HW1

January 31, 2020

1 Chem 30324, Spring 2020, Homework 1

2 Due on January 22, 2020

2.1 Problem 1: Discrete, probably

In five card study, a poker player is dealt five cards from a standard deck of 52 cards.

2.1.1 1. How many different 5-card hands are there? (Remember, in poker the order in which the cards are received does *not* matter.)

```
In [6]: import math
        def Con(n,j):
            f=math.factorial
            z=f(n)/f(n-j)/f(j)
            return z
        print ('There are %d different 5 card hands.'%Con(52,5))
```

```
There are 2598960 different 5 card hands.
```

2.1.2 2. What is the probability of being dealt four of a kind (a card of the same rank from each suit)?

```
In [7]: p2 = Con(13,1)*Con(48,1)/Con(52,5)
        print ('The probability of being dealt four of a kind is %.7f' % p2)
```

```
The probability of being dealt four of a kind is 0.0002401
```

2.1.3 3. What is the probability of being dealt a flush (five cards of the same suit)?

```
In [8]: p3 = Con(4,1)*Con(13,5)/Con(52,5)
        print ('The probability of being dealt a flush is %.6f' % p3)
```

```
The probability of being dealt a flush is 0.001981
```

2.2 Problem 2: Continuous, probably

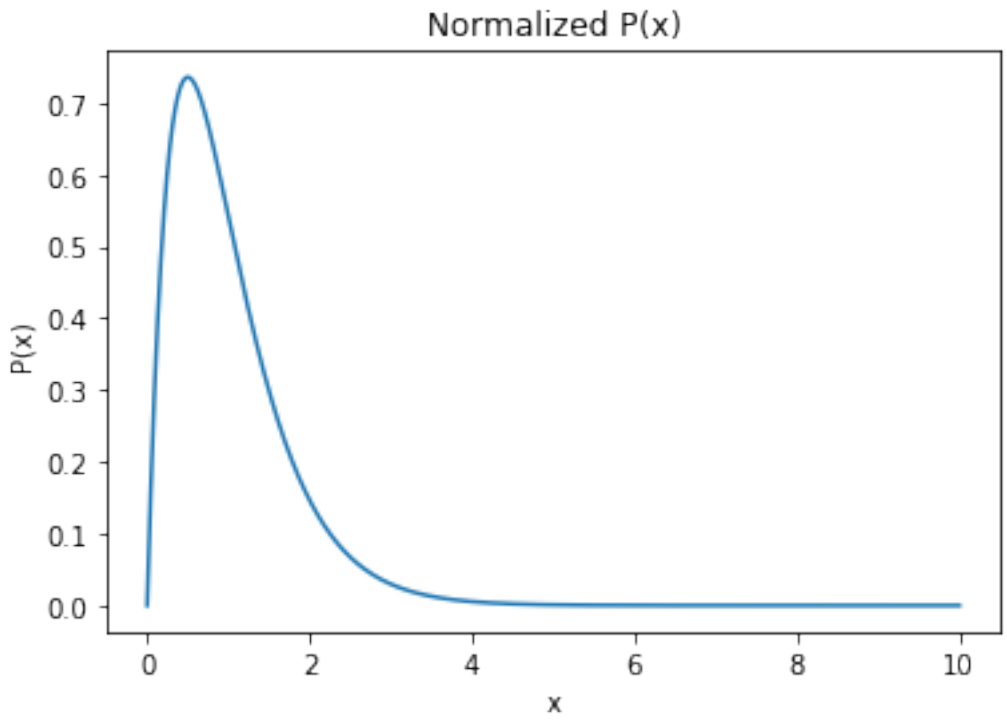
The probability distribution function for a random variable x is given by $P(x) = xe^{-2x}, 0 \leq x < \infty$.

2.2.1 1. Is $P(x)$ normalized? If not, normalize it. Plot the normalized $P(x)$.

In [9]:
```
import numpy as np
import scipy.integrate as integrate
import matplotlib.pyplot as plt

def ditr(x):
    return x*np.exp(-2*x)

FP = integrate.quad(ditr, 0, np.inf)
if FP == 1:
    print('P(x) is normalized')
else:
    print('P(x) is not normlaized')

x0 = np.arange(0, 10, 0.0001)
y0 = ditr(x0) / FP[0]
plt.plot(x0, y0)
plt.title('Normalized P(x)')
plt.xlabel('x')
plt.ylabel('P(x)')
```

```
P(x) is not normlaized
```

Out[9]:
```
Text(0, 0.5, 'P(x)')
```



2.2.2 2. What is the most probable value of x?

```
In [10]: import scipy
         max_x = scipy.optimize.fmin(lambda x: -ditr(x),0)
         print ('The most probable value of x is %.1f' %max_x)

Optimization terminated successfully.
         Current function value: -0.183940
         Iterations: 23
         Function evaluations: 46
The most probable value of x is 0.5
```

2.2.3 3. What is the expectation value of x?

```
In [11]: def editr(x):
             return x*x*np.exp(-2*x)/FP[0]
         expect_x = integrate.quad(editr, 0, np.inf)
         print('The expectation value of x is', expect_x[0])

The expectation value of x is 1.0000000000000007
```

2.2.4 4. What is the variance of x?

```
In [12]: def vditr(x):
             return x*x*x*np.exp(-2*x)/FP[0]
         second_moment = integrate.quad(vditr, 0, np.inf)
         print('The variance of x is', second_moment[0]-expect_x[0]**2)
```

```
The variance of x is 0.49999999999999933
```

2.3 Problem 3: One rough night

It's late on a Friday night and people are stumbling up Notre Dame Ave. to their dorms. You observe one particularly impaired individual who is taking steps of equal length 1m to the north or south (i.e., in one dimension), with equal probability.

2.3.1 1. What is the furthest distance the person could travel after 20 steps?

The furthest distance is 20m south or north.

2.3.2 2. What is the probability that the person won't have traveled any net distance at all after 20 steps?

```
In [13]: import math
         f = math.factorial
         p_0 = f(20)/f(10)/f(10)/(2**20)
         print(p_0)
```

```
0.17619705200195312
```

2.3.3 3. What is the probability that the person has traveled half the maximum distance after 20 steps?

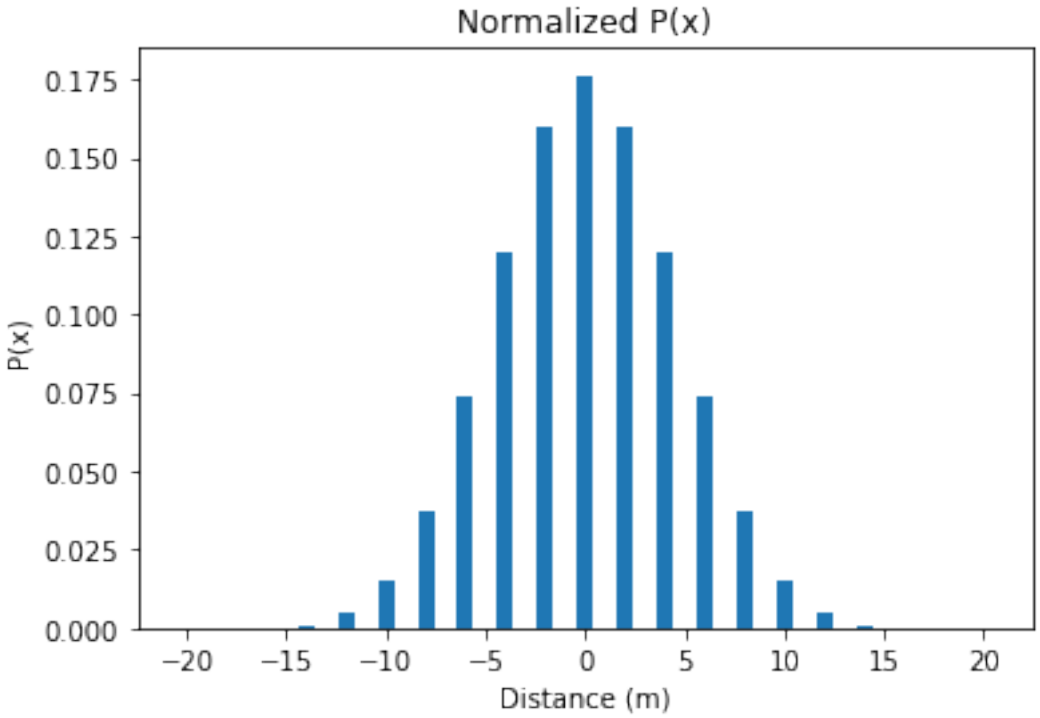
```
In [4]: p_h = f(20)/f(15)/f(5)*2/(2**20)
        print(p_h)
```

```
0.029571533203125
```

2.3.4 4. Plot the probability of traveling a given distance vs distance. Does the probability distribution look familiar? You'll see it again when we talk about diffusion.

```
In [16]: import matplotlib.pyplot as plt
         import numpy as np

         pds = np.array([])
         ds = np.array([])
         d = -20
```

```
while d < 21:
    if d % 2 == 1:
        pds = np.append(pds, 0)
        ds = np.append(ds, d)
    else:
        p = f(20)/f(10+abs(d)/2)/f(10-abs(d)/2)/(2**20)
        pds = np.append(pds,p)
        ds = np.append(ds,d)
    d = d + 1
plt.bar(ds, pds)
plt.title('Normalized P(x)')
plt.xlabel('Distance (m)')
plt.ylabel('P(x)')
plt.show()
```



2.4 Problem 4: Now this is what I call equilibrium

The Boltzmann distribution tells us that, at thermal equilibrium, the probability of a particle having an energy E is proportional to $\exp(-E/k_B T)$, where k_B is the Boltzmann constant. Suppose a bunch of gas particles of mass m are in thermal equilibrium at temperature T and are traveling back and forth in one dimension with various velocities v and kinetic energies $K = mv^2/2$.

```
In [3]: from sympy import *
        v = Symbol('x')
```

2.4.1 1. What is the expectation value of the velocity v of a particle?

```
In [2]: from sympy import *
        v = Symbol('x')
        m = Symbol('m', positive = True)
        kB = Symbol('kB', positive = True)
        T = Symbol('T', positive = True)
        FP = integrate((exp(-m*v**2/2/kB/T)),(v,-oo,+oo))
        expt_v=integrate((v*exp(-m*v**2/2/kB/T)/FP),(v,-oo,+oo))
        print(expt_v)
```

```
0
```

2.4.2 2. What is the expectation value of the kinetic energy K of a particle? How does your answer depend on the particle mass? On temperature?

```
In [4]: expt_K=integrate((m*v**2/2*exp(-m*v**2/2/kB/T)/FP),(v,-oo,+oo))
        print(expt_K)
```

```
T*kB/2
0.5*T*kB
```

```
In [6]: kB = 1.38e-23
        T = 298
        K = 1/2*kB*T
        print(K)
```

```
2.0562e-21
```