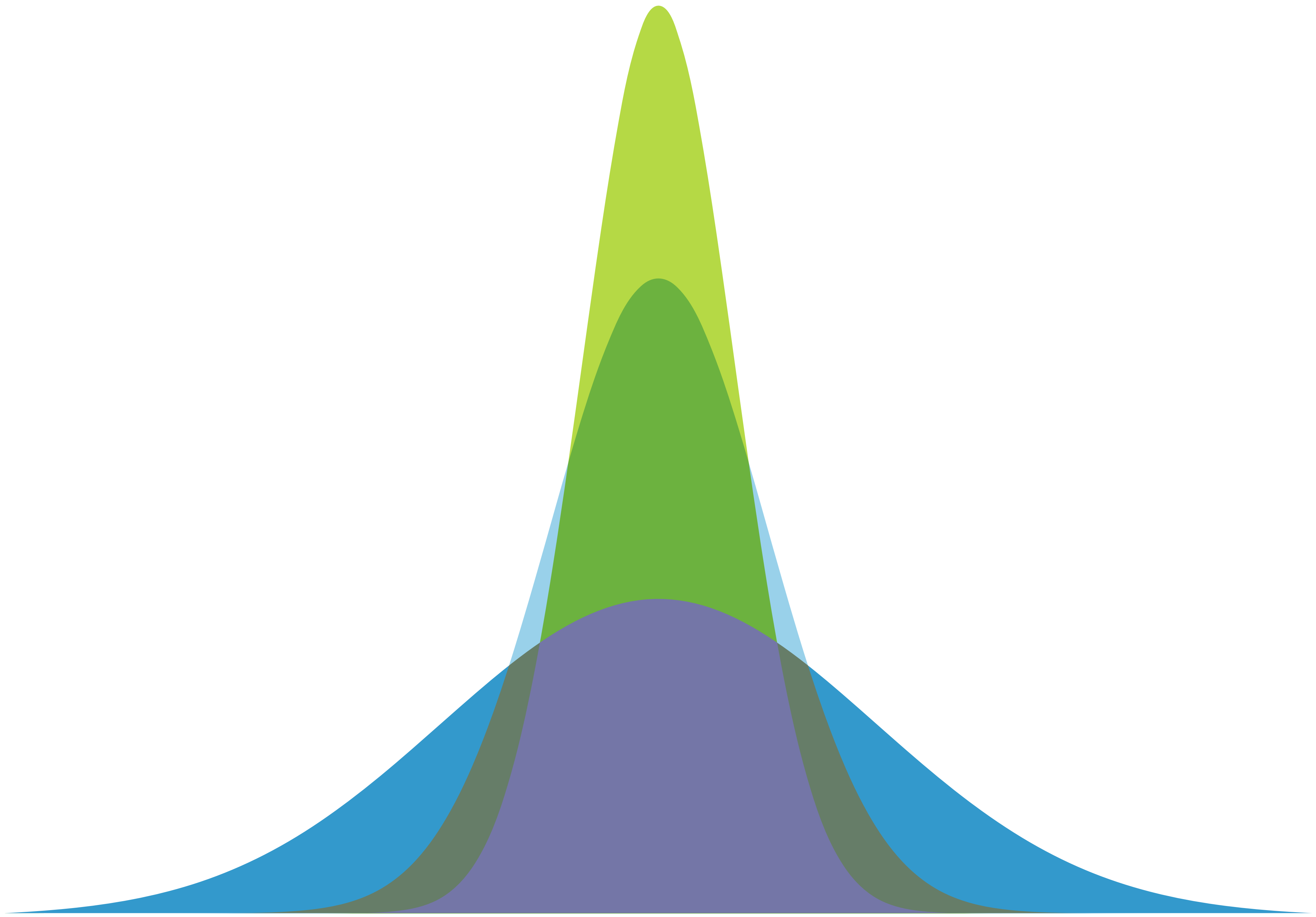
Is it normal?



Variety of names

Different names for the same concept

- Normal distribution
- Bell curve
- Gaussian distribution

Origin

Carl Friedrich Gauss (1777 - 1855)

Mathematical representation

$$y = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)}{2\sigma^2}}$$

Original equation

$$y = f(x, \mu, \sigma)$$

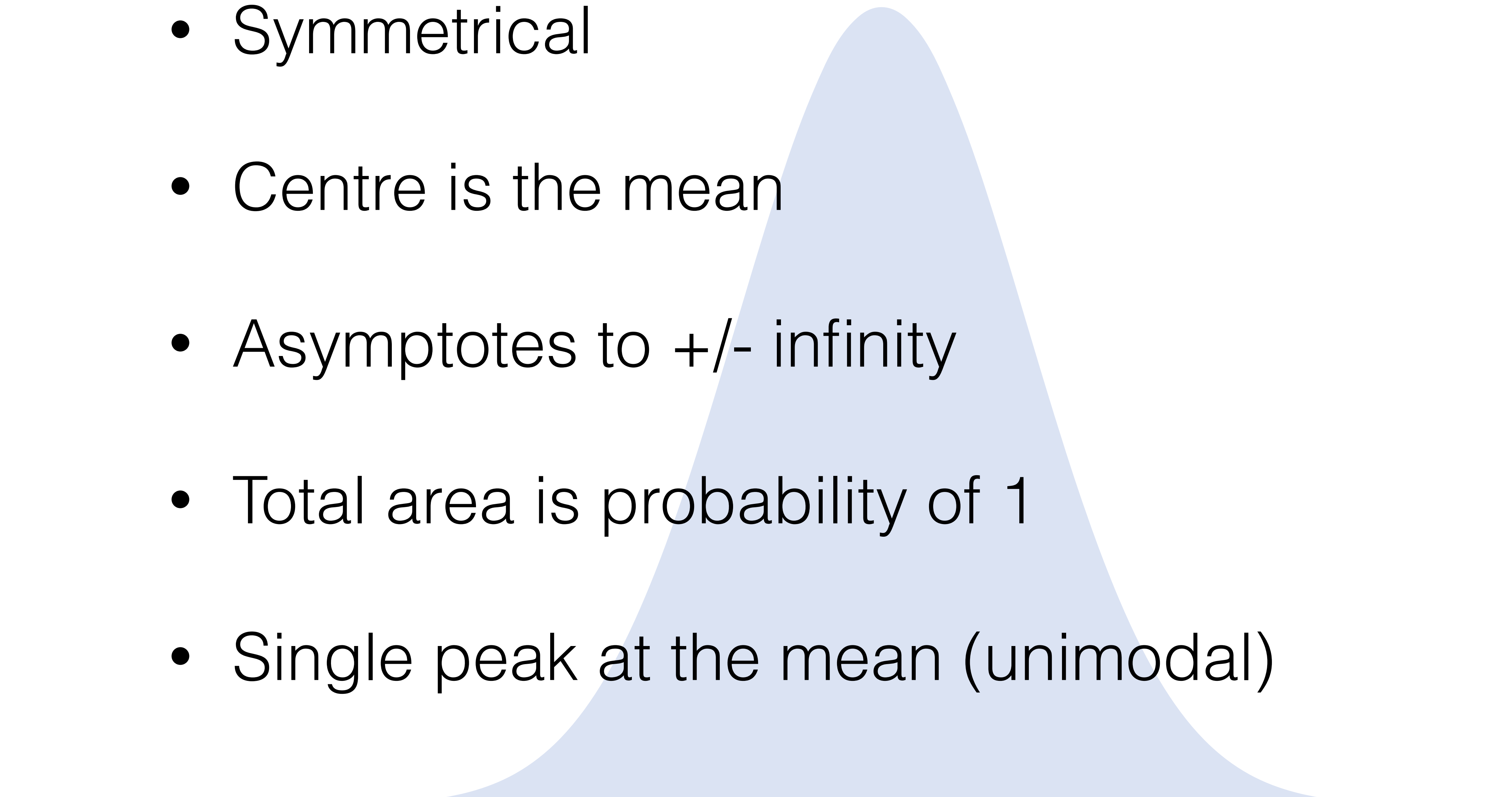
Three parameter function

Implementation in R

```
dnorm(x, mean = 0, sd = 1, log == FALSE)
```

What is the normal distribution?

- Density function - integrate to get probabilities
- Symmetrical
- Centre is the mean
- Asymptotes to +/- infinity
- Total area is probability of 1
- Single peak at the mean (unimodal)



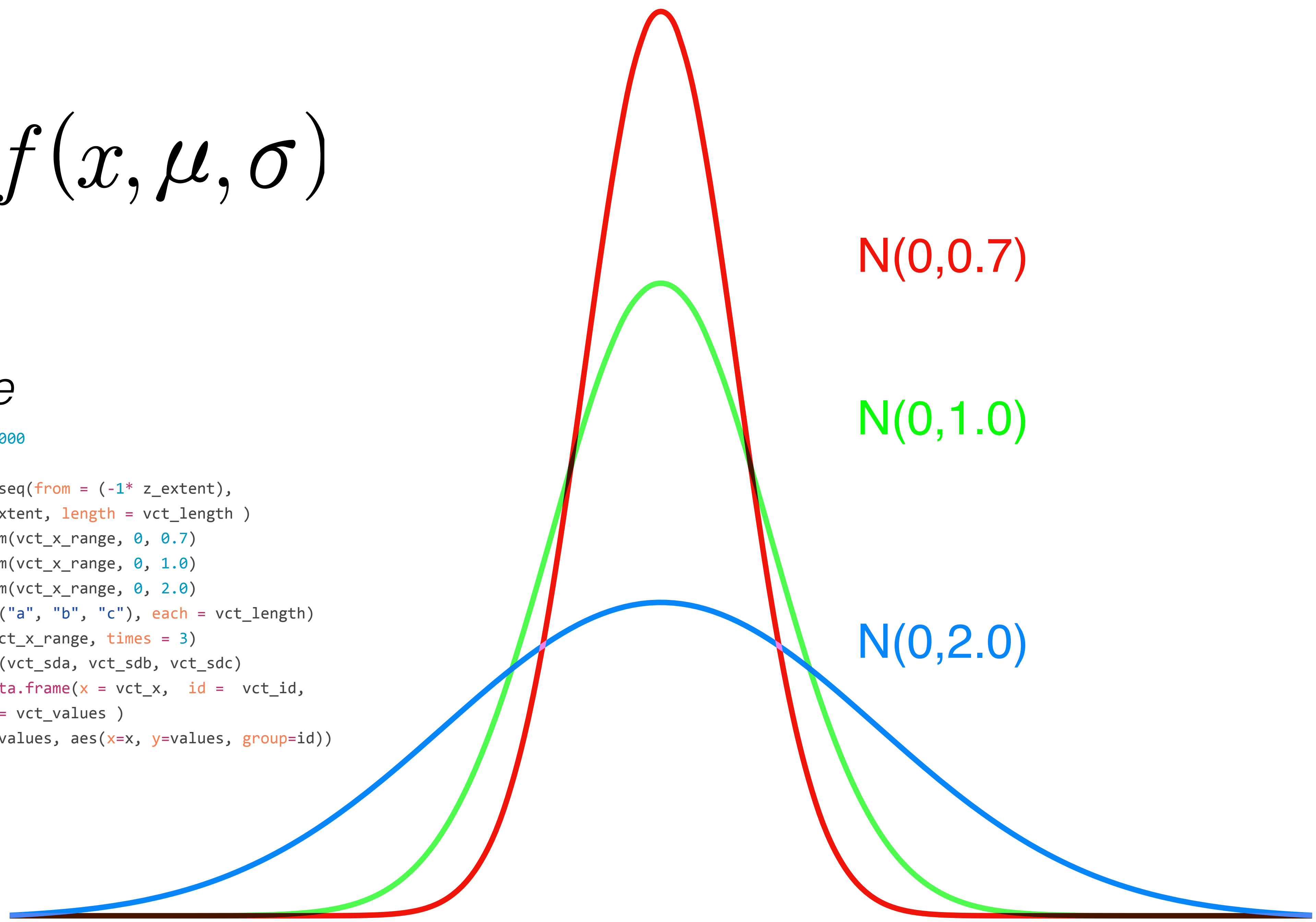
Family of distributions

Varying the standard deviation

$$y = f(x, \mu, \sigma)$$

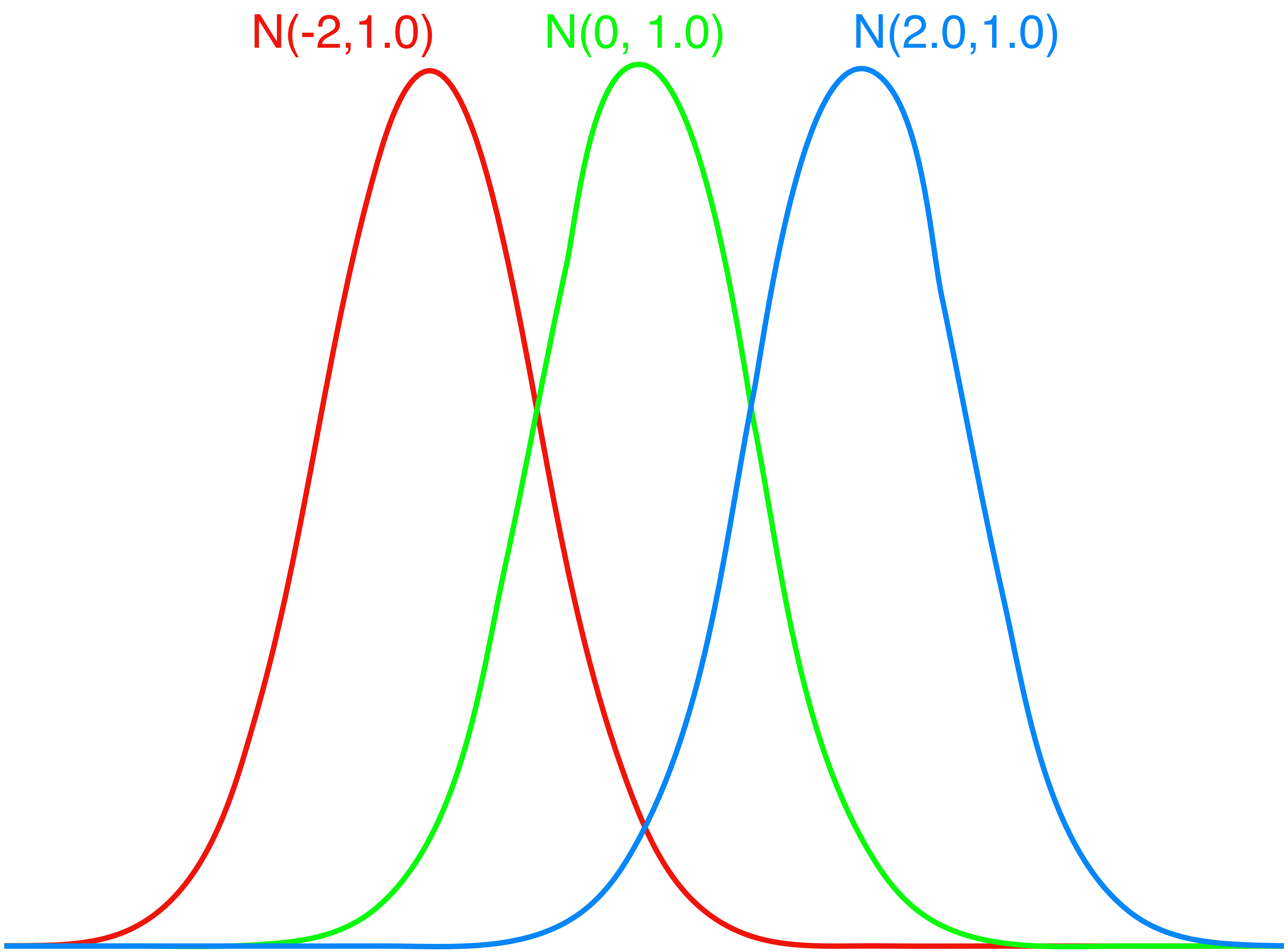
R code

```
vct_length <- 2000
z_extent <- 6
vct_x_range <- seq(from = (-1* z_extent),
      to  = z_extent, length = vct_length )
vct_sda <- dnorm(vct_x_range, 0, 0.7)
vct_sdb <- dnorm(vct_x_range, 0, 1.0)
vct_sdc <- dnorm(vct_x_range, 0, 2.0)
vct_id <- rep(c("a", "b", "c"), each = vct_length)
vct_x  <- rep(vct_x_range, times = 3)
vct_values <- c(vct_sda, vct_sdb, vct_sdc)
df_values <- data.frame(x = vct_x,  id =  vct_id,
        values = vct_values )
p <- ggplot(df_values, aes(x=x, y=values, group=id))
p + geom_line()
```



Family of distributions

Varying the mean



Standard normal distribution

Mean is zero

Subtract the mean from each observation

Standard deviation is 1

Divide each observation by the standard deviation

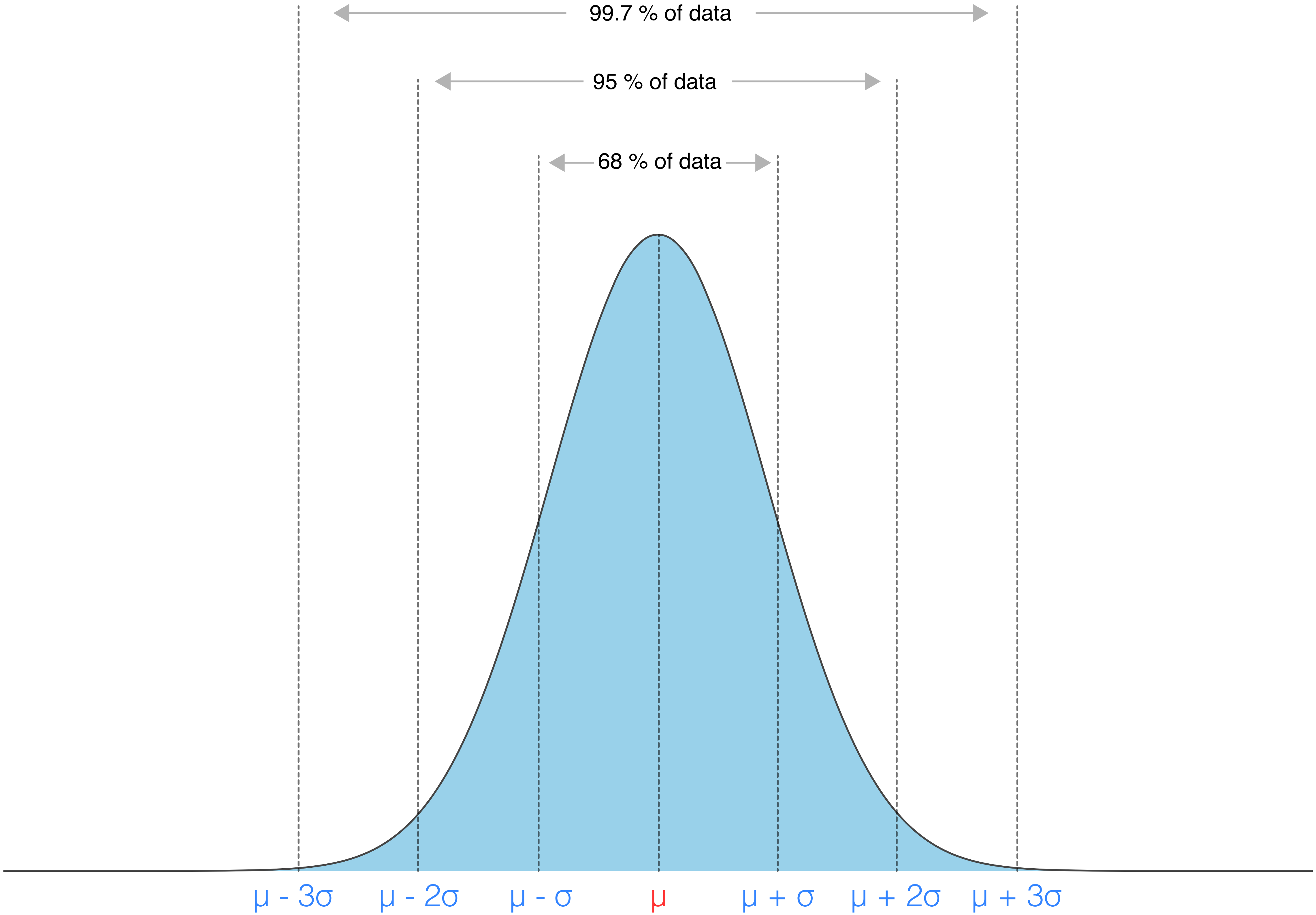
To do this in R

```
data <- 0:100  # mean = 50; sd = 29.3
converted_data <- (data - mean(data)) / sd(data)
mean(converted_data) # this is zero
sd(converted_data) # this is 1
```

Or using the scale() function

```
data <- 0:100
converted_data <- scale(data)
```

Standard normal distribution



IQ Scores

- IQ = Intelligence quotient
- Normal distribution
- A measure of intelligence, developed in 1912.
- Mean = 100
- Standard deviation = 15

What percentage of the population has an IQ between 85 to 115 ?

Calculating probabilities

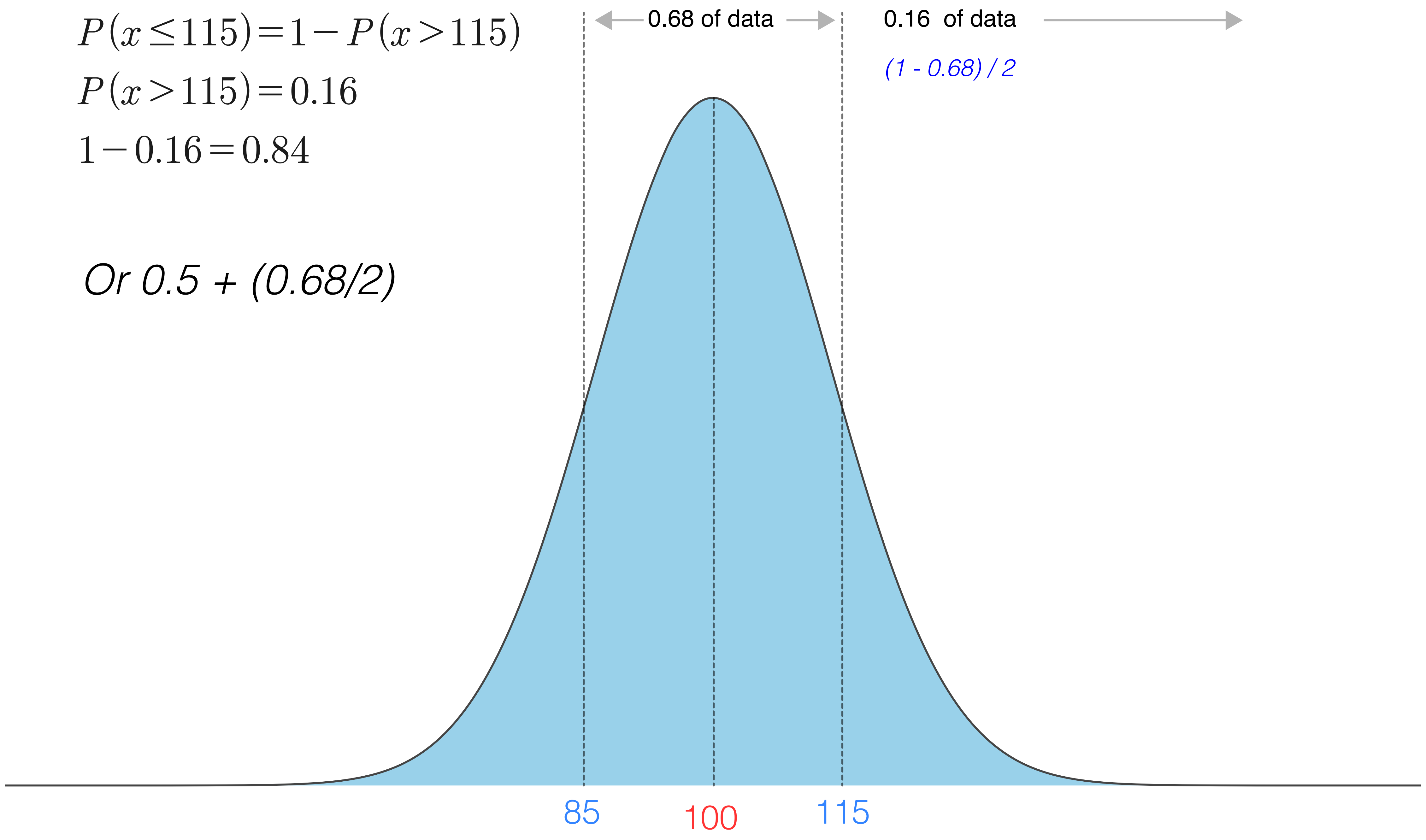
What probability is there that a randomly selected person has an IQ of less than or equal to 115?

$$P(x \leq 115) = 1 - P(x > 115)$$

$$P(x > 115) = 0.16$$

$$1 - 0.16 = 0.84$$

Or 0.5 + (0.68/2)



Simulation in R

- Random number generation
- Set of functions prefixed by “r”
- suffix is the name of the distribution.

Many other distributions: binomial (binom - eg rbinom); cauchy (cauchy); poisson (pois); uniform(unif); chi-square (chisq); F (f)…..

Calculating the previous example using simulation. 10,000 numbers

```
set.seed(123) # makes things repeatable

n_obs <- 10000

random_iq <- rnorm(n_obs, mean = 100, sd = 15)

length(random_iq[random_iq <= 115]) / n_obs # result = 0.8424
```

Families of distribution functions

Prefix	Description	Example
d	For density. Generate normal curve	***dnorm*** - generate a normal density function
r	Random number generation	***rpois*** - generate random Poisson variates
p	For cumulative distribution	***pchisq*** - cumulative Chi-squared distribution. Area to the left of a given value of x.
q	For quantile distribution	***qt*** - quantile for a Student's t distribution

Usage of distribution functions

Cumulative distribution functions

Probability of an IQ score less than or equal to 115 [“lower.tail = TRUE” is default]

```
# P(x <= 115)
pnorm(115, mean = 100, sd = 15) # result = 0.8413

# P(x > 115)
pnorm(115, mean = 100, sd = 15, lower.tail = FALSE) # result = 0.1586
```

Quantile distribution functions

Reciprocal *of Cumulative distribution functions [“lower.tail = TRUE” is default]*

```
# What IQ is asociated with P(x <= 0.8413)
qnorm(0.8413, mean = 100, sd = 15) # result is 114.997

# What is the IQ that puts you in the top 5%
qnorm(0.95, mean = 100, sd = 15) # result is 124.67

# same result as previous
qnorm(0.05, mean = 100, sd = 15, lower.tail = FALSE)
```

Example of normal distribution

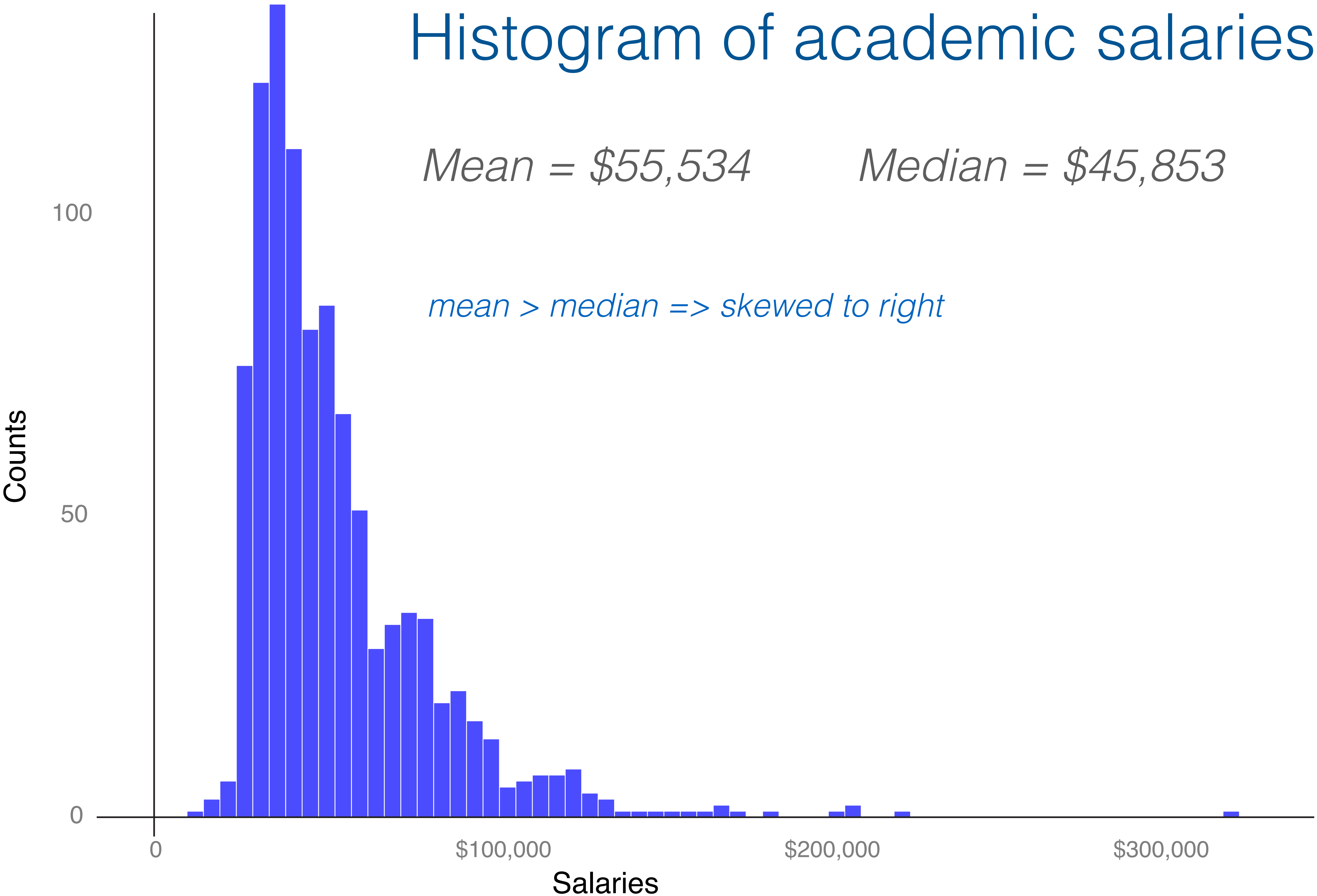
- Academic salaries
- Observations = 988
- Mean =$55,534
- Standard deviation = $29,107
- Median = $47,853
- Range = $11,000 ~ $325,500

Data source:
https://opendata.socrata.com/dataset/FGCU-salary-dataset/fjqw-ymup

Histogram of academic salaries

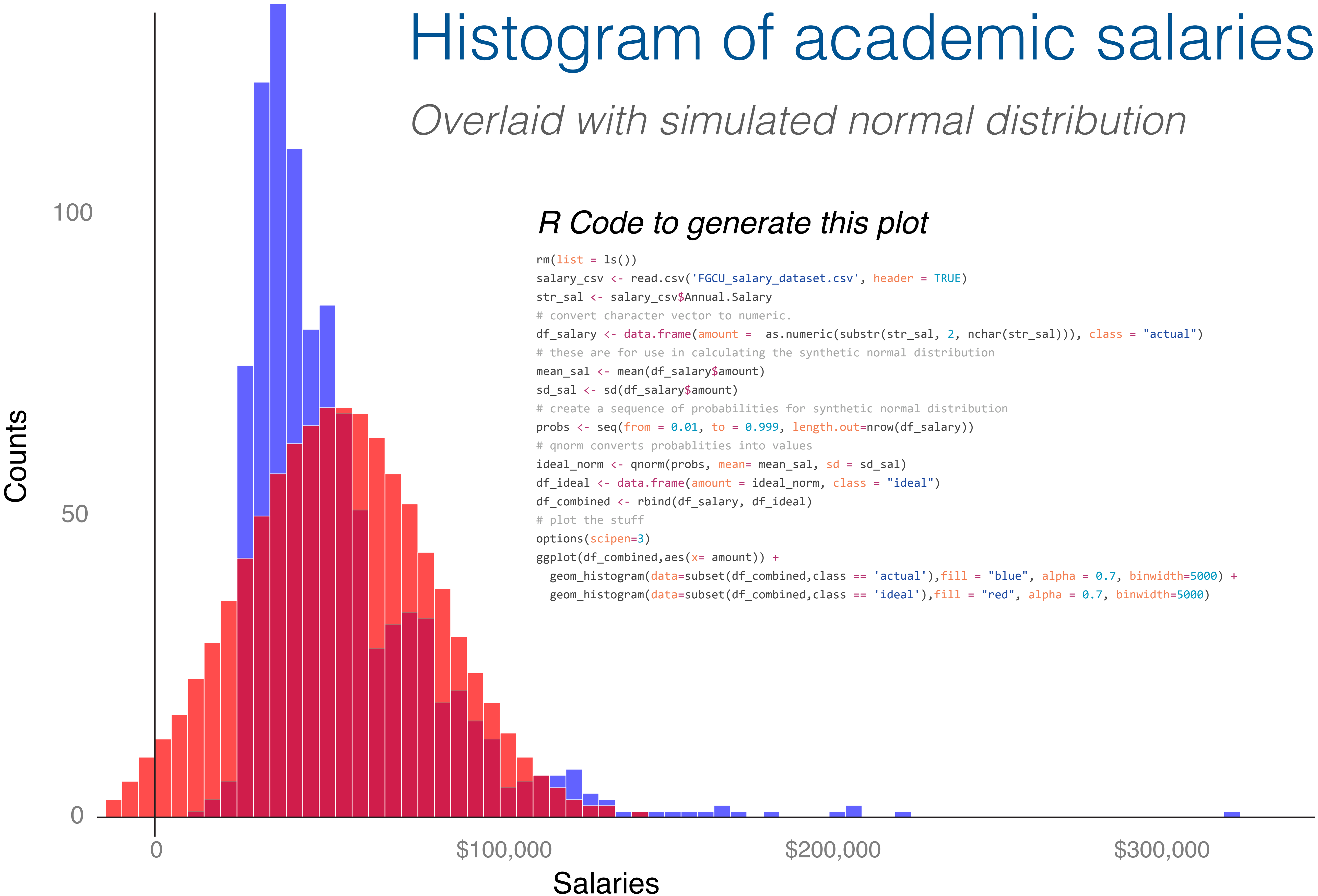
Mean = $55,534   *Median = $45,853*

mean > median => skewed to right



Histogram of academic salaries

Overlaid with simulated normal distribution

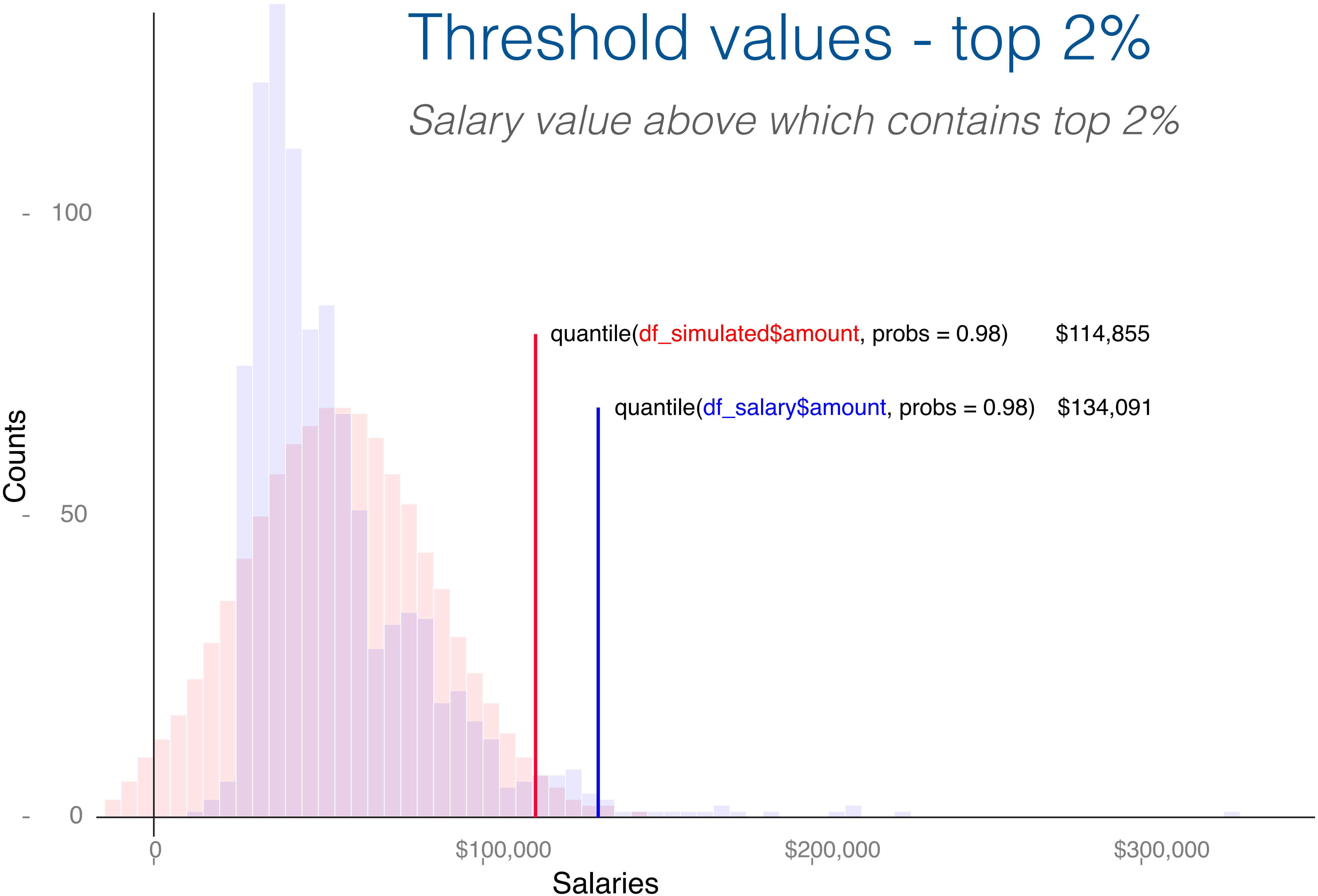


R Code to generate this plot

```
rm(list = ls())
salary_csv <- read.csv('FGCU_salary_dataset.csv', header = TRUE)
str_sal <- salary_csv$Annual.Salary
# convert character vector to numeric.
df_salary <- data.frame(amount =  as.numeric(substr(str_sal, 2, nchar(str_sal))), class = "actual")
# these are for use in calculating the synthetic normal distribution
mean_sal <- mean(df_salary$amount)
sd_sal <- sd(df_salary$amount)
# create a sequence of probabilities for synthetic normal distribution
probs <- seq(from = 0.01, to = 0.999, length.out=nrow(df_salary))
# qnorm converts probablities into values
ideal_norm <- qnorm(probs, mean= mean_sal, sd = sd_sal)
df_ideal <- data.frame(amount = ideal_norm, class = "ideal")
df_combined <- rbind(df_salary, df_ideal)
# plot the stuff
options(scipen=3)
ggplot(df_combined,aes(x= amount)) +
  geom_histogram(data=subset(df_combined,class == 'actual'),fill = "blue", alpha = 0.7, binwidth=5000) +
  geom_histogram(data=subset(df_combined,class == 'ideal'),fill = "red", alpha = 0.7, binwidth=5000)
```

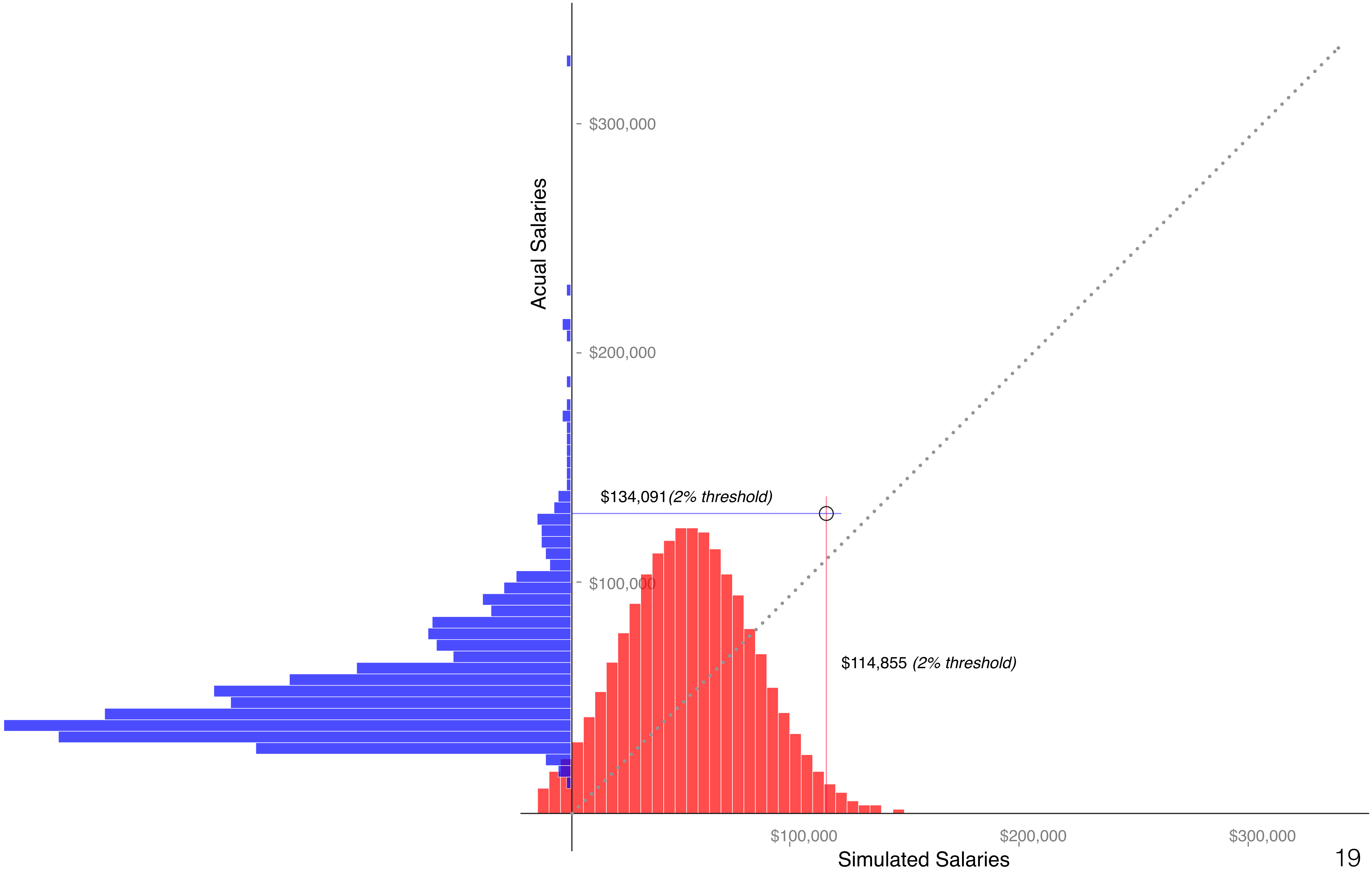
Threshold values - top 2%

Salary value above which contains top 2%

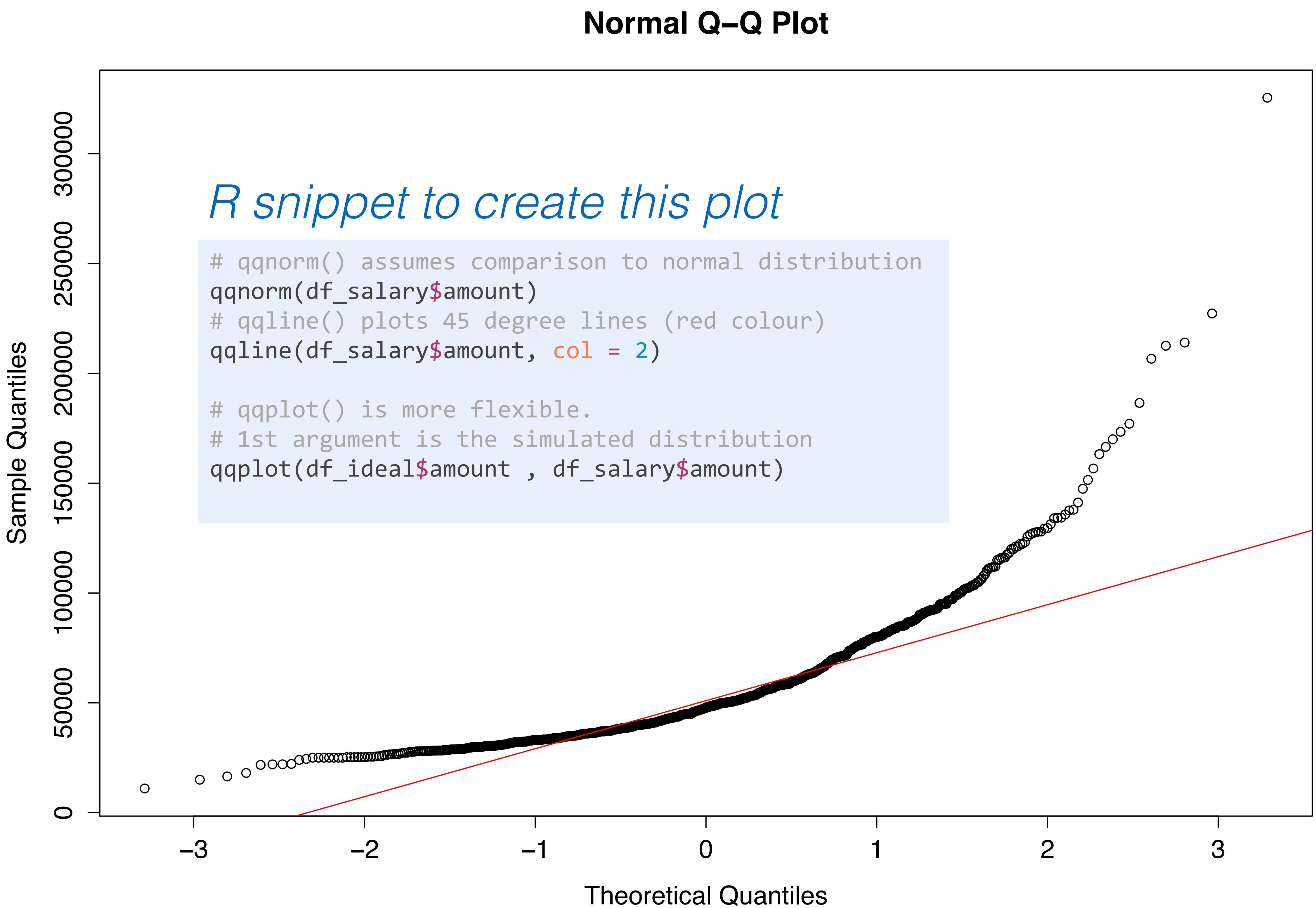


Quantile / Quantile Plot (QQ Plot)

Comparison of 2% threshold between actual and simulated data



QQ Plot in R



R snippet to create this plot

```
# qqnorm() assumes comparison to normal distribution
qqnorm(df_salary$amount)
# qqline() plots 45 degree lines (red colour)
qqline(df_salary$amount, col = 2)

# qqplot() is more flexible.
# 1st argument is the simulated distribution
qqplot(df_ideal$amount , df_salary$amount)
```

Central Limit Theorem

Main justification to use the Normal distribution

*The **sum** of independent samples from **ANY** distribution converges to the normal distribution with a sufficiently large sample size.*

Some quantities are expected to be the sum of many independent processes.

This convergence is faster if the underlying distribution is normal

The mean is simply the sum divided by a constant

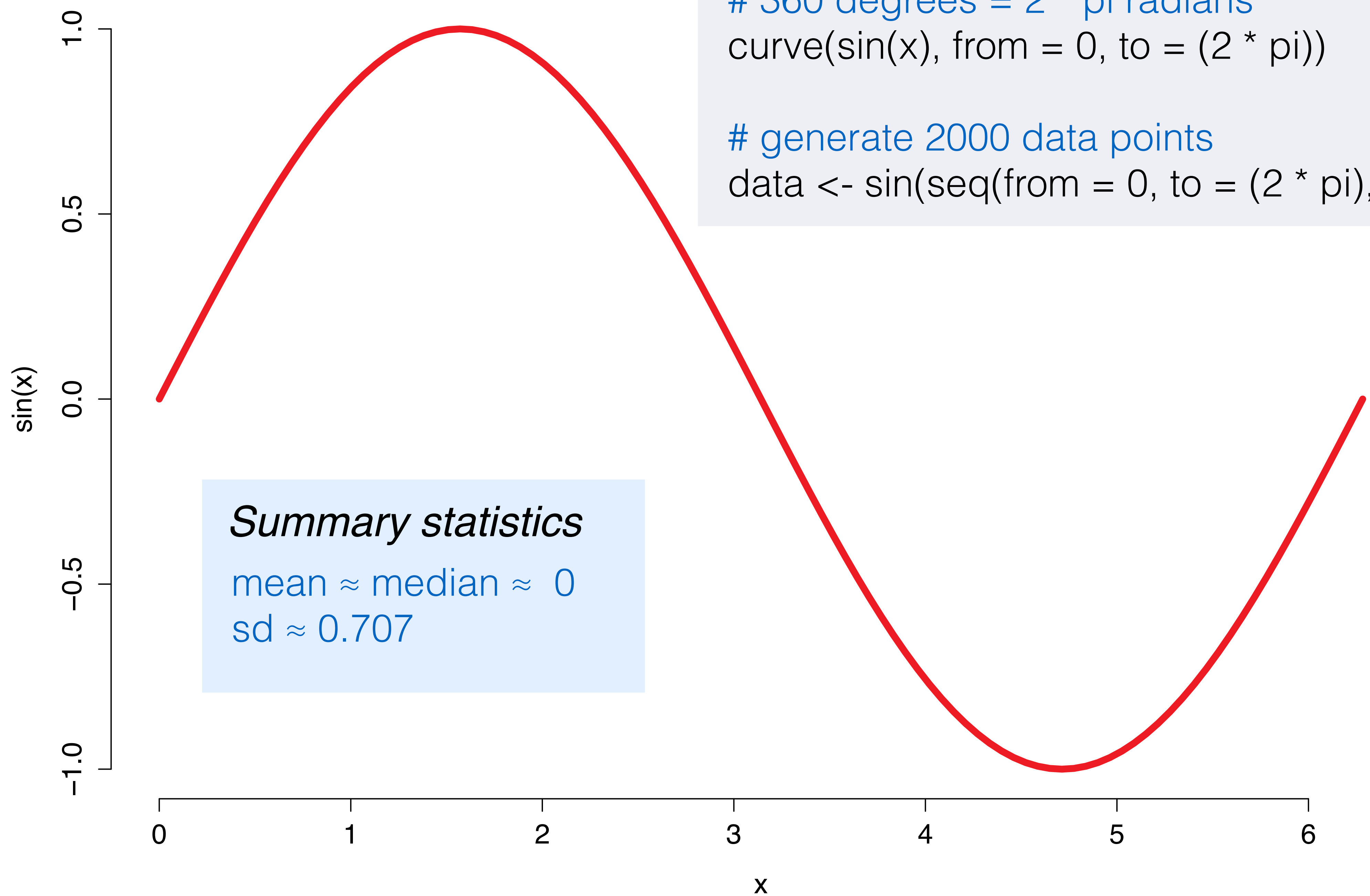
Central Limit Theorem - Simulation

Create a distribution (0 - 360 degree sine wave)

R Code to generate this plot

```
# 360 degrees = 2 * pi radians
curve(sin(x), from = 0, to = (2 * pi))

# generate 2000 data points
data <- sin(seq(from = 0, to = (2 * pi), length.out = 2000))
```

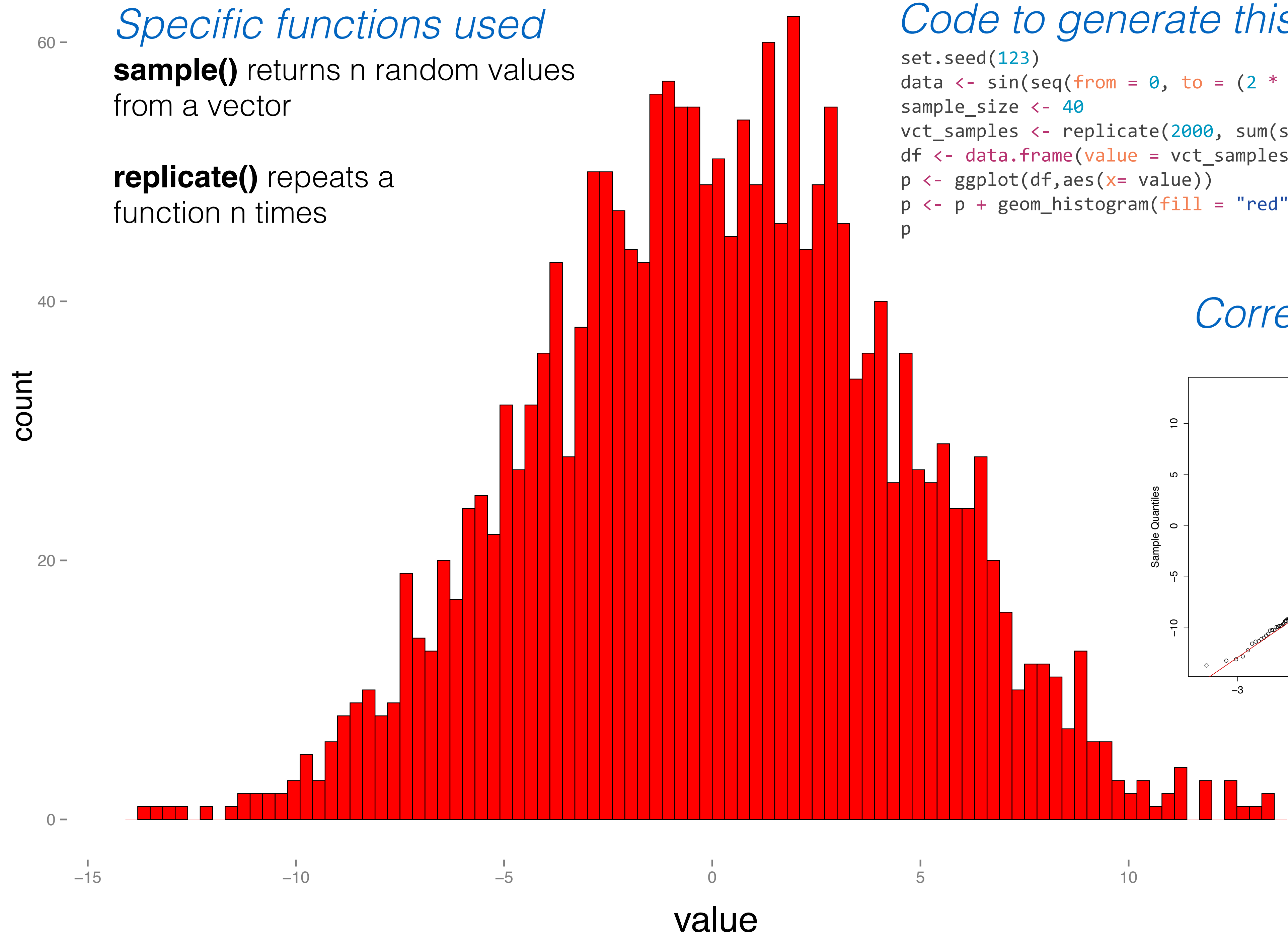


Summary statistics

mean ≈ median ≈ 0
sd ≈ 0.707

Central Limit Theorem - Simulation

Normal distribution created from 2000 samples of size = 40



Specific functions used

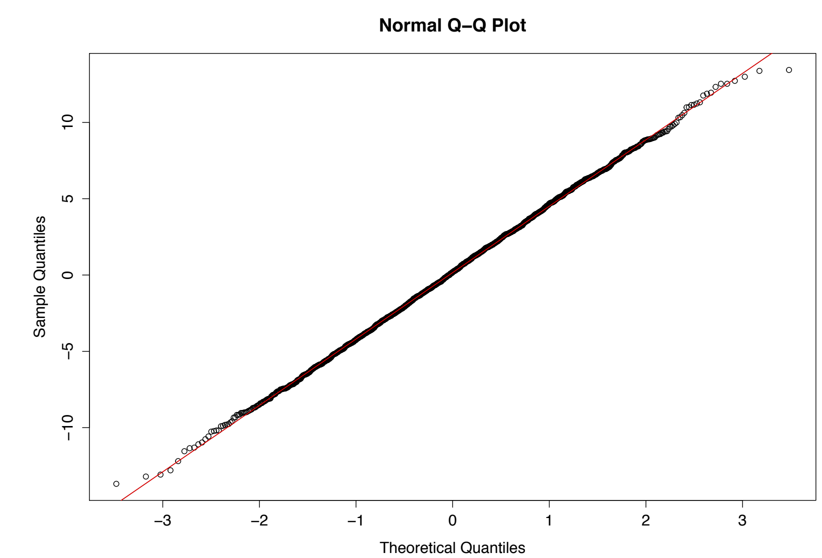
sample() returns n random values from a vector

replicate() repeats a function n times

Code to generate this plot

```
set.seed(123)
data <- sin(seq(from = 0, to = (2 * pi), length.out = 2000))
sample_size <- 40
vct_samples <- replicate(2000, sum(sample(data, sample_size, replace =FALSE)))
df <- data.frame(value = vct_samples)
p <- ggplot(df,aes(x= value))
p <- p + geom_histogram(fill = "red", alpha = 1, binwidth= 0.3)
p
```

Corresponding QQPlot

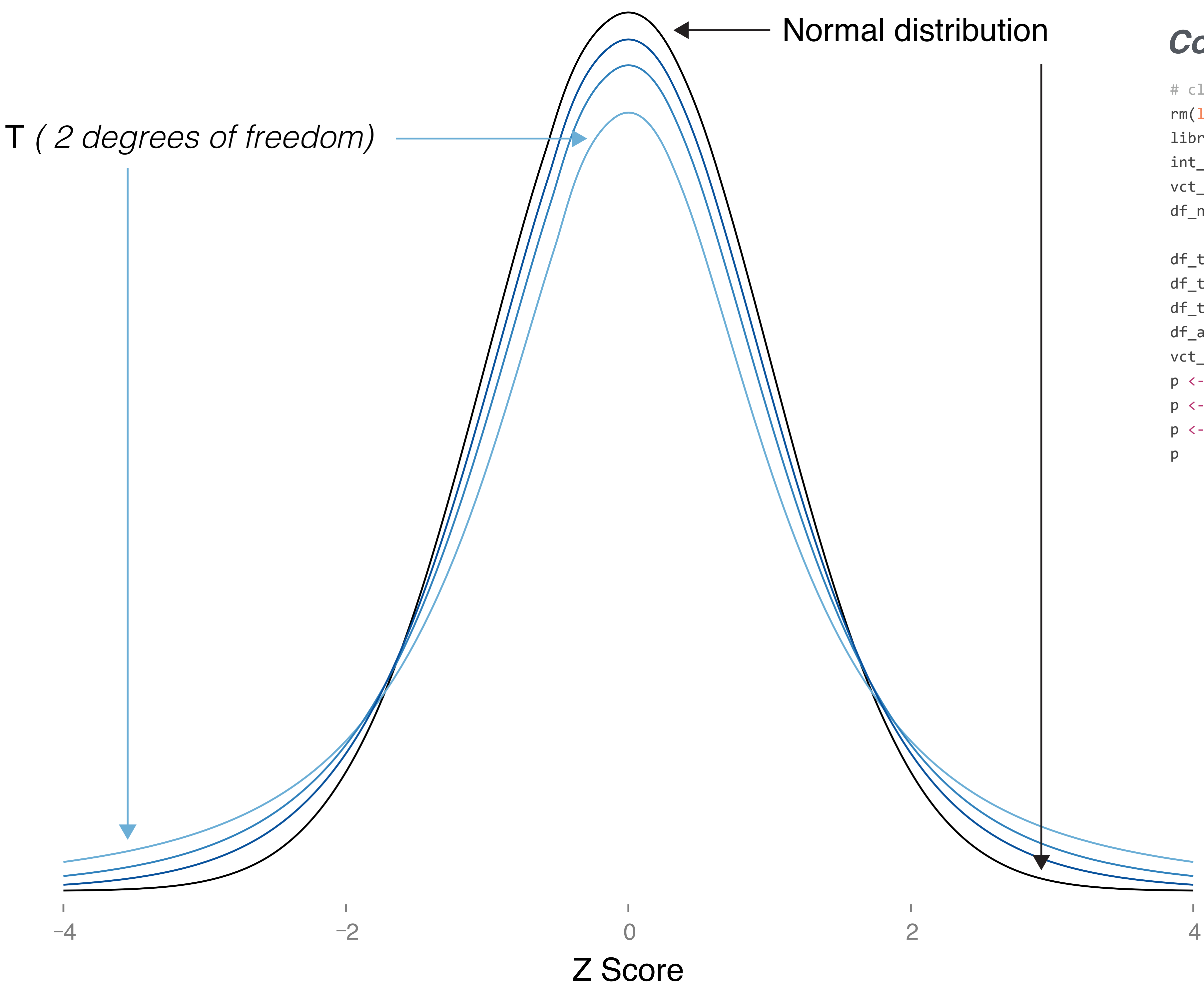


Related Distribution: Student’s T Distribution

- When the sample size is too small to converge to the Normal distribution
- Greater area under the tails. This means a higher probability of extreme values.
- Compared to the Normal distribution, there is an additional *‘degrees of freedom’* parameter

Student’s T Distribution compared to the Normal distribution (1)

Student’s T Distribution with 2, 4 & 8 degrees of freedom compared to the Normal distribution

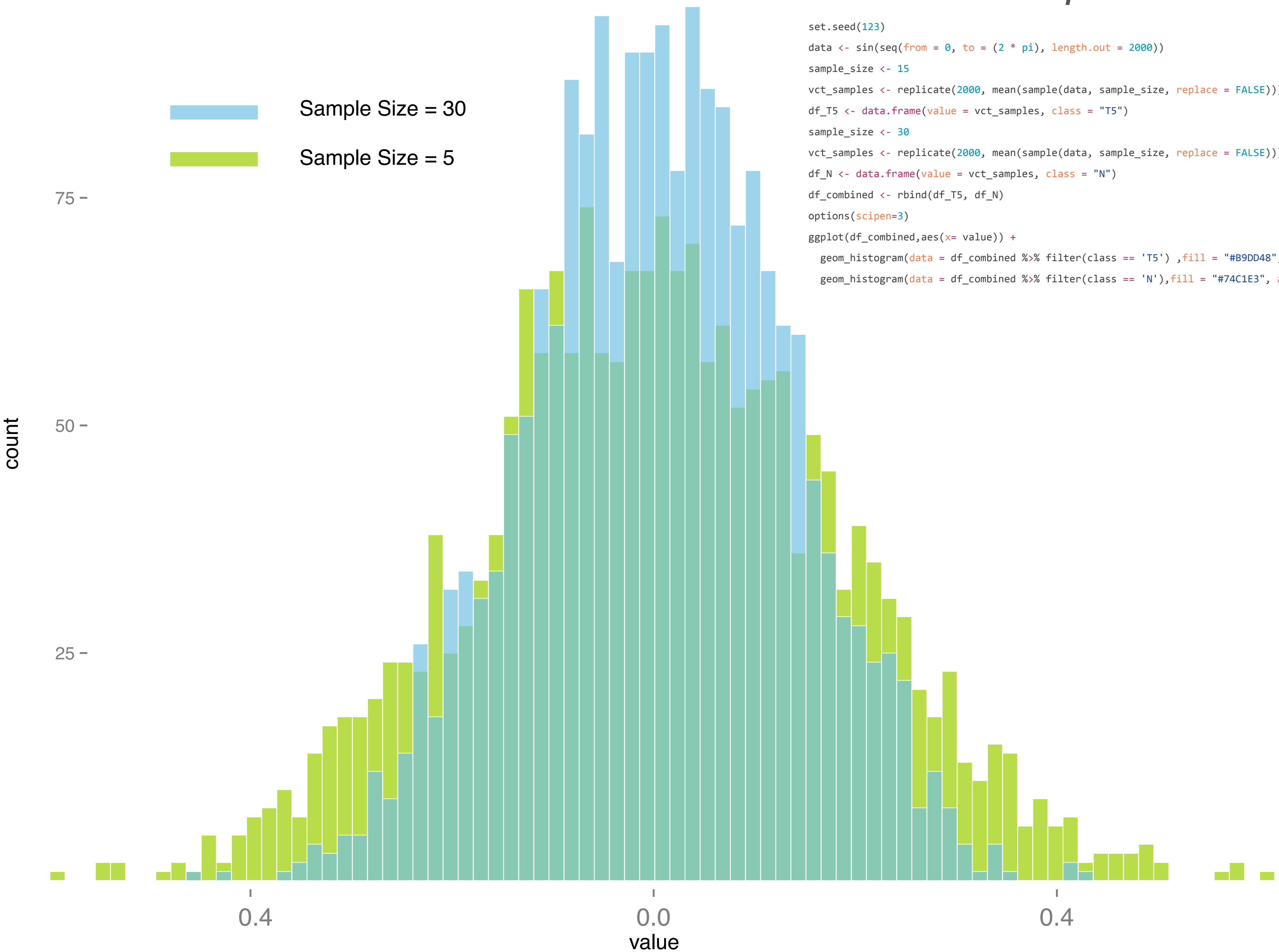


Code to create this plot

```
# clear everything
rm(list = ls())
library(ggplot2)
int_length <- 2000
vct_z <- seq(from = -4, to = 4, length.out = int_length)
df_norm <- data.frame(x = vct_z, y =
      dnorm(vct_z , mean = 0, sd = 1), group = "normal")
df_t2 <- data.frame(x = vct_z, y = dt(vct_z, 2), group = "t2")
df_t4 <- data.frame(x = vct_z, y = dt(vct_z, 4), group = "t4")
df_t8 <- data.frame(x = vct_z, y = dt(vct_z, 8), group = "t8")
df_all <- rbind(df_norm, df_t2, df_t4, df_t8)
vct_colors <- c("#000000", "#6baed6", "#3182bd", "#08519c")
p <- ggplot(df_all, aes(x = x, y = y, group = group))
p <- p + geom_line(aes(color = group))
p <- p + scale_color_manual(values = vct_colors)
p
```

Student’s T Distribution compared to the Normal distribution (2)

Simulation of 2000 size 30 samples and 2000 size 5 samples drawn from a non-normal distribution

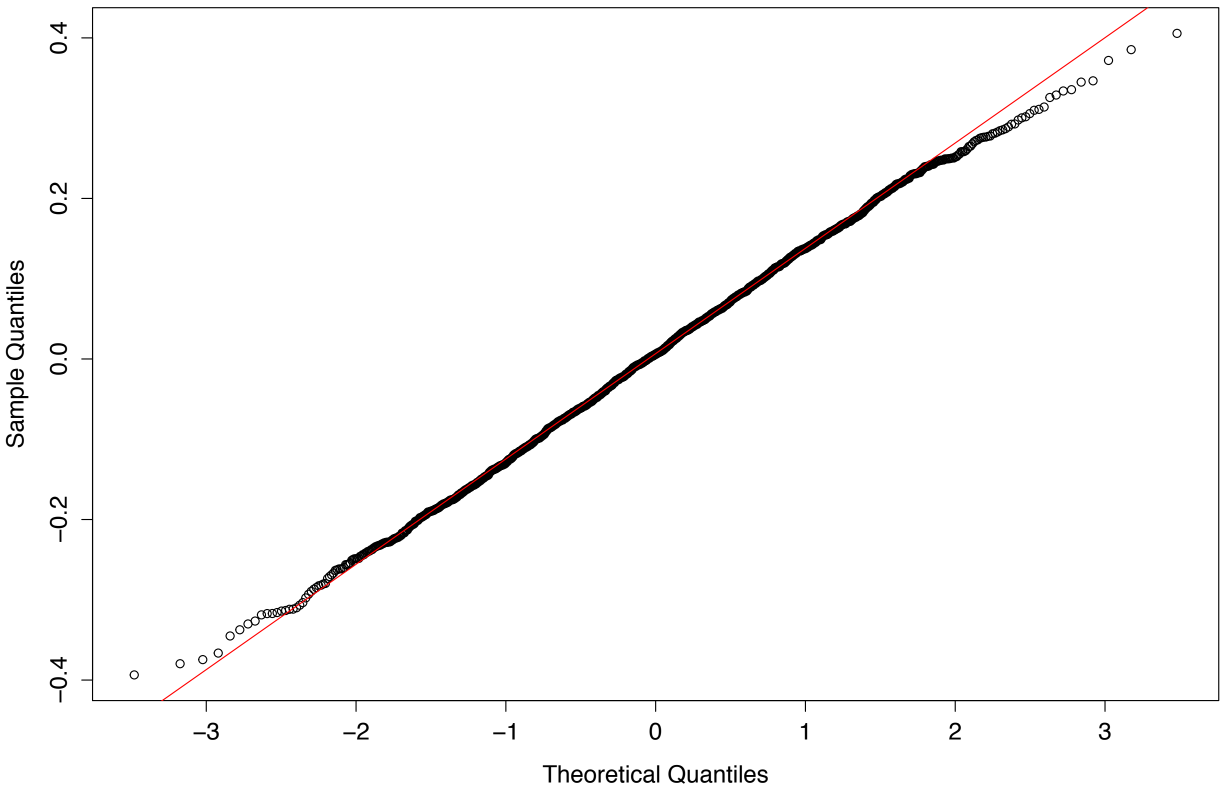


Code to create this plot

```
set.seed(123)
data <- sin(seq(from = 0, to = (2 * pi), length.out = 2000))
sample_size <- 15
vct_samples <- replicate(2000, mean(sample(data, sample_size, replace = FALSE)))
df_T5 <- data.frame(value = vct_samples, class = "T5")
sample_size <- 30
vct_samples <- replicate(2000, mean(sample(data, sample_size, replace = FALSE)))
df_N <- data.frame(value = vct_samples, class = "N")
df_combined <- rbind(df_T5, df_N)
options(scipen=3)
ggplot(df_combined,aes(x= value)) +
  geom_histogram(data = df_combined %>% filter(class == 'T5') ,fill = "#B9DD48", alpha = 0.7, binwidth=0.015) +
  geom_histogram(data = df_combined %>% filter(class == 'N'),fill = "#74C1E3", alpha = 0.7, binwidth=0.015)
```

Student's T Distribution compared to the Normal distribution (3)

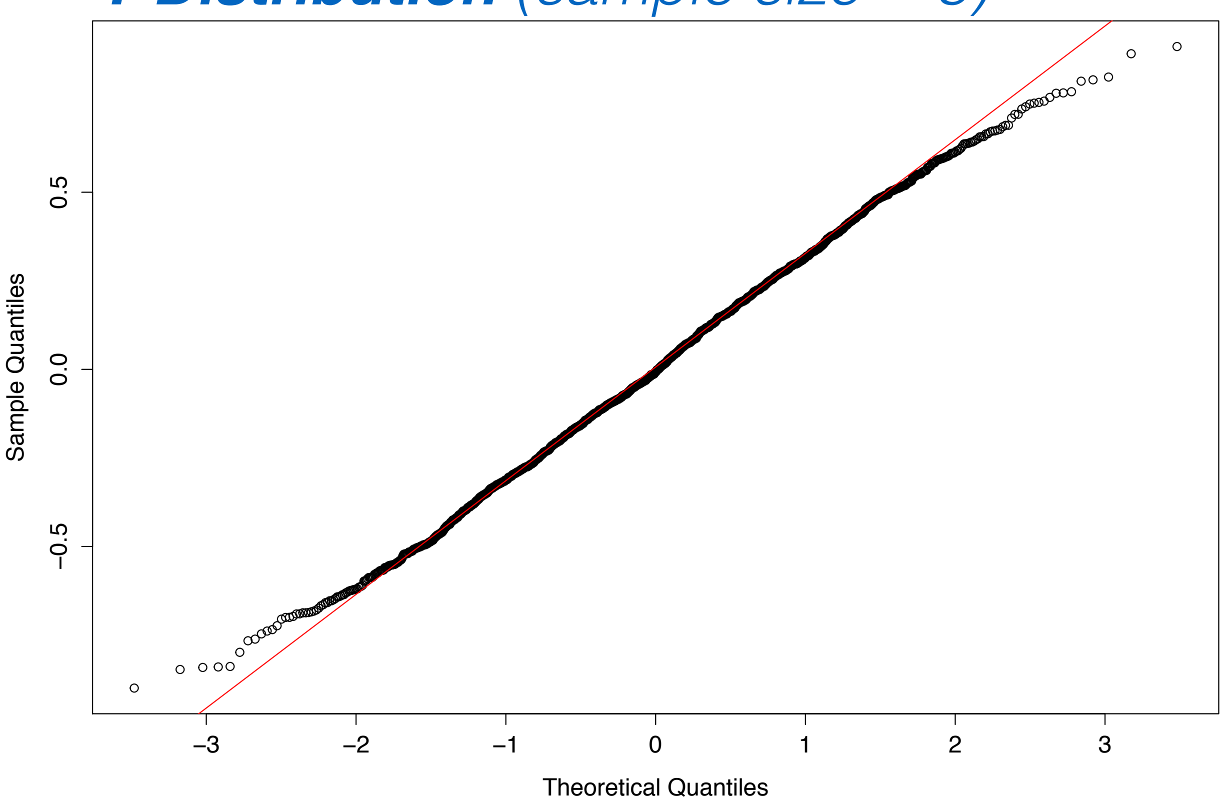
Approach a Normal Distribution *(sample size = 30)*



Code to create this plot

```
set.seed(123)
data <- sin(seq(from = 0, to = (2 * pi), length.out = 2000))
vct_N30 <- replicate(2000, mean(sample(data, size = 30, replace = FALSE)))
qqnorm(vct_N30, main = "")
qqline(vct_N30, col = 2)
```

T Distribution *(sample size = 5)*

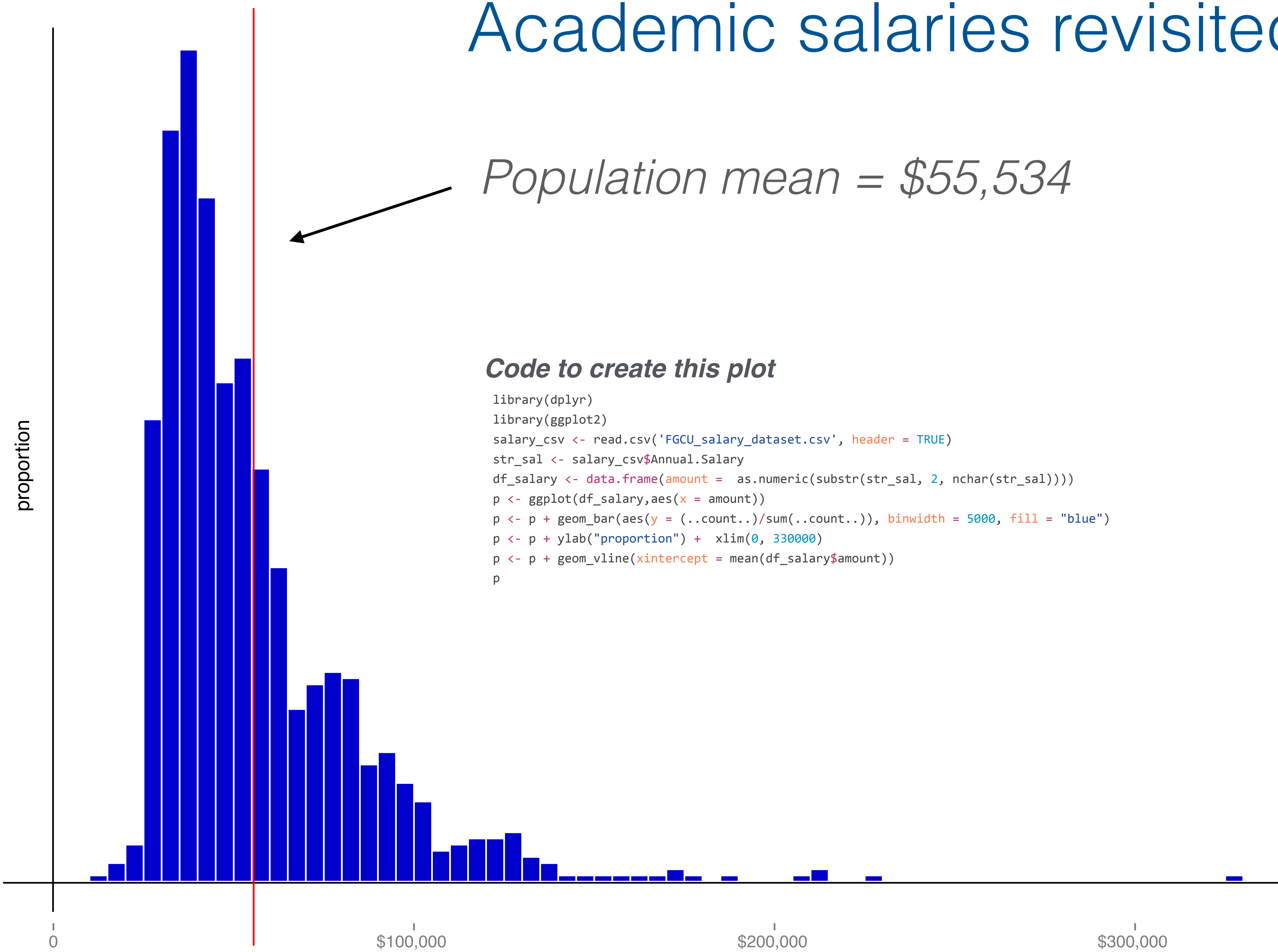


Code to create this plot

```
set.seed(123)
data <- sin(seq(from = 0, to = (2 * pi), length.out = 2000))
vct_T5 <- replicate(2000, mean(sample(data, size = 5, replace = FALSE)))
qqnorm(vct_T5, main = "")
qqline(vct_T5, col = 2)
```

Academic salaries revisited

Population mean = $55,534



Code to create this plot

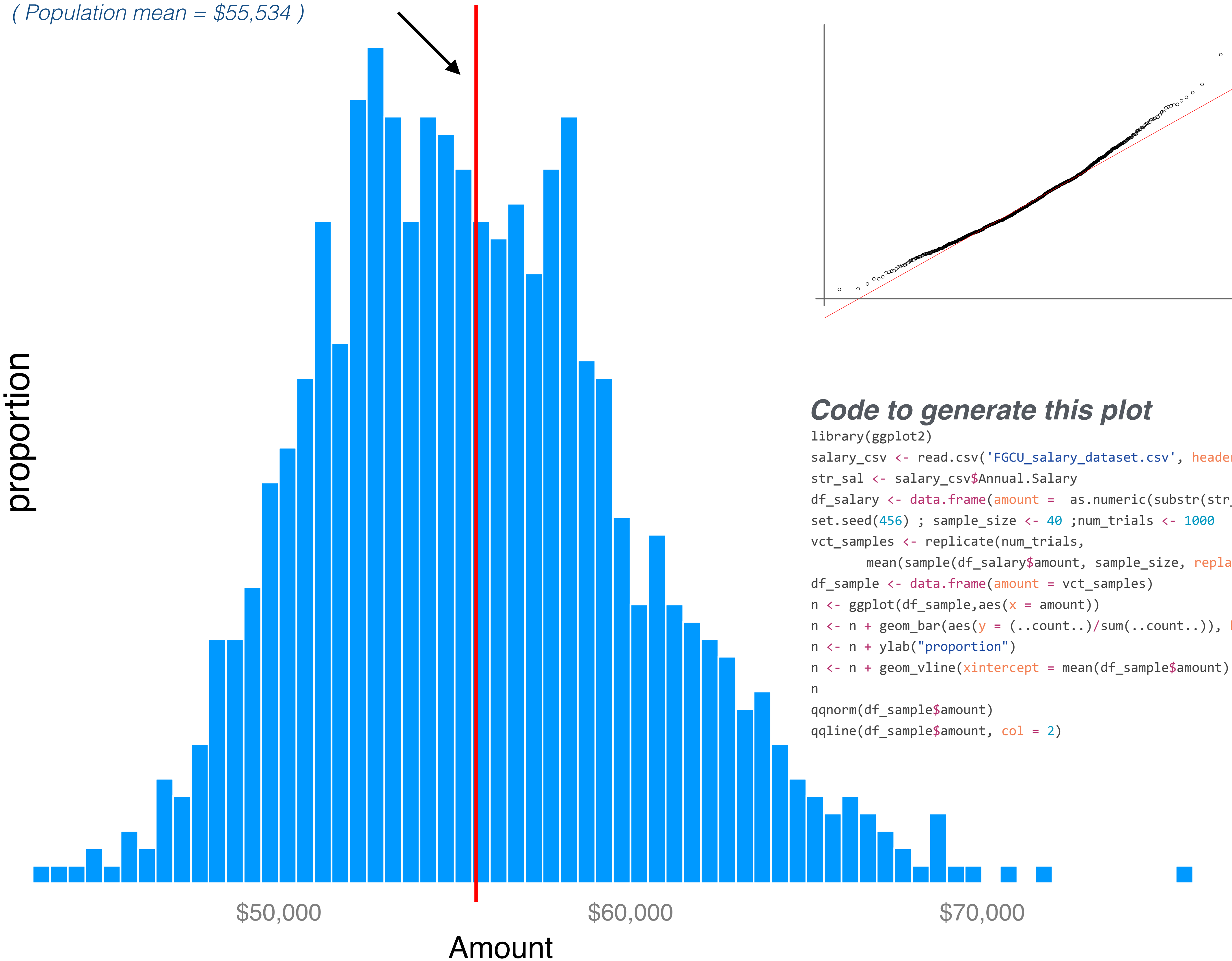
```
library(dplyr)
library(ggplot2)
salary_csv <- read.csv('FGCU_salary_dataset.csv', header = TRUE)
str_sal <- salary_csv$Annual.Salary
df_salary <- data.frame(amount =  as.numeric(substr(str_sal, 2, nchar(str_sal))))
p <- ggplot(df_salary,aes(x = amount))
p <- p + geom_bar(aes(y = (..count..)/sum(..count..)), binwidth = 5000, fill = "blue")
p <- p + ylab("proportion") +  xlim(0, 330000)
p <- p + geom_vline(xintercept = mean(df_salary$amount))
p
```

Estimating the average

(1,000 random samples of size 40)

Sample mean = $55,610

(Population mean = $55,534)



QQ Plot

Code to generate this plot

```
library(ggplot2)
salary_csv <- read.csv('FGCU_salary_dataset.csv', header = TRUE)
str_sal <- salary_csv$Annual.Salary
df_salary <- data.frame(amount =  as.numeric(substr(str_sal, 2, nchar(str_sal))))
set.seed(456) ; sample_size <- 40 ;num_trials <- 1000
vct_samples <- replicate(num_trials,
       mean(sample(df_salary$amount, sample_size, replace = FALSE)))
df_sample <- data.frame(amount = vct_samples)
n <- ggplot(df_sample,aes(x = amount))
n <- n + geom_bar(aes(y = (..count..)/sum(..count..)), binwidth = 500, fill = "green")
n <- n + ylab("proportion")
n <- n + geom_vline(xintercept = mean(df_sample$amount))
n
qqnorm(df_sample$amount)
qqline(df_sample$amount, col = 2)
```

Standard error of the sample mean

(Assuming the population standard deviation is known)

Standard Error [SE] ($4,602)

Population standard deviation ($29,107)

$$\sigma_{\bar{x}} = \frac{\sigma}{\sqrt{N}}$$

Sample size (40)

Standard Error << Population standard deviation

Simulated SE = $4,643

Code to simulate SE

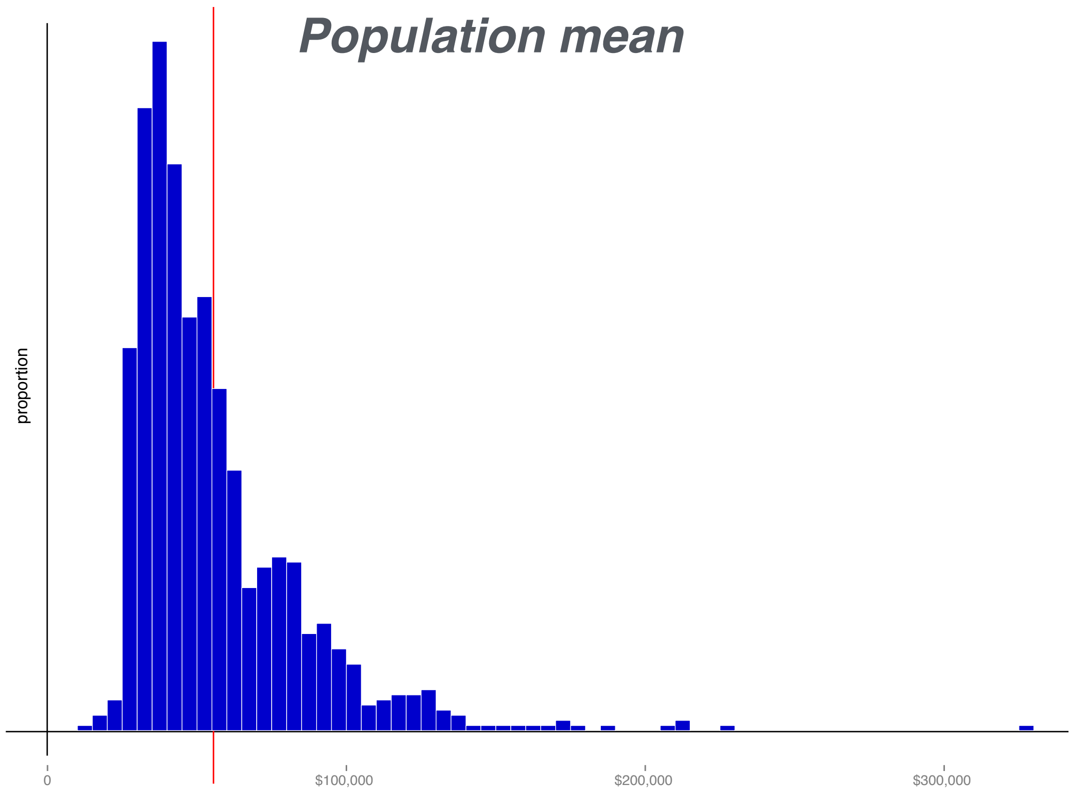
```
options(stringsAsFactors = FALSE)
salary_csv <- read.csv('FGCU_salary_dataset.csv', header = TRUE)
str_sal <- salary_csv$Annual.Salary
df_salary <- data.frame(amount =  as.numeric(substr(str_sal, 2, nchar(str_sal))))
set.seed(456) ; sample_size <- 40 ;num_trials <- 1000
vct_samples <- replicate(num_trials, mean(sample(df_salary$amount, sample_size, replace = FALSE)))
df_sample <- data.frame(amount = vct_samples)
# theoretical standard deviation of sample mean (assume population sd is known)
sd(df_salary$amount) / sqrt(sample_size)
# act   ual standard deviation of sample mean (based on simulation)
sd(df_sample$amount)
```

$\sigma_{\bar{x}} = \frac{\sigma}{\sqrt{N}}$ ***Proof (see "variance of weighted sums")***

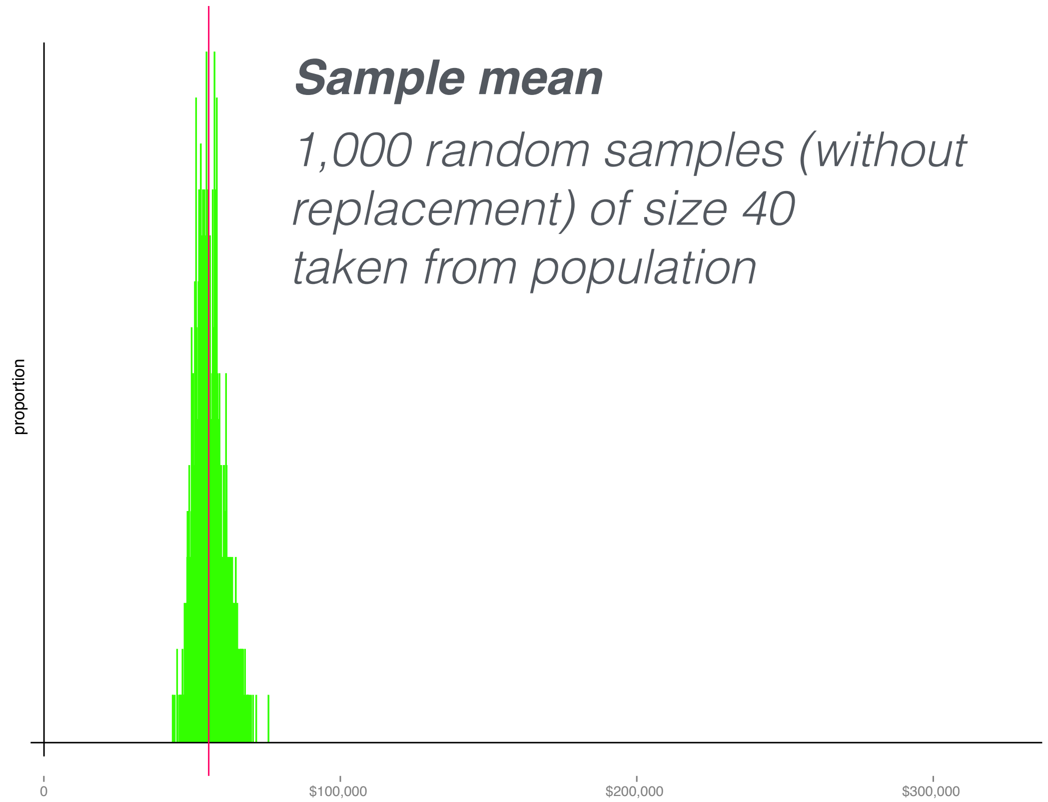
http://www.seas.upenn.edu/~ese302/lectures/Lecture_3/Lecture_3.pdf

Distributions compared

Distribution of the population mean compared to 1000 random samples (n = 40)



Population mean



Sample mean

1,000 random samples (without replacement) of size 40 taken from population

Sample mean characteristics

Shape - ***Normally distributed (from CLT)***

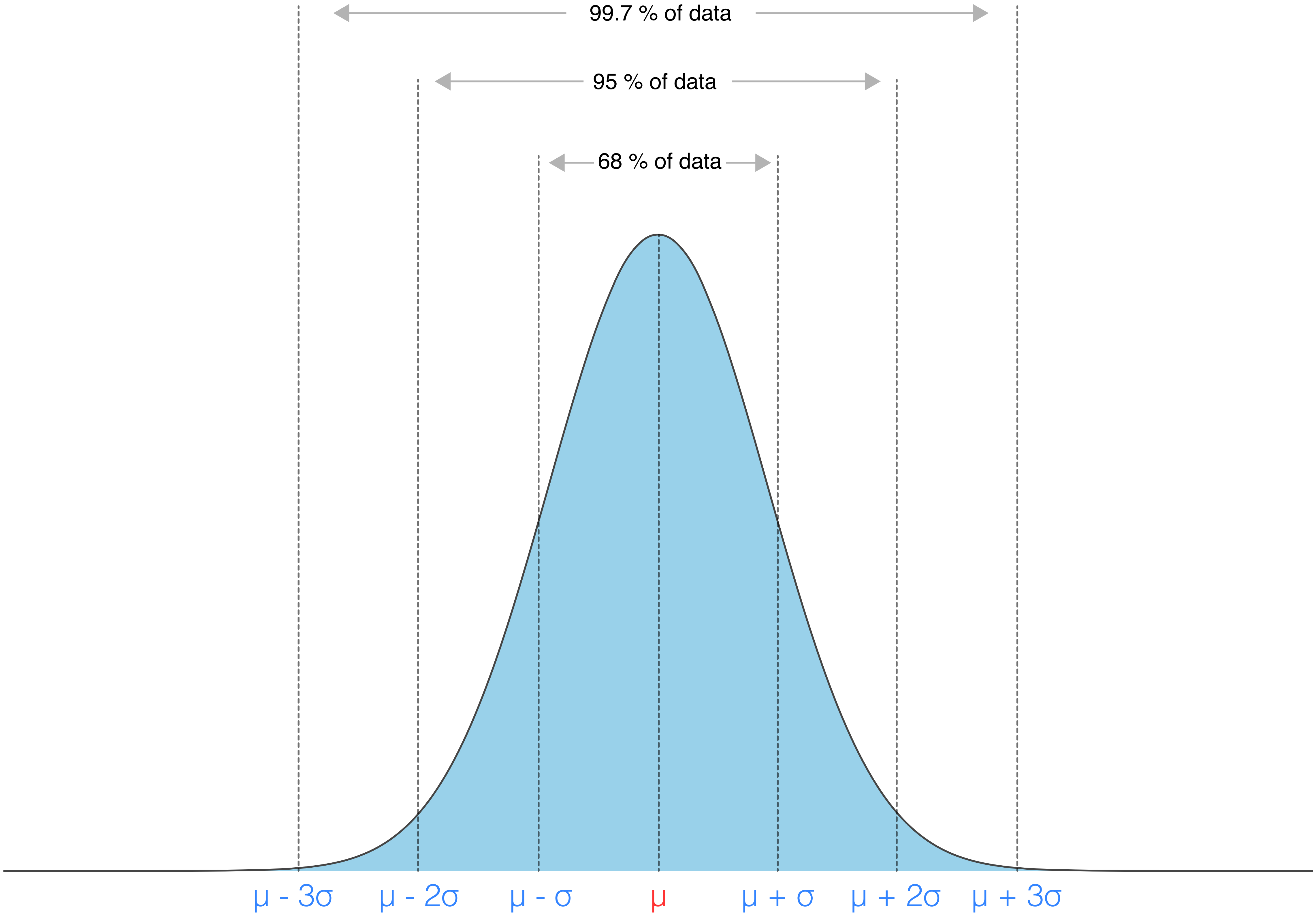
Location - ***Centred around population mean***

For proof see "unbiased estimator of population mean"

see: http://eml.berkeley.edu/~hildreth/e140_sp02/Lect6.ppt

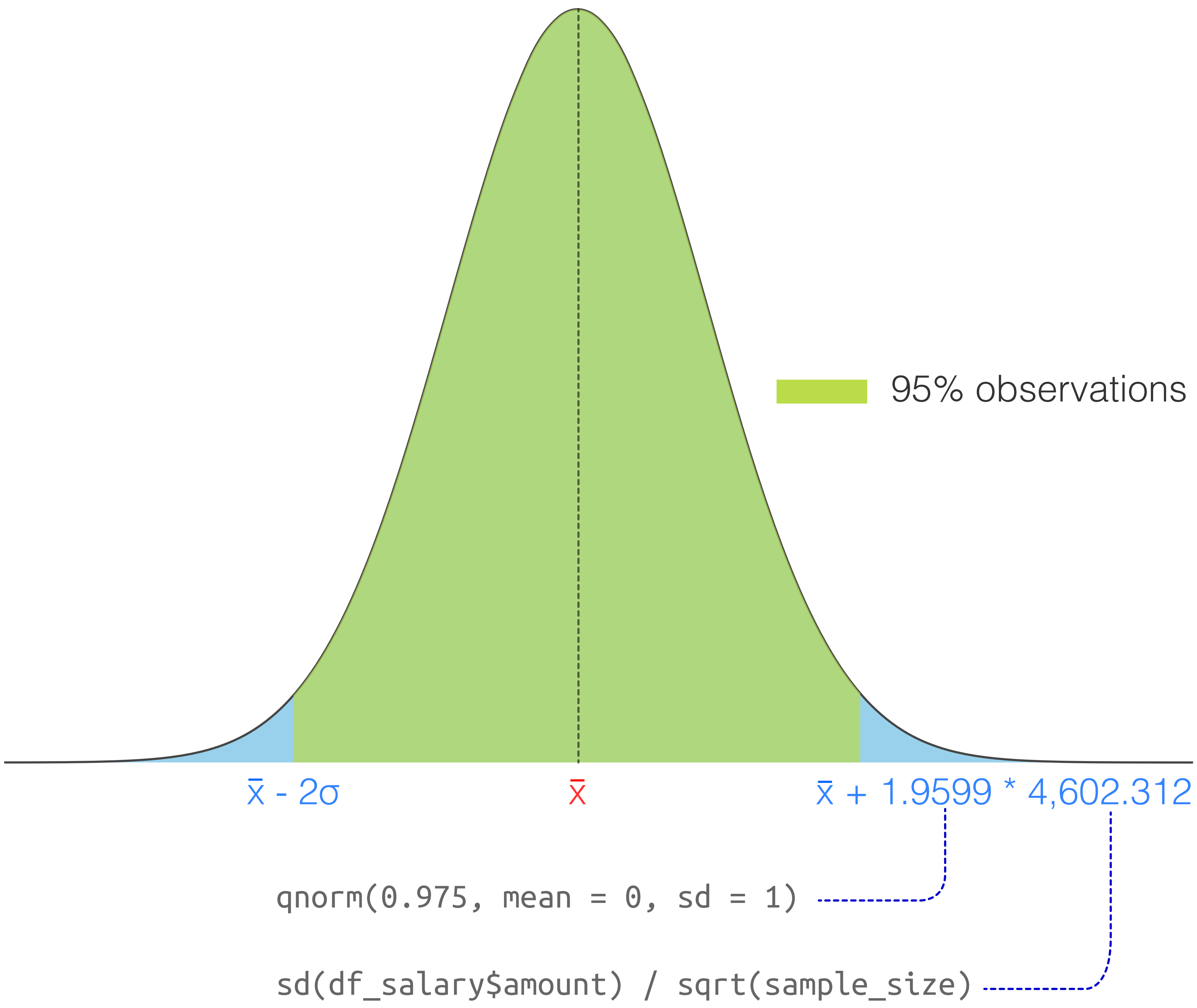
Dispersion - $\sigma_{\bar{x}} = \dfrac{\sigma}{\sqrt{N}}$

Standard normal distribution *(revisited)*



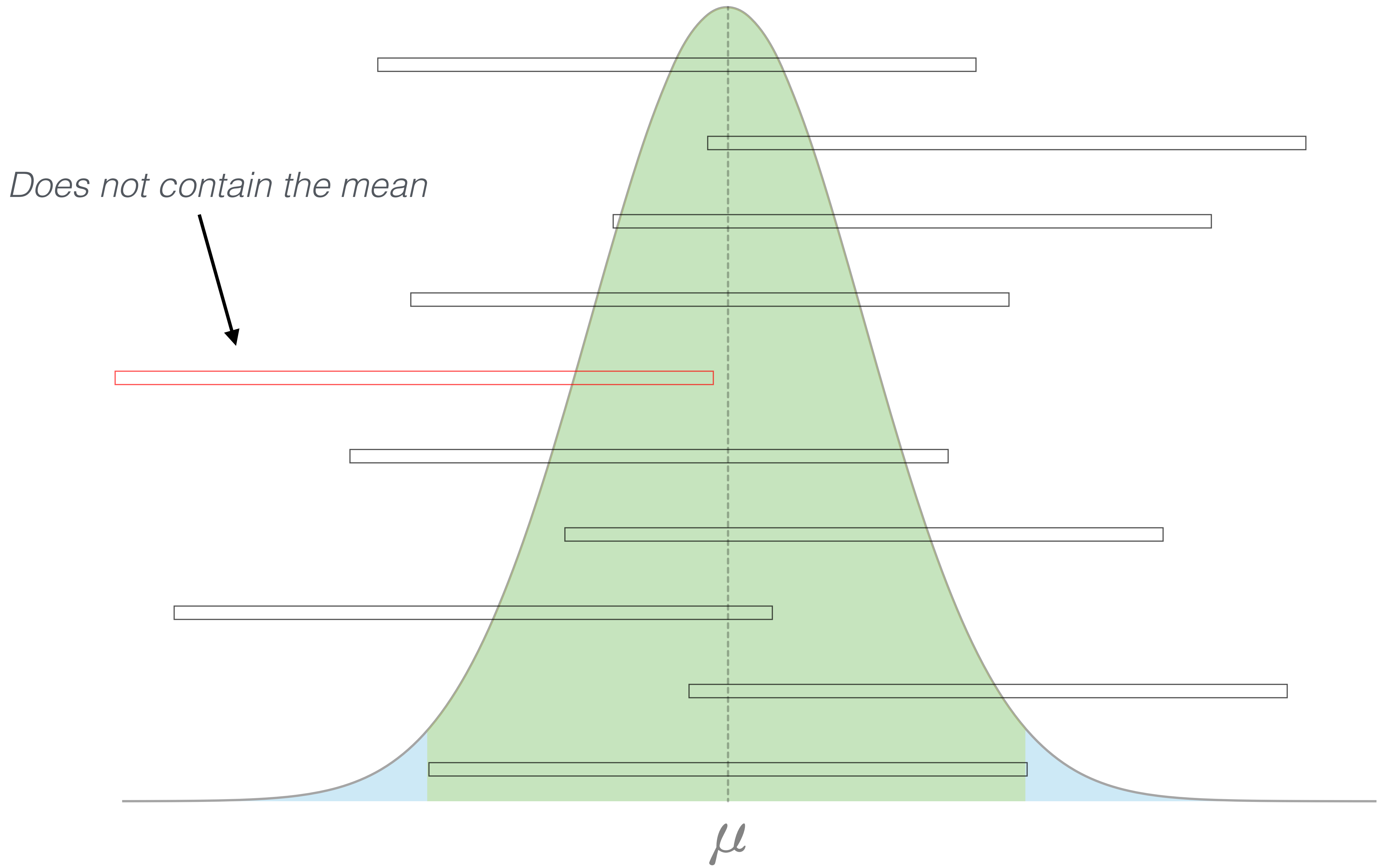
Constructing a confidence interval

*Width of interval = 2 x 1.96 * 4602 = 18,040*



Constructing a confidence interval

95 % of samples should contain the population mean



Simulating confidence intervals

Confirmation through simulation:

- *Take 1000 samples from academic salaries datasets of size 40.*
- *Construct a 95% confidences interval for each sample.*
- *Investigate the proportion of samples which contains the population mean.*

Code for this simulation

```
rm(list = ls())
#read in raw data
salary_csv <- read.csv('FGCU_salary_dataset.csv', header = TRUE)
str_sal <- salary_csv$Annual.Salary
# convert the character vector to numeric
df_salary <- data.frame(amount =  as.numeric(substr(str_sal, 2, nchar(str_sal))))
# set basic parameters for the simulation
set.seed(457)
sample_size <- 40
num_trials <- 1000
# take a sample of size 40 from population mean. Do this 1000 times. Calculate the mean
vct_samples <- replicate(num_trials, mean(sample(df_salary$amount, sample_size, replace = FALSE)))
df_sample <- data.frame(sample_mean = vct_samples)
# calculate the population mean
pop_mean <- mean(df_salary$amount)
# calculate number of standard deviations for 95% confidence interval (two tailed)
no_sd <- qnorm(0.975, mean = 0, sd = 1)
# calculate the standard error for the sample (assume we know the population sd)
sample_se <- sd(df_salary$amount) / sqrt(sample_size)
# calculate width of the confidence interval
deviation <- no_sd * sample_se
# create upper and lower bounds for each sample_mean (vectorised operation)
df_sample$l_bound <- df_sample$sample_mean - deviation
df_sample$u_bound <- df_sample$sample_mean + deviation
# calculate boolean based on whether the confidence interval contains the pop. mean
df_sample$contains_mu <- (pop_mean >= df_sample$l_bound) & (pop_mean <= df_sample$u_bound)
# proportion correct
(prop_correct <- sum(df_sample$contains_mu == TRUE) /   nrow(df_sample))
```