Predicting chemicals concentration in water streams using Gradient Boosting Regressor

Pierre Bouquet, Haojun Zhu, Eliot Walt
December 17, 2020

Abstract

Our team developed a model to predict the concentration of key elements in the Severn river in Wales based on continuous time and cheap to sample features. This project was directed by Paolo Benettin from the Laboratory of ecohydrology - ECHO at EPFL. We trained our model on water quality data from Plynlimon UHF, UK between 1984 up until 2007. We then predicted elements of interest's concentration for the year 2008.

I Introduction

The quality of river water is determined by the concentration of several chemicals. Researchers want the highest possible resolution in the measurements which translates to a high sampling frequency. However, high frequency sampling processes are often too expensive and can't be afforded by research laboratories. This issue is typically solved by sampling at low frequency and approximating the in-between (high frequency) values by linear interpolation. This method is sub-optimal as the elements concentration varies a lot due to being driven by highly dynamic natural mechanisms such as rainfall. Therefore, Dr. Benettin suggested to develop a machine learning model to predict the concentration of the following elements:

- Chloride - Cl [mg/l]
- Silicon - Si [mg/l]
- Nitrate - NO_3 [mg/l]
- Sulfate - SO_4_by_ICP [mg/l]
- Sulfate - SO_4 [mg/l]
- Dissolved Organic Carbon - DOC [mg/l]

Based on the folllowing features:

- Water flux [mm/hr]
- Logarithm of the flow [/]
- Conductivity [uS/cm]
- pH
- Day number of the year
- Month number of the year

All the above features can easily be sampled at high frequency (7 hours) and are strongly correlated to the elements we want to predict based on Dr. Benettin insights. The overall idea is shown in the graph below

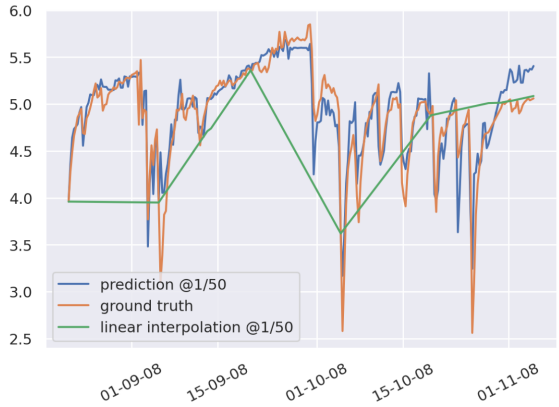

Figure 1 - Plot of the Chloride concentration with the real measurements (orange), the predicted concentration (blue) and the linear interpolation (green) at a sampling frequency T_s = 7h and interpolation frequency of T_s = 14 days

II Data pre-processing

Data Analysis We were given 2 data sets:

- Dataset_1 : Data from 1990 to 2011 sampled every two weeks with every water properties.
- Dataset_2: Data from 2007 to 2009 sampled every 7 hours with every water properties.

The water properties include 49 elements and 4 features that we use to predict them.

Feature creation As our data showed a high yearly periodicity we created the features month_nb and day_nb base on sinus function scaled on the day/month number of the year.

$$tx_{day} = \sin(\frac{nb_{day} * 2\pi}{365}) \quad (1)$$

$$tx_{month} = \sin(\frac{nb_{month} * 2\pi}{12}) \quad (2)$$

We also added the logarithm of the flow as it has extremly quick variation due to the rain and the logarithm is therefore a good indicator.

$$flow_{log} = \log_{10}(flow) \quad (3)$$

Standardization The data was standardized to have zero mean and a standard deviation of 1 before training the model. Let $\mathbf{x}_i$ be the i-th training sample, $\mu_X = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x_i}$ be the empirical average of the training set and $\sigma_X^2 = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x_i} - \mu_X)^2$ be the empirical variance of the training set. Then,

$$\tilde{\mathbf{x}}_i = \frac{\mathbf{x}_i - \mu_X}{\sigma_X^2} \tag{4}$$

where the division is computed element-wise.

III Benchmark

RMSE In order to evaluate the correctness of our model we calculate the Root Mean Squared Error between our predicted values and the expected one as follows:

$$RMSE = \sqrt{\sum_{i=0}^{N}\frac{(\hat{\mathbf{y}}_i - \mathbf{y}_i)^2}{N}} \tag{5}$$

With:
-$\hat{y_i}$: Predicted value
-y_i: Real value

Error As the RMSE values depend on the range of y and $\hat{y}$, we also considered the normalized RMSE, denoted by e, which allows to directly compare the performance of models that predict different elements.

$$e = \frac{RMSE}{\mu_Y} \tag{6}$$

where μ_Y is the empirical average of the real values, $\frac{1}{N}\sum_{i=1}^{N} y_i$.

Linear interpolation The usual way of measuring water quality is by using linear interpolation at given time frequencies. Therefore, in order to determine the efficiency of our model we started by creating a linear interpolation model. It takes different sampling frequencies and a range of time to start the sampling in the data set to give the corresponding normalized RMSE as a measure of the performance of the interpolation model with the gives frequency and starting point in the data set. The table below demonstrates the influences of the sampling frequencies to the linear interpolation.

	Cl	Si	NO_3	SO_4	$SO_{4,ICP}$	DOC
7D	20.1	26.2	29.6	21.4	18.5	54.7
14D	24.8	30.8	37.1	25.9	22.4	61.4
28D	25.7	32.1	37.9	27.0	23.5	68.5

Table 1.Error in percentage of the linear interpolation for 7, 14 and 28 days sampling frequencies

IV Model

Model selection The model selection was driven by two main factors. First, the data set was of moderate size. In particular, due to many missing values, extracting complete time series instead of single samples resulted in limited training sets. Secondly, the model needed to take advantage of any new measurements by adding it to the data set and retraining. The goal of the project being to decrease the necessary sampling frequency in the long term, treating the input as high frequency time series rather than single sets of measurements seemed counterproductive.

	Cl	Si	NO_3	SO_4	$SO_{4,ICP}$	DOC
train	1811	1823	1796	1135	1832	1814
test	452	455	448	283	458	453

Table 2. Size of the train and test sets for target elements after removing missing values

In practice, two types of models were considered ; Convolutional Neural Networks (CNN) and Gradient Boosting Regressors. The CNN approach proved itself ineffective, suffering from the restricted amount of data available. Furthermore, The data samples had to be turned into fixed frequency time series which would have required a new high frequency dataset to retrain. The CNN models were therefore discarded. Gradient Boosting Regressors are boosting models and are often used for multi-variate regression tasks as they are cheap to train, require less data than Neural Networks and still display excellent performance. The regression happens between a single set of features to one target element, allowing any new measurement to be added to the training set to improve the model over time. One drawback compared to Neural Networks is that each chemical is predicted by a dedicated model, while a single model can be used with Neural Networks.

Gradient Boosting Machines Gradient Boosting Machines (GBM) [1] are a type of Boosting models in which a regression function $\hat{f}(\tilde{\mathbf{x}})$ is expressed as a sum of M individual regressors $\hat{f}_i(\tilde{\mathbf{x}})$.

$$\hat{f}(\tilde{\mathbf{x}}) = \sum_{i=1}^{M}\hat{f}_i(\tilde{\mathbf{x}}) \tag{7}$$

In practice, the function $\hat{f}(\tilde{\mathbf{x}})$ is greedily grown. At iteration t, the model $\hat{f}_t(\tilde{x})$ consists of the the sum of the previous model $\hat{f}_{t-1}(\tilde{\mathbf{x}})$ and a weighted weak learner.

$$\hat{f}_t(\tilde{\mathbf{x}}) = \hat{f}_{t-1}(\tilde{\mathbf{x}}) + \alpha_t h(\tilde{\mathbf{x}}, \theta_t) \tag{8}$$

where α_t is the weight associated with the weak learner $h(\tilde{\mathbf{x}}, \theta_t)$ of parameters θ_t.
Given a loss function $\mathcal{L}$ and a training data set

$\{(\tilde{\mathbf{x}}_i, \mathbf{y}_i)\}_{i=1}^N$, the parameters α_t and θ_t minimize the sum of the losses associated to $\hat{f}_t(\tilde{\mathbf{x}_i})$, $\forall i = 1, 2, ..., N$.

$$(\theta_t, \alpha_t) = \arg\min_{(\theta,\alpha)} \sum_{i=1}^{N} \mathcal{L}(\mathbf{y_i}, \hat{f}_{t-1}(\tilde{\mathbf{x}_i}) + \alpha h(\tilde{\mathbf{x}_i}, \theta)) \quad (9)$$

Gradient Boosting Regressors Gradient Boosting Regressors (GBR) are a particular case of GBM in which the weak learners are regression trees [1]. Regression trees are themselves additive models of the form

$$h(\tilde{\mathbf{x}}, \{b_j, R_j\}_{j=1}^J) = \sum_{j=1}^{J} b_j 1(\tilde{\mathbf{x}} \in R_j). \quad (10)$$

The $\{R_j\}_1^J$ parameters represent disjoint regions of space and correspond to the J leaves of the regression tree. The $\{b_j\}_1^J$ are the coefficients of the non-terminal nodes and define the shapes of the $\{R_j\}_1^J$. The update (8) can be rewritten as

$$\hat{f}_t(\tilde{\mathbf{x}}) = \hat{f}_{t-1}(\tilde{\mathbf{x}}) + \alpha_t \sum_{j=1}^{J} b_{jt} 1(\tilde{\mathbf{x}} \in R_{jt}). \quad (11)$$

Or, equivalently,

$$\hat{f}_t(\tilde{\mathbf{x}}) = \hat{f}_{t-1}(\tilde{\mathbf{x}}) + \sum_{j=1}^{J} \gamma_{jt} 1(\tilde{\mathbf{x}} \in R_{jt}). \quad (12)$$

Where, γ_{jt} can be expressed using (9) as

$$\gamma_{jt} = \alpha_t b_j = \arg\min_{\gamma} \sum_{\tilde{\mathbf{x}_i} \in R_{jt}} \mathcal{L}(\mathbf{y_i}, \hat{f}_{t-1}(\tilde{\mathbf{x}_i}) + \gamma). \quad (13)$$

Our models are based on L2-loss, $\mathcal{L}(y, f) = (y - f)^2$. Therefore, the previous equation can be expressed as

$$\gamma_{jt} = \arg\min_{\gamma} \sum_{\tilde{\mathbf{x}_i} \in R_{jt}} \left[\mathbf{y_i} - \hat{f}_{t-1}(\tilde{\mathbf{x}_i}) - \gamma\right]^2 \quad (14)$$

V Training

Train and test dataset GBR models allow to perform multivariate regression to a single continuous value. Therefore, for each element to predict, we started by building disjoint training and testing set. First, the two data sets described in section II were concatenated. Then, we selected the entry of the data set corresponding to the features listed in the introduction and one of the chemicals to predict. After removing the missing values, we selected the first 80% of the data for training and the rest for test. The mean and standard deviation were then computed on the train set and used to standardize both sets. Finally, features and target (element to predict) were assigned to separate matrices to get the final sets of the form $S_{train} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ and $S_{test} = \{(\mathbf{x}_i, y_i)\}_{i=1}^K$ with $\mathbf{x}_i \in \mathbb{R}^6$ and $y_i \in \mathbb{R}$.

Model implementation The Gradient Boosting Regressor class of the scikit-learn machine learning python library was used to implement the GBR model. The main parameters are:

- **n_estimators**: Total number of regression trees constituting the model.
- **learning_rate**: Size of the gradient steps when fitting regression trees.
- **max_depth**: Maximal depth of the regression trees.

Hyper-parameter selection The values of the n_estimators and learning_rate parameters were determined by grid-search among $[125, 250, 500, 1000]$ for n_estimator and $[1, 0.5, 0.25, 0.1, 0.05]$ for learning_rate. To ensure the selected values consistently produced good results, 10-fold cross validation was conducted and the mean RSME values were considered. The default max_depth value of 3 was used.

	Cl	Si	NO_3	SO_4	$SO_{4,ICP}$	DOC
n_estimators	500	500	500	1000	500	500
learning_rate	0.1	0.05	0.05	0.05	0.05	0.1

Table 3. Hyper-parameters selection for each element's GBR model

VI Results

The limited set of features considered allowed to accurately predict the evolution of some chemicals. The errors and confidence intervals (CI) of our models for the main elements can be found in Table 4.

	Cl	Si	NO_3	SO_4	$SO_{4,ICP}$	DOC
CI ($\pm$X)	0.32	0.12	0.046	0.136	0.201	1.25
Error [%]	6.38	8.64	27.28	6.77	10.46	42.7

Table 4. Model results for the elements of interest

The graphs presenting the result for each element are given in Appendix 1

Comparison with the usual method The following table shows the equivalent sampling frequencies required to achieve the same performance as our models using linear interpolation.

	Cl	Si	NO_3	SO_4	SO_4ICP	DOC
Ts	12h	12h	5.37d	12h	28h	2.62d

Table 5. Required sampling frequencies to achieve same performance as our models using linear interpolation

Features contribution analysis We developed a simple analysis of the features contribution to the prediction of each element. To do so, we trained six different models, one for each element listed in section I, using only one of the four features. In order to analyze the importance of each feature, we computed the test error of each models and ranked them in increasing order. The following table shows the resulting features ranking for each major elements.

	1st	2nd	3rd	4th	5th	6th
Cl	C	F	E	D	A	B
Si	A	B	D	E	F	E
NO_3	E	F	C	D	A	B
SO_4	B	A	F	C	D	E
SO_4, ICP	C	F	E	A	B	D
DOC	A	B	C	D	F	E

Table 6. Ranking of the most influential features to predict the elements of interest

With:
-A: Water flux [mm/hr]
-B: Logarithm of the flow [/]
-C: Conductivity [uS/cm]
-D: pH
-E: Day number of the year
-F: Month number of the year

Independence of the model The higher the sampling time, the more independent the model as it has less constraints to the real values.
Therefore, increasing the sampling frequency will slightly weaken its accuracy but it tends to converge very quickly ($Ts = 14D$) to a fix error when the sampling frequency decreases. On the other hand, this is not the case for the benchmark accuracy.
Also every new sample improves the accuracy of the model which mean that it gets stronger over time.

VII Additional results

Other accurately predicted elements As we trained a model for all the elements in the data set, it turned out that some other elements were accurately predicted. The following is a list of elements for which the prediction error e is $< 20\%$ ($T_s = \infty$).

- Lithium - Li [ug/l] - e = 13.74%
- Sodium - Na [mg/l] - e = 5.55%
- Magnesium - Mg [mg/l] - e = 7.66%
- Sulfur - S [mg/l] - e = 10.143%
- Calcium - Ca [mg/l] - e = 9.37%
- Strontium - Sr [ug/l] - e = 12.7124%

Appendix 2 presents graphical results with our models for the above elements.

VIII Conclusion

Using our machine learning model provides a better view on the continuous time concentration of all the expected chemical presented in I and also some extra ones presented in VII. Our approach is more accurate and less expensive than the usual methods.
Therefore, one who is interested in predicting chemicals concentration should consider using Machine Learning algorithm if sampling at high frequency is not possible.

References

[1] JH. Friedman, Greedy boosting approximation: a gradient boosting machine. The Annals of Statistics 2001, Vol. 29, 1189–1232. doi: 10.1214/aos/1013203451

Appendix 1

Plot of the key chemicals' concentrations with the real measurements (orange), the predicted concentration (blue) and the linear interpolation(green) at a feature sampling frequency Ts= 7h and interpolation frequency of Ts= 28 days which is the usual sampling frequency for laboratories and NGO.

Chloride - Cl [mg/l]

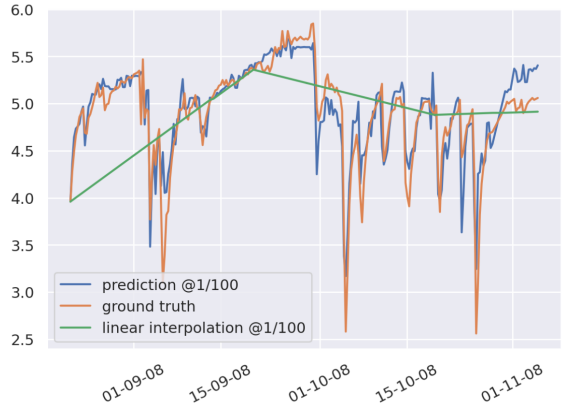

Figure 2 - Plot of the Chloride concentration

Silicon - Si [mg/l]

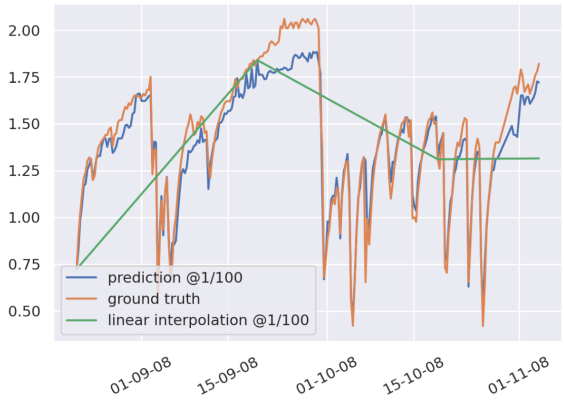

Figure 3 - Plot of the Silicium concentration

Nitrate - NO3[mg/l

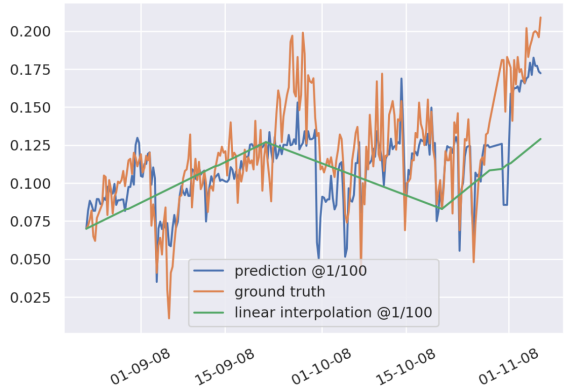

Figure 4 - Plot of the Nitrate concentration

Sulfate - SO4 by ICP [mg/l]

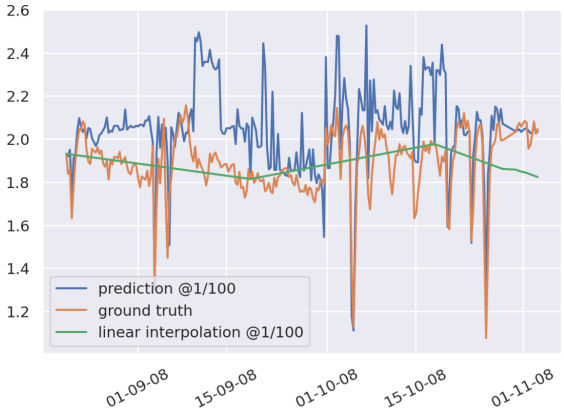

Figure 5 - Plot of the Sulfate concentration

Sulfate - SO4[mg/l]

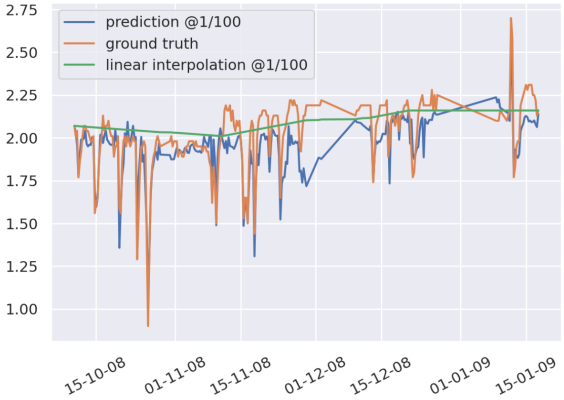

Figure 6 - Plot of the Sulfate concentration

Dissolved Organic Carbon - DOC [mg/l]

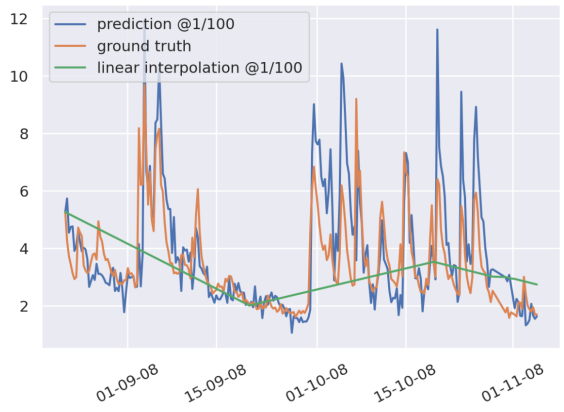

Figure 7 - Plot of the Dissolved Organic Carbon concentration

Appendix 2

Plot of the well predicted (e<20%) chemicals concentration with the real measurements (orange), the predicted concentration (blue) and the linear interpolation(green) at a feature sampling frequency $T_s = 7h$ and interpolation frequency of $T_s = 28$ days which is the usual sampling frequency for laboratories and NGO.

Lithium - Li [ug/l]

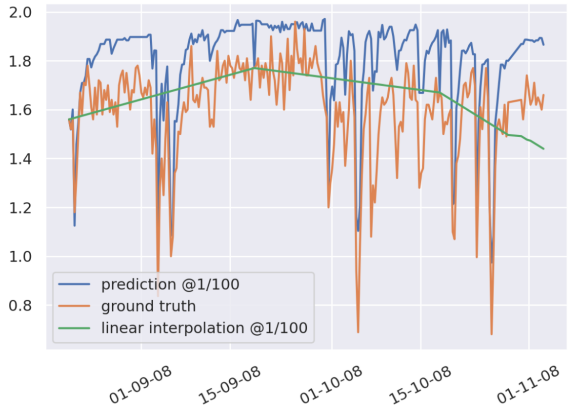

Figure 8 - Plot of the Lithium concentration

Sodium - Na [mg/l]

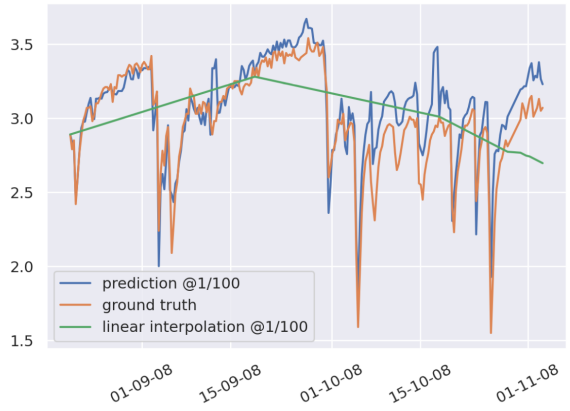

Figure 9 - Plot of the Sodium concentration

Magnesium - Mg [mg/l]

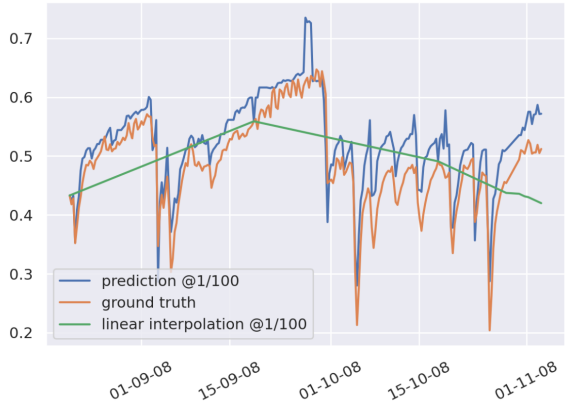

Figure 10 - Plot of the Magnesium concentration

Sulfur - S [mg/l]

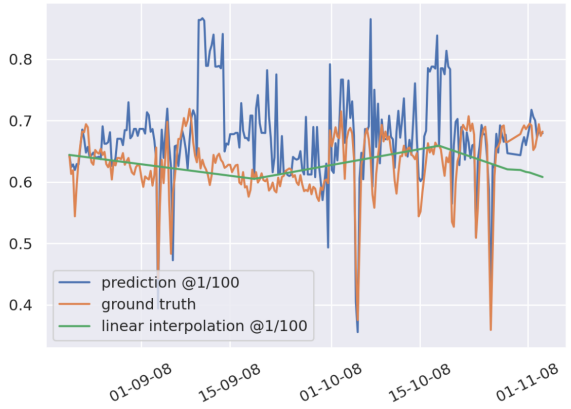

Figure 11 - Plot of the Sulfur concentration

Calcium - Ca [mg/l]

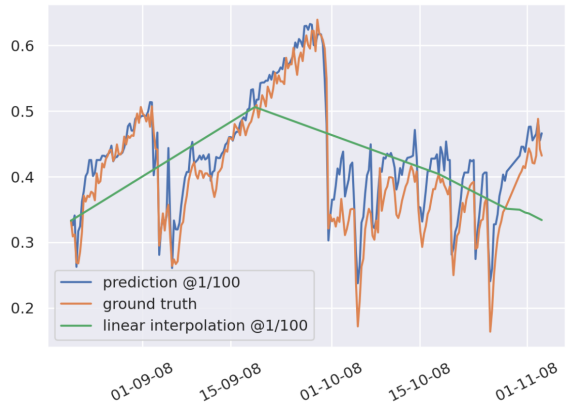

Figure 12 - Plot of the Calcium concentration

Strontium - Sr [mg/l]

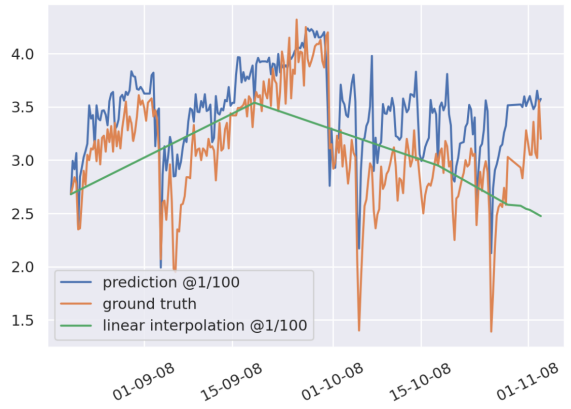

Figure 13 - Plot of the Strontium concentration