Eberswalde University
for Sustainable
Development

Lena Nahrwold, BA, BSc

Soil Moisture and Temperature Measurement

Work Report

Applied Programming in Forestry
Forest Information Technology MSc
WiSe 2023/2024

Supervisor: Prof. Dr. Luis Miranda

February 23, 2024

Contents

1 Introduction

Understanding soil water dynamics across different soil types is crucial for various applications in environmental science and land management. Soil water dynamics refer to the movement, distribution, and storage of water within the soil system, influenced by factors such as soil texture, structure, porosity, and organic matter content. Different soil types exhibit unique water retention capacities, drainage rates, and moisture distribution patterns, which significantly impact plant growth, ecosystem functions, and land-use planning (Hillel, 2007).

In agricultural and forestry contexts, knowledge of soil water dynamics informs decisions related to crop selection, irrigation management, and soil conservation practices. For instance, certain crops thrive in well-drained sandy soils, while others prefer loamy soils with higher water retention capabilities. Moreover, in ecological research and ecosystem monitoring, soil moisture and temperature data are essential for assessing habitat suitability, biodiversity patterns, and ecosystem resilience to environmental changes. Variations in soil moisture and temperature influence vegetation composition, wildlife habitat availability, and ecosystem services provision, underscoring the importance of accurate and comprehensive soil monitoring efforts.

In this project, I present the development and implementation of a soil moisture monitoring system designed to provide real-time data on soil conditions for agricultural applications. The system integrates Arduino-based sensors with Python data processing capabilities to enable efficient data collection, analysis, and visualization. The objective is to establish a user-friendly solution that provides insights into soil moisture dynamics, including the influence of air temperature and humidity.

2 Approach

2.1 Hardware Components

The hardware components used in this project include:

- Arduino UNO microcontroller board
- DHT11 temperature and humidity sensor
- Capacitive soil moisture sensor v1.2
- Micro SD card adapter module
- Powering unit

The capacitive soil moisture sensor functions at a voltage of 3.3V and utilizes a 555 timer integrated circuit to produce an analog signal that is proportional to its resonance. The analog output of the sensor is connected to one of the analog input pins (e.g., A0) on the Arduino board. Its operation is based on measuring the dielectric constant of the soil, which correlates directly with its moisture level. Comprising two typically metal electrodes inserted into the soil, the sensor detects variations in dielectric constant caused by moisture content changes. By applying a small AC voltage across these electrodes, it generates an electric field within the surrounding soil. The soil's capacitance, or its capacity to store electrical charge, fluctuates with moisture content, thereby influencing the sensor's output voltage. This voltage alteration is then associated with the soil's moisture level, offering a quantitative assessment (Okasha et al., 2021).
For detecting relative humidity and air temperature, a DHT11 sensor is employed. Operating within a voltage range of 3.3V to 5V, this analogue output sensor module encompasses three pins: two for power supply (GND and Vcc), linked to the microcontroller's power pins, and a third data pin connected to the microcontroller's A0 port for analogue-to-digital conversion.
The micro SD module comprises six pins, two for power supply and four for serial peripheral interface (SPI). Pin 1 is connected to the microcontroller's GND, pin 2 to the microcontroller's 5V, pin 3 (MISO) to D12, pin 4 (MOSI) to D11, pin 5 (SCK) to D13, and pin 6 (CS) to D5 on the microcontroller.

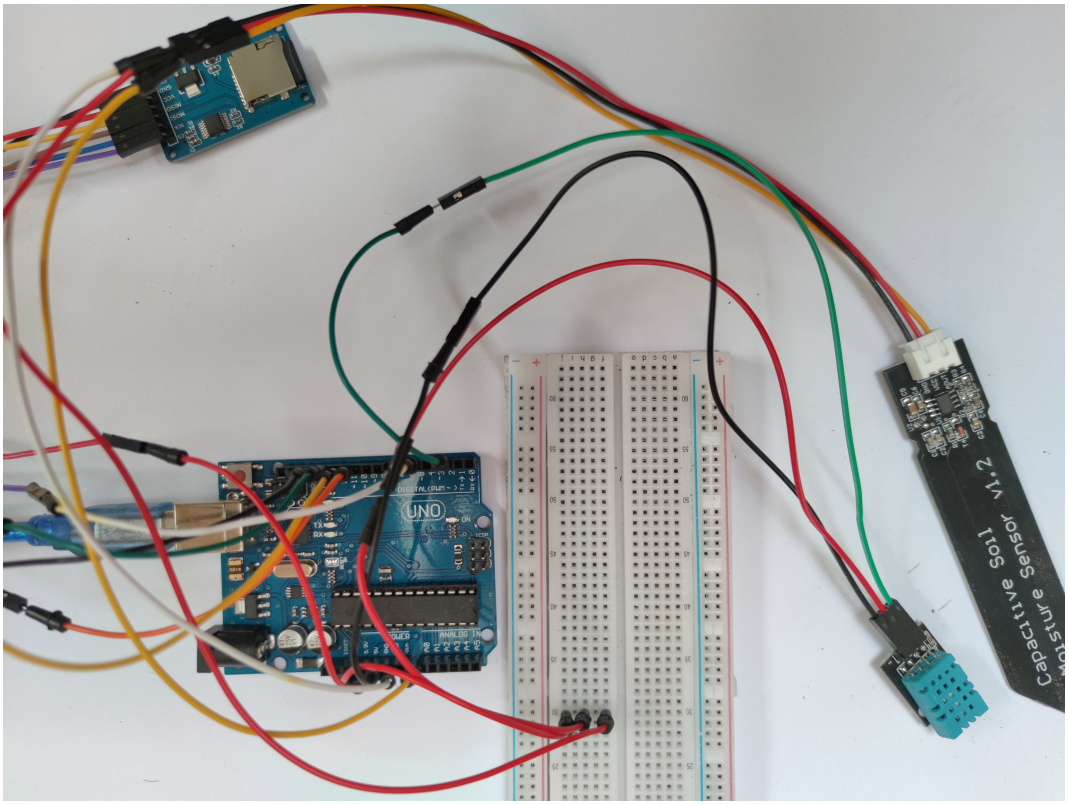

Figure 2.1: Hardware components.

Figure 2.2: Measuring in peat soil.

2.2 Arduino Code

The Arduino code (`sensor_data_collection.ino`) is responsible for reading data from the sensors and storing it in a CSV file on a micro SD card. The code initializes the sensors, establishes communication with the SD card, and implements a data collection loop with specified frequency. Sensor readings for soil moisture, temperature, and humidity are obtained and stored in the CSV file for further processing.

Libraries:

- `SPI.h`: This library enables communication with devices via the Serial Peripheral Interface (SPI) bus.
- `SD.h`: This library facilitates communication with the SD card module.
- `DHT.h`: This library provides functions to interact with DHT temperature and humidity sensors.

Constants:

- `DHTPIN`: The digital pin connected to the DHT temperature and humidity sensor.
- `DHTTYPE`: The type of DHT sensor being used (DHT11 in this case).
- `POWER_PIN`: The digital pin connected to control the power of the soil moisture sensor.
- `WATER_SENSOR_PIN`: The analog pin connected to the soil moisture sensor.
- `air`: The calibration value for the soil moisture sensor in dry air.
- `water`: The calibration value for the soil moisture sensor in water.

Global Variables:

- `file`: An object representing the file on the SD card where the data will be stored.
- `freq`: The data collection frequency in milliseconds.
- `m`: Variable to store the raw analog reading from the soil moisture sensor.
- `p`: Variable to store the calculated soil moisture percentage.
- `h`: Variable to store the humidity reading from the DHT sensor.
- `t`: Variable to store the temperature reading from the DHT sensor.
- `hic`: Variable to store the calculated heat index from the DHT sensor.

`setup()` **Function:**

- Initializes the serial communication at a baud rate of 9600.
- Initializes the SD card.

- Opens the *DATA.CSV* file on the SD card or creates it if it does not exist.
- Initializes the DHT sensor.
- Sets the pinMode for the `POWER_PIN` as an output and sets it LOW to turn off the soil moisture sensor.

`loop()` **Function:**

- Continuously runs the main program loop.
- Checks if the SD card is initialized and the file is accessible.
- If accessible, it reads data from the soil moisture and DHT sensors.
- Writes the collected data to the file on the SD card.
- Delays for the specified frequency before repeating the loop.

Calibration of Soil Moisture Sensor:

- To calibrate the soil moisture sensor readings measurements were taken in complete dry condition and in pure water.
- These values represent the sensor's analog readings in dry air (445) and water (189), respectively.
- The mapping function scales the raw readings between the air and water calibration values to the range of 0 to 100.
- This calculated percentage represents the soil moisture content, where 0 indicates dry soil and 100 indicates saturated soil.

Additional Notes:

- The code continuously checks for the presence of the SD card and the accessibility of the data file to ensure uninterrupted data logging.
- Error handling is implemented to handle cases where sensor readings fail or file operations encounter errors.

2.3 Data Processing Python Script

The Python script `data_processing.py` processes the data stored on the micro SD card and generates plots for visualization (see fig. 2.5 for an example). The script utilizes the `pandas` and `matplotlib` libraries for data manipulation and visualization. It reads the CSV file containing sensor data, combines and analyzes the data, and generates plots illustrating the relationships between soil moisture, temperature, and humidity. Additionally, the script allows users to select the soil type and generate plots with previously collected data and a specified start date. See figure 2.3 for a visualization of the script's flow.

Global Variables Initialization:

- `timestamp`: Stores the current timestamp when the script is executed.
- `soil_type`: Stores the selected soil type.
- `abbrv`: Stores the abbreviation for the selected soil type.

Function: `set_timestamp()`:

- Sets the global `timestamp` variable to the current date and time in the format `YYYY-MM-DD_HH-MM-SS`.

Function: `set_soil_type()`:

- Displays existing soil types after calling `read_soil_types()`.
- Allows the user to select an existing abbreviation or define a new one for the measured soil type.
- Creates folders for plots and data storage corresponding to the selected soil type.
- Updates the list of soil types and abbreviations in the `soil_types.txt` file.

Function: `read_soil_types()`:

- Reads existing soil types and abbreviations from the `soil_types.txt` file and returns them as a dictionary.

Function: `combine_csv_data(start_month, start_year)`:

- Combines data from CSV files with a specified start date into a single DataFrame.
- Filters CSV files based on the provided start month and year.
- Returns the combined DataFrame.

Function: `extract_month_year(file_name)`:

- Extracts the month and year from a given file name in the format `data_YYYY-MM-DD_HH-MM-SS.csv`.

Function: `create_plots(df, output_directory)`:

- Generates subplots visualizing the relationships between temperature, humidity, and soil moisture (see fig. 2.5 for an example).
- Saves the plots as PNG files in the specified output directory.
- Adds descriptions about the number of measurements and processing date to the plots.

Function: `process_data(csv_file_path)`:

- Loads data from a CSV file into a DataFrame.
- Calls `create_plots()` function to generate plots based on the loaded data.

- Stores the processed data to a CSV file on the local machine.
- Provides options to delete the CSV file from the microSD card and generate plots for a specified time range.

Function: `main()`:

- Defines the main execution flow of the script.
- Checks for the presence of the microSD card and the data file.
- Calls functions to set timestamp, select soil type, and process data.

2.4 Comparative Analysis Python Script

The script `comparative_analysis.py` reads soil data for selected soil types, generates comparison plots illustrating the relationships between soil moisture, humidity, temperature, and heat index, and saves the plots for analysis (see fig. 2.6 for an example). Users input the soil types they want to compare, and the script automatically retrieves and visualizes the corresponding data, facilitating the analysis of environmental factors across different soil types. See figure 2.4 for a visualization of the script's flow.

Function: `read_soil_data(soil_type)`**:**

- This function reads soil data (CSV files) for a specified soil type.
- It takes the soil type abbreviation as input.
- It searches for CSV files in the corresponding data folder (e.g., "./data/TE" for soil type "TE").
- It reads each CSV file into a Pandas DataFrame and adds a "soil_type" column to identify the data.
- It returns a concatenated DataFrame containing data from all CSV files of the specified soil type.

Function: `generate_comparison_plots(selected_soil_types)`**:**

- This function generates comparison plots for selected soil types.
- It takes a list of selected soil type abbreviations as input.
- It reads soil data for each selected soil type using the `read_soil_data()` function.
- It combines the data into a single DataFrame.
- It generates four scatter plots comparing temperature, humidity, and soil moisture for each selected soil type.
- It assigns different colors to each soil type for better visualization.
- It saves the generated plots as PNG files in the "plots" folder.
- It prints a message indicating the successful saving of the comparison plot.

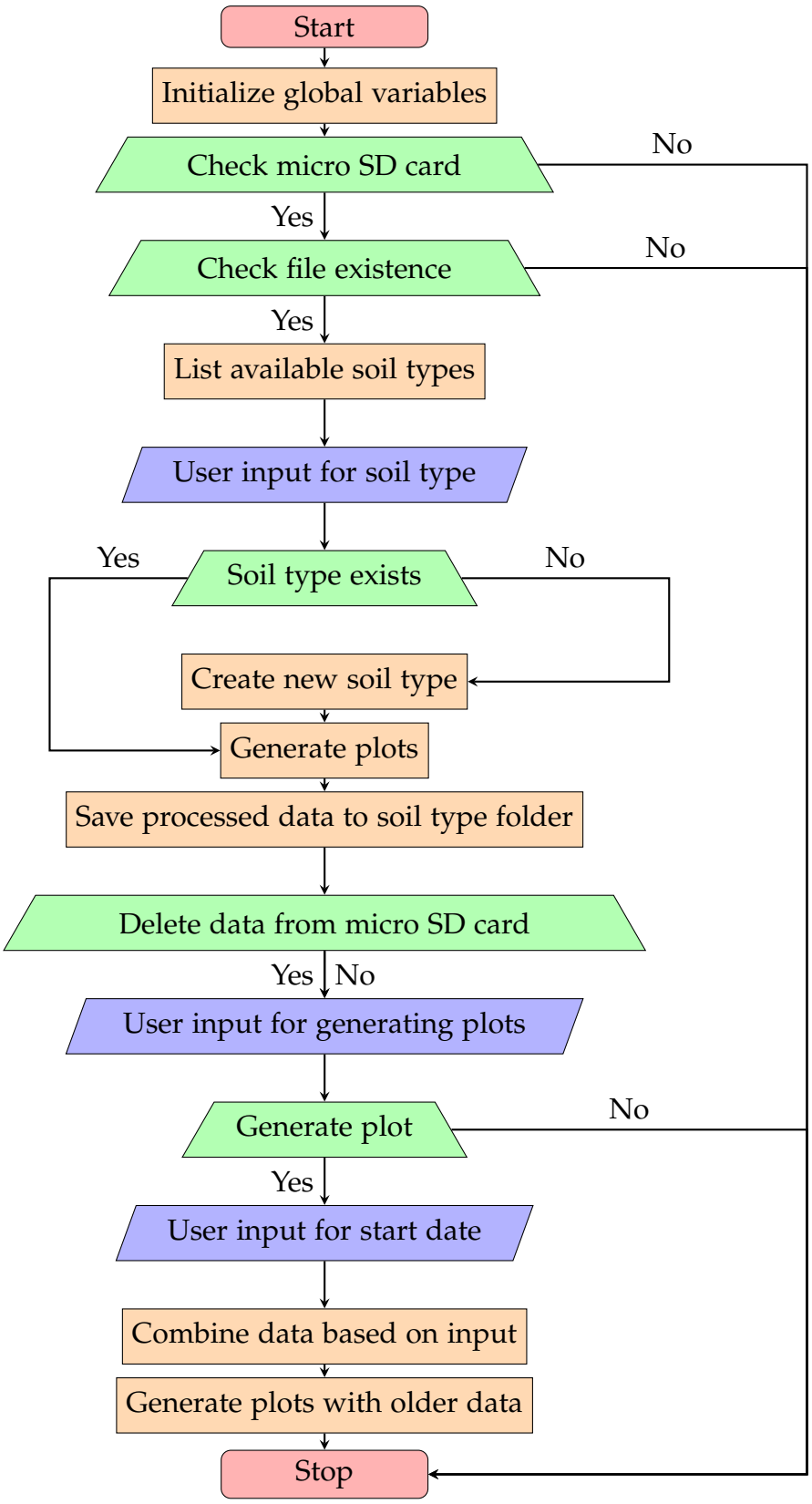

Figure 2.3: Flow Chart Soil Data Processing Script

Function: `main()`**:**

- This function defines the main execution flow of the script.
- It reads soil types and their abbreviations using the `read_soil_types()` function.
- It prompts the user to select soil types for comparison.
- It calls the `generate_comparison_plots()` function with the selected soil types as input to generate comparison plots.

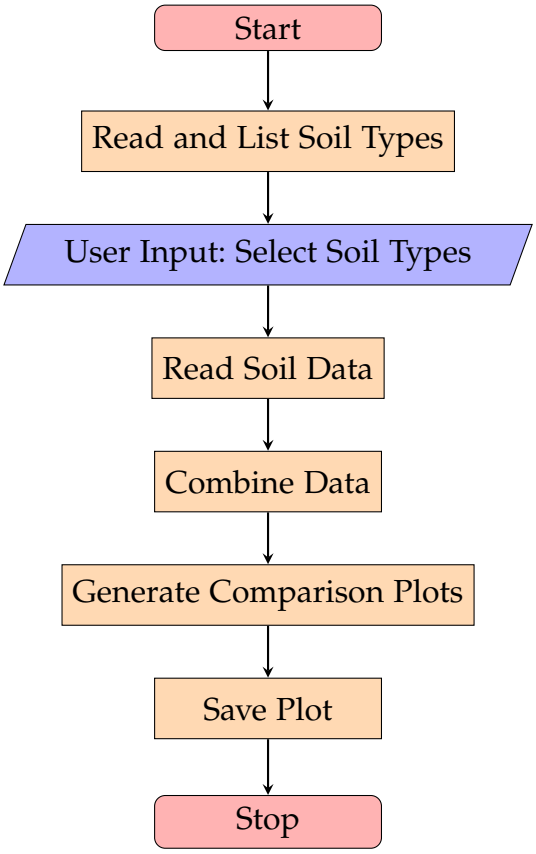

Figure 2.4: Flow Chart Soil Comparative Analysis Script.

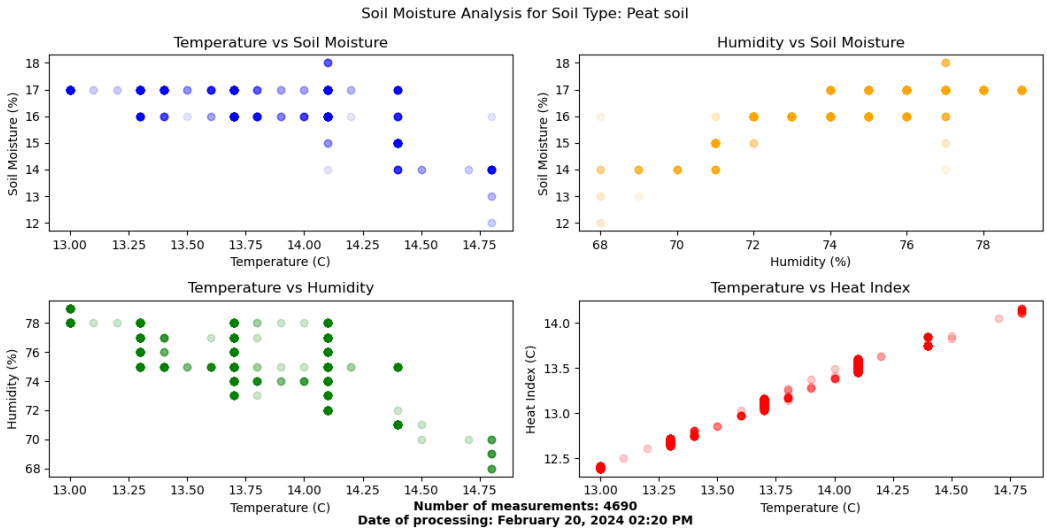

Figure 2.5: Soil Moisture Analysis of Peat Soil.

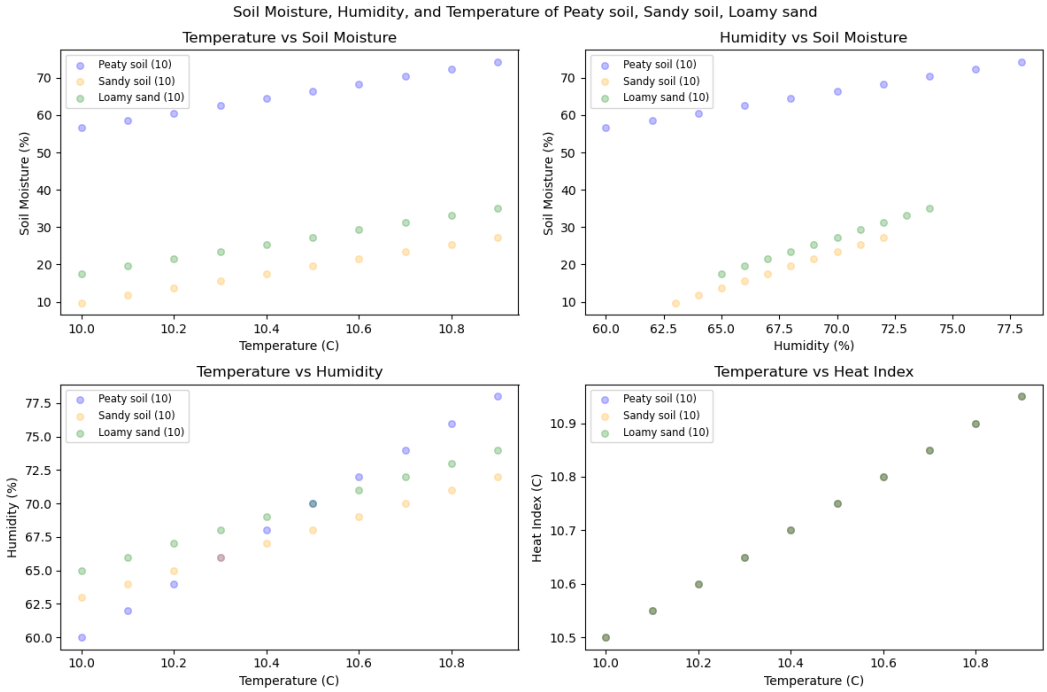

Figure 2.6: Comparative Analysis of Peaty Soil, Sandy Soil and Loamy Soil.

3 Discussion

The developed system provides an efficient and user-friendly solution for monitoring soil moisture and temperature. By combining Arduino-based sensors with Python data processing capabilities, the system offers real-time data collection, analysis, and visualization. Users can easily interpret the generated plots to gain insights into soil conditions and make informed decisions regarding irrigation, crop management, and environmental monitoring.

While the project has laid the groundwork for the development and implementation of a customized soil moisture monitoring system, the analysis of error rates and the calibration of the capacitive sensor remain crucial steps that should precede its practical application. The investigation by Okasha et al. (2021) of commercial soil moisture sensors under varying temperature and salinity conditions revealed sensitivities to these factors, emphasizing the importance of accurate calibration to ensure reliable performance, particularly in challenging environments.

For the calibration process to be truly effective, soil probes should be taken, oven dried, weighed in order to calculate the volumetric water content for different soil types. It should be noted that the calibration performed in this project may not be sufficient for accurately measuring soil moisture and drawing definitive conclusions.

Additionally, while the current system focuses on monitoring soil moisture, air temperature and humidity, the integration of another sensor for measuring soil temperature could further enhance the system's capabilities.

Moreover, the implementation of an efficient customizable data processing system represents a significant contribution of this project. By offering a streamlined process for processing, plotting, analyzing, and documenting sensor data, the project enhances the accessibility and utility of soil moisture data. The approach aims to facilitate informed decision-making in various agricultural applications.

List of Figures

Bibliography

Hillel, D. (2007). *Soil in the environment: Crucible of terrestrial life*. Elsevier Science. https://books.google.de/books?id=rZeosY3dGFIC. (Cit. on p. 1)

Okasha, A. M., Ibrahim, H. G., Elmetwalli, A. H., Khedher, K. M., Yaseen, Z. M., & Elsayed, S. (2021). Designing low-cost capacitive-based soil moisture sensor and smart monitoring unit operated by solar cells for greenhouse irrigation management. *Sensors (Basel)*, *21*(16), 5387. https://doi.org/10.3390/s21165387 (cit. on pp. 2, 12)