Experiment No: 09

AIM: To simulate a linear model in R

Objectives:

- ➔ Students will be able to implement a Program Using R to generate random numbers
- ➔ Students will learn Random Sampling and simulation in R

Software Requirements: RStudio, VSCode

Theory:

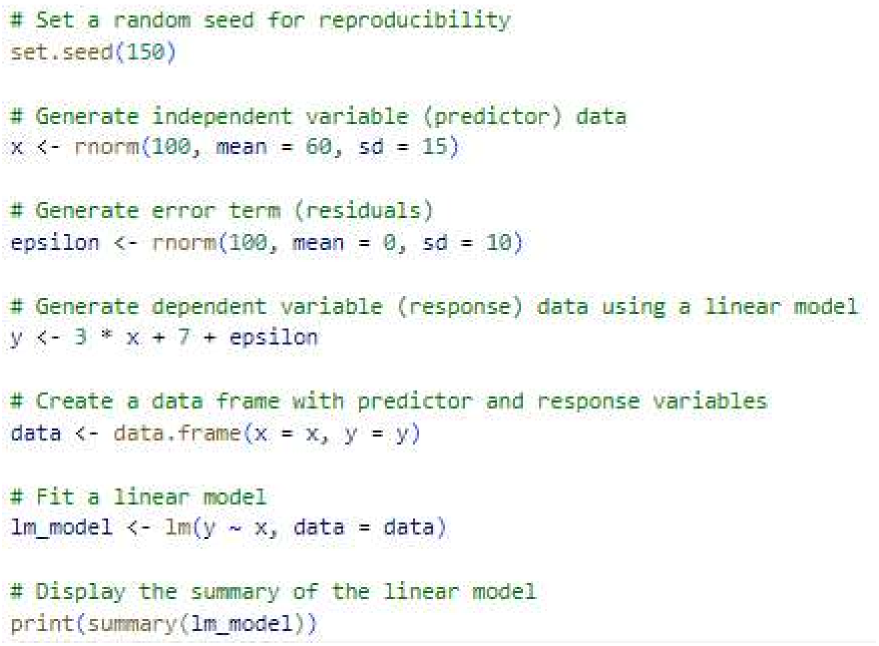
A linear model is a statistical approach used to describe the relationship between a dependent variable and one or more independent variables by assum ing a linear relationship between them. In simpler terms, it assumes that changes in the independent variables lead to proportional changes in the dependent variable. The general form of a linear model can be represented as Y = ßO + ß1X1 + ß2)Q + + E, where Y is the dependent variable, Xl, X2, etc., are independent variables, PO, etc., are coefficients representing the effect of each independent variable, and E is the error term accounting for unexplained variability.

R provides various packages like 'lm' (linear model), 'glm' (generalized linear model), and 'caret' (classification and regression training) that facilitate the creation and evaluation of linear models. The 'lm' function in R is particularly useful for fitting linear regression models. For instance, to create a simple linear model in R using the 'lm' function, you can use syntax like lm(Y ~ X1 + X2, data = dataset), where Y is the dependent variable and X1, X2 are independent variables from your dataset. R also offers extensive functionalities for model diagnostics, including checking assumptions like linearity, normality of residuals, homoscedasticity, and multicollinearity, making it a comprehensive tool for linear modeling and analysis.

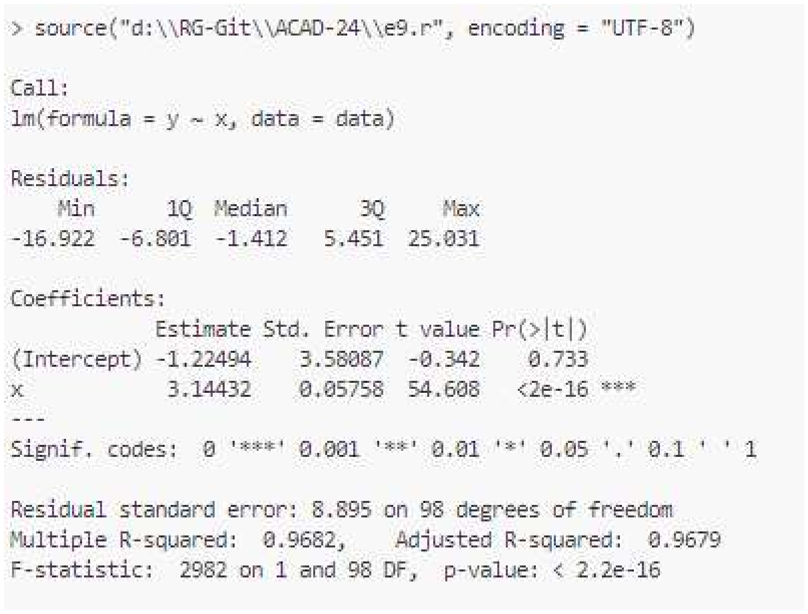
Advantages of Linear Modeling in R:

1. Easy Implementation
2. Comprehensive Model Diagnostics
3. Flexible Model Specification
4. Rich Visualization Capabilities
5. Integration with standard Packages

Implementation (Code and Output):



```
# Set a random seed for reproducibility
set.seed(150)

# Generate independent variable (predictor) data
x <- rnorm(100, mean = 60, sd = 15)

# Generate error term (residuals)
epsilon <- rnorm(100, mean = 0, sd = 10)

# Generate dependent variable (response) data using a linear model
y <- 3 * x + 7 + epsilon

# Create a data frame with predictor and response variables
data <- data.frame(x = x, y = y)

# Fit a linear model
lm_model <- lm(y ~ x, data = data)

# Display the summary of the linear model
print(summary(lm_model))
```



```
> source("d:\\RG-Git\\ACAD-24\\e9.r", encoding = "UTF-8")

Call:
lm(formula = y ~ x, data = data)

Residuals:
    Min      1Q  Median      3Q     Max 
-16.922  -6.801  -1.412   5.451  25.031 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -1.22494    3.58087  -0.342    0.733    
x            3.14432    0.05758  54.608   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8.895 on 98 degrees of freedom
Multiple R-squared:  0.9682,	Adjusted R-squared:  0.9679 
F-statistic:  2982 on 1 and 98 DF,  p-value: < 2.2e-16
```

Conclusion: In this way, using linear modeling in R is really helpful. R makes it easy to create and check these models, which show how different things are connected. You can make these models flexible to understand complex relationships better, and R also helps to visualize the data in ways that are easy to understand. Plus, it works with other tools that make the analysis even more powerful, so researchers can trust the results and use them to make smart decisions based on their experiments. Overall, using linear modeling in R is a great way to get clear insights from experimental data.