# Capacity Building in Seasonal Hydrological Forecasting

## Introduction to R and Programming Basics

**AGRHYMET, Climate Regional Center for West-Africa and Sahel**

**@Arsène KIEMA**

2025-10-06

# Pedagogical Objectives

### ℹ Learning outcomes

By the end of this session, you will be able to:

- Describe what R is and where it shines (and where it doesn't).

# Pedagogical Objectives

## Learning outcomes

By the end of this session, you will be able to:

- Describe what R is and where it shines (and where it doesn't).
- Install and configure R, RStudio, and a Miniconda/JupyterLab environment.

# Pedagogical Objectives

ℹ Learning outcomes

By the end of this session, you will be able to:

- Describe what R is and where it shines (and where it doesn't).
- Install and configure R, RStudio, and a Miniconda/JupyterLab environment.
- Run R code in scripts and notebooks.

# Pedagogical Objectives

ℹ Learning outcomes

By the end of this session, you will be able to:

- Describe what R is and where it shines (and where it doesn't).
- Install and configure R, RStudio, and a Miniconda/JupyterLab environment.
- Run R code in scripts and notebooks.
- Prepare your workstation for the rest of the training.

# What is R?

R is a free, open-source programming language (mid-1990s, Ihaka & Gentleman) widely used for **statistics**, **data analysis**, **data visualization**, and **scientific computing**.

# What is R?

### Why scientists like R

- World-class **statistical modeling** and **graphics**.

# What is R?

Why scientists like R

- World-class **statistical modeling** and **graphics**.
- Huge ecosystem: CRAN (20k+ packages), Bioconductor, rOpenSci.

# What is R?

Why scientists like R

- World-class **statistical modeling** and **graphics**.
- Huge ecosystem: CRAN (20k+ packages), Bioconductor, rOpenSci.
- Strong culture of **reproducible research** (Quarto/R Markdown).

# Strengths

- Open-source, cross-platform.

# Strengths

- Open-source, cross-platform.
- Rich packages for **time series** and **spatial / spatio-temporal** analysis (`sf`, `stars`, `terra`).

# Strengths

- Open-source, cross-platform.
- Rich packages for **time series** and **spatial / spatio-temporal** analysis (`sf`, `stars`, `terra`).
- Elegant data manipulation with **tidyverse** (`dplyr`, `tidyr`).

# Strengths

- Open-source, cross-platform.
- Rich packages for **time series** and **spatial / spatio-temporal** analysis (`sf`, `stars`, `terra`).
- Elegant data manipulation with **tidyverse** (`dplyr`, `tidyr`).
- Publication-quality graphics (`ggplot2`) and interactive maps (`leaflet`, `tmap`).

# Why R for Hydrology?

- Mature tooling for **hydrological time series** and **spatial hydrology**.

❗ Key takeaway

R is particularly strong for **exploratory analysis**, **statistics**, **mapping**, and **reproducible workflows** — exactly what we need to build operational hydrological products and training pipelines.

# Why R for Hydrology?

- Mature tooling for **hydrological time series** and **spatial hydrology**.
- Packages you'll see in this training:

! Key takeaway

R is particularly strong for **exploratory analysis**, **statistics**, **mapping**, and **reproducible workflows** — exactly what we need to build operational hydrological products and training pipelines.

# Why R for Hydrology?

- Mature tooling for **hydrological time series** and **spatial hydrology**.
- Packages you'll see in this training:
  - `tidyverse`, `data.table` (data wrangling)

❗ Key takeaway

R is particularly strong for **exploratory analysis**, **statistics**, **mapping**, and **reproducible workflows** — exactly what we need to build operational hydrological products and training pipelines.

# Why R for Hydrology?

- Mature tooling for **hydrological time series** and **spatial hydrology**.
- Packages you'll see in this training:
  - `tidyverse`, `data.table` (data wrangling)
  - `sf`, `terra`, `stars` (GIS & rasters)

**!** Key takeaway

R is particularly strong for **exploratory analysis**, **statistics**, **mapping**, and **reproducible workflows** — exactly what we need to build operational hydrological products and training pipelines.

# Why R for Hydrology?

- Mature tooling for **hydrological time series** and **spatial hydrology**.
- Packages you'll see in this training:
  - `tidyverse`, `data.table` (data wrangling)
  - `sf`, `terra`, `stars` (GIS & rasters)
- `tidymodels`, `airGR`, `HYPEtools`, and **WASS2SHydroR** (our focus)

> ! Key takeaway
>
> R is particularly strong for **exploratory analysis**, **statistics**, **mapping**, and **reproducible workflows** — exactly what we need to build operational hydrological products and training pipelines.

# Installing the R Environment

We support two workflows: **RStudio-centric** and **JupyterLab via Miniconda**. Choose one (you can have both).

## Option A — R + RStudio (simple desktop setup)

1. Install **R** from CRAN: https://cran.r-project.org/

💡 First run checklist

# Installing the R Environment

We support two workflows: **RStudio-centric** and **JupyterLab via Miniconda**. Choose one (you can have both).

## Option A — R + RStudio (simple desktop setup)

1. Install **R** from CRAN: https://cran.r-project.org/
2. Install **RStudio Desktop**: https://posit.co/download/rstudio-desktop/

First run checklist

# Installing the R Environment

We support two workflows: **RStudio-centric** and **JupyterLab via Miniconda**. Choose one (you can have both).

## Option A — R + RStudio (simple desktop setup)

1. Install **R** from CRAN: https://cran.r-project.org/
2. Install **RStudio Desktop**: https://posit.co/download/rstudio-desktop/

### First run checklist

- Open RStudio → *Console* → run `install.packages("tidyverse")`.

# Installing the R Environment

We support two workflows: **RStudio-centric** and **JupyterLab via Miniconda**. Choose one (you can have both).

## Option A — R + RStudio (simple desktop setup)

1. Install **R** from CRAN: https://cran.r-project.org/
2. Install **RStudio Desktop**: https://posit.co/download/rstudio-desktop/

💡 First run checklist

- Open RStudio → *Console* → run `install.packages("tidyverse")`.
- Set a project folder: *File* → *New Project*.

# Installing the R Environment

## Option B — Miniconda + JupyterLab (R & Python together)

1. Install **Miniconda**: https://docs.conda.io/en/latest/miniconda.html

```
conda create -n wass2s_hydro -c conda-forge -y \
  r-base=4.3 r-irkernel \
  r-tidyverse r-sf r-stars r-terra r-data.table \
  jupyterlab nodejs
conda activate wass2s_hydro
jupyter lab
```

# Installing the R Environment

## Option B — Miniconda + JupyterLab (R & Python together)

1. Install **Miniconda**: https://docs.conda.io/en/latest/miniconda.html
2. Create an environment with R + JupyterLab:

```
conda create -n wass2s_hydro -c conda-forge -y \
  r-base=4.3 r-irkernel \
  r-tidyverse r-sf r-stars r-terra r-data.table \
  jupyterlab nodejs
conda activate wass2s_hydro
jupyter lab
```

# Verifying Your Setup

Run these in a fresh R session (RStudio Console or Jupyter cell):

```r
# Versions
R.version.string
sessionInfo()

# Core packages
pkgs <- c("tidyverse", "tidymodels", "sf", "terra", "stars",
          "data.table", "ecmwfr")
installed <- sapply(pkgs, requireNamespace, quietly = TRUE)
data.frame(package = pkgs, installed = installed)
```

# Verifying Your Setup

### If something is missing

Install missing packages:

```r
install.packages(c("tidyverse","data.table"))
```

# Working Modes

ℹ Best practice

Use scripts for production code and notebooks for exploration and reporting. Keep data in `data/`, outputs in `outputs/`, and code in `R/` or `notebooks/`.

# THANK YOU FOR YOUR ATTENTATION

CILSS