Capacity Building in Seasonal Hydrological Forecasting

Spatial and Spatio-temporal Data

AGRHYMET, Climate Regional Center for West-Africa and Sahel

@Arsène KIEMA / @Mandela HOUNGNIBO

2025-10-06

Pedagogical Objectives

Learning outcomes

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

- Understand what spatial and spatio-temporal data are.
- Read and explore shapefiles and vector data with **sf**.
- Work with raster data (gridded precipitation, temperature) with **terra**.
- Handle spatio-temporal raster cubes with **stars**.
- Extract values (e.g., average rainfall) for sub-basins defined by polygons.
- Visualize spatial data (maps, overlays).
- Apply these skills in a hydrological case study: rainfall over sub-basins.

Spatial data

Spatial data links **attributes** (e.g., rainfall, discharge) to **geographic locations**.

Two main types:

- **Vector data**: points (stations), lines (rivers), polygons (basins) $\rightarrow$ handled by `sf`.
- **Raster data**: gridded values (precipitation, temperature, ETP) $\rightarrow$ handled by `terra`.
- **Spatio-temporal cubes**: rasters changing over time $\rightarrow$ handled by `stars`.

Spatial data

! Key concept

Spatial data always requires a **Coordinate Reference System (CRS)**. CRS ensures that layers align correctly. Always check CRS before analysis.

Vector Data with sf

```r
library(sf)

# Read shapefile of basins
basins <- st_read("data/basins_shapefile.shp")

# Explore
head(basins)
st_crs(basins)   # Coordinate reference system
plot(basins["BASIN_NAME"])
```

Vector Data with sf

> **Note**
>
> Use `st_transform()` to reproject your data into a common CRS.

```r
basins_utm <- st_transform(basins, crs = 32630) # UTM zone 30N
```

Visualization Vector Data

```r
library(ggplot2)

# Convert sf to data frame for ggplot
basins_df <- st_as_sf(basins)

ggplot() +
  geom_sf(data = basins_df, fill = "lightblue", color = "black") +
  ggtitle("Hydrological Basins")

# Overlay raster and vector
plot(rain_raster)
plot(st_geometry(basins), add = TRUE, border = "red")
```

Raster Data with terra

```r
library(terra)

# Read a raster (precipitation grid)
rain_raster <- rast("data/precipitation_2020.tif")

# Explore raster
rain_raster
plot(rain_raster, main = "Daily Precipitation (mm)")
```

Spatio-temporal Data with stars

```r
library(stars)

# Read a NetCDF file (daily precipitation)
precip <- read_stars("data/precipitation_daily.nc")

precip
st_dimensions(precip)   # check dimensions
st_get_dimension_values(precip, "time")[1:10]  # first 10 dates
```

Key function

`read_stars()` automatically reads NetCDF/GRIB and recognizes **time** as a dimension.

Exploring Spatio-temporal Cubes

```r
# Select the first time slice (a map for one day)
precip_day1 <- precip[,,1]
plot(precip_day1, main = "Precipitation on Day 1")

# Select a time period
precip_period <- precip[,,1:30]    # first 30 days
precip_period
```

Manipulating Dimensions

```r
# Get dimension names
names(st_dimensions(precip))

# Subset by time
precip_2020 <- precip[,,, "2020-01-01/2020-12-31"]

# Aggregate by month (mean rainfall per month)
precip_monthly <- aggregate(precip, by = "month", FUN = mean, na.rm =
precip_monthly
```

Manipulating Dimensions

Dimension handling

In `stars`, dimensions are explicit (`x`, `y`, `time`).
This makes it easy to **slice**, **aggregate**, or **filter** by time.

Extracting Time Series

```r
library(sf)

# Load station points (vector data)
stations <- st_read("data/stations.shp")

# Extract precipitation time series at station points
ts_station <- st_extract(precip, stations)
head(ts_station)
```

Basin-level Aggregation

```r
# Load basins
basins <- st_read("data/basins_shapefile.shp")

# Average rainfall per basin (zonal statistics)
precip_basins <- st_extract(precip, basins, FUN = mean, na.rm = TRUE)
head(precip_basins)
```

Basin-level Aggregation

Application

This is the basis for **forcing hydrological models**: transforming gridded rainfall into basin-average inputs.

Visualization of Spatio-temporal Data

```r
# Map of rainfall for a specific day
plot(precip[,,10], main = "Precipitation on Day 10")

# Animation over time (requires gganimate or mapview)
library(ggplot2)
precip_df <- as.data.frame(precip[,,1:5], xy = TRUE)  # first 5 days
head(precip_df)
```

Practical Exercises

Exercise 1

- Import `precipitation_daily.nc` with `stars`.
- Display the first 5 dates of the time dimension.
- Plot rainfall map for January 1st.

Exercise 2

- Aggregate daily rainfall into monthly totals.
- Plot the map for January and July.

Practical Exercises

Exercise 3

- Import `stations.shp`.
- Extract rainfall time series for each station.
- Plot the time series of Station A.

Exercise 4

- Import `basins_shapefile.shp`.
- Compute average daily rainfall per basin.
- Export results as `outputs/rainfall_basins.csv`.

Wrap-up

Key takeaways

- Spatio-temporal data = space (x,y) + time (t).
- Use **stars** for NetCDF/GRIB cubes with a time dimension.
- Always check **dimensions** and **CRS**.
- Typical workflow:
  - Import → Explore → Subset → Aggregate → Extract → Visualize → Export.
- Application: rainfall aggregated by basin, ready for hydrological modeling.

Mini-Project Day 3

Context

We want to create a spatio-temporal rainfall dataset aggregated at the basin level.

Tasks

1. Import `precipitation_daily.nc` with `stars`.
2. Import `basins_shapefile.shp`.
3. Ensure CRS alignment between raster and polygons.
4. Extract **average daily rainfall per basin**.
5. Export the results as a CSV file (`Date`, `Basin_ID`, `Rain`).
6. Plot rainfall time series for one basin of your choice.

THANK YOU FOR YOUR ATTENTION

CILSS