Hobo report – quality control

Winter term 24/25

Max Schmit

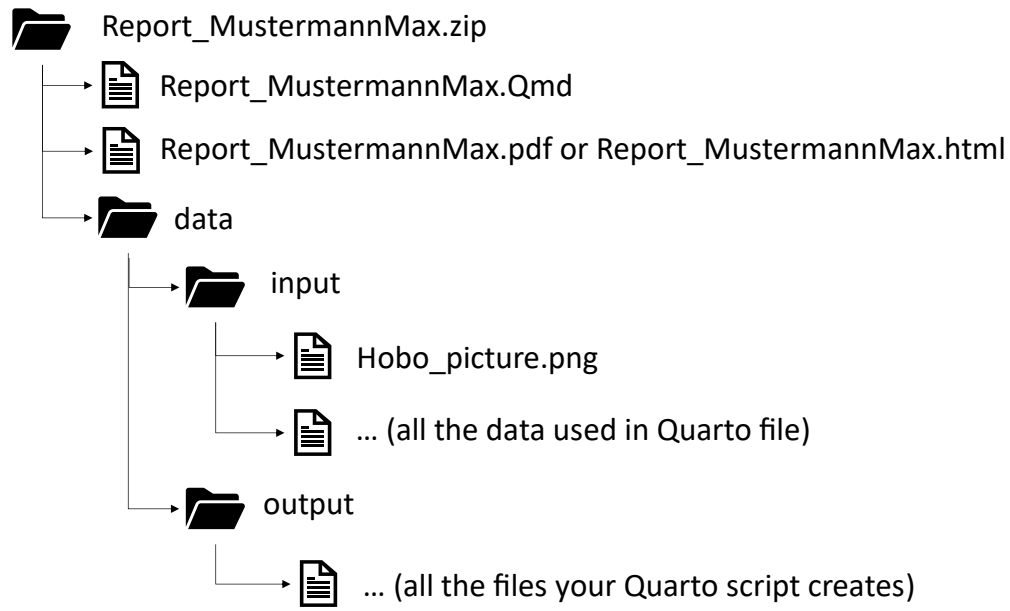
Date: December 17, 2024

General explanation

You will need to hand in one overall report, as raw Quarto (.Qmd) or RMarkdown file (.Rmd) and as a rendered version (.pdf or .html) of this document. To start your Quarto document, use the *report_template.Qmd* file as a template. It is a relatively plain Quarto template. However, you can make your own adjustments as long as the following rule-set is fulfilled:

1. The submitted Quarto file should be executable, i.e. all code chunks should work and the document should be renderable on other machines. This means that the data you load should originate from online resources (e.g. GitHub) and not from local files. The only exception is for images used in your protocol. Those can be saved locally and have to get uploaded together with your report at the end. If local files are loaded make sure to reference them as relative paths to your Quarto document to be executable on other machines.
2. The GitHub repository should not be cloned in your handed in zip file. You can either use a `github_dir` variable to store the absolute path to the cloned data git repository on your system or don't clone the repository but only work online on GitHub with the repository. If there is a `github_dir` variable, the value will be replaced before testing the executability of your document.
3. To mark your report the submitted rendered document, but also your Quarto file is considered (e.g. code chunks, analysis, text, output in Qmd). Be aware of different options for your code chunks like `eval=TRUE, message=TRUE, warnings=FALSE` etc. Be sure that all needed code chunks are rendered at the end!
4. All analysis of your protocol should be integrated in the Quarto file. You can test code or functions in separate (temporary) R scripts, but a final version of your code should go to the Quarto file. Please add only code, code chunks and variables etc. that are needed to gain the results and to render the document. Remove all clutter at the end when you revise your Qmd. Test all code chunks and render the document in a fresh R session. You can remove dummy comments / text placeholders and set your own comments there.
5. You can add more code chunks, add smaller programming comments in the chunks or your own remarks or explanations around the chunks. Be sure to answer all questions or tasks in your protocol (some should be answered with code or code output, some could be answered with small text answers). Text answers should normally be 1-2 paragraphs (i.e. total 100-250 words). Some tasks include maximum word counts. These are **maximum** word counts. You can write less. You can't write more. You don't have to write a lot, but guide the reader through your document.

6. The script chunks should be nicely styled to be readable. There are a lot of styling possibilities and we are not obliging you to use a specific one, but you should decide on a consistent style for your report. Some suggestions for styling can be found on https://google.github.io/styleguide/Rguide.html and https://style.tidyverse.org/index.html , but those are only suggestions.
7. This is a scientific report, so the figures and tables should have captions and get referenced by their caption-number in the text. (Figures have caption underneath, tables above)
8. At the end you will have to hand in your Quarto file named *Report_LastnameFirstname.Qmd* **and** a rendered version (.pdf or .html). If you used images or your script needs any data, add them to a *data/input* subfolder for the file to be renderable. Only exception is the GitHub repository, see point 2. Zip all the files to one ZIP-file named *Report_LastnameFirstname.zip.* So, create one Zip file containing all the files for uploading. The folder structure should be exactly like:



Submit this ZIP-file on Moodle until Sunday the **16th of February 2025 at 23:55**.

The report will consist of different exercises explained in this document. For some exercises, there will be a second file giving you some helping information how you could reach your goal. As those documents did grow over the years, they are not always adjusted to this year's goals. If they are not consistent to the tasks in this document, please ignore them and stick to this document.

Additionally, to your report's tasks, we will grade your overall document on the following topics:

Table 1: Overall grading topics with their explanation and points.

Topic	Explanation (Questions to ask)	Points
Code style	• Is your code well styled and commented? • Are all loaded libraries needed?	6
Quarto	• Is the document renderable by others? Did you manage to render it? • Does the code fit into the document? Are the lines not too long? • Do you use online resources to load data? • Is the submitted ZIP folder structure in the right format? • How does the rendered document look like?	10
Language	• How is your scientific language? • Do your figures and tables have captions? • Is your text and document fluently?	6

1 HOBO meta information (max 300 words) 7 points

Explain in your own words, where your Hobo was situated and why you did choose this location. Give some explanation on possible errors. Basically, **resume your meta information**, so everyone reading your report has a clear understanding of your Hobo measurement dataset and is able to analyze and understand possible flaws of your data.

Part of this task will also be your representation of your Hobo meta information in the **online spreadsheet** on http://tiny.cc/EMDAV_HoboMeta .

2 Consistent HOBO data file 9 points

To work with data, it is important to decide on a consistent format of your data. It is the aim of this exercise, to generate this consistent version of your HOBO data. At the end of this exercise all HOBO files should look exactly the same. The individual files should be uploaded to GitHub to make your data available for all students. All HOBO files can then be found in one GitHub directory.

2.1 Calibration 5 points

To get good data, it is important to calibrate the measurement devices. To do this, we did put all the HOBO devices into a bucket of warm and a bucket of cold water before reading out the measurement data. The water temperature was measured furthermore by a laboratory device with a higher accuracy. This will be the calibration value and you will find it on GitHub:
https://github.com/data-hydenv/data/blob/master/hobo/2025/calibration.csv

To calibrate your HOBO measurements, you will need to join your Hobos measurement based on the timestamp with the calibration values. To take into account, that the Hobos need some time to adjust to the temperature, you should ignore the first 2 measurements. **Create a linear relationship** between the measured and the calibration temperature and **apply this relationship on your** measured temperature to get the calibrated measurement and to minimize the systematical error.

2.2 Create consistent Hobo file 4 points

Import your raw Hobo data from GitHub and create a local file that is consistent to the following rules:

- one header line, without any quotation marks, like " or '
- observations from line 2 on
- 3 columns with the following names: *id*, *dttm* and *temp*
- *id*: consecutive id starting with 1
- *dttm*: a string with date and time information in this exact format (YYYY-MM-DD hh:mm) without any other information like "T", "Z" or "UTC" or other time zone information
- the time zone of the data should be in CEWT (UTC+1).
- *temp*: values of air temperature measurements in °C
- the data resolution should be the same as in the raw file, with trailing zeros (e.g. 3 digits)
- one line per observation (an observation is a 10-min value from the time series)
- Your file will have 6 (timesteps per hour) x 24 (hours) = 144 values per day, if no measurement is available the file should have an "NA" written at this observation point
- The timeseries start on the 31.10.2023 at 00:00 and ends on the 15.12.2024 at 23:50 (UTC+1)
- the column separator is a comma (,), the decimal separator is a point (.)
- The name of your file should be "*your_hobo_id.csv*" (e.g. *10305099.csv*)

After you created your Hobo file locally and all the format specifications are correct, **upload it to the following GitHub folder:**

https://github.com/data-hydenv/data/tree/master/hobo/2025/10_minutes

3 Quality control 17 points

Data quality control is very important to make sure that your data is really a collection of measurements of your variables of interest. Analysis of data is only as good as the quality control. False or bad data can lead to wrong conclusions during data analysis and also to wrong model applications. The aims of this exercise are:

- to implement several data quality control procedures (QCPs),
- to create a QC-flagging system
- to generate a new hourly series with quality-controlled data points (and NA values if the quality checks failed)

3.1 First impression (max. 100 words) 3 points

To start re-import your data.frame created in chapter 2 from GitHub and **plot your data as a line plot**, to get a first impression of it. Plotting the data can sometimes give you a good first impression if something went wrong. In your opinion does the data look plausible?

3.2 Quality control procedures (QCPs) 11 points

During quality control check means to check if a certain condition is fulfilled or not, flag means a systematic flagging of bad data points when the check fails.

3.2.1 Measurement range 3 points

Check each temperature data point to be in the measurement range. See the HOBO manual for specification.

3.2.2 Plausible rate of change 3 points

Check each temperature data point to have not more than 1 K (1 Kelvin) temperature change compared to the previous data point.

3.2.3 Minimum variability (Persistence) 5 points

If temperature has not changed during the last 60 minutes (i.e. data point T_i plus 5 data points before, so from T_{i-1} to T_{i-5}) the corresponding data point T_i failed in this QCP.

For those of you that are a bit more advanced in R, try to optimize this quality control by using the method defined in (Zahumensky, 2004, Guidelines on Quality Control Procedures for Data from Automatic Weather Stations) at the bottom of page 6. To be more precise, if the measured data did not change by more than 0.1°C over the previous 60 minutes the corresponding data point T_i fails the test.

3.3 Summarize (max. 250 words) 6 points

Now every 10-minute observation has 5 different flags for the different QCP. To decide whether a data point is good or not, you will have to **consolidate those QCPs to one flag**. To do so, add a column "*qc_all*" to the data.frame or tibble. If a data point fails at least one check (qc1, qc2, qc3) the data point is flagged.

Analyze the result from the QCPs. Present a table or graph to show how many data points fail during the five specific QCPs. **Discuss shortly the reasons for failure and compare the different QCPs against each other.**

At the end of this section you should have generated **one** tibble or data.frame named *qc_df* with all timesteps, data points (temperature) and your outcomes of the different QCPs.

3.4 Aggregate 3 points

In chapter Summarize 3.3 you created one flag for each 10-minutes observation. On this basis you will **decide whether the hourly value is calculated or not**. So, proceed with the following rule-set:

- If one specific hour of data (e.g. 12:00 - 12:50) has one or no flag you aggregate the six 10-minute values of each hour to an hourly average: That means your measured data is OK.
- If one specific hour of data (e.g. 12:00 - 12:50) has two or more flags the hour is considered as erroneous and the corresponding hour in the generated hourly temperature series must be set to NA. That means your measured data is classified as bad data

At the end of this exercise you should have generated **one** tibble or data.frame named *hobo_hourly* with averaged temperature values per hour or NA values (if the hour is flagged as bad data). **Follow the following rules to create this data.frame:**

- 2 columns: date_time and th
- date_time: the timestamp in UTC+1
- th: hourly Temperature values (4 digits), NA values possible
- the date_time should be continuous, so for every hour in the measurement period there should be one line

4 Fill-up 10 points

For further analysis it is important to have complete timeseries with one value for every hour. Therefore, it is important to fill-up your missing observations. The goal of this exercise will be to **fill up all the NA** in your *hobo_hourly* from chapter 3.4 based on a **regression model** between your station and one reference station.

At the end of this exercise, you will have to have an hourly temperature series without any gaps/missing values and upload it to GitHub into the "*hourly*" folder.

4.1 Find reference station 3 points

As reference stations you should use the closest official temperature sensors on the EcoSense towers from a similar altitude above ground. Therefor **compute the distance** of your Hobo to all 3 towers and select the minimal. You will find the exact position of the towers in the EcoSense GIS folder (see below).

EcoSense GIS-folder: https://bwsyncandshare.kit.edu/s/ZQmWiQcr7a8dgCD

4.2 Fill-up your missing values (max. 150 words) 5 points

To fillup your missing values, you will use the measurements from the selected reference station from chapter 4.1. Their measurements can be found on GitHub (see below).

GitHub reference stations data: https://github.com/data-hydenv/data/tree/master/reference/2025/

Build a linear regression model between your measurements and the reference station.

To **fill up your missing values**, extract the equation (slope and intercept) or use `predict()` to fill up the missing measurement points. **Check if there is no missing data left and make a plot to show that your gap filling procedure did a good job.**

4.3 Upload data 2 points

After the quality checks and the gap filling with reference stations, **upload your filled, hourly HOBO time series** with the name "*hoboid_Th.csv*" (e.g. "*10350099_Th.csv*") to the GitHub data repository (look there for the *"hourly"* subfolder and the right year). Note: In this *"*_Th.csv*" file there should be absolutely no NA values (i.e. `any(is.na(temp)) = FALSE`) as this was the aim of this exercise task.

Use the same file and data format as described in the chapters before on raw data assessment. Column separator should be a comma ("**.csv*"). The file must contain three columns:

1. dttm: Date and time in the format "YYYY.MM.DD hh" in CEWT (another name is UTC+1)
2. th: Temperature values in °C
3. origin: A flag indicating whether the value is from your HOBO station (H) or filled by regression (R)

4.4 Advancing your model analysis (optional) (+ 3 points)

The measure R^2 might not be a perfect index to judge the different models. Give your suggestions on how to optimize the model analysis and why you think your optimized analysis is better.