Mining temporal data – Transforms

Algorithmic Data Analysis – Coding Assignment 3

Giulia Ortolani

February 13, 2024

Resources: No resources
Collaborations: No collaborations

Function that decompose a time series using discrete wavelet transform

The idea is to build a function that takes in input the time series (`S`) and the fraction of weights to be retained (`frac`). The output of this function are the weights of the decomposition (`a`), the matrix of basis vectors (`W`), the approsimated time series (`Sapp`) and the ratio of energy retained in the approximate time series (`ren`).

Implementation of the `dwt` function

After computing the length n of the time series, the function computes the Haar matrix `W` (implemented recursively). Then, for each order k, the function computes 2^{k-1} weights, with $k = 1, \dots, \log_2(n)$, and adds them to the weights vector `a`.
If we don't specify the fraction of weights that we want to keep, the function simply returns the computed `a` and `W`. Otherwise, it computes the energy E associated with each weight. Then, it selects the weights with higher energy and sets all the others to zero, also returning the percentage of energy retained. With the new weights vector, it computes the approximated time series.

Application to weather data

The `dwt` algorithm is here applied to few time-series representing weather variables.
The selected dataset are:

- The minimum temperature in Kuopio (data collected hourly), in the period 1.1.2024 - 4.1.2024. The resulting approximated time series is shown in Figure 1.
- The daily snow depth in Kuopio in the period 1.11.2023 - 1.2.2024. The resulting approximated time series is shown in Figure 2.
- The maximum gust speed in Kuopio (data collected hourly), in the period 1.1.2024 - 4.1.2024. The resulting approximated time series is shown in Figure 3.
- The monthly precipitation in Kuopio in four years, data from 1.1.2020 until 1.1.2024. The resulting approximated time series is shown in Figure 4.
- The average realtive humidity in Kuopio (data collected hourly), in the period 5.2.2024 - 8.2.2024. The resulting approximated time series is shown in Figure 5.

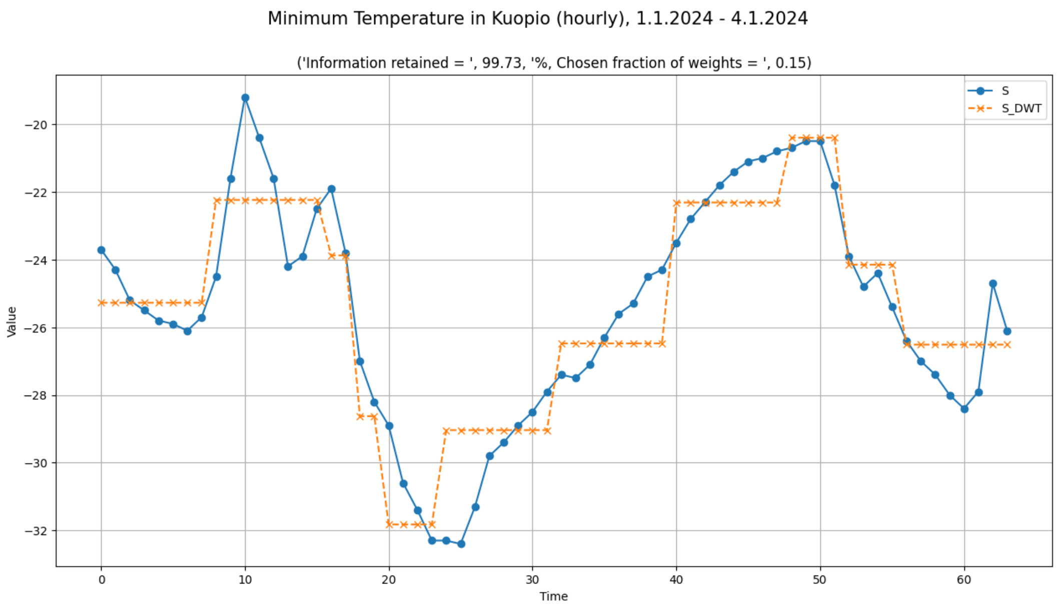

Figure 1: Minimum Temperature in Kuopio (hourly), 1.1.2024 - 4.1.2024.
The length of the time series is 64 observation. The chosen fraction of weights kept is 15%, with this choice we can obtain a quite good approximation retaining the 99.73% of the original information.

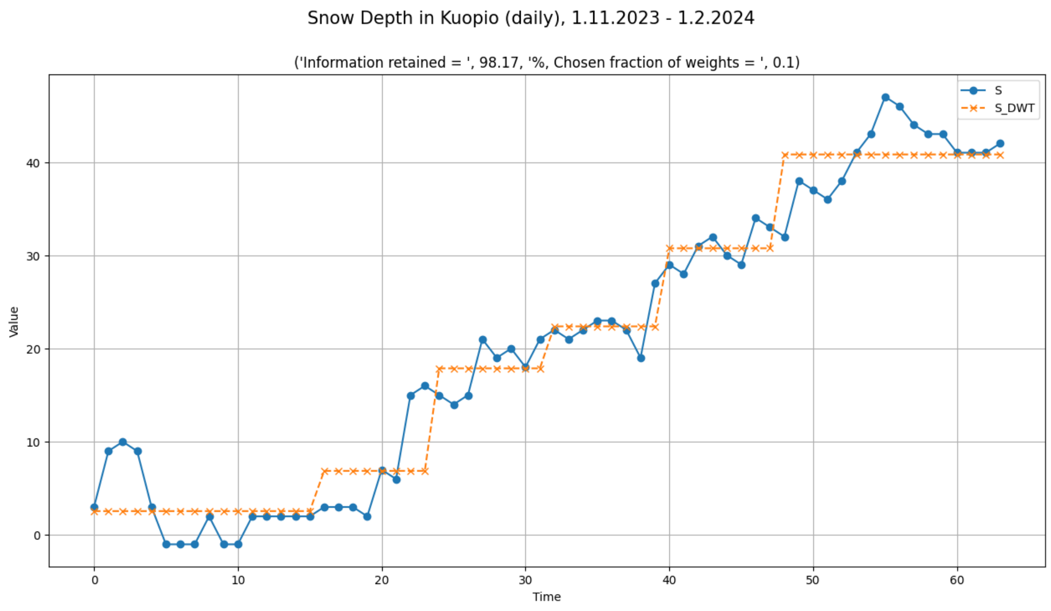

Figure 2: Snow Depth in Kuopio (daily), 1.11.2023 - 1.2.2024.
The length of the time series is 64 observation. The chosen fraction of weights kept is 10%, with this choice we can obtain a quite good approximation retaining the 98.17% of the original information.

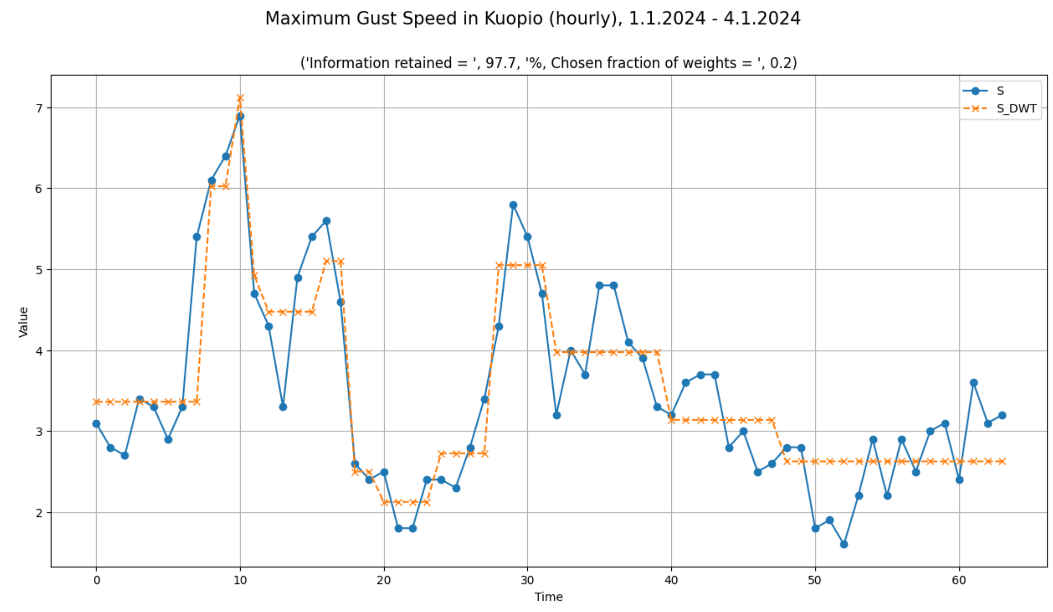

Figure 3: Maximum Gust Speed in Kuopio (hourly), 1.1.2024 - 4.1.2024.
The length of the time series is 64 observation. The chosen fraction of weights kept is 20%, with this choice we can obtain a quite good approximation retaining the 97.7% of the original information. This time series oscillates a lot, so it's necessary to retain more coefficients in order to have a good approximation.

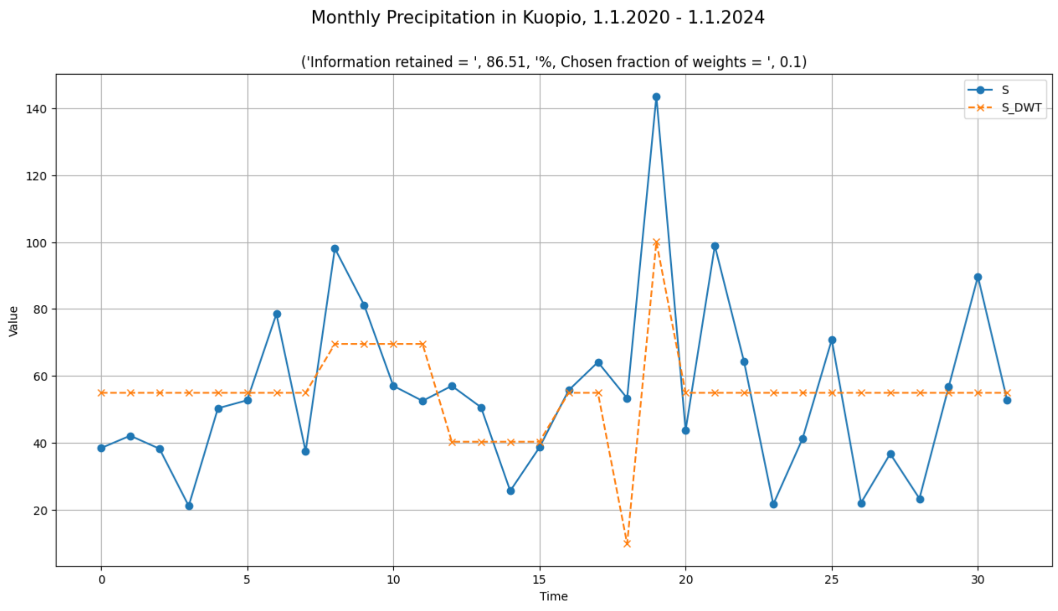

Figure 4: Monthly Precipitation in Kuopio, 1.1.2020 - 1.1.2024.
The length of the time series is 32 observation. The chosen fraction of weights kept is 10%, with this choice we retain the 86.51% of the original information. As we can see, in this case the approximated time series struggles a bit in following the oscillations of the original series.

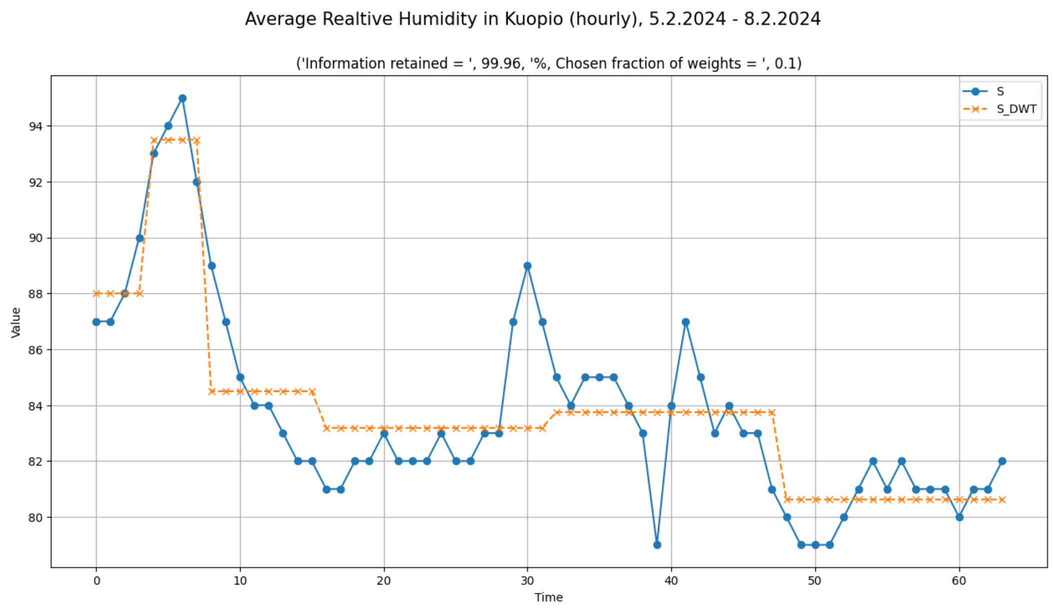

Figure 5: Average Realtive Humidity in Kuopio (hourly), 5.2.2024 - 8.2.2024.
The length of the time series is 64 observation. The chosen fraction of weights kept is 10%, with this choice we can obtain a quite good approximation retaining the 99.96% of the original information.

Function that decompose a time series using discrete Fourier transform

The discrete Fourier transform decomposes the time-series into a collection of sinusoids with associated coefficients f_k, $k = 0, \dots, n-1$. The original time-series can be reconstructed by summing all the weighted sinusoids.

Implementation of the `dft` function

This function works similarly as the `dwt` function. The Fourier coefficients are computed using the `fft` function from the `numpy.fft` package. Each coefficient is of the form $f_k = a_k + ib_i$, and the energy associated is given by $a_k^2 + b_k^2$.
Similar to the previous method, we retain coefficients with higher energy and set the others to zero. Then, we return the approximated time series (the sum of the weighted sinusoids) and the ratio of energy retained in this new series.

Comparison between the decomposition methods

Trying to decompose the same time series using the Fourier transform gives the following results:

- The comparison on the dataset containing the minimum temperature in Kuopio (data collected hourly) in the period 1.1.2024 - 4.1.2024 is shown in Figure 6.
- The daily snow depth in Kuopio in the period 1.11.2023 - 1.2.2024. The resulting approximated time series is shown in Figure 7.
- The maximum gust speed in Kuopio (data collected hourly), in the period 1.1.2024 - 4.1.2024. The resulting approximated time series is shown in Figure 8.
- The monthly precipitation in Kuopio in four years, data from 1.1.2020 until 1.1.2024. The resulting approximated time series is shown in Figure 9.

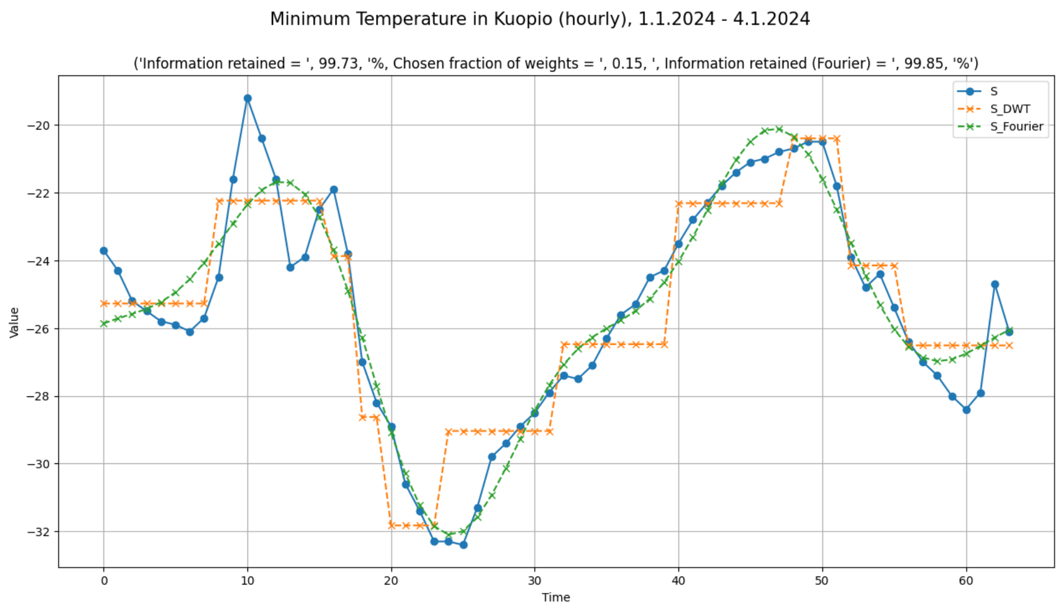

Figure 6: Minimum Temperature in Kuopio (hourly), 1.1.2024 - 4.1.2024.
Here, the Fourier decomposition is better equipped to represent the smooth growth of the time series (therefore has slightly better percentage of retained information), but both decompositions have limitations in modeling the peak at the beginning of the observation period.

- The average realtive humidity in Kuopio (data collected hourly), in the period 5.2.2024 - 8.2.2024. The resulting approximated time series is shown in Figure 10.

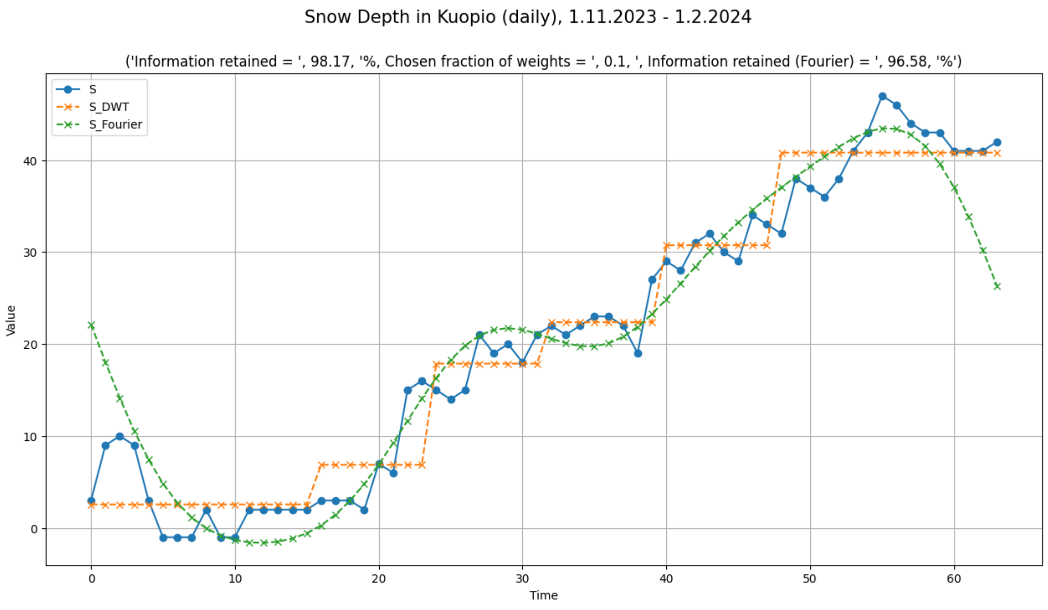

Figure 7: Snow Depth in Kuopio (daily), 1.11.2023 - 1.2.2024.
Here, both decompositions work fine in the approximation, but the Fourier decomposition fails to model the initial and final portions of the time series.

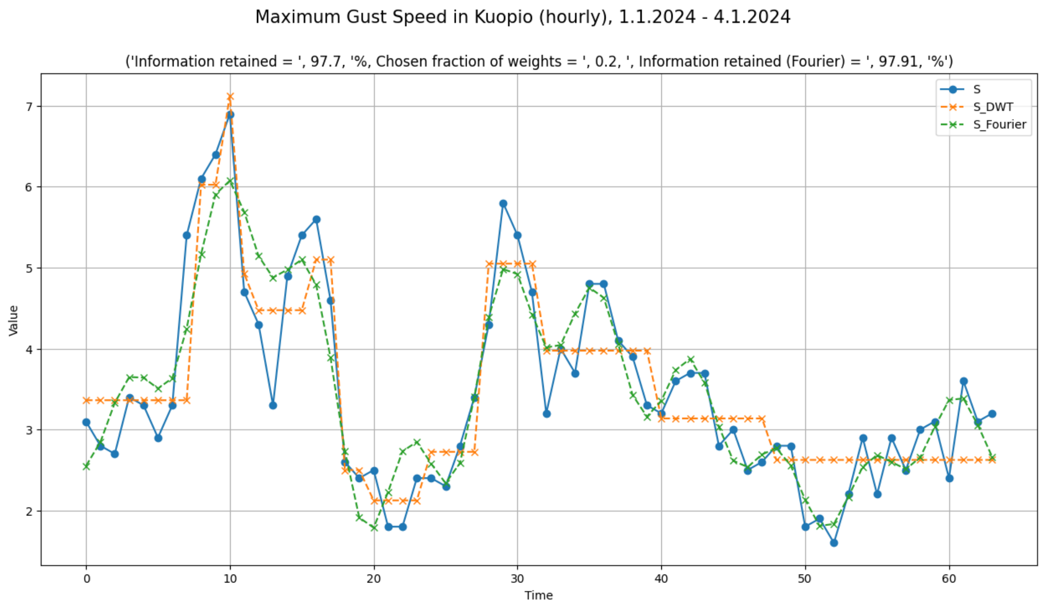

Figure 8: Maximum Gust Speed in Kuopio (hourly), 1.1.2024 - 4.1.2024.
Both decompositions work well, but the Fourier decomposition is better able to reproduce the oscillatory trend of the second half of the time series.

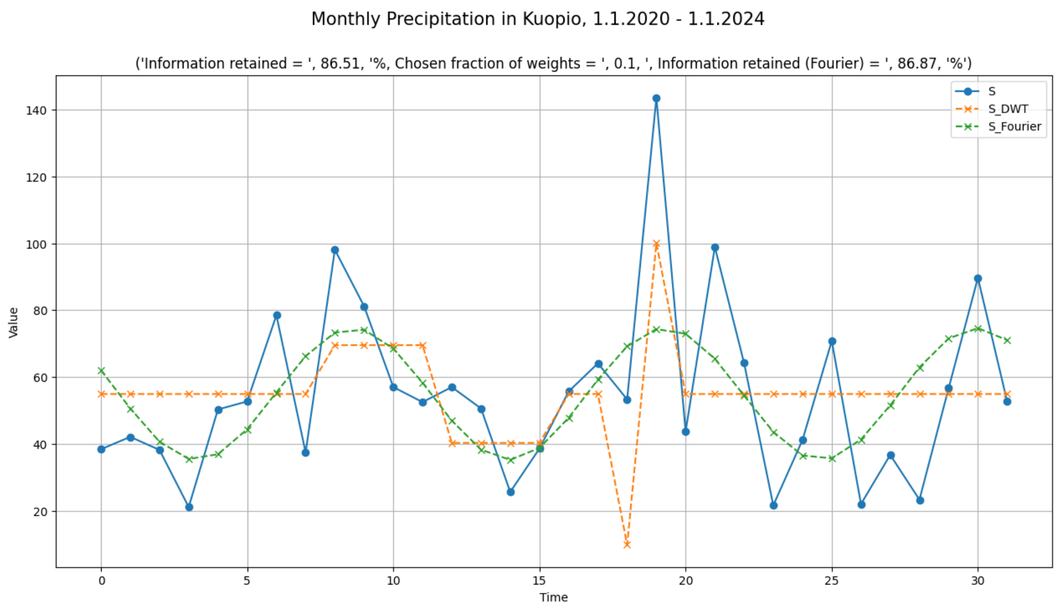

Figure 9: Monthly Precipitation in Kuopio, 1.1.2020 - 1.1.2024.
Here we have an example of approximate and not very precise modeling. Increasing the percentage of weights retained could lead to a better result. However, I chose to retain only 10% in this example to observe how the basic behavior of the two decompositions differs: the discrete wavelet decomposition consists of segments and tends to flatten out areas of low precision, while the Fourier decomposition is composed of sinusoids, so its basic behavior is that of a sinusoid. In this example, we can clearly see these two behaviors represented.

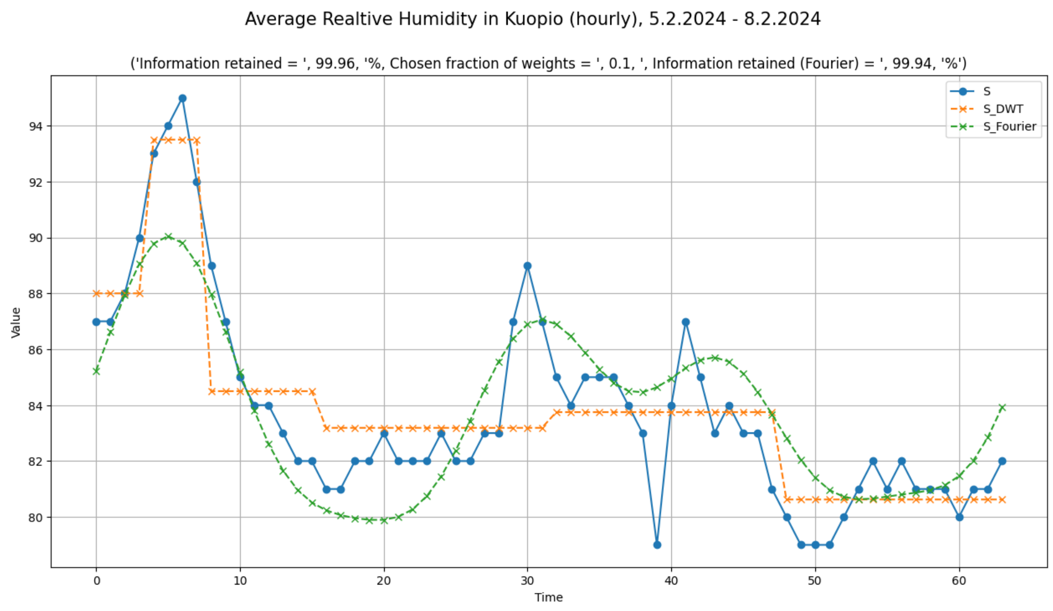

Figure 10: Average Realtive Humidity in Kuopio (hourly), 5.2.2024 - 8.2.2024.
Here, the two decompositions work well, and in particular, the discrete wavelet effectively models the initial trend. The Fourier decomposition gives an idea of the overall trend, but both fail to maintain the positions of the main peaks.