

Capacity Building in Seasonal Hydrological Forecasting

Working with Tabular Data

AGRHYMET, Climate Regional Center for West-Africa and Sahel

@Arsène KIEMA

2025-10-06



Pedagogical Objectives

Learning outcomes

By the end of this module, participants will be able to:

- Import tabular data from CSV, Excel, and databases.
- Explore and clean datasets (missing values, duplicates, formatting).
- Manipulate tabular data with `dplyr` (select, filter, group, summarize).
- Combine datasets using joins and binding functions.
- Export processed data to different formats.
- Apply these skills to a hydrological case study (precipitation and flow).

Importing Data

R has simple built-in functions to read tabular data: - `read.csv("file.csv")` → import a CSV file (comma separated). - `read.table("file.txt")` → more general, can specify separator (tab, semicolon, etc.).



Tip

Always check the working directory with `getwd()` and set it with `setwd("path")`.

Importing Data

```
# Import a CSV file
flows <- read.csv("data/flows_station.csv")

# Explore
head(flows)      # first 6 rows
tail(flows)      # last 6 rows
str(flows)       # structure
summary(flows)   # summary statistics
```

R does not have built-in Excel support → we use the readxl package.

Cleaning and Exploring Data

Common data issues in hydrology: - Missing values (e.g., station not measuring).
- Wrong data types (dates as text). - Outliers (negative precipitation). -
Duplicated records.

Tools: - `is.na()`, `na.omit()`, `replace_na()` - `lubridate` for date/time
handling - `distinct()` to remove duplicates

Data Manipulation with dplyr

Core dplyr verbs: - `select()` → choose columns - `filter()` → filter rows - `mutate()` → create/transform variables - `arrange()` → sort data - `group_by()` + `summarise()` → aggregation

Data Manipulation with dplyr

```
# Example: monthly flow statistics
monthly <- flows |>
  mutate(Month = month(Date, label = TRUE),
         Year = year(Date)) |>
  group_by(Year, Month) |>
  summarise(
    Qmean = mean(Qobs, na.rm = TRUE),
    Qmax = max(Qobs, na.rm = TRUE),
    n = n()
  )
head(monthly)
```

Combining Datasets (Joins)

Hydrological analysis often requires merging datasets:

- **Inner join:** keep only matching records.



Combining Datasets (Joins)

Hydrological analysis often requires merging datasets:

- **Inner join:** keep only matching records.
- **Left join:** keep all rows from left table.



Combining Datasets (Joins)

Hydrological analysis often requires merging datasets:

- **Inner join:** keep only matching records.
- **Left join:** keep all rows from left table.
- **Right join:** keep all rows from right table.



Combining Datasets (Joins)

Hydrological analysis often requires merging datasets:

- **Inner join:** keep only matching records.
- **Left join:** keep all rows from left table.
- **Right join:** keep all rows from right table.
- **Full join:** keep all rows from both.



Combining Datasets (Joins)

```
# Example: join flows with rainfall data  
data <- flows |>  
  left_join(rain, by = "Date")  
head(data)
```

Reshaping and Transforming Tables

In hydrology, datasets often come in **wide format** (many columns for stations or variables).

But for analysis and visualization, it is usually easier to work in **long format** (“tidy data”).

- **Wide format:** one column per variable/station.

The `tidyr` package provides tools for reshaping.

Reshaping and Transforming Tables

In hydrology, datasets often come in **wide format** (many columns for stations or variables).

But for analysis and visualization, it is usually easier to work in **long format** (“tidy data”).

- **Wide format:** one column per variable/station.
- **Long format:** one row per observation, with columns Date, Variable, Value.

The `tidyr` package provides tools for reshaping.

Reshaping and Transforming Tables

From Wide to Long

```
library(tidyr)

# Example dataset: daily flows for 3 stations
flows_wide <- data.frame(
  Date = as.Date("2020-01-01") + 0:4,
  Station_A = c(12, 15, 18, 10, 20),
  Station_B = c(22, 25, 19, 30, 28),
  Station_C = c(5, 8, 6, 7, 10)
)

flows_wide
```

```
# Convert to long format
```

Reshaping and Transforming Tables

From Long to Wide

```
# Convert to long format
flows_long <- flows_wide |>
  pivot_longer(cols = starts_with("Station"),
               names_to = "Station",
               values_to = "Flow")

flows_long
```


Exporting Data

To save processed results: - `write.csv(data, "file.csv",
row.names=FALSE)` - `write.table(data, "file.txt", sep="\t",
row.names=FALSE)`

Practical Exercises

Exercise 1

Import the file `precipitation_station.csv`.

- Count the number of missing values.
- Replace missing precipitation with 0.
- Compute annual precipitation totals.

Exercise 2

Join precipitation and flow datasets.

- Compute correlation between annual precipitation and annual mean flow.

Exercise 3

Export the final dataset to `outputs/hydro_annual.csv`.

Wrap-up

! Key takeaways

- Use **base R functions**: `read.csv()`, `read.table()`, `write.csv()`, `write.table()`.

Wrap-up

! Key takeaways

- Use **base R functions**: `read.csv()`, `read.table()`, `write.csv()`, `write.table()`.
- Use `readxl::read_excel()` for Excel files.

! Key takeaways

- Use **base R functions**: `read.csv()`, `read.table()`, `write.csv()`, `write.table()`.
- Use `readxl::read_excel()` for Excel files.
- Always **explore** (`head`, `str`, `summary`) before analysis.

! Key takeaways

- Use **base R functions**: `read.csv()`, `read.table()`, `write.csv()`, `write.table()`.
- Use `readxl::read_excel()` for Excel files.
- Always **explore** (`head`, `str`, `summary`) before analysis.
- Handle missing values with `na.omit()` or `na.rm = TRUE`.

! Key takeaways

- Use **base R functions**: `read.csv()`, `read.table()`, `write.csv()`, `write.table()`.
- Use `readxl::read_excel()` for Excel files.
- Always **explore** (`head`, `str`, `summary`) before analysis.
- Handle missing values with `na.omit()` or `na.rm = TRUE`.
- **Long format (tidy data)** is better for analysis and visualization.

! Key takeaways

- Use **base R functions**: `read.csv()`, `read.table()`, `write.csv()`, `write.table()`.
- Use `readxl::read_excel()` for Excel files.
- Always **explore** (`head`, `str`, `summary`) before analysis.
- Handle missing values with `na.omit()` or `na.rm = TRUE`.
- **Long format (tidy data)** is better for analysis and visualization.
- Use `pivot_longer()` and `pivot_wider()` from `tidyr` for reshaping.

! Key takeaways

- Use **base R functions**: `read.csv()`, `read.table()`, `write.csv()`, `write.table()`.
- Use `readxl::read_excel()` for Excel files.
- Always **explore** (`head`, `str`, `summary`) before analysis.
- Handle missing values with `na.omit()` or `na.rm = TRUE`.
- **Long format (tidy data)** is better for analysis and visualization.
- Use `pivot_longer()` and `pivot_wider()` from `tidyr` for reshaping.
- Save results with `write.csv()` to ensure reproducibility.

Mini-project of Day

Context

We want to combine rainfall and flow data to create a hydrological summary table.

Tasks

- 1 Import daily rainfall and flow data.

Mini-project of Day

Context

We want to combine rainfall and flow data to create a hydrological summary table.

Tasks

- 1 Import daily rainfall and flow data.
- 2 Clean datasets (dates, missing values, duplicates).

Mini-project of Day

Context

We want to combine rainfall and flow data to create a hydrological summary table.

Tasks

- 1 Import daily rainfall and flow data.
- 2 Clean datasets (dates, missing values, duplicates).
- 3 Aggregate by year: - Annual rainfall (sum), - Annual mean flow, - Annual maximum flow.

Mini-project of Day

Context

We want to combine rainfall and flow data to create a hydrological summary table.

Tasks

- 1 Import daily rainfall and flow data.
- 2 Clean datasets (dates, missing values, duplicates).
- 3 Aggregate by year: - Annual rainfall (sum), - Annual mean flow, - Annual maximum flow.
- 4 Join rainfall and flow summaries.

Mini-project of Day

Context

We want to combine rainfall and flow data to create a hydrological summary table.

Tasks

- 1 Import daily rainfall and flow data.
- 2 Clean datasets (dates, missing values, duplicates).
- 3 Aggregate by year: - Annual rainfall (sum), - Annual mean flow, - Annual maximum flow.
- 4 Join rainfall and flow summaries.
- 5 Save the final dataset as `outputs/hydro_summary.csv`.

**THANK YOU FOR YOUR
ATTENTION**

