Project Seminar
Flood Prediction at German corner in Koblenz

Long Phan [1],

Supervisor: Prof. Dr. Thomas Goetz [1]

[1]Department of Mathematics, University of Koblenz, Germany
*E-mail: ldphan@uni-koblenz.de

Abstract

Flooding events have been happening occasionally especially when heavily raining or affection of bad weather condition. The aim of this project seminar is to understand history data of water level measured during the flood periods and concurrently develop methods and tool to support the decision-maker.

1 Introduction

Flooding has been always being the known natural disaster around the world during history until then. Events in the year 1993, 1995, 2011 under report confirmed the flood at city Koblenz. In our use case, the location is given at German corner and in extreme weather condition the decision-maker is required to give a fast decision (e.g. call the crane to remove the container and its facilities) at the right time. A decision too early or late would cause a lot of costs and damages overall. To achieve our goal, firstly we proceed step by step by analysing issue and understanding the given data delivered by the city, then choosing the appropriate meta data to derive the necessary information and after all developing the approaches to apply these information into calculation tasks of the water level at later time.

Figure 1: High water in Koblenz 1993.

2 Flood issue and data understanding

Following the given data, the position of container is located on the land with height 67 meters to sea surface level and **PNP's**[1] position around this location is determined with 57 meters. So the distance between the PNP and the container is about 10 meters (67 - 57). So the safe distance must be at least $<$ 10 meters to keep water level away from position of container. Based on the given data of the measuring water level, we confirmed that many times in history the location of container has been flooded when water level was rising greater than 10 meters (see: figure 2). The first question comes up that how do we know at what water level should we start concerning so that the first algorithm has been developed to classify the water level into 3 controlling zones: **"Green (safe), Yellow (warning) and Red (alarm)"**. After we have classified the water level, we will able to identify whether or not the water level at the current time need to be concerned and the predicting calculate water level might be in danger zone or still in safe range.

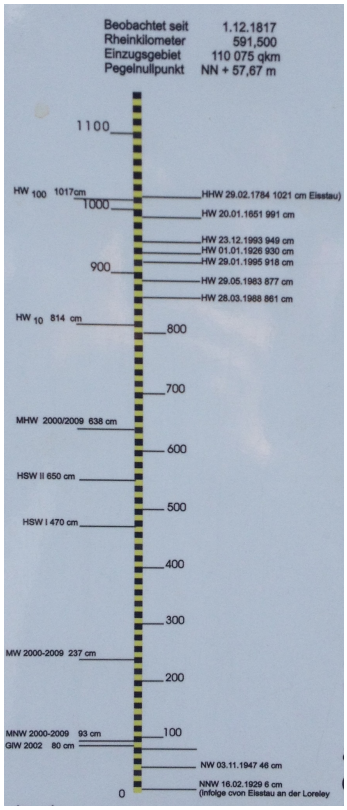

Figure 2: Pegel Koblenz Rhein.

There are 3 important parameters which will be used during the whole work:

The current water level, the current time point and the predict hours. These 3 parameters will be input by user to keep an connection between history data and the reality which is happening outdoor. Next we will step by step apply this first achieved information into the methods to find out the final answer for the question: at what water level should the decision-maker call the crane to remove the container ?

[1] https://de.wikipedia.org/wiki/Pegel

3 Approaches

Method 1

The idea is to classify data to identify at what water level would be considered dangerous, warning or safe by calculating the distance of every point to the major pre-defined points (MAX_waterlevel, MIN_waterlevel, Average_waterlevel). E.g. if the distance of water point to Min_waterlevel is smallest comparing to other points (MAX_waterlevel and Average_waterlevel), this water point will belong to the Green zone and so on. By applying this way to all measured water point, we then know which group every water point belongs and the range of every water zone (Red, Yellow, Green) and finally achieve the first conclusion that at what water level we should start concerning. Result will be 3 lists illustrated 3 groups (Red, Yellow, Green). Using the visualization technique will show us clearly the zones.

Algorithm 1 Data Classification

```
1:  dist_red = dist_yellow = dist_green = 0 {initiate distance = 0}
2:  for elem in allwaterlevelpoints do
3:      dist_red = MAX_waterlevel - elem_value
4:      dist_yellow = Average_waterlevel - elem_value
5:      dist_green = MIN_waterlevel - elem_value
6:
7:      if dist_red < dist_yellow and dist_red < dist_green then
8:          elem belongs to red_group {red_group is alarm group}
9:      else if dist_yellow < dist_red and dist_yellow < dist_green then
10:         elem belongs to yellow_group {yellow_group is warning group}
11:     else if dist_green < dist_yellow and dist_green < dist_red then
12:         elem belongs to green_group {green_group is safe group}
13:     else
14:         elem cannot be distributed
15:     end if
16: end for
```

Computation Time: is proceeded mainly from calling for-loop on all water point (see: line 2), so it's limited by upper bound of polynomial-time.

Continue with the first algorithm is to develop the idea find and calculate the value of all points at every single time points (time step 15 minutes) from 0:00 to 24:00, totally: 96 time points. At every time point it's expected that the water points are lying in different water levels, these are result by measuring at different time slots (days, months, years) in history. So we will need a way to reuse all of these data and avoid losing information as much as possible and concurrently stay close with the reality. In this case, the current water level representing the reality and total water data points (measured with current data 17043 points) are the major parameters. All the data points will need to be reordered and prioritized with respect to this current water level. The water point which is closest to the current /water level will contain the most valuable information. Furthermore, the water points before and after the current time point and least recently to the current time points will be considered with the most valuable information (e.g. 15 minutes before and after the current time point will get the most information rather than 8 hours before and after). After that we can identify the already calculated data points and apply least square method on these dataset to derive the corresponding line which is closest to the data points. The calculation task will be repeated with the same method but on different degree so that 2 different results (one as linear result and one as parabolic result) will be compared each other. The error estimate method is applied on the current water level to find out the equation form which hast smallest error estimate will be chosen. Finally, we're able to calculate the predicting water level from the computed equation by inputting the predicting hours. The difference between the calculated water level and current water level will be reused to create the artificial predicting point. This is the idea to keep the reality and history data as close as possible. The following algorithm 2 describes again how to put all above ideas together to calculate the predicting water level.

Computation Time: is proceeded mainly from calling for-loop on all water point in all time points, so it's limited by upper bound of polynomial-time.

(e.g. $NumberOfDays181days \times NumberOfElem17043points \times NumberOfTimeSlots96timepoints$)

Algorithm 2 Prediction by applying method least square data fitting
For full details on implementation, see: function lspm and _task_lspm

for $data$ in $dataset$ **do**
    Run groupby-function to sort all data by time from 0:00 to 24:00,
end for{*return will be a list of 96 points. Every point contains a list of all measured water level points. see: function _findWaterlevel*}

Sort this list of list with respect to current water level by calculating distance between the water point and current water level. The smallest the distance, the higher rank is the the water point. Return will be a sorted list {*(see: function _findAllPoints)*}

{*the number of time step is 96, with step size 15 minutes from 0.00 to 24.00. See: function calMeanValue_prio)*}
for $index$ in 96 **do**
    sumsublist = find length (number of points) at every single time point and calculate sum from 1 to this length
    for $elem$ in $dataset$ **do**
        result = (lengthsublist - i)*value_thispoint/(sumsublist)
        {*calculate sum of coefficients water points at every time point by using this. The returning result will be a list of all sub results*}
        final_result.append(result)
        {*save result at every time point into a list*}
    end for
end for
{*result will be a list of all calculated water point at 96 time points*}

Identify the time range where the calculated water points are being found with respect to current time point. A sub list will be returned based on the predicting hour forward and backward. {*e.g. predicting hour = 8 hours and current time is 10:00, the calculated water points will be gathered from 2:00 to 18:00. The return result will be list of data points which is used for finding equation (see: function _processZone_prio)*}

for $elem$ in $derived\_dataset$ **do**
    Apply method of least square on the list to find the coefficients, respectively equation which is most closest to the data points, choose the best result with least error estimate
end for

Calculate the water level with the found equation based on the giving predicting hours
{*see: function _task_lspm)*}

Create artificial result to get the water level as close as possible to the reality (current water level)
{*see: function showArtificialResult)*}

Visualize all results
{*see: function visualize*}

Method 2

The idea is to sort out all given dataset in years 1993, 1995, 2013, 2016 in a meaning order, so that the rate of changes of water level by days will be recognized clearly. The first result is the total number of days and every day is considered as a stored object which contains the data about water level was measured in every 15 minutes, should be from 0:00 to 23:45. Data will be tracked by days and timing which is suitable to find out the relationship between water level at the current time and in history time. The next algorithm helps to find all days with water level is close to the current water level at the current time so that we can identify at what level the water rising might be (in worst case, best case) in case of water level rising or decreasing depending on the current weather condition outside. After all, we apply the same technique as in first method to find out mean value, coefficient and calculate the predicting water level and create the artificial water level in expected predict hours. All between and last results will be visualized out.

Algorithm 3 Find out all data sorted by days and time (see: function rateofchange)

1: **for** $data$ in $dataset$ **do**
2: groupby dataset sorted by date (expected data from 0:00 to 24:00)
3: **end for**
4: return list of rate of changes of water level sorted by days

Algorithm 4 Prediction by finding out the data which are closest with the current time and water level

1: **for** $data$ in $sortedlistfromabovealgorithm$ **do**
2: **if** the time at day == the current time and the rounded up water level of that day == the rounded up current water level **then**
3: save data of the whole day
4: **end if**
5: **end for**
6: return a list of all days when the water level go nearby the current water at the giving current time

Computation Time: is proceeded mainly from calling for-loop on 181 days in all time slots (96 points), similarly it's also limited by upper bound of polynomial-time.

4 Test and interpretation of results

In this part, we proceed to test and interpret the results output in figures as following. Given input by user to CLI (Command Line Interface) is:

- Input the current water level (in cm): 650
- Input the current time (from 0:00 to 23:00 o'clock): 10.0
- Input the predicting hours (max 12 hours): 8

The first line is the result in normalized format and the second line is the result already converted. The third and forth lines are the artificial result calculated based the error difference between history data and data at current time.

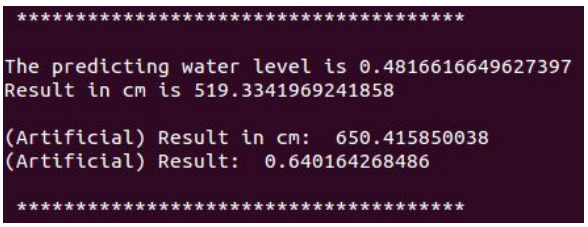

Figure 3: Result from method 1

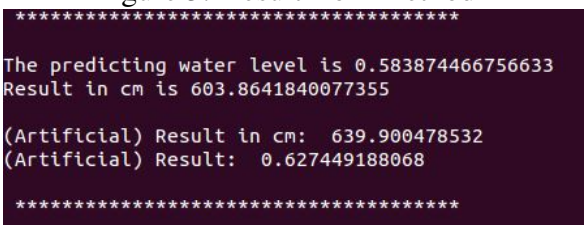

Figure 4: Result from method 2

Comparing the result between method 1 and method 2, we can easily recognized the difference since the result of the first method is calculated by the mean value of all data point at every single time point and the result of the second method is limited only by the days with water level close to the water level at current time. Hint: User can try both method to get al result and get the average of result or the range where the water level might be.

The next following figures show the visualized results exported out by both methods.

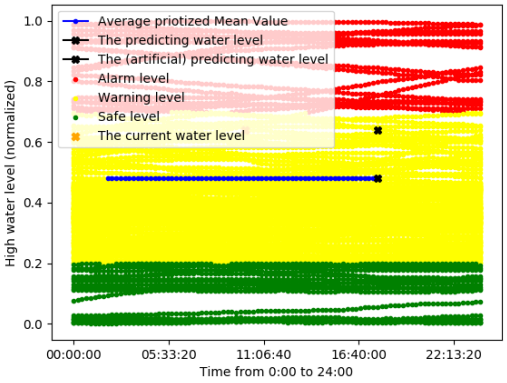

Figure 5: Result from method 1

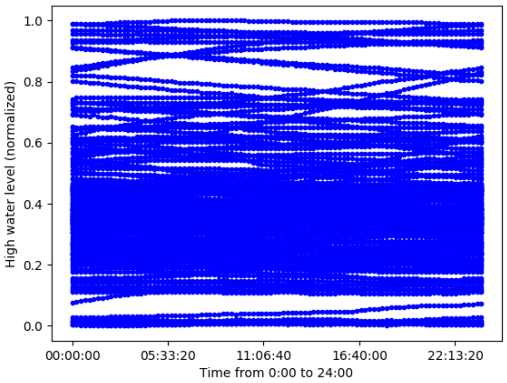

Figure 6: All water level sorted by days

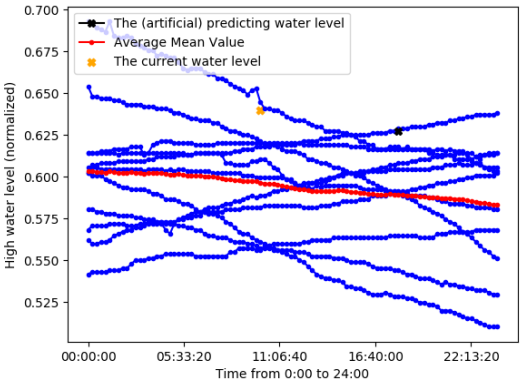

Figure 7: Result from method 2

The blue line in figure 5 is the result calculated by mean value at every point. Data which are used to calculated is starting from 2:00 AM to 18 AM (due to current time +/- predict time). The mark 'X' at the end of blue line show the predicting water level corresponding to the history data and the above 'X' show the artificial predicting water level corresponding to the current time.

The blue lines in figures 6 & 7 retrieve all history water data and find out the similarities close enough to the current reality and the red line demonstrates the mean value of all blue lines. In figure 7 we easily recognize the worst case and best case scenarios when water level keep rising and decreasing. Hence, for the future prediction we were able to scale the result in the calculated range. In order to keep the result closer with the reality situation, it recommends that user keep updating with the current weather condition outside. E.g. the weather at current time is bad with heavily rainfall and weather service informs this situation will last many days. Therefore, with this information, we only focus on all the blue lines above the red line with rising up tendency.

5 User Manual: Question & Answer

Question 1: How to know whether the current water level is in danger or still safe?

Answer: please see the explain about classification of dataset and Algorithm 1

Question 2: How to help user know clearly about state of water regarding with location of container?

Answer: Location of Container on the map gives height comparing to the sea level rising, e.g. about 67 meter. Water lever rising is measured at the PNP (PegelNullPunkt) is started at 57 meters, so the safe distance is x $<$ 67 - 57 = 10 meters. However, user should start concerning as the water level is already in early warning (min of yellow zone) and considering the time cost would take to remove the container as well in extreme weather condition.

Question 3: In approach 2, how to know that the water level will be rising up or down?

Answer: The question is to know what physical factor affect on this water level rising process. It's raining. User should keeps updating with information locally on the current weather condition outdoor and try to run program simultaneously. Program will help user to navigate where the water level will be landing. With the updated information, we can easily all calculated cases manually and focus on the result closest to the reality. E.g. if currently there is heavily raining outdoor, the tendency will be up in next hours. User should try both methods and compare the results to get the most possibly best result.

Question 4: How to install program and reproduce the result?

Answer: Program has been developed in Python3 and distributed as Open Source GPL v2 on Github. User can run program by inputting the parameters via Command Line Interface. Result will be output via logging-file and under visualized figures and concurrently all of input parameters and results will be stored into csv-file and synchronized to database. About the technical detail how to install and run, please find program at:

`https://github.com/lphan/utilkit/tree/master/floodpred`

6 Conclusion and further study case

Exact weather forecast and prediction of natural disaster respectively is extremely difficult, especially when delivering exact result in short time even with help of supercomputer (Source?). However, in some cases we're still able to observe and target some doable use case and find a fast solution to support decision-maker. For this purpose, the program *floodpred* has been developed to answer the question *at what water level should user start concerning?*. The answer is clearly divided into sub-answers by first analysing the monitoring on water level into 3 layers (green, yellow and red) and hence building different methods to compute and predict the water level at the expected future time in appropriate reality. All the calculated results is getting better and more accurate just by keeping the raw-data updated via Excel file and CLI of program without having to change any line of code. After computation, all the results will be stored into database to be served as cache layer to deliver a fast result in case of computing with the same input parameters. Moreover, the result has been computed in a reasonable polynomial computing time to support a fast decision either. Furthermore, program and this document are distributed into Open Source Community so that everyone can freely join and keep contributing additional features (e.g. investigate the water flow, study shallow water equations) to this software in the future.

First thank for the helping from the working group in project FloodEvac by sharing scientific documents. Following are the scientific references in which I've been studying:

References and Notes

[1] Infoflyer FloodEvac http://www.floodevac.org/uploads/5/2/9/9/52993787/infoflyer_floodevac_engl.pdf

[2] Decision Support Systems for Urban Evacuation Logistics in Practice, Marc Goerigk, Horst W. Hamacher, Sebastian Schmitt

[3] Sink location to find optimal shelters in evacuation planning, P. Heler1 H. W. Hamacher1

[4] Ouyed & Dobler - Chapter 4 - Interpolation/Extrapolation Techniques http://pjl.ucalgary.ca/courses/physics381/computational-physics/Ouyed-Chapter-4-Interpolation-Extrapolation-Techniques.pdf

[5] S. Gopal Krishna Patro, Kishore Kumar Sahu Normalization: A Preprocessing Stage https://arxiv.org/abs/1503.06462

[6] Flood level indicator and risk warning system for remote location monitoring using flood observatory system https://www.researchgate.net/publication/234830161_Flood_level_indicator_and_risk_warning_system_for_remote_location_monitoring_using_flood_observatory_system

[7] Daher, E.; Kubicki, S. and Guerriero, A. 2017. Data-driven development in the smart city: Generative design for refugee camps in Luxembourg, Entrepreneurship and Sustainability Issues, 364-379. http://dx.doi.org/10.9770/jesi.2017.4.3S(11)