Untitled

September 11, 2020

[1]:
```python
import math
import numpy as np
from scipy.special import eval_hermite as hn
import matplotlib.pyplot as plt
%matplotlib inline
```

[30]:
```python
plt.rcParams['figure.figsize']=10,6
```

This work has only the purpose of show how to plot different statistics related to quantum-optics, so we wont teach the bacground theory behind the equations because that is not the goal. There will be just equations and code. Let's start

Remeber the mathematic definition of Fock or number states

$$|n\rangle = \frac{1}{\sqrt{2^n n!}}\left(x - \frac{\partial}{\partial x}\right)^n |0\rangle$$

0.1 Coherent-States

0.1.1 Photon Distribution ρ_n

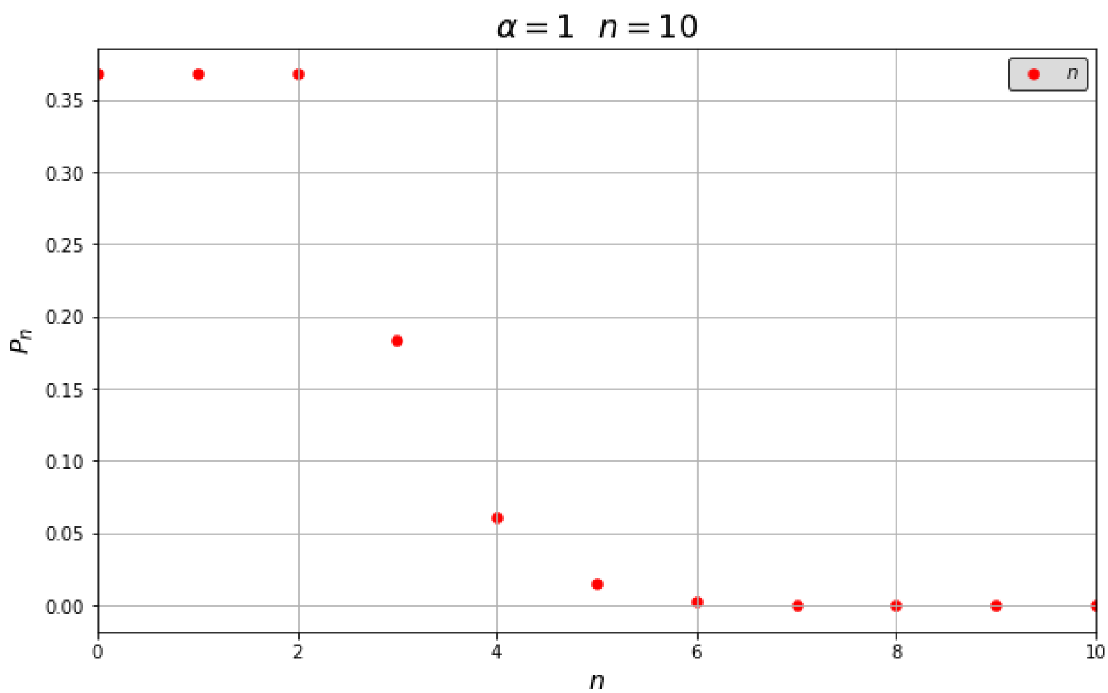
$$\rho_n = |\langle n|\alpha\rangle|^2 = e^{-|\alpha|^2}\frac{\left(|\alpha|^2\right)^n}{n!}$$

The probability of finding n photons in a coherent state is given by the Poisson distribution with mean $|\alpha|^2$.

[65]:
```python
def pcsnp_dist(alpha,n):
    Aux = np.array([np.exp(-np.power(np.absolute(alpha),2))*\
    (np.power(np.power(alpha,2),0))/(np.math.factorial(0))])

    for i in np.arange(len(n)-1):
        Pn = np.exp(-np.power(np.absolute(alpha),2))*\
        (np.power(np.power(alpha,2),np.floor(n[i])))/(np.math.factorial(np.
↪floor(n[i])))

        Aux = np.append(Aux,Pn)

    return Aux
```

[62]:
```python
alpha = 1
n = 10
n_points = np.linspace(start=0,stop=n,num=n+1)

plt.scatter(n_points,pcsnp_dist(alpha,n_points) , color = 'red' , linewidth = 2␣
↪,
         marker = 'o' , s = 20 ,
         label = r'$n$')
plt.title(r'$\alpha = $' + str(alpha) + r'$\ \ n = $' + str(n), fontsize = 18 ,␣
↪color = 'black')
plt.ylabel(r'$P_n$' , fontsize = 14 , color = 'black')
plt.xlabel(r'$n$' , fontsize = 14, color = 'black')
plt.yticks(c = 'black')
plt.xticks(color = 'black')
plt.xlim(0,n)
plt.legend(loc='best' , facecolor = 'lightgray' , edgecolor = 'black')
plt.grid()
plt.show()
```

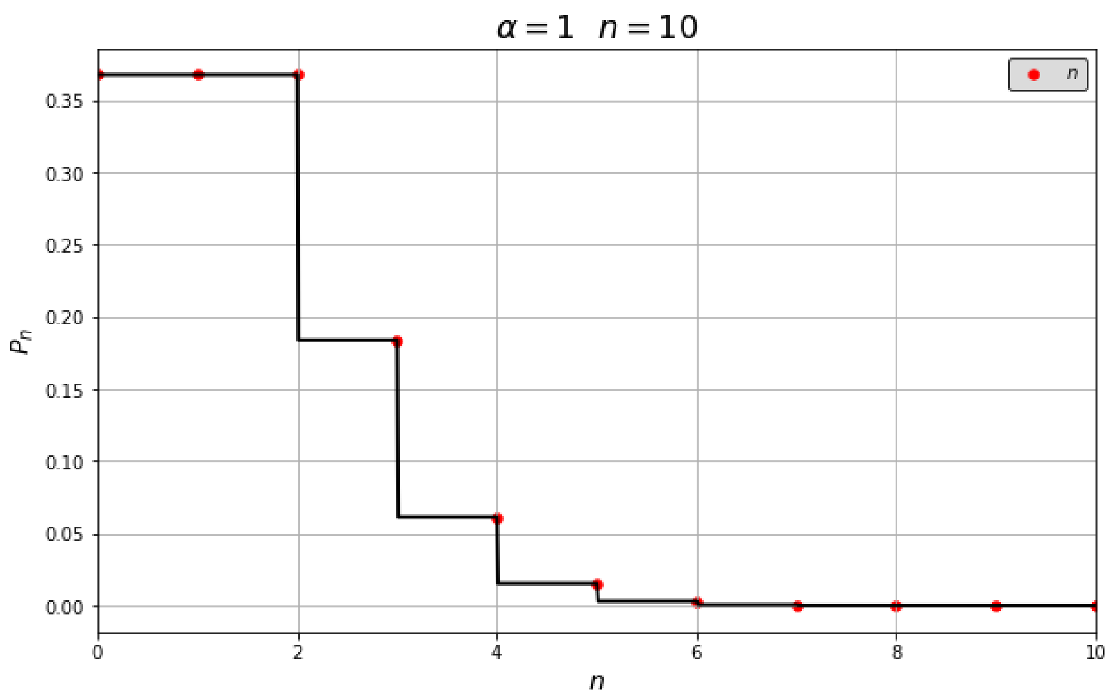
[62]:



[53]:
```python
alpha = 1
n = 10
lon = 1000
dist = np.linspace(start=0,stop=n, num=lon)
n_points = np.linspace(start=0,stop=n,num=n+1)
```

```
Aux_dist = np.array([np.exp(-np.power(np.absolute(alpha),2))*\
    (np.power(np.power(alpha,2),0))/(np.math.factorial(0))])
Aux_n = Aux_dist


for i in np.arange(len(n_points)-1):
    Pn = np.exp(-np.power(np.absolute(alpha),2))*\
    (np.power(np.power(alpha,2),np.floor(n_points[i])))/(np.math.factorial(np.
 ↪floor(n_points[i])))
    Aux_n = np.append(Aux_n,Pn)

for i in np.arange(len(dist)-1):
    Pn = np.exp(-np.power(np.absolute(alpha),2))*\
    (np.power(np.power(alpha,2),np.floor(dist[i])))/(np.math.factorial(np.
 ↪floor(dist[i])))
    Aux_dist = np.append(Aux_dist,Pn)

plt.scatter(n_points,Aux_n , color = 'red' , linewidth = 2 ,
         marker = 'o' , s = 20 ,
         label = r'$n$')
plt.plot(dist,Aux_dist , color = 'black' , linewidth = 2)
plt.title(r'$\alpha = $' + str(alpha) + r'$\ \ n = $' + str(n), fontsize = 18 ,␣
 ↪color = 'black')
plt.ylabel(r'$P_n$' , fontsize = 14 , color = 'black')
plt.xlabel(r'$n$' , fontsize = 14, color = 'black')
plt.yticks(c = 'black')
plt.xticks(color = 'black')
plt.xlim(0,n)
plt.legend(loc='best' , facecolor = 'lightgray' , edgecolor = 'black')
plt.grid()
plt.show()
```

[53]:

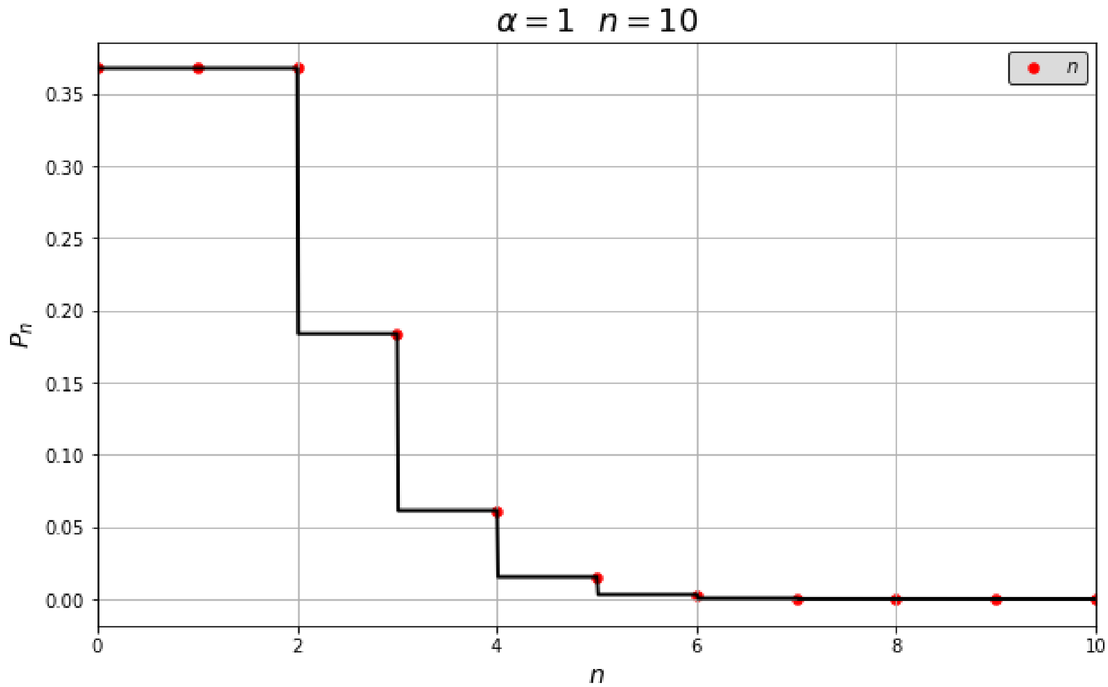


[56]:

```
alpha = 1
n = 10
lon = 1000
dist = np.linspace(start=0,stop=n, num=lon)
n_points = np.linspace(start=0,stop=n,num=n+1)

Aux_dist = np.array([np.exp(-np.power(np.absolute(alpha),2))*\
    (np.power(np.power(alpha,2),0))/(np.math.factorial(0))])
Aux_n = Aux_dist


plt.scatter(n_points,pcsnp_dist(alpha,n_points) , color = 'red' , linewidth = 2␣
 ↪,
         marker = 'o' , s = 20 ,
         label = r'$n$')
plt.plot(dist,pcsnp_dist(alpha,dist) , color = 'black' , linewidth = 2)
plt.title(r'$\alpha = $' + str(alpha) + r'$\ \ n = $' + str(n), fontsize = 18 ,␣
 ↪color = 'black')
plt.ylabel(r'$P_n$' , fontsize = 14 , color = 'black')
plt.xlabel(r'$n$' , fontsize = 14, color = 'black')
plt.yticks(c = 'black')
plt.xticks(color = 'black')
plt.xlim(0,n)
plt.legend(loc='best' , facecolor = 'lightgray' , edgecolor = 'black')
```

```
plt.grid()
plt.show()
```

[56]:

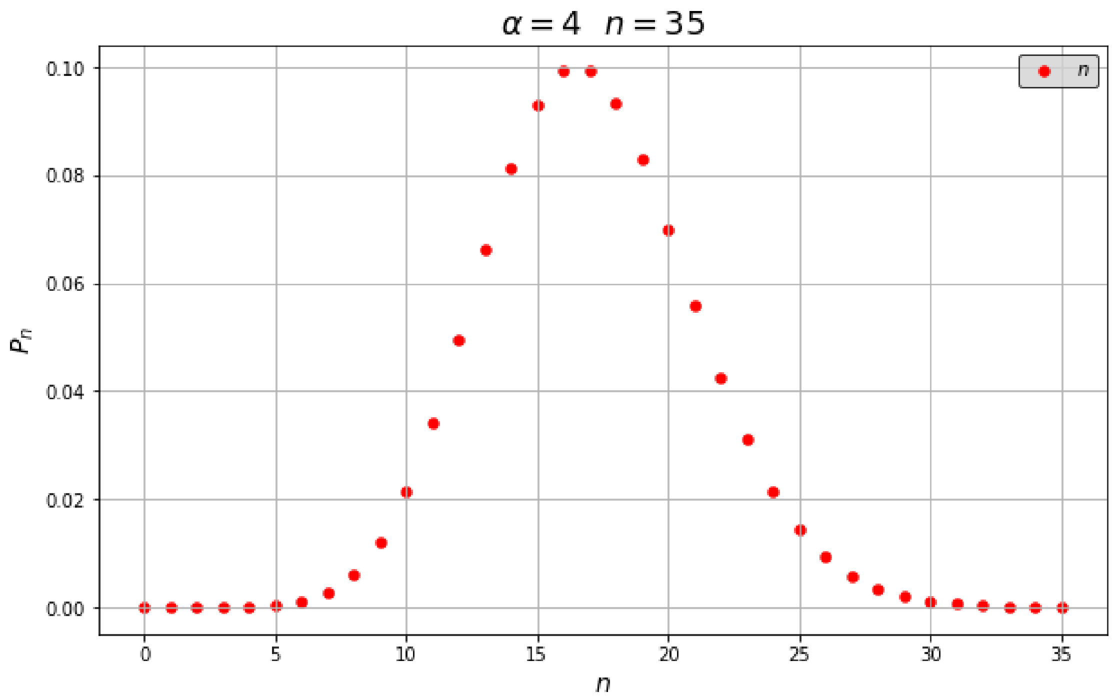


[67]:

```
alpha = 4
n = 35
lon = 1000
dist = np.linspace(start=0,stop=n, num=lon)
n_points = np.linspace(start=0,stop=n,num=n+1)

Aux_dist = np.array([np.exp(-np.power(np.absolute(alpha),2))*\
    (np.power(np.power(alpha,2),0))/(np.math.factorial(0))])
Aux_n = Aux_dist


plt.scatter(n_points,pcsnp_dist(alpha,n_points) , color = 'red' , linewidth = 2␣
↪,
         marker = 'o' , s = 20 ,
         label = r'$n$')
plt.title(r'$\alpha = $' + str(alpha) + r'$\ \ n = $' + str(n), fontsize = 18 ,␣
↪color = 'black')
plt.ylabel(r'$P_n$' , fontsize = 14 , color = 'black')
plt.xlabel(r'$n$' , fontsize = 14, color = 'black')
plt.yticks(c = 'black')
plt.xticks(color = 'black')
```

```
plt.legend(loc='best' , facecolor = 'lightgray' , edgecolor = 'black')
plt.grid()
plt.show()
```

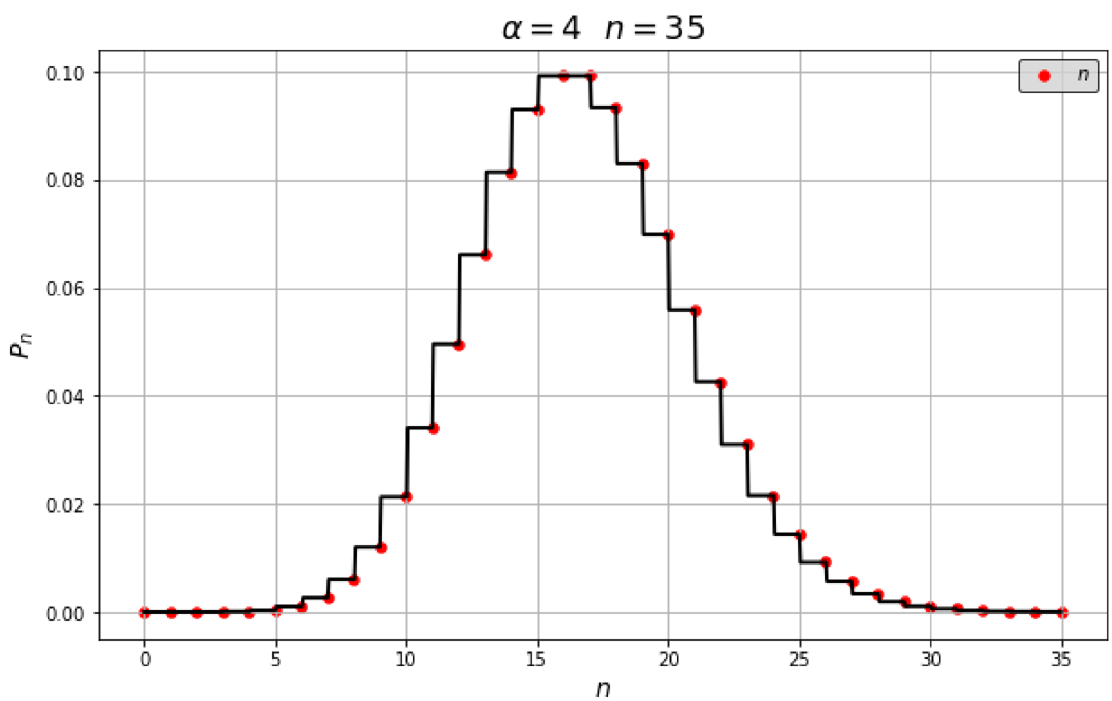
[67]:




```
[66]: alpha = 4
n = 35
lon = 1000
dist = np.linspace(start=0,stop=n, num=lon)
n_points = np.linspace(start=0,stop=n,num=n+1)

Aux_dist = np.array([np.exp(-np.power(np.absolute(alpha),2))*\
    (np.power(np.power(alpha,2),0))/(np.math.factorial(0))])
Aux_n = Aux_dist


for i in np.arange(len(n_points)-1):
    Pn = np.exp(-np.power(np.absolute(alpha),2))*\
    (np.power(np.power(alpha,2),n_points[i]))/(np.math.factorial(n_points[i]))
    Aux_n = np.append(Aux_n,Pn)

for i in np.arange(len(dist)-1):
    Pn = np.exp(-np.power(np.absolute(alpha),2))*\
    (np.power(np.power(alpha,2),np.floor(dist[i])))/(np.math.factorial(np.
↪floor(dist[i])))
```

```
    Aux_dist = np.append(Aux_dist,Pn)

plt.scatter(n_points,Aux_n , color = 'red' , linewidth = 2 ,
        marker = 'o' , s = 20 ,
        label = r'$n$')
plt.plot(dist,Aux_dist , color = 'black' , linewidth = 2)
plt.title(r'$\alpha = $' + str(alpha) + r'$\ \ n = $' + str(n), fontsize = 18 ,␣
 ↪color = 'black')
plt.ylabel(r'$P_n$' , fontsize = 14 , color = 'black')
plt.xlabel(r'$n$' , fontsize = 14, color = 'black')
plt.yticks(c = 'black')
plt.xticks(color = 'black')
plt.legend(loc='best' , facecolor = 'lightgray' , edgecolor = 'black')
plt.grid()
plt.show()
```

[66]:



[44]:
```
len(Aux)
```

[44]: 100

0.1.2 Photon-Statistics of Coherent-States

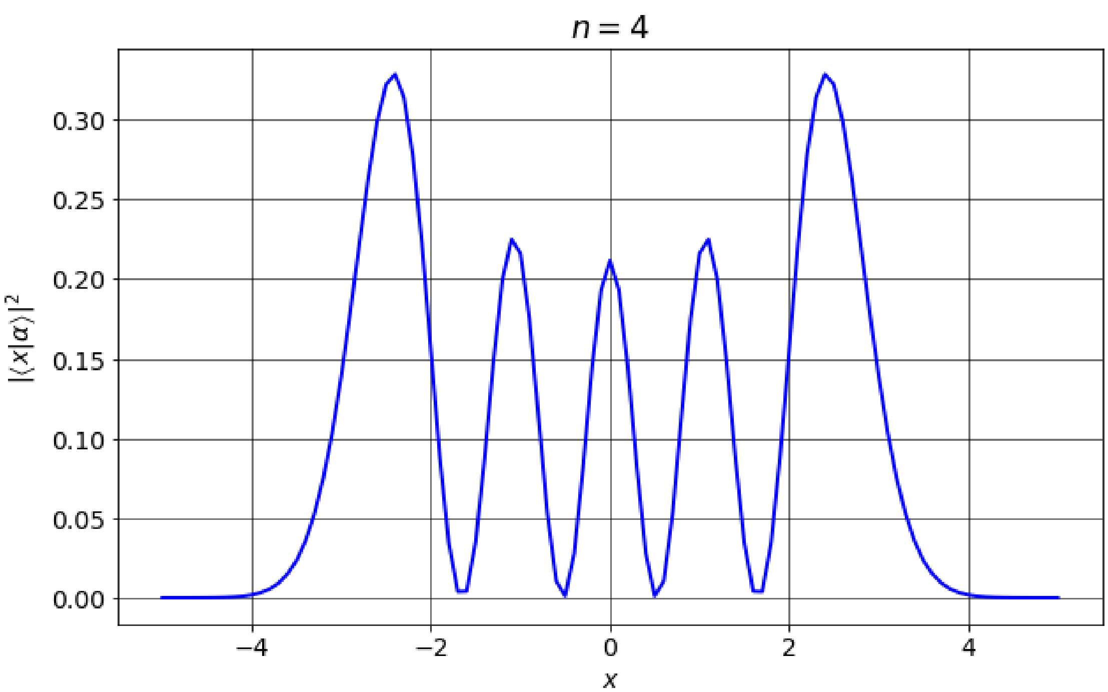
$$|\langle x|\alpha\rangle|^2 = \frac{1}{\sqrt{2^n n! \sqrt{\pi}}} H_n(x) e^{-\frac{1}{2}x^2}$$

Where $H_n(x)$ are the *Hermite polynomials*.

[5]:
```python
def pscstf(x,n):
    BK = (1/np.sqrt(np.power(2,n)*np.math.factorial(n)*np.sqrt(np.pi)))*\
    hn(n,x)*\
    np.exp(-0.5*pow(x,2))

    S = np.power(np.absolute(BK),2)

    return S
```

[68]:
```python
n = 4
x = np.linspace(start=-5, stop=5, num=101)

plt.plot(x, pscstf(x,n) , color = 'blue' , linewidth = 2)
plt.title(r'$n = $' + str(n), fontsize = 18 , color = 'black')
plt.ylabel(r'$\left| \left< x|\alpha \right> \right|^{2} $' , fontsize = 14 ,␣
 ↪color = 'black')
plt.xlabel(r'$x$' , fontsize = 14, color = 'black')
plt.yticks(color = 'black', fontsize = 14)
plt.xticks(color = 'black', fontsize = 14)
plt.grid(color = 'black' , alpha = 0.7)
plt.show()
```

[68]:



[0]:
```python
```