

## Computación

Aula Virtual: <https://famaf.aulavirtual.unc.edu.ar/course/view.php?id=747>

Resguardo tutoriales: <https://www.famaf.unc.edu.ar/~moreschi/docencia/Computacion/>

### Tutorial Problemas 1 de la Guía N° 6

#### Problema 1:

Cuando se estudia probabilidad y procesos estocásticos, aparece naturalmente la función gaussiana o de Gauss o normal, definida por:

$$g(y) = \frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{(y-\mu)^2}{2\sigma^2}};$$

que está centrada en  $\mu$  con ancho característico  $\sigma$ , que es su desviación estándar. A  $\sigma^2$  se la denomina la varianza de la distribución.

Suponga un proceso estocástico que se repite  $N$  veces y tiene una distribución de probabilidad  $P(y)$  dada por la función de Gauss, que asignamos a pasos en el espacio de las  $x \in [x_{\min}, x_{\max}]$ .

- a) Usando PYTHON, simule una distribución para este proceso con los comandos:

```

1 import numpy as np
2
3 xmin = 0.
4 xmax = 1.4
5
6 # elección de la semilla -----
7 np.random.seed(11)
8
9 # elección de Nrep -----
10 Nrep = 1001
11 x = np.linspace(xmin, xmax, Nrep)
12 y = np.random.randn(Nrep)
    
```

donde estamos usando la función `RANDN` del paquete `RANDOM` de `NUMPY` y asignaremos la distribución al espacio de las  $x$ .

- b) Defina una función  $gau(yy, prom, ancho)$  que sea la función gaussiana evaluada en  $yy$ , centrada en  $prom$  y con desviación estándar  $ancho$ .
- c) Genere un gráfico de  $y$  vs  $x$ , mostrando el resultado en pantalla y guardando el mismo en un archivo con nombre `'graficos/p1-normal-'+str(Nrep)+' .png'`
- d) Genere un gráfico del histograma de este proceso y de la función de Gauss, con la elección de valores  $pro = 0.$  y  $anc = 1.$ , haciendo uso de los comandos:

```

1 numbines = 50
2
3 n, bins, patches = plt.hist(y, numbines, density=False, facecolor='g',
4 alpha=0.75, label='histograma')
5 suma = n.sum()
6 hbin = (bins[-1]-bins[0])/numbines
    
```

```

7 integ = suma*hbin
8 plt.plot(bins , (integ*gau(bins,pro,anc)), 'r',
9          label='integ * Gaussiana mu='+str(pro)+' , sigma='+str(anc) )

```

muéstrela en pantalla y guarde el gráfico en un archivo con el nombre 'graficos/p1-histograma-  
'+str(Nrep)+' .png'

- e) Genere un gráfico del histograma relativo de este proceso y de la función de Gauss, con la elección de valores  $pro = 0.$  y  $anc = 1.$ , haciendo uso de los comandos:

```

1 n, bins, patches = plt.hist(y, numbines, density=True, facecolor='g',
2 alpha=0.75, label='histograma ')
3 plt.plot(bins , (gau(bins,pro,anc)), 'r',
4          label='Gaussiana mu='+str(pro)+' , sigma='+str(anc) )

```

muéstrela en pantalla y guarde el gráfico en un archivo con el nombre 'graficos/p1-histograma-  
relativo-'+str(Nrep)+' .png'

- f) Genere un gráfico del histograma de la variable  $x$  con el comando:

```

1 n, bins, patches = plt.hist(x, numbines, density=False, facecolor='g',
2 alpha=0.75, label='histograma ')

```

muéstrela en pantalla y guarde el gráfico en un archivo con el nombre 'graficos/p1-histograma-  
x-'+str(Nrep)+' .png'

- g) Repita todo lo anterior ahora con la elección  $N = 10001$  y compare las simulaciones con la función gaussiana.

Los gráficos deben ser completados con título, etiquetas de curvas, etiquetas de ejes, etc.

## Tutorial:

- Guarde en el archivo p1.py las siguientes instrucciones:

```

1 """
2 definimos una variable con random y
3 hacemos el histograma correspondiente
4 """
5 import numpy as np
6 import matplotlib.pyplot as plt
7
8 def gau(yy,prom,ancho):
9     return np.exp(-(yy-prom)**2/(2.*ancho**2))/( np.sqrt(2.*np.pi)*ancho )
10
11
12 numbines = 50
13 xmin = 0.
14 xmax = 1.4
15 pro = 0.
16 anc = 1.
17
18 # elección de la semilla -----

```

```

19 np.random.seed(11)
20
21 # elección de Nrep -----
22 Nrep = 1000
23 x = np.linspace(xmin, xmax, Nrep)
24 y = np.random.randn(Nrep)
25 print('x =',x)
26 print('y =',y)
27 print('len(x) =',len(x))
28 print('len(y) =',len(y))
29
30 plt.figure( figsize=(10, 7.5) )
31 plt.title('Distribución normal con '+str(Nrep)+' puntos')
32 plt.xlabel('x')
33 plt.ylabel('y')
34 plt.grid()
35 plt.plot(x , y,'o-', label='normal')
36 plt.legend(loc="best")
37 plt.savefig('graficos/p1-normal-'+str(Nrep)+'.png', dpi=100)
38 plt.show()
39
40
41 plt.figure( figsize=(10, 7.5) )
42 plt.title('Histograma de la distribución normal con '+str(Nrep)+' puntos')
43 plt.xlabel('y')
44 plt.ylabel('dn/dy')
45 plt.grid()
46 n, bins, patches = plt.hist(y, numbins, density=False, facecolor='g',
47                             alpha=0.75, label='histograma')
48 suma = n.sum()
49 hbin = (bins[-1]-bins[0])/numbins
50 integ = suma*hbin
51 plt.plot(bins , (integ*gau(bins,pro,anc)), 'r',
52          label='integ * Gaussiana mu='+str(pro)+' , sigma='+str(anc) )
53 plt.legend(loc="best")
54 plt.savefig('graficos/p1-histograma-'+str(Nrep)+'.png', dpi=100)
55 plt.show()
56
57
58 plt.figure( figsize=(10, 7.5) )
59 plt.title('Histograma relativo de la distribución normal con '
60          +str(Nrep)+' puntos')
61 plt.xlabel('y')
62 plt.ylabel('(1/N) dn/dy')
63 plt.grid()
64 n, bins, patches = plt.hist(y, numbins, density=True, facecolor='g',
65                             alpha=0.75, label='histograma')
66 plt.plot(bins , (gau(bins,pro,anc)), 'r',
67          label='Gaussiana mu='+str(pro)+' , sigma='+str(anc) )
68 plt.legend(loc="best")
69 plt.savefig('graficos/p1-histograma-relativo-'+str(Nrep)+'.png', dpi=100)
70 plt.show()
71
72 #print(' suma =',suma)
73 #print(' hbin =',hbin)
74 #print('suma*hbin =',(suma*hbin))

```

```

75 #print('      n =',n)
76 #print('      bins =',bins)
77 #print('patches =',patches)
78
79 plt.figure( figsize=(10, 7.5) )
80 plt.title('Histograma de x')
81 plt.xlabel('x')
82 plt.ylabel('dn/dx')
83 plt.grid()
84 n, bins, patches = plt.hist(x, numbins, density=False, facecolor='g',
85                             alpha=0.75, label='histograma')
86 plt.legend(loc="best")
87 plt.savefig('graficos/p1-histograma-x-'+str(Nrep)+'.png', dpi=100)
88 plt.show()
89
90 # elección de Nrep -----
91 Nrep = 40000
92 x = np.linspace(xmin, xmax, Nrep)
93 y = np.random.randn(Nrep)
94 print('x =',x)
95 print('y =',y)
96 print('len(x) =',len(x))
97 print('len(y) =',len(y))
98
99 plt.figure( figsize=(10, 7.5) )
100 plt.title('Distribución normal con '+str(Nrep)+' puntos')
101 plt.xlabel('x')
102 plt.ylabel('y')
103 plt.grid()
104 plt.plot(x , y,'o-', label='normal')
105 plt.legend(loc="best")
106 plt.savefig('graficos/p1-normal-'+str(Nrep)+'.png', dpi=100)
107 plt.show()
108
109
110 plt.figure( figsize=(10, 7.5) )
111 plt.title('Histograma de la distribución normal con '+str(Nrep)+' puntos')
112 plt.xlabel('y')
113 plt.ylabel('dn/dy')
114 plt.grid()
115 n, bins, patches = plt.hist(y, numbins, density=False, facecolor='g',
116                             alpha=0.75, label='histograma')
117 suma = n.sum()
118 hbin = (bins[-1]-bins[0])/numbins
119 integ = suma*hbin
120 plt.plot(bins , (integ*gau(bins,pro,anc)), 'r',
121          label='integ * Gaussiana mu='+str(pro)+' , sigma='+str(anc) )
122 plt.legend(loc="best")
123 plt.savefig('graficos/p1-histograma-'+str(Nrep)+'.png', dpi=100)
124 plt.show()
125
126 #quit()

```

- Desde la terminal ejecute:

```
python3 p1.py
```

e interprete el resultado.

Alternativamente ejecute:

```
python3
```

y vaya agregando uno a uno los bloques del programa.

- Estudie cada paso del programa y agregue comentarios explicativos.
- Altere el programa para probar distintas cosas.

Se deberían generar los siguientes gráficos:

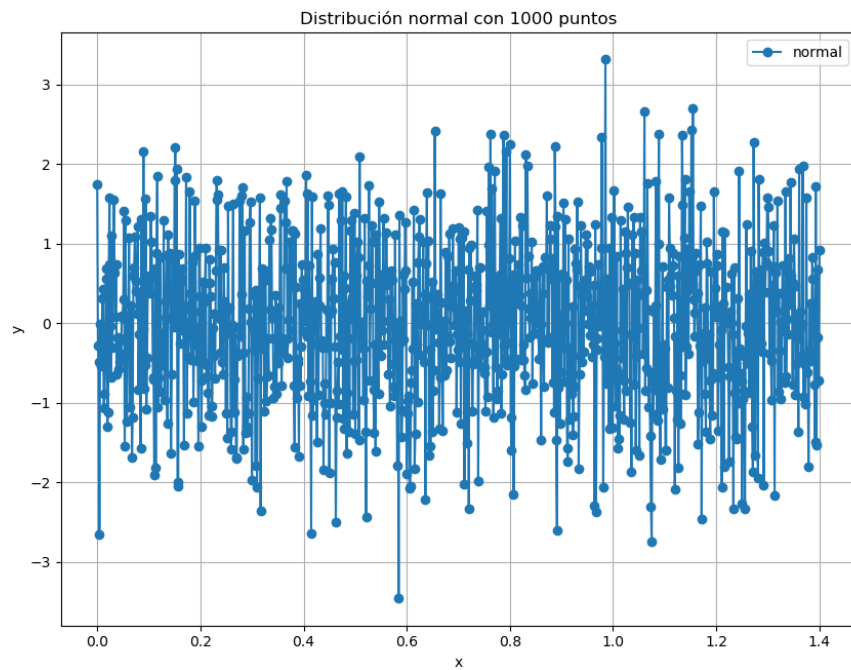


Figura 1: Distribución normal con 1000 puntos.

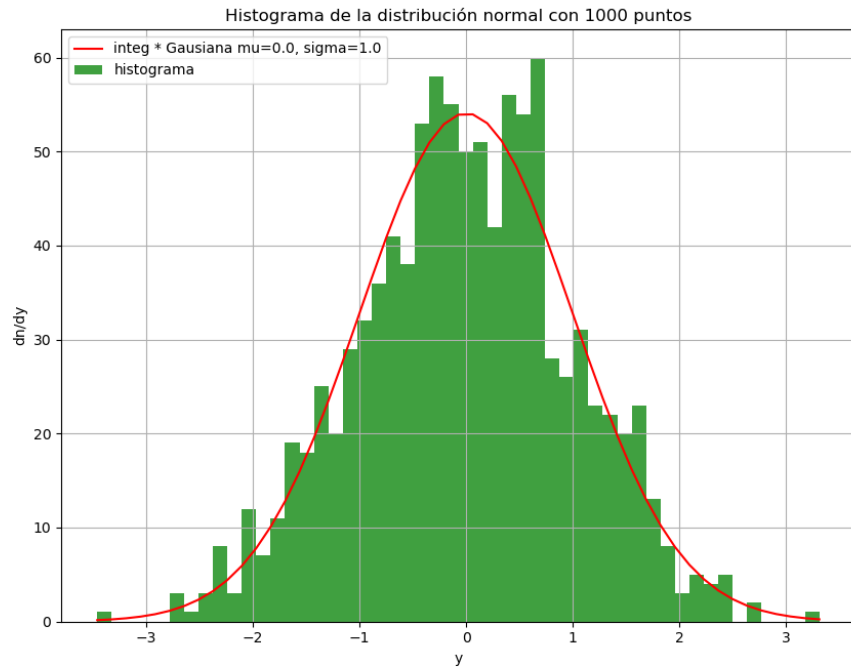


Figura 2: Histograma de la distribución con 1000 puntos.

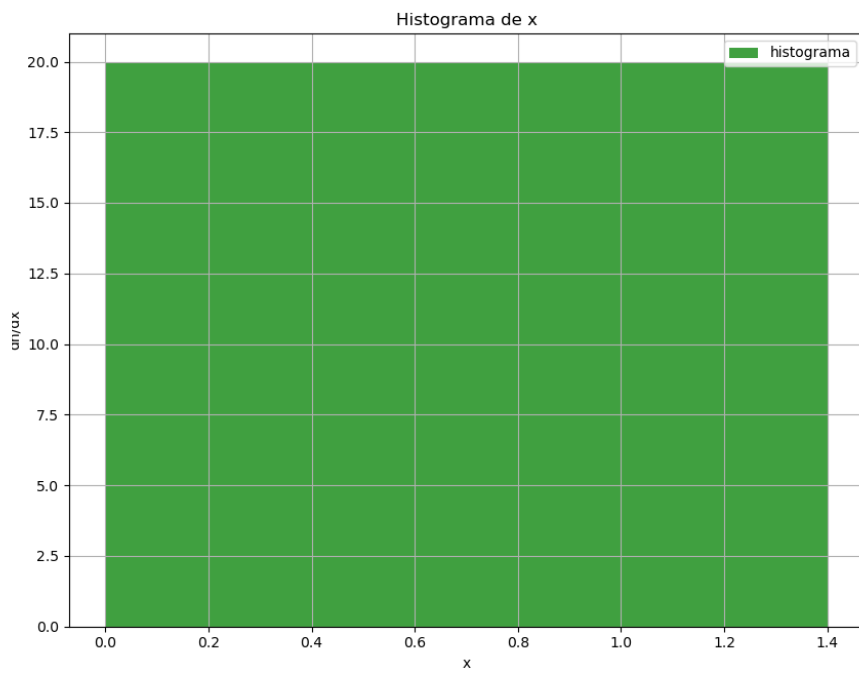


Figura 3: Histograma de la variable  $x$  con 1000 puntos.

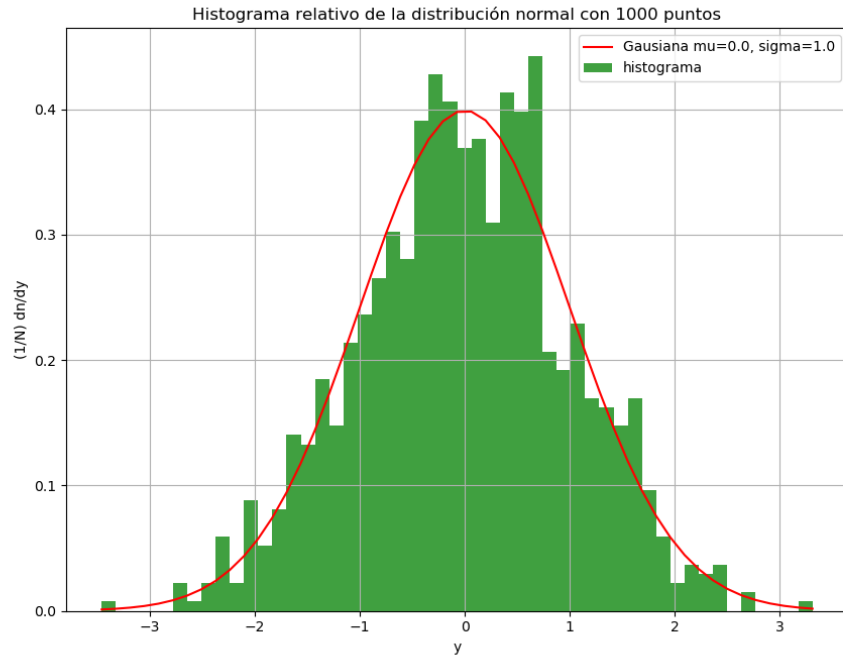


Figura 4: Histograma relativo de la distribución con 1000 puntos.

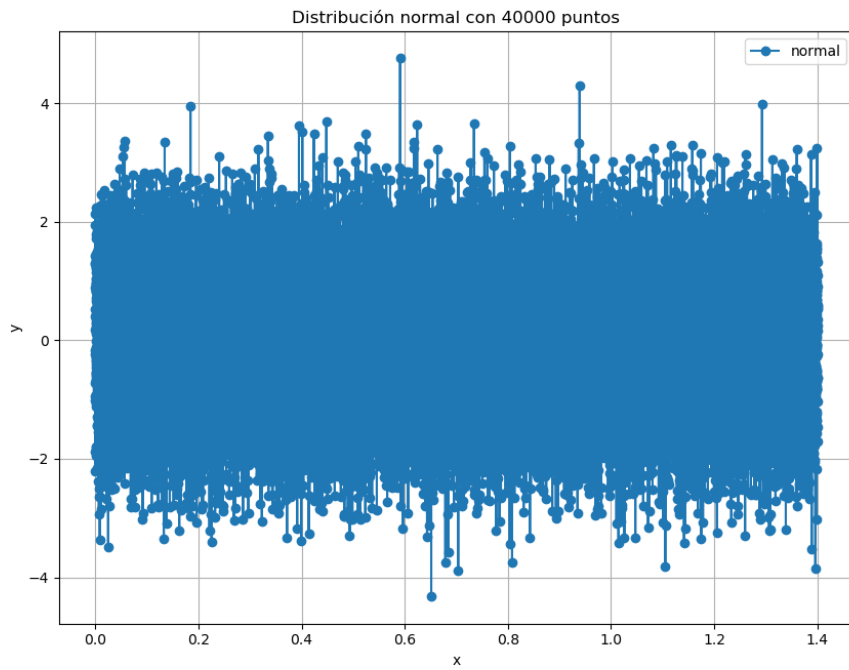


Figura 5: distribución normal con 40000 puntos.

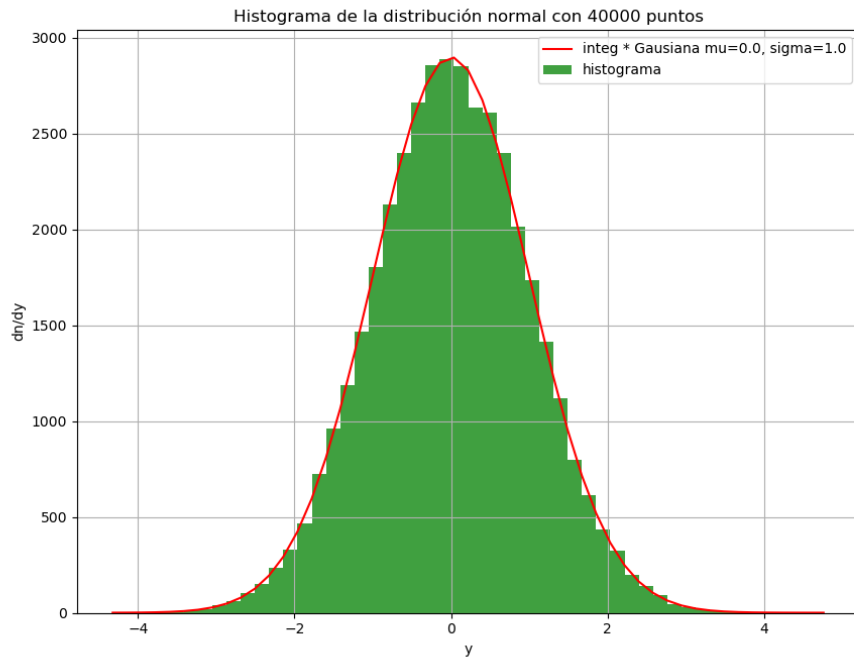


Figura 6: Histograma de la distribución con 40000 puntos.