**File I/O In R programming**

Md. Neaz Ali

M.Sc in Statistics

Department of Statistics

Islamic University, Kushtia – 7003

Bangladesh

# 1. Reading & Writing Data Files

|  |  |  |  |
| --- | --- | --- | --- |
| **File Type** | **Read Function** | **Write Function** | **Package(s)** |
| CSV (.csv) | read.csv("file.csv") | write.csv(df, "file.csv") | base R, readr |
| Text (.txt) | read.table("file.txt", header=TRUE) | write.table(df, "file.txt") | base R, readr |
| Excel (.xlsx) | readxl::read\_excel("file.xlsx") | writexl::write\_xlsx(df, "file.xlsx") | readxl, writexl |
| RData (.RData) | load("file.RData") | save(df, file="file.RData") | base R |
| RDS (.rds) | readRDS("file.rds") | saveRDS(df, "file.rds") | base R |

# 2. Useful Packages for Data I/O

## **readr**: Fast reading of CSV and TXT files

|  |
| --- |
| readr::read\_csv("file.csv")  readr::write\_csv(df, "file.csv") |

## **readxl**: Reading Excel files

|  |
| --- |
| readxl::read\_excel("file.xlsx") |

## **writexl**: Writing Excel files

|  |
| --- |
| writexl::write\_xlsx(df, "file.xlsx") |

## **data.table**: Fast reading and writing, especially large data

|  |
| --- |
| data.table::fread("file.csv")  data.table::fwrite(df, "file.csv") |

## **rio**: A unified interface for importing/exporting many formats

|  |
| --- |
| rio::import("file.xlsx")  rio::export(df, "file.xlsx") |

# 3. Working with File Paths & Directories

## Get current working directory

|  |
| --- |
| getwd() |

## Set working directory

|  |
| --- |
| setwd("path/to/your/folder") |

## Construct file path (platform-independent)

|  |
| --- |
| file.path("folder", "subfolder", "file.csv") |

# 4. Example Code Snippets

|  |
| --- |
| # Read CSV with readr  library(readr)  df <- read\_csv("data/my\_data.csv")  # Write data frame to Excel with writexl  library(writexl)  write\_xlsx(df, "output/my\_data.xlsx")  # Load RData file  load("data/my\_saved\_data.RData")  # Save object as RDS  saveRDS(df, "data/my\_data.rds")  # Set working directory  setwd("/Users/yourname/Documents/RProjects") |

# 5. Sink()

Redirect R output (like console messages, print outputs, summaries) to a file instead of the console.

|  |
| --- |
| # Start redirecting output to a file  sink("output.txt")  # Any output printed here goes to output.txt  print(summary(mtcars))  cat("This text goes into the file.\n")  # Stop redirecting output; return output to console  sink() |

**10 Simple Problems to Practice**

1. Read a CSV file named students.csv into R using **base R** and **readr** packages.
2. Write a data frame df to a text file output.txt with tab-delimited columns.
3. Read an Excel file sales\_data.xlsx into R using the **readxl** package.
4. Save the built-in dataset mtcars as an RDS file named mtcars\_data.rds.
5. Load an RData file my\_workspace.RData that contains several objects.
6. Use data.table package to read a large CSV file named bigdata.csv.
7. Export a data frame df to an Excel file report.xlsx using **rio** package.
8. Change your current working directory to "C:/Users/YourName/Documents".
9. Combine folder and file name to create a path to "data/file.csv" using file.path().
10. Read a tab-separated text file log.txt without a header into R.

# 5. Advanced Topics

### 1. Handling Compressed Files

**Read compressed CSV files (.gz, .bz2, .zip)**  
Using **data.table** or **readr** (automatic decompression):

|  |
| --- |
| data <- data.table::fread("data.csv.gz")  data <- readr::read\_csv("data.csv.bz2") |

## Manually unzip and read

|  |
| --- |
| utils::unzip("archive.zip", exdir = "folder")  df <- read.csv("folder/data.csv") |

### 2. Reading/Writing JSON Files

## Using jsonlite package

|  |
| --- |
| library(jsonlite)  # Read JSON file  data <- fromJSON("data.json")  # Write JSON file  toJSON(data, pretty = TRUE, auto\_unbox = TRUE) %>%  writeLines("output.json") |

### 3. Reading/Writing XML Files

|  |
| --- |
| library(xml2)  # Read XML  xml\_data <- read\_xml("data.xml")  # Extract content  texts <- xml\_text(xml\_find\_all(xml\_data, "//tagname")) |

### 4. Database Connections

## Using DBI and RSQLite

|  |
| --- |
| library(DBI)  library(RSQLite)  # Connect to SQLite database  con <- dbConnect(RSQLite::SQLite(), "my\_database.sqlite")  # List tables  dbListTables(con)  # Read a table  df <- dbReadTable(con, "my\_table")  # Write a data frame as a new table  dbWriteTable(con, "new\_table", df)  # Query with SQL  res <- dbGetQuery(con, "SELECT \* FROM my\_table WHERE column > 100")  # Disconnect  dbDisconnect(con) |

### 5. Reading/Writing from URLs and APIs

## Read CSV directly from a URL

|  |
| --- |
| df <- read.csv("https://example.com/data.csv") |

Download files from web

|  |
| --- |
| download.file("https://example.com/data.csv", destfile = "data.csv") |

## API call and parse JSON

|  |
| --- |
| library(httr)  library(jsonlite)  res <- httr::GET("https://api.example.com/data")  json\_data <- content(res, "text")  data <- fromJSON(json\_data) |

### 6. Parallel and Chunked File Reading for Large Files

## Using **data.table::fread** for fast reading

|  |
| --- |
| bigdata <- data.table::fread("large\_file.csv") |

## Reading large files in chunks using **readr**

|  |
| --- |
| library(readr)  chunk\_callback <- function(df, pos) {  print(paste("Read chunk starting at", pos))  # Process df chunk here  }  read\_csv\_chunked("large\_file.csv", DataFrameCallback$new(chunk\_callback), chunk\_size = 1000) |

### 7. Saving and Loading R Objects with Compression

## Save R objects compressed with save()

|  |
| --- |
| save(df, file = "data.RData", compress = "gzip") |

## Read compressed RDS files

|  |
| --- |
| df <- readRDS("data\_compressed.rds") |