Inroduction to R

Sangbaptist

2024-10-05

Table of contents

Introduction to R

R is a programming language and software environment primarily used for statistical computing, data analysis, and graphical representation. It was developed as an open-source alternative to the S language.

Pioneers of R

Creators: R was created by `Ross Ihaka` and `Robert Gentleman`, two statisticians from the `University of Auckland, New Zealand`.

R was first released in `1993`, with its stable version `1.0.0` coming out in `2000`. It was inspired by the S programming language, developed by `John Chambers` at `Bell Labs`.

R was created to provide a free, open-source platform for statisticians, data analysts, and researchers to perform statistical analysis, manipulate data, and create visualizations. Its purpose is to make it easier to apply complex statistical methods and work with large data sets.

```
# Load the gt package
library(gt)

# Create the data for the table
comparison <- data.frame(
  Feature = c("Primary Use", "Learning Curve", "Cost", "Community & Libraries",
              "Data Visualization", "Speed & Efficiency", "Integration", "Scalability"),
  R = c("Statistical computing, data analysis, and visualization",
        "Steep for beginners, especially for those unfamiliar with statistical concepts",
        "Free and open source",
        "Large, focused on statistics, 18,000+ packages",
        "Excellent (with `ggplot2`, `plotly`, etc.)",
        "Slower, memory-intensive",
        "Excellent (interfaces with Python, C++, SQL)",
        "Limited in enterprise or high-performance settings"),
  Python = c("General-purpose programming, data analysis, machine learning",
             "Easier for beginners due to simpler syntax and versatility",
             "Free (for basic use, with libraries)",
             "Huge, versatile, with over 450,000 packages",
             "Good (with `matplotlib`, `seaborn`, `plotly`)",
             "Faster, more efficient for large datasets",
             "Excellent (integrates with R, Java, C++)",
             "Highly scalable, used widely in production environments"),
  SAS = c("Statistical analysis and business intelligence",
          "Steep but well-supported for statistical analysis",
          "Expensive commercial license",
          "Specialized community, closed source",
          "Limited compared to R",
          "Fast, optimized for statistical procedures",
          "Limited to SAS ecosystem",
          "Well-suited for large-scale applications in business"),
  MATLAB = c("Numerical computing, engineering, and data analysis",
             "Steep, especially for advanced mathematical operations",
             "Expensive commercial license",
             "Focused academic and engineering community",
             "Good, though not as intuitive as R",
             "Optimized for matrix operations but slower for general programming",
```

```
            "Can integrate with Python, R, C++, but requires MATLAB-specific licenses",
            "Can be scaled with MATLAB Parallel Server")
)

# Generate the table using gt
gt_table <- gt(comparison, rowname_col = "Feature") %>%
  tab_header(
    title = "Comparison between R, Python, SAS, and MATLAB"
  ) %>%
  tab_style(
    style = list(
      cell_text(weight = "bold") # Set the header text to bold
    ),
    locations = cells_column_labels(everything()) # Apply style to all column headers
  ) %>%
  tab_style(
    style = list(
      cell_text(weight = "bold", color = "blue") # Set the row names text to bold
    ),
    locations = cells_stub() # Apply style to the row names (stub)
  )

# Render the table
gt_table
```

	R	P
Primary Use	Statistical computing, data analysis, and visualization	G
Learning Curve	Steep for beginners, especially for those unfamiliar with statistical concepts	E
Cost	Free and open source	F
Community & Libraries	Large, focused on statistics, 18,000+ packages	H
Data Visualization	Excellent (with ‘ggplot2‘, ‘plotly‘, etc.)	G
Speed & Efficiency	Slower, memory-intensive	F
Integration	Excellent (interfaces with Python, C++, SQL)	E
Scalability	Limited in enterprise or high-performance settings	H

Advantages of R

1. Open Source: R is free to use, modify, and distribute, which makes it accessible to everyone.
2. Comprehensive Statistical Packages: R has over 18,000 packages (available on CRAN) covering a wide range of statistical and machine learning techniques, making it a go-to tool for statisticians and data scientists.
3. Powerful Data Visualization: R excels in data visualization with libraries like ggplot2, plotly, and shiny, allowing users to create high-quality plots and interactive web apps.
4. Active Community: R has a large, supportive user community and strong contributions from statisticians and data scientists worldwide. The R user community regularly contributes new packages and offers extensive help through forums and mailing lists.
5. Cross-platform Compatibility: R works on various operating systems, including Windows, macOS, and Linux.
6. Integration with Other Languages: R can interface with other languages like Python, C++, and Java, making it more flexible in projects that require multiple programming environments.

Disadvantages of R

1. Slow Execution: R is an interpreted language, and sometimes it runs slower than compiled languages like C++ or Java, especially when handling large datasets or complex operations.
2. Steep Learning Curve: R's syntax and object-oriented structure can be difficult to master for beginners. Concepts like data frames, functions, and vectorization are key parts of the language, but not always intuitive to new users.
3. Memory Usage: R processes all objects in-memory, which can be limiting when dealing with very large datasets. Unlike languages like Python or SQL, which can handle out-of-memory computations more efficiently, R struggles with memory optimization.
4. Poor Scalability: R isn't the best choice for production environments or enterprise applications due to its lack of scalability in handling massive, high-frequency data.
5. Less User-Friendly for Some Tasks: Some tasks, especially general-purpose programming, can be cumbersome in R compared to more general languages like Python or JavaScript.

Comparison Highlights:

- R vs Python: Both are popular in data science, but Python is more versatile as a general-purpose language. R has an edge in statistical analysis and data visualization, while Python excels in machine learning and web development.
- R vs SAS: SAS is widely used in the corporate world for business intelligence and statistical analysis but is expensive. R is free and flexible but not as supported in large enterprise environments.

-R vs MATLAB: MATLAB is more commonly used in engineering and numerical computing, whereas R is preferred for statistics. MATLAB is expensive, while R is free and open-source.

Summary

R is a powerful tool for statistical computing and visualization, with a vast ecosystem of packages tailored for data analysis and research. While it has some limitations in terms of speed, memory usage, and scalability, its extensive statistical functions and strong community make it a preferred choice in academic and research settings.