5.2 Simulación Monte Carlo

La simulación de Monte Carlo es una técnica que combina conceptos estadísticos **(muestreo aleatorio)** a partir de una distribución de probabilidad, la utilización del computador por la rapidez, permite realizar simulación matemática de los problemas tomando observaciones para hacer deducciones con respecto al sistema real.

Wikipedia "**El método de Montecarlo** es un modelo no determinista o estadístico numérico, usado para aproximar expresiones matemáticas complejas y costosas de evaluar con exactitud. El método se llamó así en referencia al Casino de Montecarlo (Mónaco) por ser “la capital del juego de azar”, al ser la ruleta un generador simple de números aleatorios. El nombre y el desarrollo sistemático de los métodos de Montecarlo datan aproximadamente de 1944 y se mejoraron enormemente con el desarrollo de la computadora" .

Ejemplo 5.2

para este ejemplo vamos a proceder de acuerdo a los pasos definidos anteriormente:

1. Análisis o Definición del Sistema donde exista un problema a solucionar. En este ejemplo vamos a utilizar los datos de la tabla 2.1 del capitulo 2 la misma que en la columna de NO_ATENDIDOS presenta la cantidad de pacientes no atendidos en el HRZ, que representa un costo para el sistema de salud ya que los pacientes son derivados en muchos de los casos a establecimientos privados; el análisis es para determinar por medio de simulación Monte Carlo la cantiad de pacientes que fueron derivados a otros hospitales o clínicas y cual sería el costo que deberiamos optimizar incorporando nuevas opciones de atencón, teniendo que cada paciente derivado cuesta al estado un promedio de 50 dólares.
2. Desarrollo formal del modelo(formulación del Modelo) ##### Variables de Entrada Para el modelo vamos a definir las **variables de entrada** en este caso los datos observados en la tabla 2.1:

```
- Días de la semana
- Cantidad de Pacientes
- Cantidad de Pacientes Atendidos
- Cantidad de Pacientes No Atendidos
```

Modelo Matemático

Luego definimos el **modelo matemático**

```
- Definimos la **demanda diaria de pacientes**, se obtiene la **distribución de la probabilidad fdp()** donde suponemo
s que el comportamiento de los datos observados en los dos meses van a describir apropiadamente el futuro para nuestra
 simulación.
- Luego establecemos la **distribución acumulada FDA()** de la demanda
- Generamos la de rangos **mínimo y máximo** de cada una de dias de la demanda a menudo llamados **números índice**
- Generamos los numeros aleatorios con cualquiera de los métodos de generación vistos en el capitulo 3
- Con cada uno de los números aleatorios localizamos su posición dentro de los rangos para de esa forma  obtener una n
ueva observación.
```

Variables de Salida las variables salida vamos a tener los resultados de la simulación, que puede ser una cantidad importante de observaciones con las demandas de nuestro ejemplo: para este caso de estudio será de 52 semanas de un año:

- Tabla con demandas simuladas
- Calcular la media y la desviación típica
- El Costo total de la cantidad de pacientes no atendidos
- Gráficos

1. Recolección de Datos

   Los datos para la simulación montecarlo fueron obtenidos con las observaciones de los meses de abril y mayo 8 semanas, estos datos los agrupamos por día: Lunes(1), Martes(2), Miercoles(3), Jueves(4), Viernes(5), Sabado(6), Domingo(7) como se muestra en la tabla:

DIA SEMANA	TOTAL PACIENTES	ATENDIDOS	NO ATENDIDOS
lUNES (1)	170	126	44
Martes (2)	159	121	37
Miércoles(3)	148	116	30
Jueves (4)	145	115	29
Viernes (5)	138	101	36
Sábado (6)	115	82	33
Domingo (1)	111	82	29

1. Implementación vía software la implementación igual que todos los métodos y herramientas para la simulación tratados en los capítulos anteriores utilizando un python notebook con los siguientes pasos:
   A. Creamos un DataFrame con los datos
   B. Calculamos la fdp(no atendidos)
   C. Calculamos la DFP(no atendidos)
   D. generamos los rangos de la DFP()
   E. Generamos 52 números aleatorios
   F. Simulamos 52 demandas con los números aleatorios
   G. calculamos los costos de cada paciente **no atendidos**
   H. Calculamos estadísticos
2. Verificación Para este paso correr la simulación con el programa desarrollado y comenzamos a analizar si la información resultante se encuentra dentro de las métricas de desempeño, la observación del comportamiento con alguna distribucíon de frecuencia, la media, la varianza y la desviación estándar así como el comportamiento del generador de números aleatorios.

3. Validación. En este caso ya los datos simulados y verificados confrontamos con los usuarios de los datos (medimos el comportamiento del sistema de interés) para que a su vez puedan determinar si es correcta la simulación y puedan utilizarlo para la toma de decisiones. Como advertimos en la **5**

In [81]:
```
import pandas as pd
import numpy as np
datos = pd.read_csv('C:/Users/JORGEANIBAL/LIBRO/Data/datos1.csv')
datos.head()
```

Out[81]:

	DIA	DIASEMANA	FECHA	TOTAL_PACIENTES	ATENDIDOS	NO_ATENDIDOS
0	1	Lunes	3/4/2017	147	79	68
1	1	Lunes	10/4/2017	167	116	51
2	1	Lunes	17/4/2017	189	149	40
3	1	Lunes	24/4/2017	166	130	36
4	1	Lunes	1/5/2017	150	121	29

In [82]:
```
##### Obtenemos los datos para el modelo
```

Seleccionamos las columnas que vamos a simular en este caso los datos de NO_ATENDIDOS

In [83]:
```
demanda = datos.filter(items=["DIA", "DIASEMANA", "NO_ATENDIDOS"])
```

In [84]:
```
# desplegamos los datos de los 5 primeros registros
demanda.head()
```

Out[84]:

	DIA	DIASEMANA	NO_ATENDIDOS
0	1	Lunes	68
1	1	Lunes	51
2	1	Lunes	40
3	1	Lunes	36
4	1	Lunes	29

Agrupamos los datos por la columna DIA

In [85]:
```
demandas = demanda.groupby("DIASEMANA")
```

Sumamos los datos agrupados por cada DIA

In [86]:
```
demandas.sum()
```

Out[86]:

DIASEMANA	DIA	NO_ATENDIDOS
Domingo	63	261
Jueves	32	235
Lunes	9	393
Martes	18	335
Miercoles	27	274
Sábado	54	294
Viernes	40	290

Para nuestro modelo utilizamos la media de los datos

In [87]:
```
# Obtenemos la media de cada uno de las frecuencias
demandas.mean()
```

Out[87]:

DIASEMANA	DIA	NO_ATENDIDOS
Domingo	7.0	29.000000
Jueves	4.0	29.375000
Lunes	1.0	43.666667
Martes	2.0	37.222222
Miercoles	3.0	30.444444
Sábado	6.0	32.666667
Viernes	5.0	36.250000

Asignamos las medias a demandas

In [88]:
```
tot = demandas.mean()
tot
```

Out[88]:

DIASEMANA	DIA	NO_ATENDIDOS
Domingo	7.0	29.000000
Jueves	4.0	29.375000
Lunes	1.0	43.666667
Martes	2.0	37.222222
Miercoles	3.0	30.444444
Sábado	6.0	32.666667
Viernes	5.0	36.250000

Calculamos las probabilidades de la Columna NO_ATENDIDOS

In [89]:
```
# Ordenamos por Día
suma = tot['NO_ATENDIDOS'].sum()
n=len(tot)
suma
x1 = tot.assign(Probabilidad=lambda x: x['NO_ATENDIDOS'] / suma)

x2 = x1.sort_values('DIA')
a=x2['Probabilidad']
a
```

Out[89]:
```
DIASEMANA
Lunes        0.182993
Martes       0.155986
Miercoles    0.127583
Jueves       0.123101
Viernes      0.151912
Sábado       0.136895
Domingo      0.121530
Name: Probabilidad, dtype: float64
```

Calculamos la probailidad acumulada FDP

In [90]:
```
a1= np.cumsum(a)      #Cálculo la suma acumulativa de las probabilidades
x2['FPA'] =a1
x2
```

Out[90]:

DIASEMANA	DIA	NO_ATENDIDOS	Probabilidad	FPA
Lunes	1.0	43.666667	0.182993	0.182993
Martes	2.0	37.222222	0.155986	0.338979
Miercoles	3.0	30.444444	0.127583	0.466562
Jueves	4.0	29.375000	0.123101	0.589663
Viernes	5.0	36.250000	0.151912	0.741575
Sábado	6.0	32.666667	0.136895	0.878470
Domingo	7.0	29.000000	0.121530	1.000000

In [91]:
```
x2['Min'] = x2['FPA']
x2['Max'] = x2['FPA']
x2
```

Out[91]:

DIASEMANA	DIA	NO_ATENDIDOS	Probabilidad	FPA	Min	Max
Lunes	1.0	43.666667	0.182993	0.182993	0.182993	0.182993
Martes	2.0	37.222222	0.155986	0.338979	0.338979	0.338979
Miercoles	3.0	30.444444	0.127583	0.466562	0.466562	0.466562
Jueves	4.0	29.375000	0.123101	0.589663	0.589663	0.589663
Viernes	5.0	36.250000	0.151912	0.741575	0.741575	0.741575
Sábado	6.0	32.666667	0.136895	0.878470	0.878470	0.878470
Domingo	7.0	29.000000	0.121530	1.000000	1.000000	1.000000

In [92]:
```
lis = x2["Min"].values
lis2 = x2['Max'].values
lis[0]= 0
for i in range(1,7):
    lis[i] = lis2[i-1]
    print(i,i-1)
x2['Min'] = lis
x2
```

```
1 0
2 1
3 2
4 3
5 4
6 5
```

Out[92]:

DIASEMANA	DIA	NO_ATENDIDOS	Probabilidad	FPA	Min	Max
Lunes	1.0	43.666667	0.182993	0.182993	0.000000	0.182993
Martes	2.0	37.222222	0.155986	0.338979	0.182993	0.338979
Miercoles	3.0	30.444444	0.127583	0.466562	0.338979	0.466562
Jueves	4.0	29.375000	0.123101	0.589663	0.466562	0.589663
Viernes	5.0	36.250000	0.151912	0.741575	0.589663	0.741575
Sábado	6.0	32.666667	0.136895	0.878470	0.741575	0.878470
Domingo	7.0	29.000000	0.121530	1.000000	0.878470	1.000000

Generado Dataset de Montecarlo podemos ya simular como es el caso nuestro a 52 semanas

In [93]:
```
# Generamos los 62 # aleatorios por cualquiera de los métodos estudiados y luego podemos realizar la simulación

# Generador de números aleatorios Congruencia lineal
# Borland C/C++ xi+1=22695477xi + 1 mod 2^32

n, m, a, x0, c  = 52, 2**32, 22695477, 4, 1
x = [1] * n
r = [0.1] * n
for i in range(0, n):
        x[i] = ((a*x0)+c) % m
        x0 = x[i]
        r[i] = x0 / m
# llenamos nuestro DataFrame
d = {'ri': r }
dfMCL = pd.DataFrame(data=d)
dfMCL
```

Out[93]:

	ri
0	0.021137
1	0.992121
2	0.164339
3	0.063835
4	0.661529
5	0.400824
6	0.248127
7	0.586098
8	0.107008
9	0.849288
10	0.624741
11	0.476402
12	0.051529
13	0.232983
14	0.081332
15	0.736542
16	0.268061
17	0.883159
18	0.647850
19	0.000471
20	0.508672
21	0.177622
22	0.999577
23	0.524479
24	0.764966

	ri
25	0.750150
26	0.087186
27	0.736931
28	0.685749
29	0.532911
30	0.416229
31	0.706958
32	0.198789
33	0.116261
34	0.647896
35	0.141468
36	0.633563
37	0.160312
38	0.442957
39	0.769847
40	0.891846
41	0.842619
42	0.720730
43	0.225738
44	0.463035
45	0.576381
46	0.754969
47	0.785257
48	0.331067
49	0.483528
50	0.778536

	ri
51	0.571038

In [94]:
```
max = x2 ['Max'].values
min = x2 ['Min'].values
```

In [95]:
```
print(min)
print(max)
```

```
[0.         0.18299284 0.3389791  0.4665619  0.589663   0.741575
 0.8784704 ]
[0.18299284 0.3389791  0.4665619  0.589663   0.741575   0.8784704
 1.        ]
```

In [96]:
```
def busqueda(arrmin, arrmax, valor):
   #print(valor)
    for i in range (len(arrmin)):
    #   print(arrmin[i],arrmax[i])
        if valor >= arrmin[i]  and valor <= arrmax[i]:
            return i
            print(i)
    return -1

xpos = dfMCL['ri']
posi = [0] * n
print (n)
for j in range(n):
    val = xpos[j]
    pos = busqueda(min,max,val)
    posi[j] = pos
```

```
52
```

In [97]:
```
x2 = x2.astype({"DIA" : int})
x2
```

Out[97]:

DIASEMANA	DIA	NO_ATENDIDOS	Probabilidad	FPA	Min	Max
Lunes	1	43.666667	0.182993	0.182993	0.000000	0.182993
Martes	2	37.222222	0.155986	0.338979	0.182993	0.338979
Miercoles	3	30.444444	0.127583	0.466562	0.338979	0.466562
Jueves	4	29.375000	0.123101	0.589663	0.466562	0.589663
Viernes	5	36.250000	0.151912	0.741575	0.589663	0.741575
Sábado	6	32.666667	0.136895	0.878470	0.741575	0.878470
Domingo	7	29.000000	0.121530	1.000000	0.878470	1.000000

In [98]:
```
x2.dtypes
```

Out[98]:
```
DIA              int32
NO_ATENDIDOS   float64
Probabilidad   float64
FPA            float64
Min            float64
Max            float64
dtype: object
```

In [99]:
```
x2.axes[0]
```

Out[99]:
```
Index(['Lunes', 'Martes', 'Miercoles', 'Jueves', 'Viernes', 'Sábado',
       'Domingo'],
      dtype='object', name='DIASEMANA')
```

In [100]:
```
x2.set_index('DIA')
```

Out[100]:

DIA	NO_ATENDIDOS	Probabilidad	FPA	Min	Max
1	43.666667	0.182993	0.182993	0.000000	0.182993
2	37.222222	0.155986	0.338979	0.182993	0.338979
3	30.444444	0.127583	0.466562	0.338979	0.466562
4	29.375000	0.123101	0.589663	0.466562	0.589663
5	36.250000	0.151912	0.741575	0.589663	0.741575
6	32.666667	0.136895	0.878470	0.741575	0.878470
7	29.000000	0.121530	1.000000	0.878470	1.000000

In [101]:
```
import itertools
import math
simula = []
for j in range(n):
    for i in range(n):
        sim = x2.loc[x2["DIA"] == posi[i]+1]
        simu  = sim.filter(['NO_ATENDIDOS']).values
        iterator = itertools.chain(*simu)
        for item in iterator:
            a=item
        simula.append(round(a,2))
simula
```

Out[101]:
```
[43.67,
 29.0,
 43.67,
 43.67,
 36.25,
 30.44,
 37.22,
 29.38,
 43.67,
 32.67,
 36.25,
 29.38,
 43.67,
 37.22,
 43.67,
 36.25,
 37.22,
 29.0,
 36.25,
 43.67,
 29.38,
 43.67,
 29.0,
 29.38,
 32.67,
 32.67,
 43.67,
 36.25,
 36.25,
 29.38,
 30.44,
 36.25,
 37.22,
 43.67,
 36.25,
 43.67,
 36.25,
 43.67,
 30.44,
 32.67,
 29.0,
```

```
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
```

```
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
```

```
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
```

```
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
```

```
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
```

```
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
```

```
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
```

```
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
```

```
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
```

```
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
```

```
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
```

```
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
```

```
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
```

```
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
```

```
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
```

```
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
```

```
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
```

```
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
```

```
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
```

```
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
```

```
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
```

```
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
43.67,
37.22,
43.67,
36.25,
37.22,
29.0,
36.25,
43.67,
29.38,
43.67,
29.0,
29.38,
32.67,
32.67,
43.67,
36.25,
36.25,
```

```
29.38,
30.44,
36.25,
37.22,
43.67,
36.25,
43.67,
36.25,
43.67,
30.44,
32.67,
29.0,
32.67,
36.25,
37.22,
30.44,
29.38,
32.67,
32.67,
37.22,
29.38,
32.67,
29.38,
43.67,
29.0,
43.67,
43.67,
36.25,
30.44,
37.22,
29.38,
43.67,
32.67,
36.25,
29.38,
...]
```

In [102]:
```
dfMCL["Simulación"] = pd.DataFrame(simula)
dfMCL["Costo de Atención"] = dfMCL["Simulación"] * 50
```

In [80]:
```
dfMCL
```

Out[80]:

	ri	Simulación	Costo de Atención
0	0.021137	43.67	2183.5
1	0.992121	29.00	1450.0
2	0.164339	43.67	2183.5
3	0.063835	43.67	2183.5
4	0.661529	36.25	1812.5
5	0.400824	30.44	1522.0
6	0.248127	37.22	1861.0
7	0.586098	29.38	1469.0
8	0.107008	43.67	2183.5
9	0.849288	32.67	1633.5
10	0.624741	36.25	1812.5
11	0.476402	29.38	1469.0
12	0.051529	43.67	2183.5
13	0.232983	37.22	1861.0
14	0.081332	43.67	2183.5
15	0.736542	36.25	1812.5
16	0.268061	37.22	1861.0
17	0.883159	29.00	1450.0
18	0.647850	36.25	1812.5
19	0.000471	43.67	2183.5
20	0.508672	29.38	1469.0
21	0.177622	43.67	2183.5
22	0.999577	29.00	1450.0
23	0.524479	29.38	1469.0
24	0.764966	32.67	1633.5

	ri	Simulación	Costo de Atención
25	0.750150	32.67	1633.5
26	0.087186	43.67	2183.5
27	0.736931	36.25	1812.5
28	0.685749	36.25	1812.5
29	0.532911	29.38	1469.0
30	0.416229	30.44	1522.0
31	0.706958	36.25	1812.5
32	0.198789	37.22	1861.0
33	0.116261	43.67	2183.5
34	0.647896	36.25	1812.5
35	0.141468	43.67	2183.5
36	0.633563	36.25	1812.5
37	0.160312	43.67	2183.5
38	0.442957	30.44	1522.0
39	0.769847	32.67	1633.5
40	0.891846	29.00	1450.0
41	0.842619	32.67	1633.5
42	0.720730	36.25	1812.5
43	0.225738	37.22	1861.0
44	0.463035	30.44	1522.0
45	0.576381	29.38	1469.0
46	0.754969	32.67	1633.5
47	0.785257	32.67	1633.5
48	0.331067	37.22	1861.0
49	0.483528	29.38	1469.0
50	0.778536	32.67	1633.5

	ri	Simulación	Costo de Atención
51	0.571038	29.38	1469.0

In [ ]:

In [ ]: