CEVE 101: Project 03

Your Name (your netid)

2024-10-23

Welcome to your first programming assignment! In this project, you’ll explore a rainwater harvesting simulation using Julia. This assignment is designed to introduce you to basic programming concepts, data analysis, and modeling in an engaging and practical way.

## Overview and aims

By the end of this assignment, you will:

* Run and understand the provided Julia code for the rainwater harvesting simulation.
* Modify the code by changing parameters like roof size or selecting a different weather station.
* Justify your modifications and discuss their impact on the results.
* Identify one gap in the current methods and suggest improvements.

## Getting started

Let’s begin by setting up our environment and loading the necessary packages and data. As discussed in class, you’ll need to install VS Code, Quarto, and JuliaUp. You’ll probably also want to install the Julia extension for VS Code. Don’t forget to activate and instantiate the environment, as discussed in class, and to build the IJUlia package.

### Loading packages

First, we need to load the required packages and utility functions. The RainwaterHarvesting package contains all the functions we’ll use.

using Revise  
using RainwaterHarvesting

Line 1

Revise allows us to update code without restarting Julia.

Line 2

We load the RainwaterHarvesting package that contains our simulation tools.

### Setting the file path

Next, specify the path to the rainfall data file. For this example, we’ll use data/1.txt. For your project, **choose a different station**!

filepath = "data/1.txt"

Line 1

This is the path to the rainfall data file for the station of interest.

"data/1.txt"

### Reading the Rainfall Data

Now, we can read in the rainfall data.

rainfall\_data = RainfallData(filepath)

Line 1

This creates a RainfallData object containing the rainfall data for the specified station.

RainfallData for station: ABAIARA  
 Location: -7.3615277777778, -39.0355  
 Years of data: 41  
 Total days: 14974

If we dig under the hood a bit, we can see that there are two main pieces of information:

* station\_info: a dictionary containing information about the station, such as its name and location.
* annual\_data: a dictionary mapping years to AnnualRainfallData objects, each containing the date and precipitation data for a given year.

|  |
| --- |
| Think |
| If you look at [types.jl](./RainwaterHarvesting/src/types.jl), you’ll find that we only keep years with at least 363 days of data. Why might it be a problem if there is lots of missing data in a given year? |

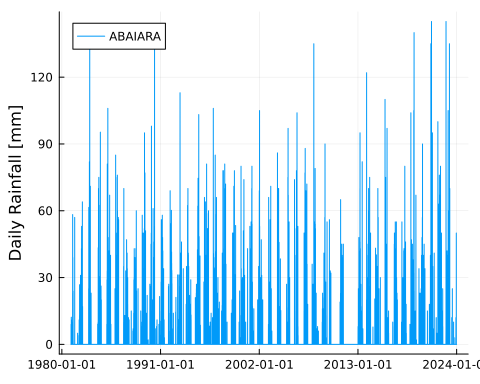
### Plotting the Data

We can plot the data for one or all of the years to verify that it looks correct.

plot(rainfall\_data)

Line 1

This plot function is defined in [viz.jl](./RainwaterHarvesting/src/viz.jl) and is a thin wrapper around the Plots.jl package.



### Your Tasks

1. Choose a Station and Load Data
   1. Select a different station (not 1.txt) from the data folder.
   2. Update the filepath variable to point to your chosen station.
   3. Repeat the steps above to load and plot the data for your station.
2. Analyze Available Data
   1. Identify which years are kept after data cleaning.
   2. Discuss whether the available data is sufficient for meaningful risk assessments.
3. Investigate Historical Droughts
   1. Research the years of major droughts in Ceará, Brazil.
   2. Determine if your data includes these years.
   3. Discuss the implications of missing data during drought years on your analysis.

## Understanding the Theoretical Model

Let’s explore the model that simulates the rainwater harvesting system.

### Mass Balance Equation

The mass balance model for the rainwater tank is given by:

* : the volume of water in the tank at time
* is the volume of water added to the tank at time
* is the volume of water consumed at time
* : the maximum volume of the tank

### Inflow ()

Inflow is calculated as

* : the runoff coefficient, which accounts for losses due to evaporation, spilling, etc.
* : the area of the roof (we will use square meters)
* : the precipitation at time (we will use millimeters per day)
* : the first flush volume (we will use millimeters). The first flush volume is often discarded, so that each time it rains the first bit of water (which is usually dirty) is not used (see [here](https://rainwaterharvesting.tamu.edu/pre-storage-treatment/) for more details).

### Consumption

Consumption is modeled as:

This makes two major assumptiuons. First, the household begins using water from the tank after the 150th day of the year. This is based on the idea that the rainy season lasts for about the first 150 days of the year, and so they may have other sources of water before then. The second assumption is that each household consumes 74.1 liters per day during the dry season. How does this compare to your household’s water usage?

## Model Implementation and Testing

### Defining the Model Parameters

We can define the model parameters with sensible default values. You will have the opportunity to change these values later.

param = ModelParameters(  
 runoff\_coefficient=0.85,  
 roof\_area\_m2=45.0,  
 first\_flush\_mm=2.0,  
 tank\_capacity\_L=16000.0  
)

Line 1

This creates a ModelParameters object with the specified parameters.

Line 2

The runoff coefficient () is a measure of how much of the rain that falls on the roof ends up in the tank.

Line 3

The roof area () is the area of the roof that the rain falls on.

Line 4

The first flush volume () is the volume of rain that is discarded because it is dirty.

Line 5

The tank capacity () is the maximum volume of the tank.

### Running the Simulation for One Year

Let’s run the model for the year 1981.

rainfall\_1981 = rainfall\_data.annual\_data[1981]  
results\_1981 = run\_timesteps(rainfall\_1981, param)  
p1 = plot(results\_1981)

Line 1

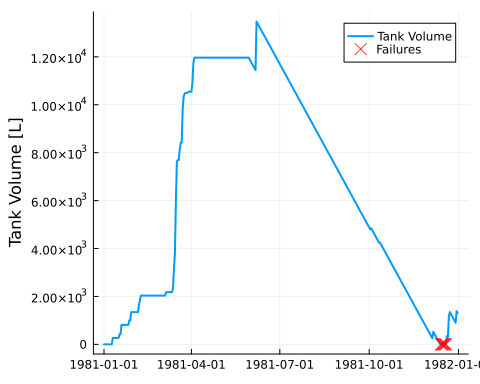
We select the annual rainfall data for the year 1981.

Line 2

We run the simulation for the year 1981.

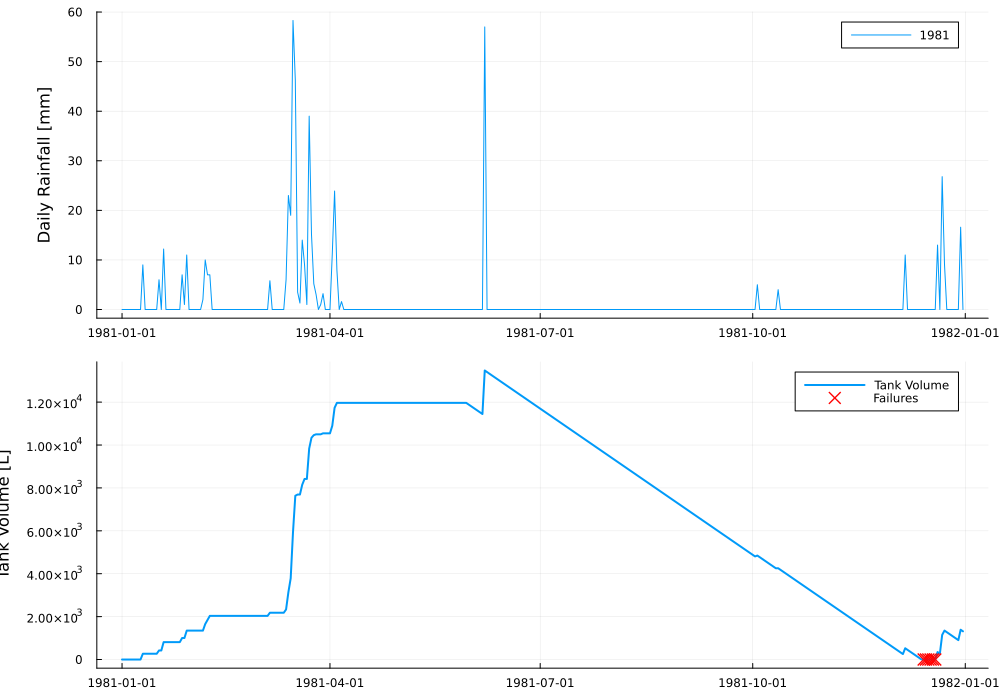
Line 3

We plot the results, again using a plot function defined in [viz.jl](./RainwaterHarvesting/src/viz.jl).



To compare rainfall and tank volume:

p2 = plot(rainfall\_1981)  
plot(p2, p1, layout=(2, 1), size=(1000, 700), link=:x)



Observe how the tank’s water level responds to rainfall events. Note any periods when the tank runs dry or overflows.

### Your Tasks

#### Modify Model Parameters

1. Adjust parameters like roof\_area\_m2 or tank\_capacity\_L in the param object.
2. Rerun the simulation with your new parameters.
3. Plot the results to visualize the impact.

#### Justify Your Modifications

1. Explain why you chose to modify these parameters.
2. Discuss how the changes affect the simulation outcomes.

#### Perform Reliability Analysis

1. Run the simulation for all available years.
2. Calculate the number of years where the system failed to meet the demand.
3. Determine the system’s reliability percentage.
4. Discuss and interpret your findings.

You can run simulations for all years at once with the following code:

all\_years = sort(collect(keys(rainfall\_data.annual\_data)))  
all\_results = [run\_timesteps(rainfall\_data.annual\_data[year], param) for year in all\_years]  
any\_failures = [!isempty(result.failure\_dates) for result in all\_results]  
println("Number of years with failures: ", sum(any\_failures), " out of ", length(all\_years))

Line 1

We get all the years in order.

Line 2

We run the simulation for each year.

Line 3

We check if any failures occurred.

Line 4

We print the number of years with failures and the total number of years.

Number of years with failures: 4 out of 41

#### Identify a Gap in the Methods

1. Find one limitation in the current model (e.g., data quality, assumptions). Explain why it’s a limitation.
2. Suggest how to address the identified gap.
3. Discuss how this would enhance the analysis. (Note: You don’t need to implement the change—just propose it.)