Project Report: Finding Beta Value in Weibull Function using Newton-Raphson Method

Environment – R language

Code

```{r}
g <- function(b, x) {
  n <- length(x)
  sum1 <- sum(sapply(x, function(i) { i^b * log(i) }))
  sum2 <- sum(sapply(x, function(i) { i^b }))
  sum3 <- sum(sapply(x, function(i) { log(i) }))
  return((sum1 / sum2) - (1 / b) - (sum3 / n))
}

```

In R, we use sapply to apply a function to each element of the vector x. The ^ operator is used for exponentiation, and log() calculates the natural logarithm of each element in the vector. The rest of the code is quite similar to the original Python function.

```{r}
dg <- function(b, x) {
  n <- length(x)
  sum1 <- sum(sapply(x, function(i) { i^b * log(i) * log(i) }))
  sum2 <- sum(sapply(x, function(i) { i^b * log(i) }))
  sum3 <- sum(sapply(x, function(i) { i^b }))
  return((sum1 / sum2) - (1 / (b^2)) - (sum2 / sum3))
}

```

```{r}
newton_raphson <- function(x, initial_guess, tolerance = 1e-6, max_iterations = 100) {
  n <- length(x)
  b <- initial_guess

  for (i in 1:max_iterations) {
    f <- g(b, x)
    df <- dg(b, x)
    delta <- f / df
    b <- b - delta

    if (abs(delta) < tolerance) {
      return(b)
    }
  }

  return(NULL)
}

```

```{r}
# Example usage
x <- c(1, 2, 3, 4, 5, 11)  # Sample array of x axis values
initial_guess <- 1  # Initial guess for b
b <- newton_raphson(x, initial_guess)
cat("The value of b is:", b, "\n")
```