MLE of Location Parameter of a Cauchy Distribution by Using Different Algorithms

HW 3 of STAT 5361 Statistical Computing

Biju Wang[1]

09/18/2018

[1]bijuwang@uconn.edu

Contents

Chapter 1

Proofs and Loglikelihood Function Plot against θ

1.1 Proofs

The likelihood function is

$$L(\theta) = \prod_{i=1}^{n} \frac{1}{\pi[1 + (X_i - \theta)^2]}$$

Hence, the log likelihood fucntion is

$$l(\theta) = \log L(\theta) = \log \prod_{i=1}^{n} \frac{1}{\pi[1 + (X_i - \theta)^2]} = -n \log \pi - \sum_{i=1}^{n} \log[1 + (\theta - X_i)^2] \tag{1.1}$$

Further

$$l'(\theta) = -\sum_{i=1}^{n} \frac{d}{d\theta} \log[1 + (\theta - X_i)^2] = -2 \sum_{i=1}^{n} \frac{\theta - X_i}{1 + (\theta - X_i)^2} \tag{1.2}$$

Compute the second derivative according to $l'(\theta)$

$$l''(\theta) = -2 \sum_{i=1}^{n} \frac{d}{d\theta} \frac{\theta - X_i}{1 + (\theta - X_i)^2} = -2 \sum_{i=1}^{n} \frac{1 - (\theta - X_i)^2}{[1 + (\theta - X_i)^2]^2} \tag{1.3}$$

Therefore, the Fisher information is

$$\begin{aligned}
I_n(\theta) &= -E_X[l''(\theta)] \\
&= 2nE_X \left[\frac{1 - (\theta - X)^2}{[1 + (\theta - X)^2]^2} \right] \\
&= 2n \int_{-\infty}^{\infty} \frac{1 - (x - \theta)^2}{[1 + (x - \theta)^2]^2} \frac{1}{\pi[1 + (x - \theta)^2]} dx \\
&= \frac{2n}{\pi} \int_{-\infty}^{\infty} \frac{1 - x^2}{(1 + x^2)^2} \frac{1}{1 + x^2} dx = \frac{2n}{\pi} \int_{-\infty}^{\infty} \left(\frac{x}{1 + x^2} \right)' \frac{1}{1 + x^2} dx = \frac{2n}{\pi} \left[\frac{x}{(1 + x^2)^2} \Big|_{-\infty}^{\infty} + \int_{-\infty}^{\infty} \frac{2x^2}{(1 + x^2)^3} dx \right] \\
&= \frac{4n}{\pi} \int_{-\infty}^{\infty} \frac{x^2}{(1 + x^2)^3} dx = \frac{4n}{\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \frac{\tan^2 \alpha}{(1 + \tan^2 \alpha)^3} \sec^2 \alpha d\alpha = \frac{4n}{\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \frac{1 - \cos 4\alpha}{8} d\alpha = \frac{4n}{\pi} \cdot \frac{\pi}{8} \\
&= \frac{n}{2}
\end{aligned} \tag{1.4}$$

1.2 Loglikelihood Function Plot against θ

The following plot is the curve of log likelihood function

```
set.seed(20180909)
sample <- rcauchy(10, 5)

log_sum <- function(x, sample){
  log_sum <- 0
  for (i in 1:length(sample)) {
    log_sum <- log_sum -log(pi) - log(1 + (x - sample[i])^2)
  }
  log_sum
}

library("ggplot2")
```

```
## RStudio Community is a great place to get help:
## https://community.rstudio.com/c/tidyverse.
```

```
ggplot(data.frame(x = c(0, 10)), aes(x = x)) +
stat_function(fun = function(x) log_sum(x, sample)) +
labs(x = expression("Values of"~theta), y = expression("Log Likelihood Function"~l(theta))) +
theme(plot.title = element_text(hjust = 0.5)) +
ggtitle(expression("Log Likelihood Function vs."~theta))
```

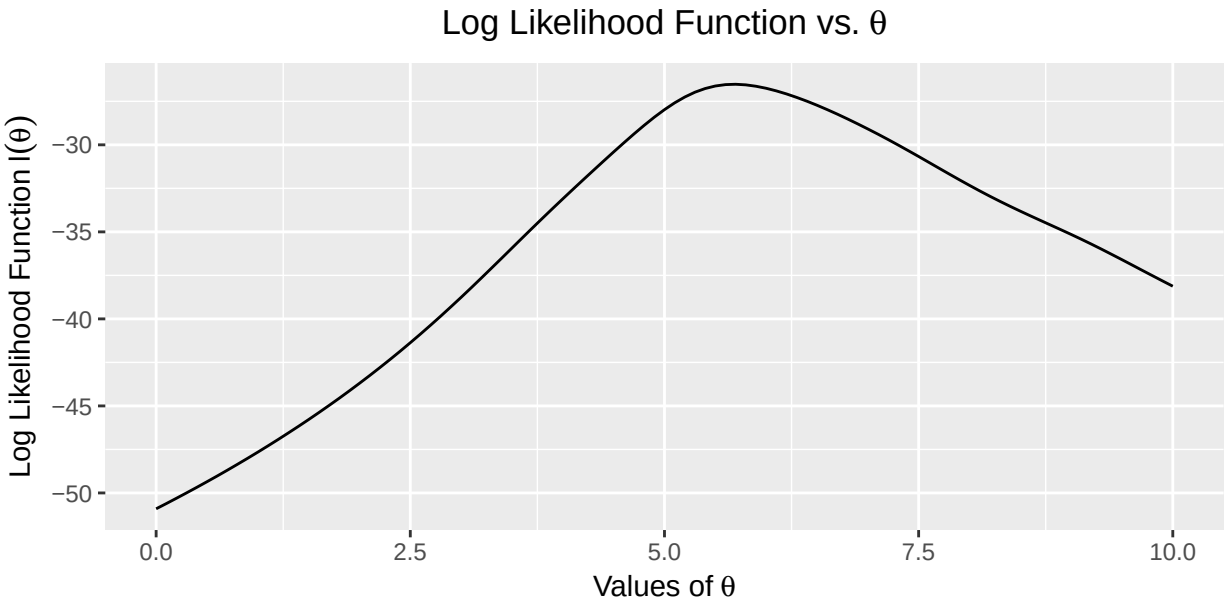

Figure 1.1: Log Likelihood Function vs. θ

Chapter 2

Newton-Raphson Method

From the tables and figure below we can see, when the initial values is not less than 5 and around 5, the outcomes are quite close to the true value 5. For instance, from the table when the initial values are $5, 5, 5, 6, 6.5$, the roots are $5.69, 5.69, 5.69, 5.69$. While for other initial values, the roots are quite unstable, for some the roots are very large and for others the roots only have magnitude of 10.

```
set.seed(20180909)
sample <- rcauchy(10, 5)

dev1_log_sum <- function(x){
  dev1_log_sum <- 0
  for (i in 1:length(sample)) {
    dev1_log_sum <- dev1_log_sum - 2 * (x - sample[i])/(1 + (x - sample[i])^2)
  }
  dev1_log_sum
}

dev2_log_sum <- function(x){
  dev2_log_sum <- 0
  for (i in 1:length(sample)) {
    dev2_log_sum <- dev2_log_sum - 2 * (1 - (x - sample[i])^2)/(1 + (x - sample[i])^2)^2
  }
  dev2_log_sum
}

newton.raphson <- function(init, fun, fun.dev, maxiter = 100, tol = .Machine$double.eps^0.2){
  x <- init
  for (i in 1:maxiter) {
    x1 <- x -fun(x)/fun.dev(x)
    if(abs(x1 - x) < tol) break
    x <- x1
  }
  if(i == maxiter)
    message("Reached the maximum iteration!")

  return(data.frame(root = x1, iter = i))
}

init <- seq(-10, 20, by = 0.5)
```

```
res <- data.frame(init = init, root = rep(NA, length(init)))
for (i in 1:length(init)) {
  res$root[i] <- newton.raphson(init[i], dev1_log_sum, dev2_log_sum)$root
}

res_trans <- t(as.matrix(round(res, 2)))
rownames(res_trans) <- c("Initial Values", "Roots")

library("pander")
library("ggplot2")
pander(res_trans, split.table = 100, style = 'rmarkdown')
```

Table 2.1: Table continues below

Initial Values	-10	-9.5	-9	-8.5	-8
Roots	-2.162e+31	-2.097e+31	-2.031e+31	-1.965e+31	-1.9e+31

Table 2.2: Table continues below

Initial Values	-7.5	-7	-6.5	-6	-5.5
Roots	-1.834e+31	-1.768e+31	-1.701e+31	-1.635e+31	-1.569e+31

Table 2.3: Table continues below

Initial Values	-5	-4.5	-4	-3.5	-3
Roots	-1.502e+31	-1.436e+31	-1.369e+31	-1.302e+31	-1.235e+31

Table 2.4: Table continues below

Initial Values	-2.5	-2	-1.5	-1	-0.5
Roots	-1.167e+31	-1.1e+31	-1.032e+31	-9.648e+30	-8.97e+30

Table 2.5: Table continues below

Initial Values	0	0.5	1	1.5	2
Roots	-8.295e+30	-7.623e+30	-6.962e+30	-6.327e+30	-5.76e+30

Table 2.6: Table continues below

Initial Values	2.5	3	3.5	4	4.5	5	5.5
Roots	-5.397e+30	-5.999e+30	7.515e+31	21.08	19.38	5.69	5.69

Table 2.7: Table continues below

Initial Values	6	6.5	7	7.5	8	8.5	9
Roots	5.69	5.69	1.608e+31	-4.779e+30	20.56	19.38	5.69

Table 2.8: Table continues below

Initial Values	9.5	10	10.5	11	11.5	12	12.5
Roots	-4.38e+30	19.38	5.69	3.72e+30	2.215e+31	21.08	21.08

Table 2.9: Table continues below

Initial Values	13	13.5	14	14.5	15	15.5	16	16.5
Roots	21.08	19.38	19.38	19.38	20.56	21.08	21.08	20.56

Initial Values	17	17.5	18	18.5	19	19.5	20
Roots	20.56	20.56	-1.413e+30	21.08	21.08	21.08	21.08

```
ggplot(res, aes(x = init, y = root)) + geom_point() +
scale_x_continuous(breaks = round(seq(min(res$init), max(res$init), by = 1),1)) +
labs(x = "Initial Values", y = "Roots from Newton-Raphson") +
theme(plot.title = element_text(hjust = 0.5)) +
ggtitle("Scatter Plot of Roots vs. Initial Values")
```

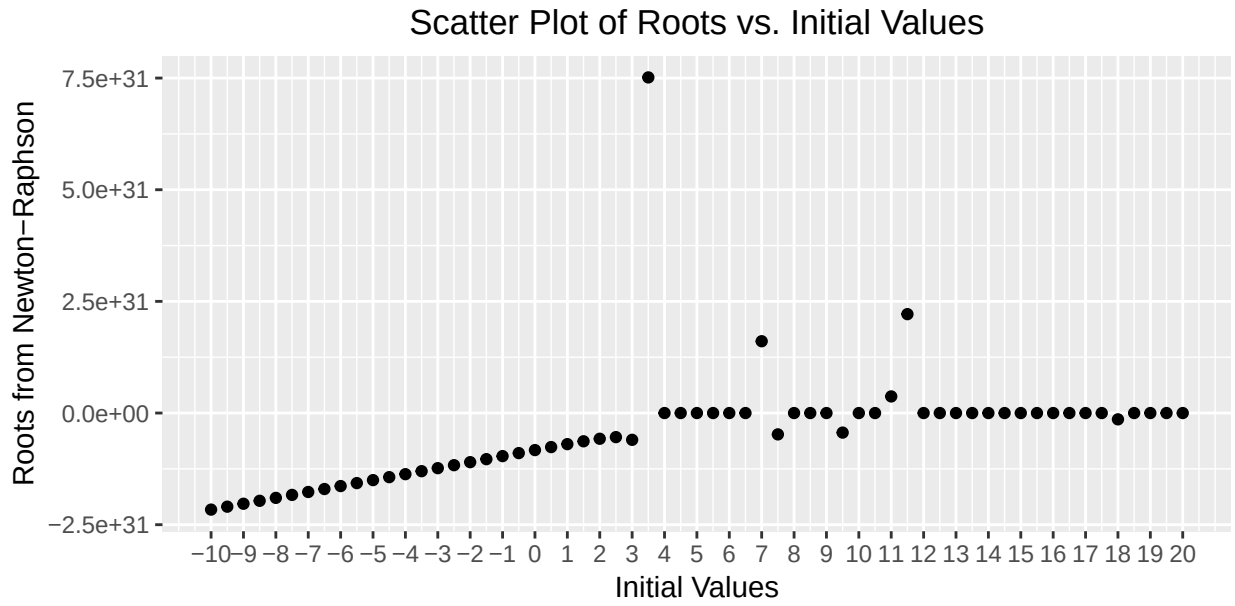

Figure 2.1: Scatter Plot of Roots vs. Initial Values

Chapter 3

Fixed-Point Iterations

From the plots below, we can see when the α decreases, the outcomes become better.

```
set.seed(20180909)
sample <- rcauchy(10, 5)

dev1_log_sum <- function(x){
  dev1_log_sum <- 0
  for (i in 1:length(sample)) {
    dev1_log_sum <- dev1_log_sum - 2 * (x - sample[i])/(1 + (x - sample[i])^2)
  }
  dev1_log_sum
}

fixed.point <- function(init, fun, alpha, maxiter = 100, tol = .Machine$double.eps^0.2){
  x <- init
  for (i in 1:maxiter) {
    x1 <- alpha * fun(x) + x
    if(abs(x1 - x) < tol) break
    x <- x1
  }
  if(i == maxiter)
    message("Reached the maximum iteration!")

  return(data.frame(root = x1, iter = i))
}

init <- seq(-10, 20, by = 0.5)

res1 <- data.frame(init = init, root = rep(NA, length(init)))
res2 <- data.frame(init = init, root = rep(NA, length(init)))
res3 <- data.frame(init = init, root = rep(NA, length(init)))

for (i in 1:length(init)) {
  res1$root[i] <- fixed.point(init[i], dev1_log_sum, 1)$root
}
for (i in 1:length(init)) {
  res2$root[i] <- fixed.point(init[i], dev1_log_sum, 0.64)$root
}
```

```
for (i in 1:length(init)) {
  res3$root[i] <- fixed.point(init[i], dev1_log_sum, 0.25)$root
}

library("ggplot2")
library("gridExtra")

p1 <- ggplot(res1, aes(x = init, y = root)) + geom_point() +
scale_x_continuous(breaks = round(seq(min(res1$init), max(res1$init), by = 1),1)) +
labs(x = "Initial Values", y = "Roots from Fixed-Point") +
theme(plot.title = element_text(hjust = 0.5)) +
ggtitle(expression("Scatter Plot of Roots vs. Initial Values for"~alpha == 1))
p2 <- ggplot(res2, aes(x = init, y = root)) + geom_point() +
scale_x_continuous(breaks = round(seq(min(res2$init), max(res2$init), by = 1),1)) +
labs(x = "Initial Values", y = "Roots from Fixed-Point") +
theme(plot.title = element_text(hjust = 0.5)) +
ggtitle(expression("Scatter Plot of Roots vs. Initial Values for"~alpha == 0.64))
p3 <- ggplot(res3, aes(x = init, y = root)) + geom_point() +
scale_x_continuous(breaks = round(seq(min(res3$init), max(res3$init), by = 1),1)) +
labs(x = "Initial Values", y = "Roots from Fixed-Point") +
theme(plot.title = element_text(hjust = 0.5)) +
ggtitle(expression("Scatter Plot of Roots vs. Initial Values for"~alpha == 0.25))

grid.arrange(p1, p2, p3, nrow = 3)
```

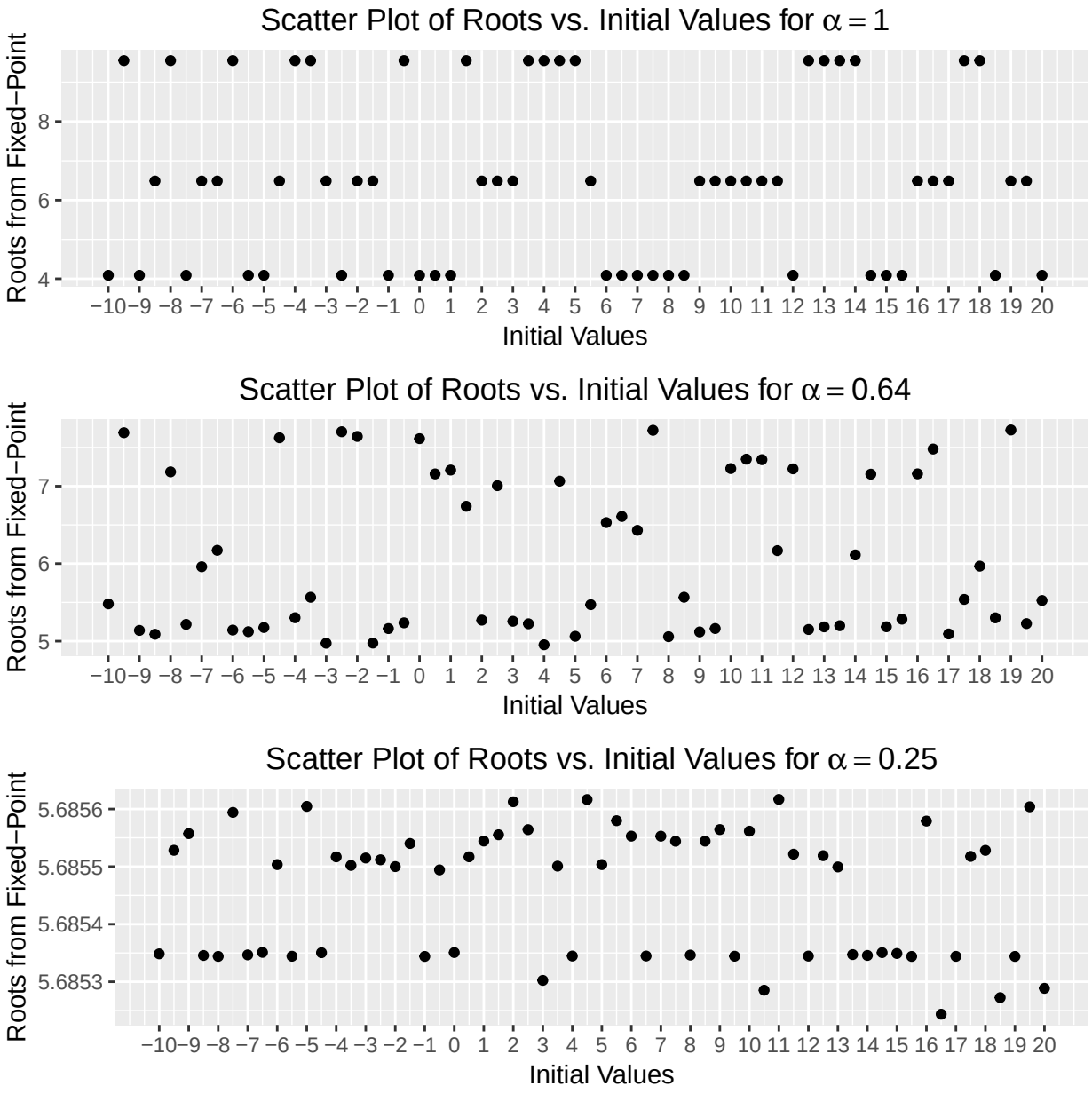

Figure 3.1: Scatter Plot of Roots vs. Initial Values for Different Scaling Choices α

Chapter 4

Fisher Scoring Method and Newton-Raphson Method

From the graph below we can see, by combining Fisher scoring and Newton-Raphson together, the outcomes are quite robust no matter how we choose the initial values comparing with using Newton-Raphson alone.

```
set.seed(20180909)
sample <- rcauchy(10, 5)

dev1_log_sum <- function(x){
  dev1_log_sum <- 0
  for (i in 1:length(sample)) {
    dev1_log_sum <- dev1_log_sum - 2 * (x - sample[i])/(1 + (x - sample[i])^2)
  }
  dev1_log_sum
}

dev2_log_sum <- function(x){
  dev2_log_sum <- 0
  for (i in 1:length(sample)) {
    dev2_log_sum <- dev2_log_sum - 2 * (1 - (x - sample[i])^2)/(1 + (x - sample[i])^2)^2
  }
  dev2_log_sum
}

I_n <- 10/2


fisher.scoring <- function(init, fun, I_n, maxiter = 100, tol = .Machine$double.eps^0.2){
  x <- init
  for (i in 1:maxiter) {
    x1 <- x + fun(x)/I_n
    if(abs(x1 - x) < tol) break
    x <- x1
  }
  if(i == maxiter)
    message("Reached the maximum iteration!")

  return(data.frame(root = x1, iter = i))
```

```
}


newton.raphson <- function(init, fun, fun.dev, maxiter = 100, tol = .Machine$double.eps^0.2){
  x <- init
  for (i in 1:maxiter) {
    x1 <- x -fun(x)/fun.dev(x)
    if(abs(x1 - x) < tol) break
    x <- x1
  }
  if(i == maxiter)
    message("Reached the maximum iteration!")

  return(data.frame(root = x1, iter = i))
}

init <- seq(-10, 20, by = 0.5)
res <- data.frame(init = init, root = rep(NA, length(init)))
for (i in 1:length(init)) {
  res$root[i] <- newton.raphson(fisher.scoring(init[i], dev1_log_sum, I_n)$root,
                                dev1_log_sum, dev2_log_sum)$root
}

library("ggplot2")
options(digits = 8)
ggplot(res, aes(x = init, y = root)) + geom_point() +
scale_x_continuous(breaks = round(seq(min(res$init), max(res$init), by = 1),1)) +
labs(x = "Initial Values", y = "Roots from Fisher Scoring and Newton-Raphson") +
theme(plot.title = element_text(hjust = 0.5)) +
ggtitle("Scatter Plot of Roots vs. Initial Values")
```

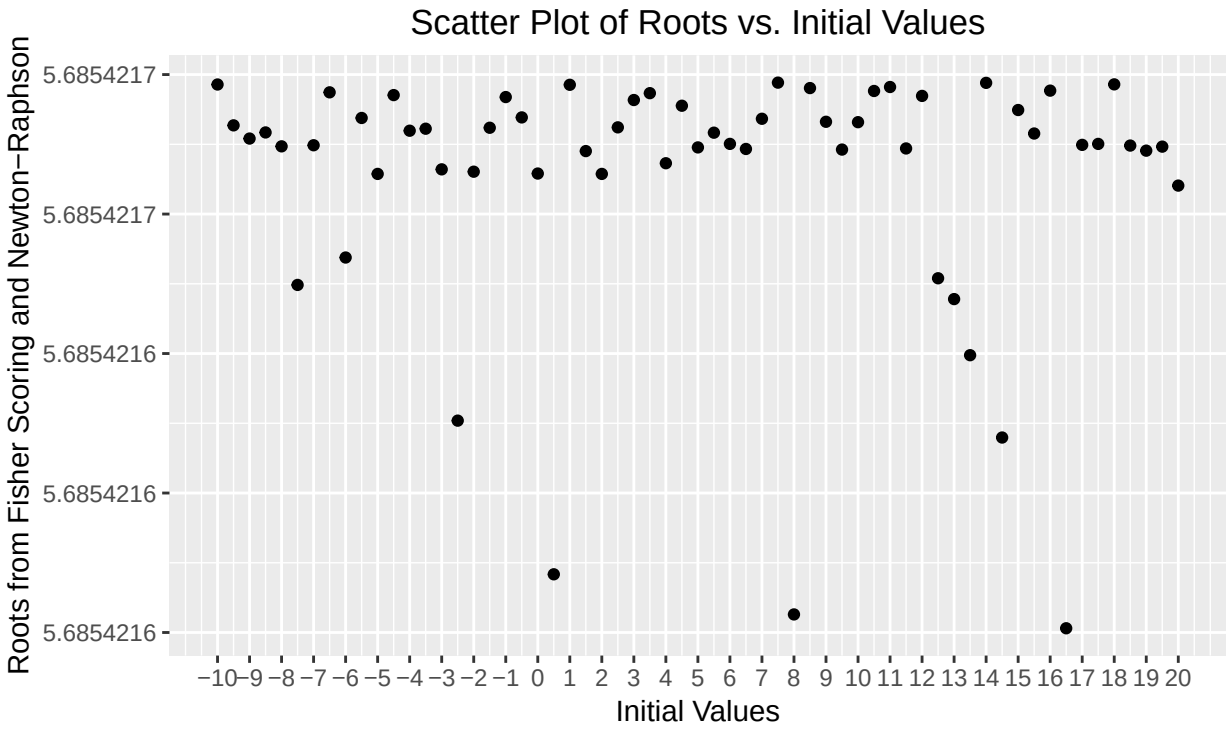

Figure 4.1: Scatter Plot of Roots vs. Initial Values

Chapter 5

Comments

From the results above we can see Newton-Raphson method is the most unstable one since initial values affect the results a lot. While Fishering Scoring refined by Newton-Raphson is the most stable one.
In order to compare the speed, we set initial value as $3, 5, 7$ for these three algorithms. For Fixed-Point method, since the results are best when the scaling parameter $\alpha = 0.25$. Under such condition, we have the following outcomes. In the Fisher Scoring and Newton-Raphson column, the first value represents the root and iteration times from Fishering Score method, the second value means the root and iterations from Newton-Raphson method after we use the root from Fisher Scoring as the initial value for it. It's easy to see the last algorithm is not only the most stable one, but also the fastest one.

Table 5.1: Roots and Iteration times for Different Algorithms When Initial Value is 3

	Newton-Raphson	Fixed-Point	Fisher Scoring and Newton-Raphson
Roots	-5.999e+30	5.685	5.685_5.685
Iteration Times	100	7	6_1

Table 5.2: Roots and Iteration times for Different Algorithms When Initial Value is 5

	Newton-Raphson	Fixed-Point	Fisher Scoring and Newton-Raphson
Roots	5.685	5.686	5.685_5.685
Iteration Times	5	7	4_1

Table 5.3: Roots and Iteration times for Different Algorithms When Initial Value is 7

	Newton-Raphson	Fixed-Point	Fisher Scoring and Newton-Raphson
Roots	1.608e+31	5.686	5.685_5.685
Iteration Times	100	8	6_1