551 Assignment 2

Matthew Stoebe

2024-09-14

Homework Questions:

##Chapter 4: ###4.10

Survey weighting: Compare two options for a national opinion survey: (a) a simple random sample of 1000 Americans, or (b) a survey that oversamples Latinos, with 300 randomly sampled Latinos and 700 others randomly sampled from the non-Latino population. One of these options will give more accurate comparisons between Latinos and others; the other will give more accurate estimates for the total population average.

(a) Which option gives more accurate comparisons and which option gives more accurate population estimates?

(b) Explain your answer above by computing standard errors for the Latino/other comparison and the national average under each design. Assume that the national population is 15% Latino, that the items of interest are yes/no questions with approximately equal proportions of each response, and (unrealistically) that the surveys have no problems with nonresponse.

####Code

```
#parameters
p_yes <- 0.5
n_total <- 1000

# Option (a) - Simple Random Sample
n_latino_a <- 150
n_nonlatino_a <- 850

# Option (b) - Oversampling Latinos
n_latino_b <- 300
n_nonlatino_b <- 700

# Function to calculate the standard error for the comparison between two
groups
standard_error_comparison <- function(p1, p2, n1, n2) {
  sqrt((p1 * (1 - p1)) / n1 + (p2 * (1 - p2)) / n2)
}

# Function to calculate the standard error for the total population
```

```r
standard_error_total <- function(p, n) {
  sqrt((p * (1 - p)) / n)
}

# Standard errors for Latino vs Non-Latino comparison in both options
se_comparison_a <- standard_error_comparison(p1 = p_yes, p2 = p_yes, n1 =
n_latino_a, n2 = n_nonlatino_a)
se_comparison_b <- standard_error_comparison(p1 = p_yes, p2 = p_yes, n1 =
n_latino_b, n2 = n_nonlatino_b)

# Standard errors for the total population in both options (same for both
since n_total = 1000)
se_total_a <- standard_error_total(p = p_yes, n = n_total)
se_total_b <- standard_error_total(p = p_yes, n = n_total)

cat("Option (a) - Simple Random Sample:\n")

## Option (a) - Simple Random Sample:

cat("SE (Latino vs Non-Latino comparison):", round(se_comparison_a, 4), "\n")

## SE (Latino vs Non-Latino comparison): 0.0443

cat("SE (Total Population):", round(se_total_a, 4), "\n\n")

## SE (Total Population): 0.0158

cat("Option (b) - Oversampling Latinos:\n")

## Option (b) - Oversampling Latinos:

cat("SE (Latino vs Non-Latino comparison):", round(se_comparison_b, 4), "\n")

## SE (Latino vs Non-Latino comparison): 0.0345

cat("SE (Total Population):", round(se_total_b, 4), "\n")

## SE (Total Population): 0.0158
```

####Answer For the national opinion survey, Option (a) (simple random sample of 1000 Americans) provides a more accurate estimate for the total population average with a standard error of 0.0158. However, Option (b) (oversampling Latinos with 300 Latinos and 700 non-Latinos) offers more accurate comparisons between Latinos and non-Latinos, with a smaller standard error of 0.034 compared to 0.043 in Option (a).

The difference arises because Option (b) increases the Latino sample size, reducing the standard error for group comparisons. However, for overall population estimates, both options have the same total sample size, so their standard errors are equal for that purpose.

##Chapter 5: ###5.2,

Continuous probability simulation: The logarithms of weights (in pounds) of men in the United States are approximately normally distributed with mean 5.13 and standard deviation 0.17; women's log weights are approximately normally distributed with mean 4.96 and standard deviation 0.20. Suppose 10 adults selected at random step on an elevator with a capacity of 1750 pounds. What is the probability that their total weight exceeds this limit?

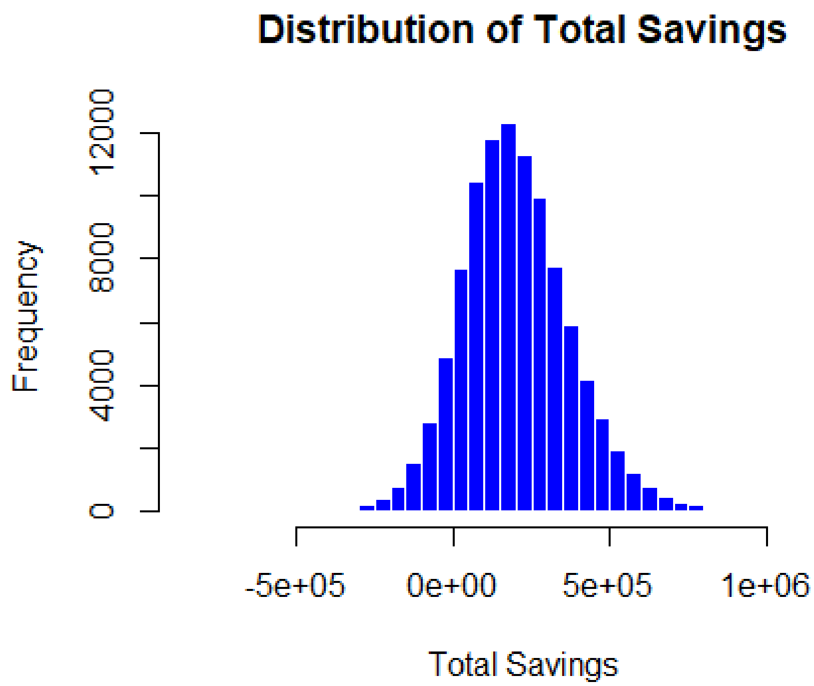
####Code

```
mean_log_men <- 5.13
sd_log_men <- .17
mean_log_women <- 4.96
sd_log_women <- .20

weight_limit <- 1750

n_individuals <-10

# Proportion of men and women (assume 50/50 split)
prop_men <- 0.5
n_men <- round(n_individuals * prop_men)
n_women <- n_individuals - n_men

n_sim <- 10000

exceeds_limit <- replicate(n_sim, {

  #rlnorm to avoid having to transform from log
  men_weights <- rlnorm(n_men, meanlog = mean_log_men, sdlog = sd_log_men)
  women_weights <- rlnorm(n_women, meanlog = mean_log_women, sdlog =
sd_log_women)

  # Calculate the total weight
  total_weight <- sum(men_weights) + sum(women_weights)

  # Check if the total weight exceeds the elevator limit
  return(total_weight > weight_limit)
})

prob_exceed <- mean(exceeds_limit)

# Print the result
cat("Estimated probability that the total weight exceeds the elevator's
capacity:", prob_exceed, "\n")

## Estimated probability that the total weight exceeds the elevator's
capacity: 0.0428
```

###5.6, Propagation of uncertainty: We use a highly idealized setting to illustrate the use of simulations in combining uncertainties. Suppose a company changes its technology for widget production, and a study estimates the cost savings at $5 per unit, but with a standard error of $4. Furthermore, a forecast estimates the size of the market (that is, the number of widgets that will be sold) at 40 000, with a standard error of 10 000. Assuming these two sources of uncertainty are independent, use simulation to estimate the total amount of money saved by the new product (that is, savings per unit, multiplied by size of the market).

####Code

```
mean_savings_per_unit <- 5
sd_savings_per_unit <- 4
mean_market_size <- 40000
sd_market_size <- 10000

# Number of simulations
n_sim <- 100000

total_savings <- replicate(n_sim, {

  # Simulate savings per unit and market size
  simulated_savings_per_unit <- rnorm(1, mean = mean_savings_per_unit, sd =
sd_savings_per_unit)
  simulated_market_size <- rnorm(1, mean = mean_market_size, sd =
sd_market_size)

  # Calculate total savings
  simulated_savings_per_unit * simulated_market_size
})

# Summary of the results
cat("Mean total savings:", mean(total_savings), "\n")

## Mean total savings: 199414.6

cat("Standard deviation of total savings:", sd(total_savings), "\n")

## Standard deviation of total savings: 172026

# Plot the distribution of total savings
hist(total_savings, breaks = 50, main = "Distribution of Total Savings",
     xlab = "Total Savings", col = "blue", border = "white")
```

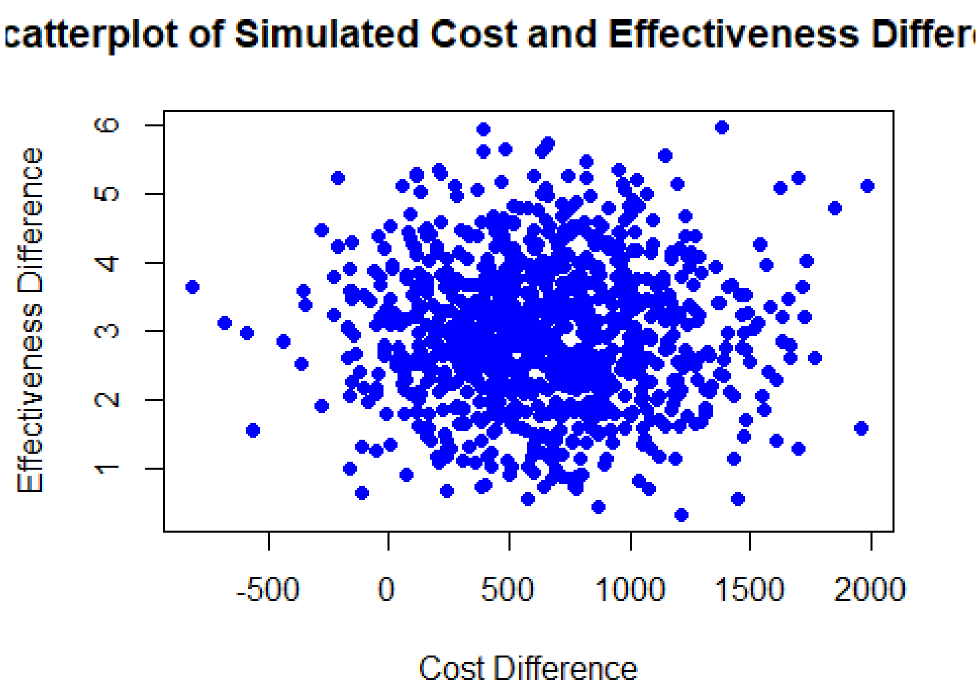


###5.10 Inference for a ratio of parameters: A (hypothetical) study compares the costs and effectiveness of two different medical treatments. • In the first part of the study, the difference in costs between treatments A and B is estimated at $600 per patient, with a standard error of $400, based on a regression with 50 degrees of freedom. • In the second part of the study, the difference in effectiveness is estimated at 3.0 (on some relevant measure), with a standard error of 1.0, based on a regression with 100 degrees of freedom. • For simplicity, assume that the data from the two parts of the study were collected independently. Inference is desired for the incremental cost-effectiveness ratio: the difference between the average costs of the two treatments, divided by the difference between their average effectiveness, a problem discussed further by Heitjan, Moskowitz, and Whang (1999). (a) Create 1000 simulation draws of the cost difference and the effectiveness difference, and make a scatterplot of these draws. (b) Use simulation to come up with an estimate, 50% interval, and 95% interval for the incremental cost-effectiveness ratio. (c) Repeat, changing the standard error on the difference in effectiveness to 2.0.

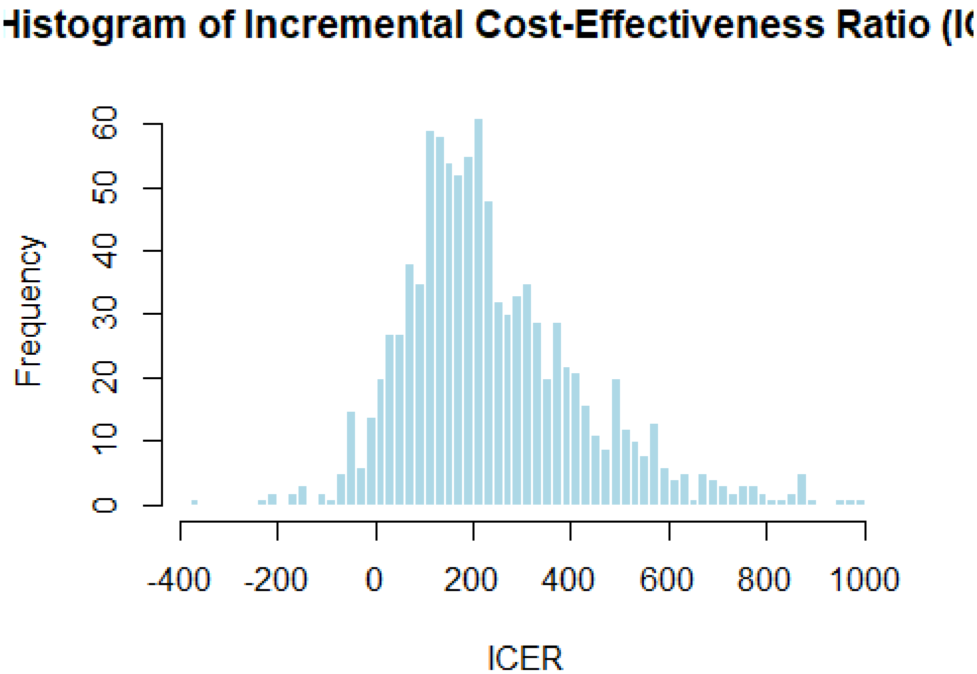
####Code a

```
# Parameters
mean_cost_diff <- 600
se_cost_diff <- 400
df_cost <- 50  # degrees of freedom for cost difference

mean_effect_diff <- 3.0
se_effect_diff <- 1.0
df_effect <- 100  # degrees of freedom for effectiveness difference
```

```r
n_sim <- 1000  # Number of simulations

# Simulate cost differences from a t-distribution
cost_diff_sim <- mean_cost_diff + rt(n_sim, df = df_cost) * se_cost_diff

# Simulate effectiveness differences from a t-distribution
effect_diff_sim <- mean_effect_diff + rt(n_sim, df = df_effect) *
se_effect_diff

# (a) Scatterplot of cost differences vs. effectiveness differences
plot(cost_diff_sim, effect_diff_sim,
     xlab = "Cost Difference",
     ylab = "Effectiveness Difference",
     main = "Scatterplot of Simulated Cost and Effectiveness Differences",
     col = "blue", pch = 16)
```




```r
icer_sim <- cost_diff_sim / effect_diff_sim
icer_filtered <- icer_sim[abs(icer_sim) < 1000]


hist(icer_filtered,
     breaks = 50,
     main = "Histogram of Incremental Cost-Effectiveness Ratio (ICER)",
     xlab = "ICER",
     col = "lightblue",
     border = "white")
```



####Code b

```
# Calculate the ICER from the simulated cost and effectiveness differences
icer_sim <- cost_diff_sim / effect_diff_sim

# Calculate the point estimate (mean) of ICER
mean_icer <- mean(icer_sim)

# Calculate the 50% confidence interval
ci_50_lower <- quantile(icer_sim, 0.25)
ci_50_upper <- quantile(icer_sim, 0.75)

# Calculate the 95% confidence interval
ci_95_lower <- quantile(icer_sim, 0.025)
ci_95_upper <- quantile(icer_sim, 0.975)

# Print results
cat("Estimated median ICER:", mean_icer, "\n")

## Estimated median ICER: 256.8202

cat("50% confidence interval for ICER:", ci_50_lower, "to", ci_50_upper,
"\n")

## 50% confidence interval for ICER: 115.7942 to 342.7522

cat("95% confidence interval for ICER:", ci_95_lower, "to", ci_95_upper,
"\n")
```

```
## 95% confidence interval for ICER: -45.93554 to 812.4017
```

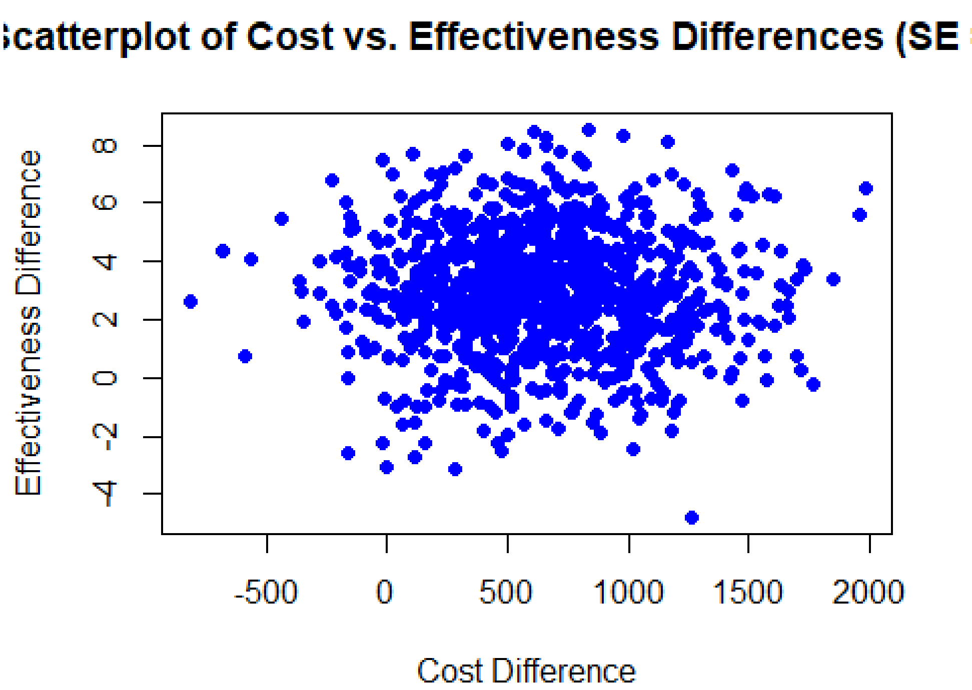
####Code c

```
# Parameters
mean_cost_diff <- 600
se_cost_diff <- 400
df_cost <- 50  # degrees of freedom for cost difference

mean_effect_diff <- 3.0
df_effect <- 100  # degrees of freedom for effectiveness difference

n_sim <- 1000

se_effect_diff <- 2.0

# Simulate new effectiveness differences
effect_diff_sim <- mean_effect_diff + rt(n_sim, df = df_effect) *
se_effect_diff

# Calculate the new ICER
icer_sim <- cost_diff_sim / effect_diff_sim

# Remove infinite and NaN values
icer_sim <- icer_sim[is.finite(icer_sim)]

# Calculate the median ICER
mean_icer <- mean(icer_sim)

# Calculate the 50% confidence interval
ci_50_lower <- quantile(icer_sim, 0.25)
ci_50_upper <- quantile(icer_sim, 0.75)

# Calculate the 95% confidence interval
ci_95_lower <- quantile(icer_sim, 0.025)
ci_95_upper <- quantile(icer_sim, 0.975)

# Print results
cat("With SE of effectiveness difference = 2.0\n")

## With SE of effectiveness difference = 2.0

cat("Estimated median ICER:", mean_icer, "\n")

## Estimated median ICER: 171.9942

cat("50% confidence interval for ICER:", ci_50_lower, "to", ci_50_upper,
"\n")

## 50% confidence interval for ICER: 80.80463 to 334.5234
```

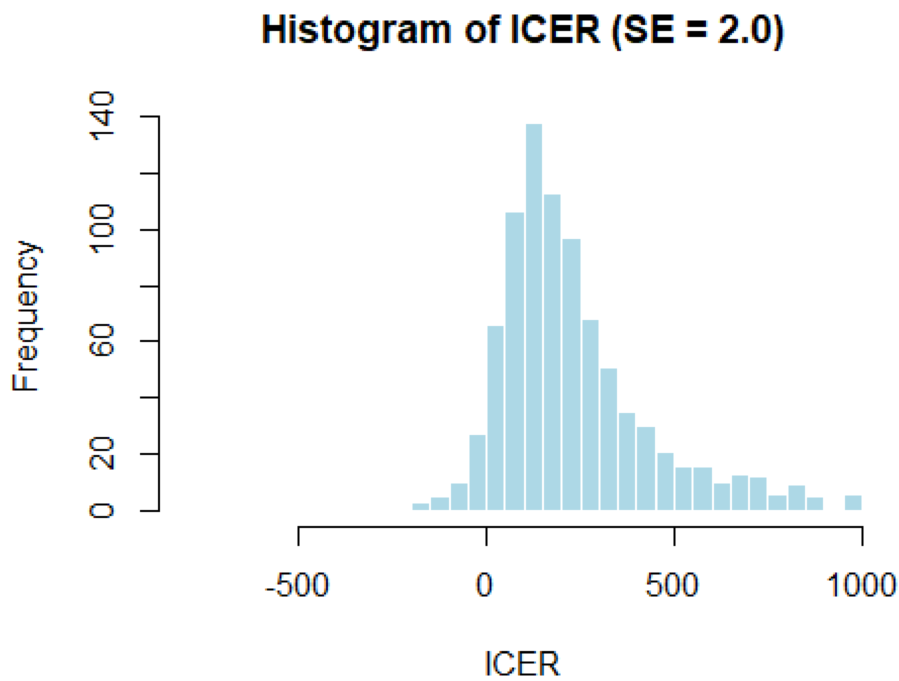
```
cat("95% confidence interval for ICER:", ci_95_lower, "to", ci_95_upper,
"\n")
```

```
## 95% confidence interval for ICER: -1371.823 to 2050.814
```

```
# Remove extreme ICER values for plotting
icer_filtered_2 <- icer_sim[effect_diff_sim > 0 & abs(icer_sim) < 1000]

# (1) Scatterplot of cost differences vs. effectiveness differences with SE =
2.0
plot(cost_diff_sim, effect_diff_sim,
     xlab = "Cost Difference",
     ylab = "Effectiveness Difference",
     main = "Scatterplot of Cost vs. Effectiveness Differences (SE = 2.0)",
     col = "blue", pch = 16)
```



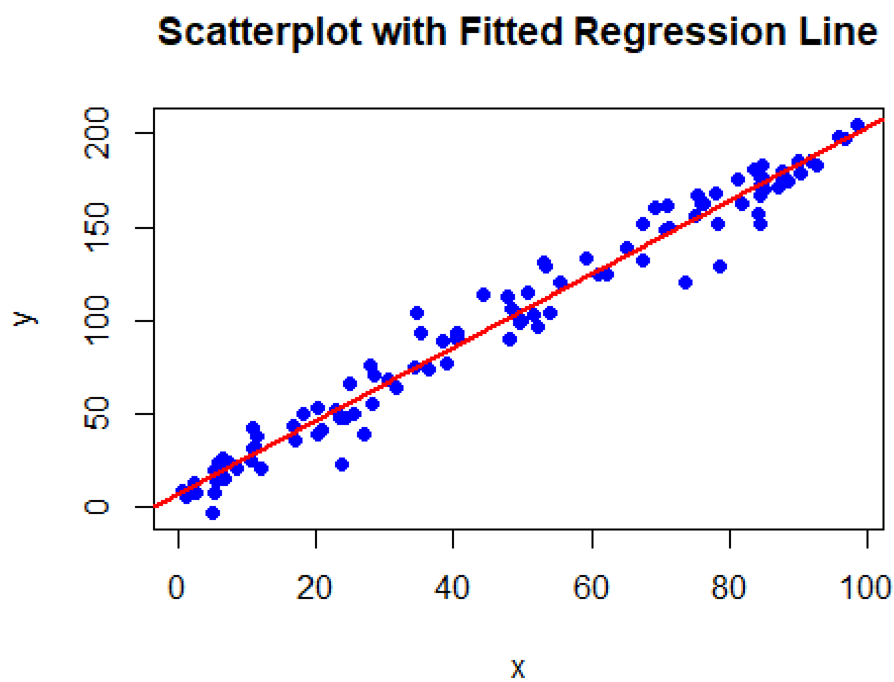

```
# (2) Histogram of the new ICER with SE = 2.0
hist(icer_filtered_2,
     breaks = 50,
     main = "Histogram of ICER (SE = 2.0)",
     xlab = "ICER",
     col = "lightblue",
     border = "white")
```



##Chapter 6: ###6.2, Programming fake-data simulation: Write an R function to: (i) simulate n data points from the model, y = a + bx + error, with data points x uniformly sampled from the range (0, 100) and with errors drawn independently from the normal distribution with mean 0 and standard deviation σ; (ii) fit a linear regression to the simulated data; and (iii) make a scatterplot of the data and fitted regression line. Your function should take as arguments, a, b, n, σ, and it should return the data, print out the fitted regression, and make the plot. Check your function by trying it out on some values of a, b, n, σ.

```
simulate_regression <- function(a,b,n,sigma){
  x <- runif(n, min=0, max=100)

  error <- rnorm(n, mean=0,  sd = sigma)

  y <- a + b*x + error

  model <- lm(y~x)

  plot(x, y, main = "Scatterplot with Fitted Regression Line",
       xlab = "x", ylab = "y", pch = 19, col = "blue")
  abline(model, col = "red", lwd = 2)

  # Return the data as a data frame
  data <- data.frame(x = x, y = y)

   return(data)
```

```
}


simulate_regression(5,2,100,10)
```

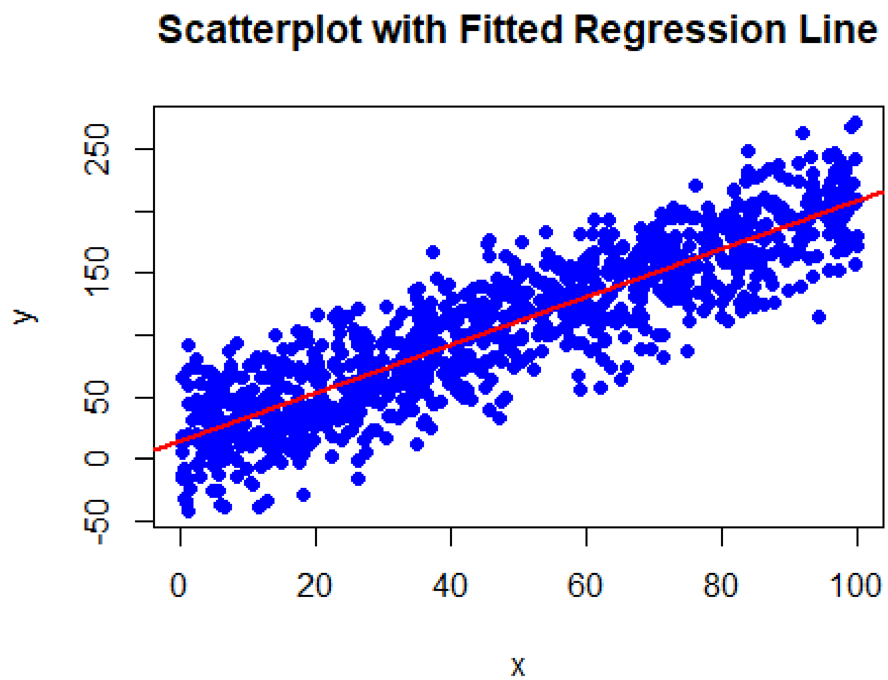



```
##              x          y
## 1   87.6638635 174.690961
## 2   20.1443563  52.967798
## 3   69.2480923 160.156952
## 4   90.4846675 178.411337
## 5    2.4925280   7.441136
## 6   16.6073247  43.265350
## 7   47.8982531 112.683921
## 8   52.2726289  96.801953
## 9   40.4332587  89.712434
## 10  55.3855550 120.064163
## 11  78.7219833 129.298830
## 12  34.3718708  74.466278
## 13  67.3235282 132.382669
## 14  84.4486856 151.238250
## 15  48.0569089  90.350453
## 16  53.2608704 129.002919
## 17  39.0313057  76.634471
## 18  98.5029128 204.842459
## 19  84.5489887 166.981086
## 20  90.0854797 184.783798
## 21  34.5318846 104.411167
```

```
## 22  10.7311713  31.487708
## 23   5.8199415  23.764133
## 24  50.7878472 115.099396
## 25  49.4381195  98.474599
## 26  48.9687208 103.884590
## 27   5.5990304  13.896102
## 28  35.1962842  93.346805
## 29  87.0667459 170.719555
## 30  96.8320978 197.046261
## 31  23.8028328  22.915742
## 32  24.9974678  66.739481
## 33  71.1776221 149.257076
## 34  84.1802024 156.986075
## 35  78.0538880 167.737557
## 36  61.1457949 125.131300
## 37   6.8624081  15.675316
## 38  67.3556509 152.102541
## 39  27.7762012  76.018110
## 40  31.7638177  64.249560
## 41  20.1923977  38.800498
## 42  10.7469929  42.861920
## 43  76.3294200 162.467875
## 44  40.4486058  93.158736
## 45   0.4389781   8.892393
## 46  48.4969248 106.474417
## 47  78.3006596 152.046517
## 48  51.7158793 103.320642
## 49  28.5688319  70.534769
## 50  75.9670639 162.989582
## 51  75.1686264 156.404080
## 52  30.5866049  68.487382
## 53  62.0691937 124.941027
## 54  81.1126874 175.018991
## 55  10.4005228  24.732201
## 56  22.8032690  51.691757
## 57  81.7214589 163.075553
## 58   5.3291223  19.412409
## 59  88.5767504 174.881095
## 60  12.0153220  21.301694
## 61  73.5503780 120.100384
## 62  92.6994721 183.621208
## 63  84.5335384 176.616036
## 64  85.0915627 169.881100
## 65  16.9167479  35.842317
## 66  65.0033069 138.708282
## 67  91.8442832 184.903054
## 68  87.8059090 179.491907
## 69  23.5523754  48.248972
## 70  25.5871407  49.655414
## 71  75.4817728 167.050685
```

```
## 72   8.5835797  21.248293
## 73   5.0507031  -2.899296
## 74  38.4863961  88.710223
## 75   2.4561471  12.883572
## 76  70.7552373 147.987419
## 77  49.7146000  99.810330
## 78  83.5539642 180.869054
## 79  11.3960305  38.254194
## 80   7.3020403  24.018850
## 81  24.3269308  48.146355
## 82  53.8475807 103.801325
## 83   1.1018149   5.681589
## 84  36.4478843  73.825168
## 85   5.3523570   7.820391
## 86  85.1415181 175.002749
## 87  62.1354025 124.774908
## 88  53.2200175 130.870871
## 89   6.3480182  26.526221
## 90  27.0337725  39.707438
## 91  96.0515233 197.960151
## 92  18.1428564  50.047295
## 93  44.2986624 113.862459
## 94  28.0830321  55.580659
## 95  20.7201846  41.825758
## 96  71.0905698 161.562836
## 97  59.3471495 133.228367
## 98  10.9956242  33.029715
## 99  84.4040118 172.810994
## 100 84.6550289 183.148475
```

```r
simulate_regression(10,2,1000,30)
```




```
##               x          y
## 1    90.9142719 165.3290270
## 2    23.3603904  92.7738792
## 3    27.5927717  96.2455429
## 4    12.3331129  17.6423327
## 5    25.6946112 104.3724968
## 6    61.2180729 103.5374135
## 7    37.4908168  45.2595808
## 8    26.4995667  38.7408755
## 9    49.9593372 147.0406423
## 10   30.3768162  96.1409975
## 11   82.3236343 168.6040776
## 12   80.0870967 162.7242740
## 13   12.0358163  66.2386393
## 14   48.4805087 131.8464224
## 15    4.5892377   1.9636677
## 16   10.8807146  74.8786375
## 17   18.9391501  29.6114736
## 18   59.4281991  93.3099439
## 19   77.7364845 189.0782480
## 20   83.1579168 163.2344327
## 21   99.7487309 206.5652365
## 22   93.3982028 230.1385896
## 23    1.8315077  32.2201446
## 24    3.8452105  12.9618553
## 25   25.3781096  23.6741805
## 26   58.9750209 121.0629387
```

```
## 27   36.8582692  80.4663849
## 28   67.0633893 111.8961431
## 29    5.9535385  46.3700904
## 30   98.2779667 234.6121526
## 31   71.0940234 165.3844810
## 32   57.0337214 105.6673612
## 33   24.0895249  57.0135118
## 34   36.5065560  67.0024558
## 35   36.8787169 125.5659108
## 36   93.2520222 147.3040603
## 37   35.0719909  66.2007289
## 38   77.0903163 178.9982195
## 39   42.4548149 119.8479035
## 40   53.5608491 104.9574992
## 41   62.4447936  80.3673535
## 42    6.4082979 -37.9521145
## 43   19.2997699  32.2907493
## 44    5.6131059 -12.1668736
## 45   83.8659053 188.3886860
## 46   50.2194183 122.8488241
## 47   57.9638013 129.7599889
## 48   85.6737892 161.4320342
## 49   84.2174526 167.9831178
## 50   72.7707097 169.8883820
## 51   53.9364191 107.9262074
## 52   18.0036204  66.4623402
## 53    1.1024901 -15.5087066
## 54   41.0720412  93.2443432
## 55   73.6510143 144.4979778
## 56    9.1144827  53.3225028
## 57   44.2370986 113.0624851
## 58   88.7289158 194.5847254
## 59    4.6708763  45.3932365
## 60    3.1332486  34.1929335
## 61   26.6837470  47.8933205
## 62   71.8427120 194.5985750
## 63   16.0618566   7.8675034
## 64   72.7071137 202.9557088
## 65   89.6848599 225.5745457
## 66   55.1840713 102.5540137
## 67   60.9125562 159.4612090
## 68   82.1150463 165.1720554
## 69   71.1193449 180.5951024
## 70   10.4015987  38.9067922
## 71   87.2447631 204.2303512
## 72   62.8322873 170.9168905
## 73   59.1032613 181.9966111
## 74   68.1594711 128.3209067
## 75   43.1036081 136.8566288
## 76    7.9907090  67.5234259
```

```
## 77    2.7804315  13.2587333
## 78   41.7260289  99.7489114
## 79    7.0943544  87.2344180
## 80   13.1904348  34.1678731
## 81   35.3997305  90.3221227
## 82   47.8086316 134.9050458
## 83    1.1201247  19.0000964
## 84   48.0538507  90.3599003
## 85   17.3234253   1.7335192
## 86   23.3595716  31.1163997
## 87   34.7469050 128.7429579
## 88    4.7022947  -4.3730925
## 89   34.1593026  78.6071728
## 90   86.7317689 149.0531890
## 91   11.3358844  38.8790703
## 92   67.4253569 147.2581292
## 93    1.2022525  72.9045436
## 94   99.0439282 267.0872244
## 95   70.1489215 165.8080344
## 96   44.5721338  98.2685724
## 97   47.7891468 114.6258289
## 98   34.3563213  39.6582306
## 99   87.6863601 126.6971268
## 100  70.3336882 154.5454456
## 101  74.0672176 151.2675000
## 102   2.1903386  80.7981523
## 103  52.8041887  99.5036118
## 104   1.3262452  16.9400679
## 105  83.5122869 224.3091938
## 106  30.3031960  16.7742808
## 107  47.5632717  78.5122329
## 108  93.1623432 242.6888851
## 109  50.4975497  96.7932794
## 110  48.3820639 139.7270186
## 111  32.1427533  60.1073730
## 112  22.6161896  77.2170987
## 113  82.0400759 166.6764383
## 114  53.6797503 141.5304811
## 115  20.6915830  48.6072284
## 116  93.1311624 202.6711177
## 117  20.1092682  44.8116851
## 118  26.1977573  -0.2053706
## 119  77.4554753 165.0435120
## 120   9.3776968  28.8677598
## 121  14.6200574   9.8833222
## 122   8.1357123   0.2796000
## 123  28.7432797  76.9642854
## 124  15.6539434  57.9159798
## 125  73.9972893 163.0759619
## 126  78.7273735 185.2763943
```

```
## 127  87.1422772 207.6401148
## 128  44.6467275 138.4598813
## 129  56.1328461 134.6816396
## 130  70.9897008 197.0786654
## 131   0.7242002 -33.9100827
## 132  52.9676149 104.2697541
## 133  32.6437240 118.4647720
## 134  35.6980886 101.7042249
## 135   1.2835850  91.3908688
## 136  61.1249883  88.4065327
## 137  47.3720935  46.9375520
## 138  86.5243537 189.9371278
## 139  68.7260706 166.4728771
## 140  25.2778677  32.5773071
## 141  40.1451665  65.1808827
## 142  40.3066965 116.1743474
## 143  63.0163043 113.7969690
## 144  95.7264161 245.7557371
## 145  37.0311831  25.4814896
## 146  83.8318184 247.7999995
## 147  99.9040182 172.2747085
## 148   0.5509224  -7.5367995
## 149  33.4281150  42.5079123
## 150   7.6693344  18.0906171
## 151  11.6293418  15.2326822
## 152  92.6094915 193.0503093
## 153  71.5677134 167.8862964
## 154  26.1335178  92.7081150
## 155  50.4011026  78.2579596
## 156  13.8595800 100.1160738
## 157  45.6776184 103.2607732
## 158  31.1537376  59.0555060
## 159  23.4931351 108.3645503
## 160  48.5107606 139.4098292
## 161  20.2101995  75.7066890
## 162  85.7135469 229.4981377
## 163  24.7381435  54.0814178
## 164  59.5512426 145.6643310
## 165  42.2753079  56.7150919
## 166  42.0810216 133.1217515
## 167  45.0131959  78.3661924
## 168  75.3595755 194.0306031
## 169   7.2089673  44.3040604
## 170  14.3587833  46.0697769
## 171   6.6925498   6.8592025
## 172  48.9341160 130.6797418
## 173   6.5060044  15.3019962
## 174  10.7562071 -20.7522661
## 175  81.6826682 215.4725632
## 176  41.5803753  98.0048187
```

```
## 177  99.7607421 242.6096953
## 178  67.4040895 153.5807139
## 179  40.9648128 100.1286267
## 180  40.8248190  80.3517300
## 181  86.7221774 179.2484988
## 182  13.1778121   8.1523161
## 183  74.2469431 137.8001233
## 184  39.6541815 141.0888680
## 185  56.9762538  91.2999020
## 186   4.1188885  40.6496432
## 187  25.1513410  60.4272644
## 188  79.0009442 170.3109471
## 189  45.0327033  58.0273746
## 190  26.8741508  56.0842736
## 191  37.3425047 167.2880807
## 192  82.3181462 157.4885516
## 193  36.7272826 124.5016313
## 194  13.7082225  78.7898057
## 195  20.4294078  41.5980023
## 196  82.6126670 172.1313855
## 197  72.2140973 162.6912025
## 198  22.4873598  41.9419676
## 199  57.7400250 137.8017006
## 200  60.9161383 181.8394286
## 201  29.7878817  93.2642439
## 202  98.2993573 168.0406151
## 203   4.8775949  63.4172120
## 204  87.7015647 208.1794739
## 205   3.0163116  18.1231891
## 206   3.4119856  15.7007445
## 207  63.8674847 181.0455805
## 208  91.8668903 262.2647742
## 209  17.2580575  32.7332837
## 210  14.2293817  59.3840798
## 211  93.4590801 180.9213477
## 212  69.5042537  89.0441460
## 213  30.6104728  51.3541630
## 214  74.8152299 150.1309673
## 215  34.3864353  98.1263722
## 216   9.7571146  50.0856039
## 217  47.8708865 142.9012246
## 218  18.5977722 101.5580440
## 219  85.0791542 227.8574148
## 220  98.0182801 195.0629836
## 221  82.2554442 143.2169998
## 222   5.6250182  29.8633436
## 223  49.2558241  85.2858149
## 224  40.7462799  71.5030402
## 225  63.9883366 166.8531084
## 226  83.1723609 162.5335904
```

```
## 227  19.8070282  71.3257234
## 228   5.5397913  33.6787646
## 229  23.4288443  61.2938952
## 230  11.6355984 -38.0182079
## 231  82.8884504 127.3498434
## 232  54.6785265  99.4578315
## 233  32.0881802  70.8981066
## 234  80.2152916 187.0946746
## 235   6.1758427  48.5766101
## 236  50.6207943 117.0219622
## 237  74.3156691 132.8858374
## 238  35.1096448  75.4382854
## 239  72.4940123 170.0693470
## 240  74.9034387 143.1168761
## 241  58.5444879 132.9212628
## 242  19.9048974  77.7423126
## 243  67.4121615 150.9910816
## 244  93.1087474 202.3006507
## 245  50.7373148 114.9440423
## 246  40.1325095 128.7418297
## 247  12.8870178  -2.3367344
## 248   1.1593633  45.4246477
## 249   3.6542816  19.0158499
## 250  35.2157830 126.4525105
## 251  65.4864426 150.5609658
## 252  34.6901250  63.4923904
## 253  77.8428987 145.6114034
## 254  70.3875123 173.6028656
## 255  95.7738277 181.7529302
## 256  67.7761870 119.1877583
## 257  27.4660426  48.4162761
## 258  47.1225543 141.7795367
## 259  63.1714503 131.3848289
## 260  35.7617685 100.8494851
## 261  26.2572161 121.1590067
## 262  92.7938980 161.9411596
## 263  13.4716656  51.1994997
## 264   5.4424501  35.8899671
## 265  12.3582552  19.2136356
## 266  61.1000674 151.1391701
## 267  74.1506566 162.5777594
## 268  50.4669843 174.2841057
## 269  14.0794684  36.4441760
## 270  10.8223080  56.1186099
## 271  24.7632147  21.6946891
## 272  15.2722569  89.3862675
## 273  36.5038668  53.1595589
## 274  31.9280562  74.0437997
## 275  89.1239782 168.9995036
## 276  59.0172164 136.6373825
```

```
## 277  32.8446678  82.0662455
## 278  68.4395923 114.6962312
## 279   8.3348742  93.5106166
## 280  68.3003666 149.5741862
## 281  91.2334297 176.5335128
## 282  25.6829398  33.9792662
## 283  46.8181764 145.8430682
## 284  83.8520015 161.8318752
## 285   2.2902931  41.8400376
## 286   2.4593168  20.7394473
## 287  58.7474342  66.9141777
## 288  20.1463114  61.3068245
## 289  21.3416377  26.5538734
## 290  45.8400419  63.8271160
## 291  10.1011666  74.9705646
## 292   6.0110045 -36.0867829
## 293  91.0502591 189.7458689
## 294  55.2179818 136.8868322
## 295  17.7306058  93.9438134
## 296  24.1582736  57.3897361
## 297  27.7712232  90.2346998
## 298   0.1204326  11.7807200
## 299  95.7401602 206.2659393
## 300  79.0463247 189.6322699
## 301  13.9083863   2.3786588
## 302   6.3453940  56.1962465
## 303  26.3435857  79.5478001
## 304  85.6278328 155.9189267
## 305   8.3492960  28.1886168
## 306  29.4254799  44.6096542
## 307  66.0432489  73.6374990
## 308  99.3826781 221.4756550
## 309  58.0291759 132.0101740
## 310  84.9318803 147.5198844
## 311  41.9430673  78.0715789
## 312   6.7190948   9.5946589
## 313  41.0976476  87.5814394
## 314  51.1955530 121.2585649
## 315  20.9725454  50.1020522
## 316  50.5615299 134.8957767
## 317  15.1037359  79.5122821
## 318  20.4699360  32.1744686
## 319  69.5262098 143.3904536
## 320   8.9569322  47.3443262
## 321  82.4087401 153.4879079
## 322  63.4103809 158.3843674
## 323  32.3931156  94.0168303
## 324   2.8794749  61.0698629
## 325  49.0795882 138.4655071
## 326  30.3160225  55.6840595
```

```
## 327   5.5503635  17.3096167
## 328  49.3511290  74.2191035
## 329  27.5232696  66.1127822
## 330  75.7381669 159.7126641
## 331  58.9546602  55.4801826
## 332  33.3608568  67.5505741
## 333  84.6354291 127.8763113
## 334   5.5115628  54.1623047
## 335  92.5947435 205.5786681
## 336  12.7111266  46.4875414
## 337  78.0296456 145.2938515
## 338  61.0596565 112.2027409
## 339  29.8940869  61.3219221
## 340  65.0043480 122.0622518
## 341  72.6766245 174.0416267
## 342  45.1639872 109.0920978
## 343  80.8553335 111.9769089
## 344  77.5522277 179.3925347
## 345  82.8218658 157.1519144
## 346  92.0232928 195.2244319
## 347  46.4233839  98.4730131
## 348  52.1045374  72.9574457
## 349  36.5166292 109.8214283
## 350  60.3813865 135.0401797
## 351  44.6459333 118.5073471
## 352  49.3676399 121.2880679
## 353  19.6894695  26.2882559
## 354  72.0937594 161.4802994
## 355   5.3901593  12.8066263
## 356  35.9914321  30.8987986
## 357  19.6466391  65.6612355
## 358  71.2390813 167.1689086
## 359  34.9457145  87.8917367
## 360  64.8858013  64.7047602
## 361  87.3324445 164.9372895
## 362  42.6371725  97.0480624
## 363  93.2564940 229.6321131
## 364  58.7566081 134.3986355
## 365  27.6919977  20.3775979
## 366  46.3100204 116.6955133
## 367  32.6918629  67.3206924
## 368  32.9798277  65.2429094
## 369  60.4295284 165.9808140
## 370   1.0939997  60.1509337
## 371  83.2525997 160.9140086
## 372  94.0776192 163.1469280
## 373  11.1008016  44.8586606
## 374  88.1492830 200.4570974
## 375  58.5951477  93.0672632
## 376   8.3085159  -0.8492121
```

```
## 377  13.4636172  69.1415659
## 378  96.9659969 167.2996558
## 379  58.2593185 123.6611194
## 380  45.9574910 135.9047013
## 381   0.3744807  65.6774089
## 382  36.6469778 112.9731041
## 383  30.6708890 100.9279699
## 384  29.2975287  67.4398107
## 385  65.0003673 157.5939013
## 386  35.4350493 104.2018008
## 387  33.6843880  64.1470474
## 388  27.1649039  68.8007259
## 389  48.7757866 117.4991602
## 390  60.4710246  87.4505634
## 391  20.5506530  65.2815071
## 392  44.7245081 104.1495806
## 393  59.2590195 156.6680629
## 394  14.6832158  19.0354275
## 395  84.0808590 158.2927034
## 396  53.7390087 118.6343703
## 397  36.1534772  54.0665608
## 398  86.0425969 165.4409250
## 399  92.5366395 179.4869180
## 400  88.8543076 203.0228809
## 401   0.7006807 -34.6999174
## 402  61.0510871 144.7547878
## 403  44.7526681  93.9579028
## 404  58.3810846 117.7352813
## 405  85.4015984 161.7364197
## 406  18.2253197   3.4702393
## 407  16.7763980  31.7565265
## 408  81.1072777 135.8340984
## 409   2.6749561  -4.8719310
## 410  16.5073597  89.3745682
## 411  82.1373724 185.2389286
## 412  58.5233702 128.3473081
## 413  60.2454315 154.8292188
## 414  68.3338093 137.1986211
## 415  72.1549242 127.1380544
## 416  84.4181639 206.3164141
## 417   4.0715872  34.6625352
## 418  63.1183542 153.2572615
## 419  68.1329296 160.5923213
## 420  33.0815890  38.6166112
## 421  97.5172319 175.9631723
## 422  77.7566557 203.3684554
## 423  65.7248691  98.3587263
## 424  73.9140686 139.5898850
## 425  41.9463420  57.5778230
## 426   8.8066802  21.1961606
```

```
## 427   0.5308006 -15.8922278
## 428   0.2894681  15.5385870
## 429  59.9711221 150.3646659
## 430  65.2663524 116.1249357
## 431  33.0265691  89.6046855
## 432  38.2531261 107.4320611
## 433  97.7309596 203.1056937
## 434   6.5882246  50.5911646
## 435  97.9734976 225.6479892
## 436  78.1775442 186.2200732
## 437  83.8863987 200.4811556
## 438  76.4107829 134.9374519
## 439  12.5704763  82.6655514
## 440  94.3204483 115.0677105
## 441  34.8617588 103.1058445
## 442   9.1851830  66.5400763
## 443  76.2604622 155.9963782
## 444  36.9831395  75.5761706
## 445  48.1962034  95.9972206
## 446  46.9087287 129.3686762
## 447  10.2599956 -19.4478000
## 448  58.6139681 148.6247165
## 449   7.7885217  31.6239243
## 450  63.4560301 132.8111988
## 451  17.7041778  14.3958714
## 452  34.4943036 136.3848896
## 453  71.1533791 100.9256839
## 454  33.2406688  76.6167946
## 455  35.3688928 109.5033163
## 456   4.3476081  50.3627730
## 457   7.2181362  64.7704400
## 458  17.2623809  48.3682244
## 459  84.5969944 122.8074604
## 460  99.5835296 156.8726586
## 461  66.5443580 159.0442596
## 462  86.2582109 123.7796510
## 463  43.4998938 112.8204052
## 464  18.3158273  94.7129419
## 465  51.0846596 120.4357409
## 466  40.9401773 110.8698728
## 467  76.1620198 194.8479525
## 468  17.6968434  74.7835072
## 469  55.7254281 135.2269151
## 470  56.7024905 106.1078874
## 471  28.6770494  53.8904074
## 472  37.2825523 103.5232203
## 473  77.2547628 136.0864246
## 474  54.8044075 141.3418596
## 475  91.3794847 224.3013788
## 476  46.3132340  82.9866326
```

```
## 477  38.9537757  77.0503342
## 478  56.2990194 121.6500175
## 479  61.2560570 166.7649635
## 480  18.6466439  10.5060049
## 481  21.8908076  36.7753156
## 482  67.6463678 161.9115946
## 483  56.2083544 127.8109386
## 484  59.6899331 151.5661298
## 485  68.3654802 147.4357513
## 486  17.2839250  54.2153908
## 487  12.8801842 -33.7166451
## 488  35.3327109  66.5161425
## 489  72.1960789 173.8569312
## 490  28.6753266  46.3691799
## 491  45.5171049  40.6149550
## 492  52.4003601 146.3844032
## 493  80.0839050 125.5177187
## 494  83.0613543 165.6977843
## 495   2.2906226  -3.8026777
## 496  45.6055257 176.2903914
## 497  74.8179164 153.5901581
## 498  80.3189024 134.4578042
## 499  19.6144570  85.4588037
## 500   9.1178842  13.1073986
## 501  84.6324789 198.7133814
## 502  81.3800114 177.5417518
## 503  58.2229928 129.1418161
## 504  48.8159243 116.7722608
## 505  82.2896716 150.6840116
## 506  97.2286878 241.0487722
## 507  33.8056759  77.2023384
## 508  12.1208605  47.7660272
## 509  22.4534529  19.0177574
## 510  84.6766545 139.1826964
## 511  11.1942214  82.9170999
## 512  77.8872938 129.0544559
## 513   3.9066217  70.8360763
## 514  38.5602991  47.0089750
## 515  49.1465046  95.6907283
## 516  43.1296499 126.3090459
## 517  59.7630175 160.5986774
## 518  58.9223841 151.1140366
## 519  62.3170126 123.9445649
## 520  28.3400833  58.5488292
## 521  72.7239364 194.6555874
## 522  99.8419052 180.2896206
## 523  18.7219069  95.6969836
## 524  26.0130866  30.0888727
## 525   1.1571000  21.0617821
## 526  61.1249307 192.1871158
```

```
## 527  35.3215710  92.9791923
## 528  69.8276507 186.9998437
## 529  80.4661533 196.7336821
## 530  75.9827135 220.4672529
## 531   1.5200014 -24.1106052
## 532   3.2286450   3.6211455
## 533  16.0991785  25.4866127
## 534  47.4768291  96.6569065
## 535  95.7698435 152.4088161
## 536  23.7396119  15.7125092
## 537  75.2779414 154.7642756
## 538  27.0217075  15.4090538
## 539  37.8003138  87.7917112
## 540  68.1258151 131.4539022
## 541  25.6354562  31.4233042
## 542  37.1465915  86.5125226
## 543  97.1335928 152.9840341
## 544  99.1650051 204.1681603
## 545  68.3458518 178.8332947
## 546  81.6522026 218.1546554
## 547  70.9976393 155.4220132
## 548  23.2648554  91.0996927
## 549  86.2784838 211.3642354
## 550   5.1539275   6.9163113
## 551  82.2863553 204.6719477
## 552  38.1881200  93.3923406
## 553  40.9449324  53.8005727
## 554  95.9673897 189.7323036
## 555  84.6317289 199.4546132
## 556  96.0183412 224.5793492
## 557  60.3805025 124.6515824
## 558  45.2820115 119.6325983
## 559  87.0375858 145.1329704
## 560  11.8696341  -6.1579386
## 561   5.6466083   7.6592062
## 562  18.2969206 -28.8184154
## 563  73.7493857 130.1695352
## 564  70.5015857 155.8204976
## 565  96.1367921 216.7662363
## 566  39.8569418 107.8392756
## 567  39.1311852 123.3217983
## 568  34.1784919  77.3314704
## 569  87.3702993 140.7678071
## 570  33.8760159  57.5642880
## 571  44.1965942  73.1664355
## 572   0.1461641  18.1204024
## 573  36.3540727  97.1259631
## 574  42.8048811  48.1488158
## 575  97.4069546 214.3593576
## 576  72.4104546 142.9981432
```

```
## 577  53.0921684 141.9385661
## 578  27.2799078   6.1498285
## 579  82.8453779 118.9007059
## 580  34.9969909  96.4443490
## 581  61.9986940 116.0076559
## 582  82.7580780 126.2941928
## 583  74.3029461 199.2198014
## 584  75.1217117 112.2326106
## 585  68.6434880 104.5027650
## 586  14.5380026  29.7492822
## 587  68.3717751 147.6555786
## 588  19.3114243  14.9999630
## 589  90.0282676 189.1037060
## 590  93.3050952 198.4790819
## 591  23.2416155  72.7944842
## 592  33.1404226  75.3318147
## 593  78.2041980 172.8256583
## 594  24.3529093  46.1795604
## 595  99.6540433 271.3180675
## 596  13.0706034  45.3076687
## 597  90.2479012 195.5055324
## 598  64.3554711 148.6550539
## 599  55.7402163 159.0130645
## 600  14.7050499  58.1754115
## 601  20.8293297  92.1475009
## 602  29.7954647  92.3802488
## 603  72.0117925 174.5591008
## 604  44.0913757 138.4108693
## 605  49.9484087 116.8614896
## 606  43.7544710 101.9610645
## 607  57.5123101 154.7288579
## 608  25.0239152  75.0563482
## 609  81.4032817 197.2599533
## 610  39.0290015 112.6715186
## 611  91.6923284 172.8509356
## 612  13.0419930  52.1795715
## 613  72.6861636 149.7743485
## 614  88.4602160 167.8339367
## 615  70.1302609 160.0084112
## 616  21.2673909  46.0712754
## 617  55.8587966 153.0621398
## 618  25.2319287  55.6703593
## 619  79.8516582 115.1949252
## 620  57.3681825 145.7135826
## 621  37.9520774 108.3179254
## 622  82.7449367 180.2399790
## 623   6.4037700  14.4369822
## 624  34.9627004 117.5352239
## 625  19.8672525  82.0220178
## 626  66.9755885 162.9839964
```

```
## 627  97.5912791 185.0755261
## 628  73.0080426 130.5811293
## 629  32.8766371  68.2172638
## 630  48.4269353 111.9227073
## 631  64.4490048 110.4697395
## 632   9.5970686  40.2750484
## 633  36.8749471 116.9575666
## 634  88.3087594 178.8188998
## 635  19.5766239  15.6789838
## 636  12.9054395  31.2558256
## 637  20.6583072  59.3565978
## 638  23.7482195  33.3004180
## 639  35.3740031  57.1334749
## 640  45.4945335 163.4374064
## 641   8.6305237  38.3878944
## 642  53.9777684 157.3369584
## 643  44.0419089 112.7821640
## 644  49.9776019 134.0103224
## 645  97.0687008 189.1132293
## 646  37.8892018 146.5333460
## 647  49.5695214 151.4823632
## 648  17.5397323  68.7662331
## 649  13.2875106  27.4006654
## 650  92.9934646 202.2896913
## 651  26.2916434 -16.2575780
## 652  34.6671266  96.8696925
## 653   5.4716707 -24.5841792
## 654   6.6876184   0.4475519
## 655  75.2188797 172.2468392
## 656   4.9365602  22.8912276
## 657  33.0958428  81.4938007
## 658  46.7179917  81.9420696
## 659  47.8050753 114.4095747
## 660  91.3466343 168.9378980
## 661  51.3399950 134.2297077
## 662  85.8610708 187.4414119
## 663  92.7869951 193.6483366
## 664  67.7642762 168.4326076
## 665  93.4144766 221.6506746
## 666  96.7087477 231.9405243
## 667  18.9255516  43.5686399
## 668  10.3519924  11.2432846
## 669  27.4997714  81.1493199
## 670  52.8368085 117.1211482
## 671  67.8770622 186.9891227
## 672  83.8250346 230.6025680
## 673  30.6957990  34.1688040
## 674  38.6647286  69.2846182
## 675  56.6780888 116.8885292
## 676  68.7702546 171.5821857
```

```
## 677  35.1980507 113.6047220
## 678  99.8158555 209.0786787
## 679  12.7166312  40.3130419
## 680  25.2244596  38.4808894
## 681  31.4534330  75.3019363
## 682  20.4567999  45.8117475
## 683  33.9297763  86.5368538
## 684  13.9556454  14.1826586
## 685  11.9112252  38.2299282
## 686  35.0821897 137.5615265
## 687  40.4098538  88.9441620
## 688  46.9849828  34.0878107
## 689  81.9538031 162.9356540
## 690  80.9486795 158.5800996
## 691  32.9454965  78.1566190
## 692  28.5369874  59.2235482
## 693  70.7000566 159.1957176
## 694   5.8940822  61.1762275
## 695  65.4160840 171.2971041
## 696  72.9464319 176.4685329
## 697  98.0348968 231.8553804
## 698  66.5951937 111.8751210
## 699  13.3121670  29.8497880
## 700  16.4890675  11.7376071
## 701  98.4913399 179.8431507
## 702  53.0649773 128.0713865
## 703  32.8210266  47.4501987
## 704  80.3497054 186.5466684
## 705  16.5441929  16.1117160
## 706  57.6787527  96.4466074
## 707  25.1590402  75.1023117
## 708   6.3072245  25.4838171
## 709  35.5203215  64.1557506
## 710  83.7694726 181.3748425
## 711  95.6709778 243.1004998
## 712  13.5919862  26.1311872
## 713  72.8212759 152.5458634
## 714  71.9032699 158.9840974
## 715   2.2299691  32.7490734
## 716  17.5667550  -2.3917270
## 717  70.6408889 136.3874377
## 718  83.8236285 231.4943930
## 719  14.9397254  31.5852849
## 720   2.8605513  33.3729935
## 721   2.1892189  47.4794513
## 722  14.9035381  -2.5648912
## 723  50.7028771 124.4102592
## 724   9.3719840  35.1187726
## 725  53.0829383 116.2178211
## 726  54.1383354 121.7998328
```

```
## 727  19.4377313  74.7116978
## 728  35.0761714  74.7305535
## 729  71.4161024  82.5763343
## 730  92.9909949 220.5761560
## 731  72.2252982 140.9102899
## 732  61.9926014  56.9761937
## 733  41.7272337  49.8202556
## 734  85.0024125 206.1297162
## 735   5.0376552  51.2751994
## 736  21.0346055  67.5740475
## 737   0.5645328 -32.0307272
## 738  12.3842695  45.9839010
## 739   5.0513612  42.7480534
## 740  14.5343085  69.9003229
## 741  95.1862748 196.7847008
## 742  69.4615156 182.3524459
## 743  75.2354713 123.4397127
## 744  96.9538963 195.2472151
## 745   0.3250627 -12.5626334
## 746  27.1505175 102.4290300
## 747  51.4651617  83.6678386
## 748  10.0369118  55.1359673
## 749  27.7505639  58.3491395
## 750   3.7688043  30.4566473
## 751  29.0135023  74.6575035
## 752  11.0340315  21.0079480
## 753  65.0648064 117.2603320
## 754  43.6553689 117.4969892
## 755   4.6061063  10.9406286
## 756  34.9673834  12.4125035
## 757  79.1302646 174.2670769
## 758  75.3282245 181.4272696
## 759   1.0948373  18.8775263
## 760   8.5665365  19.7396980
## 761  72.8802505 163.0834498
## 762  52.7868365 117.2169909
## 763  17.8821407  41.7963413
## 764  77.7574945 144.9068581
## 765   9.7513117  -7.3210445
## 766  88.7242790 228.3333082
## 767  59.0078983 125.6515150
## 768  16.3668311  74.1841263
## 769  68.1553108 109.9042288
## 770  96.9201076 205.1982827
## 771  96.7498695 246.0872279
## 772  33.0744958  72.2642718
## 773  91.2266768 238.0515565
## 774  63.7811314 110.9842768
## 775  37.4681774  84.6376041
## 776  79.2483218 138.7886956
```

```
## 777   7.1155177  68.6180735
## 778  27.1208116  66.2328317
## 779  29.6648284  80.7564158
## 780  86.1354417 163.0355070
## 781  62.8671946 174.0466391
## 782  47.6460154 105.5513647
## 783  45.4199471 173.4887711
## 784  62.7523187 175.4755388
## 785  80.2087803 158.0779564
## 786  19.4016432  68.7701838
## 787  60.3549114 113.3649799
## 788  17.7433219  22.3262886
## 789   4.4427621  57.7220766
## 790  92.7807257 210.1824488
## 791  36.6503225 119.8686633
## 792  63.3058207 192.2344658
## 793  69.3330718 164.3199869
## 794  41.6437416  72.4651160
## 795  72.7661947 145.0844160
## 796   3.9081282  47.9134042
## 797  69.0791604 180.5851433
## 798  91.9848261 184.1946007
## 799  88.4223533 236.5573761
## 800   8.8096177   9.9818546
## 801  72.9757475 179.9100181
## 802   4.3384982  70.8690610
## 803  43.0518264  75.4072139
## 804  23.8532109  62.3952540
## 805  31.9116821 110.4860311
## 806  17.8296987  23.7792167
## 807  63.5990588  74.7640066
## 808  49.1192966 159.3092290
## 809  54.3454396 165.4063591
## 810  14.8978348  76.4703328
## 811  61.6877287 107.3356862
## 812  91.5774415 139.4663407
## 813  53.7432673 141.9959617
## 814  27.4931956  71.3948059
## 815  13.4227193  75.7586244
## 816   5.2033790  66.6885195
## 817  53.2582855  87.6265898
## 818  40.6337745  71.7435100
## 819  96.3592288 196.7511661
## 820  65.4367359 170.8713605
## 821  89.4368492 160.7033269
## 822  84.8813646 166.0924680
## 823  41.5859904 119.6467791
## 824  19.8515207  43.0843594
## 825  98.2266829 220.4716341
## 826  67.5213261 156.4723361
```

```
## 827   1.0789159 -41.4447127
## 828  52.3881093 143.9998504
## 829  94.0705648 205.5119480
## 830  54.5025400 122.2691027
## 831  85.2222867 180.6235758
## 832  30.3281275 122.4215745
## 833  22.6497779 114.8231578
## 834  38.4778490  91.7143665
## 835  25.8265500  47.5867064
## 836  31.5566368  62.9343970
## 837  26.4956818  76.9119660
## 838  80.5967750 185.8238484
## 839  40.4374534  61.9237607
## 840  34.5168184  50.7442110
## 841   5.3011476   3.3702421
## 842  74.0195536 194.7976272
## 843  47.9913572 162.9694157
## 844  84.0279360 196.5274506
## 845  31.3947121  35.6702170
## 846  17.8973245  66.5649905
## 847  77.1455286 119.0633727
## 848  22.2446679   2.5823669
## 849  48.0056035  49.6817998
## 850   6.4822778  16.4247092
## 851  94.3057752 199.5332469
## 852  28.7876759  22.9944943
## 853  78.1830892 169.2701491
## 854  59.8620097 135.0316812
## 855  67.1211901 131.6590917
## 856   7.5497020  75.5965241
## 857  89.9866526 195.4222891
## 858  63.7655885 150.2676050
## 859  53.9205927 183.5119636
## 860  70.6768499 189.9796854
## 861  16.1336052  39.6289632
## 862   2.2448101  73.4764758
## 863  13.1880307  41.3975145
## 864  35.2519216 100.5630902
## 865  74.7489674 132.7480986
## 866  91.6594520 193.1741921
## 867   8.1783015 -14.0362813
## 868  17.9005455  36.9447638
## 869  73.5234855 130.2187050
## 870  89.7623495 135.6052234
## 871   6.1329427  52.0747085
## 872  11.6307009  21.3544135
## 873  14.9930426  32.8506715
## 874   6.9864491  29.3003286
## 875  95.7913008 219.7312932
## 876  80.8619623 132.6504620
```

```
## 877  78.2926845 126.7011747
## 878  28.3944302  24.0401023
## 879  83.3246879 131.5463872
## 880  82.2540657 206.8041842
## 881  71.8321836 147.1814918
## 882  25.9678927  37.1730488
## 883  94.0365043 213.7483570
## 884  71.5864641 131.2723844
## 885  68.0218705  89.3513377
## 886  70.4527276 148.4081738
## 887  78.1742760 196.9360525
## 888  67.8747989  89.1050881
## 889  62.5472401  96.1330214
## 890  66.9927762 137.8389231
## 891  45.7381321  79.5664274
## 892   3.0349290 -13.3954855
## 893  15.4949990  33.8458309
## 894  58.9204709 147.8459016
## 895  56.3205237 125.8359084
## 896  54.0859143 119.2332832
## 897  68.3520011 154.2173963
## 898  15.8243428  72.7229772
## 899  66.3920239 156.8008767
## 900  37.1003909  81.9160089
## 901  99.1113822 211.4071990
## 902  20.1619658  40.2506434
## 903  86.8525394 233.8711555
## 904  42.5209466  83.1493408
## 905  14.2552989  64.8196598
## 906  95.5367716 201.9826349
## 907  94.1246236 206.0876655
## 908  60.9607610 159.5712859
## 909  23.5639822 102.3105289
## 910  71.9063616 163.5611364
## 911  85.2745940 164.0798706
## 912  22.9209306  76.3033385
## 913  29.7649475  69.2341547
## 914  23.5654548  38.9175625
## 915  13.7858747  18.2256682
## 916  50.5304563 150.3840950
## 917  44.7101399 125.6368696
## 918  37.6138863  82.9093457
## 919  51.3781934 152.0665686
## 920  60.4908770 122.4785293
## 921   9.4809600  16.0951953
## 922  98.2359343 235.1830819
## 923  57.0996541 145.3755099
## 924  54.7019221 132.3452425
## 925  15.9312244  71.2954353
## 926  71.9312283 153.6624762
```

```
## 927  86.1736986 177.4772295
## 928   4.6713573 -26.1159467
## 929  18.8486650  10.3133681
## 930  83.6620220 203.2347760
## 931  17.7795314  23.0607380
## 932  20.3426831 115.7597650
## 933  16.4025671  26.2758957
## 934  70.5283038 165.7563349
## 935  92.6734628 192.7926930
## 936  82.0726589 192.9511263
## 937   7.9541918  47.3425139
## 938  61.6386127 139.5348379
## 939  34.0390801  71.7348753
## 940   6.1924917  -1.3894442
## 941  19.8074832  27.3331905
## 942  54.0604011  99.9831882
## 943   0.2201621   5.1802841
## 944  23.2474338 118.2785499
## 945  57.7513927 115.5274470
## 946  12.7315238  65.5282171
## 947  62.7500467 181.0468685
## 948  84.0236510 193.4209946
## 949  14.6082814  72.9087466
## 950  64.7223075 141.0792679
## 951  17.2521740  33.4387586
## 952  98.4438104 199.0527009
## 953   8.8404605  66.0913706
## 954  22.7591546  75.1815810
## 955  27.4074932  84.0323744
## 956   0.1405333 -15.3237943
## 957  98.6634395 176.9839488
## 958  95.9145926 214.1075500
## 959  24.6692404  58.0353697
## 960  17.3460119 102.6376987
## 961  62.7830304  95.4077644
## 962  63.3161657 149.7350234
## 963  50.4735171 120.2332167
## 964  38.9797114 136.8533037
## 965  25.6320836  63.1121816
## 966  40.8382950 102.4759885
## 967  32.2735578  72.6441419
## 968  43.3312124 103.0513640
## 969  37.1435601  44.0329415
## 970  45.5450363 112.4501207
## 971  35.6999712  27.9405635
## 972  32.0924953  33.6726865
## 973  39.5445084  61.1745355
## 974  40.4201159 114.4662358
## 975  54.8536098 136.1297919
## 976  69.9548431 147.9288357
```

```
## 977   24.7545079 115.3769538
## 978   74.8506438  86.3544398
## 979   44.2908970  70.0723188
## 980   14.7134966  13.8071335
## 981   77.5317977 168.1239856
## 982   50.5420844 100.9180668
## 983   31.6420158  93.6813230
## 984   50.3396237 128.8707558
## 985   98.5622654 209.7743197
## 986   36.3373139  63.5112395
## 987   12.7420277  22.2015408
## 988   97.1795389 192.2920329
## 989   25.8373311  99.1590957
## 990   69.3429546 151.3818987
## 991   20.3024730  77.0932579
## 992   63.9586638 114.3541362
## 993   33.5629706  50.4203051
## 994   70.3610809 100.2104938
## 995   58.5661038 111.8574195
## 996   72.5936780 154.4446149
## 997   77.7737945 154.5832525
## 998   46.4846401 124.6993866
## 999   43.3517396  77.7743702
## 1000 23.9130954  84.8667672
```

###6.3 Variation, uncertainty, and sample size: Repeat the example in Section 6.2, varying the number of data points, n. What happens to the parameter estimates and uncertainties when you increase the number of observations?

```
simulate_regression_no_charts <- function(a,b,n,sigma){
  x <- runif(n, min=0, max=100)

  error <- rnorm(n, mean=0,  sd = sigma)

  y <- a + b*x + error

  model <- lm(y~x)
 # Commenting out charts so that i can run a better sensitivity test

  #plot(x, y, main = "Scatterplot with Fitted Regression Line",
  #   xlab = "x", ylab = "y", pch = 19, col = "blue")
  #abline(model, col = "red", lwd = 2)

  # Return the data as a data frame
  data <- data.frame(x = x, y = y)

   return(data)
}
```

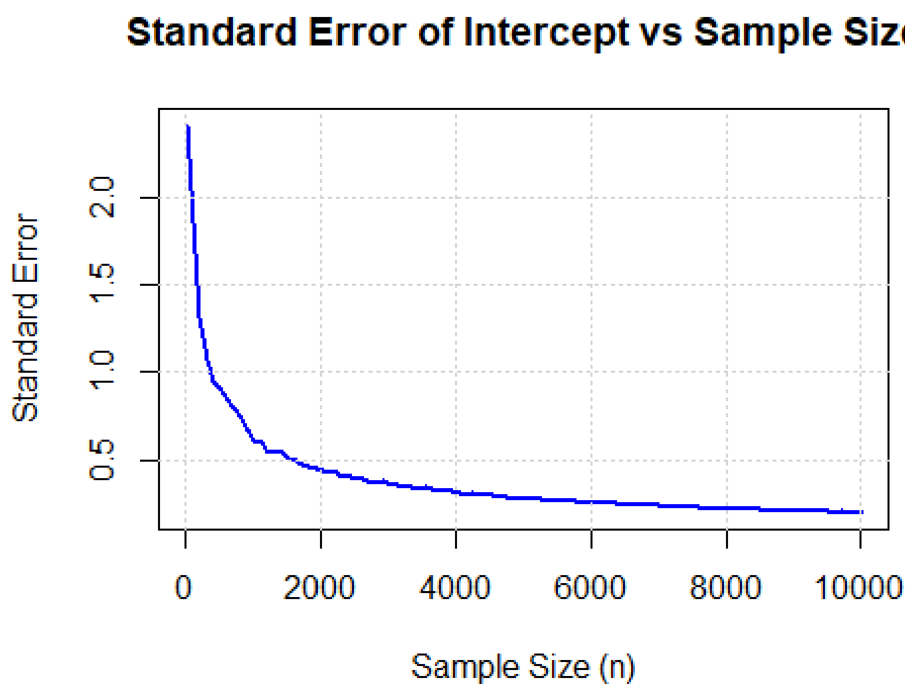
```r
#parameters
a <- 5
b <- 7
signma <- 10

#sample sizes
n_values <- round(seq(10,10010, length.out= 100))

results <- data.frame(
  n = integer(),
  est_intercept = numeric(),
  se_intercept = numeric(),
  est_slope = numeric(),
  se_slope = numeric()
)

for (n in n_values) {
  data <- simulate_regression_no_charts(a, b, n, signma)

  # Fit the model
  model <- lm(y ~ x, data = data)
  summary_model <- summary(model)

  # Extract estimates and standard errors
  est_intercept <- summary_model$coefficients["(Intercept)", "Estimate"]
  se_intercept <- summary_model$coefficients["(Intercept)", "Std. Error"]
  est_slope <- summary_model$coefficients["x", "Estimate"]
  se_slope <- summary_model$coefficients["x", "Std. Error"]

  # Store the results
  results <- rbind(
    results,
    data.frame(
      n = n,
      est_intercept = est_intercept,
      se_intercept = se_intercept,
      est_slope = est_slope,
      se_slope = se_slope
    )
  )
}

print(results)

##         n est_intercept se_intercept est_slope    se_slope
## 1      10    -0.2984588    2.4101669  7.162654 0.050137633
## 2     111     7.3727881    1.8531282  6.952761 0.031970378
## 3     212     5.3932445    1.3197307  6.982085 0.023437812
## 4     313     4.0111361    1.0699926  7.034475 0.019141893
```

```
## 5     414     4.6106675    0.9525570  7.007419 0.016817652
## 6     515     6.3444114    0.9159616  6.980748 0.015514418
## 7     616     4.6143350    0.8455657  7.024108 0.014792695
## 8     717     4.9485038    0.7921526  7.001470 0.013721477
## 9     818     5.0116671    0.7352097  7.007775 0.012948442
## 10    919     5.1000719    0.6737492  6.992266 0.011916869
## 11   1020     5.9156279    0.6178553  6.989265 0.010528769
## 12   1121     5.1283602    0.6010279  7.002346 0.010288827
## 13   1222     4.9594849    0.5382616  7.003365 0.009543735
## 14   1323     4.2330312    0.5416612  7.010894 0.009445748
## 15   1424     5.3348407    0.5496858  6.988232 0.009438682
## 16   1525     4.5655485    0.5102539  7.008805 0.008792467
## 17   1626     5.7362872    0.5002362  6.992965 0.008652677
## 18   1727     4.4731593    0.4796339  7.015517 0.008260663
## 19   1828     5.5801376    0.4657659  6.997201 0.008056305
## 20   1929     4.7133330    0.4509385  6.998627 0.007913647
## 21   2030     5.7470892    0.4402052  6.995001 0.007623775
## 22   2131     5.0835853    0.4341420  6.992251 0.007507131
## 23   2232     5.2179512    0.4250741  7.004620 0.007467892
## 24   2333     4.9209104    0.4093710  6.997465 0.007051967
## 25   2434     5.0425427    0.4003017  6.996227 0.006962289
## 26   2535     4.9233922    0.3985352  6.999535 0.006900093
## 27   2636     4.5096096    0.3943884  7.008106 0.006775909
## 28   2737     5.1883453    0.3689939  6.996792 0.006506945
## 29   2838     4.9362564    0.3724861  7.003019 0.006568310
## 30   2939     4.3604489    0.3786644  7.011372 0.006501795
## 31   3040     4.6565427    0.3616583  7.004852 0.006262471
## 32   3141     5.0777500    0.3577895  7.000204 0.006195164
## 33   3242     4.5397245    0.3517099  7.006402 0.006096925
## 34   3343     4.6333283    0.3450395  7.000983 0.006028924
## 35   3444     5.3078306    0.3381938  6.991388 0.005884604
## 36   3545     5.2781372    0.3436849  6.994544 0.005964373
## 37   3646     4.8357722    0.3340386  7.005328 0.005786795
## 38   3747     5.1376405    0.3294579  6.998767 0.005753022
## 39   3848     5.9735569    0.3286409  6.987086 0.005670815
## 40   3949     4.9603220    0.3207530  7.002166 0.005596659
## 41   4050     5.2866163    0.3084550  6.997414 0.005423093
## 42   4151     5.3308501    0.2992990  6.993675 0.005255446
## 43   4252     5.1497685    0.3126358  6.998261 0.005433060
## 44   4353     4.8401143    0.2993205  7.005662 0.005203035
## 45   4454     4.8600307    0.2999982  7.004064 0.005214689
## 46   4555     5.3611081    0.2981406  7.000372 0.005165529
## 47   4656     5.5530314    0.2955763  6.995460 0.005099453
## 48   4757     4.7496006    0.2926770  7.002983 0.005006755
## 49   4858     4.7367297    0.2844766  7.003609 0.004925294
## 50   4959     5.4639274    0.2804441  6.992806 0.004856386
## 51   5061     4.6498547    0.2782795  7.007442 0.004852511
## 52   5162     4.8804580    0.2755414  7.000578 0.004779313
## 53   5263     4.9284283    0.2771678  7.001859 0.004853677
## 54   5364     5.1079361    0.2709390  6.999100 0.004680756
```

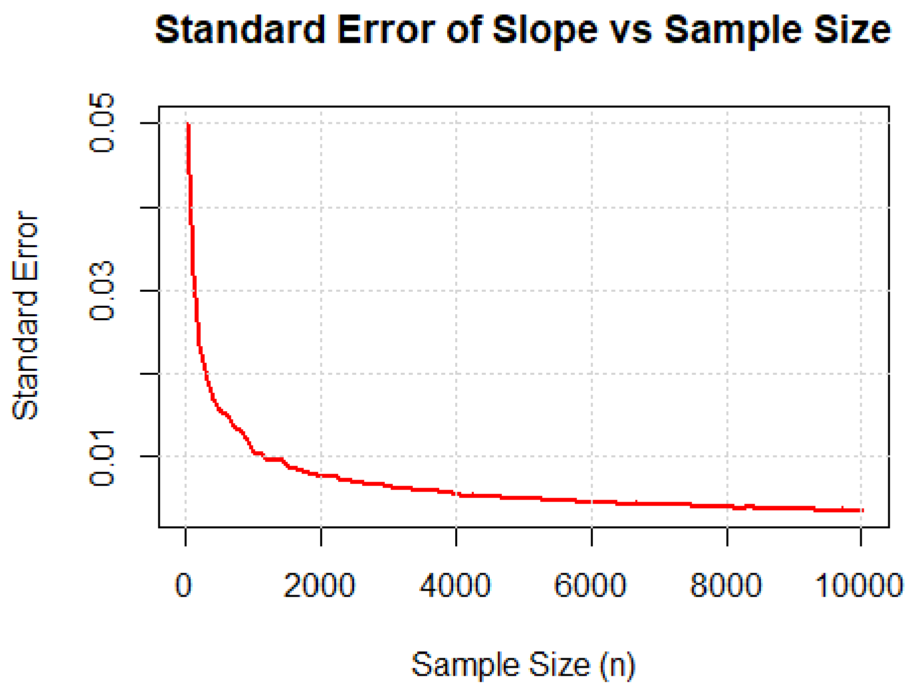
```
## 55   5465     5.1406614    0.2714867  6.998872 0.004706354
## 56   5566     4.9203340    0.2732268  7.003475 0.004741380
## 57   5667     4.7369441    0.2660238  7.005365 0.004608589
## 58   5768     5.2309232    0.2652074  6.992225 0.004590946
## 59   5869     4.7387268    0.2571412  7.007317 0.004479423
## 60   5970     4.8717427    0.2596526  7.002470 0.004469115
## 61   6071     5.1475136    0.2558333  6.999820 0.004431443
## 62   6172     4.9949495    0.2529722  6.998019 0.004388836
## 63   6273     5.1054490    0.2581740  7.000782 0.004437717
## 64   6374     5.0172783    0.2506240  7.000130 0.004370658
## 65   6475     4.6123852    0.2466795  7.002378 0.004279847
## 66   6576     5.0467488    0.2461438  6.998204 0.004234478
## 67   6677     4.9717707    0.2475338  7.002306 0.004299052
## 68   6778     4.9388025    0.2453780  6.997397 0.004235900
## 69   6879     5.2936976    0.2408908  6.996231 0.004177189
## 70   6980     5.1749472    0.2405274  6.994755 0.004144314
## 71   7081     5.0230013    0.2381989  7.001269 0.004133001
## 72   7182     5.3301244    0.2357985  6.994927 0.004083435
## 73   7283     5.3269282    0.2308252  6.997470 0.004041912
## 74   7384     4.9751620    0.2341563  7.002358 0.004055961
## 75   7485     5.4344347    0.2364857  6.995487 0.004086507
## 76   7586     5.4547275    0.2325164  6.993019 0.004011427
## 77   7687     4.8147862    0.2265517  7.002176 0.003918621
## 78   7788     4.4946509    0.2267092  7.005510 0.003930355
## 79   7889     5.1187201    0.2219431  6.997319 0.003847380
## 80   7990     4.6942037    0.2216507  7.006248 0.003850532
## 81   8091     4.8208285    0.2203868  7.001230 0.003827092
## 82   8192     5.1196123    0.2186119  6.998099 0.003778028
## 83   8293     4.9074552    0.2189527  6.998105 0.003816781
## 84   8394     4.9459619    0.2185346  6.999838 0.003807026
## 85   8495     5.0884462    0.2165774  7.002375 0.003751738
## 86   8596     4.8876756    0.2123849  6.999536 0.003687993
## 87   8697     4.7411749    0.2134085  7.006389 0.003706996
## 88   8798     5.0704611    0.2136191  6.999597 0.003688751
## 89   8899     4.8813264    0.2108691  7.002701 0.003646473
## 90   9000     5.1340683    0.2119200  6.992801 0.003668026
## 91   9101     5.2174219    0.2093906  6.996973 0.003642060
## 92   9202     5.1883996    0.2117396  7.000336 0.003666011
## 93   9303     5.2671235    0.2060914  6.996249 0.003562593
## 94   9404     4.8933164    0.2049586  7.003132 0.003542078
## 95   9505     5.0508799    0.2052855  6.996256 0.003539322
## 96   9606     5.0673573    0.2025747  6.997692 0.003499642
## 97   9707     5.3504103    0.2058822  6.994924 0.003568242
## 98   9808     5.2250351    0.2006811  6.998058 0.003458256
## 99   9909     5.1334261    0.1997695  6.999076 0.003463779
## 100 10010     5.0527729    0.2004298  6.997055 0.003481720
```

```r
plot(results$n, results$se_intercept, type = "l", col = "blue", lwd = 2,
     xlab = "Sample Size (n)", ylab = "Standard Error",
     main = "Standard Error of Intercept vs Sample Size")
```

```r
grid()
```




```r
plot(results$n, results$se_slope, type = "l", col = "red", lwd = 2,
     xlab = "Sample Size (n)", ylab = "Standard Error",
     main = "Standard Error of Slope vs Sample Size")

grid()
```



Answer: While the slope and intercept do increase or vary throughout the samples, the main difference is that the standard error for both go down dramatically as the number of samples goes up. that said there is a point of diminishing returns as standard error plateaus

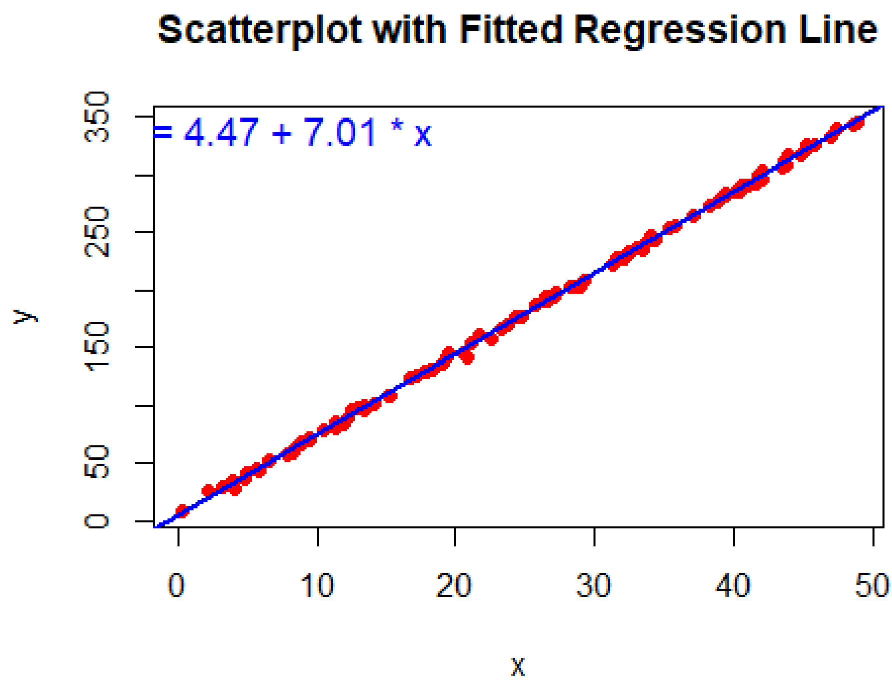
##Chapter 7:

###7.2, Fake-data simulation and regression: Simulate 100 data points from the linear model, y = a + bx + error, with a = 5, b = 7, the values of x being sampled at random from a uniform distribution on the range [0, 50], and errors that are normally distributed with mean 0 and standard deviation 3. (a) Fit a regression line to these data and display the output. (b) Graph a scatterplot of the data and the regression line. (c) Use the text function in R to add the formula of the fitted line to the graph.

```r
# Parameters
n <- 100
a <- 5
b <- 7
sigma <- 3

x <- runif(n, min = 0, max = 50)
error <- rnorm(n, mean = 0, sd = sigma)
y <- a + b * x + error

model <- lm(y ~ x)
summary_model <- summary(model)
print(summary_model)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.3331 -1.5826 -0.0354  1.8830  6.4907
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.47453    0.51961   8.611 1.24e-13 ***
## x            7.00876    0.01825 384.042  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.514 on 98 degrees of freedom
## Multiple R-squared:  0.9993, Adjusted R-squared:  0.9993
## F-statistic: 1.475e+05 on 1 and 98 DF,  p-value: < 2.2e-16
```

```
plot(x, y, main = "Scatterplot with Fitted Regression Line",
     xlab = "x", ylab = "y", pch = 19, col = "red")
abline(model, col = "blue", lwd = 2)

coefficients <- coef(model)
formula_text <- paste0("y = ", round(coefficients[1], 2),
                       " + ", round(coefficients[2], 2), " * x")
# Position the text on the plot
text_x <- min(x) + 7
text_y <- max(y)- 10
text(text_x, text_y, labels = formula_text, col = "blue", cex = 1.2)
```



###7.6 Formulating comparisons as regression models: Take the election forecasting model and simplify it by creating a binary predictor defined as x = 0 if income growth is less than 2% and x = 1 if income growth is more than 2%. (a) Compute the difference in incumbent party's vote share on average, comparing those two groups of elections, and determine the standard error for this difference. (b) Regress incumbent party's vote share on the binary predictor of income growth and check that the resulting estimate and standard error are the same as above.

```
election_data = read.table("ROS-Examples-
master/ElectionsEconomy/data/hibbs.dat", header=TRUE)


election_data$x <- ifelse(election_data$growth > 2,1,0)

mean_vote_low <- mean(election_data$vote[election_data$x == 0], na.rm = TRUE)

mean_vote_high <- mean(election_data$vote[election_data$x == 1], na.rm=TRUE)

diff_means <- mean_vote_high - mean_vote_low

# Number of observations in each group
n_low <- sum(election_data$x == 0)
n_high <- sum(election_data$x == 1)
```

```r
# Standard deviation for each group
sd_low <- sd(election_data$vote[election_data$x == 0], na.rm = TRUE)
sd_high <- sd(election_data$vote[election_data$x == 1], na.rm = TRUE)

# Standard error of the difference
se_diff <- sqrt((sd_low^2 / n_low) + (sd_high^2 / n_high))

cat("Difference in mean vote share:", diff_means, "\n")

## Difference in mean vote share: 5.5075

cat("Standard error of the difference:", se_diff, "\n\n")

## Standard error of the difference: 2.502052

model <- lm(vote~x, data=election_data)

estimate = coef(model)['x']

se_estimate <- summary(model)$coefficients["x", "Std. Error"]

cat("Regression estimate for x:", estimate, "\n")

## Regression estimate for x: 5.5075

cat("Standard error of the estimate:", se_estimate, "\n\n")

## Standard error of the estimate: 2.502052
```