Distributions

1. Introduction

This is the R track of the *Distributions* workshop prepared by Los Angeles County, ISAB. Section numbering is intended to be consistently referenced from the main workshop document available at

https://github.com/lacounty-isab/workshops/tree/master/distributions

1.1. Preparation

Since R is oriented toward statistics, no special preparation is required to work with distributions. You can obtain a list of *baked-in* distributions by entering

```
?Distributions
```

These are all part of the **stats** package which is available in every R installation.

1.2 Conventions

R has a consistent naming convention for functions that work with distributions - a single letter followed by the name of the distribution. The four single letters are

- `d` - density function
- `p` - percent point function (CDF)
- `q` - inverse of CDF
- `r` - random sampler

If we take the binomial distribution as an example, then `dbinom` is a *binomial density function.*

```
dbinom(4, 10, 0.3)
```

```
## [1] 0.2001209
```

This gives the probability of getting 4 successes after 10 attempts where each success has a probability of 0.3; which is about 20 percent.

2 Discrete Distributions

2.1 Binomial Distribution

The binomial distribution has a `binom` suffix for each of its standard methods. We'll assume 100 trials with 20% chance of success at each trial. The value of the random variable is the number of successes. Let's start with the density function `dbinom`. This is equation (7) of the math supplement.

$$f_X(i) = \sum_{i=0}^{n} \binom{n}{i} p^i (1-p)^{n-i}$$

Here we consider the case of $n = 100$ and $p = 0.2$. We suspect that the highest probability will be around 20 successes since the probability of each success is 0.2 and there are 100 trials. The `dbinom` function accepts numeric vectors, so we'll check `10`, `20`, and `30` in one go.

```r
dbinom(c(10, 20, 30), 100, 0.2)
```

```
## [1] 0.003362820 0.099300215 0.005189643
```

Sure enough, the density functon shows a 9.9% chance of obtaining exactly 20 successes. The probability for 10 and 30 are each less than 1%.

Now let's check the associated cumulative distribution function, or CDF. The CDF at a point is the cumulation of probability from the density function for all points equal or less. Since we expect the average to be around 20, the CDF should be close to 0.5 at 20.

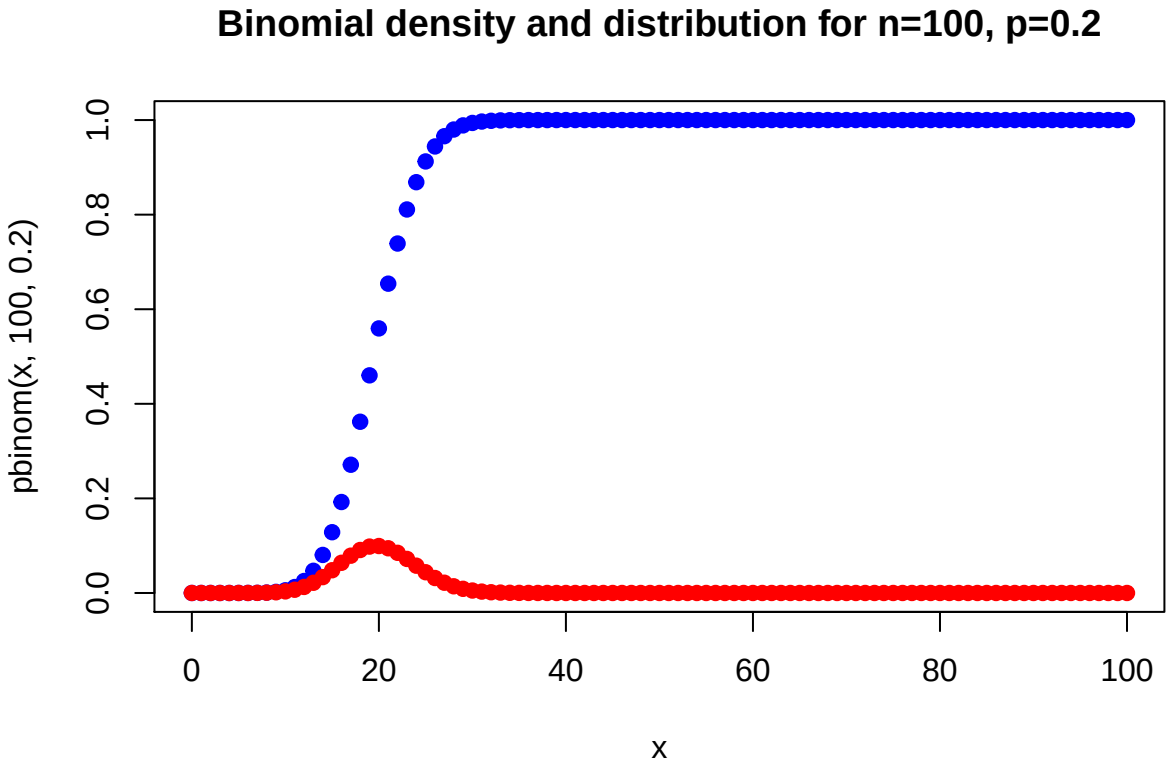
```r
pbinom(c(10, 20, 30), 100, 0.2)
```

```
## [1] 0.005696381 0.559461585 0.993940665
```

This shows that the probability of having *less than or equal to 10 successes* is 0.57%. The probability of having less than or equal to 30 successes is 99.4%.

To better see the relationship between the density and cumulative distributions, we can plot them.

```r
x <- seq(0, 100)
plot(x, pbinom(x, 100, 0.2), col="blue", pch=19)
points(x, dbinom(x, 100, 0.2), col="red", pch=19)
title(main="Binomial density and distribution for n=100, p=0.2")
```



Quantiles are the inverse of the CDF function. For any probability, a quantile function tells us the value of the random variable for which the CDF is equal to that probability.

How many successes can we expect to have 25%, 50%, and 75% of the time?

```r
qbinom(c(0.25, 0.5, 0.75), 100, 0.2)
```

```
## [1] 17 20 23
```

The reason these come out to exact integers is because the binomial random variable only takes integer values. The `qbinom` can be interpreted as *the smallest integer number of successes with probability greater than or equal to the argument*. So it's not a strict inverse.

```
pbinom(c(16, 17, 19, 20, 22, 23), 100, 0.2)
```

```
## [1] 0.1923376 0.2711890 0.4601614 0.5594616 0.7389328 0.8109128
```

Sometimes it's useful to generate samples from a distribution. In other words, the numbers are generated as if they came from random variable values associated with outcomes of the experiement. The `rbinom` function is used for this.

```
b_sample <- rbinom(1000, 100, 0.2)
b_sample[1:100]
```

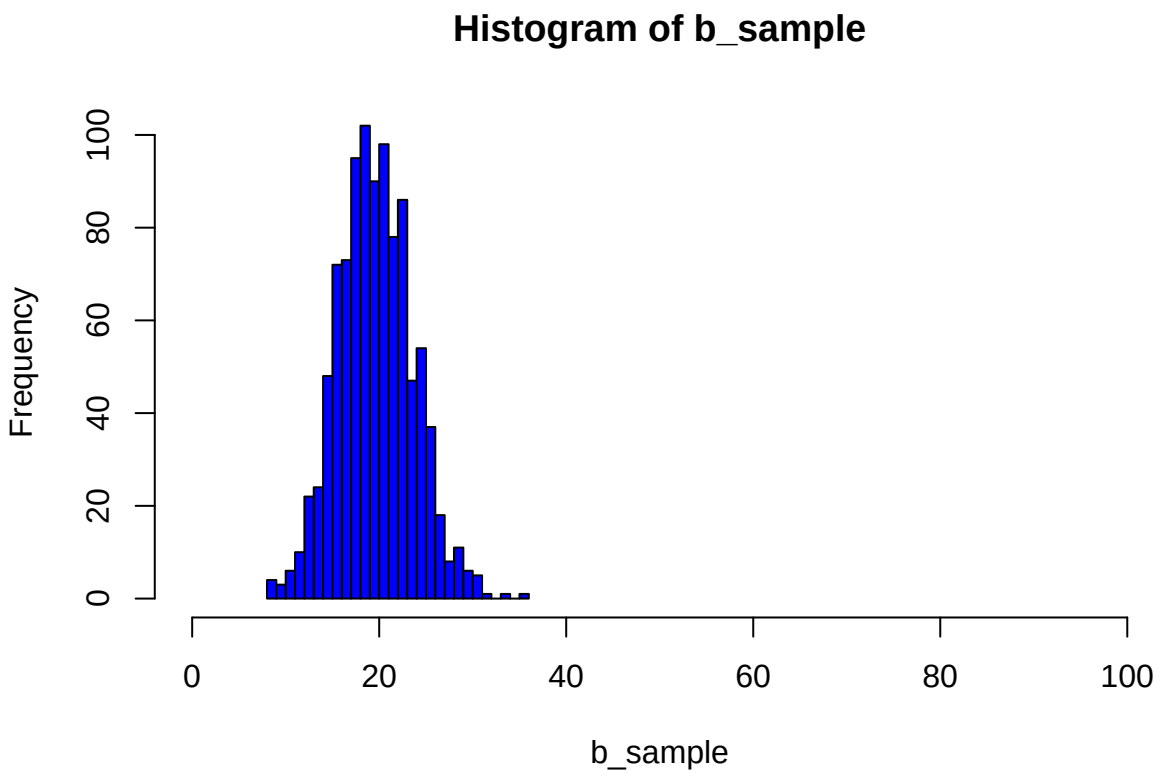
```
##   [1] 16 18 17 21 21 25 16 13 15 14 23 23 18 26 26 22 19 14 16 20 12 25 23
##  [24] 20 15 21 19 23 15 19 19 24 23 23 21 16 20 25 16 13 16 25 16 20 21 21
##  [47] 13 27 15 23 19 18 24 19 16 18 18 16 26 21 21 22 24 18 22 17 16 19 16
##  [70] 24 17 19 17 25 21 23 20 16 17 22 15 23 14 24 25 20 12 16 28 25 15 24
##  [93] 19 17 17 18 19 16 22 26
```

This simulates the arduous task of

1. flipping an unfair coin (with 20% chance of heads) 100 times,
2. noting the number of heads,
3. repeating steps 1 and 2 for 1,000 times

It's fun to plot a histogram of our sample. It should have a shape "approaching" our density function.

```
hist(b_sample, breaks=30, xlim=c(0, 100), col="blue")
```



2.2 Poisson Distribution

The Poisson random variable represents the number of occurrences of an event over a fixed interval of time. This could be the number of crimes reported per hour. The parameter of the random variable, usually denoted by λ, turns out to be the expected number of occurrences. The Poisson density function is given by

$$f_X(i;\lambda) = e^{-\lambda}\frac{\lambda^i}{i!}$$

For each i, this represents the probability of observing i counts within the unit time period. R provides the usual four random variable functions for the Poisson random variable.

- `dpois` - Poisson density (PMF) function
- `ppois` - Poisson probability (CDF) function
- `qpois` - Poisson quantile (inverse CDF) function
- `rpois` - Poisson random sample

Let's consider a case of a Poisson process where the expected number of occurrences per time period is 5.

Since 5 is the expected value, we expect the mass function to be strong near 5. Let's check this.

```
dpois(c(3,5,7), 5) * 100
```

```
## [1] 14.03739 17.54674 10.44449
```

We see a 14% of getting a 3 count, a 18% chance of getting a 5 count and a 10% chance of getting a 7 count. Now let's check the cumulative distribution function.

```
ppois(c(3,5,7), 5) * 100
```

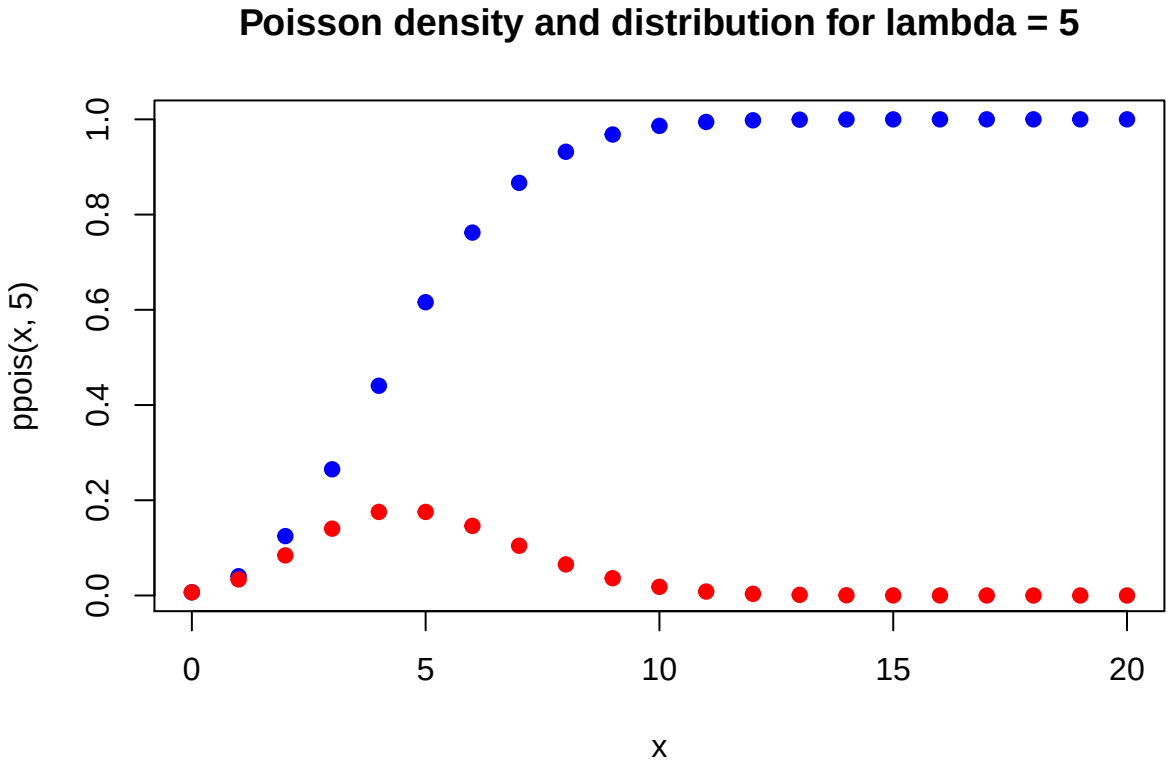
```
## [1] 26.50259 61.59607 86.66283
```

The chance of getting

- 3 or less is 27%
- 5 or less is 62%
- 7 or less is 87%

Let's view this graphically.

```
x <- seq(0, 20)
plot(x, ppois(x, 5), col="blue", pch=19)
points(x, dpois(x, 5), col="red", pch=19)
title(main="Poisson density and distribution for lambda = 5")
```



The standard quantiles are giving by

```
qpois(c(0.25, 0.5, 0.75), 5)
```

```
## [1] 3 5 6
```

- 3 is the smallest count that occurs at least 25% of the time
- 5 is the smallest count that occurs at least 50% of the time
- 6 is the smallest count that occurs at least 75% of the time

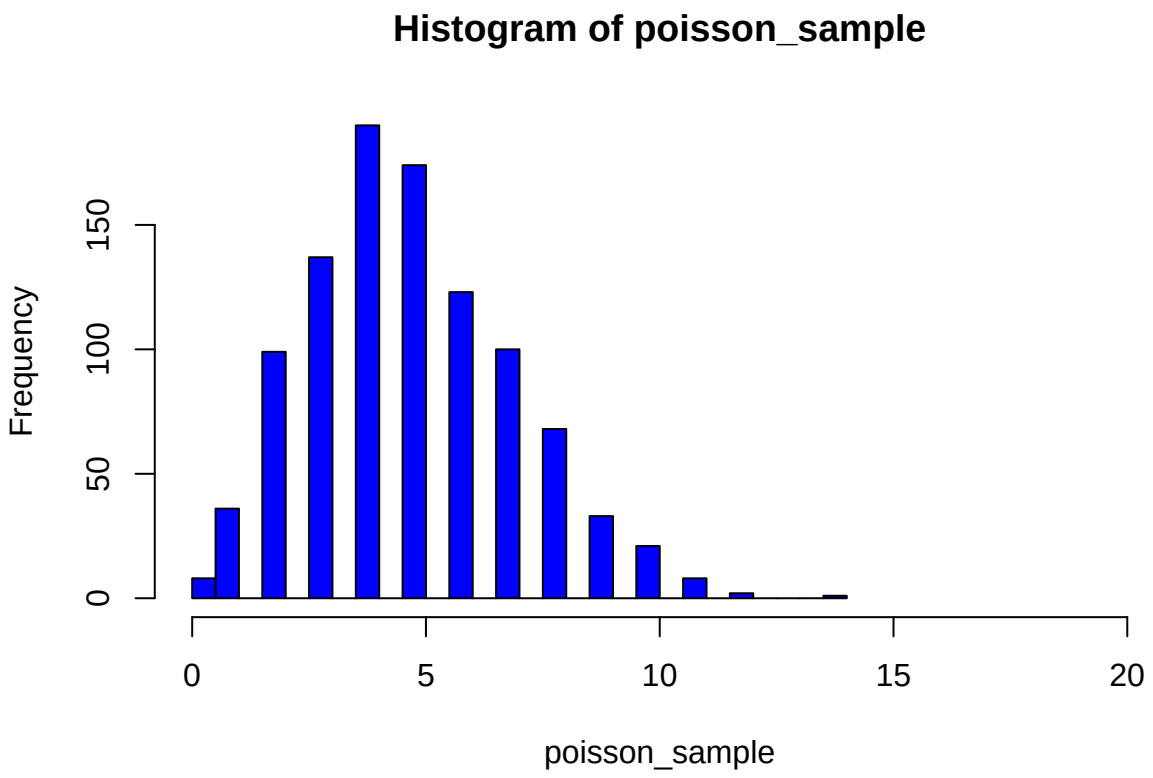
Finally, let's run a Poisson simulation with 1,000 trials with $\lambda = 5$.

```
poisson_sample <- rpois(1000, 5)
poisson_sample[1:100]
```

```
##   [1]  5  3  7  4  6  3  3  7  4  7  8  3  2  4  7  5  5  4  6  4  5  6  5
##  [24]  5  6  7  7  4  3  5  8  3  4  3  3  1  3  4  8  4  6  1  1  9  3  4
##  [47]  6  8  5  2  5  7  5  4  3  1  9  7  9  4  3  3  7  2  1  5  7  3  3
##  [70] 10  4  2  2  5  4  6  2  5  5  2  5  4  5  5  0  6  1  5  7  4  4  3
##  [93]  4  2  9  3  2  2  7  3
```

Let's check that our histogram looks anything like our PMF.

```
hist(poisson_sample, breaks=20, xlim=c(0, 20), col="blue")
```



3 Continuous Distributions

3.1 Normal Distribution

The following functions address the normal distribution.

- `dnorm` - density function
- `pnorm` - cumulative distribution function
- `qnorm` - quantiles
- `rnorm` - random samples

```
dnorm(c(-3, -2, -1, 0, 1, 2, 3))
```

```
## [1] 0.004431848 0.053990967 0.241970725 0.398942280 0.241970725 0.053990967
## [7] 0.004431848
```

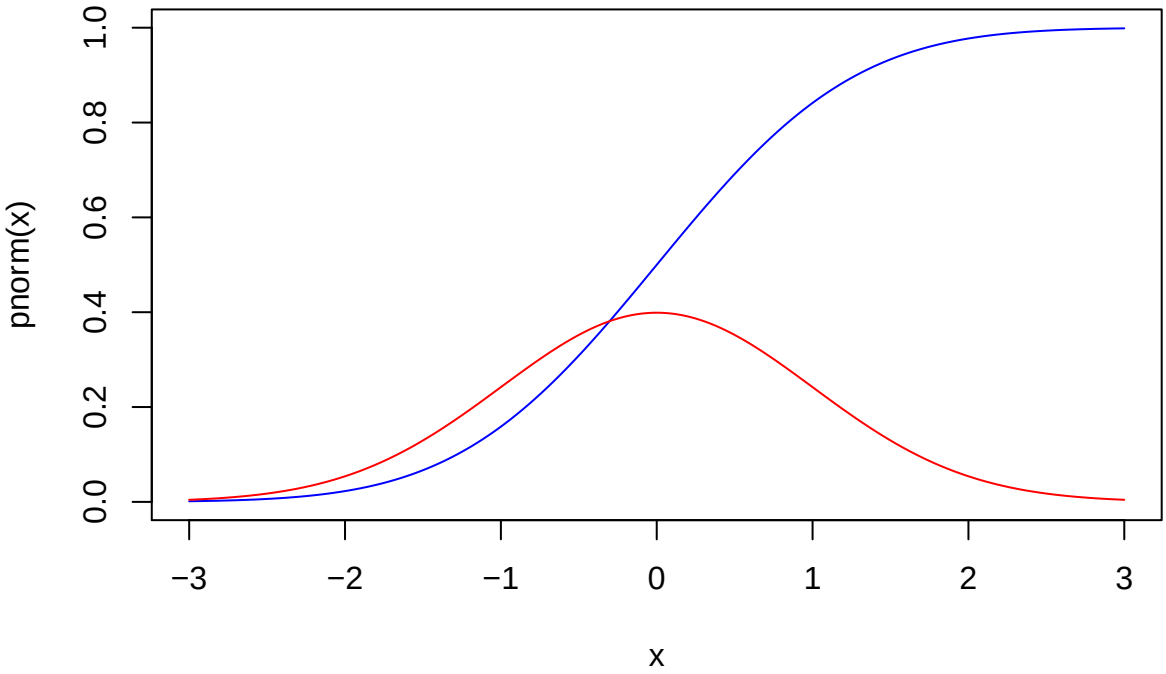
The symmetric character of the normal distribution is apparent here. We don't interpret these as probabilities (like we did with discrete mass function) since the continuous case represents an infinitesimal range of the random variable. But we still extract probabilities from the CDF.

```
pnorm(c(-3, -2, -1, 0, 1, 2, 3)) * 100
```

```
## [1]  0.1349898  2.2750132 15.8655254 50.0000000 84.1344746 97.7249868
## [7] 99.8650102
```

Graphically the normal is the familiar bell curve.

```
x <- seq(-3, 3, by=0.02)
plot(x, pnorm(x), col='blue', type='l', lty='solid')
lines(x, dnorm(x), col='red')
title(main="Standard Normal PDF and CDF", ylab="", xlab="")
```



The `qnorm` function provides quantile information. In other distributions we've shown quantile information for quarters, i.e. quartile information. But due to the role of the normal distribution in the Central Limit Theorem, we'll examine quantiles for 90%, 95%, and 97.5% since these are often referenced in the study of confidence intervals.

```
qnorm(c(0.5, 0.9, 0.95, 0.975))
```

```
## [1] 0.000000 1.281552 1.644854 1.959964
```

Since the normal distribution is symmetric about the origin, we expect half to lie on one side of 0 and half on the other side. The last number says that 97.5% of the curve lies to the left of **1.96**. This is a number we'll commit to memory during the study of confidence intervals.

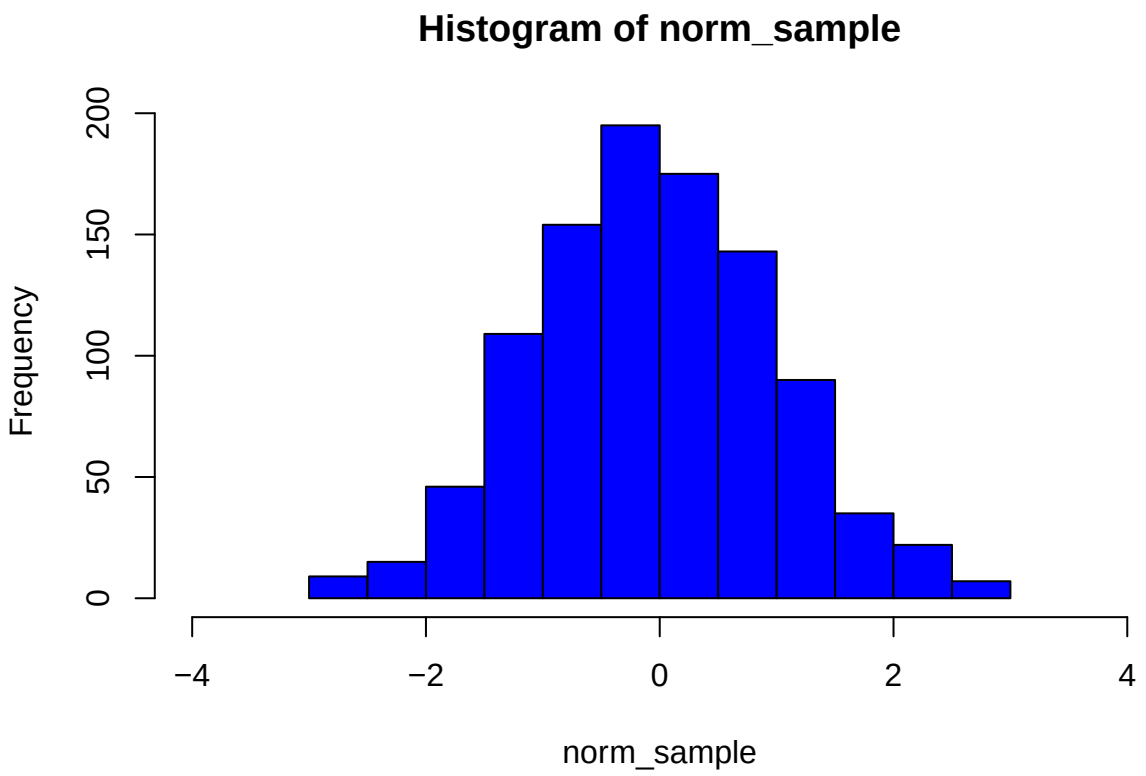
Finally, let's run a normal simulation with 1,000 trials using $\mu = 0$ and $\sigma = 1$.

```
norm_sample <- rnorm(1000)
round(norm_sample[1:100], 2)
```

```
##   [1] -1.15  0.39 -0.97  0.64 -1.07  1.87  1.94 -1.30 -0.93 -0.11  1.02
##  [12]  1.00 -2.06 -0.73 -1.03 -0.16  0.85  1.28  2.17  0.90  0.74 -1.24
##  [23]  1.12 -1.24 -0.56 -0.72 -1.25  0.97  0.26  0.28  0.10 -0.62 -0.48
##  [34] -0.19  0.37  0.93  1.06  0.84 -0.16 -1.14 -1.71  0.68  0.58 -1.05
##  [45] -0.43 -1.31  0.94 -0.70  0.12 -0.37 -1.34  1.08  0.98  1.01 -0.33
##  [56] -1.98  0.05 -1.20 -0.64  0.92  0.93 -0.32  0.09  1.43  1.86  0.08
##  [67]  0.82 -0.30  0.42  1.96  0.63  1.73  1.31  0.32  0.81 -0.92 -1.77
##  [78] -1.60 -0.27 -0.71  1.16  0.63  1.20  0.45 -1.28  0.33 -0.22  1.14
##  [89]  1.12 -0.17 -1.44 -0.33 -0.75 -0.15 -0.37  0.05 -0.07  1.23  0.01
## [100]  1.28
```

Let's check that our histogram looks anything like our PMF.

```
hist(norm_sample, breaks=20, xlim=c(-4, 4), col="blue")
```



3.2 Exponential Distribution

The *exponential distribution* is a continuous distribution that complements the discrete Poisson distribution that we encountered above. Whereas a Poisson random variable represents a count of events that occur over a fixed unit of time, an exponential random variable represents the amount of time between one such event and the next.

The density function for the exponential function is

$$f_X(x;\lambda) = \lambda e^{-\lambda x}$$

where the expected wait time is $1/\lambda$ Let's go through the usual moments with $\lambda = 1/3$. The exponential function decreases as x gets larger.

```
x <- dexp(c(1,2,3,4), 1/3)
round(x, 2)
```

```
## [1] 0.24 0.17 0.12 0.09
```

Since the expected wait time is $1/\lambda = 3$ units of time, we expect the CDF to cross 50% there.

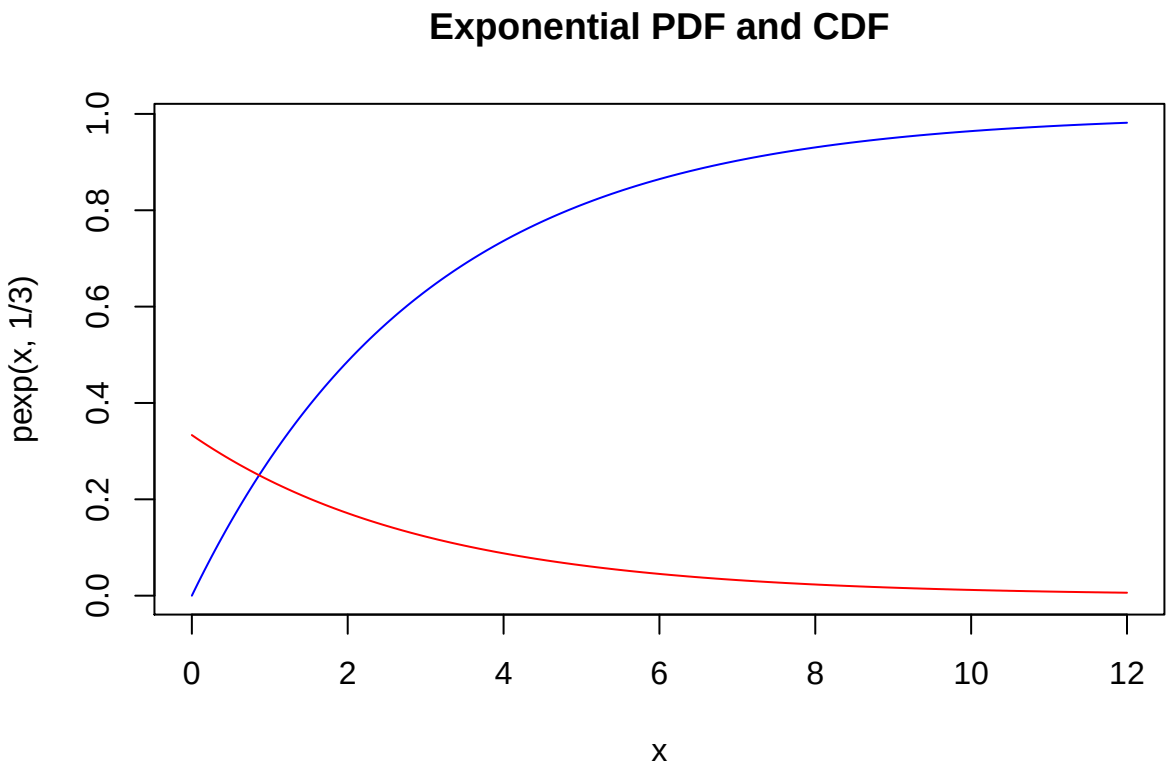
```
x <- pexp(c(1,2,3,4), 1/3)
round(x, 2)
```

```
## [1] 0.28 0.49 0.63 0.74
```

The is the PDF and CDF graphically.

```
x <- seq(0, 12, by=0.02)
plot(x, pexp(x, 1/3), col='blue', type='l', lty='solid')
```

```
lines(x, dexp(x, 1/3), col='red')
title(main="Exponential PDF and CDF", ylab="", xlab="")
```



Once again, we can see the relationship between the CDF $F_X(x;\lambda)$ and the PDF $f_X(x;\lambda)$.

$$F_X(x;\lambda) = \int_0^x f_X(t;\lambda)dt$$

The standard quartiles are given by

```
x = qexp(c(0.25, 0.5, 0.75), 1/3)
round(x, 2)
```

```
## [1] 0.86 2.08 4.16
```

Notice that the median is 2.08. This is an example the median and the average do not agree, even for a continuous random variable. These values tell us we should expect a wait time of

- 0.86 time units or less 25% of the time
- 2.08 time units or less 50% of the time
- 4.16 time units or less 75% of the time

Let's generate some exponential distributed time intervals for a simulated Poisson process.

Finally, let's run an exponential simulation with 1,000 trials with $\lambda = 1/3$.

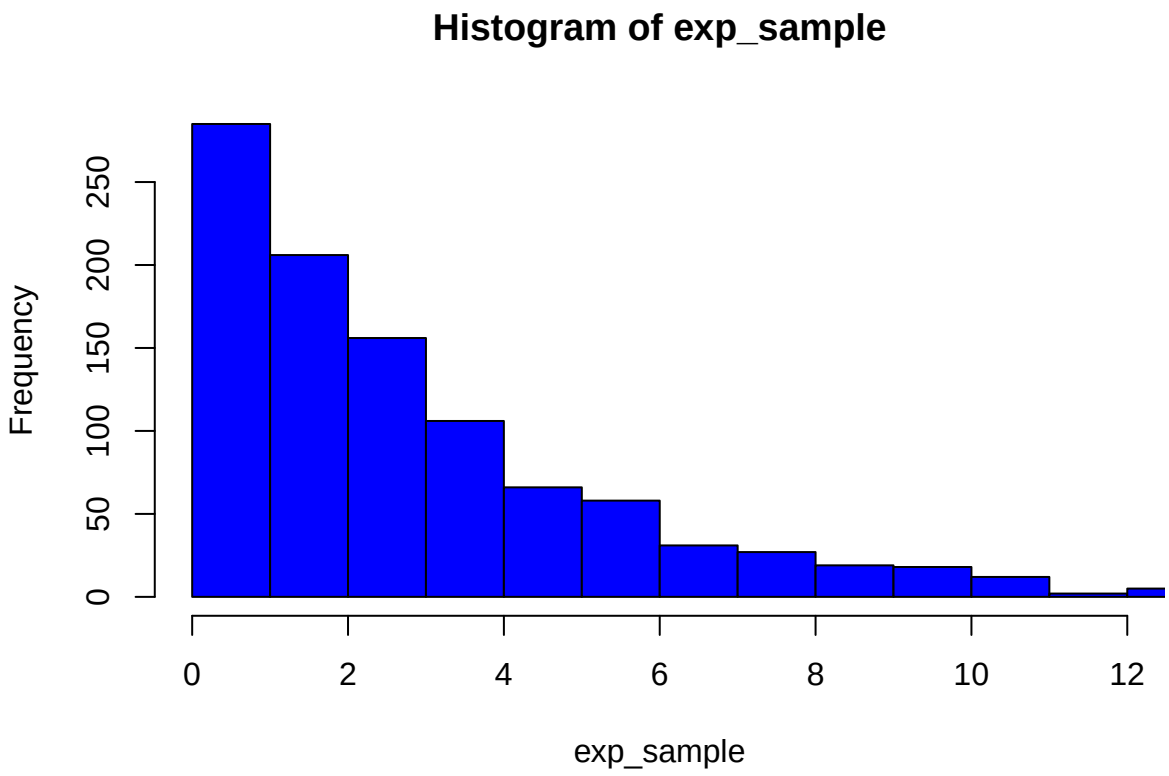
```
exp_sample <- rexp(1000, 1/3)
round(exp_sample[1:100], 2)
```

```
##   [1]  2.95  1.83  4.17  0.85  1.20  0.70  0.33  3.05  4.46  4.04  0.16
##  [12]  3.12  0.69  0.07  1.71  1.30 29.62  1.14  0.07  1.47  3.39  1.77
##  [23]  0.13  2.77  2.02  0.35  3.14  0.01  2.50  1.40  0.96  0.01 15.82
##  [34]  8.94  3.12  0.63  0.11  1.14  0.50  5.60  0.78  1.24  0.35  2.03
```

```
##  [45]  3.68  0.04  2.75  1.18  0.53  2.98  1.08  0.58  7.93  5.89  3.30
##  [56]  4.05  8.15  1.45  1.63  9.32  3.12  2.09  1.65  0.88  0.03  2.93
##  [67]  3.51  7.48  3.67  1.09  5.18  0.80  2.89  1.49  1.50  5.63  1.61
##  [78] 10.99  9.58  1.58  0.60  6.38  3.60  2.78 10.28  1.58  3.97  0.02
##  [89]  0.32  6.32  0.68  3.58  5.83  1.17  2.47  2.43  2.90  2.12  2.98
## [100]  2.96
```

Let's check that our histogram looks anything like our PMF.

```
hist(exp_sample, breaks=24, xlim=c(0, 12), col="blue")
```



Gamma Distribution

A Gamma random variable is often interpreted as a sum of r independent exponential random variables with the same parameter λ. The density function is

$$f_X(x; \lambda, r) = \frac{\lambda}{\Gamma(r)}(\lambda x)^{r-1} e^{-\lambda x}$$

So we have r Poisson processes with an expected wait time of $1/\lambda$. Let's continue using our Poisson process from before with an expected wait time of 3 time units; but this time let's assume we are waiting for 5 of them (one after another). So in this particular case, $\lambda = 1/3$ and $r = 5$.

The R functions related to the Gamma random variable take two parameters:

- `shape` - this corresponds to r in the density function above
- `scale` - this corresponds to λ in the density function above

Let's check a few density function values.

```
x <- dgamma(c(5, 10, 15, 20, 25), shape=5, scale=3)
round(x, 2)
```

```
## [1] 0.02 0.06 0.06 0.03 0.02
```

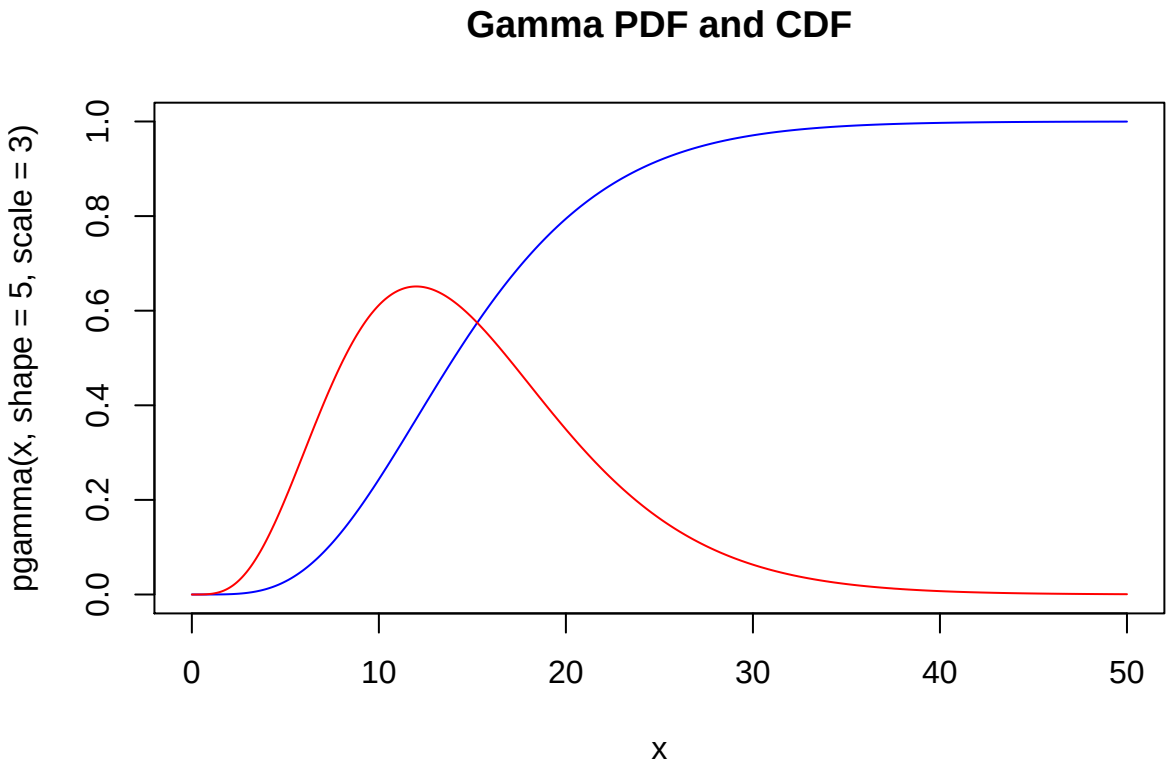
The cumulative distribution function gives a better feel for the distribution.

```
x <- pgamma(c(5, 10, 15, 20, 25), shape=5, scale=3)
round(x, 2)
```

```
## [1] 0.03 0.24 0.56 0.79 0.92
```

The is the PDF and CDF graphically.

```
x <- seq(0, 50, by=0.02)
plot(x,  pgamma(x, shape=5, scale=3), col='blue', type='l', lty='solid')
lines(x, dgamma(x, shape=5, scale=3) * 10, col='red')
title(main="Gamma PDF and CDF", ylab=" ", xlab=" ")
```



The PDF was multiplied by 10 to emphasize its shape. The quartiles are given by

```
x <- qgamma(c(0.25, 0.5, 0.75), shape=5, scale=3)
round(x, 2)
```

```
## [1] 10.11 14.01 18.82
```

This means that we can expect a trial (which consists of 5 independent Poisson processes, each with expected duration of 3 time units) to take less than

- 10.11 time units 25% of the time
- 14.01 time units 50% of the time
- 18.82 time units 75% of the time

Let's simulate 1,000 trials through a random number generator.

```r
g35_sample <- rgamma(1000, shape=5, scale=3)
round(g35_sample[1:100], 2)
```

```
##   [1] 24.09 19.21 19.23  8.91  9.96  5.61 18.67 14.81 23.66 17.73 12.69
##  [12]  6.72 15.44 30.94 15.94 14.28  7.62  4.94 21.26 14.85 14.86  7.73
##  [23] 27.15 10.23 15.58 19.73 14.85  7.29  7.22 12.11 10.36  6.23 14.88
##  [34] 12.62  8.72 11.17  9.85 14.99 17.99 25.89 20.88  5.49 11.00 16.50
##  [45] 14.57 10.90 23.34 16.83 15.17 10.47 19.11 23.64 14.58 12.86  6.79
##  [56] 13.43 10.38 18.12 14.74 14.07 13.30 37.42 17.67 28.20  5.50 19.48
##  [67] 20.70  5.21 28.00  8.34 13.16 12.87 27.44 13.69 22.24 19.98 18.70
##  [78]  9.82 12.26 19.05 25.11 10.12 14.76  9.84 14.43 20.64 11.03 11.63
##  [89] 19.01 14.09 14.20 11.98  5.56 29.31 18.77 10.00 10.57 17.47  7.49
## [100] 30.47
```

Now let's check the histogram.

```r
hist(g35_sample, breaks=25, xlim=c(0, 50), col="blue")
```