Stats with R: part 1

PGDM Research and Analytics cell
Madras School of Economics
Reference - Econometrics with R

Discrete distributions

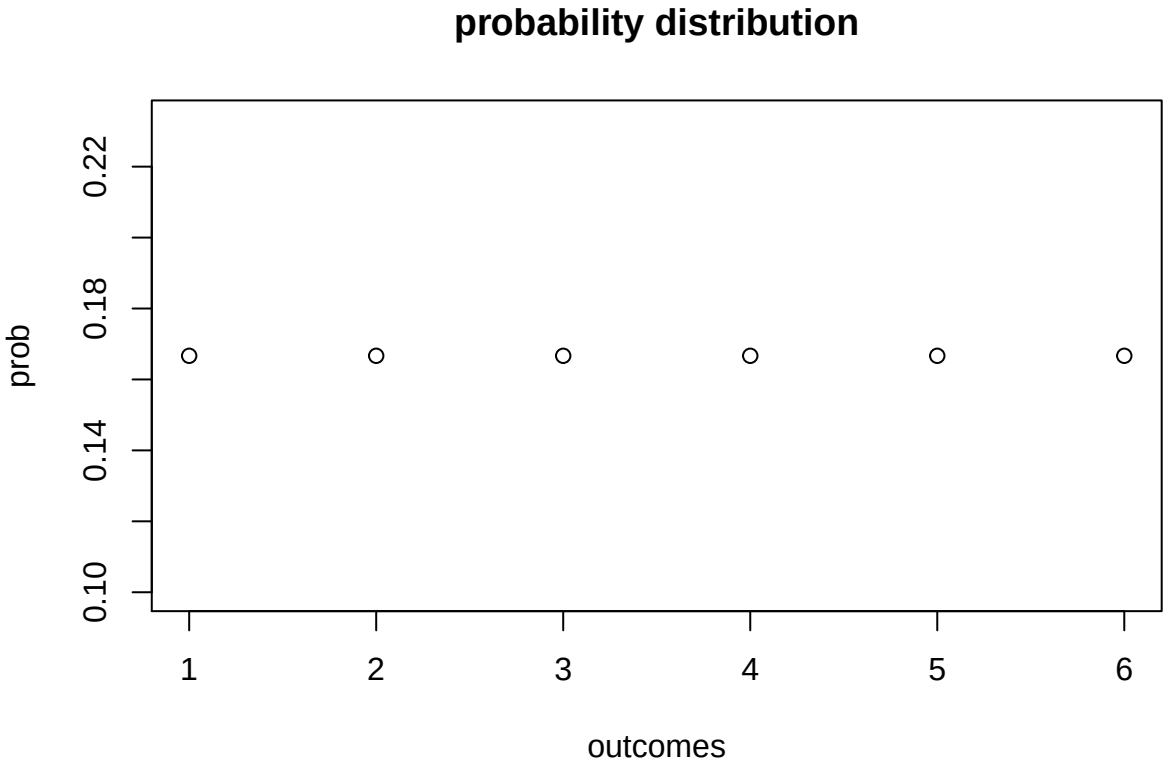
We use the standard **sample** function to draw random samples from a specified set of elements. Below we see the simulation of a die roll. The idea is to randomly select 1 number out of 6 possible outcomes.

```
sample(1:6, 1)
```

```
## [1] 3
```

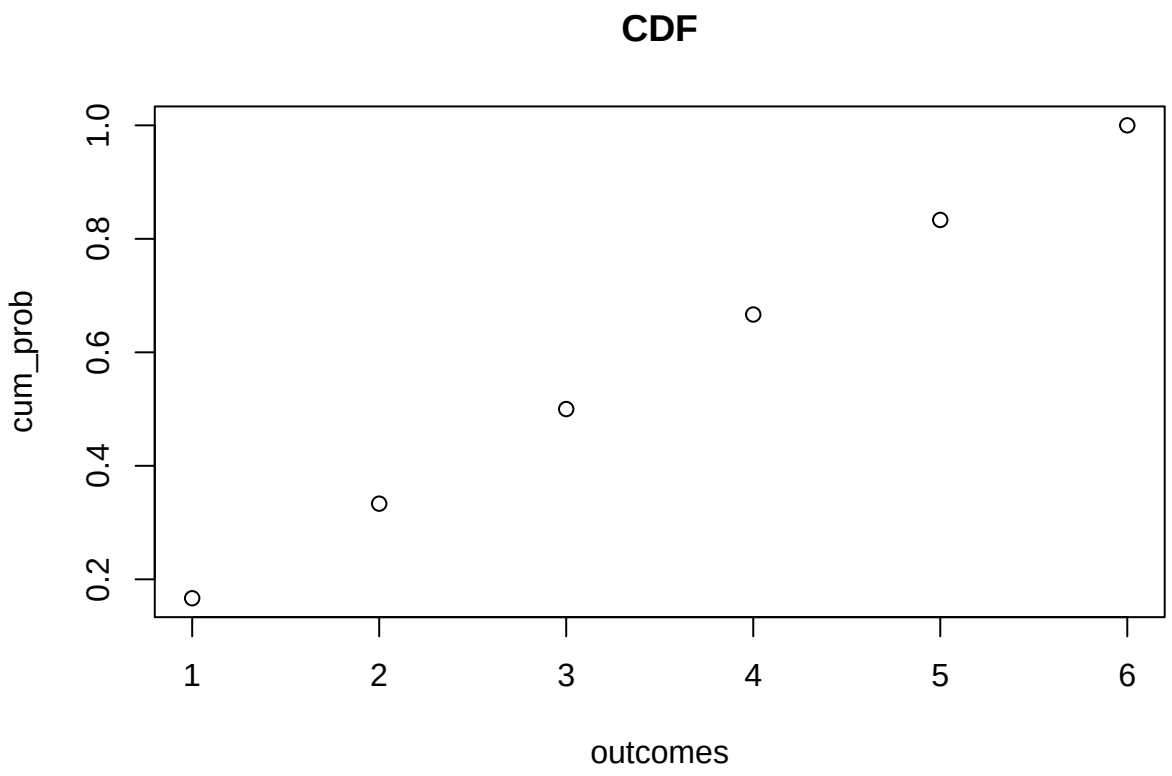
The **probability distribution** of a discrete random variable is defined as the list of all possible values taken by the random variable along with their associated probabilities that sum to 1. The **cumulative probability distribution** function gives the probability that the random variable takes on a value less than or equal to a specified value. In our case, since each outcome has the same probability, we can set up a vector of probabilities by using the **rep** function that repeats a certain value a certain number of times.

```
prob <- rep(1/6, 6)

# plot the distribution

plot(prob,
     xlab = "outcomes",
     main = "probability distribution")
```



To plot the CDF, we need a cumulative sum of the probability vector we defined earlier and this is computed using the **cumsum** function.

```
cum_prob <- cumsum(prob)
plot(cum_prob,
     xlab = "outcomes",
     main = "CDF")
```



Bernoulli trials

Now we learn how to simulate a simple bernoulli random experiment - tossing a coin. We will sample an element two possible outcomes - heads and tails.

```
sample(c("H", "T"), 1)
```

```
## [1] "T"
```

A classic example of a Bernoulli experiment - we toss a coin $n = 10$ times (conduct n bernoulli trials that are independent of each other) and we are then interested to find the likelihood of observing say 5 heads given that the probability of getting a heads is 0.5 in each trial. Number of successes k follows a **binomial distribution**.

$$k \sim B(n, p)$$

$$f(k) = P(k) = \binom{n}{k} \cdot p^k (1-p)^{1-k}$$

Using the **dbinom** function we can compute the probability $P(k = 5|n = 10, p = 0.5)$ as this function takes in arguments that specify - number of successes (k), number of trials (n) and the probability of success (p).

```
dbinom(x = 5,
       size = 10,
       prob = 0.5)
```

```
## [1] 0.2460938
```

Now if we are interested in finding the probability $P(4 \leq X \leq 7)$ that is the probability of getting $4, 5, 6$ or 7 successes in 10 trials then we would input the vector of possible success values in the x argument of **dbinom** and then sum the resulting probabilities.

```
sum(dbinom(x = 4:7, size = 10, prob = 0.5))
```

```
## [1] 0.7734375
```

Alternatively, we can arrive at the same computation by using the principle of the CDF as follows: $P(4 \leq X \leq 7) = P(X \leq 7) - P(X \leq 3)$. The **pbinom** would help us do just that.

```
pbinom(q = 7, size = 10, prob = 0.5) - pbinom(q = 3, size = 10, prob = 0.5)
```

```
## [1] 0.7734375
```

Expected values, mean and variance

We can think of the expected value of a random variable as the long run average value of its outcomes when the number of repeated trials is very large. For a discrete random variable, it is computed as the weighted average of its possible outcomes, weighted by their respective probabilities.

$$E[Y] = \sum_{i=1}^{n} y_i p_i$$

A simple arithmetic average of a numeric vector in R can be done using the **mean** function.

```
mean(1:6)
```

```
## [1] 3.5
```

We will now show how to sample with replacement while simulating an experiment. Note that the **seed** function is used to ensure that the random numbers generated across different machines is the same. We will see that if we sample many times of times, then the sample average turns out to be very close to the actual mean.

```
set.seed(1)
mean(sample(1:6,
            size = 10000,
            replace = T))
```

```
## [1] 3.5138
```

Now we get to the **variance** which measures the dispersion of outcomes of a random variable. Variance of a discrete random variable is given as:

$$\sigma_Y^2 = Var(Y) = E[(Y - \mu_y)^2] = \sum_{i=1}^{n}(y_i - \mu_y)^2 p_i$$

Note that the above function only defines population variance. The sample variance is computed as follows:

$$s_Y^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})$$

The sample variance is computed using the **var** function in R.

```
var(1:6)
```

```
## [1] 3.5
```

Distributions of Continuous variables

Continuous random variables can take on a continuum of possible values within their specified range and hence for these random variables, we characterize the distribution as a **probability density function**. Using an example, here are the various computations and formulas illustrated:

$$f_X(x) = \frac{3}{x^4},\ x > 1$$

The above is the PDF for a random variable X taking on values from 1 till ∞. We first show that the integral over this range sums to 1.

$$\int f_X(x)dx = \int_1^\infty \frac{3}{x^4}dx = 1$$

The expected value is given as:

$$E[X] = \int x \cdot f_X(x)dx = \int_1^\infty x \cdot \frac{3}{x^4}dx = \frac{3}{2}$$

The variance can be expressed as $Var(X) = E[X^2] - (E[X])^2$ and hence below we write the computation for $E[X^2]$.

$$E[X^2] = \int x^2 f_X(x)dx = \int_1^\infty x^2 \cdot \frac{3}{x^4}dx = 3$$

We can do these analytical calculations in R by first defining the appropriate functions and then using the **integrate** function. Note that in order to extract the integral value we will reference the function along with the $ operator which extracts named elements from a list.

```
f <- function(x) 3 / x^4
g <- function(x) x * f(x)
h <- function(x) x^2 * f(x)

area = integrate(f,
                 lower = 1,
                 upper = Inf)$value
area
```

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

```
EX <- integrate(g,
                lower = 1,
                upper = Inf)$value
EX
```

```
## [1] 1.5
```

```
Var <- integrate(h,
                 lower = 1,
                 upper = Inf)$value - EX^2
Var
```

```
## [1] 0.75
```

Now before moving further, we note some common prefixes in R that help us generate various measures from standard distribution functions.

1. **d** stands for density - example - **dnorm** would generate the pdf value according to a specified value from the normal distribution.
2. **p** stands for probability - example - **pnorm** would generate the CDF value according to a specified value from the normal distribution.
3. **q** stands for quantile - example - **qnorm** would generate the quantile (inverse CDF) for a given CDF value from the normal distribution.
4. **r** stands for random number - example - **rnorm** would generate a random number from the normal distribution.

The normal distribution

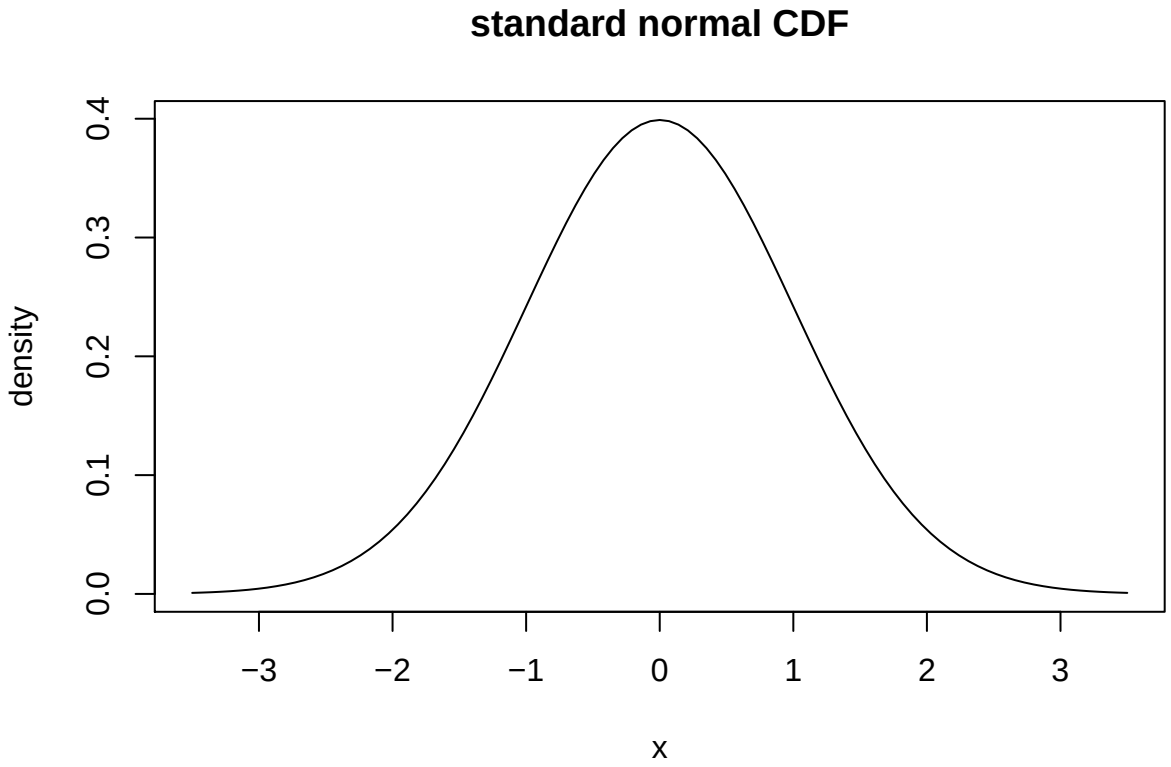
The normal distribution is bell shaped and characterized by its mean and variance. It's PDF is given as:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(x-\mu)^2}{s\sigma^2}\right]$$

The standard normal CDF is usually denoted as Φ (capital Phi) and its derivative gives us the pdf denoted as ϕ (small phi). A typical characterization of this is given by:

$$\Phi(c) = P(Z \leq c),\ Z \sim N(0,1)$$

```
curve(dnorm(x),
      xlim = c(-3.5, 3.5),
      ylab = "density",
      main = "standard normal CDF")
```



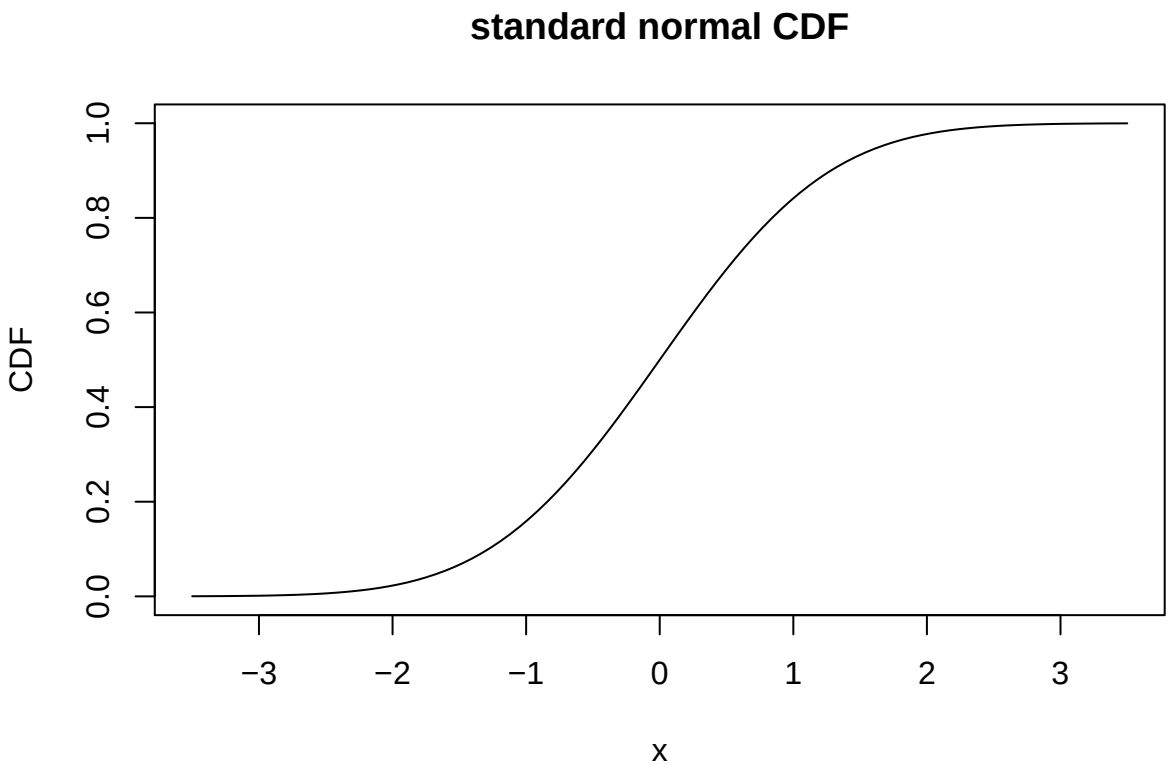
We can get the PDF value of the normal distribution by passing a vector of values to **dnorm**.

```
dnorm(x = c(-1.96, 0, 1.96))
```

```
## [1] 0.05844094 0.39894228 0.05844094
```

On a similar note we can generate the CDF of the standard normal by utilizing the **pnorm** function.

```
curve(pnorm(x),
      xlim = c(-3.5, 3.5),
      ylab = "CDF",
      main = "standard normal CDF")
```



A common result that we can verify is that in a standard normal distribution, 95% of the probability mass lies in the range $[-1.96, 1.96]$. We confirm this by calculating:

$$P(-1.96 \leq Z \leq 1.96) = 1 - 2 \times P(Z \leq -1.96)$$

```
1 - 2*pnorm(-1.96)
```

```
## [1] 0.9500042
```

So far we have seen computations using the standard normal distribution. However we can explicitly specify the **mean** and **standard deviation** of a variable and accordingly compute the PDF and CDF. In the example below we compute $P(3 \leq Y \leq 4)$ for $Y \sim N(5, 25)$.

```
pnorm(4, mean=5, sd = 5) - pnorm(3, mean=5, sd=5)
```

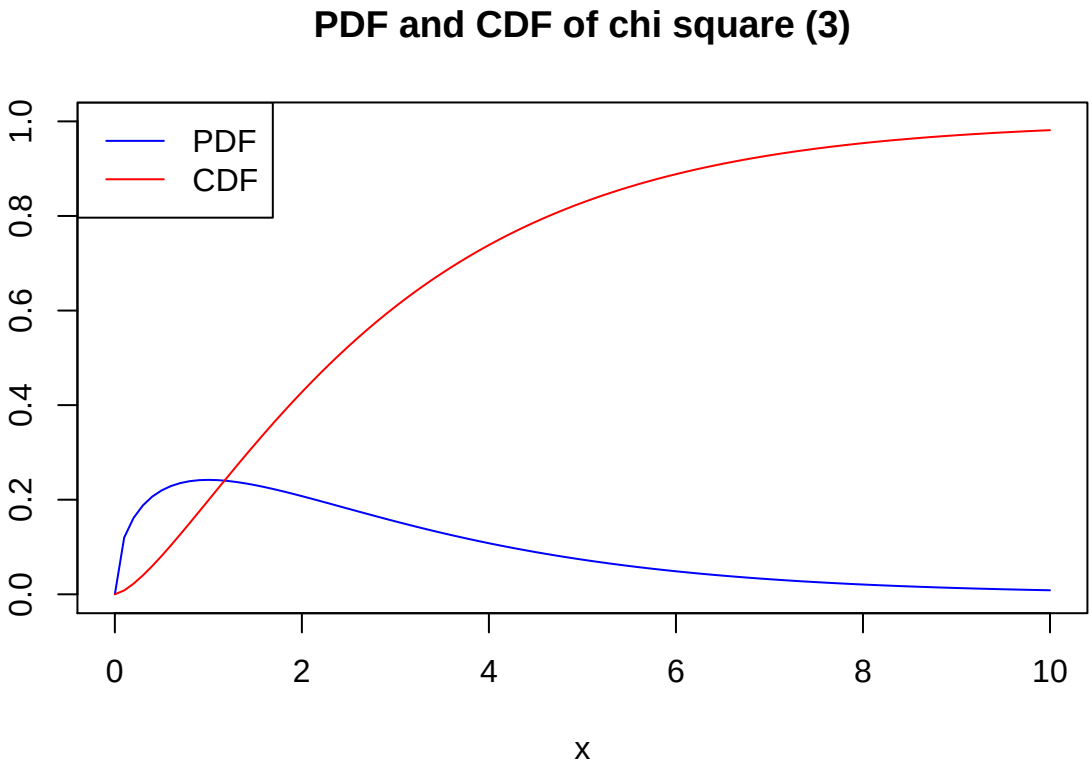
```
## [1] 0.07616203
```

Chi square distribution

The sum of M squared standard normal variables follows a **chi square** distribution with M degrees of freedom. So for $Z_m \sim N(0, 1)$ we have:

$$\sum_{m=1}^{M} Z_m^2 \sim \chi_M^2$$

```r
curve(dchisq(x, df = 3),
      xlim = c(0, 10),
      ylim = c(0, 1),
      col = "blue",
      ylab = "",
      main = "PDF and CDF of chi square (3)")

curve(pchisq(x, df = 3),
      xlim = c(0, 10),
      add = TRUE,
      col = "red")

legend("topleft",
       c("PDF", "CDF"),
       col = c("blue", "red"),
       lty = c(1, 1))
```



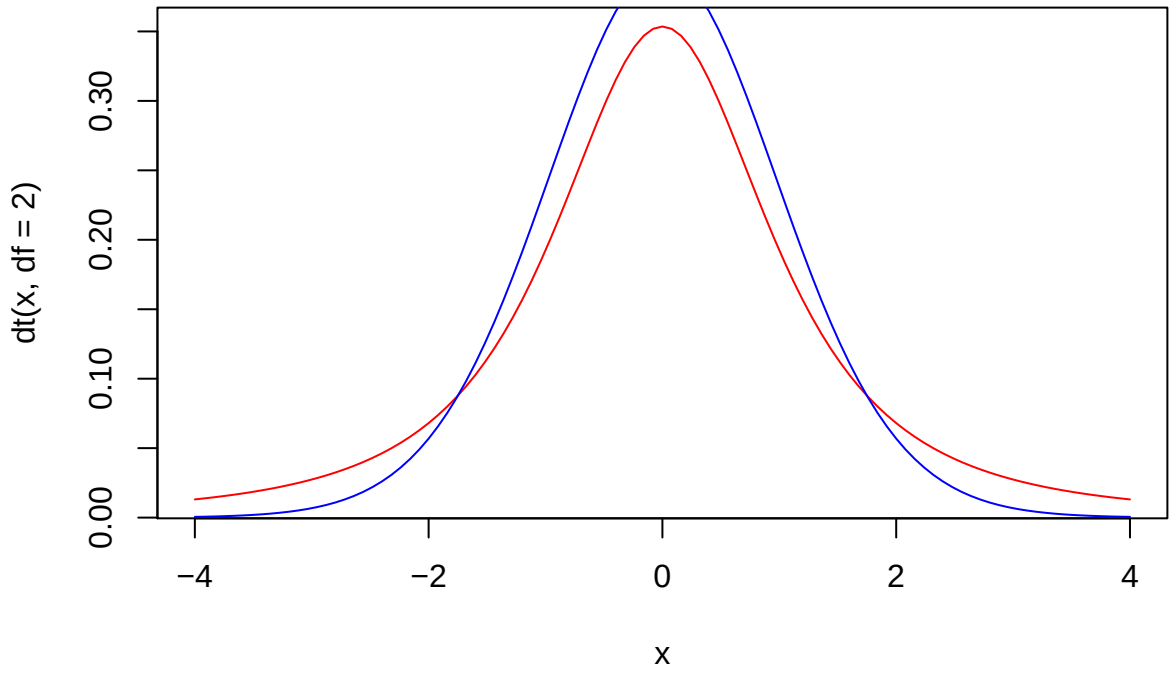
Student t distribution

If Z is a standard normal variable and W is a χ^2_M random variable and if they both are independent, then:

$$X = \frac{Z}{W/M} \sim t_M$$

The above expression follows a t distribution with M degrees of freedom. We note that if M is very large, then the t distribution resembles a standard normal distribution.

```
curve(dt(x, df = 2),
      xlim = c(-4, 4),
      col = 2)

curve(dt(x, df = 30),
      xlim = c(-4, 4),
      col = 4,
      add = TRUE)
```



The F distribution

The ratio of two independently distributed χ^2 random variables divided by their respective degrees of freedom forms the F distribution. For $W \sim \chi^2_M$ and $V \sim \chi^2_n$ we have:

$$\frac{W/M}{V/n} \sim F_{M,n}$$

Here is a simple example of calculating $P(Y \geq 2)$ for an F distribution with numerator degrees of freedom as 3 and denominator degrees of freedom as 14. Since it is 'greater than' we will add the additional argument **lower.tail = false** in the function.

```r
pf(2, df1 = 3, df2 = 14, lower.tail = FALSE)
```

```
## [1] 0.1603538
```