ETH zürich

Air Quality Forecast Challenge

V. Gillioz, C. Sicard, K. Pyszkowski

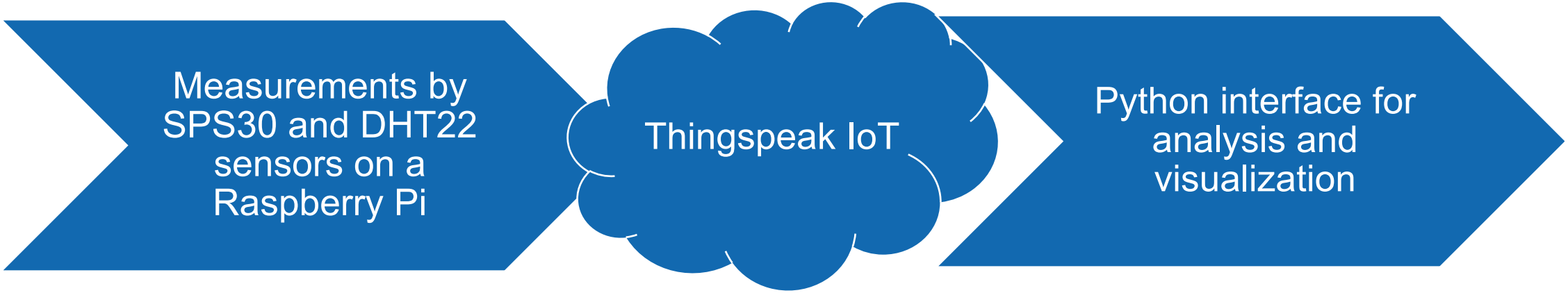
17.05.2021

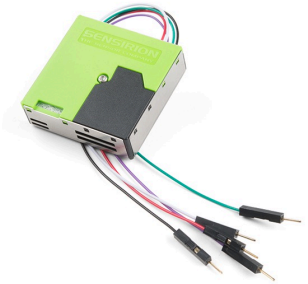
Agenda

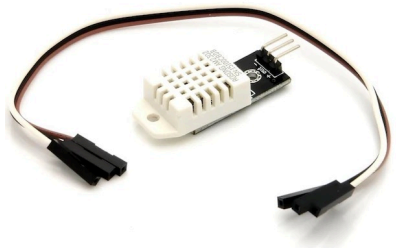
1. Data Pipeline
2. Data Visualization
3. Data Preprocessing
4. Forecasting PM2.5 Time Series Using a Recurrent Neural Network
5. Comparison of Our Model with Other Time-Series Forecasting Methods
    a. Data Prediction using Fast Fourier Transform
    b. Data Prediction using Facebook Prophet
    c. Data Prediction using SARIMA
6. Conclusion

1. Data Pipeline

Setup of Data Pipeline





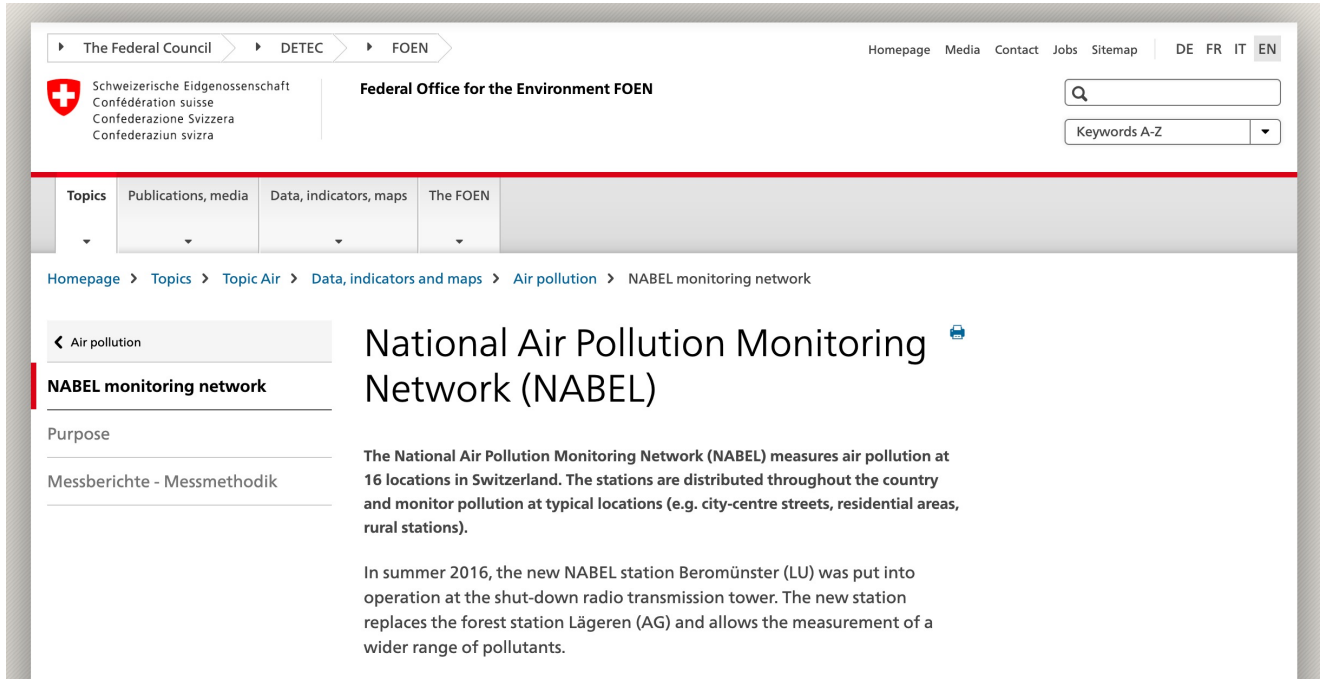


- Every **5 minutes :**
  - 1 measurement is made
  - Saved on Raspberry Pi in CSV format
  - Uploaded on Thingspeak IoT

But it after 6 days it stopped working…

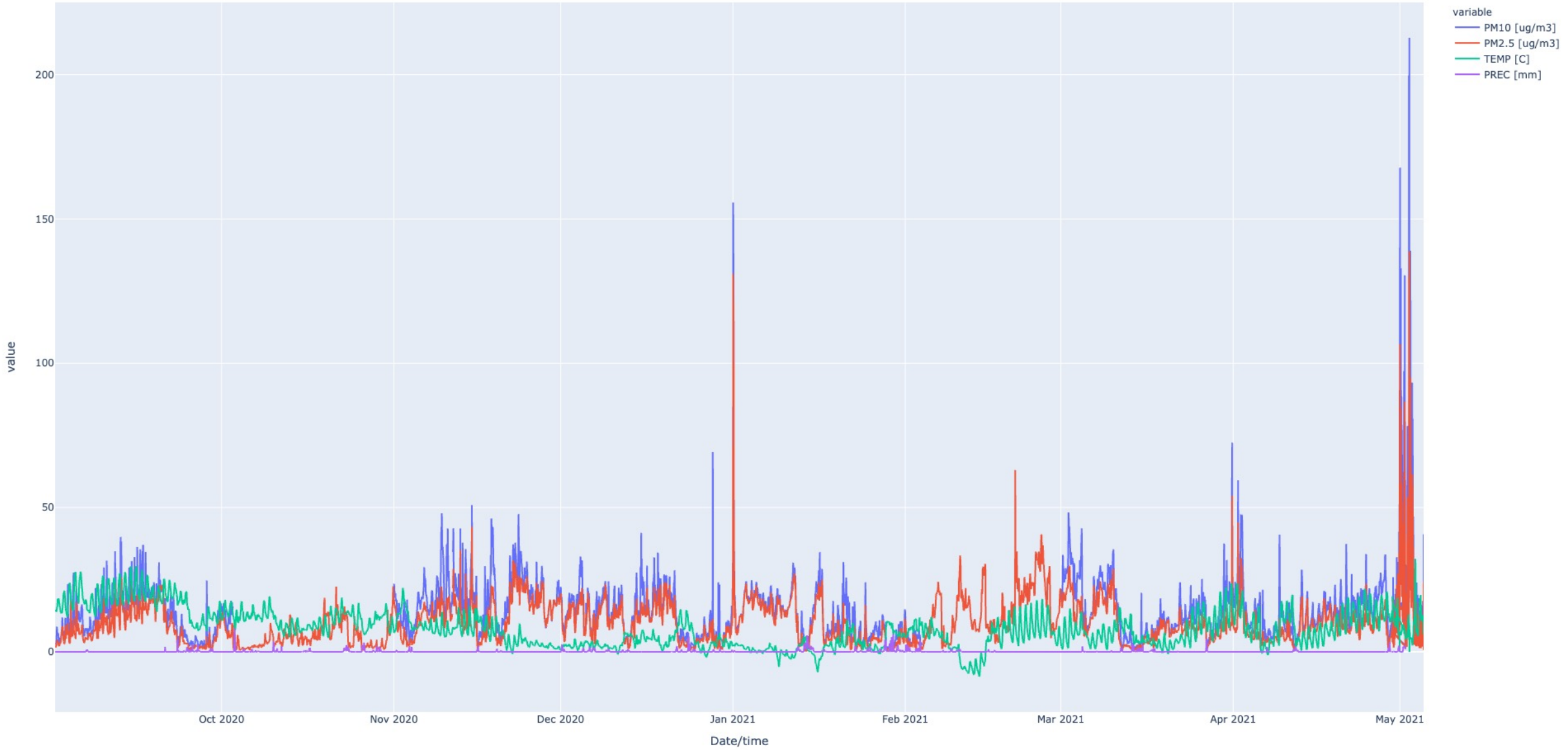
Issues with the Initial Data Pipeline

- Only 6 days of measurements, so ~1’500 measurements
- Get a new Raspberry ? **Not enough time**
- Use public data
  - National Air Pollution Monitoring Network
    - 1 measurement **every hour**
    - In a few big cities of Switzerland
- **First strategy** : Get the same amount of data samples from NABEL so that we have the initially planned number of collected data
  - **Bad strategy** → Not the same time intervals (5 minutes vs. 1 hour)



2. Data Visualization

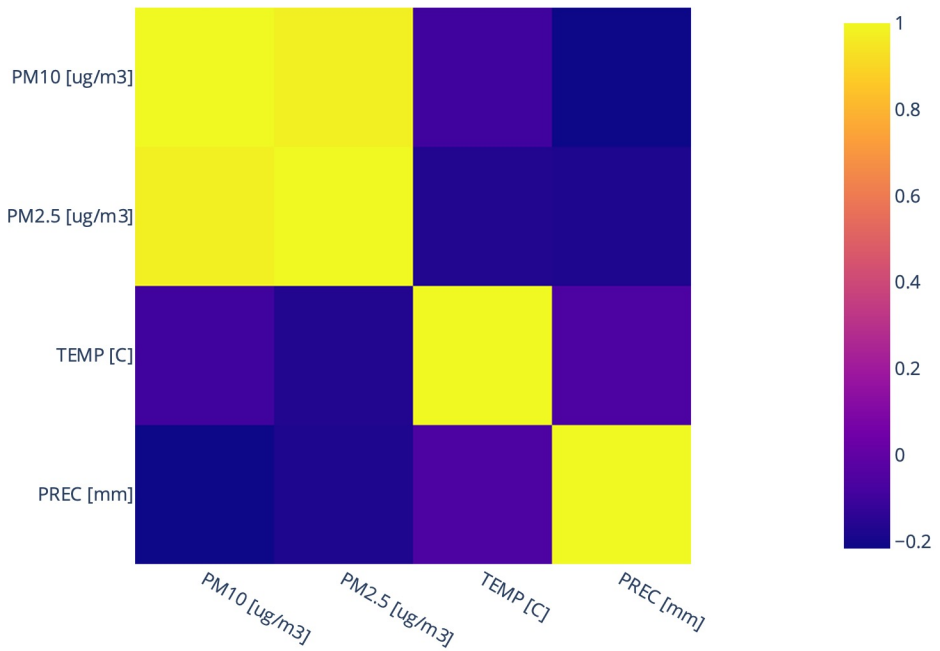
Dataset Visualization



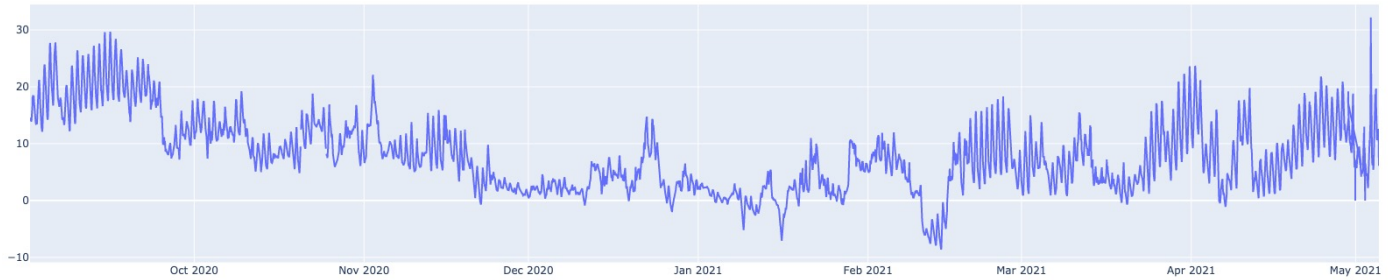
Variables Visualization

- We first **merged** our two sources of data
- But not great since the **time intervals** between two measurements **are different**
- There are also a lot of **missing values**

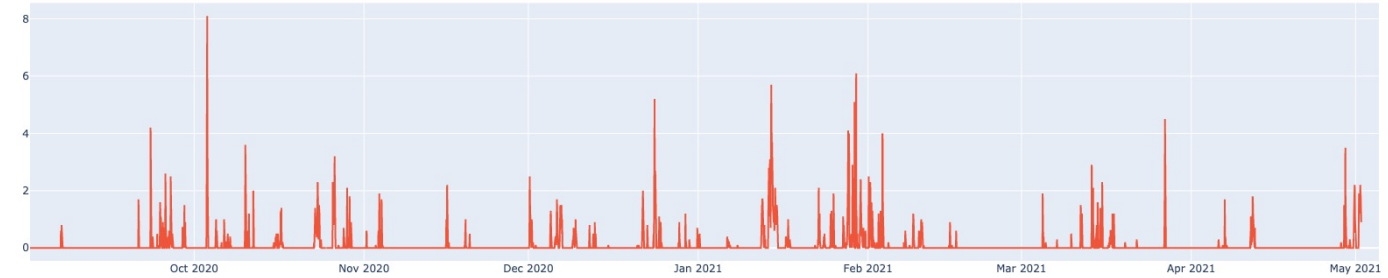
Correlogram of the dataset and its variables



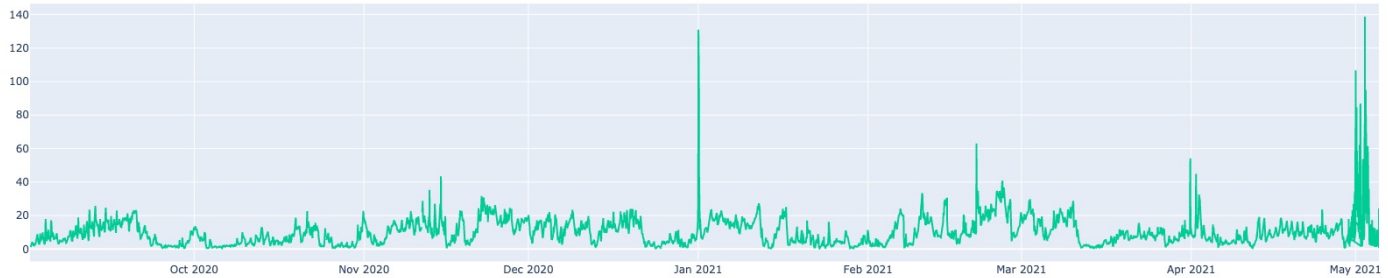
Temperature (in °C)



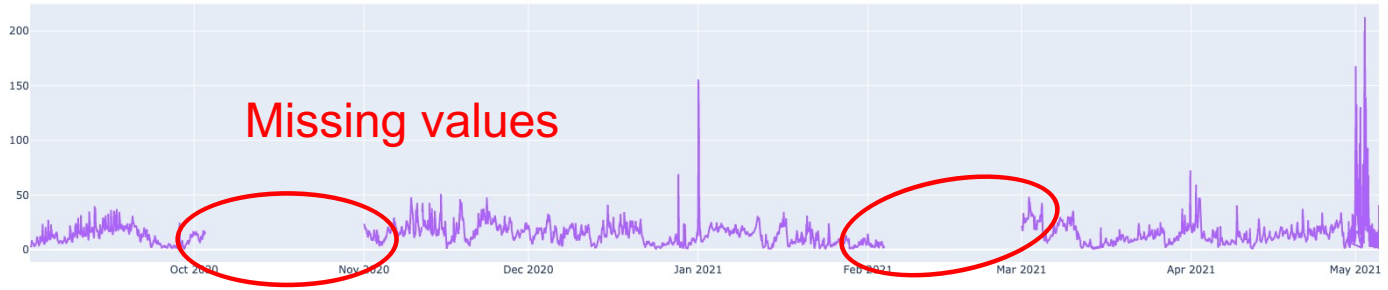
Precipitation (in mm)



PM2.5 (in µg/m3)



PM10 (in µg/m3)



ETH zürich

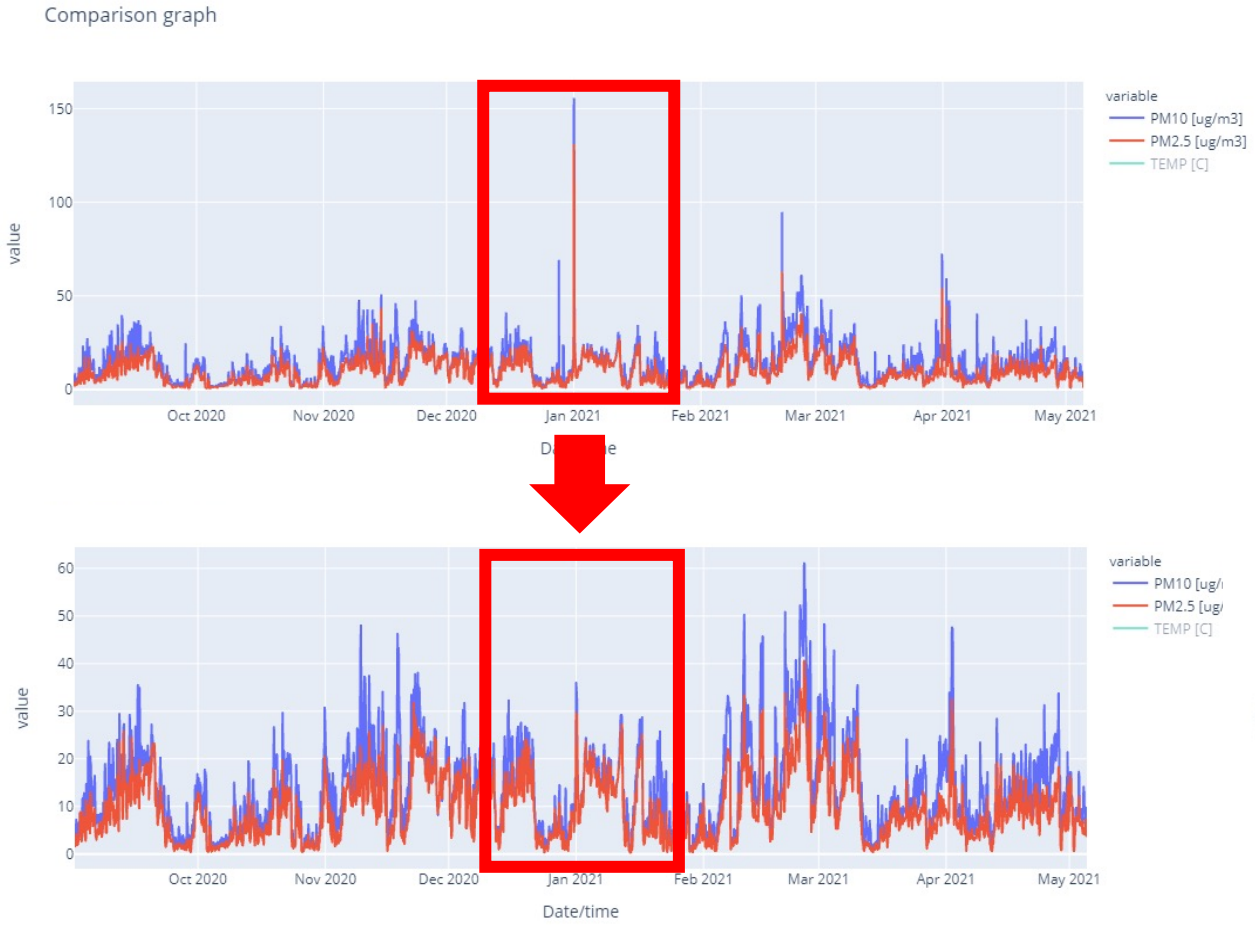
3. Data Preprocessing

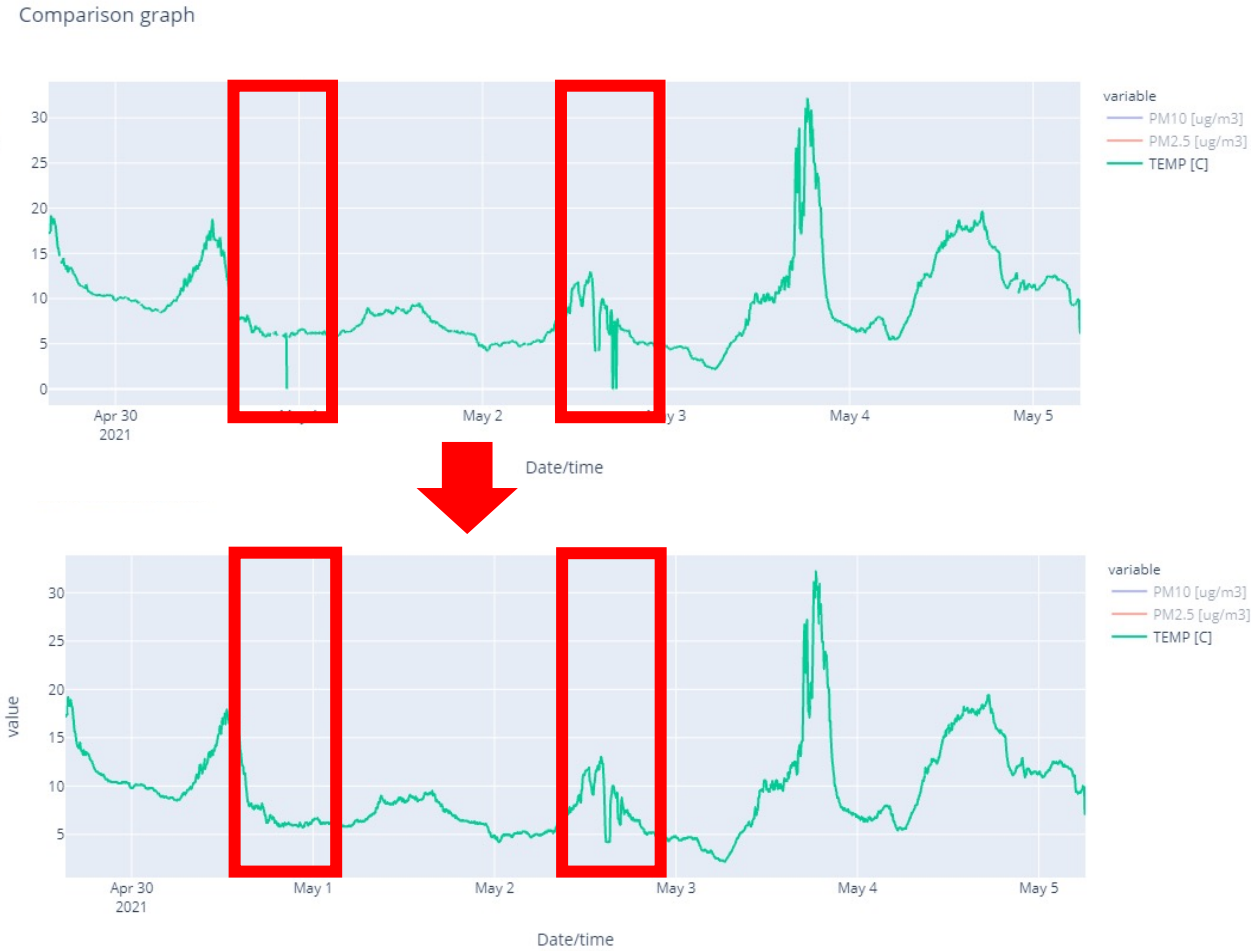
Imputation of Large Missing Data Intervals

- **Dynamic Time Warping** (DTW) had bad results in finding similarities between the two curves
- However, the high correlation (0.93) between PM10 and PM2.5 indicates **strong similitudes between the two variables**
- We interpolated the larges missing gaps **linearly**, by multiplying PM2.5 by the average of PM10/PM2.5
- It increased the correlation by a small value, but **we consider it still coherent** (0.95)

Removal of Outliers

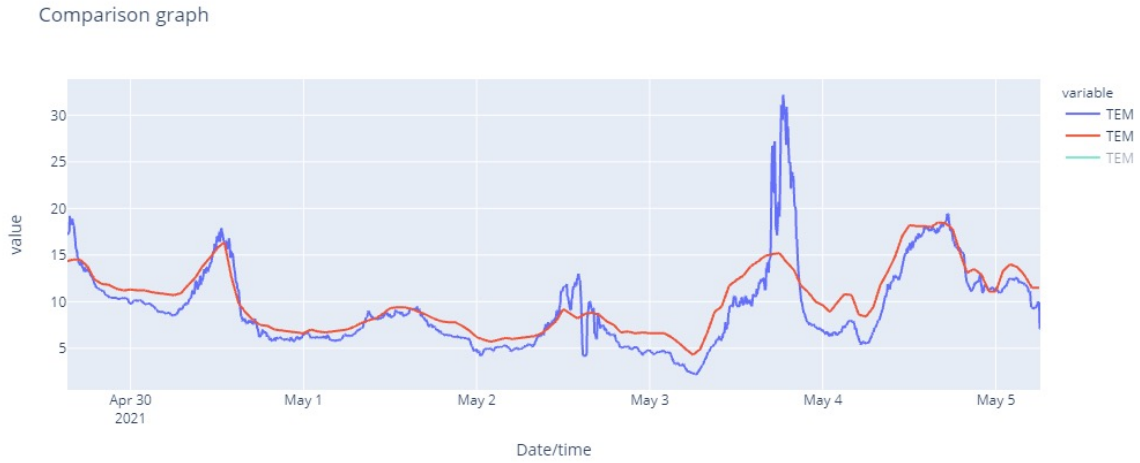
- Outliers are detected using **Standard Deviation** on a sliding window
- If the **Absolute Deviation** of the sample divided by the **Standard Deviation** of the interval is above a chosen threshold, the outlier is removed and then **linearly interpolated**
- This method has given good results, but may be slightly improved by using the median instead of the mean because it is less influenced by the outliers

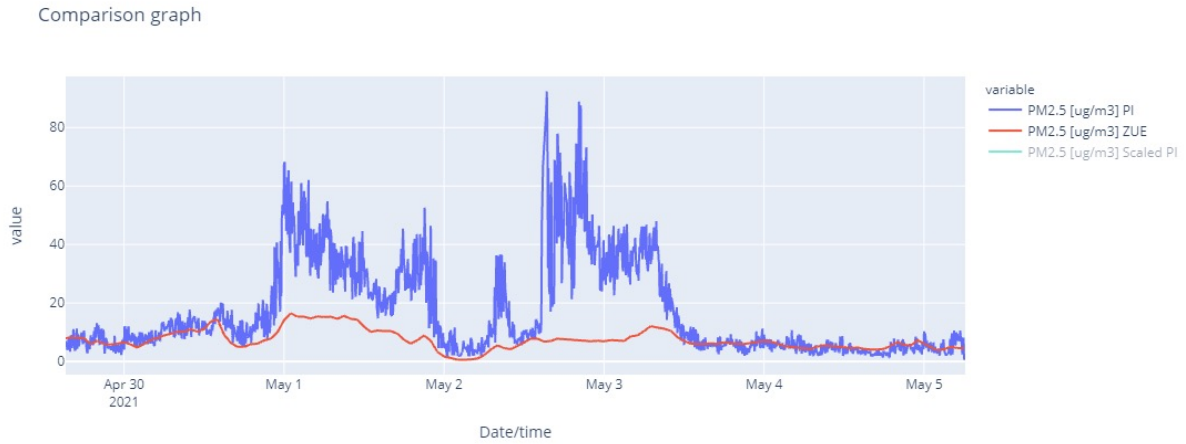


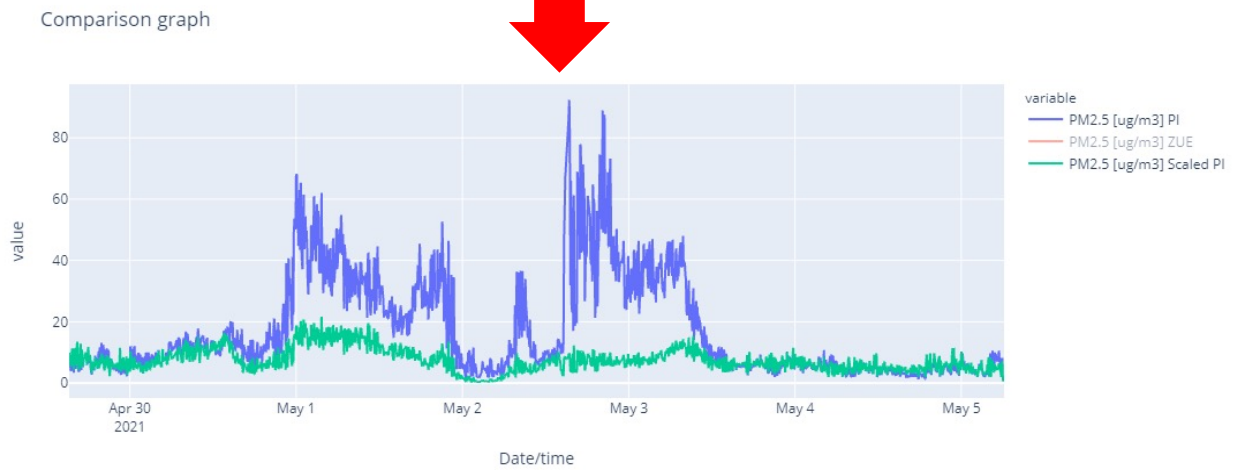


Rescaling the Raspberry Pi Data Using the NABEL Mean

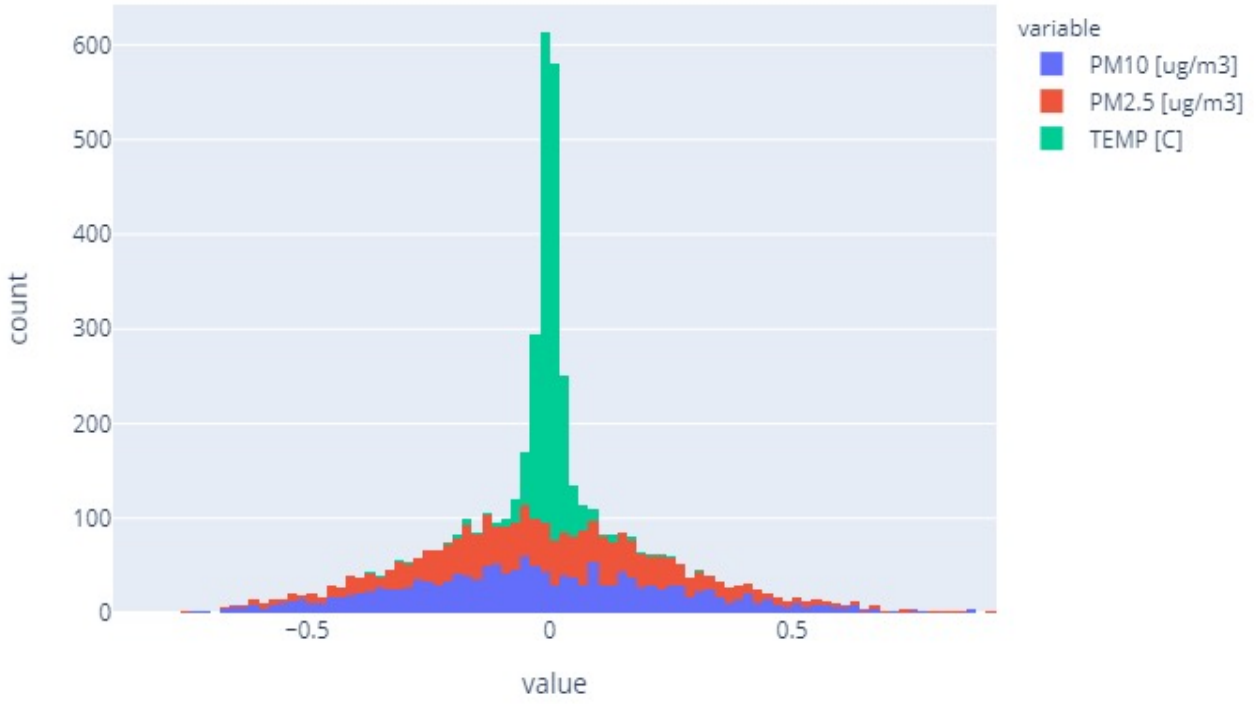
In order to create coherence between our measured data and the NABEL data, we rescale the Raspberry Pi measures with respect to the NABEL data, which corresponds to means on a 1-hour interval





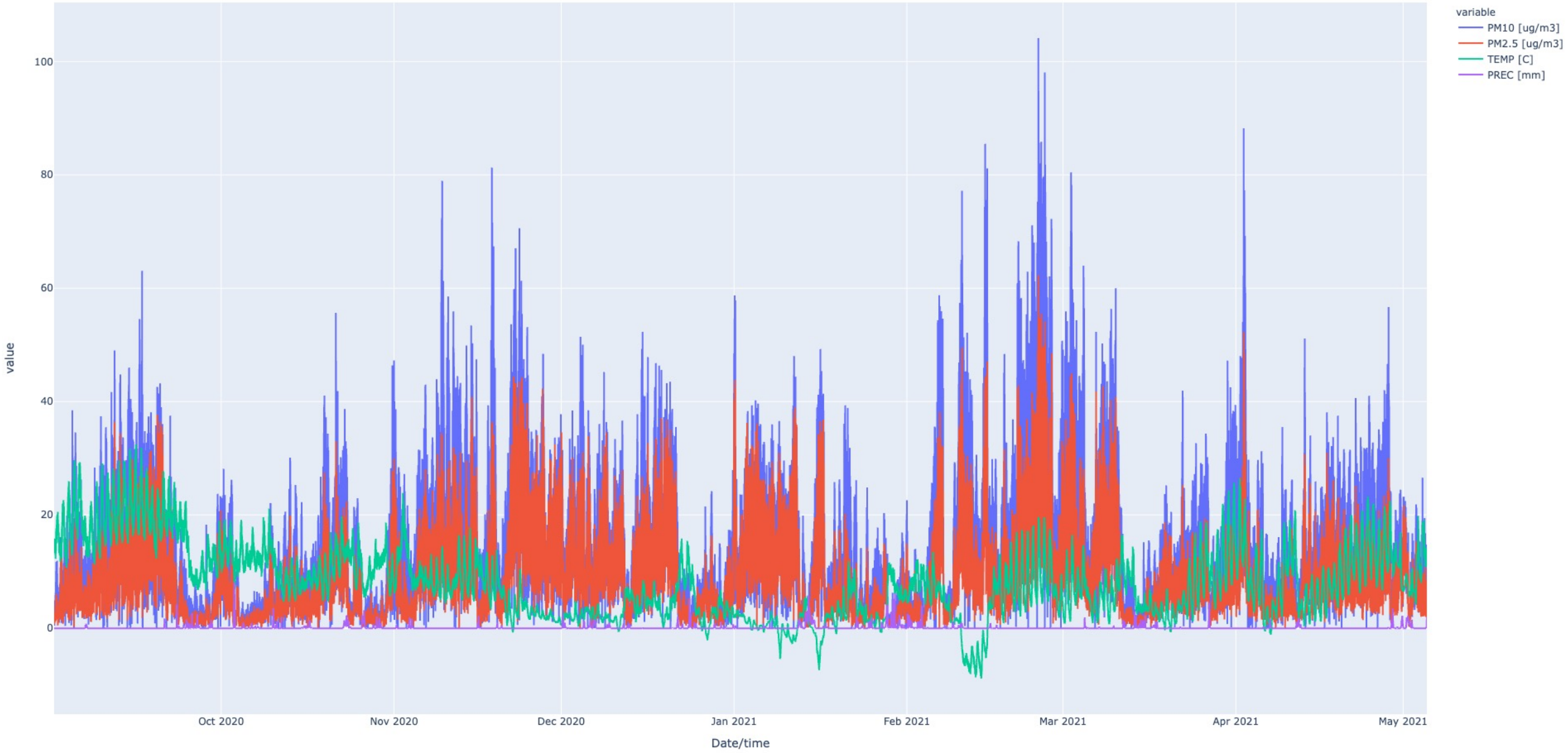


Adding Raspberry Pi Noise to the NABEL Data



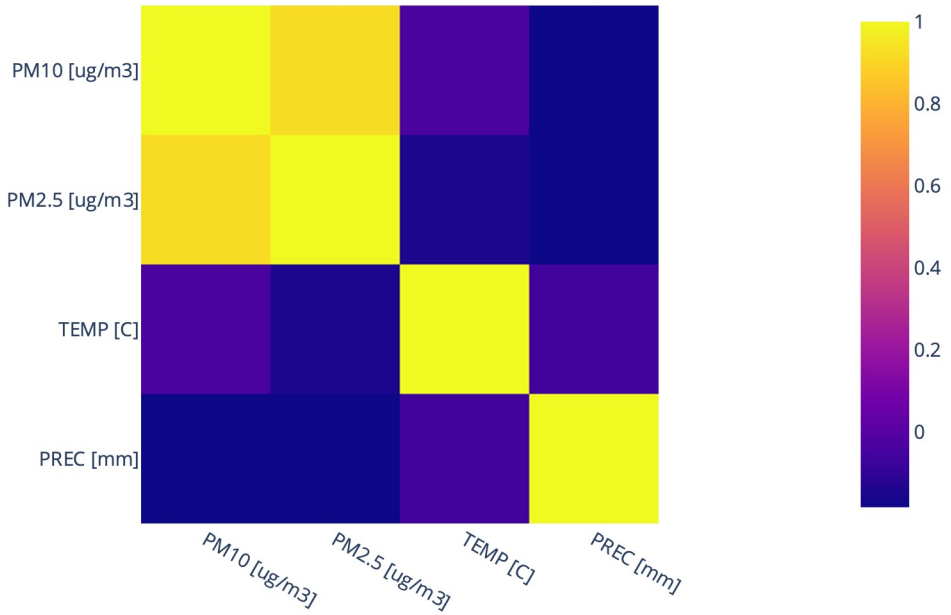
- We compute the variation of the rescaled Raspberry Pi noise around the NABEL Data
- The variation at each point is divided by the NABEL value at this point to compute a proportional deviation
- This proportional deviation takes the form of a Multivariate Gaussian Distribution, with PM10 and PM2.5 noise being strongly correlated
- We then added random noise with the same distribution to the NABEL Data for which we did not have corresponding Raspberry Pi Data
- The correlation between PM10 and PM2.5 after noising is 0.92, very close to our original correlation of 0.93

Imputed Dataset Visualization

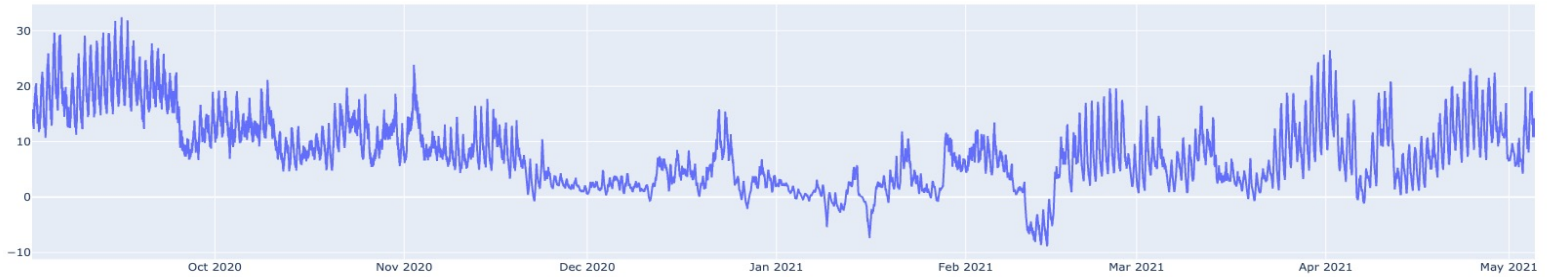


Imputed Dataset Visualization

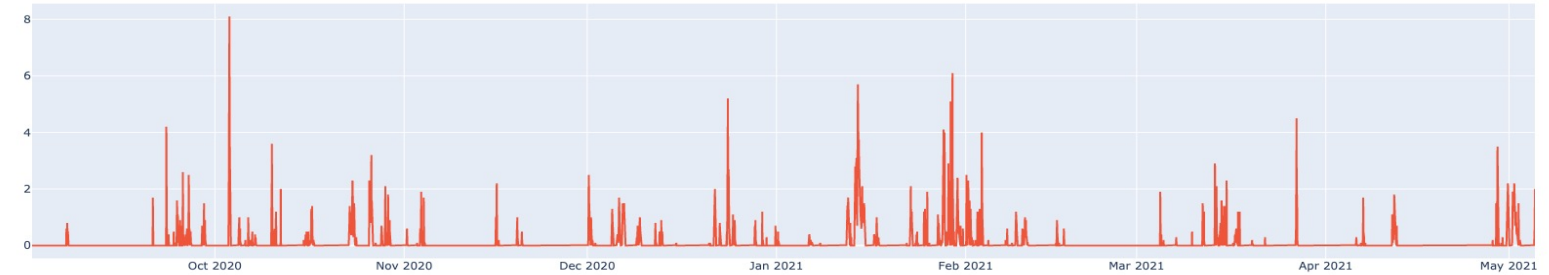
Correlogram of the imputed dataset and its variables



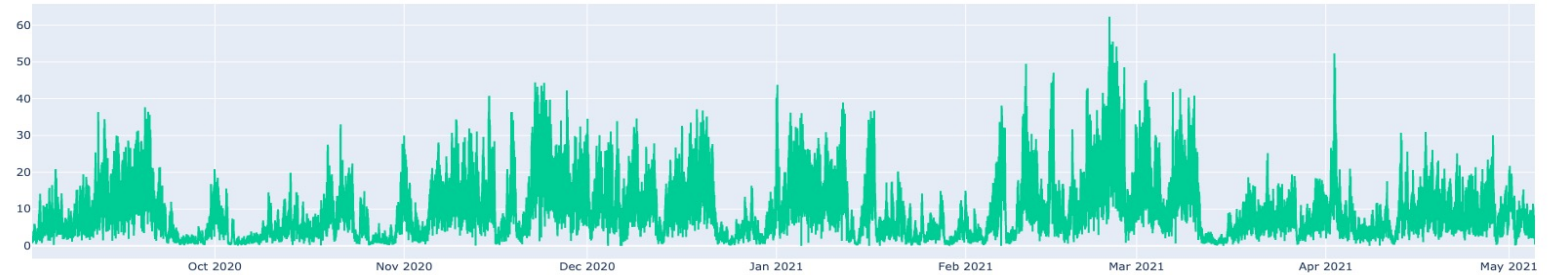
Temperature (in °C)



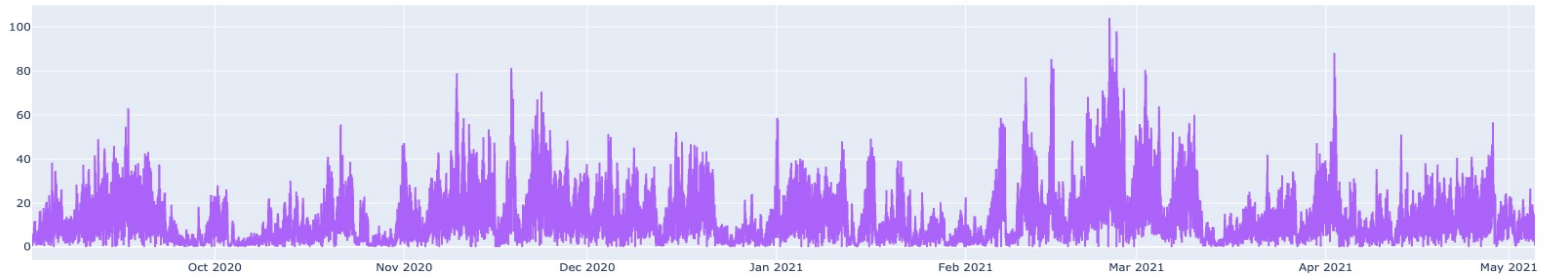
Precipitation (in mm)



PM2.5 (in µg/m3)



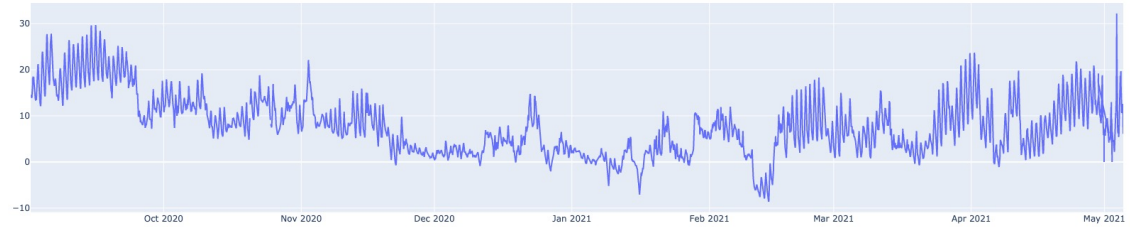
PM10 (in µg/m3)



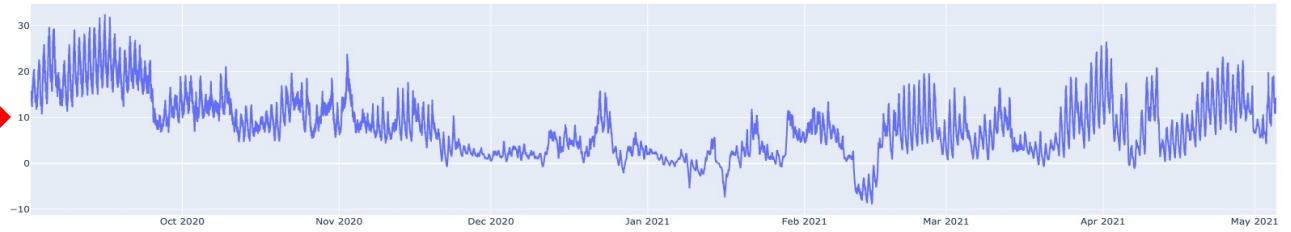
ETH zürich

Comparison with the Non-Preprocessed Dataset

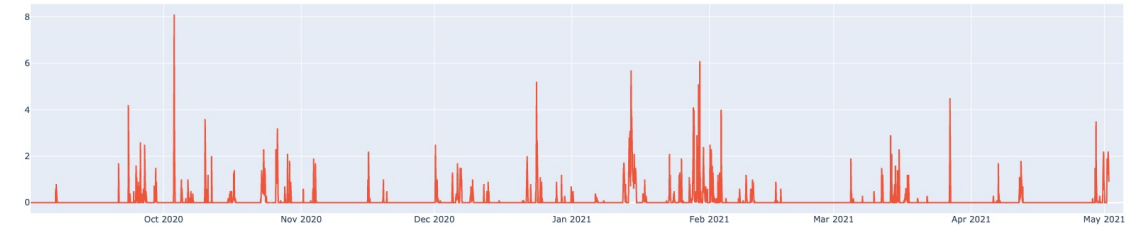
Temperature (in °C)



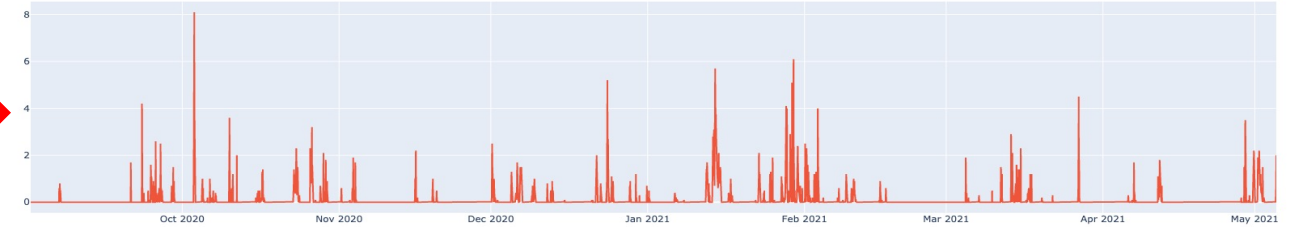
Temperature (in °C)



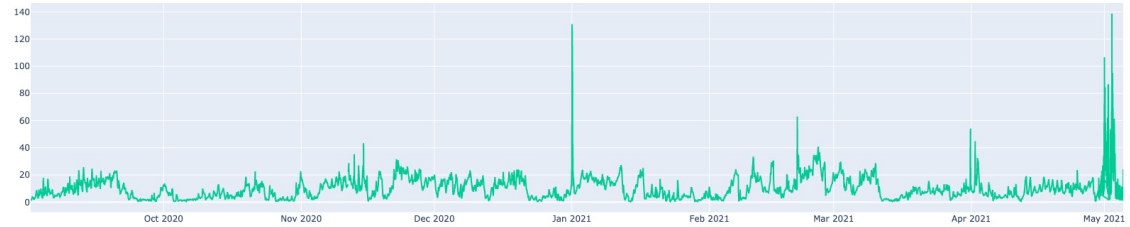
Precipitation (in mm)



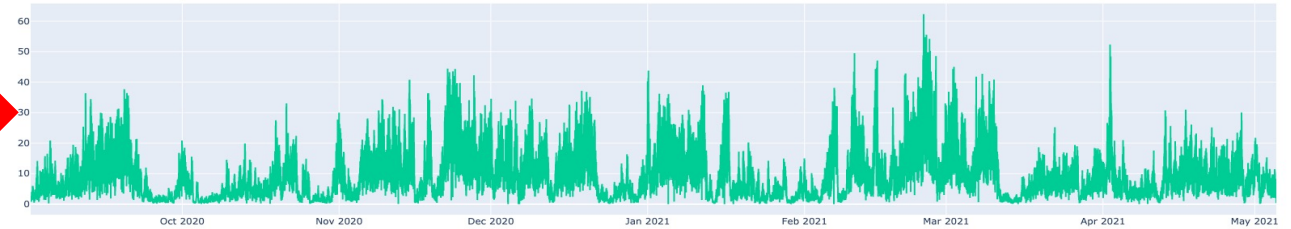
Precipitation (in mm)



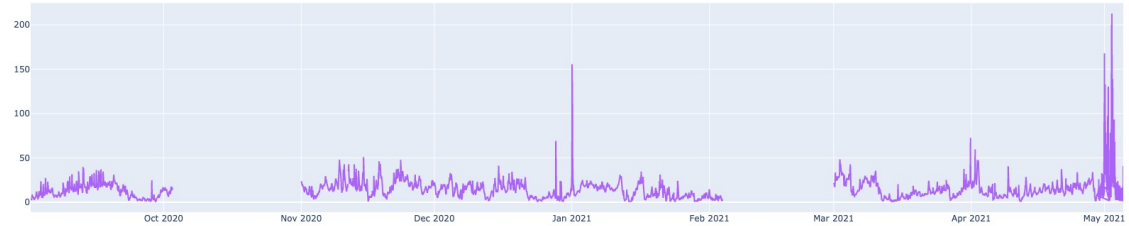
PM2.5 (in µg/m3)



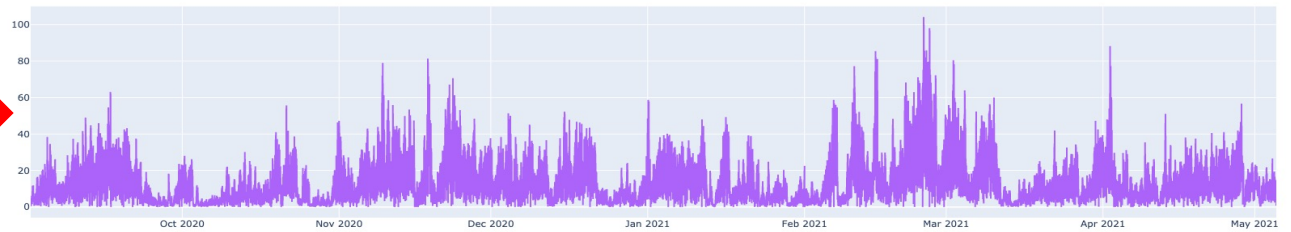
PM2.5 (in µg/m3)



PM10 (in µg/m3)



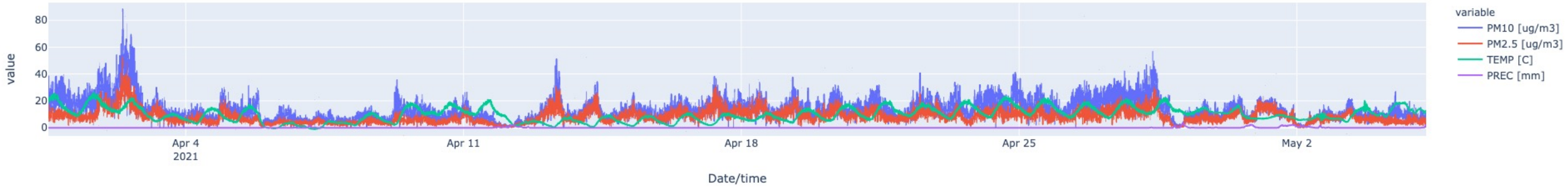
PM10 (in µg/m3)



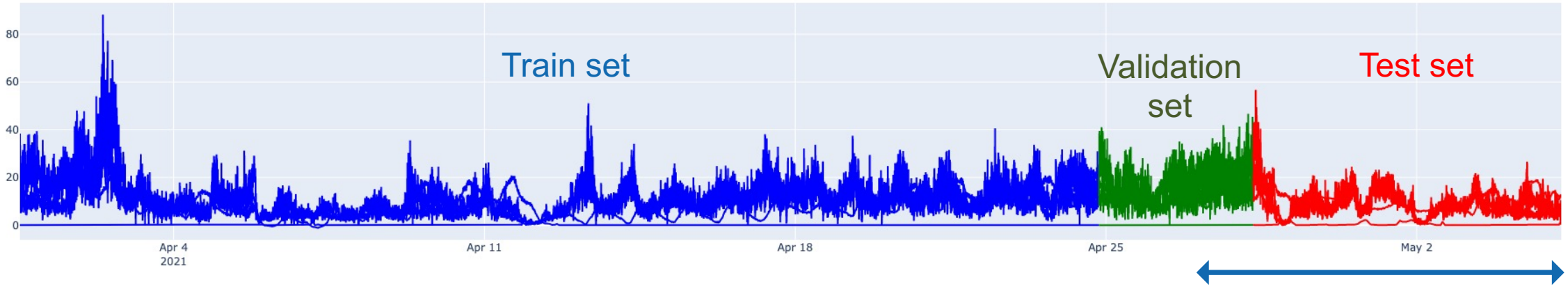
4. Forecasting PM2.5 Time-Series Using a Recurrent Neural Network (RNN)

Reducing the Dataset

- Our data after the pre-processing represents only ~2% of the dataset
- As only ~1’600 of the data comes from our measurements, we wanted it to be included in both **validation** and **testing** of our dataset
- For this purpose, we decided to take only the last **10’000** data samples to get the train-validation-test split of our data



- We divided the reduced dataset as follows : **70% for training**, **10% for validation** and **20% for testing**



Long Short-Term Memory (LSTM)

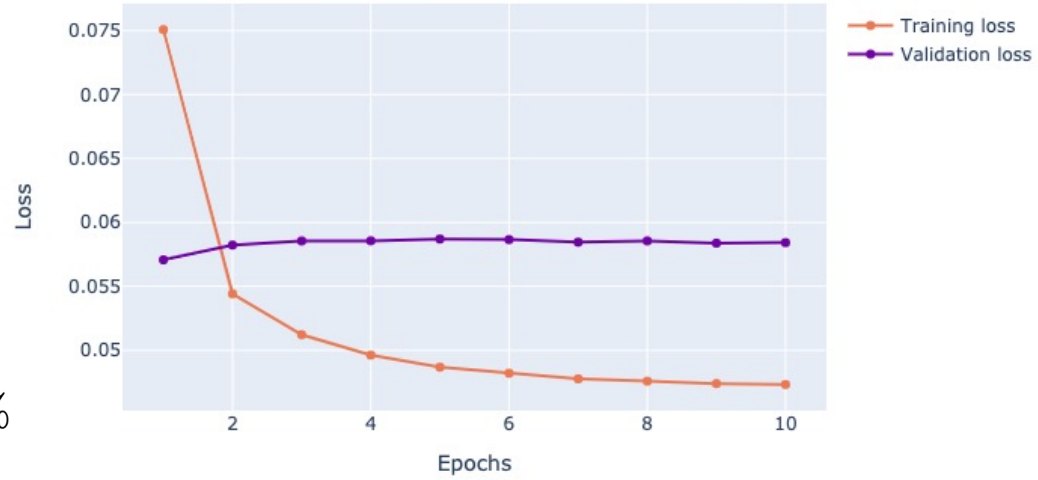
MAPE (in %)		Number of neurons in the input layer							
		1	2	10	25	50	100	150	200
Number of iterations (epochs)	1	51.43	46.65	72.70	81.14	82.53	68.70	58.91	57.50
	2	81.67	70.60	83.88	76.09	61.83	53.43	47.08	43.84
	10	86.17	63.66	42.62	40.41	38.26	35.50	35.02	33.79
	25	49.33	38.32	39.41	38.17	36.30	36.10	35.22	35.08
	50	37.75	38.69	38.50	36.84	35.68	35.64	35.59	35.36
	100	36.71	36.31	35.77	35.32	35.41	35.43	35.45	35.69
	150	36.67	35.23	36.48	35.64	35.52	35.65	35.81	35.86
	200	35.56	35.60	34.41	35.27	35.17	35.39	35.64	35.73

- Error metric : $MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{y_i - \hat{y}_i}{y_i}$
  - **Note**: This error metric cannot use actual values equal to 0
  - To prevent this, we decided to add 1 to each value, turning into this formula instead:

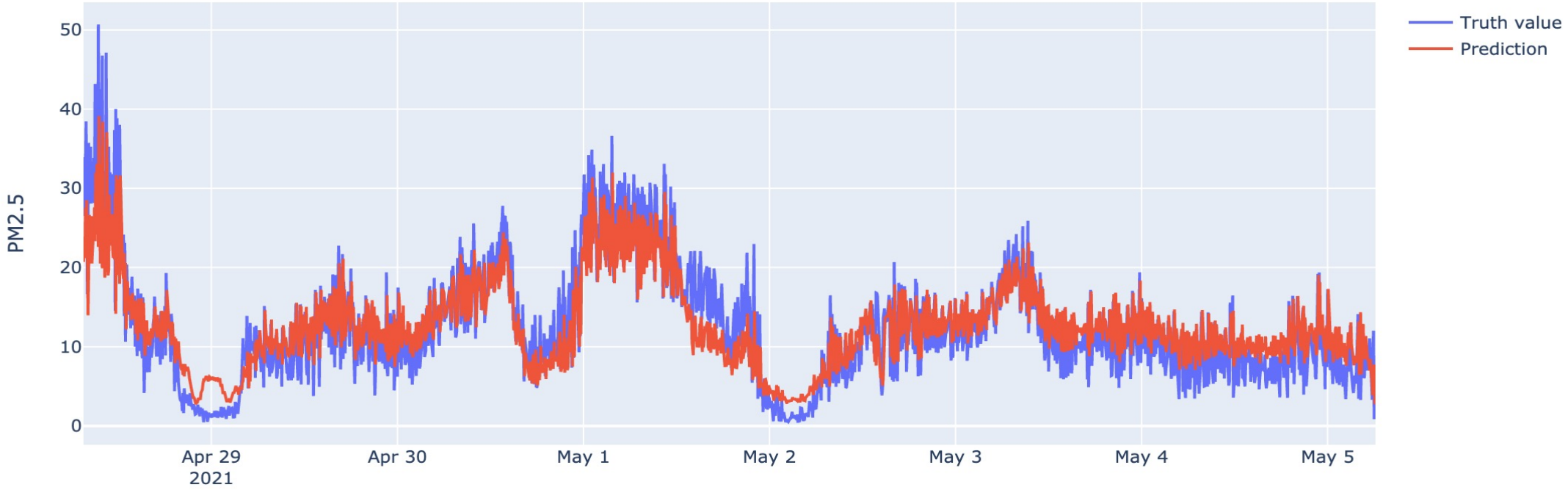
$$MAPE_{modified} = \frac{1}{n}\sum_{i=1}^{n}\frac{(y_i+1)-(\hat{y}_i+1)}{y_i+1} = \frac{1}{n}\sum_{i=1}^{n}\frac{y_i - \hat{y}_i}{y_i+1}$$

- Best number of epochs : **10**
- Best number of neurons in the input layer : **200**

Best Tested Parameters for RNN-LSTM



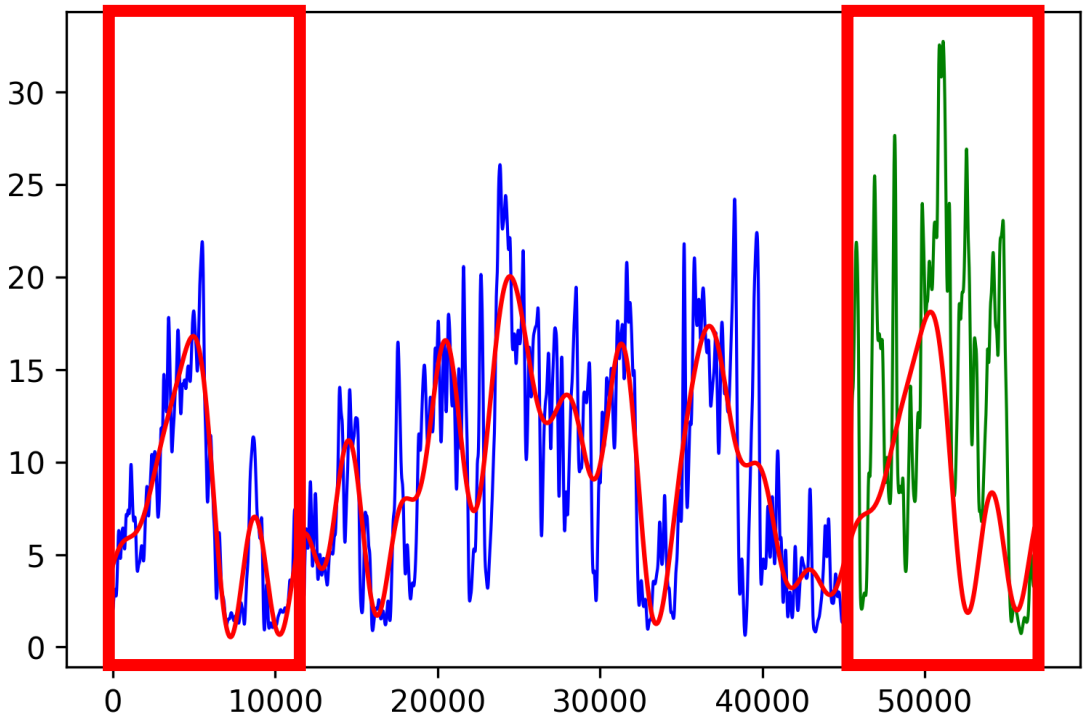
- Number of epochs : **10**
- Number of neurons in the input layer : **200**
- **Error percentage :** $MAPE_{modified} = \frac{1}{n}\sum_{i=1}^{n}\frac{y_i - \hat{y}_i}{y_i + 1} = 34.41\%$
- The prediction is performed on the test set after the model has been trained using training and validation sets



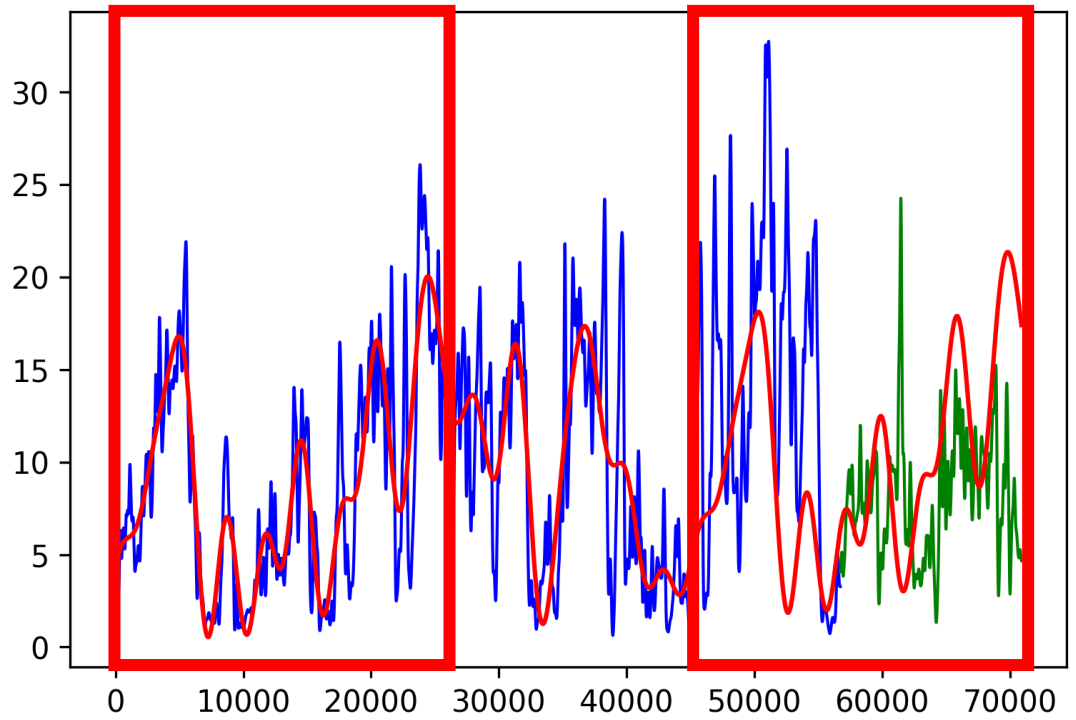
5. Comparison of Our Model with Other Time-Series Forecasting Methods

a. Data Prediction using Fast Fourier Transform

Data Prediction with FFT



Best Validation Loss - 58.74



Test Loss - 91.37

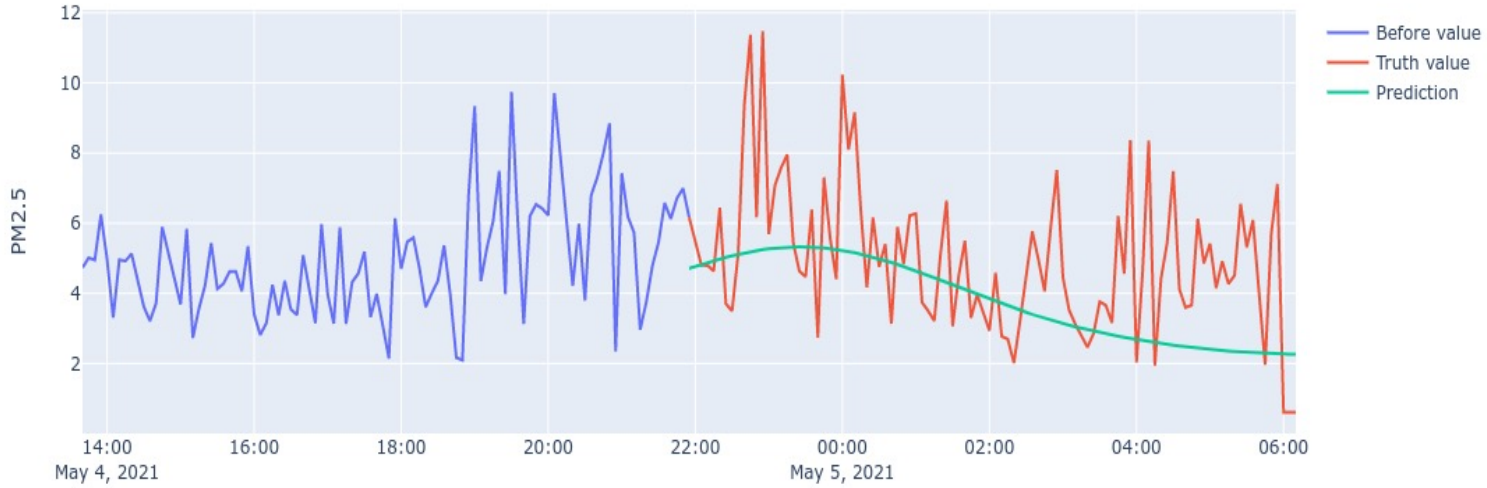
- We ran a cross-validation to find the optimal split between training and validation sets and the best "blurring" parameter, with MAPE Loss to compare it with the other models we trained

5. Comparison of Our Model with Other Time-Series Forecasting Methods

b. Data Prediction using Facebook Prophet

Data Prediction with Facebook Prophet

The Facebook Prophet model is for forecasting time series data based on an additive model where non-linear trends are fit with seasonality.



$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

Time series are decomposed with the following equation:

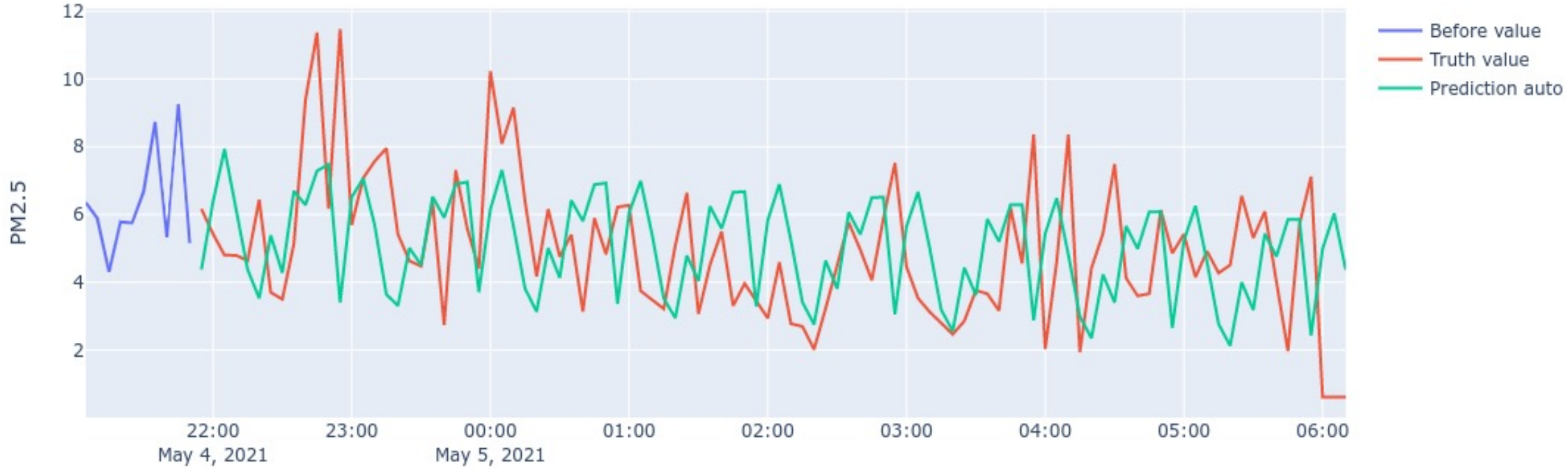
- $g(t)$ represents the trend function
- $s(t)$ the seasonality
- $h(t)$ potential **holidays** (irregular patterns that can occur over a period of time).

5. Comparison of Our Model with Other Time-Series Forecasting Methods

c. Data Prediction using SARIMAX

Best Tested Parameters for SARIMA

- Order: $(1, 2, 4)$
- Seasonality order : $(1, 1, 0, 12)$
- **Error percentage :** $MAPE_{modified} = \frac{1}{n}\sum_{i=1}^{n}\frac{y_i - \hat{y}_i}{y_i + 1} = 38.37\%$
- The prediction is performed on the test set (100 samples) after the model has been fitted on 900 previous values.



6. Conclusion

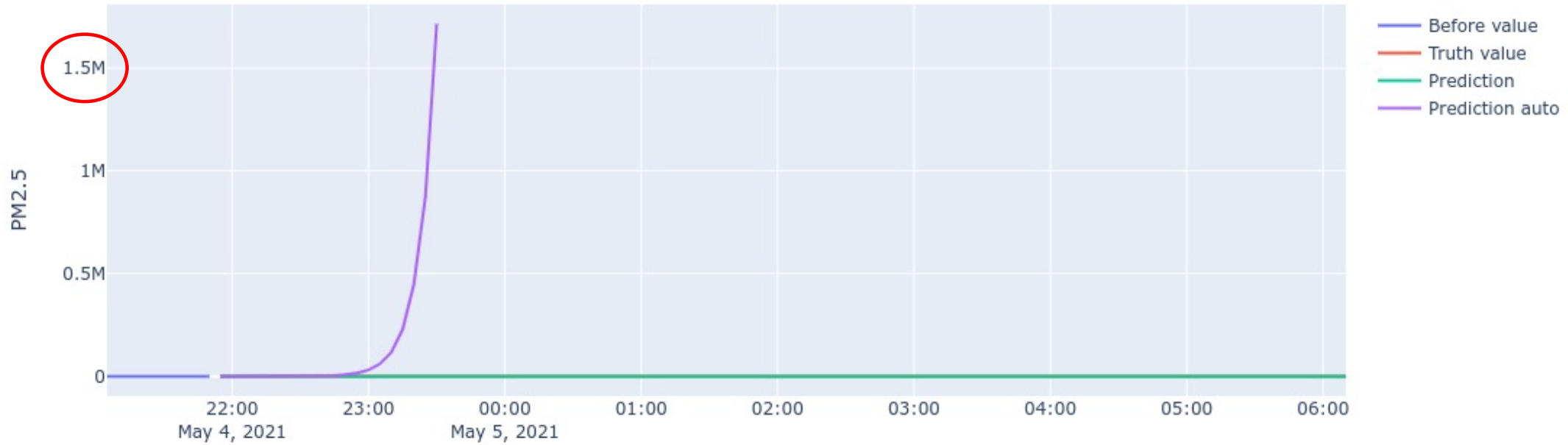
Conclusion

- The data must be gathered and used carefully
- The quality of the data has a lot of influence on the quality of the models
- **Bad data** → (likely) **bad results with a good model**
- Measured data can be quite different from the reality, due to among others:
  - Cheap hardware, which might give not very precise measurement
  - Sensors are put in places that are not 100% representative of the overall weather conditions (too exposed to the sun, too exposed to humidity…)
- Error was never below 34% for LSTM, which is still quite good but not totally reliable
- 100% accuracy is not possible when predicting air quality

One Last Example

End of the World on May 4th

- (Wrongly) Training models can lead to surprising results…



References

- J. Brownlee, *Multivariate Time Series Forecasting with LSTMs with Keras*, Machine Learning Mystery, 2017 (link)
- S. J. Taylor, B. Letham, *Forecasting at Scale*, Facebook Open Source, (link) (paper)
- R. C. Staudemeyer, E. Rothstein Morris, *Understanding LSTM -- a tutorial into Long Short-Term Memory Recurrent Neural Networks*, 2019 (link)

Thank you for your attention!

And a special thank you to Sachit whose help dealing with the Raspberry issues was really appreciated and useful