Lab 1 Physics 434

Ondrea Robinson

October 2019

1 A Little Statistics

1.1 Converting a Probability into a Sigma

1.1.1

The normal or Gaussian distribution is the standard distribution for most probabilistic data. It is a bell-shaped curve that centers around an average value with the highest incidence, with values far from the average decreasing in incidence.

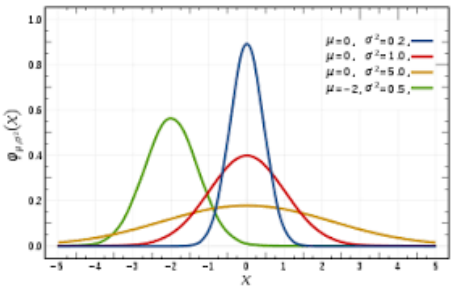


Figure 1: Graphs of three different normal PDFs

1.1.2

When we integrate the standard normal distribution, we have to integrate between two chosen points to find the probability that a measurement will be chosen between these two points. This is called the Cumulative Distribution Function, or CDF The CDF function is defined so that

$$cdf(b) = \int_{-\infty}^{b} pdf(x)dx$$

. Where pdf is the Probability Density Function defining the distribution we're looking at. In python, we can use the stats.norm.cdf() function:

```
import scipy
from scipy import stats
z = 1
table = stats.norm.cdf(z)
print(table)
```

From this code for values $z = [1, 3.5, 4]$ we get $table = [0.84, 0.99, 0.99]$. This matches the Z-table found on Wikipedia.

1.1.3

In this example we calculate the inverse function of the CDF called the Point Percentage Function. Instead of entering a sigma value we enter a probability and are returned the associated sigma. To show this, can enter the value given by the cdf function back into the ppf function and receive the original values back. To do this in python we use the code:

```
prob = stats.norm.ppf(table)
```

So that for $table = [0.84, 0.99, 0.99]$ it returns $sigma = [1, 2.32, 2.35]$.

1.1.4

The ppf function returns a sigma value that is centered around a mean of zero. So if the sigma value returned is negative it just means the sigma associated with that probability is on the left of the mean.

1.2 The Lognormal Distribution

1.2.1

In the lognormal distribution, the data variable x is distributed such that if you take $log(x) = y$, y is normally distributed. This distribution is fairly common and used to model the time it takes to solve a Rubik's cube, the size of living tissues and city sizes. The lognormal PDF is defined so that

$$pdf(x) = \frac{1}{x\sigma\sqrt{2\pi}} exp(-\frac{(\ln(x-\mu))^2}{2\sigma^2})$$

.

In python, we can graph the lognormal distribution with different parameters. The parameter s in this case defines sigma for the underlying normal distribution. The python code for the plots is:

```
s1 = 0.340
s2 = 0.626
s3 = 0.954

r1 = stats.lognorm.rvs(s1, size = 100000)
```

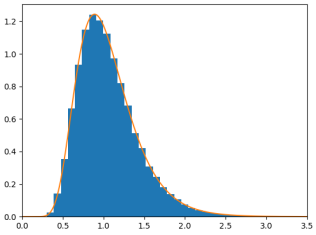
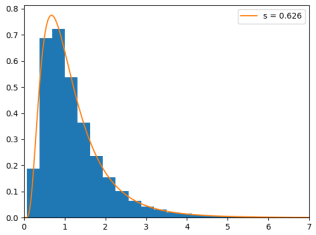
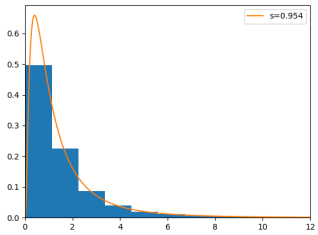


Figure 2: sigma = 0.340






```
r2 = stats.lognorm.rvs(s2, size = 100000)
r3 = stats.lognorm.rvs(s3, size = 100000)

figure(1)
plt.xlim([0, 3.5])
plt.hist(r1, 50, density=True)
x1 = np.linspace(0,3.5,100000)
plt.plot(x1,stats.lognorm.pdf(x1,s1), label='s = 0.340')

figure(2)
plt.hist(r2, 50, density=True)
plt.xlim([0, 7])
x2 = np.linspace(0,7,100000)
plt.plot(x2,stats.lognorm.pdf(x2,s2), label='s = 0.626')
plt.legend()

figure(3)
plt.xlim([0, 12])
plt.hist(r3, 50, density=True)
x3 = np.linspace(0,12,100000)
plt.plot(x3,stats.lognorm.pdf(x3,s3), label='s=0.954')
plt.legend()
```

This gives us the three figures above for three different sigma values. We can also plot the three PDFs against each other to get:

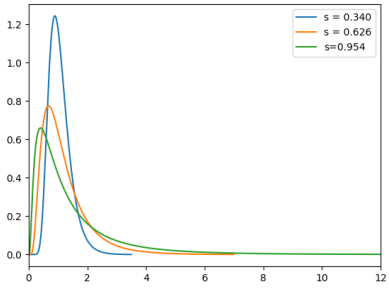


Figure 3: All three lognormal PDFs with different sigma

1.3 Finding Sigma

I have selected a value relevant to my first graph (Figure 1) of $x = 1.5$. From the histogram this x value has a relatively high probability. The question we want to ask is "if we have a signal-free, lognormal background distribution, what is the probability of a chosen measurement (or less) being from our background and what value of the standard deviation (sigma) does this probability correspond to in a normal distribution?" In this case the python code we want to use is

```
measurement = 1.5
prob = stats.lognorm.cdf(measurement,s1)
sigma = stats.norm.ppf(prob)
print(prob)
```

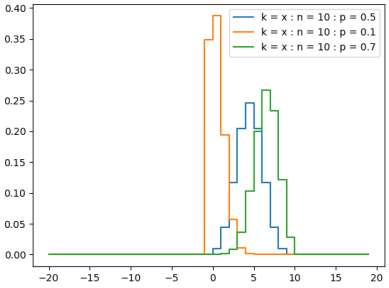
2 The Binomial Distribution

The two previous distributions we've looked at have both been continuous. That is, both the lognormal and the normal distributions have had continuous PDFs that take continuous parameters. The binomial distribution is a discretized distribution that is used to calculate the probability of getting a certain result in an experiment with only two outcomes; flipping a coin, for example. It takes a probability p that determines the rate of getting a boolean outcome like true in true/false or yes in yes/no. By convention p determines the rate of "success" (yes and true in the examples above). This distribution also considers the number of independent experiments, n.

The probability of getting k successes in n trials is given by the probability mass function:

$$f(k, n, p) = \frac{n!}{k!(n-k)!} p^k (1-p)^{n-k}$$

.



Using different parameters n, k and p we can get three different plots of binomial distributions using the code below:

```
x = np.arange(-20,20,1)
plt.step(x,stats.binom.pmf(x,10,0.5),label = 'k = x : n = 10 : p = 0.5')
plt.step(x,stats.binom.pmf(x,10,0.1),label = 'k = x : n = 10 : p = 0.1')
plt.step(x,stats.binom.pmf(x,10,0.7),label = 'k = x : n = 10 : p = 0.7')
plt.legend()

plt.show()
```

If we wanted to ask the same question of the binomial distribution as we asked for the lognormal, we might ask "What is the probability that my background binomial distribution gives me a k value of 10 for $n = 100$ trials, with $p = .25$, and what does this probability correspond to in sigma if my background was normally distributed?"

We can calculate this in python using the code:

```
k = 10
n = 100
p = 0.25
probBinom = stats.binom.cdf(k,n,p)
sigBinom =stats.norm.ppf(probBinom)
print(sigBinom)
print(probBinom)
```

out: 0.00013710 This probability corresponds to -3.63σ in the normal (continuous) distribution, but since our function is discrete this value corresponds to 3σ.