento de dos dados y sumo sus resultados
  for (number in 1:rep){
    # genero los dos lanzamientos de los dos datos
    random1 = set_number_dado(runif(1)) # la distribución que se escogió
    #para hacer la simulación fue una distribución uniforme (para garantizar
    #tener un dado justo)
    random2 = set_number_dado(runif(1))
    vect = append(vect, random1 + random2)
  }
  df = as_tibble(vect) %>%
    mutate(value = as.factor(value))
  return(as_tibble(vect))
}
```

Ahora, después de haber definido la función, lo que sigue es hacer, formalmente, la simulación con 100000 lanzamientos de dos dados justos. Además, se procederá a realizar la gráfica del histograma de esta prueba. La imagen muestra la distribución de probabilidad asociada a la suma que resulte del lanzamiento de estos dos dados justos.

```
# Se realiza la simulación de Monte Carlo
prueba = monte_carlo(100000)# se repite 100000 veces el lanzamiento de
#los dos dados para garantizar alcanzar la distribución teórica.

# Histograma que muestra que se alcanzan la distribución teórica propuesta por
#Enders en el ejemplo de la pag. 204.
graph = prueba %>%
  ggplot(aes(x = value)) +
  geom_bar(aes(y = (..count..)/sum(..count..))) +
  ylab("Probabilidad suma de los dos dados") +
  theme_light(); graph
```




```
# El historgrama representa una distribución empírica igualita a la
#distribución teórica discreta para la suma del resultado de los dos dados.
```

En conclusión, se muestra el poder que tiene la simulación de Monte Carlo, dado que literalmente simulando mucha muchas veces el lanzamiento de dos dados justos[9], fue posible encontrar la distribución de probabilidad

[9]Obviamente, simulando computacional y no físicamente el lanzamiento de un dado, aunque si el lector lanzara físicamente los dos dados 100000 veces debería obtener el mismo histograma de la figura anterior

de la suma de dos dados justos y, además, dicho resultado fue equivalente al cálculo teórico obtenido por un procedimiento analítico.

Los resultados anteriores se pueden extender para procesos aleatorios más sofisticados, así mismo, realiar simulaciones es muy común a la hora de trabajar con series de tiempo. En particular, a la hora de trabajar con series no estacionarias es necesario emplear en el cálculo de algunos estadísticos métodos de simulación, dado que los supuestos de regresión estándar dejan de aplicar, en particular el supuesto de normalidad. Este es el caso, por ejemplo, no es posible calcular la función de densidad de manera analítica para el estadístico de Dickey Fuller, por ende, la forma en la que Dickey y Fuller calcularon la distribución de probabilidad de dicho estadístico (y, al mismo tiempo, los intervalos de confianza que aparecen en los textos guía de series de tiempo y en los paquetes econométricos) fue precisamente a partir de una simulación de Monte Carlo, como la que se ilustró en el presente ejemplo.

Bibliografía

Enders, Walter. 2008. *Applied Econometric Time Series*. Wiley.