

UNIVERSIDAD POLITÉCNICA SALESIANA

EI VECINO - CUENCA

Estudiante: Gustavo Guallpa

Profesor: Ing. Diego Quisi

Asignatura: Simulación

Fecha: 27/01/2020.

Tema: Prueba de Generación de Números Pseudoaleatorios.

METODO DE LOS CUADRADOS MEDIOS

```
In [12]: #Importacion de librerias.  
import collections  
import math  
import matplotlib.pyplot as plt  
import pandas as pd
```

```
In [13]: def valuesDiv1(long,numeroOriginal,digitos):  
    numero=int(math.floor((long/2)-(digitos/2)))  
    #print('NUMERO',numero)  
    sNumero=str(numeroOriginal)  
    #print('SNUMERO',sNumero)  
    numUI=sNumero[numero-2:numero+5]  
    #print('NUM',numUI)  
    return int(numUI)
```

```
In [14]: def valuesDiv(long,numeroOriginal,digitos):  
    numero=int(math.floor((long/2)-(digitos/2)))  
    #print('NUMERO',numero)  
    sNumero=str(numeroOriginal)  
    #print('SNUMERO',sNumero)  
    numUI=sNumero[numero-1:numero+6]  
    #print('NUM',numUI)  
    return int(numUI)
```

```

In [15]: def generarNumeroPseudoaleatorios(iteraciones,semilla,digitos):
#Declaracion de la variables.
iteraciones=iteraciones
#Semilla
Xo=semilla
digitos=digitos
#Valores para la tabla
Xn=0
XnPo=0
longuitud=0
Ui=0
Rn=0
#Areglo de todos los numeros generados
arregloRn=[]

#Verificar primero que Xo sea mayor a 3.
if len(str(Xo))>3:
    print('PROCESO NORMAL')
    for i in range(iteraciones):
        #Cuando es cero se empieza con la semilla inicial
        if (i==0):
            Xn=Xo
            XnPo=Xn**2
            #print('Xn*Xn',XnPo)
            longuitud=len(str(XnPo))
            #print('LEN',Longuitud)
            Ui=valuesDiv1(longuitud,XnPo,digitos)
            #print('UI',Ui)
            #Agregamos al arreglo de semillas
            #print('Ui',Ui)
            #Valor
            Rn=Ui/10000000
            Rn=round(Rn, 2)
            arregloRn.append(Rn)
            print(Rn)
            print('***')
        #Cuando es diferente de cero, La semilla es el valor de Ui.
        else:
            Xn=Ui
            XnPo=Xn**2
            #print('Xn*Xn',XnPo)
            longuitud=len(str(XnPo))
            #print('LEN',Longuitud)
            Ui=valuesDiv(longuitud,XnPo,digitos)
            #print('Agregamos',Xn)
            #or o in arregloSemillas:
            # print(o)
            #print('UI',Ui)
            Rn=Ui/10000000
            Rn=round(Rn, 2)
            arregloRn.append(Rn)
            print(Rn)
            print('***')

else:

```

```

        print('LA SEMILLA ES MENOR A 3')
    print('LA FRECUENCIA DE REPETICION ES DE:')
    #Vemos la frecuencia de repeticion
    counter=collections.Counter(arregloRn)
    print(counter)

    return arregloRn

```

```

In [16]: iteraciones=100
        semilla=74731897457
        digitos=7
        arregloRn = generarNumeroPseudoaleatorios(iteraciones,semilla,digitos)

```

PROCESO NORMAL

0.56

0.92

0.58

0.15

0.06

0.02

0.3

0.12

0.53

0.12

SACAMOS EL VALOR DE n.

```

In [17]: n= int (math.sqrt(len(arregloRn)))
        print(n)

```

10

AHORA VAMOS A CLASIFICAR LOS NUMEROS

```
In [18]: def clasificarNumeros(n,arregloRn):
    grupos = []
    inicio=0.00
    a=0
    b=1
    ranNumeros= {}

    for i in range(n+1):
        grupos.append(round(inicio,2))
        inicio=inicio+(1/n)

    for i in range(len(grupos)-1):
        valInferior=grupos[a]
        valSuperior=grupos[b]
        ranNumeros.update({str(valInferior)+","+str(valSuperior):[]})
        for i in arregloRn:
            if i==0.00:
                if i>=valInferior and i<= valSuperior:
                    ranNumeros[str(valInferior)+","+str(valSuperior)].append(i)
            else:
                if i>valInferior and i<= valSuperior:
                    ranNumeros[str(valInferior)+","+str(valSuperior)].append(i)
        a=b
        b=a+1
    return ranNumeros
```

```
In [19]: diccionario = clasificarNumeros(n,arregloRn)
sumaOi=0.00
print("Intervalo " , "      Ei","      Oi", "      (Oi-Ei)^2/Ei")
for intervalo, rangos in enumerate (diccionario.items()):
    porcentajeOi= ((len(rangos[1])-n)**2)/n
    numRepeticion= len(rangos[1])
    sumaOi+=porcentajeOi
    print(intervalo+1,"      ",str(n)+"("+rangos[0]+") ", numRepeticion,"
print('Valor Chi-Cuadrado',sumaOi)
sumaOi=sumaOi
if sumaOi<=16.9:
    print('La Diferencia entre la distribución de la muestra y la distribución un
else:
    print('La Diferencia entre la distribución de la muestra y la distribución un
```

Intervalo	Ei	Oi	(Oi-Ei)^2/Ei
1	10(0.0,0.1)	7	0.9
2	10(0.1,0.2)	14	1.6
3	10(0.2,0.3)	13	0.9
4	10(0.3,0.4)	13	0.9
5	10(0.4,0.5)	9	0.1
6	10(0.5,0.6)	16	3.6
7	10(0.6,0.7)	4	3.6
8	10(0.7,0.8)	8	0.4
9	10(0.8,0.9)	9	0.1
10	10(0.9,1.0)	7	0.9

Valor Chi-Cuadrado 13.0

La Diferencia entre la distribución de la muestra y la distribución uniforme se acepta(h0 es válida)

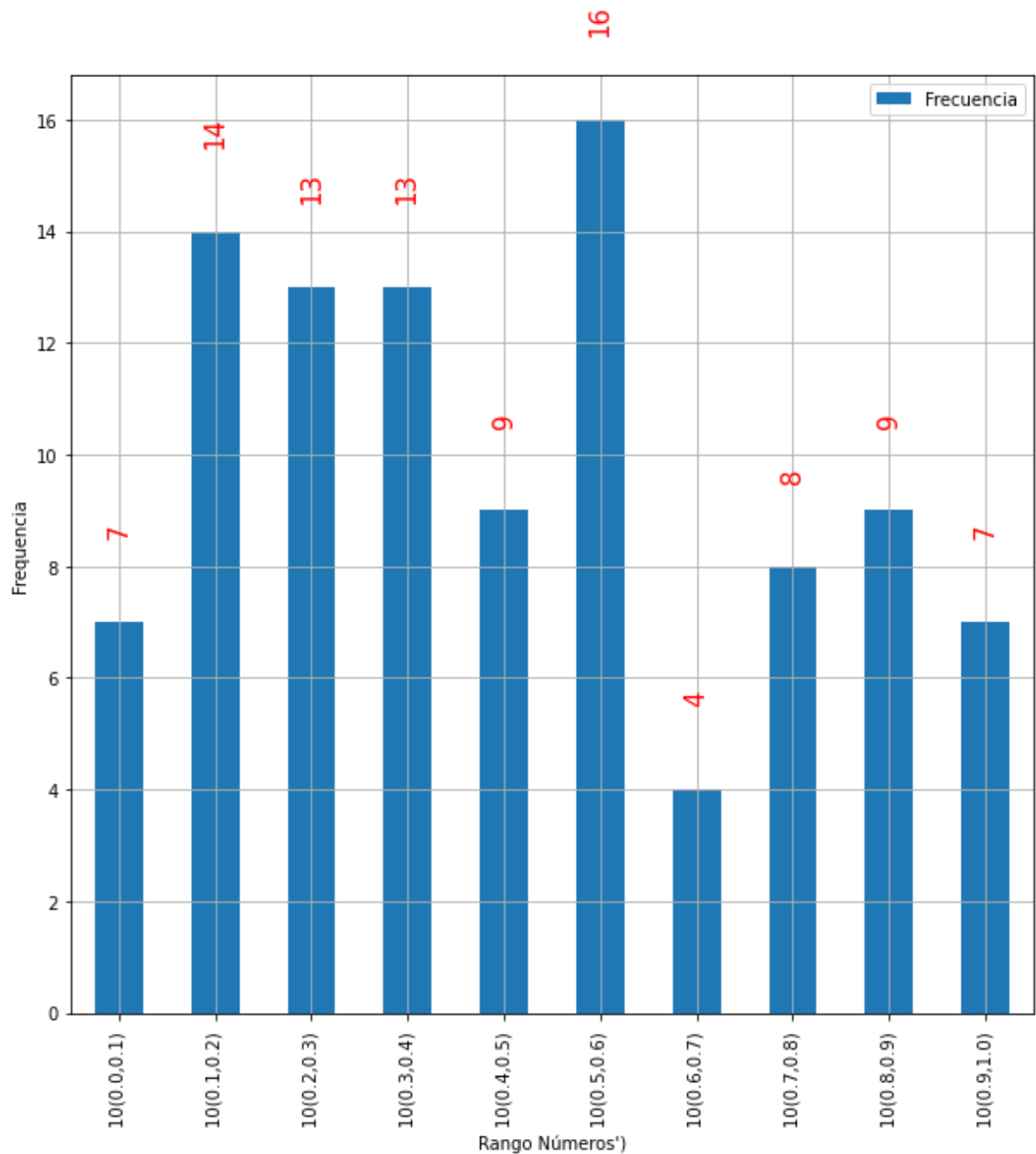
HISTOGRAMA CUADRADOS MEDIOS

```
In [22]: plt.rcParams['figure.figsize'] = [10, 10]
rango=[]
frecuencia = []
for intervalo, rangos in enumerate (diccionario.items()):
    valStr = str(n)+"("+rangos[0]+") "
    rango.append(valStr)
    valor= len(rangos[1])
    frecuencia.append(valor)
tupla1=tuple(rango)
tupla2=tuple(frecuencia)
```

```
In [23]: data = {"Rango": tupla1,
                "Frecuencia": tupla2,
                "Repeticion": tupla2}

frecuencia = pd.DataFrame(data)
ax = frecuencia.plot.bar("Rango", "Frecuencia")

for i, bar in enumerate(ax.patches):
    ax.text(bar.get_x() + bar.get_width() / 2, bar.get_height() + 1.5,
            f"{frecuencia['Frecuencia'][i]}",
            horizontalalignment='center', verticalalignment='bottom',
            fontsize=15, rotation=90, color='r')
ax.set_xlabel("Rango Números")
ax.set_ylabel("Frecuencia")
ax.grid(True)
plt.show()
```



METODO DE CONGRUENCIA LINEAL

```
In [27]: def nextSeed(multiplicador,xoAnterior,incremento,modulo):  
         numero=multiplicador*xoAnterior+incremento  
         numXn=numero % modulo  
         return numXn
```

```
In [28]: def generarNumeroPseudoaleatorios2(multiplicador, incremento, modulo, iteraciones, se
#Declaracion de la variables.
a=multiplicador
b=incremento
m=modulo
iteracion=iteraciones #Número de iteraciones
Xo=semilla#Semilla
#Valores para la tabla
Xn=0
Un=0

#Areglo de todos los numeros generados
arregloRn2=[]

#Verificar que los parametros sean correctos.
if a>0 and b>0 and m>0:
    print('PROCESO NORMAL')
    for i in range(iteracion):
        #Cuando es cero se empieza con la semilla inicial
        if (i==0):
            Xn=Xo

            #print(Xn)
            #Cuando es diferente de cero, la semilla es el valor de Ui.
        else:
            Xn=nextSeed(a,Xn,b,m)
            #print(Xn)
            Xn=Xn
            Un=Xn/m
            Un=round(Un,2)
            arregloRn2.append(Un)
            print(Un)
            print('***')
    else:
        print('VALORES INGRESADOS INCORRECTOS')

    print('LA FRECUENCIA DE REPETICION ES DE:')
    #Vemos la frecuencia de repeticion
    counter=collections.Counter(arregloRn2)
    print(counter)

    return arregloRn2
```



```
In [29]: multiplicador=74731897457#a
incremento=37747318974#b
modulo=19#m
iteraciones=100
semilla=7
arregloRn2 = generarNumeroPseudoaleatorios2(multiplicador,incremento,modulo,iteraciones,semilla)

0.32
***
0.0
***
0.63
***
0.37
***
0.89
***
0.84
***
0.95
***
0.74
***
0.16
***
0.32
***
```

SACAMOS EL VALOR DE n.

```
In [30]: n= int (math.sqrt(len(arregloRn2)+1))
print(n)
```

10

AHORA VAMOS A CLASIFICAR LOS NUMEROS

```
In [31]: def clasificarNumeros2(n,arregloRn):
    grupos = []
    inicio=0.00
    a=0
    b=1
    ranNumeros= {}

    for i in range(n+1):
        grupos.append(round(inicio,2))
        inicio=inicio+(1/n)

    for i in range(len(grupos)-1):
        valInferior=grupos[a]
        valSuperior=grupos[b]
        ranNumeros.update({str(valInferior)+","+str(valSuperior):[]})
        for i in arregloRn:
            if i==0.00:
                if i>=valInferior and i<= valSuperior:
                    ranNumeros[str(valInferior)+","+str(valSuperior)].append(i)
            else:
                if i>valInferior and i<= valSuperior:
                    ranNumeros[str(valInferior)+","+str(valSuperior)].append(i)
        a=b
        b=a+1
    return ranNumeros
```

```
In [32]: diccionario2 = clasificarNumeros(n,arregloRn2)
    sumaOi1=0.00
    print("Intervalo " , "      Ei","      Oi", "      (Oi-Ei)^2/Ei")
    for intervalo, rangos in enumerate (diccionario2.items()):
        porcentajeOi= ((len(rangos[1])-n)**2)/n
        numRepeticion= len(rangos[1])
        sumaOi1+=porcentajeOi
        print(intervalo+1,"      ",str(n)+"("+rangos[0]+") ", numRepeticion,"
    print('Valor Chi-Cuadrado',sumaOi1)
    if sumaOi1<=16.9:
        print('La Diferencia entre la distribución de la muestra y la distribución ur
    else:
        print('La Diferencia entre la distribución de la muestra y la distribución ur
```

Intervalo	Ei	Oi	(Oi-Ei)^2/Ei
1	10(0.0,0.1)	11	0.1
2	10(0.1,0.2)	11	0.1
3	10(0.2,0.3)	0	10.0
4	10(0.3,0.4)	22	14.4
5	10(0.4,0.5)	0	10.0
6	10(0.5,0.6)	0	10.0
7	10(0.6,0.7)	11	0.1
8	10(0.7,0.8)	11	0.1
9	10(0.8,0.9)	22	14.4
10	10(0.9,1.0)	11	0.1

Valor Chi-Cuadrado 59.300000000000004

La Diferencia entre la distribución de la muestra y la distribución uniforme no se acepta(h0 no es válida)

HISTOGRAMA CONGRUENCIA LINEAL

```
In [33]: rango=[]
frecuencia = []
for intervalo, rangos in enumerate (diccionario2.items()):
    valStr = str(n)+"("+rangos[0]+") "
    rango.append(valStr)
    valor= len(rangos[1])
    frecuencia.append(valor)

tupla1=tuple(rango)
tupla2=tuple(frecuencia)
```

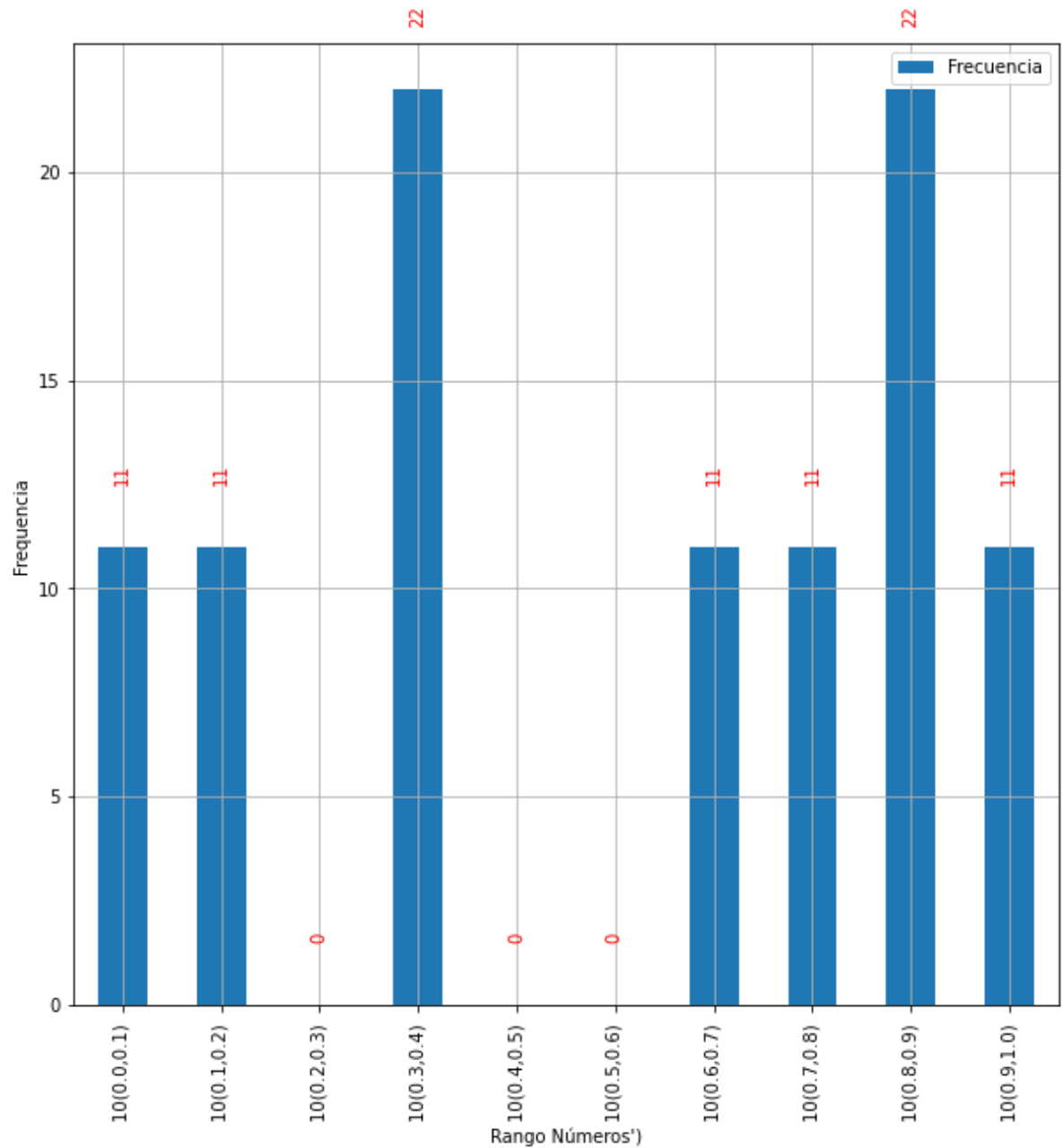
```
In [34]: data = {"Rango": tupla1,
                "Frecuencia": tupla2,
                "Repeticion": tupla2}

frecuencia = pd.DataFrame(data)
ax = frecuencia.plot.bar("Rango", "Frecuencia")

for i, bar in enumerate(ax.patches):
    ax.text(bar.get_x() + bar.get_width() / 2, bar.get_height() + 1.5,
            f"{frecuencia['Frecuencia'][i]}",
            horizontalalignment='center', verticalalignment='bottom',
            fontsize=10, rotation=90, color='r')

ax.set_xlabel("Rango Números")
ax.set_ylabel("Frecuencia")
ax.grid(True)

plt.show()
```



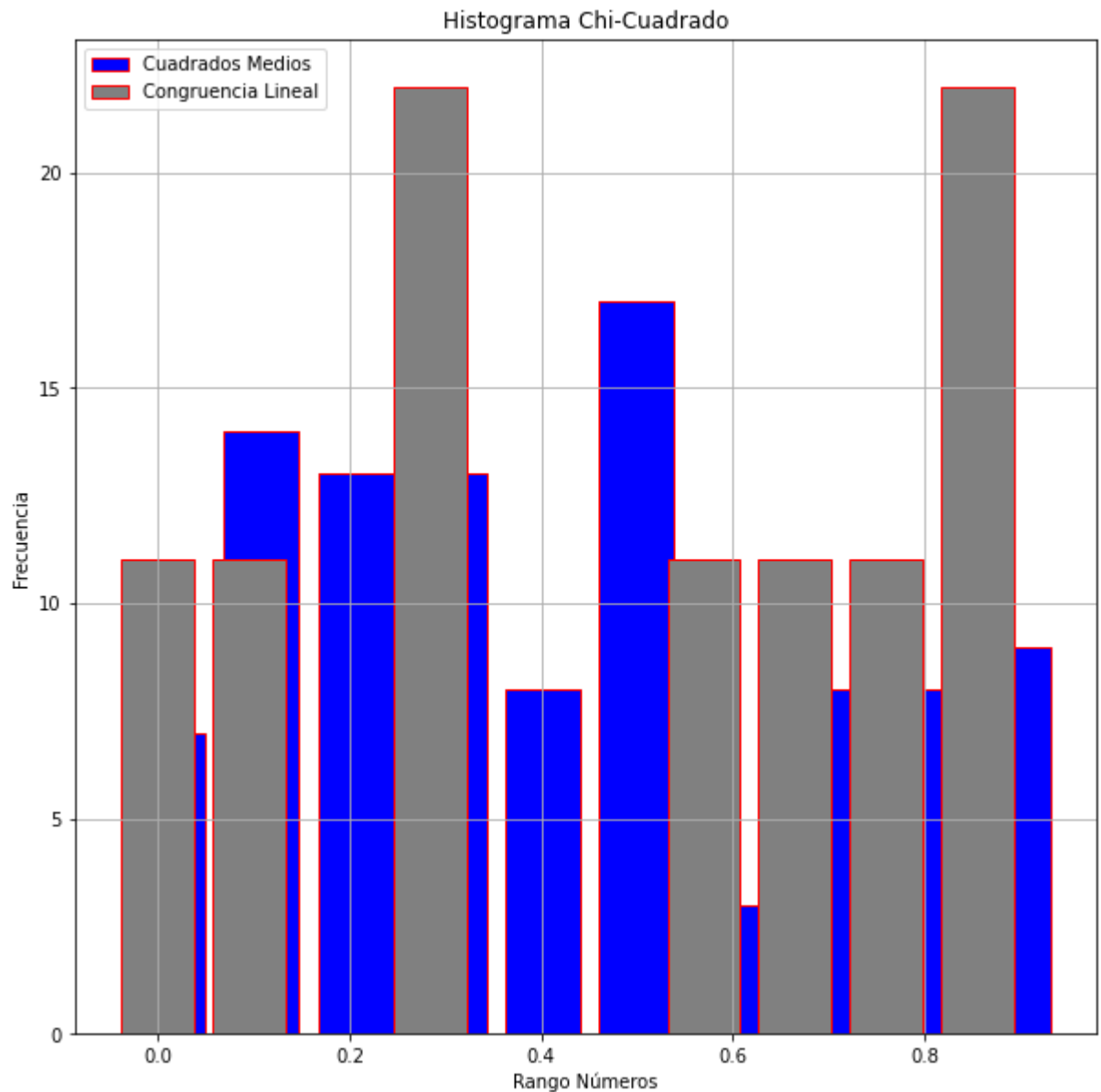
COMPARACION DE HISTOGRAMAS.

```

In [35]: plt.rcParams['figure.figsize'] = [10, 10]
plt.hist(arregloRn,color='b', rwidth=0.80,align='left',edgecolor='red',label='Cuadrados Medios')
plt.hist(arregloRn2,color='grey', rwidth=0.80,align='left',edgecolor='red',label='Congruencia Lineal')
plt.xlabel('Rango Números')
plt.ylabel('Frecuencia')
plt.title('Histograma Chi-Cuadrado')
plt.legend(loc='upper left')
plt.grid(True)
plt.show()

print('Valor Chi-Cuadrado Cuadrados Medios:',suma0i)
print('Valor Chi-Cuadrado Congruencia Lineal:',suma0i1)
if (suma0i<suma0i1):
    print('EL MEJOR METODO ES EL DE CUADRADOS MEDIOS')
else:
    print('EL MEJOR METODO ES EL DE CONGRUENCIA LINEAL')

```



```
Valor Chi-Cuadrado Cuadrados Medios: 13.0  
Valor Chi-Cuadrado Congruencia Lineal: 59.300000000000004  
EL MEJOR METODO ES EL DE CUADRADOS MEDIOS
```