Lab 2

Sierra Wilde and Michael Higgins

Problem 1

I have chosen the average number of background, cosmic ray counts per day to be X = 0.75, and the average number of gamma rays emitted per day to be Y = 2.0.

The Poisson distribution is a discrete distribution that describes the probability of measuring a certain number of counts in an interval of time. The distributions are described by:

$$P_{background}(k) = \frac{e^{-X} X^{-k}}{k!} \; P_{gamma}(k) = \frac{e^{-Y} Y^{-k}}{k!}$$

where k is the number of counts.

In [1]:

```python
import numpy as np
import scipy as sp
import matplotlib.pyplot as plt
from scipy import stats
```

A.

In order to see how the probability distribution of the background changes as more days are added together, the Poisson distribution with X = 0.75 is convloved with itself multiple times. The convolution of two background distributions are:

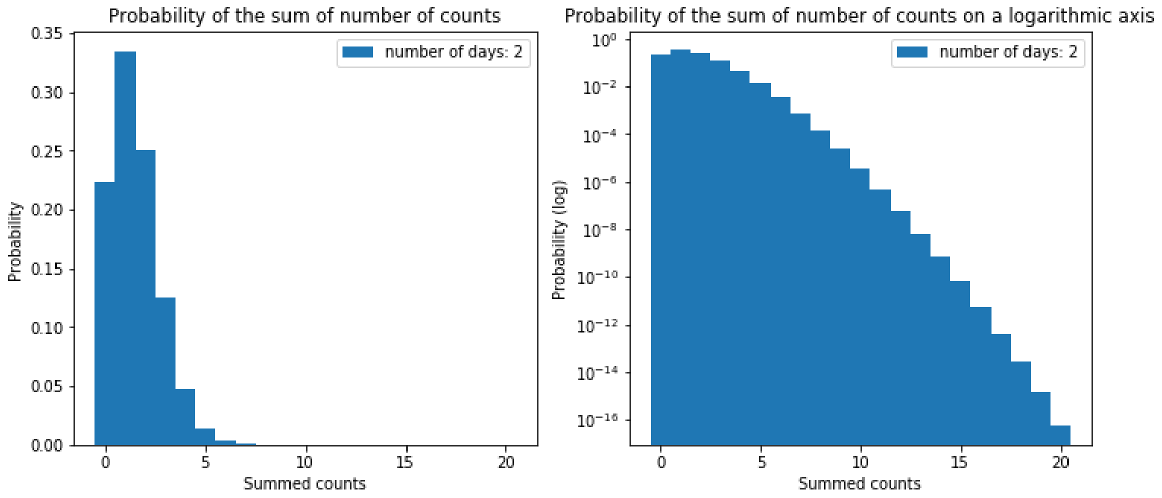
$$P_{sum}(x) = P_{background}(x)^{*} P_{background}(x) = \int P_{background}(k) P_{background}(x-k)dk$$

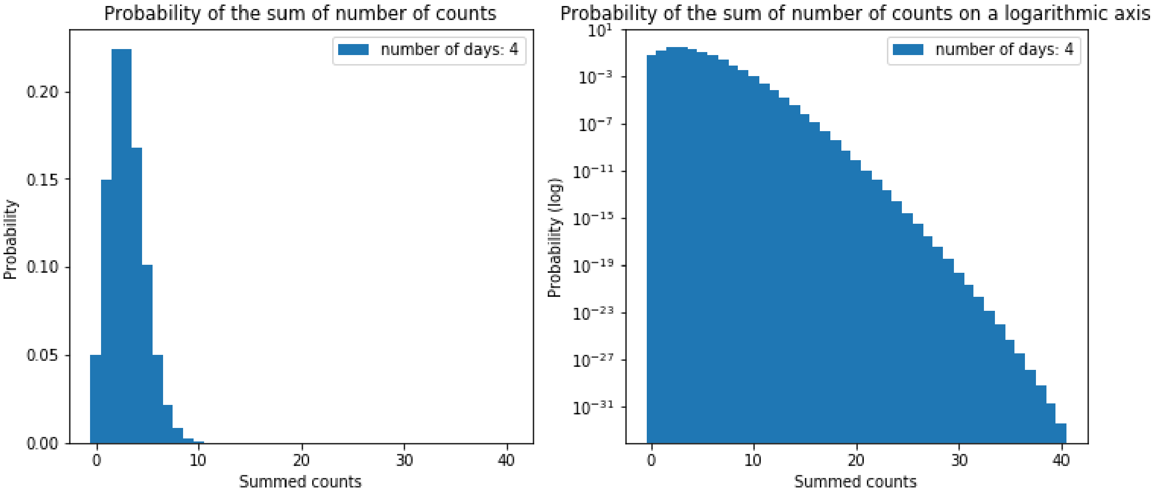
The convolution of more than two distributions follows the form above, only switching one of the $P_{background}(x)$ with the previous convoluted distribution.

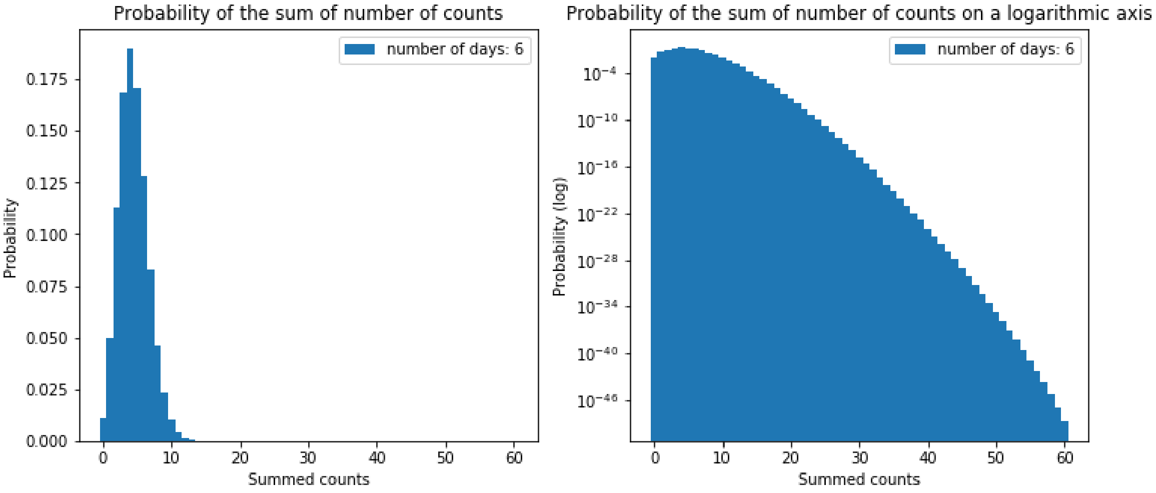
Convolving the background Poisson distribution multiple times in order to obtain the sum of the background over multiple days:

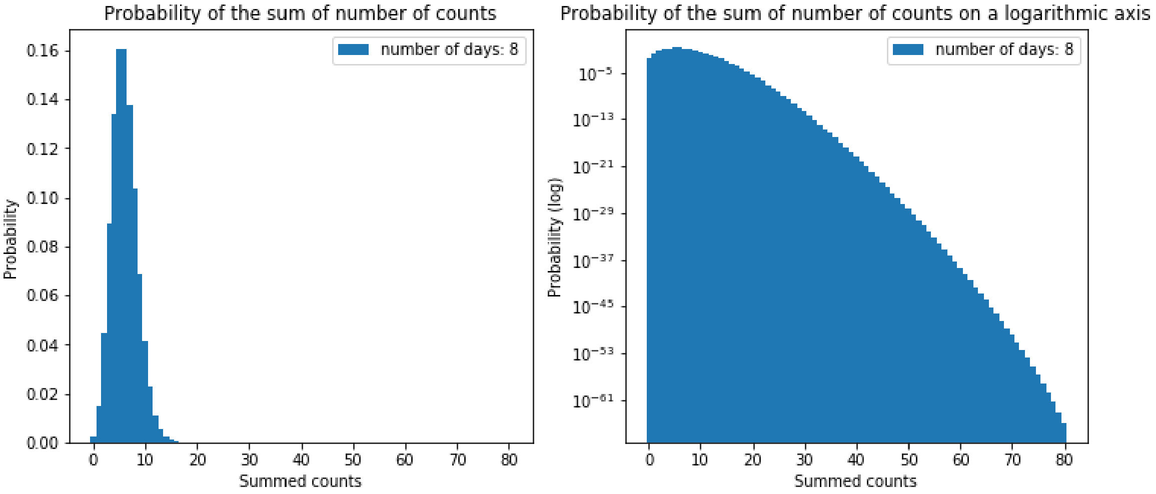
In [2]:

```
days = [2, 4, 6, 8]
k = np.arange(0, 11)
for d in days:
    #making convolutions according to the number of days over which the Poiss
    sums = np.arange(0, len(k)*d-(d-1))
    #making a list of the summed values that can take place given the number
    for i in range(0, d-1):
        if i==0:
            new_pmf = stats.poisson.pmf(k, 0.75)
        new_pmf = np.convolve(stats.poisson.pmf(k, 0.75), new_pmf)
    avg_pmf = new_pmf
    #plotting the background convolutions
    fig, axes = plt.subplots(1, 2, figsize=(12, 5))
    for ax in axes:
        ax.bar(sums, avg_pmf, label='number of days: {}'.format(d), width=sum
        ax.set_xlabel('Summed counts')
        ax.legend()
    axes[0].set_title('Probability of the sum of number of counts')
    axes[0].set_ylabel('Probability')
    axes[1].set_title('Probability of the sum of number of counts on a logari
    axes[1].set_ylabel('Probability (log)')
    axes[1].set_yscale('log')
plt.show()
```




Probability of the sum of number of counts
number of days: 4
Probability
0.00
0.05
0.10
0.15
0.20
0
10
20
30
40
Summed counts
Probability of the sum of number of counts on a logarithmic axis
number of days: 4
Probability (log)
10^{1}
10^{-3}
10^{-7}
10^{-11}
10^{-15}
10^{-19}
10^{-23}
10^{-27}
10^{-31}
0
10
20
30
40
Summed counts


Probability of the sum of number of counts
number of days: 6
Probability
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0
10
20
30
40
50
60
Summed counts
Probability of the sum of number of counts on a logarithmic axis
number of days: 6
Probability (log)
10^{-4}
10^{-10}
10^{-16}
10^{-22}
10^{-28}
10^{-34}
10^{-40}
10^{-46}
0
10
20
30
40
50
60
Summed counts


Probability of the sum of number of counts
number of days: 8
Probability
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
0
10
20
30
40
50
60
70
80
Summed counts
Probability of the sum of number of counts on a logarithmic axis
number of days: 8
Probability (log)
10^{-5}
10^{-13}
10^{-21}
10^{-29}
10^{-37}
10^{-45}
10^{-53}
10^{-61}
0
10
20
30
40
50
60
70
80
Summed counts

The width of the distribution decreases as the number of times the distribution is summed with itself increases. The distribution looks more Gaussian, that is more bell-shaped, but the logarithm of the distribution does not look like a parabola, which is how the logarithm of a true Gaussian distribution is characterized.

A Gaussian distribuition is defined by:

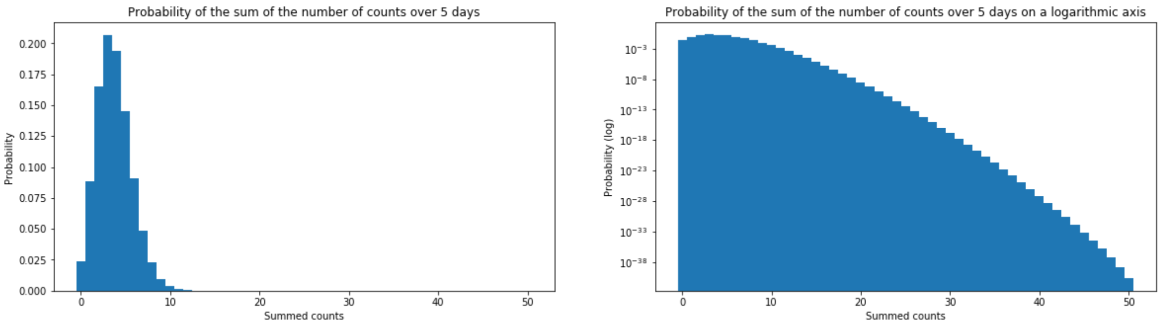
$P(x) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{\frac{(x-\mu)^2}{2\sigma^2}}$, where μ is the mean and σ is the standard deviation.

B.

Plot of the probability distribution for the average number of backgrounds counts over 5 days

In [3]:

```
day = 5
sum = np.arange(0, len(k)*day-(day-1))
#making a list of the summed values that can take place given the number of d
for i in range(0, day-1):
    if i==0:
        new_pmf = stats.poisson.pmf(k, 0.75)
    new_pmf = np.convolve(stats.poisson.pmf(k, 0.75), new_pmf)
avg_pmf = new_pmf
#plotting the background convolutions
fig, axes = plt.subplots(1, 2, figsize=(20, 5))
for ax in axes:
    ax.bar(sum, avg_pmf, width=sum[1]-sum[0])
    ax.set_xlabel('Summed counts')
axes[0].set_title('Probability of the sum of the number of counts over 5 days
axes[0].set_ylabel('Probability')
axes[1].set_title('Probability of the sum of the number of counts over 5 days
axes[1].set_ylabel('Probability (log)')
axes[1].set_yscale('log')
plt.show()
print(np.sum(avg_pmf*sum))
```



```
3.7499999625182614
```

Here we can see that indeed this distribution is not Gaussian. The right tail is more prominent than the left tail, and a Gaussian is symmetric about its mean. When looking at the logarithmic plot, it is very obvious that this is not Gaussian: the plot deviates considerably from a parabola.

The sum of two Poisson distributions:

$P_1(k) = \frac{e^{-\lambda_1}\lambda_1^{k}}{k!}$ $P_2(k) = \frac{e^{-\lambda_2}\lambda_2^{k}}{k!}$ where λ_1 and λ_2 are the expected count numbers respectively for each distribution.

$P(k)^*P(k) = \sum_j^k\frac{e^{-\lambda_1}\lambda_1^j}{j!}\frac{e^{-\lambda_2}\lambda_2^{k-j}}{(k-j)!}$

$P(k)^*P(k) = \sum_j^k\frac{k!}{j!(k-j)!}\frac{e^{-\lambda_1}\lambda_1^je^{-\lambda_2}\lambda_2^{k-j}}{k!}$

Using binomial coefficients:

$P(k)^*P(k) = \sum_j^k\binom{k}{j}\frac{e^{-\lambda_1}\lambda_1^je^{-\lambda_2}\lambda_2^{k-j}}{k!}$

setting $\lambda = \lambda_1+\lambda_2$:

$P(k)^*P(k) = \frac{e^{-\lambda}}{k!}\sum_j^k\binom{k}{j}\lambda_1^j\lambda_2^{k-j}$

Factoring $\binom{k}{j}\lambda_1^j\lambda_2^{k-j}$:

$P(k)^*P(k) = \frac{e^{-\lambda}}{k!}(\lambda_1+\lambda_2)^k$

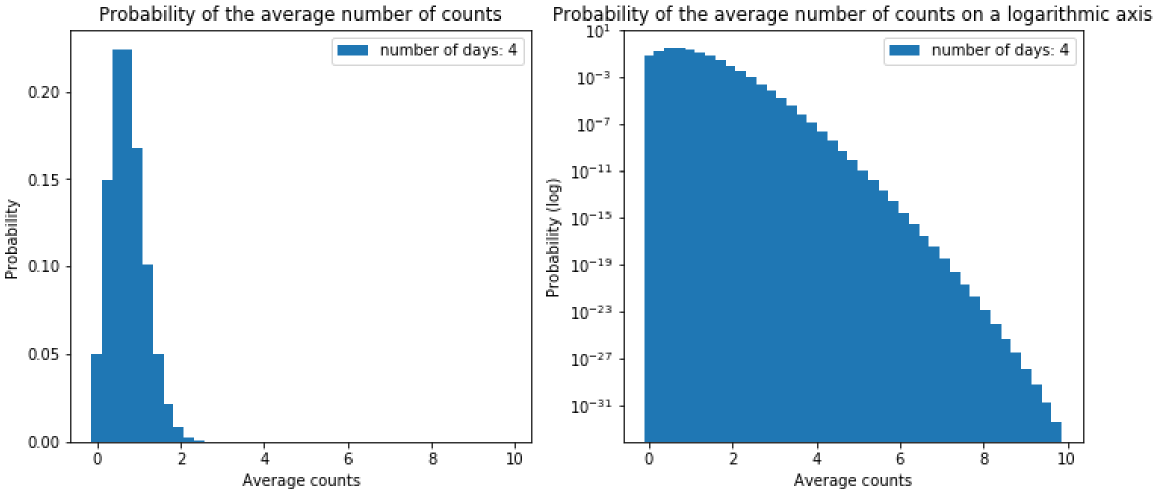
$P(k)^*P(k) = \frac{e^{-\lambda}\lambda^k}{k!}$

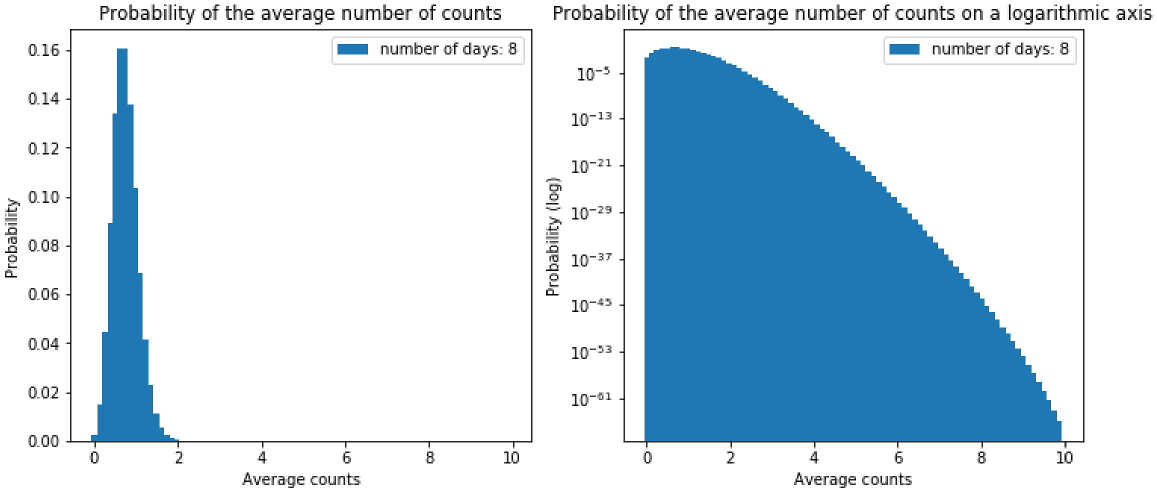
This is a new Poisson distribution with a new expected value equal to the sum of the old expected values. The above proof can be extrapolated for summing 5 Poisson distributions. The summed distribution will look like another Poisson distribution with a new expected value equal to the sum of the original expected values. This makes sense because summing multiple distributions should give a greater expected value. The total number of cosmic rays counted over five days should give a value of about five times the expected value, which is consistant with the above proof if the λ's are all the same.

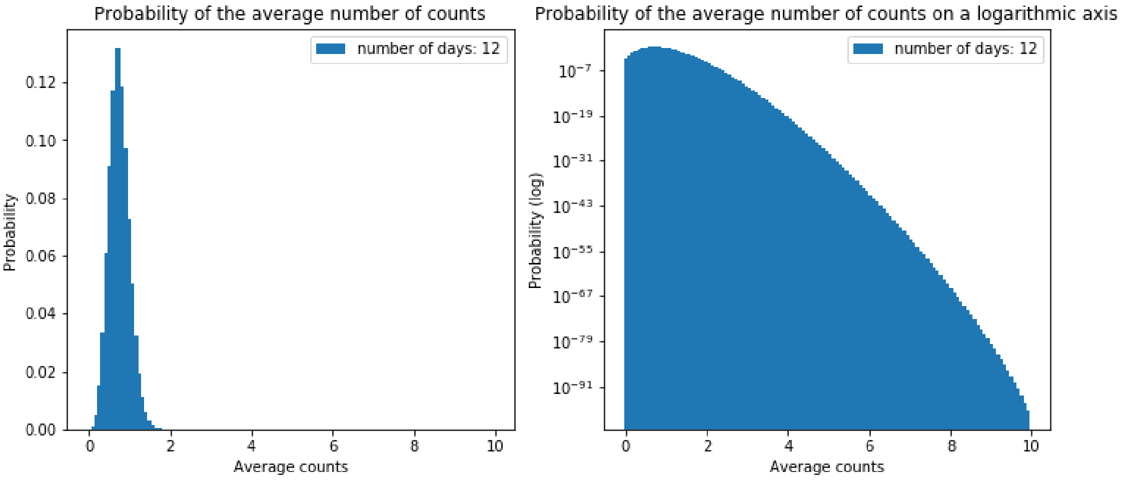
C.

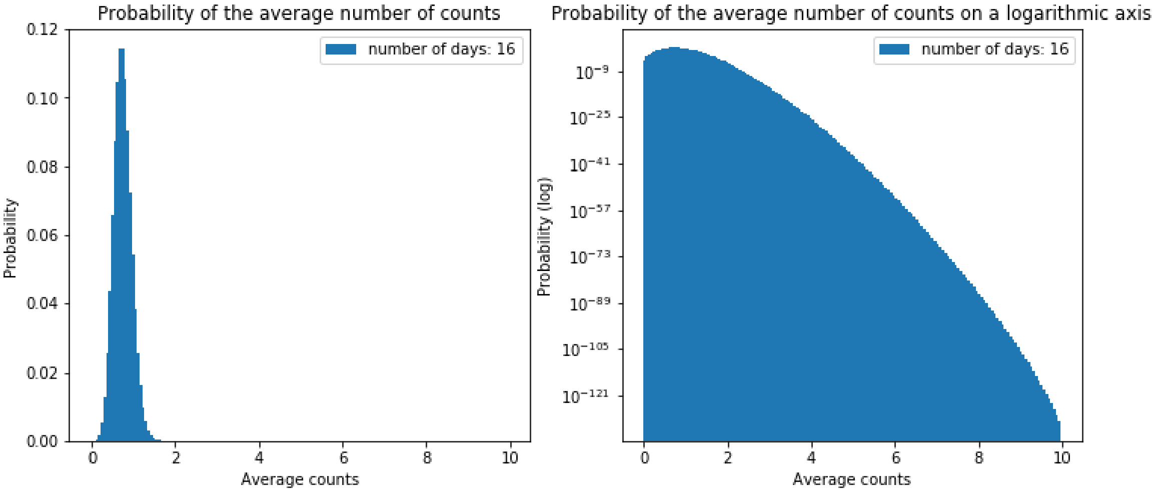
In [4]:

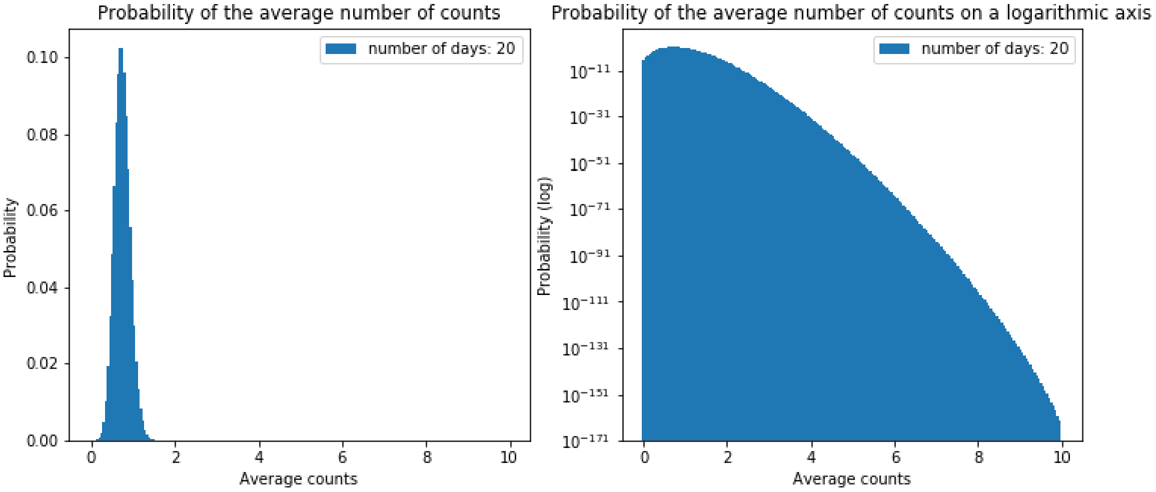
```
more_days = [4, 8, 12, 16, 20]
for d in more_days:
    #making convolutions according to the number of days over which the Poiss
    avg = 10*np.arange(0, len(k)*d-(d-1))/len(np.arange(0, len(k)*d-(d-1)))
    #making a list of the average values that can take place given the number
    for i in range(0, d-1):
        if i==0:
            new_pmf = stats.poisson.pmf(k, 0.75)
        new_pmf = np.convolve(stats.poisson.pmf(k, 0.75), new_pmf)
    avg_pmf = new_pmf
    #plotting the background convolutions
    fig, axes = plt.subplots(1, 2, figsize=(12, 5))
    for ax in axes:
        ax.bar(avg, avg_pmf, label='number of days: {}'.format(d), width=avg[
        ax.set_xlabel('Average counts')
        ax.legend()
    axes[0].set_title('Probability of the average number of counts')
    axes[0].set_ylabel('Probability')
    axes[1].set_title('Probability of the average number of counts on a logar
    axes[1].set_ylabel('Probability (log)')
    axes[1].set_yscale('log')
plt.show()
```











The probabiity distribution of the average number of counts over up to 20 days still appears to follow the Poisson distribution. Even though the distribution has started to look more bell-shaped, the logarithmic plot is not parabolic. The average of Poisson distributions follow the same description as the sum of the distributions discussed in part B., but divides the counts by the number of distributions convolved together. The expected value should therefore remain the same in this case, because adding the expected value to itself multiple times and then dividing by that number gives the original expected value. The central limit theorem states that all distributions convolved enough times, will approach a Gaussian distribution. As the number of times a Poisson distribution is averaged, the more symmetrical the distribution appears, and the more parabolic the logarithm of the distribution becomes. After averaging the distribution 20 times, the new distribution looks very symmetric and bell-shaped, but the log plot does not look very parabolic yet. The Poisson distribution needs to be convolved many more times in order to more closely approximate a Gaussian distribution.

D.

N = 150 days, number of gamma rays measured: 300.

Obtaining the summed background distribution:

In [5]:

```
#doing 14 convolutions to sum the number of background counts over 15 days
for i in range(0, 149):
    if i==0:
        new_pmf = stats.poisson.pmf(k, 0.75)
    new_pmf = np.convolve(stats.poisson.pmf(k, 0.75), new_pmf)
#new backgound pmf
background_pmf = new_pmf
background_sum = np.arange(0, len(k)*150-149)
```

In [6]:

```
#obtaining the new expected value of the background
background_avg = np.sum(background_pmf*background_sum)
print('The expected value of the background distribution over 5 days: {}.'.fo
```

```
The expected value of the background distribution over 5 days: 112.49999018
19237.
```

The σ value is associated with the probability that the measurement would have of being background and have that same probability if it were Gaussian at that σ value away from the mean. In order for a signal-like measurement to be significant, it must have a σ of 5 or higher.

$\sigma = 2erf^{-1}(\int_3^{\infty} P_{sum}(k)dk)$, where $P_{sum}(k)$ is the sum of the background distribution over 15 days.

In [7]:

```
sigma = stats.norm.ppf(stats.poisson.cdf(300, background_avg))
print('The sigma value of the measurement of 15 over 5 days is {}'.format(sig
```

```
The sigma value of the measurement of 15 over 5 days is inf
```

Since the σ value is so great, the measurement can be considered significant.

Problem 2

I am picking the chi-squared distribution, which is the sum of multiple squared Gaussian distributions and is given by:

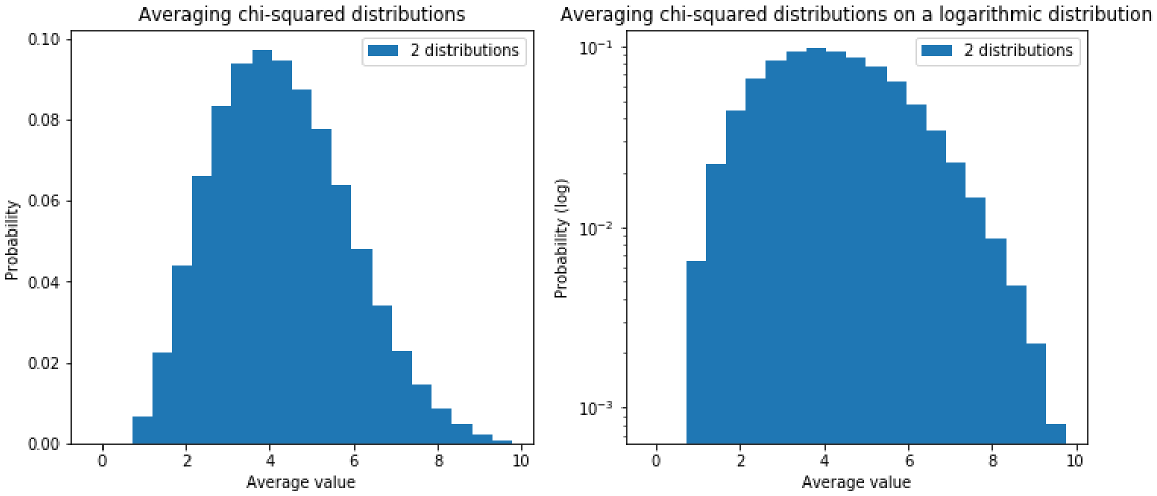
$P_k(x) = \frac{1}{2^{\frac{k}{2}}\Gamma(\frac{k}{2})}x^{\frac{k}{2}-1}e^{\frac{-x}{2}}$, where k is the degrees of freedom, or the number of independent Gaussian distributions squared and summed together.

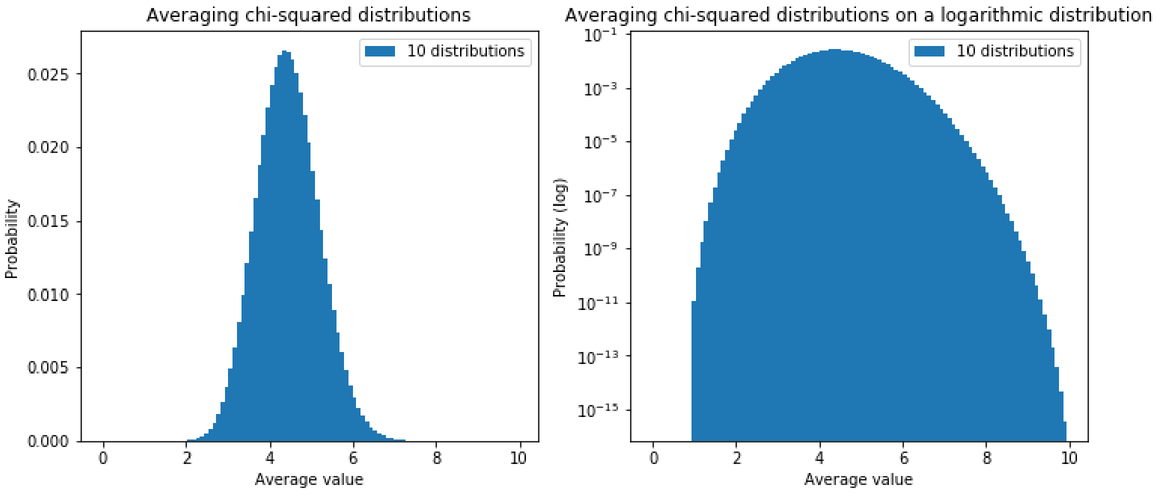
A.

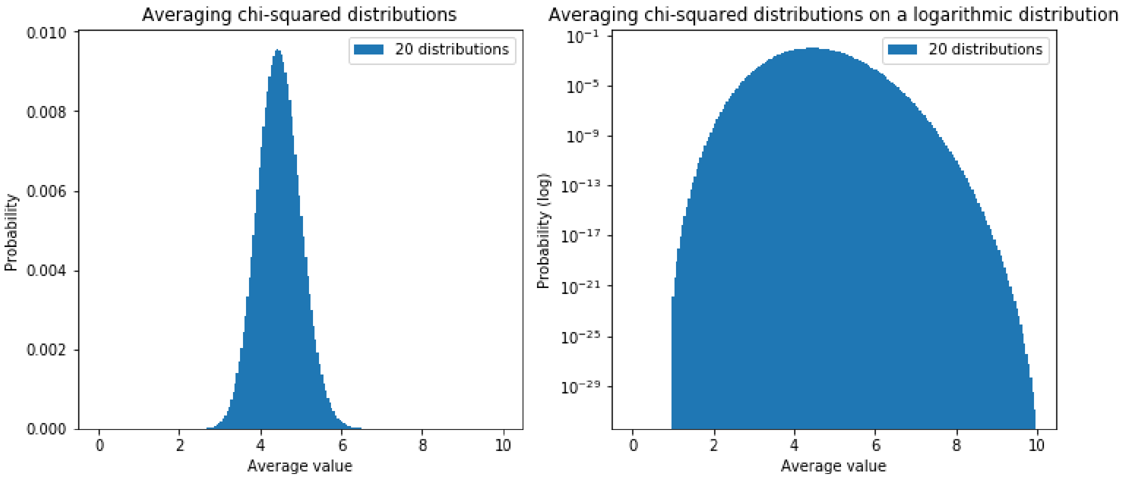
Convolving a chi-squared distribution with 5 degrees of freedom:

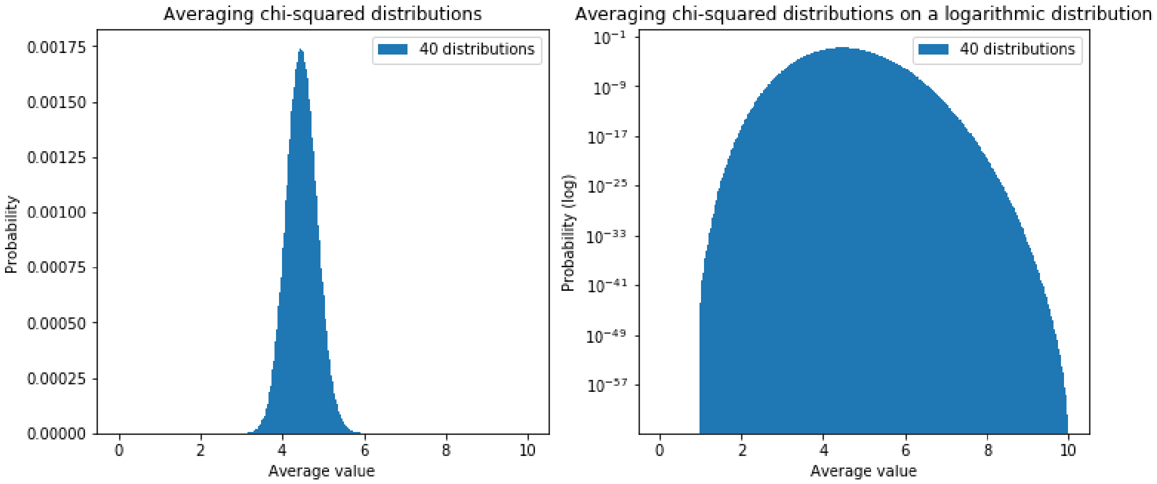
In [8]:

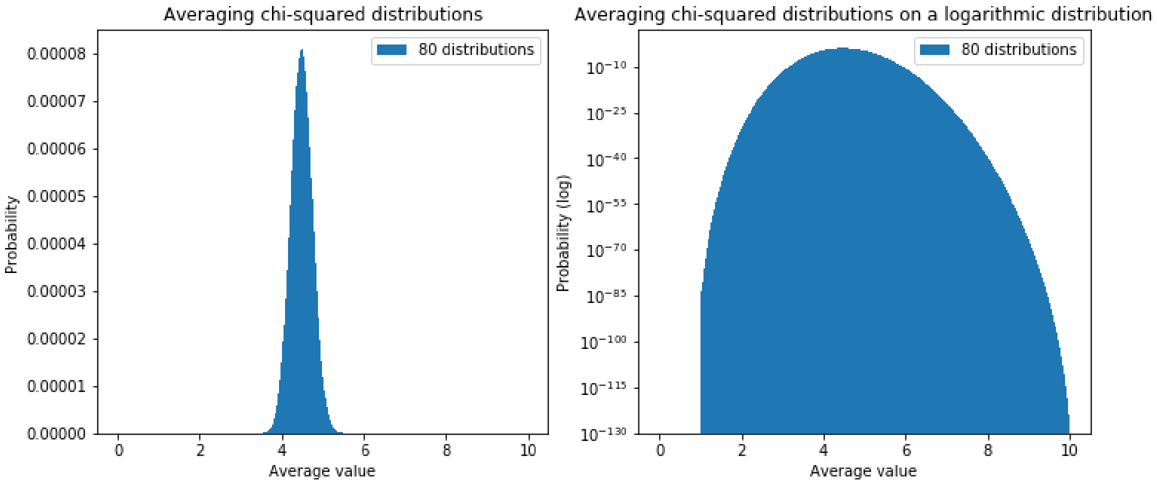
```
iterations = [2, 10, 20, 40, 80, 160]
#making convolutions according to the number of iterations over which to aver
for it in iterations:
    #list of average values from the distribution
    average = 10*np.arange(0, len(k)*it-(it-1))/len(np.arange(0, len(k)*it-(i
    for i in range(0, it-1):
        if i==0:
            new_pdf = stats.chi2.pdf(k, 5)
        new_pdf = np.convolve(stats.chi2.pdf(k, 5), new_pdf)
    avg_pdf = new_pdf
    #plotting the chi-squared convolutions
    fig, axes = plt.subplots(1, 2, figsize=(12, 5))
    for ax in axes:
        ax.bar(average, avg_pdf, label='{} distributions'.format(it), width=a
        ax.set_xlabel('Average value')
        ax.legend()
    axes[0].set_title('Averaging chi-squared distributions')
    axes[0].set_ylabel('Probability')
    axes[1].set_title('Averaging chi-squared distributions on a logarithmic d
    axes[1].set_ylabel('Probability (log)')
    axes[1].set_yscale('log')
plt.show()
```

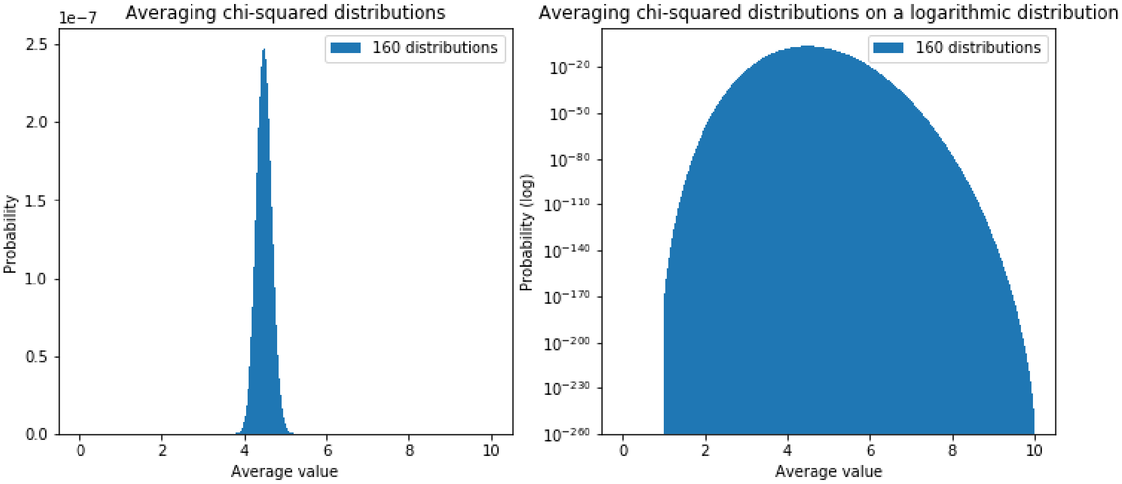





Averaging chi-squared distributions
20 distributions
Probability
Average value
0.010
0.008
0.006
0.004
0.002
0.000
0
2
4
6
8
10
Averaging chi-squared distributions on a logarithmic distribution
20 distributions
Probability (log)
Average value
10^{-1}
10^{-5}
10^{-9}
10^{-13}
10^{-17}
10^{-21}
10^{-25}
10^{-29}
0
2
4
6
8
10


Averaging chi-squared distributions
40 distributions
Probability
Average value
0.00175
0.00150
0.00125
0.00100
0.00075
0.00050
0.00025
0.00000
0
2
4
6
8
10
Averaging chi-squared distributions on a logarithmic distribution
40 distributions
Probability (log)
Average value
10^{-1}
10^{-9}
10^{-17}
10^{-25}
10^{-33}
10^{-41}
10^{-49}
10^{-57}
0
2
4
6
8
10


Averaging chi-squared distributions
80 distributions
Probability
Average value
0.00008
0.00007
0.00006
0.00005
0.00004
0.00003
0.00002
0.00001
0.00000
0
2
4
6
8
10
Averaging chi-squared distributions on a logarithmic distribution
80 distributions
Probability (log)
Average value
10^{-10}
10^{-25}
10^{-40}
10^{-55}
10^{-70}
10^{-85}
10^{-100}
10^{-115}
10^{-130}
0
2
4
6
8
10



B.

As the number of distributions averaged increases, the closer the new pdf approaches a Gaussian distribution. The averaged distribution starts to look pretty symmetrical and bell-shaped after about 10 convolutions, however it takes longer for the logarithmic plot to approach the characteristic parabolic shape. According the the central limit theorm, every distribution will approach a Gaussian distribution after a certain number of convolutions, and this chi-squared distribution is no exception. It will take many intervals in order to approach this, because even after 160 distributions are averaged, the logarithmic plot still does not look parabolic.

Problem 3

A.

Width of background Gaussian distribution, X = 2.5

Signal strength of an X-ray or a UV ray in a CCD, Y = 10.2

The signifcance of a detection of an X-ray or ultraviolet signal against a Gaussian background distribution (equation given in problem 1) is determined by how many σ's the measurement is. For a Gaussian distribution, the σ value just corresponds to the number of standard deviations the measurement is away from the mean. It must be 5 or greater in order to be considered significant.

The σ value for measuring a signal of 7.5 is given by:

$\sigma = 2erf^{-1}(\int_{10.2}^{\infty} P(x)dx)$, where $P(x)$ is the probability distribution of a Gaussian.

In [9]:

```python
#calculating the sigma value
prob = stats.norm.sf(10.2, scale=2.5)
print('The probability of having a measurement from the background that is 10
sig = stats.norm.isf(prob)
print('The sigma value of a measurement of 10.2 is {}'.format(sig))
```

```
The probability of having a measurement from the background that is 10.2 or
higher is 2.2517850388525404e-05
The sigma value of a measurement of 10.2 is 4.08
```

Since 4.08σ < 5σ, the measurement was not significant enough to claim a discovery.

B.

What is the probability that the background produces a measurement of 7.5 if it is described by a Gaussian distribution with a mean of 0 and a width of 2.5 per pixel over 10,000 pixels?

In order to obtain the new background distribution, the Gaussian distribution for a pixel must be multiplied by the number of pixels, which is 10,000.

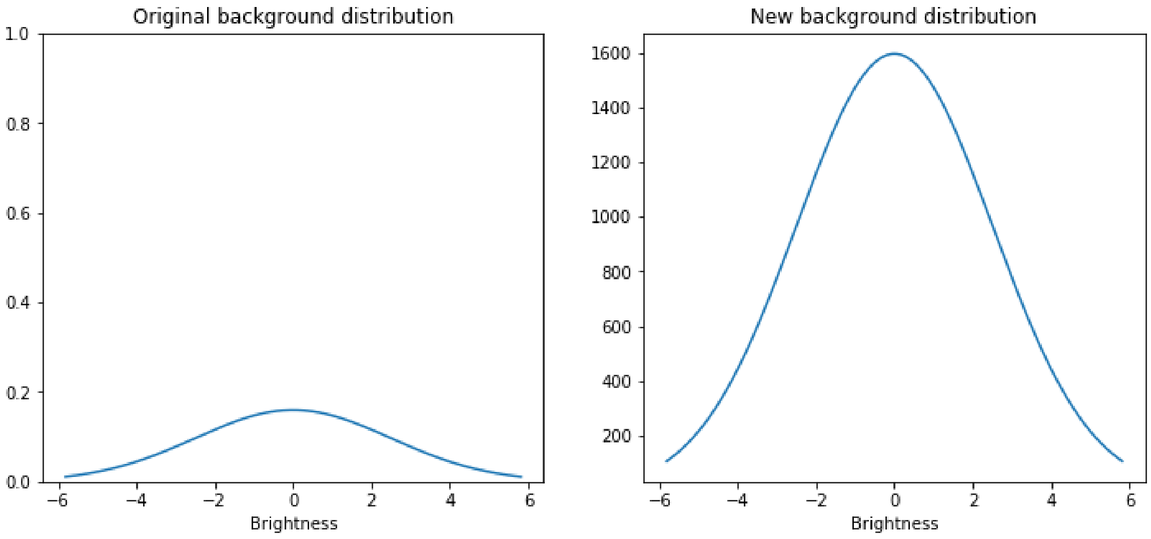
$P_{new}=10000\cdot P_{old}$

In [10]:

```
#plotting the old backround distribution against the new background distribut
x = np.linspace(stats.norm.ppf(0.01, scale=2.5), stats.norm.ppf(0.99, scale=2
old_norm_pdf = stats.norm.pdf(x, scale=2.5)
new_norm_pdf = old_norm_pdf*10000
fig, axes = plt.subplots(1, 2, figsize=(12, 5))
axes[0].set_title('Original background distribution')
axes[0].set_xlabel('Brightness')
axes[0].plot(x, old_norm_pdf)
axes[0].set_ylim(0, 1)
axes[1].set_title('New background distribution')
axes[1].set_xlabel('Brightness')
axes[1].plot(x, new_norm_pdf)
```

Out[10]:

```
[<matplotlib.lines.Line2D at 0x1bed5a369e8>]
```



C.

$\sigma = 2erf^{-1}(\int_{10.2}^{\infty}10000\cdot P(x)dx)$, where $P(x)$ is the original probability distribution in A..

In [11]:

```
prob2 = stats.norm.sf(10.2, scale=2.5)*10000
print('The probability of a measurement of 10.2 with the new background is {}
signal1 = stats.norm.isf(prob2)
print('The sigma value of a measurement of 10.2 with the new backgound is {}'
```

```
The probability of a measurement of 10.2 with the new background is 0.22517
850388525404
The sigma value of a measurement of 10.2 with the new backgound is 0.754819
9735694366
```

Since 0.75σ < 5σ, there is not enough significance to claim discovery. The new σ from the background distribution encompassing all 10,000 pixels indicates that a signal needs greater brightness in order to have significance.

Problem 4

A.

Using the relation in 3.A. with $\sigma = 5$:

$5 = 2erf^{-1}(\int_{x}^{\infty} P(x)dx)$

$erf(\frac{5}{2}) = \int_{x}^{\infty} P(x)dx$

In [12]:

```
#calculating the probability and signal value of getting 5 sigma
sigma5prob = stats.norm.sf(5)
signal2 = stats.norm.isf(sigma5prob, scale=2.5)
print('The signal needed to get 5 sigma probability with the original backgro
```

```
The signal needed to get 5 sigma probability with the original background d
istribution is 12.500000000000002
```

B.

Using the relation in 3.B.:

$5 = 2erf^{-1}(\int_{x}^{\infty} P(x)\cdot 10000dx)$

$\frac{erf(\frac{5}{2})}{10000} = \int_{x}^{\infty} P(x)dx$

In [13]:

```
#calculating the probaility and signal value of getting 5 sigma with the new
sig4 = stats.norm.isf(sigma5prob/10000, scale=2.5)
print('The signal needed to get 5 sigma probbility with the new background di
```

```
The signal needed to get 5 sigma probbility with the new background distrib
ution is 16.376344031763335
```

C.

The signal from the new distribution needs to be about 1.3 times brighter than that of the original distribution in order for it to warrant a discovery. This happens because the probability drops off more at values farther from the mean. 10000 still is not enough to greatly affect the probability of brightnesses above the values around the 5 sigma significance level.

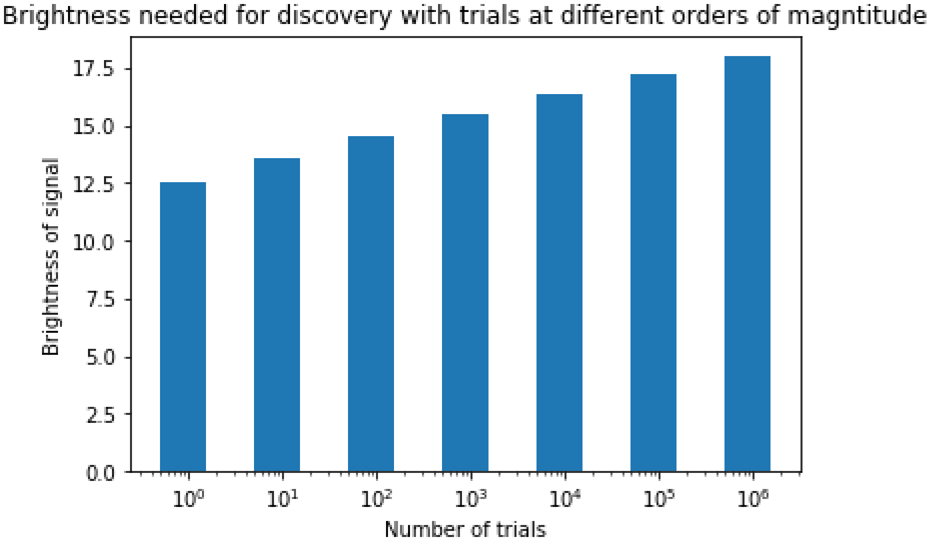
D.

Calculating the needed brightness in order to get a 5 σ value for different numbers of trials varying by order of magnitude:

In [14]:

```
#finding the signal for each number of trials
trials = [1, 10, 100, 1000, 10000, 100000, 1000000]
signals = []
for each in trials:
    signals.append(stats.norm.isf(sigma5prob/each, scale=2.5))

#plotting the 5 sigma signal against number of trials
plt.figure()
plt.title('Brightness needed for discovery with trials at different orders of
plt.bar(trials, signals, width=trials)
plt.xlabel('Number of trials')
plt.xscale('log')
plt.ylabel('Brightness of signal')
plt.show()
```



The effect on the brightness needed to produce a probability of 5σ is very modest compared to the increase in number of trials. Increasing the number of trials by a factor of 10^6 leads to an increase in brightness of only 1.4. The number of trials does not greatly affect the sensitivity of the significance threshold.

In [ ]: