# Predicting fluvial flood arrival times by making use of a deep learning model

Ron Bruijns

Water Management
Civil Engineering
TU Delft

# Predicting fluvial flood arrival times by making use of a deep learning model

by

Ron Bruijns

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on 24th of June, 2022.

| | | |
|---|---|---|
| Student number: | 5124174 | |
| Thesis committee: | Dr. R. Taormina, | TU Delft |
| | Dr. M. Hrachowitz, | TU Delft |
| | Dr. E. Ragno, | TU Delft |
| | PhD candidate R. Bentivoglio, | TU Delft |
| | Dr. J.D. Nuttall, | Deltares |
| Other supervisors: | Dr. X. Li, | Deltares |
| | ir. R. Dahm, | Deltares |

TU Delft

# Abstract

Fluvial flooding poses a major threat to mankind and annually leads to major economic losses with many casualties worldwide. The consequences of this can be mitigated when accurate and rapid predictions are available when the water will arrive at which location. Current numerical simulations take a significant amount of time due to their computational demand, which is not affordable during a calamity where each second can make the difference between life and death. Literature has shown promising results in fluvial flood forecasting by using a deep learning model, which generally takes only a fraction of the computation time compared to conventional models. However, these models have thus far only been trained on static landscapes with changing boundary conditions, which prevents the model's application elsewhere.

This research explores therefore the possibilities of creating and training a generic non-location bound deep learning model which can predict the spatial distribution of fluvial flood arrival times per grid cell. The architecture of the created deep learning model consists of five parallel encoder-decoders, which takes the elevation, slopes, elevated elements, land roughness, and initial water levels into consideration, depending on the dataset. The model is trained, validated and tested on four unique datasets, which consists of 30,000 flooding samples. The degree of complexity within a sample increases with each dataset number.

The average error for the four consecutive test datasets were 0.91, 1.41, 1.25, and 1.76 hours per cell. The differences in the predicted and the groundtruth are relatively small although the deviation tends to become larger at the end of the simulation, in regions with a strong gradient in arrival time, and in hilly and complex landscapes.

In addition, the model has been tested on various benchmark landscapes to examine specific flow phenomena, as well as a realistic test scenario in the Dutch dike ring 48. The model shows satisfactory performances for landscapes other than those present in the dataset, untill it encounters a feature on which the model was not trained for, such as undershots or irregularly shaped waterways.

This research has shown the potential of deep learning in predicting fluvial flood arrival times on unseen before landscapes. Recommendations for further studies include the use of an active dike breach and a variable flood location.

# Preface

As this report may be the last one of my academic career, it is a good moment to shortly reflect on it. It is truly astonishing how the last six years have been. When my future self would have told me as high school graduate what would all be ahead of me, I would not have believed him. From all the new people I met, to all the places I could visit and live, to all the experiences I could enjoy. Not only these external factors were truly remarkable, but also the internal change as a person is fascinating. I will carry all these memories for a lifetime, and want to thank everyone who contributed to this.

I also want to thank my supervisors for their help throughout the process of my thesis. I would like to thank Riccardo for his contribution to introduce me to the amazing field of full potentials of artificial intelligence, while also keeping the meetings informal and relaxed. Secondly, I would like to thank Roberto, who always had time to help me with one of my many small questions when I would walk by. Thirdly, thanks Elisa and Markus for your fresh view on the topic during the intermediate meetings during the progress.

Lastly, I would like to express my gratitude to my supervisors from Deltares. Jonathan's help was essential in the process of understanding and developing the used deep learning model, while Xiaohan's support was indispensable to work with the numerical flood software. And last but definitely not least Ruben, who has a thorough knowledge of basically anything I have asked, while also making sure that the overarching goal remained clear and logical along the way.

Once again, thanks to all.

*Ron Bruijns*
*Delft, June 2022*

# Contents

# List of Figures

# List of Tables

# Acronyms

| Abbreviation | Explanation |
|---|---|
| AE | Autoencoder |
| BL | Benchmark Landscape |
| CNN | Convolutional Neural Network |
| DEM | Digital Elevation Map |
| Delft3D FM | Delft3D Flexible Mesh Suite |
| DL | Deep Learning |
| ED | Encoder-decoder |
| EE | Elevated Element |
| FAT | Flood Arrival Time |
| FNN | FeedForward neural Network |
| Gt | Groundtruth |
| IWL | Initial Water Level |
| LC | Land Cover |
| ML | Machine Learning |
| P | Predicted |
| WW | Waterway |

# 1 Introduction

## 1.1. Flooding

Flooding poses a natural thread to mankind. Globally there were more than 360,000 fatalities worldwide due to inundation (Alves 2022) in the last six decades, and the annual global economic loss of flooding between 2016 - 2035 is estimated to be US$597 billion. The economic and human loss of flooding depends on several variables, such as flow velocity, inundation, level of contamination, and water depth, of which the latter is the most important indicator (Merz et al. 2013) (Schröter et al. 2014). These variables are determined by the type of flooding. Pluvial floods tend to have high rainfall intensities but short durations, whereas coastal floods tend to have large water depths and flow velocities, which can be enhanced by (spring) tides and waves. The contamination of salt water is an additional source of costs that can reduce crop yields in the years to come. Fluvial flooding caused by dike breaches tend to have long inundation periods with significant water depths (Paprotny et al. 2021).

There are thus severe economic and social losses attached to the consequences of flooding, which are enhanced by the ongoing climate change which intensifies a majority of the regional hydrological cycles due to the increased amount of air vapor that can be held in the atmosphere (Collins et al. 2013). This leads to an increased risk of flooding in the future on a global scale (Willner et al. 2018), which enlarges the need for adaptation and mitigation measures to minimize further losses (Hirabayashi et al. 2013) (Jongman et al. 2012).

Losses to static structures and buildings are hard to prevent during a flooding event as they can not be mobilized. People however can be evacuated, given accurate flooding information and sufficient time. The information should thus not only be reliable but also has to be obtained quickly. While radar information can indicate pluvial flooding a few days in advance, fluvial floods are less predictable due to the uncertainty and lack of information on the local scale of dike stability for an entire river section.

To understand flooding caused by the failure of these levees, it is key to understand the dike breach development. It is shown that a dike breach is usually not a sudden collapse of the entire structure, but rather a progressive process where the interaction between water flow and sediment transport plays a crucial role (Guan et al. 2014). The total breach size and thus the extent of flooding behind the dike depends not only on these angles, but also on the erosion rate which is determined by the material characteristics of the dike and the flow velocity of the water (Spinewine et al. 2004). Water levels, dike failure, and the subsequent flood can be modelled by making use of a variety of models and software, each having its advantages and limitations.

## 1.2. Flood forecasting

Different types of methods exist to predict flood propagation, of which the oldest method is by observation. These observations can range from high-tech equipment to century-old eyewitnesses and landscape formations. This usually involves single, specific flooding examples, which does not permit accurate extrapolation when boundary conditions change for a new scenario.

Next, there are simplified or non-physics-based models. These models discard the physical process of flooding, e.g. dividing the landscape into different buckets based on the elevation. These buckets are filled completely after which they start to fill the next bucket. This process is repeated till the total amount of water in each bucket equals the total defined discharge into the landscape. Although the computational process can be up to orders of magnitude faster than conventional models, they give a rather restricted view of the temporal and spatial evolution of the flood and are therefore impractical during a calamity (Lhomme et al. 2008). Current hydrodynamic models are typically based on solving different physical-based equations which are used to make well-founded predictions about how the flooding will unfold. Different types of models can be used depending on the research goal and data availability. A 1D model is suitable to calculate water depths and velocities in one dimension by solving the Saint-Venant equation. The depth-averaged Navier-Stokes should be solved to model the spatial evolution of the flood on a 2D plane. This is the most commonly used method for predicting flooding (Teng et al. 2017). Even more sophisticated models exist which calculate flow in three dimensions, where the three-dimensional Navier-Stokes equation is solved for modelling phenomena such as vertical turbulence or vortices. As this requires an extensive amount of computation time, a 2D model is usually preferred (Teng et al. 2017).

## 1.3. Numerical software

Delft3D Flexible Mesh Suite (Delft3D FM) is the used numerical software in this work, which offers a wide variety of 1D, 2D, and 3D hydrodynamic applications for coastal, river, and estuarine areas (*Deltares* 2022). It is, among many other features, able to calculate the breach growth development and the flow of water which flows through the gap. However, a simple point source flooding with a constant, set discharge is possible as well. Next, the water from this flood flows over a predefined grid over a certain terrain. The software is able to include additional features such as a heterogeneous land roughness map and a layer which describes initial water levels (IWLs). These layers can be embedded by manually adding them within the graphical user interface of Delft3D FM. A great advantage of Delft3D FM is that adding these layers is also possible by making use of an external programming software, where in this work Python 3.8 is used. After activating a batch file, the simulation is run and the desired output variable and its output frequency are delivered. This can be retrieved by the programming software as well. This means that numerous flood simulations can continuously be made without manually starting single flooding simulations, as will be elaborated upon in the chapter of the methodology.

The computational cost of solving the shallow water equations is however high, especially for a large grid with a fine resolution. Nonetheless, first responders need quick and accurate information to secure entire communities during an emergency. Pre-ran flooding simulations are not likely to suit situation-specific boundary conditions which make the outcome non-representative. Similarly, the computational cost of running a new simulation during a flooding event is too high. There is therefore a need to accelerate the time it takes to model floods. This could be done in several ways, such as simplifying the model by disregarding certain flow phenomena, or by choosing a coarser spatial and/or temporal resolution. Important details can however be lost by doing so, with severe consequences as result. Another suggestion could be to use a more sophisticated machine with superior hardware to perform the calculations, although these computers may not be available in the short time span of an emergency. An optimal solution would be to have an accurate model, requiring only a fraction of the computational time to predict the initial phase of the flood, regardless of the computational resources available

As elaborated in the previous section, both physical as well as conceptual models can perform well. It is thus not necessary to know each physical phenomenon within a landscape in detail to create proper predictions, as long as the model is able to establish relationships between input and output and underlying patterns. Data-driven models can discover these underlying relationships from the available data, without explicit knowledge of the physical laws governing the system (Solomatine et al. 2009).

## 1.4. Deep learning

Artificial Intelligence (AI) falls under the category of data-driven models, as it is able to detect patterns and relationships in its training data and apply them on unseen data for predictive purposes. A main disadvantage of AI is its black-box nature however, where it is hard to retrieve what the exact relations between input and output data are and whether the supplied data is representative enough (Goodfellow et al. 2016). On the contrary, it is able to find non-linear relations that may not be covered by current, possibly limited, knowledge of certain processes. According Nearing & Gupta (2015), when a process-based model is outperformed by a data-driven model, the full information content of the input and output data is not fully understood and thus not taken advantage of. It is thus possible that these AI models can outperform conventional models, especially when data on landscape characteristics are sparse (Kratzert et al. 2019). In addition, AI models are generally speaking able to compute output significantly faster than physically-based or even lumped models. In short, AI models may have the potential to accurately predict flooding in a short period of time. The next section explains therefore some commonly used AI algorithms.

### 1.4.1. Artificial Neural Networks and Feedforward Neural Networks

Deep learning (DL) models consist of multiple (hidden) processing layers, which can extract features on different levels of abstraction within a dataset (Najafabadi et al. 2015). Once the model is *trained* (calibrated), it is able to create predictions on unseen data. A great advantage of DL compared to other machine learning (ML) techniques is that there is no need to supply a DL with information regarding which features are relevant, as a DL model is able to extract that itself. It is doing so by making use of (non-)linear functions in the hidden layers which are sequentially connected with each other (Bengio 2009). The structure is mimicked from the structure of the human brain. Here, neurons transmit information through axons to other multiple neurons. This biological structure is translated in artificial neural networks (ANN) in computers. This structure starts with an input layer, which is attached to a hidden layer through a connection with weights and biases. Depending on the type of model, there can be multiple hidden layers of different sizes, which are either partly or fully in connection with the next hidden layer, where again each connection has its own weights and biases attached to it. The last hidden layer is then connected to an output layer, which is the final result of the model. This forms the basis for one of the simplest forms of ANNs, namely a Feedforward Neural Network (FNN). A simple example can be seen in Figure 1.1. Although an FNN is not used in this research, it explains the main DL related mechanisms in a simple form. These mechanisms do apply to more complicated algorithms, which are explained later in this section.

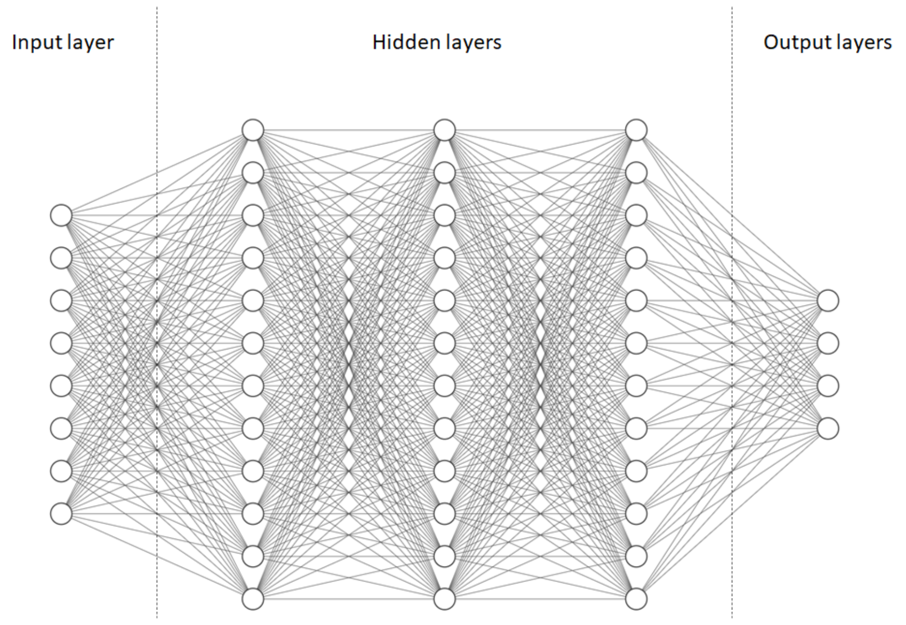


Figure 1.1: Schematized figure of an artificial neural network.

Random values are initially assigned to each weight and bias when the first data sample is presented. This results in a poor outcome, as the model used these random values to compute the output (Da Silva et al. 2017). The offset from the actual outcome can be quantified with a certain loss function, which keeps track of the difference between the predicted and the groundtruth. This can be the absolute error, mean squared error, or any other formula which keeps track of the difference between the modelled and the actual output. The choice of loss function depends on the aim of the model; a good prediction of extremes is more likely to involve a power function, whereas predictions of median values may require a lower power function (Rudy & Sapsis 2021).

The algorithm makes consequently use of backpropagation, which looks to what extent each neuron in the model is responsible for the total loss. The model updates these weights and biases accordingly, to gain a better outcome in the next run which decreases the total loss (Li et al. 2012). This process is repeated until the accuracy of the model is not increasing anymore. The performance of the trained dataset is scored based on a validation dataset, which consists of data samples that do not occur in the training dataset. Once the model is trained on the best validation score, the model can be tested against a test dataset. These samples do again not occur in the training nor validation dataset.

When too little data is supplied to the model, it may not have learned all correlations between all features. Instead, the model may have focussed on only a few training data specific samples, resulting in overfitting. This leads to a model which performs well for only the training data but deteriorates when applied to new data (Bilbao & Bilbao 2017). It is therefore key to have sufficient and diverse training data, as this data calibrates the model. Although this applies to any DL model, a FNN faces additional limitations. For example a relative low performance when dealing with a dataset which includes many features with a resulting higher dimensionality (Glorot & Bengio 2010) (Klambauer et al. 2017). This is referred to as the 'curse of dimensionality', which states as a rule of thumb an exponential dependence between the number of features and the number of training samples to obtain accurate predictions (Poggio et al. 2017). Furthermore, FNNs suffer from an absence of memory as there is no connection between time steps when making use of a time-series. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) models do take this into account. This research does not focus on these kinds of algorithms, although they are shortly discussed in the chapters of the results and the recommendation. Hence, background information on these algorithms can be found in Appendix A.

### 1.4.2. Convolutional Neural Networks

Convolutional Neural Networks (CNN) are characterized by local connections and sharing weights, and their main advantage is to effectively extract spatial information (Wang et al. 2020). The convolutional layers are made of filters (kernels) of a predefined size extracting features from the original image. CNNs also feature pooling layers which down-sample the feature maps to facilitate the propagation of meaningful information and reduce computational costs (Chen et al. 2016). Convolutional and pooling layers are stacked together to form the basic processing blocks of CNNs, see Figure 1.2. Although spatial information is 'lost' when going deeper into the network, information is gained in terms of what features were found in the image as the original image is presented in a higher form of abstraction (Soleymani et al. 2018).

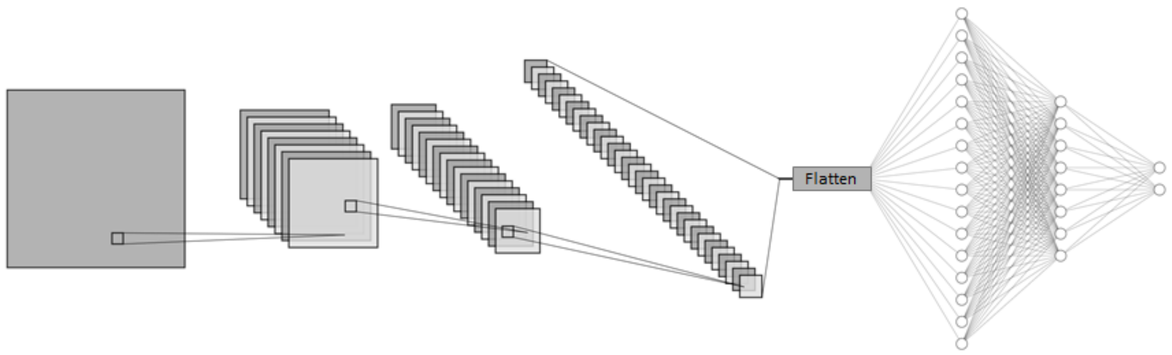


Figure 1.2: Schematic overview of a CNN, where the original 2D image is converted to a 3D dataframe by making use of multiple convolutions. The data is then flattened to produce an output in the form of two nodes.

Since the filter's dimensions are larger than one, they share their weights and biases across multiple pixels, leading to feature detection across the entire original image. CNNs are therefore more efficient compared to FNNs when dealing with spatial data (LeCun et al. 1998). Lastly, CNNs have the advantage of translational equivariance (Weiler et al. 2018), which implies that objects can be recognized everywhere in an image, whereas FNNs have to learn the location of where these objects can be located. Figure 1.2 shows the architecture of a typical CNN used for image classification. The features extracted by the sequence of convolutional blocks are ultimately processed by a FNN to perform the prediction. The generated output is thus in the form of the likelihood of occurrence of certain classes, which is not yet a 2d image which is however necessary to make flood predictions.

### 1.4.3. Encoder-decoder

A similar approach as the CNN can be taken to not only downsample an image, but also to upsample it. This results in a convolutional autoencoder. The essence of an autoencoder is to have the input layer the same as the output layer, while having somewhere within the model a compressed version of itself (Bank et al. 2020). It consists of an encoder, a latent space, and a decoder part. A slight variation on the autoencoder is an encoder-decoder (ED), where the original data is translated to a new image by applying filters to the input layer to obtain a new output layer. This results in a different representation of the original data, which enables the user to generate a certain altered output from the input layers (Cho et al. 2014). A schematic example is shown in Figure 1.3. Multiple relevant spatially distributed maps can be used as input layers, leading to one final output map of the feature of interest. An elaboration of this architecture is given in the chapter of the methodology.

### 1.4.4. Hyperparameters

In addition to the general architectures described above, has each deep learning model also hyper parameters, which are parameters that guide the learning process during training (Feurer & Hutter 2019). These parameters control e.g. the size of the kernel, the number of strides corresponding to the number of pixels are skipped when the kernel moves, or the type of padding which describes the number of pixels added to the boundary of an image when any part of the filter exceeds this border (Liu et al. 2018). Another key hyperparameter is the activation function, which describes the method to transform the input signal of a neuron into an output signal (Sharma et al. 2017).

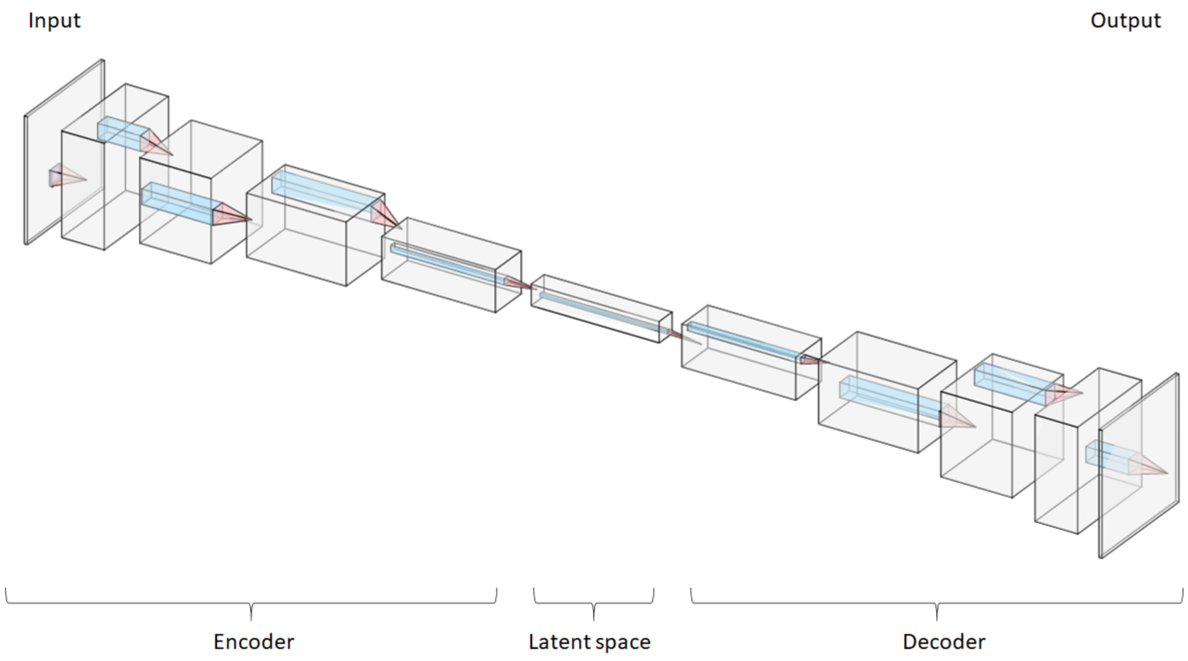


Figure 1.3: Schematized figure of a convolutional autoencoder, where the original data is transformed and squeezed back to an output of the same shape

Next, the optimization function describes the method to find the minimum loss function. Furthermore, the dropout rate is the ratio of neurons removed along with the inputs and outputs. Depending on the model, this can reduce overfitting and reduce the size of the network (Srivastava et al. 2014). In addition, the learning rate is one of the most important hyperparameters which describes the step size as it moves to a minimum of the loss function. A too large value results in divergence, while a too small value results in slow convergence (Jacobs 1988). A flexible learning rate is therefore recommended, as it is beneficial for both computational time as well as performance to start with a relatively high learning rate which decreases over the number of epochs. The total number of epochs that the model requires for training can be limited by the patience parameter, which limits training time and reduces the chance of overfitting when the model's accuracy stops increasing during the training process. The batch size describes the number of training samples which is given to the DL model, after which it will update its weights and biases.

## 1.5. Artificial intelligence and flood forecasting

The advantages of AI in the field of flood forecasting are increasingly becoming more apparent in the literature. Chang et al. (2014) and Jhong et al. (2018) have shown to model pluvial flooding in Taiwan during a typhoon accurately by making use of a hybrid DL model and a physical model. Choubin et al. (2019) used a combination of ML algorithms to retrieve fluvial flood susceptibility maps. Kabir et al. (2020) also employed ML to model fluvial floods. Their results showed that CNNs outperform support vector regression both in terms of computational time and accuracy. Guo et al. (2021) have shown that urban flood predictions can be accelerated by up to 200 times when making use of a DL algorithm.

The variety of applications of AI in the field of flood forecasting is thus vast and full of potential, although the research is still in its infancy. Some literature has trained a DL model on a variety of catchments so it can be applied to other regions, although this usually only entails streamflow forecasting. An example is Kratzert et al. (2018), who categorized a range of catchments into groups that share some similarities. These groups can be pre-trained, resulting in initial weights and biases which are already closer to the specific catchments. This improves accuracy and lowers the number of epochs which are required to train the model on specific catchments. This makes general rainfall-runoff patterns transferable to single basins (Kratzert et al. 2018). In cases where a spatial flood is modelled, the research is usually focussed on a fixed landscape with varying boundary conditions. The generalizability of these models to other areas is therefore non-existent. An exception is Löwe et al. (2021), which explored the possibilities of combing hyetographs as well as topographical data to create maximum water depth predictions in urban pluvial flood events, which are not bounded by a fixed domain. Similarly, Guo et al. (2022) studied the performance of a CNN to predict water depths and flow velocities on varying (sub)catchments while maintaining the design rainfall event constant. Both studies showed promising results with a significant reduction in computational time. However, these studies focussed on pluvial flooding, while fluvial flooding in unseen catchments has not been studied so far. However, there is a need to have such a model so that action can be taken quickly and adequately in the event of a calamity, where there is limited time to wait for long numerical simulations. Following objective and research questions are formulated to tackle this knowledge gap.

## 1.6. Objective & Research questions

The computational demand of flood modelling takes a significant amount of time. This may drastically be reduced when making use of a deep learning model. The objective of this research is to determine whether such a generic deep learning model can accurately predict the arrival time of the water per cell after flooding on unseen terrains, with a significant reduction in computational time compared to current numerical methods.

### 1.6.1. Research questions

The main research question is as follows: ”*How well can a deep learning model predict fluvial flood arrival times after a point source flooding?*” To answer this main research question, following sub research questions have been formulated:

1. What is the overall accuracy of a deep learning model which can predict fluvial flood arrival times for different synthetic datasets?
2. How do spatial and temporal characteristics influence the error distribution?
3. How is the model performing when applying it to theoretical flow situations?
4. How is the model performing when applying it to a realistic study case?

# 2 Methodology

The performance of a DL model strongly depends on the variety, complexity and size of available training data. Obtaining this data from past flood events is not only cumbersome, it would also be insufficient as tens of thousands of unique fluvial flood samples are required to train the model. It was therefore decided to generate synthetic landscapes which were consequently flooded by making use of flood modelling software Delft3D FM.

The aim of this research is to develop a generic model that can predict fluvial flood arrival times (FATs) for arbitrary topographies. This is a crucial difference compared to existing literature, which mainly used a fixed landscape with varying boundary conditions. This usually results in datasets consisting of a single catchment, where variables such as runoff or rainfall intensity differ between the training samples. Such trained model can therefore only be applied to that specific area as it is trained on fixed landscapes, limiting interpolation and extrapolation tasks to within the same domain. The training process of a generic DL model requires an extensive dataset with numerous varying landscapes. These synthetic landscapes are therefore automatically generated in Python, as retrieving actual data may be cumbersome. The parameters of the landscape settings can easily be updated, which is convenient if a new dataset is required to train a new model. This may be required if future test landscapes do not fall in current landscape ranges, or when features are not yet included in current training data.

Next, fluvial flooding on these landscapes is simulated by making use of Delft3D FM. The output, which is a temporal evolution of the water depth, was then converted to FAT maps, which were then used to train, validate and test a DL model. Each of these steps is elaborated upon in more detail in the following sections.

As no actual flooding observations will be used as training data, it should be noted that the primary objective of this study is to investigate whether a DL model can surrogate a numerical model, which is the reason why the training data of the DL model will consist solely out of synthetic numerical output data as will be elaborated upon in the chapter of the methodology. In other words, the aim of this research is not to compare a model to real life flooding examples, but to compare the performance of a DL model against a numerical model. Besides the novelty in terms of set up of this research, there is an actual application perspective as well. As this research may be the starting point for future studies and possibly later for real life applications, it is interesting to study under what circumstances the DL model can surrogate a numeric model, and during which conditions the DL model should not be used. The aim is therefore to push the DL model to its limits. A substantial part of this research focusses consequently on finding these application boundaries.

## 2.1. Temporal and spatial dataset characteristics

Firstly, the temporal scales are discussed. According to an interview with the Dutch waterboard Rijn & IJssel, the first 48 hours of the course of a fluvial flood caused by a dike breach can be highly uncertain as numerical simulations may require similar times to finalize their computations. Accurate predictions by such numerical model are thus only available after this duration, so a surrogate model is required for the interim period. It was therefore decided to simulate flooding for up to 48 hours. The output time step is one hour, which was based on a balance between capturing a sufficient degree of hydrodynamic phenomena and practical use, such as storage.

Secondly, the spatial scales are discussed, which consist of the cell size and the number of cells within the grid. The former depends on the desired degree of detail within an image and the total area that will be captured within the grid, whereas the latter determines computation time. A rectangular grid of 64 by 64 cells was therefore chosen. As these values are part of the sequence of the power two they are also convenient from a DL point of view.

Computational experiments were subsequently carried out to find a suitable cell size. Simulations started initially with a grid resolution of 25x25m, which is in accordance with the Dutch guidance document for modelling floods (De Bruijn et al. 2018). However, this spatial scale in combination with a temporal scale of one hour posed problems in terms of data outcome. The relative fine spatial resolution, and thus small area, caused the area to behave like a filling container as the entire domain becomes flooded rapidly. Flood propagation was not visible at this scale, requiring either a finer temporal resolution with shorter simulation times, more grid cells, coarser spatial resolution, or a smaller discharge through the breach. A finer temporal resolution and shorter simulation times were not desired from an application point of view. More grid cells would lead to a higher computational numerical cost to generate the training data, which was not preferred. Limiting the fixed discharge through the dike breach to a few dozen cubic meters per second was also not desired as it would lead to unrealistic flow situations, although it is dependent on the cell size. Computation experiments suggested that a proper balance between cell size and discharge would be 200x200m and 250 m$^3$/s, respectively. The sixteen-fold increase in surface area led to a realistic and visible flood propagation, and hence current settings were used for the numerical model. It should be noted however that the above dimensions are all linked to each other. Other configurations of the setup of the simulations that are slightly different could also work, considering the above trade-offs.

## 2.2. Dataset composition

Human intervention transforms a landscape from a natural to an artificial landscape, which is highly the case for a country like the Netherlands. As a result, a natural elevation map without any anthropogenic adjustments is not representative of contemporary landscapes. A model calibrated on such non-representative landscapes has little chance of good performances when applied to a real case study. The complexity within a landscape increases therefore to counter the above challenge. The difference in performance can be analyzed between each dataset, leading to conclusions regarding the applicability of the model in certain areas. Each dataset type includes the added complexity components of previous datasets numbers, although the specific dataset samples are not reused in each dataset. The first and simplest dataset includes only a natural Digital Elevation Model (DEM), after which complexities are added such as elevated elements (EEs), heterogeneous land cover (LC), and waterways (WWs). The exact procedure of each of these additions is explained in more detail in the following sections, and some visual examples can be found in Appendix B.

### 2.2.1. Dataset 1: natural landscape generation

The sloping landscapes were created in Python by using a noise function. A noise generator is required for terrain generation as a completely random DEM would result in static noise which is not realistic and would result in poor performance in real-life applications. Perlin noise is a commonly used terrain generator in video games, which can create natural-looking landscapes with just a few input arguments. The first parameter is the number of octaves, which corresponds to the number of waves present in the landscape. The second term is the seed value to initialize the field, for which a unique value was chosen to enhance variability in the dataset. These two variables are the main contributors to the overall layout of the landscape. The entire landscape was subsequently multiplied with a certain multiplication factor, to enhance or decrease the degree of relief. As a result, along with sloping there are also flat landscapes, which enhances the variability and thus the quality of the training dataset. Finally, to further increase variability, the whole landscape was elevated by a certain elevation. The exact values and ranges for each parameter are shown in Table 2.1.

Table 2.1: Overview of all relevant input parameters to generate each landscape

| Category | Characteristic | Value | Unit |
|---|---|---|---|
| Spatial and temporal resolution | Grid resolution | 200x200 | m |
| | Domain size | 64x64 | cells |
| | Simulation time | 48 | h |
| DEM generator | Number of octaves | 0 - 10 | - |
| | Seed number | Simulation number | - |
| | Overall multiplication factor | 0-8 | - |
| | Overall raise | 0-5 | m |
| Elongated structures | Number of EEs | 1-8 | - |
| | Minimum height EE | 2 | m |
| | Maximum height EE | 8 | m |
| | Number of WWs | 2-5 | - |
| | Min WW length | 20 | cells |
| | Freeboard WW | >0.5 | m |
| | Depth WW | 2.0 | m |
| Land roughness | Min roughness | 0.015 | $s/m^{1/3}$ |
| | Max roughness | 0.3 | $s/m^{1/3}$ |
| | Number of squares | 0-200 | - |
| Scaling ratios | Slope x direction | [-1, 1] | - |
| | Slope y direction | [-1, 1] | - |
| | DEM | [0, 1] | - |
| | EE | [0, 1] | - |
| | Roughness | [0, 1] | - |
| | IWL | [0, 1] | - |
| Dataset sizes | Number of samples per dataset | 30,000 | - |
| | Number of training samples | 24,000 | - |
| | Number of validation samples | 3,000 | - |
| | Number of testing samples | 3,000 | - |
| Flood characteristic | Discharge | 250 | $m^3/s$ |
| | Location | [0, 0] | - |
| Output definition (FAT) | Minimum depth | 0.0001 | m |

### 2.2.2. Dataset 2: elevated elements

The term EE is the general term for elongated but narrow structures that can retain water for a certain amount of time. These structures resemble structures such as inland dikes, railways, roads, or noise barriers. EEs can overflow, depending on their height and adjacent water depth. The EEs in the Delft3D FM simulations were therefore chosen to have a varying height, to represent both overflow-able dikes as well complete water retaining structures.

The start and end point of each EE are randomly chosen on the map. A line is drawn between the points, after which the intersected cells are raised. Several EEs can intersect. The DEM value is in that case overwritten with the last defined EE height. Relevant parameters such as the number of EEs per simulation and the heights can be seen in Table 2.1.

### 2.2.3. Dataset 3: land roughness

A homogenous LC and thus land roughness was used in previous datasets. The default setting of Delft3D FM was used for this, which corresponds to a Manning value of 0.023 $s/m^{1/3}$. LC maps usually have a less natural character compared to a DEM. Perlin noise is therefore not suitable for creating synthetic land roughness maps. It was therefore decided to approximate roughness maps by stacking many rectangles with land roughness values on top of each other. Each rectangle has an arbitrary width and height, with a particular land roughness value chosen from a bucket of land roughness values. The range and number of those bins are listed in Table 2.1, and are virtually unique for each sample.

### 2.2.4. Dataset 4: waterways

Many real-life landscapes include WWs of some sort, which act as both a storage buffer as well as a flow accelerator once filled. These channels are in the synthetic dataset created similarly as the generation of the EEs, where a random start and end point is determined. However, an important difference between the EEs and WWs is the orientation: as rectangular cells are used, water can only move to its directly adjacent cells, which are the four orthogonally neighboring cells. For water to propagate diagonally, WWs must be at least two cells wide. This would result in WWs of 400 meters wide, which is unusual in the Netherlands. As a result, it was decided to keep the width limited to one cell size, creating only WWs in the two orthogonal directions. This is not the case for EEs from a computational perspective, as EEs are able to block water when they are placed diagonally with an average width of at least one cell.

The WWs have a uniform bed level depth, which corresponds to 2.5 meters lower than the lowest surrounding DEM value of the WW. A uniform depth was chosen to eliminate additional variables, such as different flood propagation velocities at a changing water depth. Subsequently, an IWL input layer was created, where the water depth within the WWs is two meters. When two or more WWs intersect, the lowest IWL and bed level are chosen for both WWs.

The range of the number of WWs as well as the length was inspired by the Dutch water system: the average summed WW length per simulation corresponds to the average WW length per unit area of the Netherlands. A minimum WW length of 20 cells was chosen to keep the ratio between the length and width proportional. The exact range of the characteristics of the WWs can be found in Table 2.1.

### 2.2.5. Benchmark landscapes

The four trained DL models are validated and tested against unseen data and are scored based on their performance. Additional synthetic test landscapes have been created to examine specific flow phenomena, which are referred to as benchmark landscapes (BLs). The model has only been tested on these flow situations, not trained nor validated. Each BL is briefly explained and visualized in Table 2.2.

Table 2.2: Representation of the 15 BLs

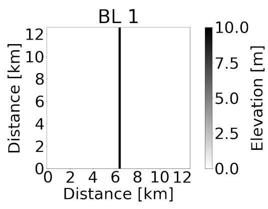
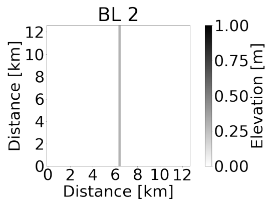
| Nr. | Representation | Explanation |
|---|---|---|
| 1 |  | Represents a non-overflowable EE, where only the left part should get flooded while the right part should remain completely dry. This BL represents a long elevated road or dike which stretches across the entire domain, leaving one side of the landscape dry. |
| 2 |  | Represents an overflowable EE. The height of the EE was such chosen that the lower right panel starts to flood the moment the left side is covered with water. This BL represents e.g. a noise barrier which overflows after a certain duration. |

Table 2.2: Representation of the 15 BLs

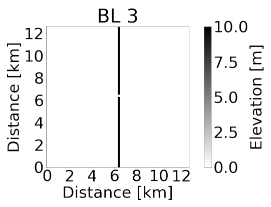
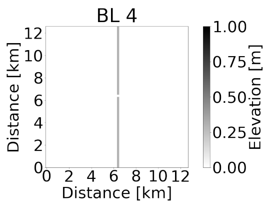
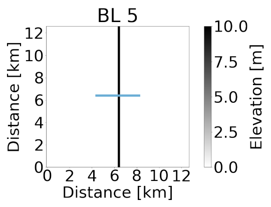
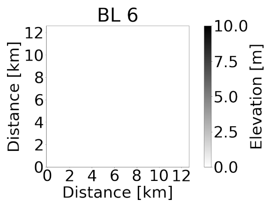
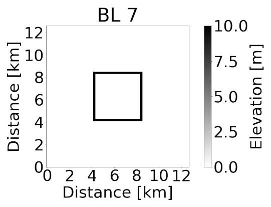
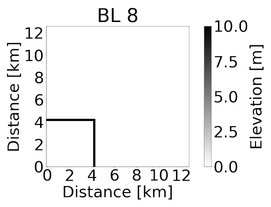
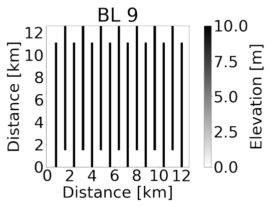
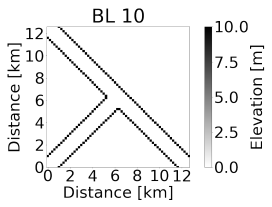
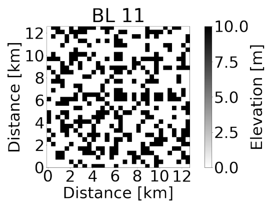
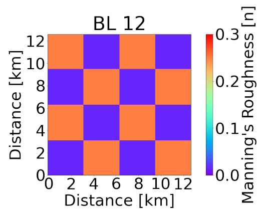
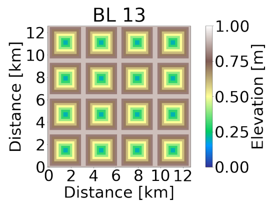
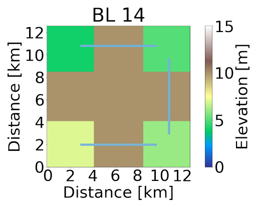
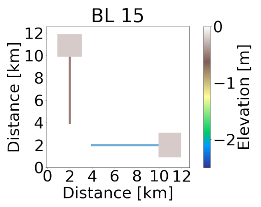
| Nr. | Representation | Explanation |
|---|---|---|
| 3 |  | Represents a non-overflowable EE, with a small gap where water can enter the right panel. This BL represents a structure such as an elevated highway with a flyover. |
| 4 |  | Represents a combination between BL2 and BL3: an overflowable EE with a gap in between. This BL is comparable to a mildly elevated road or railway, with a tunnel underneath which acts as an undershot. This undershot has too little capacity to convey all the water, resulting in overtopping of the EE after a certain duration. |
| 5 |  | Represents a similar landscape as BL3, except the opening has been replaced by a WW. The main reason for including this BL is that EEs in the training data set can only be crossed by WWs, resulting in a similar situation as presented in this BL. A change in performance between BL3 and BL5 indicates whether more training data with punctured EEs is required for improved predictions. |
| 6 |  | Represents the least complex, homogeneous landscape, without elevation differences. This landscape evaluates the performance of predicting the correct propagational speed of the water in the absence of any feature. |
| 7 |  | Represents a non-overflowable dike ring-like structure within a landscape which is not intended to be flooded. |
| 8 |  | Similar as BL7, but the location of the flood is now enclosed by a non-overflowable structure. This BL resembles a flood of e.g. a summer dike that remains within the boundaries of a winter dike, or a breach which is encapsulated at a greater distance. |
| 9 |  | Represents sharp turns in flood propagation when the end of an EE is reached, which can resemble e.g. the end of a noise barrier or the end of an elevated road. |

Table 2.2: Representation of the 15 BLs

| Nr. | Representation | Explanation |
|---|---|---|
| 10 |  | Represents splitting of the flow when an EE or other object is encountered. |
| 11 |  | Represents a highly unrealistic and maze-like landscape, with essentially many short EEs which can not overflow. The main goal of this BL is to examine the performance of the DL model in a highly complex landscape. |
| 12 |  | A flat area with highly varying roughness values in the shape of a chessboard, which is used to evaluate the performance of the DL model in terms of capturing the effect of varying land roughnesses on flood propagation. |
| 13 |  | Represents a pit-like landscape. The essence of this landscape is to evaluate the performance of the DL model's understanding of filling up each of these pits, after which it starts to overflow into the next pit. |
| 14 |  | Represents four consecutively lowered reservoirs connected by WWs, representing a water system which floods one reservoir after another. The main aim is to examine the DL model's comprehension of water conveyance through a narrow strip in the domain, which can cause flooding elsewhere. |
| 15 |  | Represents two dredged channels, ending in a small depression in the landscape, where one of the channels is filled with water. This BL provides insight into the understanding of the difference in propagational velocity of the water when a water body is already present within the landscape. Although the orientation of the channels is different, it is the same distance from the origin of the flood, resulting in the same FAT at the start of the channels. |

### 2.2.6. Dike ring 48

The model is also tested on a realistic scenario, since non-synthetic data may contain features that are not included in the training data. This may give further insight into application limitations of the model, which can serve as a starting point for future research.

The study area consists of a part of the Dutch dike ring 48. The motivation behind this area lies in the fact that all input variables used for the DL model were present in this area, with values comparable to those seen by the DL model in the synthetic training datasets.

Dike ring 48 is a cross-border dike ring in the Netherlands and Germany (Kuijpers et al. 2005). The Rhine, IJssel, and the Oude IJssel form the main WWs in the system (DWW et al. 2005). The majority of the land use is destined for agriculture and horticulture, although there are also nature reserves. The main urban areas are constituted by the cities of Doetinchem, Westervoort, Duiven, and Zevenaar, which can be seen in the map in C.1 in Appendix C. The region has a total of 176,000 inhabitants (Arends 2014). Between the cities there are important highways, railways and inner dikes, which function as EEs and thus water retaining structures. The management of dike ring area 48 on Dutch territory has been placed with Rijkswaterstaat and the Rijn en IJssel Water Board (Arends 2014).

The input data of dike ring 48, consisting of the DEM, EE, roughness and WWs, was tailored to the input shape as required by the DL model, implying that the original data with a cell size of 25 by 25 meters was resampled to a cell size of 200 by 200 meters. The flood originates near the bifurcation of the two rivers of the Nederrijn and the IJssel, where the positioning of the rivers acts as a natural barrier for the flood to spread out in two directions only. To do this, the mesh was rotated 30 degrees compared to the true north and then flipped, although the absolute orientation is irrelevant to the DL model as long as the flooding originates in the left lower corner. The location of the mesh is also shown in C.1 in Appendix C. As the bifurcation does not form a perfect angle of 90-degree angle, it was decided to start the flooding of 250 m3/s a few pixels outside the original dike ring, as this would contain some additional pixels on both the north as well as the west side of the domain, which resulted in capturing the area both inside as well as outside of the diked areas. This provides the opportunity to investigate whether the DL model predicts flooding only within the dike ring, and was therefore considered to be of added value. The input maps are shown in Figure 2.1.

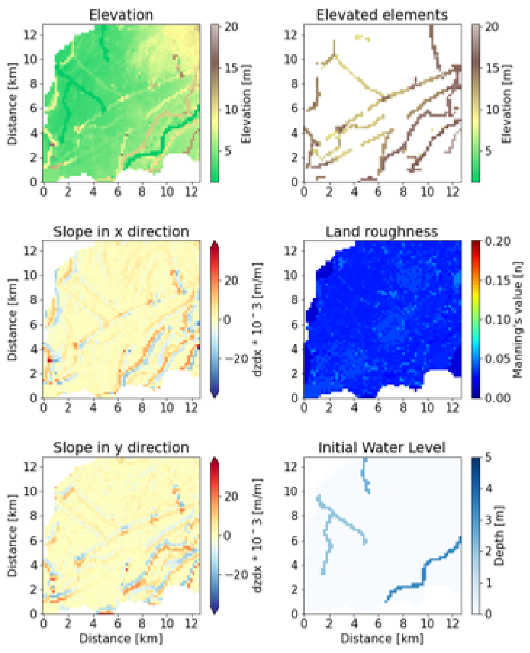


Figure 2.1: Input data for dike ring 48.

The chapter of the results shows the performance of each model on all datasets, as well as all models on dike ring 48. If a higher numbered model is used for a lower-numbered dataset, the relevant input layer will be filled with only zeros. For example, the fourth model which takes an IWL layer as input is applied on the first dataset. As the first dataset has no IWLs, a layer of solely zeros is used. Conversely, the entire input layer has been omitted if it is the other way around, as the corresponding model has not been trained on additional input layers. The only modification to the input data between the first three and the fourth model, is elevating the DEM to the height which corresponds to the IWL within the WWs. The empty WWs would otherwise act as a retention basin, which is not the case. Hence, the only difference is the WWs have been incremented by the DEM instead of an IWL.

## 2.3. Numerical model setup

The aforementioned landscapes were subsequently used as input for Delft3D FM to simulate fluvial flooding. These layers are thus randomly generated and updated for each simulation, leading to unique input and output. The original aim of this research was to predict flooding after an active dike breach. This type of breach has some implications, as the DL model should not only understand the behavior of flooding over an area, but also understand the physics behind a levee breach and its outflow. The flow is determined by the width and height of the gap in the dike which depend on dike properties, but also on the difference in water height upstream and downstream of the breach. This continuously fluctuating variable plays a crucial role in the aforementioned size of the breach, as well as the direct discharge. This therefore has serious consequences for the propagation of the flood over the entire domain. Deriving, understanding, and predicting all these processes solely from the DEM is a challenge for a DL model. Nevertheless, this may be possible by using a time-series approach, where the output of a previous time step is used as input for the next time step, which is of a different DL algorithm than the one used in this study. However, it was beyond the scope of this research to study the development of active levee breaches using a DL model. It is therefore strongly encouraged to explore this in further research as will be elaborated upon in the chapter of the recommendation.

Instead, a constant discharge into the domain was used. The location of the discharge originates from a single point. The subsequent flood can thus be considered to have resulted from an instantaneous dike failure, which remains stable over time. This approximates thus a fluvial flood, which will be named 'flood' hereafter. As the constant flow rate is instantaneously achieved instead of a slow build up, it gives a slight conservative yet safer view on flood propagation. The set amount of water entering the domain per second depends on multiple trade-offs, as elaborated upon in the section of the spatial and temporal characteristics. Trial and error gave proper results for a discharge of 250 m3/s into the domain. With this quantity, a substantial part of the landscape would get flooded while the water usually does not reach the other side of the boundary and avoids therefore filling up of the domain. This depends however strictly on the landscape characteristics. Exploring the predictive power of the DL model with different discharges was not within the scope of this research, but is strongly encouraged to do so in future research as can be read in the recommendations.

The location of the flood's origin could theoretically differ for each data sample and could be anywhere on the map, provided it is given as input to the model. A fixed location of the flooding was chosen for this study to reduce the number of variables and thus the size of the training data. The location of the flood originates therefore in all cases in the lower left corner. A corner is a favorable location in terms of the time it takes for the water to reach the other side of the domain. Flood propagation remains thus visible for an extended period of time. The propagation would be less apparent when the location would be in the middle of the area. However, this location imposes some limitations on the applicability of the model, as will be elaborated in the chapter of the recommendations.

## 2.4. Data preprocessing

The water depth maps over time have been converted into one FAT map indicating the time of arrival of the water per cell. Although water depths and flow velocities are crucial information during a calamity, they require a different type of architecture as it is a time series. Architectures which can handle such time series, such as LSTMs and RNNs, were developed to study the predictions of water depths. These performed poorly however as can be read in the chapter of the results, hence it was chosen to adhere to the FAT. It is a simplification of the original data containing the spatial and temporal evolution of flooding in a single image, as well as a convenient measure to quantify the extent of flooding which is key information during a calamity. This was confirmed by the waterboard during a discussion as well.

The slope in x and y directions were retrieved from the DEM and is used as the main source of input for the DL model, as this showed the highest performance in terms of predictive power. This is in line with rational reasoning as well, as the absolute DEM values do not influence the propagation rate, while the gradient of the DEM between adjacent cells does. Removing the DEM altogether resulted in a slightly lower overall performance, so it was decided to use the DEM as a second input layer which skips a part of the convolutions, as can be seen in the next section.

The scalers which were applied to all data was based on the training data. This assists the DL model in understanding the equally importance of small values in the input data such as the slopes, as well as relative large values such as EEs. Second, a DL model tends to converge faster when the data is scaled (LeCun et al. 2012). All data was therefore scaled between zero and one, the only exception being the slope in x and y directions, as a non-zero value for the inflection point has little physical significance. It was therefore decided to scale the gradient of the landscape between -1 and 1. The scaling ranges are also shown in Table 2.1. Each dataset was divided into training, validation and test data with a 80/10/10 split.

Table 2.3: Overview of the used DL model hyperparameters

| Parameter | Value | Unit |
|---|---|---|
| Kernel size | 5x5 | cells |
| Strides | 2 | cells |
| Padding | same | - |
| Activation function | ReLu | - |
| Optimization function | Adam | - |
| Drop out rate | 0 | |
| Batch size | 32 | Samples |
| Early stopping patience | 25 | Epochs |
| Reduced learning rate | 5 | Epochs |
| Trainable parameters | 44,142,000 | |

## 2.5. Deep learning model architecture

The used DL model architecture in this work was inspired by Tompson et al. (2017), who uses three EDs to predict the pressure field of smoke to accelerate fluid simulations. These fluid simulations were modelled in a three-dimensional space. This is similar to the FAT, albeit the FAT being a simplified derivation from fluid flow. Another commonly used structure in recent years for image segmentation tasks is the architecture of a U-Net (Ronneberger et al. 2015). These two architectures are comparable, as they both concatenate layers of a different form of abstraction before and after the latent space of the ED. The difference however is that the proposed architecture of Tompson et al. (2017) concatenates the three layers after the transposed convolutions of the ED. These three routes all share the same dimensions at the end of the EDs, whereas the U-Net has intermediate concatenation (”copy and crop”) layers. The skipped connections of a U-Net are thus only shared with layers that have the same dimensionality on the other side of the latent space.

Since both architectures show promising results in the literature, a similar proposed structure in this research is used, albeit with some modifications. The created DL model in this research has two input branches. The first and main input branch consists of the slope in x and y directions, and depending on the data set, the EEs, roughness and IWLs layers are added. The DL model performans a multi step convolution on all input layers, which implies multiple convolutional layers after another which increases the dimensionality of the data. Next, the DL model splits these layers into five parallel EDs, as shown in Figure 2.2. Each of the five routes has a different number of layers, resulting in different dimensions of the latent space and carry therefore a different level of feature abstraction within each layer. After the latent space, a decoder returns the data to a similar width and height dimensions of 64x64x32, and then merges it with the second input layer where convolutional filters reduce the depth of the image to one. This results in a concatenated image of 64x64x161. The architecture is thus comparable to two architectures mentioned above, in which layers before and after the latent space are directly connected with each other. The performance of an actual U-Net architecture has been examined, but it performed suboptimally compared to the current setup.

Other architectures have also been explored, as will be discussed shortly in the results section. However, the architecture shown in Figure 2.2 outperformed other architectures with the combination of hyperparameters shown in Table 2.3. An overview of other architectures are discussed in the chapter of the results and discussion.

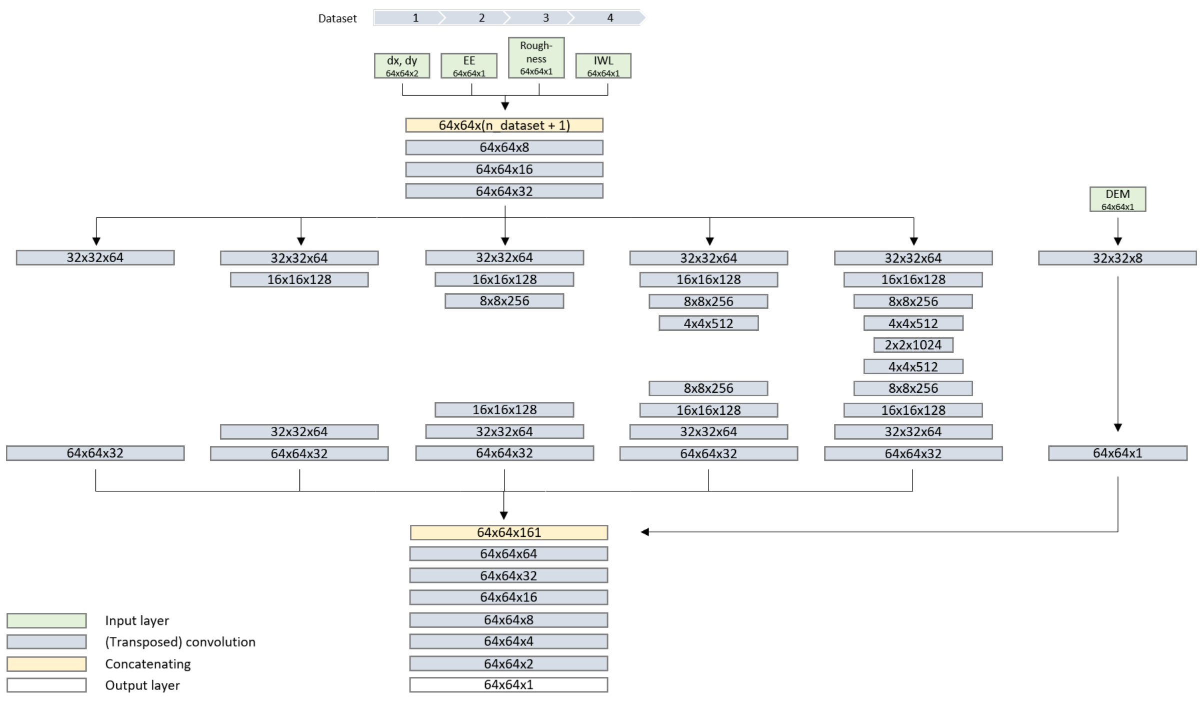


Figure 2.2: Schematic overview of the used DL model, where the main source of information flows through 5 parallel EDs, from where it is concatenated with a secondary input which consists of the DEM.

## 2.6. Post processing

Several processing steps have been applied to the predictions of the DL model. First, the output was scaled back to hourly values, with the values truncated between the minimum and maximum values of zero and fifty hours. A value larger than 48 indicates a FAT larger than the simulation time. Secondly, it is of physical importance to maintain the continuity of the flood. The predicted FAT map was therefore categorized into a binary flooded / non-flooded map. If on t = 48 there was a flooded spot surrounded by dry cells, i.e. it was not connected to the origin of the flood by wet cells, then the flooded spot would be changed to a dry spot.

## 2.7. Evaluation metrics

The mean squared error is used as loss function to penalize the model when making mispredictions. A power function has been chosen since a single misprediction of e.g. 5 hours has more severe consequences in terms of evacuating citizens than 5 mispredictions of 1 hour. This loss function will be minimized when training the model, after which it is validated and tested by unseen data. The result of this loss function on the validation dataset is used to compare different DL model set ups. The second dataset was used for this, due to the relatively large complexity compared to the first dataset and its early availability compared to the third and fourth dataset. The loss of each DL model quantifies the performance of the model on the validation dataset. The most optimal setup is consequently used to create predictions on the test dataset.

The score of the mean squared error is however limited to a general number that applies over an entire prediction, since the output consists of a single image. This implies that the prediction is computed instantaneously, i.e. without consecutive timesteps. However, it is still possible to regard the prediction as a time series. A discretized FAT map consisting of 48 values can be converted to 48 maps consisting of binary values, where zeros represent dry pixels on the corresponding timestep, while ones indicate wet pixels. A major advantage of considering the groundtruth and the prediction as a binary classification task is the ability to calculate confusion matrices which can be used for further statistical analysis. Each FAT image has thus been transformed to a set of binary flood extent maps. An example of such transformation can be seen in Figure 2.3.

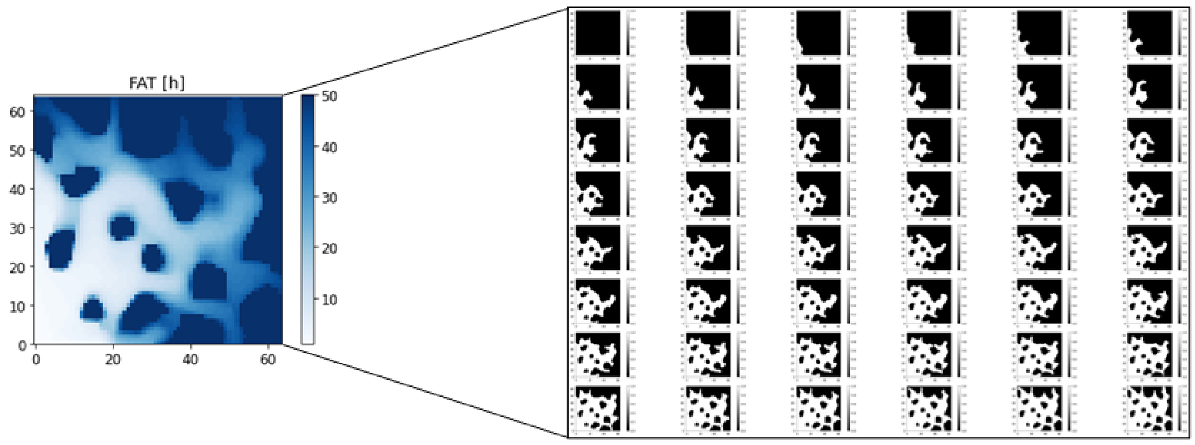


Figure 2.3: Transformation of a FAT map to its corresponding 48 binary maps

These values were used to create more statistical relevant metrics, of which the first one is the aggregate error bias ($B_a$), which indicates underprediction ($B_a$ < 0) or overprediction ($B_a$ > 0), and can be calculated as follows:

$$B_a = \frac{\sum_1^N Gt - P}{N} \tag{2.1}$$

Where P denotes the predicted value, Gt the groundtruth, and N the total number of cells. The next metric is the precision, which considers the misclassified flooded cells as well. It can be considered as the fraction of correctly predicted flooded cells among all predicted flooded cells:

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

Where TP and FP stand for true positives and false positives, respectively. The opposite metric is the False Alarm Ratio (F), which indicates the ratio of the number of flooded cells by the DL model compared to the Gt:

$$FalseAlarmRatio = \frac{FP}{FP + TP} \tag{2.3}$$

The next metric is the recall, also known as hit rate or sensitivity. The recall provides a measure of how many flooded pixels are captured correctly by the DL model:

$$Recall = \frac{TP}{TP + FN} \tag{2.4}$$

Where FN denotes false negatives. Since both the recall and precision represent something different while both being highly useful, it can be convenient to compute the $F_1$-score, which is the harmonic mean of the two. It furthermore works well on unbalanced data, which is the case for the first and last phase of the flood:

$$F_1 = 2 * \frac{precision * recall}{precision + recall} \tag{2.5}$$

The last metric is the accuracy, which represents the average number of correctly predicted pixels, per pixel. Since it is in binary form, it does not take the severity of a wrongly predicted pixel into consideration but rather the percentage of correctly classified cells:

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{2.6}$$

# 3 Results & Discussion

## 3.1. Non-functioning models

Following chapters mainly focus on the best performing DL model architecture. However, numerous types of models with different inputs and outputs have been produced, with different performances. This section is therefore devoted to some of the models that did not perform as desired, as this may serve future research purposes.

The first set-up failed due to a mismatch between the spatial and temporal resolution, as briefly touched upon in the chapter of the methodology. At the time, an hourly resolution was used in combination with cell sizes of 25x25 meters, with two consequences. The first one is the lack of an observable propagational character of the flood, as a static hydrodynamic situation was quickly obtained due to the small surface area. As a result, the domain acted as a container that slowly filled up, rather than a flood that spreads over the landscape. The second implication entailed flooding which would flood a substantial, if not the entire, domain within the simulation period. This led to a trained DL model which would always predict that the entire terrain would get flooded, which is situation specific and is not always the case in real life.

Furthermore, an active dike breach was initially chosen, which is controlled by the Verheij-Van der Knaap beach growth development parameters and formula. This takes numerically the water level upstream, downstream, initial breach width, dike material, and some parameters into consideration to determine per timestep the breach size. The discharge through the breach is thus controlled by these parameters, and gets updated and hence changed in the numerical model for each timestep. This implies that the DL model should not only be able to correctly predict flood propagation over the landscape, but should also understand the physics of a dynamic dike breach. This makes a simulation considerably more complex as the final output depends on a greater number of parameters, which have a strong influence on the output. This is especially the case for the current set up of the DL model, which does not produce a time series as output, but rather an immediate estimation of FAT for all time steps. A step by step approach, which may for example include an LSTM to learn order dependence in sequence problems, could in theory take an involving dike breach into consideration.

Such LSTM model was therefore tested as well in combination with a CNN (Conv-LSTM). Attempts were made to predict water depth with this architecture. However, poor results were obtained, resulting in an output of solely zeros, a static flooding which was the average of the beginning and the end of the flood, or a flooding pattern that kept the same shape throughout the simulation while only getting a larger water depth for this entire shape for each timestep. This may be logical from a minimum-loss perspective, which is relatively low when having the same shape of flooding for each timestep which is incrementally increasing for each timestep, although from a hydrodynamic perspective this makes little sense. Another possible explanation for the poor performance is that an LSTM performs better when it has data of the first few timesteps to make a prediction on the next, which does not fit entirely in this research as the main goal is to emulate the numeric model entirely, instead of using the first few timesteps with a model like Delft3D FM.

Lastly, a timeseries-like approach has been examined with the current DL model architecture. The data was preprocessed resulting in 48 timestep maps that resembled a binary classification of flooded and non-flooded cells. A single training sample consisted of a random timestep as input, with subsequent timestep as output. This trained model was then used in a loop of 48 steps which took its own output as input, to create a timeseries of the flood. This led unfortunately to rapid vanishing or exploding flood propagation, which is slightly as expected as there is no forget gate to control the flow of information, which is a strong property of regular LSTMs. However, it is believed that there is a high potential in the use of LSTMs, as will be elaborated upon in the recommendation section.

Taking all the performances of the created models into consideration, the model described in the chapter of the methodology achieves the best scores when applying it to the test datasets. This model is therefore used to compute the predictions for the test datasets in following sections.

## 3.2. Flooding examples

### 3.2.1. Figure explanation

The upcoming sections of the results and conclusion only take the performance of the DL model's predictions of the test dataset into account. This evaluation consists of 12,000 unique samples, as each of the four test datasets consists of 3,000 simulations. Samples from one dataset are thus not reused in other datasets. A few representative examples will be given in the following section to give an impression of the used input, output, and the created prediction. These examples show furthermore some important behaviors which apply to a substantial part of the dataset, which is discussed first. More examples of each dataset can be found in Appendix B. Next, a spatial and temporal analysis of the entire dataset is given. This is followed by a discussion of the results of the BLs, after which the results of dike ring 48 are presented.

The figures in the following subsections have all a similar layout. Each dataset consists of 'base input', which consists of the DEM and its gradient in the x and y direction. EEs, heterogenous LC, and IWLs are added depending on the dataset number, which constitutes the 'additional input'. The 'output' column is divided into the output of Delft3D FM, which is considered to be the groundtruth. The predicted map corresponds to the output of the DL model. Both output maps are plotted onto the DEM for reference. The error between these two outputs is shown in the last column, where the first image is the actual error between the two. The second image shows the relative error, which may be of importance as a small error in prediction in the final phase of the flood is relative inconsequential compared to the same error in the initial phase of the flood. This relatively error is defined as the actual error divided by the groundtruth FAT.

### 3.2.2. Dataset 1

The only input in dataset 1 is a naturally sloping DEM and its gradient in the x and y direction, where a discharge of 250 $m^3$/s starts at the lower left corner, as explained in the chapter of the methodology. As can be seen from the examples in Figure 3.1, the FAT map follows the same contour lines as in the DEM, where it spreads out radially away from the source of flooding.

The second important observation is related to the connective characteristic of the flood. Since the simulations do not feature phenomena such as multiple flooding locations, seepage, or rainfall, disconnected flooded cells cannot occur. The predictions capture this connectivity quite well, as the post-processing removal of disconnected areas had a relatively limited effect on the final accuracy as can be read at the end of this section.

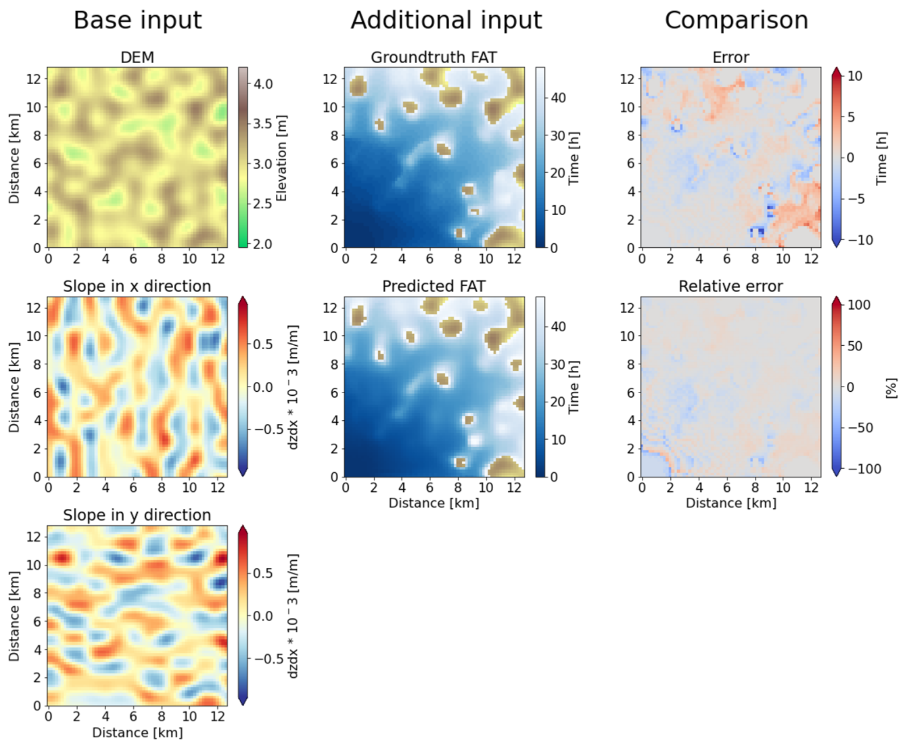


Figure 3.1: Flooding example of the first dataset.

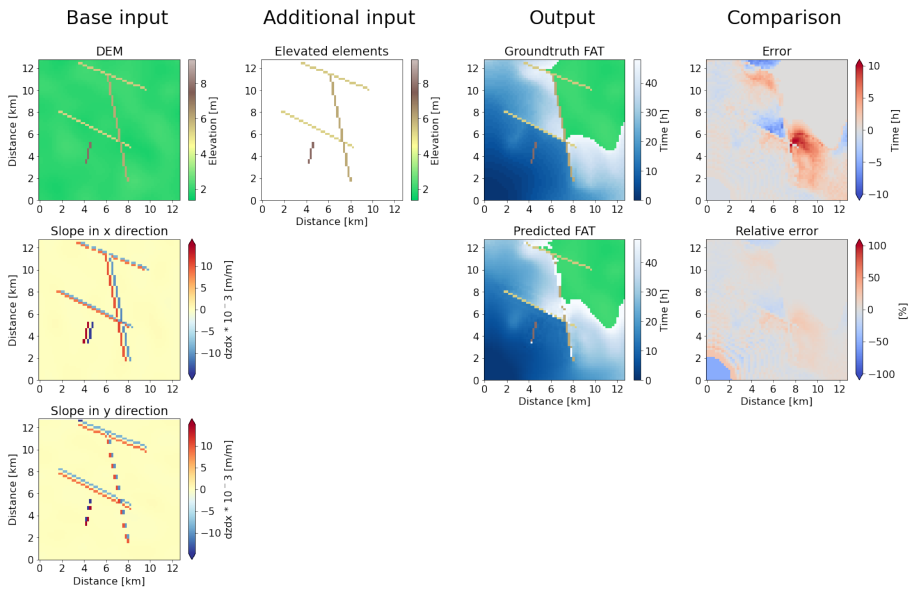


Figure 3.2: Flooding example of the second dataset.

### 3.2.3. Dataset 2

The second dataset includes EEs. These narrow but crucial features have a major effect on flood propagation, which make proper predictions considerably more challenging. This can be seen in the results of Figure 3.2, where the error tends to be larger than shown in the previous dataset.

The location and orientation of the EEs are of importance. EEs located in a region which remains dry during the simulation have little to no effect on the output. In addition, if both sides of an EE eventually flood, the side of the EE where the water reaches the earliest is predicted with lower errors than the other side. A possible explanation of this is that the FAT value of this pixel is both in the numerical model as well as in the DL model inherently dependent on its neighboring cells. An EE forces the water to take a significantly longer travel route. This increase in travel length leaves more room for mispredictions, which can accumulate over distance resulting in larger errors. This effect is enhanced when the flow is perpendicular to the EE, while limited when the flow is parallel as less water is effectively blocked. If only one side floods, then the dry side has the lowest errors, as the DL model has little difficulty understanding that no water should be present after the flooding front, albeit with minor assistance from the post processing step. Lastly, sometimes flooding is predicted on top of the EE which should not be the case, as can be seen in Figure 3.2.

### 3.2.4. Dataset 3

In addition to EEs, the third dataset features different varying land roughness maps for each of the simulations. These are presented as an additional input layer in the DL model. The effect of varying land roughness is relatively limited on the final flood propagation compared to features such as EEs, hence the comparable performance of the DL model, which is shown in Figure 3.3. As these sections focus solely on visual inspections, a more in-depth analysis between the datasets is required to compare this dataset to the previous two, which is given later in this chapter.

### 3.2.5. Dataset 4

The fourth dataset contains all previous features, as well as WWs. The main observation from Figure 3.4 shows that the WWs act in some simulations as conduits in the system, which can form a direct connection between different depressions in the landscapes, without having to fill the entire depression. Instead, these narrow features connect multiple depressions with a small line of a WW and thus a continuous FAT. On the contrary, some WWs in the dataset have no significant impact on flood propagation, as can be seen in the examples in Appendix B.

## 3.3. General performances

The overall performance of each model per number of training samples is shown in Figure 3.5. Adding more training samples than the currently given 24,000 is not likely to improve the results due to the convergence of all lines. This also applies to the complex datasets, which indicates that more complexity per landscape could be added to improve the predictive power of the trained models.

As expected, model performances decreases as the complexity of the dataset increases. The only exception to this trend is the third dataset, which shows a better performance compared to the second dataset. This can be explained when the difference in land roughness values between the datasets is analyzed. In the first two datasets, the default Manning value was 0.023 $s/m^{1/3}$, while for the third and fourth dataset the average Manning value was 0.045 $s/m^{1/3}$. The increase in surface roughness leads to a decrease in flood extent of 6.0%, resulting in fewer errors. An elaboration of this can be found in the section of the spatial analysis. As a result, dataset 3 which has a higher degree of complexity performs better than dataset 2.

The DL model shows across the datasets generally better predictions for the first part of the simulation, corresponding roughly to the first 24 hours of the flood. The differences in the predicted and the ground truth is relatively small but tends to become larger with a larger FAT value. This can be seen back in all four examples of Figure 3.1 to 3.4. Most errors develop at the flood's boundary at the end of the simulation, where the predictions tend to become less accurate.

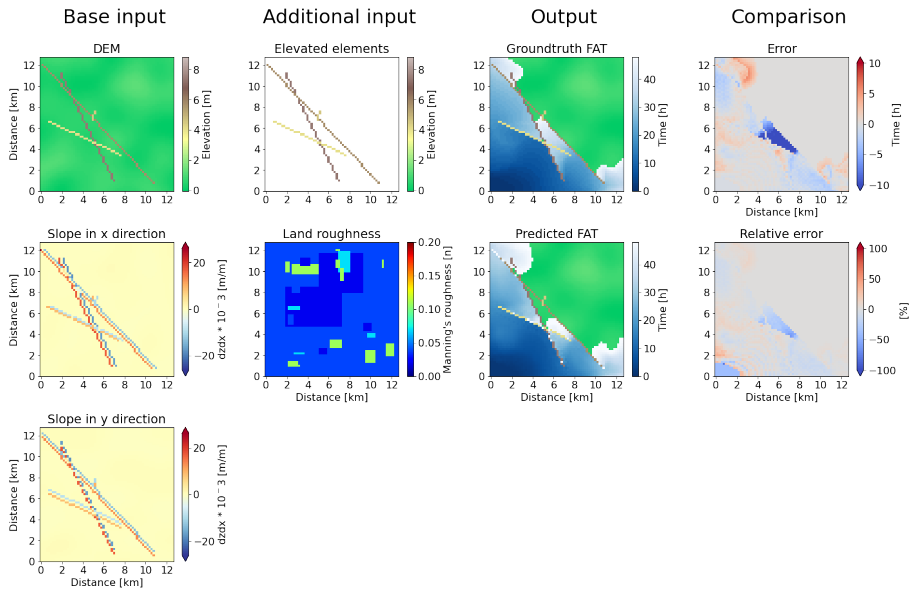


Figure 3.3: Flooding example of the third dataset

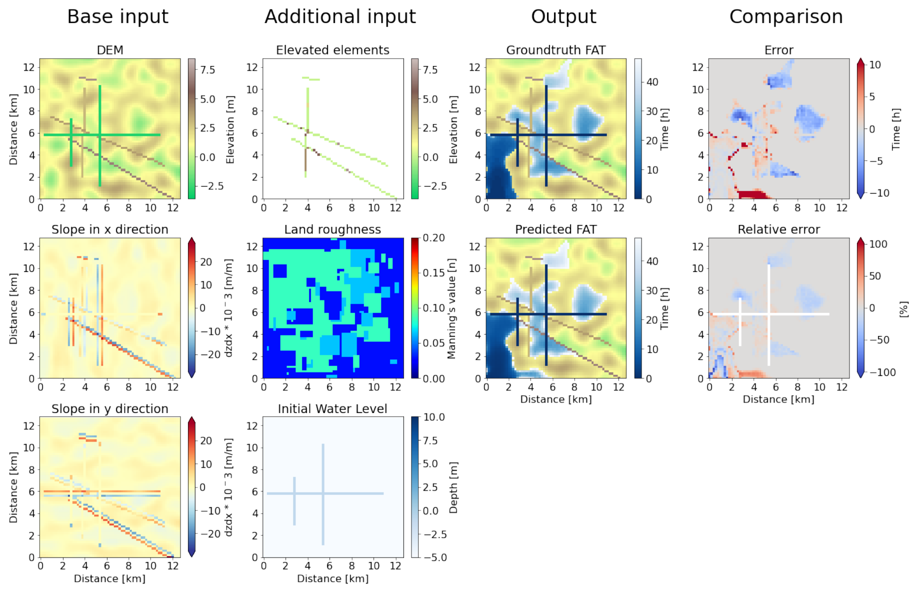


Figure 3.4: Flooding example of the fourth dataset

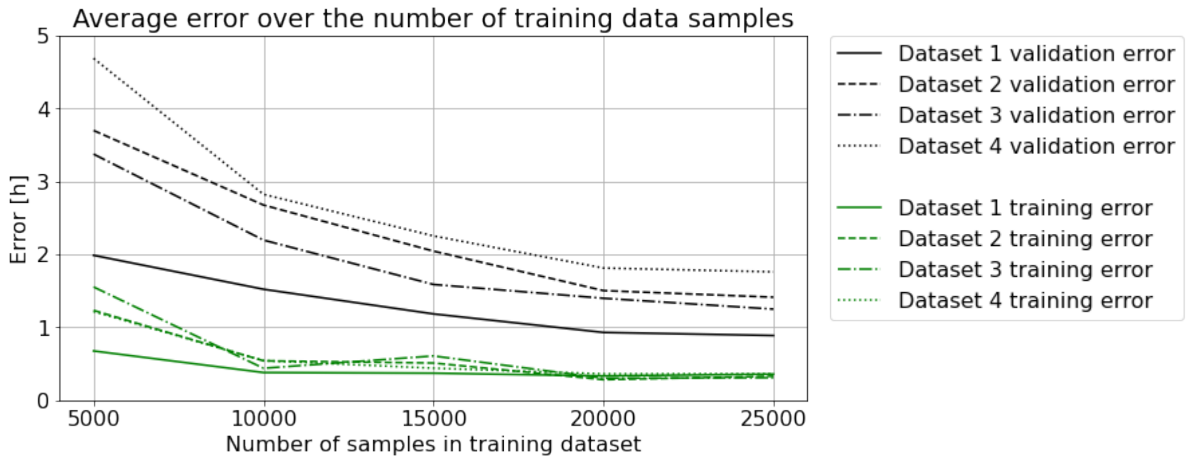


Figure 3.5: The hourly error of the trained DL model per number of training samples for the training and validation dataset. The hourly error has been retrieved and scaled back from the mean squared error.

In addition, the largest errors occur in regions with strong FAT gradients, as is for example shown in the last example of Figure B.1 and the third example in Figure B.2, both in Appendix B. Larger errors occur in locations where a low FAT is followed by a sharp increase in FAT value. This jump in FAT value is caused due to a slow flood development, which may be the cause of greater uncertainty in prediction. The kernels (filters) within the DL model slide over the hidden layers, which perform small calculations on it. The kernel shares multiple cells as its dimensions are larger than a single pixel. While this is necessary to give the DL model spatial awareness, it also leads to sharing of weights and biases over the kernel's dimensions, resulting in a smoothed and averaged output. This makes strong increases in FAT a hard task to fulfill.

Some general visual observations became apparent when examining other individual samples. First, there is occasionally a tendency to interpolate flooding between two flooded regions in the immediate vicinity, which are not separated by an EE. An example is shown in the lower right corner of the second example of Figure B.2. Another process which tends to be hard to predict is the rise in water level in the proximity of the initial location of the flood. Since these pixels get flooded relatively quickly, they have low FAT values. As simulation time increases, the water level starts to rise gradually resulting in additional flooded cells at the end of the simulation, which are located in a region which was predominantly flooded at the initial phase of the flood. These flooded cells with a high FAT value are in close proximity of pixels with a low FAT value, resulting in a steep increase in FAT gradient with the same complications as elaborated upon in the previous paragraph. An example of this is also shown in Figure B.2 in the lower left corner.

The next finding concerns the observation of a relatively large error when at the end of the simulation time a depression is at the tipping point to be completely filled. A small prediction error can result in (an absence of) flooding of the next depression, misclassifying a relatively large area. An example is shown in the third example of Figure B.1. However, it should be noted that this usually involves a misclassification of only a few hours, which is relatively small for larger FAT values.

Lastly, the post processing step reduces the error of the predictions of the first till the fourth test dataset with 4.21%, 9.03%, 4.22% and 16.19%, respectively.

### 3.3.1. Computational time

The main goal of this research is to create an accurate surrogate model which reduces the computational time compared to conventional methods. The required computational time to predict each test dataset is therefore presented in Table 3.1, where the percentage of reduction in time compared to the numerical software of Delft3D FM is shown. In addition, the performance of each model on other datasets is also presented. The error below the diagonal is smaller than above it, as the model misses relevant input layers as explained in the chapter of the methodology.

The computational time required by the DL model is thus two orders of magnitude smaller on a CPU. The acceleration on a GPU was on average 0.0086% compared to Delft3D FM. The training time on the CPU and GPU was in the order of one hour and two days for each dataset, respectively.

Table 3.1: The reduced computational time for each trained model compared to Delft3D FM, and its corresponding performance across all datasets. The diagonal is shown in bold, which is the dataset on which the model is trained upon.

| | Dataset 1 | | Dataset 2 | | Dataset 3 | | Dataset 4 | |
|---|---|---|---|---|---|---|---|---|
| | Error [h] | Time [%] | Error [h] | Time [%] | Error [h] | Time [%] | Error [h] | Time [%] |
| Model 1 | **0.91** | **98.27** | 2.70 | 98.37 | 3.45 | 98.64 | 5.46 | 98.55 |
| Model 2 | 1.15 | 98.21 | **1.41** | **98.20** | 2.43 | 98.74 | 3.95 | 98.78 |
| Model 3 | 1.24 | 97.81 | 1.45 | 98.57 | **1.25** | **98.44** | 4.04 | 98.57 |
| Model 4 | 1.20 | 98.57 | 1.44 | 98.40 | 2.39 | 98.42 | **1.76** | **98.60** |

## 3.4. Spatial analysis

The model's applicability for varying landscape characteristics is investigated with a 'spatial analysis'. Although 'spatial' may indicate an analysis on a 2D surface, here it is considered as the relation between the error and a collection of landscape features. The analysis is divided into different subsections, based on different landscape characteristics such as the flood extent, elevation, flood propagation velocity, roughness, and the number of EEs and WWs. Additional correlations between variables can be seen in Appendix D. These include histograms of the frequency of occurrence, which is crucial to assess the robustness of the figures. The ultimate goal of the spatial analysis is to estimate whether a particular landscape is suitable to apply the DL model to.

### 3.4.1. Flood extent

Different topographies and landscape characteristics lead to different flooding patterns, resulting in a different flood extent. The domain wide error per flood extent is shown in Figure 3.6. The smallest errors occur on both sides of the figure, which is a balance between two phenomena:

1. As explained in the previous section, the DL model faces some challenges in predicting the exact location of the flooding front, although it is able to roughly understand where it is. Even without the post processing step it generally understood that no flooding can take place behind this front. This means that predicting the maximum FAT value behind the flooding front is done relatively easily, resulting in lower errors in the dry regions than in the wet regions. There is therefore a limited area where errors will occur, reducing the total error across the entire domain, resulting in a relatively low mean error for low flood extents as can be seen in Figure 3.6. The error per cell which will actually flood is however larger, as there is a low probability of a gradual increase in FAT values which decreases the total performance, as explained in previous section. This is also confirmed by Figure 3.8 and Figure 3.9, where there is a reduction in error for an increased flood extent.

2. The second process which plays a role is the flatness of a landscape. Large elevation gradients lead to strong fluctuations in FAT values, which are typically more complex and thus challenging for the model to predict, resulting in larger errors. Flat regions are therefore beneficiary in terms of model performance, especially when looking at the error per flooded pixel.

The extent of flooding is strongly influenced by the number of EEs and their height, as well as the natural relief of the landscape. The performance of the model thus depends on the degree of hilliness, which affects the total flood extent as shown in Figure 3.7. The elevation difference is defined as the height of the maximum DEM minus the lowest DEM value within the same data sample. It is a first approximation of how mountainous a particular landscape is. As more water is required to fill up a non-flat terrain, less area is flooded. On the contrary, a flat landscape has usually larger flood extents with lower errors due to the small gradients in FAT, as can be seen on the right side of Figure 3.6. The key message here is that there is a correlation between error and flood extent, but not a causation, hence the different y-axis labels for Figures 3.6 and 3.7. The common denominator between these two is the topography of the landscape. The second and third dataset in Figure 3.7 have larger elevation differences compared to the first dataset, which is due to the EEs. The same applies to the fourth dataset, where the WWs are lowered in the DEM. The relation between this topography and error is explained in the following subsection.

### 3.4.2. Elevation

Figure 3.10 shows the average error per elevation value for the four datasets, where four main points can be concluded. Firstly, the fourth dataset shows a significantly lower error for the lowest DEM values compared to the other datasets. This is caused by the additional IWL input, which has per definition a FAT value of zero since water is present from the beginning of the simulation. The FATs of these WWs are therefore easy to predict, which lowers the average error for lower elevations for the fourth dataset.

Secondly, this effect is reversed for the second and third datasets. Visual inspections have shown that if an EE is located in a depression where on both sides of the element the elevation of the landscape is significantly lower, then the DL model tends to predict flooding on top of the EE. An example is shown in the first example of Figure B.3. This is likely caused by the low elevation of the surrounding area, where

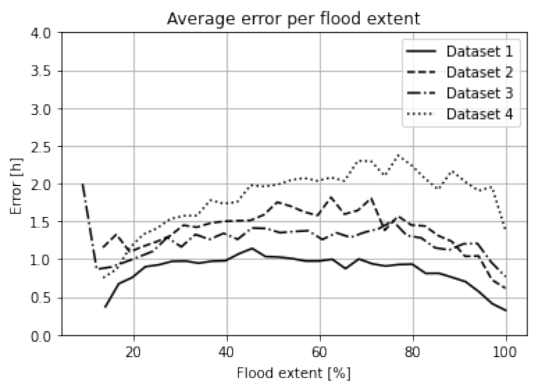


Figure 3.6: Average error for the entire map per flood extent

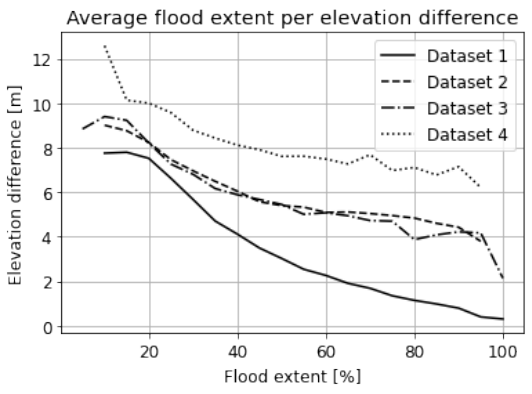


Figure 3.7: The average elevation difference over the flood extent

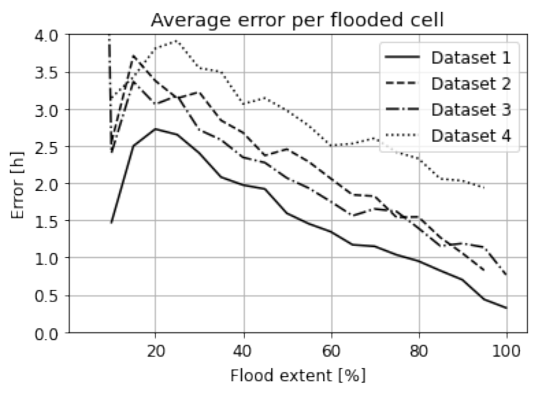


Figure 3.8: Average error per wet cell

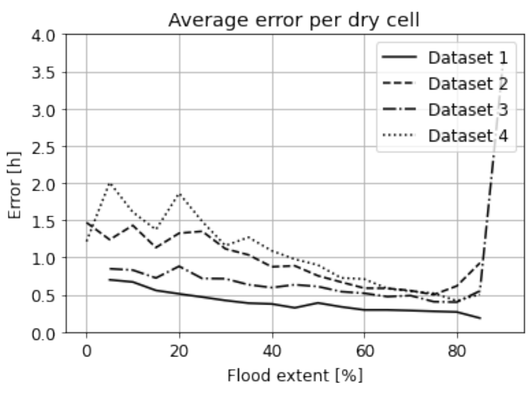


Figure 3.9: Average error per dry cell

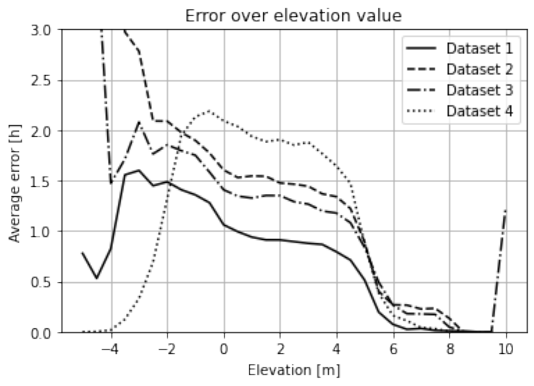


Figure 3.10: The average error per individual elevation value for all the datasets

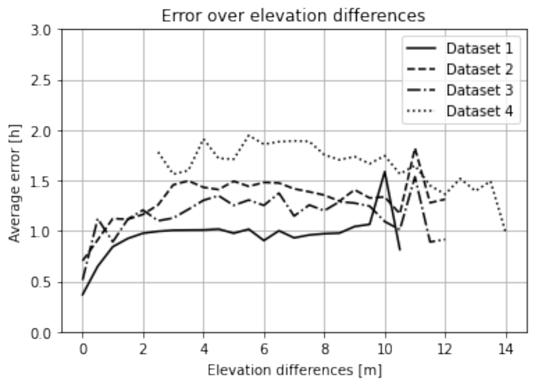


Figure 3.11: The average error over the largest difference in DEM value in a simulation

the EE is just a relatively thin line which gets diluted by the kernel. These errors could be removed by adding another post processing step, which removes all flooding on top of certain elevated elements, although one should be certain to know on forehand which EEs are non-overflowable. The next post processing step of labeling all non-connected flooding parts will then remove these residual flooding outbursts. However, such post processing has not been taken as it is not possible to determine on forehand which EEs can overflow.

Thirdly, there is a decreasing trend in average error for higher DEM values. An explanation can be found in the fact that the largest errors in prediction occur in regions which will get wet. The dry regions on the other hand have a relatively low error, as explained in the previous section due to the understanding of no flooding behind the flooding front. This is confirmed by Figures 3.8 and 3.9 as well, which shows a significantly higher error for flooded cells. The consequence of this implies that higher laying areas get on average flooded less frequently, resulting in more often drier regions for larger DEM values, with a resulting lower average error.

The last main observation is the enhanced effect of this from an elevation of 5 meters and higher. This coincides exactly with the height of an average EE, which is 5.00 meters as can be seen in Table 2.1. Usually, though definitely not always, these EEs block all the water, resulting in little to no flooding on top of these elements which can be predicted with relative ease, which causes a drop in error.

The error over the elevation difference within a simulation is plotted in Figure 3.11. It becomes evident from the figure that the error is the lowest for the landscapes which tend to be flat, although the error does not increase after an elevation difference of roughly 3 meters. In fact, there is a small tendency of a decreasing error for larger elevation differences, which is especially the case for dataset 4. This is likely caused by a smaller flood extent, as elaborated upon in the previous section.

## 3.4.3. Gradients

The landscape and its gradient play thus a key role in flood propagation. It is therefore of great relevance to know the effect of specific slope values on the expected error. This is done for the slope in both x as well as y direction. Several conclusions can be drawn from Figures 3.12 and Figure 3.13, where the error over the slope in x and y direction is presented, respectively.

Firstly, the narrow line of dataset 1 compared to the other datasets can be explained by the absence of EEs, which normally would form a sharp increase or decrease in gradient. As the same parameter settings were used to generate the natural varying landscape for all datasets, it shows the delineation between the gradients generated by the Perlin noise generator, while gradients larger than roughly

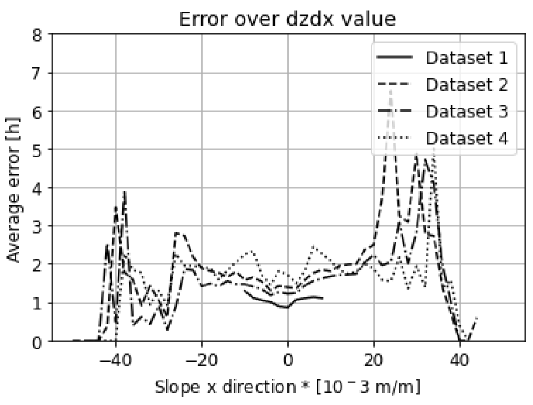


Figure 3.12: The error of the DEM gradient in x direction

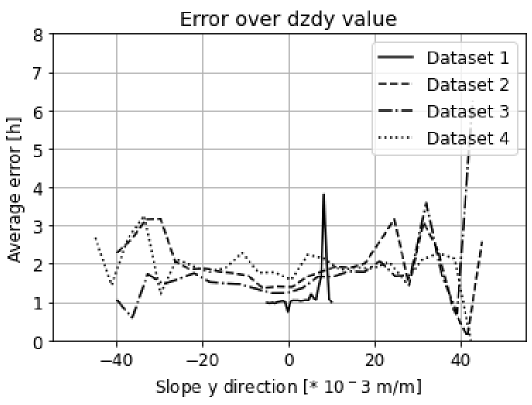


Figure 3.13: The error of the DEM gradient in y direction

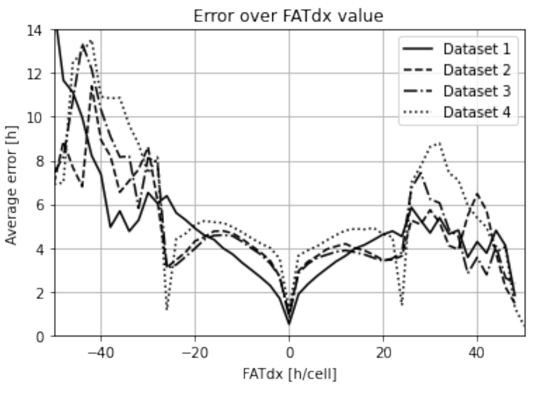


Figure 3.14: The error of the FAT gradient in x direction

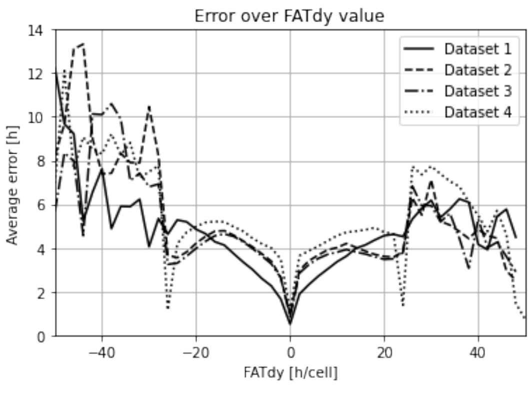


Figure 3.15: The error of the FAT gradient in y direction

$10 * 10^{-3}$ are caused by the EEs or WWs. The effects may look small, but there is a tendency for larger errors for larger inclined landscapes. The peak in Figure 3.13 in dataset 1 is caused by a single outlier, as there is a small frequency of occurrence for such gradients.

Secondly, the global shapes of the graphs are not symmetrical. Especially Figure 3.12 shows a clear trend of larger errors for larger positive values. As the flood in all simulations originates at (x,y) = (0,0) and mainly propagates towards increasingly positive x and y directions, slopes larger than $+10 * 10^{-3}$ indicate the presence of an EE facing the origin of the flood. The flow is either completely blocked by this EE, or delayed when the elevation is not high enough. If the latter, larger errors are expected at the right side of the EE as the continuation of flow is harder to predict. However, in most cases, the elevation of such an EE is sufficient to retain the water leading to no overflow, causing higher errors at the side of the EE facing the origin of the flood, leading to larger errors at the positive side of the graph.

A similar analysis is performed on the gradient of the groundtruth FAT values, which is depicted along the x and y directions in Figure 3.14 and Figure 3.15, respectively. FAT gradient values near zero mainly indicate flooding in flat regions. Flood propagation in this part of the graph follows predominantly the gently sloping natural elevation, with a resulting low error as elaborated upon in the previous section. Larger gradients in FAT result in larger errors. Moreover, the positive side of the graph indicates flooding from left to right, which tends to have a lower error than vice versa. This implies that flooding in the direction towards the origin to the flood has larger errors compared to water propagating away. This has two possible causes: 1) by the on average longer travel distance when traveling towards the origin of flooding as the water needs to make a turn, and 2) as floods tend to spread out radially away from the origin of flooding, the model is trained more on this direction. If in future research a random point of flooding on the domain is chosen, it is assumed that the shape of the figure becomes more symmetric.

### 3.4.4. EEs and WWs

The effect of the number of EEs and WWs can be seen in Figure 3.16 and Figure 3.17, respectively. A clear relation emerges from the figures, where the error increases for an increasing number of EEs and WWs, as the landscape becomes more complex.

### 3.4.5. Roughness

Figure 3.18 shows the average domain wide error over the domain wide roughness value, while in Figure 3.19 the average of the pixel specific error and pixel specific land roughness is plotted. They both show a similar trend, where large Manning values result in lower errors. On the one hand this is odd, as larger roughness values occur less often in the training dataset as can be seen in Appendix D. On the other hand, larger Manning values lead to a lower propagational speed of the flood, resulting in less flooding and thus lower errors. The downward going trend in Figure 3.18 is more pronounced than in Figure 3.19 as the final flood extent is more influenced by the average overall land roughness value rather than some individual roughness cells.

### 3.4.6. Overprediction and underprediction

The distribution of the error between underprediction and overprediction does not seem to be evenly spread within the flooded and dry regions. Underprediction, defined as predicting a too large FAT value in a particular cell, is by definition bounded by regions which get flooded ultimately. Overprediction on the other hand can take place in both flooded as well as dry areas. Thus, overprediction and underprediction in flooded areas partially offset each other, while the errors in dry areas are only due to overprediction. As a result, the balance between overprediction and underprediction is relatively moderate in flooded areas while overprediction is dominant in non-flooded areas. These two factors minimize the total loss over the dataset, which means that these two factors have to level out each other. As a result, there is an average of underprediction in flooded areas and overprediction in dry areas. This can be seen back in Table E.1 in Appendix E.

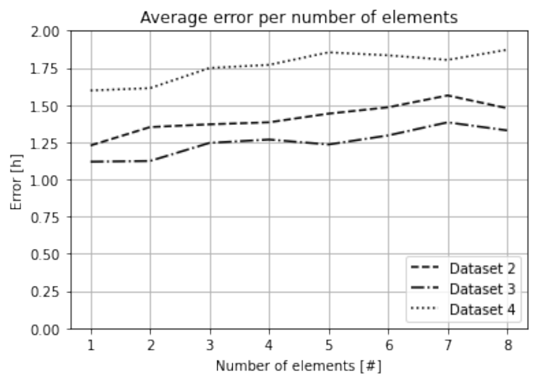


Figure 3.16: Error over the number of elevated elements

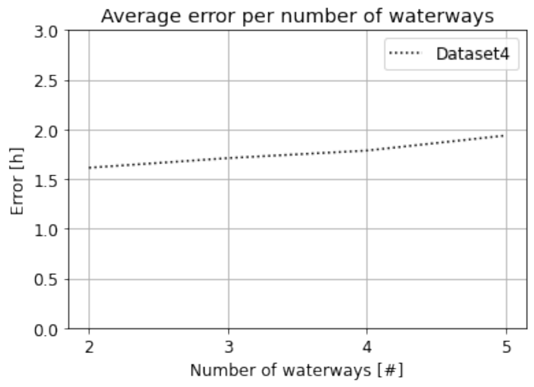


Figure 3.17: Error over the number of water ways

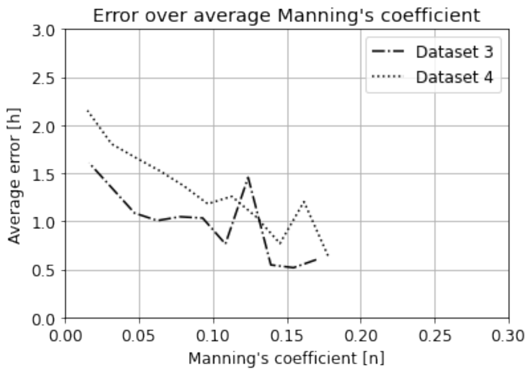


Figure 3.18: Error over the domain wide averaged roughness value

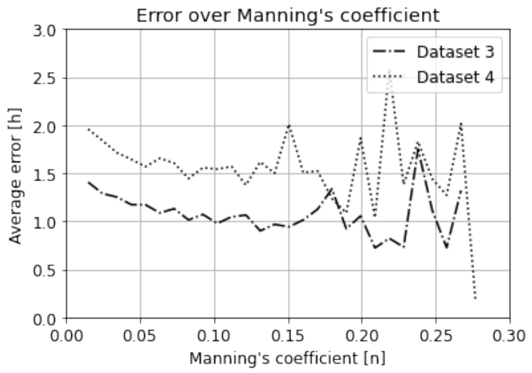


Figure 3.19: Error over the pixel unique roughness values

## 3.4.7. Correlation

Although the above relationships are useful for estimating whether a particular landscape is suitable to apply the DL model to, it does not reflect the relationship between the other variables and the relationship between them. Furthermore, the figures do not show the frequency of occurrence which is crucial to assess the robustness of the figures. A correlation matrix between each variable is therefore included in Appendix D, as well as histograms of the frequency of occurrence of each combination of variables.

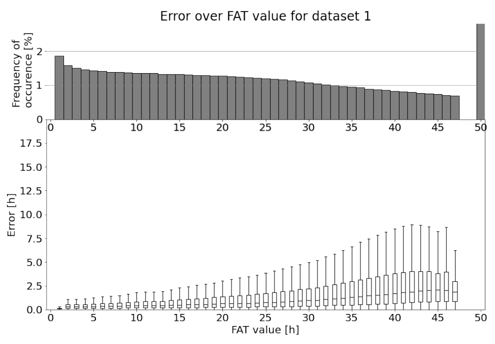
## 3.4.8. Concluding remarks

The model is especially performing well on landscapes with low gradients in DEM and FAT, implying that flat landscapes with a limited number of EEs with an absence of WWs are modelled best. The error per flooded cell can be significant when only a fraction of the total domain is flooded, but decreases as more of the terrain is flooded. The same applies to simulations with large roughness values. The model has a decent understanding that an EE is generally not able to be overflown. If water does flow over or around the EE, then the side that floods the latest faces greater errors. The orientation of the EE is of importance in estimating the final error, as the largest errors occur on the side facing the origin of the flood. The direction of the flood affects the average error as well, which is lower for a flood that is moving away from the origin. Lastly, there is a strong increase in error for areas with a large gradient in FAT, again indicating that an area with a steady increase in FAT value can be predicted better than an area with irregular jumps in FAT value.

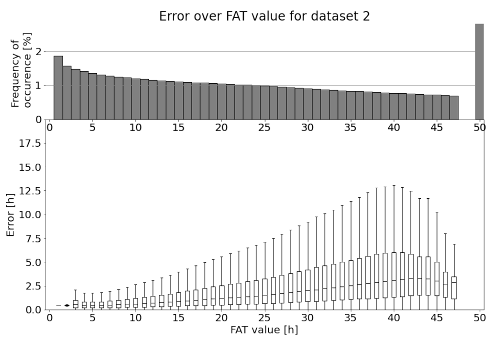
## 3.5. Temporal analysis

The model's performance in time is evaluated with a temporal analysis. Figure 3.20a - 3.20d shows the deviation per predicted FAT value. The majority of each prediction falls within a range of a few hours from the groundtruth. This is particularly the case for the onset of the flood, although the deviation tends to increase for larger time steps, and starts to appear earlier for more complex datasets. Two confusion matrices between the predicted and the groundtruth values are shown in F.1 in Appendix F.

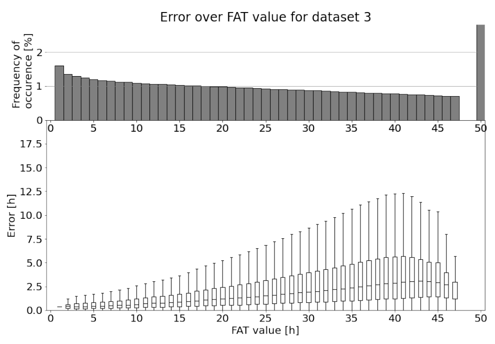
After post processing the output into 48 binary images, a confusion matrix can be computed for each timestep, which is shown in Figure F.2 in Appendix E. The confusion matrix forms the foundation for other relevant metrics, of which the first one is the $B_a$. Figure 3.21 shows that the average value throughout all timesteps is around zero. This indicates that there is not a massive overall tendency towards either underprediction or overprediction when considering all pixels of all simulations. This is as expected, as the total error should be as close to zero as possible to have a minimum loss. The only exception is the fourth dataset, which has to do with the additional post processing step that WWs receive a FAT value of zero if they did not already have so. This implies that when a relatively high FAT value is present at a WW, it is brought back to a lower value. However, the initial overpredicted value leads to compensation with lower values elsewhere, resulting in an average underprediction throughout the simulation. Next, it becomes clear that at the beginning of the flood there is a tendency toward underprediction, which turns into overprediction in the interval between 35 to 40 hours. This can be explained by the fact that the propagational flood velocity is relatively high at the beginning of the simulation, as the water has little area to cover which results in a rapid spread of the flood. This flood advancement slows down over time as more area gets covered. Although the DL model is able to capture this shift in velocity relatively well, it is not perfect yet in doing so with a resulting underprediction followed by overprediction.



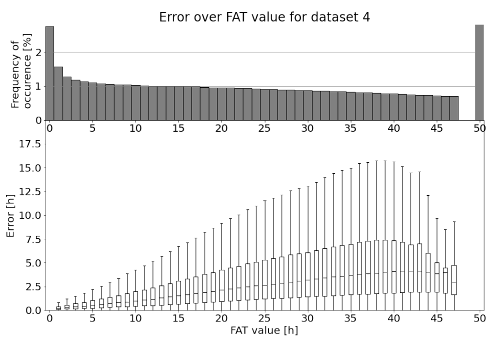
(a) Dataset 1



(b) Dataset 2



(c) Dataset 3



(d) Dataset 4

Figure 3.20: Error distribution per FAT value of all datasets

This effect is reflected in the precision, where an increasing number of false alarms are present after the initial phase of the flood. The hit rate remains nonetheless high, resulting in an average decent F1 score that only decreases slightly over time.

Lastly, the accuracy is for all datasets persistently decreasing over time, where at the end of the simulation around 5% of the pixels are wrongly classified by the DL model. The accuracy is lower for more complex landscapes, indicating once again that the DL model has an increasing difficulty in predicting more complicated landscapes, especially at the end of the simulation.

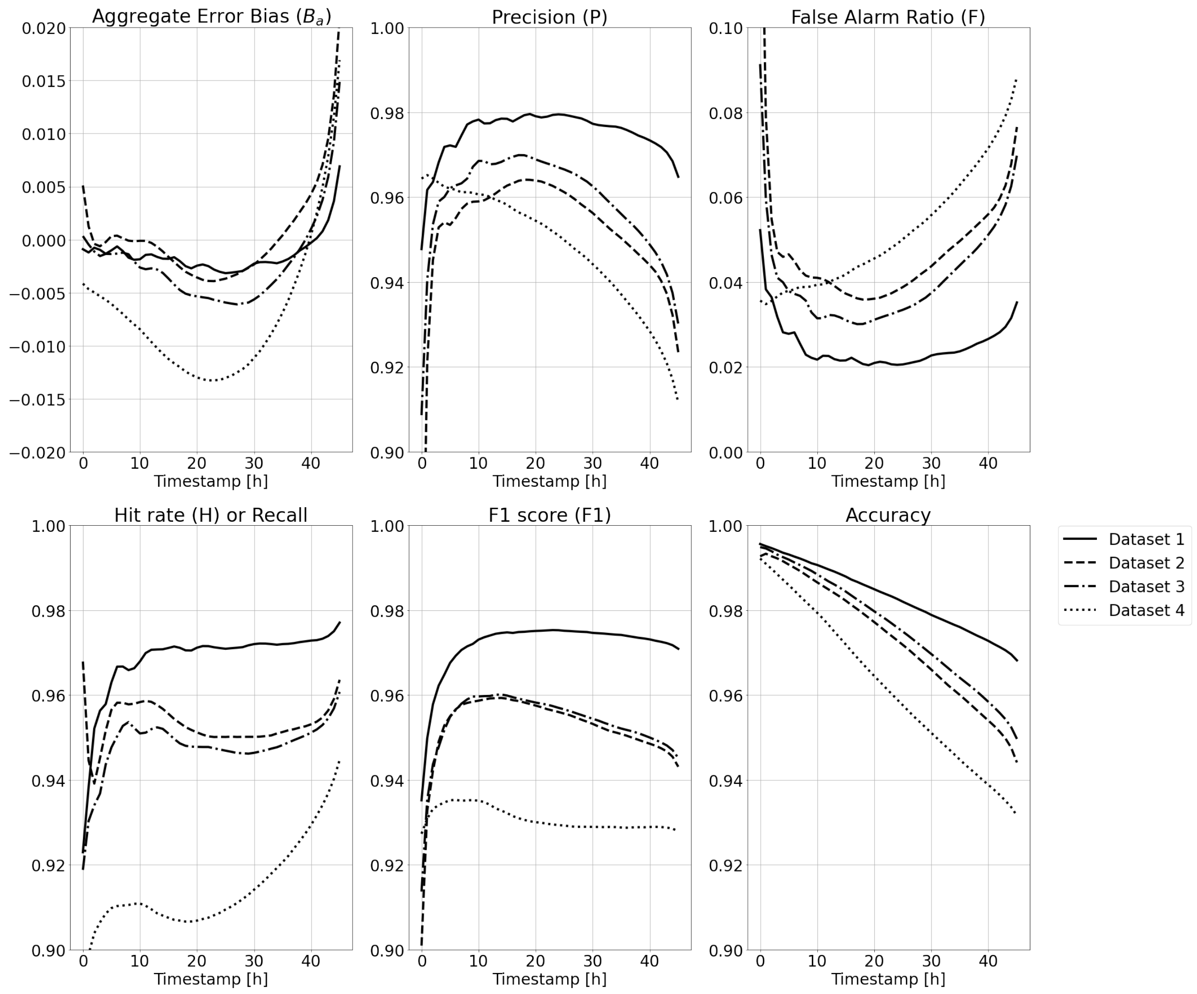


Figure 3.21: Metrics over time for each dataset

## 3.6. Benchmark landscapes

Previous temporal and spatial analyses have given general performances which apply to all test datasets. However, they do not give an indication on the performance of distinct flow conditions. 15 BLs were therefore created to test the model for these specific cases, as described in the section of the methodology. An explanation for each prediction is given below, where only the relevant input map is shown. An overview of all input layers are shown in Appendix G. Only model 4 has been applied to the BLs since it is trained on all features which occur in the BLs.

Figure 3.22 shows incorrectly flooding behind the EE for BL1. The post processing step is not able to eliminate this part due to the fact that the DL model predicts flooding on top of the EE on the upside of the image. Nonetheless, the majority of the left and initial stage of the flooding are predicted relatively well.

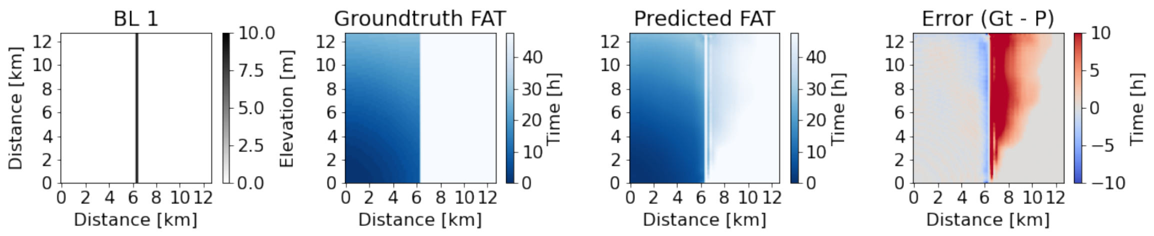


Figure 3.22: Prediction of a non-overflowable EE (BL1).

The main purpose of BL2 is to examine the DL model's understanding of an overflowable EE, which results in a jump in FAT value. This jump is visible in the abrupt change in color and thus FAT in Figure 3.23. The difference in FAT on top of the EE in the groundtuth image is 8 hours, whereas the predicted value is 1.5 hours. This deficit can be seen back on the right side of the image, where more area is incorrectly flooded due to the incorrect prediction of the jump in FAT. Despite this, the DL model is able to recognize that the lower part of the EE is ought to overflow first, resulting in a wet lower part of the right side and a dry part at the upper right corner.

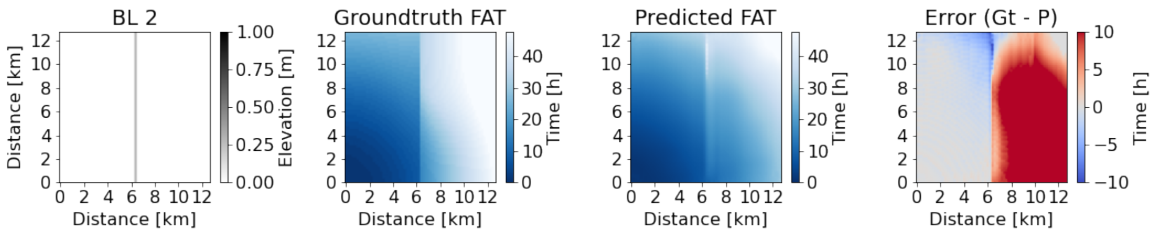


Figure 3.23: Prediction of an overflowable EE (BL2).

The output of BL3 can be seen in Figure 3.24, and shows an underestimation of the flow through the narrow gap, implying that the DL model faces challenges in predicting the continuation of the flow of water once reaching an undershot in an EE. The undershot effect is overruled in the prediction of BL4 by the overflowable EE as can be seen in Figure 3.25, which is comparable to the prediction of BL2.

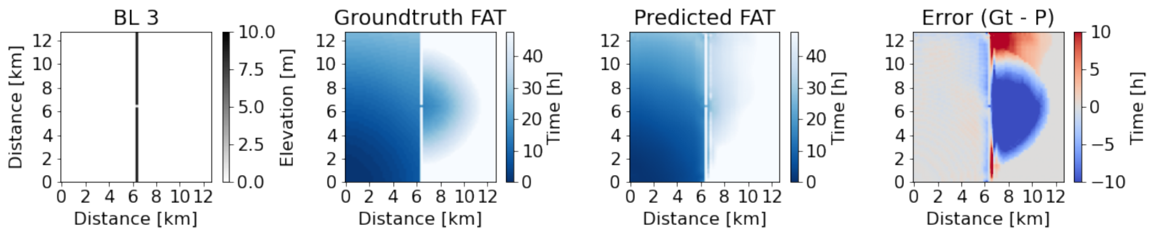


Figure 3.24: Prediction of a non-overflowable EE with an undershot (BL3).

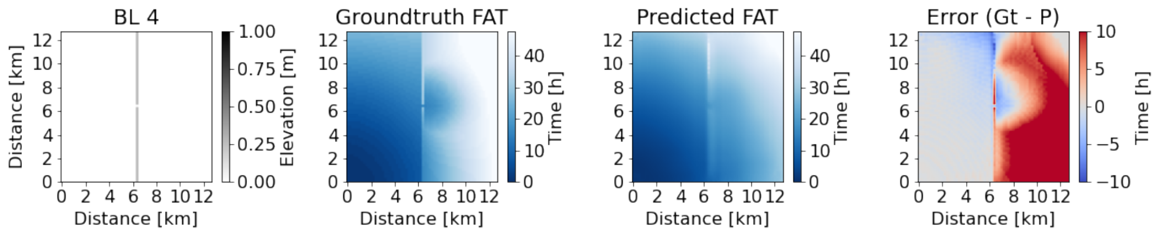


Figure 3.25: Prediction of an overflowable EE with an undershot (BL4).

The result of BL5 is noteworthy, which shows a better prediction in the vicinity of the WW as can be seen in Figure 3.26. This is likely due to a higher frequency of occurrence of similar situations in the training dataset, where EEs can be crossed by WWs in the fourth dataset. Thus, the presence of a WW in the opening of the EE is beneficiary for model performance, indicating that more training data with undershots may improve performance.

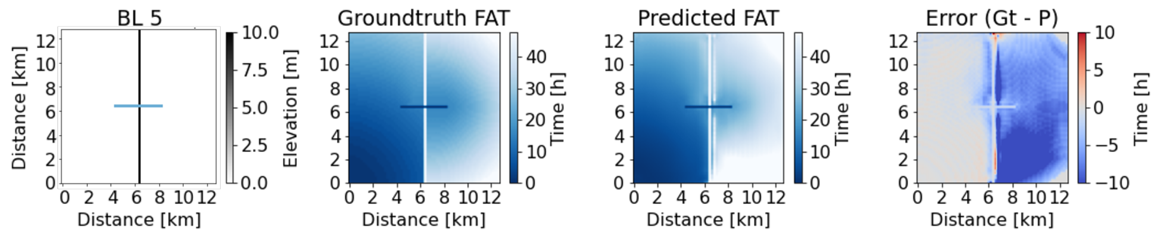


Figure 3.26: Prediction of a non-overflowable EE intersected by a WW (BL5).

Figure 3.27 shows for BL6 a similar trend as was shown in the section where examples of the datasets were presented, where the error of FAT continues to increase with rising FAT values, with a maximum error near the flood front where it decreases thereafter. It should be noted that due to the presence of EEs, varying roughness, and WWs, there is not a single training sample in the fourth dataset which has seen a completely flat, homogeneous landscape, making the prediction of this BL more challenging than expected.

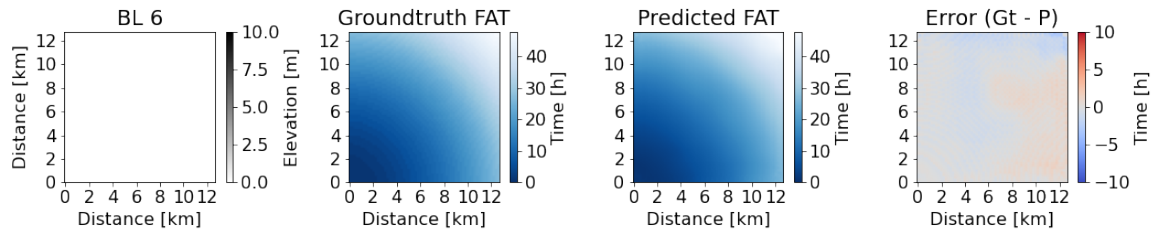


Figure 3.27: Prediction of an homogeneous landscape (BL6).

BL7 shows a decent prediction of a landscape with an enclosed subsection as is shown in Figure 3.28. However, flooding is still predicted within this area. Except for the right part at the subsection, the model performs reasonably well. A similar result can be seen for BL8 in Figure 3.29, where only the adjacent area near the flood origin can be flooded due to the enclosement. Contrary to what happened at e.g. BL1, there is no flooding on top of this enclosement, resulting in no flooding elsewhere in the domain due to the post processing step.

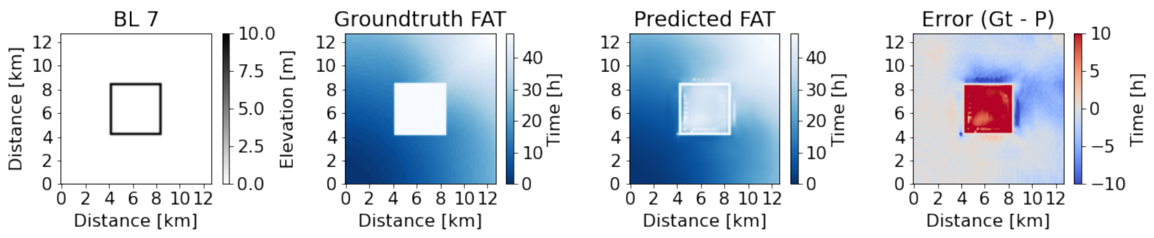


Figure 3.28: Prediction of a non-overflowable dike ring (BL7).

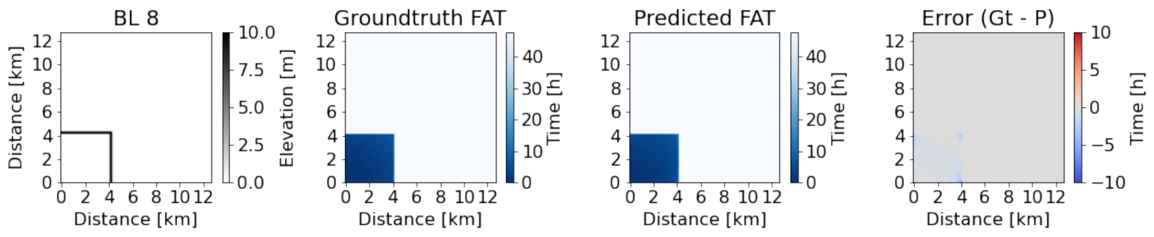


Figure 3.29: Prediction of an encapsulated flood (BL8).

Figure 3.30 and Figure 3.31 show for BL9 and BL10 a similar performance in terms of initial timesteps, where the FAT is predicted correctly. The results degrade rather rapidly afterwards, indicating that water flowing through relatively narrow sections is difficult to predict. A possible explanation for this may be due to the intrinsic set-up of the DL model. The kernel, which moves over the hidden layers, gets diluted with zero values from the surrounding areas due to the convolutions. The same process plays a role in BL11 in Figure 3.32, leading to poor performances. Note that there is little to no training data available where water flows for a longer period of time through a narrow section.

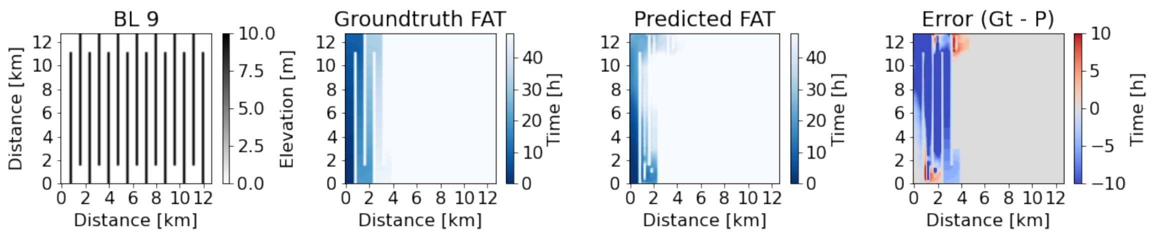


Figure 3.30: Prediction of sharp turns in narrow flow conditions (BL9).

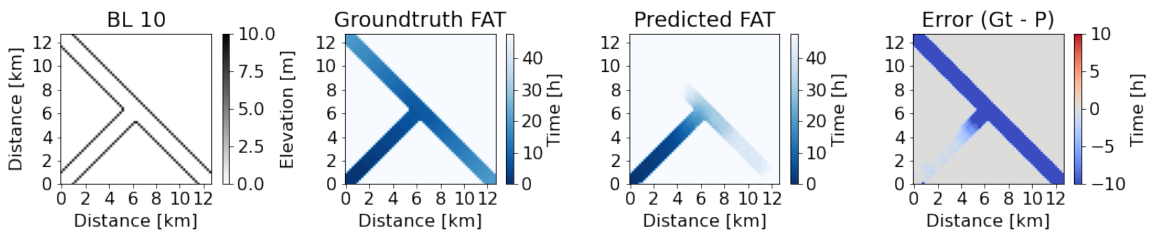


Figure 3.31: Prediction of a bifurcation point (BL10).

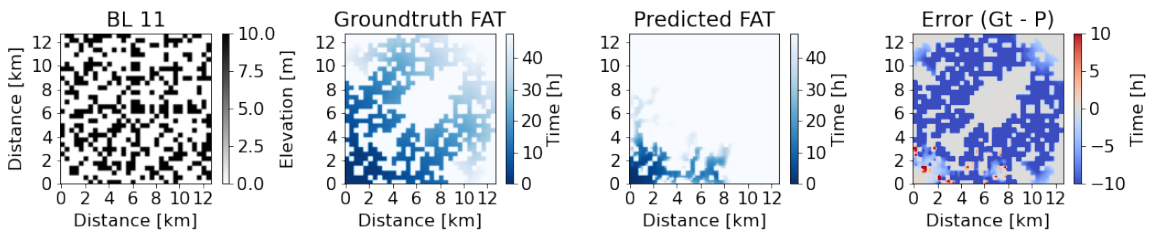


Figure 3.32: Prediction of a highly complex, maze-like landscape (BL11).

BL12 examines the effect of a strong change in land roughness, which is shown in Figure 3.33. A similar pattern between the groundtruth and predicted output can be seen, although somewhat blurred. It shows a clear link between LC and FAT, which is quite well understood by the DL model although there is some severe overprediction.

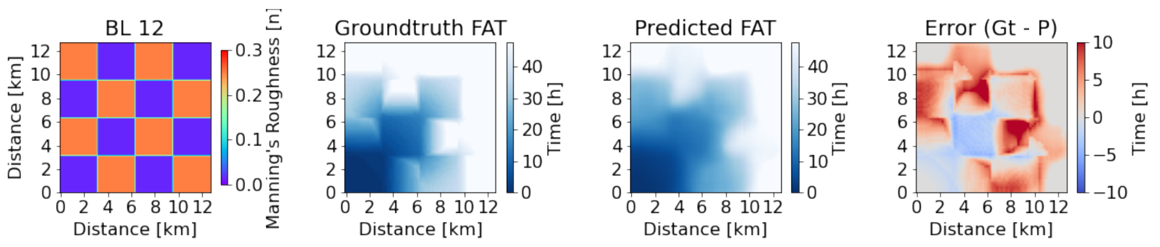


Figure 3.33: Prediction of a change in propogational speed in a strongly fluctuating land roughness pattern (BL12).

Figure 3.34 shows a similar pattern for BL13. Considering that this is one of the most complex landscapes presented to the DL model, it performs relatively well, especially for the initial phase of the simulation time. The process of filling up each pit before it can overflow to the next is predicted quite well. This is likely caused due to the presence of filling up depression after depression in the training dataset as well. This partly explains the good performance, albeit that the training data does not comprise landscapes with such levels of complexity.

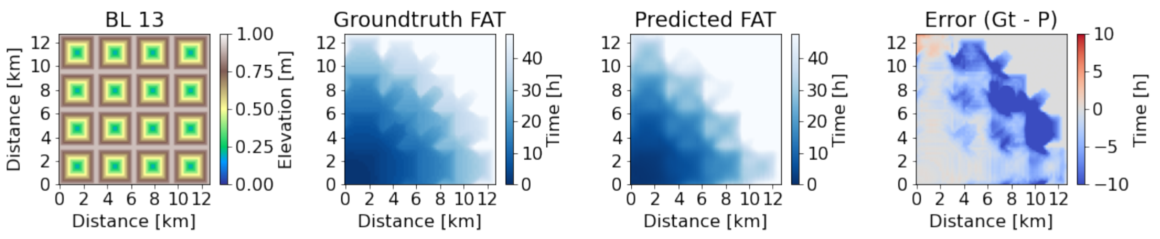


Figure 3.34: Prediction of filling up many pits (BL13).

BL14 takes the filling process into account as well as can be seen in Figure 3.35, although in this situation it is caused by WWs. Two interesting things become apparent from these results, the first being that the DL understands how to fill each reservoir in sequence, although it underestimates the flow. This causes the last reservoir not to fill in the prediction. Second, contrary to what could have been seen in e.g. BL9 and BL10, the DL model recognizes that water can be conveyed in a narrow strip from one reservoir to another, provided a WW is present. This is on the one hand as expected due to the explicit occurrence of WWs in the fourth dataset. On the other hand, the training dataset does not feature filling reservoirs interconnected by channels on this level of complexity.

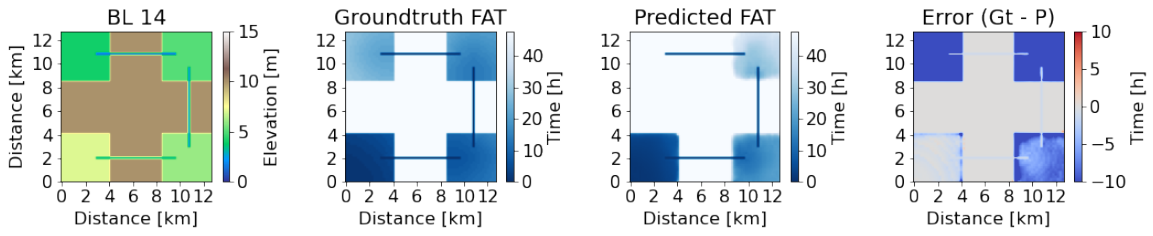


Figure 3.35: Prediction of filling up multiple reservoirs (BL14).

Lastly, Figure 3.36 shows the propagational speed of the flood in two different channels. The lower channel has an IWL in the channel, whereas the left channel has not. The groundtruth image shows, as expected, a darker color and thus lower FAT for the channel where an IWL is present, which can also be seen back in the prediction. Although slightly off, it predicts a lower FAT for this depression, and the difference between the actual and the predicted value is smaller compared to the channel where no water is present.

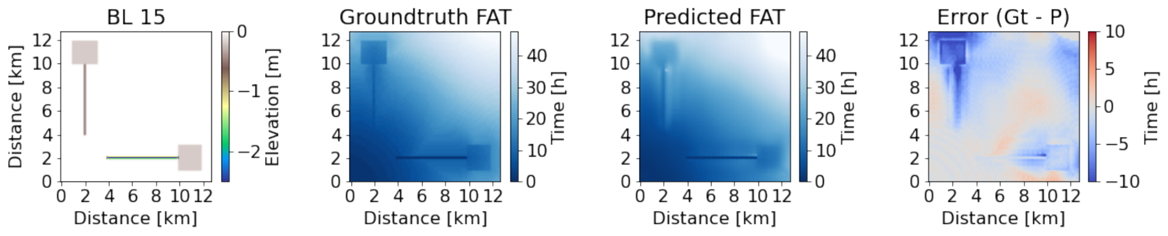


Figure 3.36: Prediction of a change in propagational speed when encountering an IWL compared to an empty channel (BL15).

## 3.7. Dike ring 48

The performances of the 4 DL models have been further verified on a realistic case study for dike ring 48. The results of each model are shown in Figure 3.37. The models have been applied in a similar fashion as described in the first section of this chapter.

The first observation is that the largest errors tend to be at the end of the simulation, which remains a recurring pattern. Second, the final shape of model three and four are comparable to the groundtruth, although there are some clear visual differences. The first model suffers the most in terms of results, as it has not been trained on EEs. This results in severe overprediction over a substantial portion of the domain. In addition, the shape of the dry areas within the flooded area have rounded shapes rather than sharp edges. This again has to do with the lack of EEs, which would result in the training data always in naturally varying heights and thus a lack of an abrupt FAT boundary.

Next, the second model predicts a substantial area which gets flooded at 4 kilometers on the right and 10 kilometers on top with respect to the origin of the flood. Besides, in the lower right part of the prediction of the second model, the flooding front stagnates at first which is represented by the dark part of the flooding, followed by a sudden but delayed continuation with a relatively wide white FAT values. However, this should not be the case, leading to a large error in the flooding front of a few pixels wide. Lastly, the second model predicts a couple kilometers above the origin of the flood flooding outside of the dike ring, which should not be the caseexcept the area which is connected with a WW.

The common factor between the second and third model is the overprediction in the area between the Y-shaped WWs. This is expected, since the water is not diverted to other parts of the WW first because the IWL input layer has been removed. Therefore, the model is not able to know that the water is propagating through and stored somewhere else in the channel, which results in an excessive amount of water within the Y-shaped area, leading to overprediction in this section. The difference between these two models is a smaller magnitude of error in the wrongly predicted outburst on distance [4, 10] and [0, 3]. Furthermore, the third model is better at predicting the location of the flooding front compared to the second model.

The fourth model also incorrectly predicts a patch of water 3 kilometers above the origin of the flood, and misses a considerable flooded area at a distance of 5 kilometers in both x and y direction from the flood's origin. However, in contrast to the second and third model, the fourth model is able to predict the FAT within the Y-shaped WW relatively well. The errors increase significantly when reaching the WW, which may be explained by the fact that the WW in dike ring 48 is of increased complexity compared to regular WWs. The training data consisted out of solely straight WWs (as well as EEs), while an irregular WW shape is present in the dike ring. The generalizability of the model is apparently not sufficient to take this into account.

Overall, the average error is 4.22, 2.61, 2.28, and 2.81 hours for model 1 to model 4, respectively. Thus, the third model performs best. This is not as expected, as WWs are present in the landscape, which can not be correctly accounted for in the prediction of the DL model. However, a slightly better result is obtained when taking the average prediction of the second, third and fourth model, as shown in the last row in Figure 3.37, which has an average hourly error of 2.23 hours. The first model is not taken into consideration in this average due to its poor performance.

Similar to the tests on BL3-5, all models predict poorly when water has to flow through a narrow opening. This is twice the case in dike ring 48, namely near [2, 3] and [5, 5], where there is a gap in the EE. The BL landscapes 3, 4, and 5 have already shown poor performance for these specific situations, which can be seen back in the performance in this realistic test scenario. It is therefore included in the recommendation section to include more training data which resembles this flow phenomenon.

Next, the irregular-shaped EEs and WWs may cause suboptimal performance, as the models have only been trained on straight ones. The DL model is not able to predict water conveyance in a non-straight WW well, resulting in an underestimated flooded area surrounding the WW, especially at the upper tip of the flooding. This is similar to BL14, where the flow through the WWs was underestimated which resulted in less flooding at t = 48. The DL model has only been trained on synthetic data and has never seen a landscape like this before, which casts a different perspective on the performance of the model. The performance of the dike ring is however lower than for the synthetic dataset, implying that real flooding examples may be of a different form of complexity than the synthetic data.

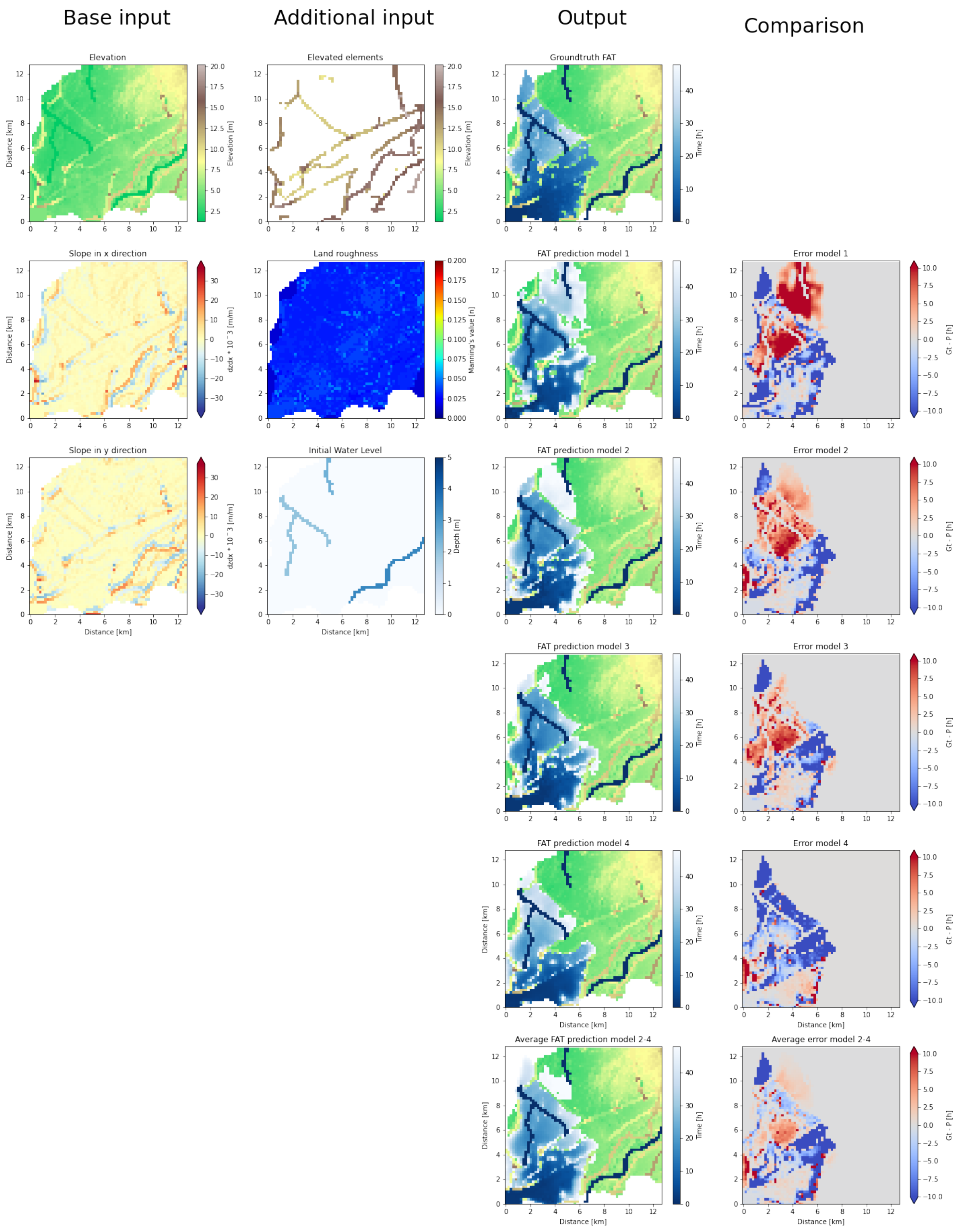


Figure 3.37: Dike ring 48 model performance comparison

# 4 Limitations & Recommendations

The performed research in this report explored a novel method to predict the FAT. There are numerous limitations and recommendations for this research, as the used methodology is still in its infancy. Current limitations pose on first instance restrictions on the applicability to apply this research to real life scenarios, although it also offers possibilities. There is still a field of interesting possibilities to explore, which is not only able to widen the scope of future research, but it can possibly enhance the performance of the model as well. The limitations with corresponding recommendations can be read below. First, the spatial and temporal resolution will be discussed, followed by a discussion regarding some flood characteristics. Lastly, some miscellaneous points are addressed.

## 4.1. Spatial and temporal resolution

The key motivation to use an exponent of two as domain size originates from a computer science point of view, where it is convenient to decrease and increase the image size by two when making use of an auto-encoder and decoder. As the numerical simulation time increases vastly with an increasing number of grid cells, a proper trade-off between computational time and area to cover had to be found. This led to a fixed number of grid cells within the domain, implying that flooding can not be predicted outside of this domain with the current set up. Changing the size of the domain is not possible, as the parameters within the DL model are trained on a set of input and output sizes. However, it is possible to train a model on a certain window, after which this window is repeatedly applied until the entire domain is covered. By doing so, one is not forced to stick with one domain size. It is key however to incorporate knowledge of the entire domain within the window, as flood propagation within such window may be influenced by flood propagation elsewhere in the domain. It is recommended to study this in further research so that the DL model is not bound to a fixed temporal and spatial scales.

In addition, the result section showed a decreased performance of the DL model for a strong gradient in FAT value. Especially regions where FAT values were skipped, i.e. where the flood development was low, suffered from poor performances. It is recommended to assess the overall DL model performance when a finer spatial resolution would be used, as this enables the observations of lower flood advancements, resulting in less missing FAT values and thus a possible better overall performance.

A relatively short simulation time was chosen for the numerical model as the primary aim of this study was to explore the possibilities to predict FATs during a calamity. It is expected that after the first 48 hours numerical models have finalized the computations for the further development of the flood and take over for a higher model precision. Current simulation time puts however a limit on the maximum flood forecast, and it would be interesting to study the performance of a DL model when the simulation time would be increased.

## 4.2. Applicability: flood characteristics

The first main limiting factor of applying the current DL model is the fixed flood location of (0, 0). This implies that the flood is only allowed to propagate in two perpendicular directions. The true north or south of the landscape is irrelevant, as the input data can be rotated such that the flood starts at the lower left corner. Regardless, it is not often evident prior to a numerical simulation whether a flood develops into solely two directions, which adds an additional source of uncertainty. It is furthermore a possible cause of the relatively poor performance when the flood is propagating towards the origin of the flood, as outlined in the results and discussion section. On a positive note, it is a relatively easy step to expand the current datasets by adding a varying starting location of the flood, and present this as an additional input layer to the DL model. A first feasibility check of this could be done without even expanding current dataset. When all input and their corresponding output layers would be randomly augmented by rotating the image, new flooding examples are created. The model's performance can therefore relatively easily be reassessed when flooding randomly starts at one of the four corners, provided it is given as an additional input layer. If the model performs well, it may be a first indication of proper understanding of a pure random starting point of the flood. More training data is required however to do so. It is strongly encouraged to explore the possibility of a random starting location as it would vastly increase the applicability of the model.

The second main limiting factor of applying the current DL model to realistic scenarios is the use of the set, fixed discharge. The applicability of the model would be increased if the discharge becomes flexible between the simulations. It would be even more the case if the inflow is adjustable intra-simulations. An ultimate goal could be to train a DL model to predict on forehand the discharge through e.g. a dike breach. A dike breach depends however on multiple variables which change during a flood, such as the upstream and downstream water level at the breach. It may be required to switch to a time series forecasting approach, where these variables can be predicted per timestep by a(nother) DL model, which can serve as an input for the DL model which predicts the flood propagation. The output of the latter model can then serve again as input for the former model. A possible suitable architecture could incorporate an LSTM for doing this. New training data with varying discharges should be created for doing so, which is strongly recommended for future research.

Lastly, the FAT is a relatively shallow source of information in terms of flooding. Water depths and flow velocities are of huge relevance as well when decisions need to be made on which regions need to be evacuated, which are not taken into consideration in the model. Moreover, relevant physical equations, such as the momentum and continuity balance, can not properly be checked with the current set up. This implies that the current model can not be penalized when e.g. the water balance does not close. Water depth data is available however, so future researchers are encouraged to explore the full potential of the dataset in further research.

## 4.3. Dataset quality

A synthetic dataset was generated to acquire the vast number of data samples which are required to train the DL model. Although the hydrodynamic processes remain the same, real life landscapes look differently than the synthetic dataset as could be seen in the case study area of dike ring 48. To add on top of that, waterworks such as pumps, water retaining structures which fail after a certain water depth, undershots, or 1D2D links were not included in the synthetic data, and can thus not be modelled. It is therefore recommended to perform a follow-up research in the future which does include such elements, as well as non-straight EEs and WWs which are narrower than 200 meters.

Furthermore, Delft3D FM was considered as being the groundtruth, and was used to train the DL model. Possible shortcomings or limitations within the numerical model are therefore present in the training data, and thus in the predictions as well. It would therefore be interesting to compare in future research the performance of the model to an actual flooding example.

## 4.4. Additional recommendations

As briefly touched upon in the section above, there is a vast source of information hidden in the used data. Each sample in this study retrieved the FAT of the water per pixel from 48 timesteps. It is therefore strongly recommended to train a DL model to predict water depth per timestep, which can either be done by creating a new dataset with the above mentioned recommendations, or using the same dataset which was used for this research.

Furthermore, an absolute DEM map was used as an additional source of input as this slightly increased the model's performance. It is however recommended to leave this absolute valued input layer out when training the DL model on landscapes which are persistently increasing in one direction, such as mountainous regions. As the slope value of such region may fall within the training dataset, the difference between the minimum and maximum DEM value may not if there is a persistent increase in elevation into a certain direction, leading to possible poor performances. It is thus recommended to give up a small percentage of the overall performance for the sake of generalizability.

Lastly, the objective of this research was to predict the flood arrival time with a low error. This means that, as could be seen in the result section, errors occur on both the wet as well as the dry pixels. This may not be desirable, depending on the use case. Over-caution in the flooded region is desired when people need to be evacuated, resulting in a higher wish for a model which puts more emphasis on predicting the wet pixels correctly. On the contrary, when the location of an emergency field hospital needs to be determined, there should be no risk of flooding, resulting in a trained model which focuses on the dry cells. It would therefore be interesting for further research or application purposes to use custom loss functions for the same model, to be able to predict flooding for multiple users.

# 5 Conclusion

The main conclusions of this research will be drawn by answering each sub research question, which are listed underneath. This is subsequently followed by final remarks.

1. **What is the overall accuracy of a deep learning model which can predict fluvial flood arrival times for different synthetic datasets?**
   The used deep learning model was a convolutional neural network with an encoder-decoder architecture with 5 parallel branches, where each of them went into a different level of abstraction. Four models with this architecture were trained, each on a unique dataset. The complexity of a sample increases per dataset, where the elevation, slopes, elevated elements (EEs), land roughness, and initial water levels were taken into consideration, depending on the dataset. The error of the prediction on the flood arrival time (FAT) of each model on the corresponding test dataset was equal to 0.91, 1.41, 1.25 and 1.76 hours, respectively.

2. **How is the model performing when splitting the data samples into different categories?**
   As could be seen in the answer of previous research question, a more complex landscape is on average harder to predict than uncomplicated ones. This can be seen back in the different performances of flat and hilly landscapes, of which the former sees lower errors than the latter. Furthermore, an increased number of EEs and WWs decrease the model performance, as well as areas with a high gradient in FAT value.

   Furthermore, there is a tendency for a slight underprediction until around the first 35 hours of the flood, which is in terms of loss function compensated for a higher degree of over prediction in the remaining time of the simulation. The model is therefore more reliable for this first phase of the flooding than for the latter part. The exception is the fourth dataset, which sees a consistent underprediction due to post processing steps, which have decreased the final error significantly. The predictions of the first three models have an average F1-score of over 0.95, while the prediction of the last model has an average F1-score of 0.84.

3. **How is the model performing when applying it to specific theoretical flow conditions?**
   The largest errors occur when water has to pass through a narrow opening. The degree of error is limited when a WW is present in the opening, indicating that this type of errors could be decreased when adding this explicitly to the training dataset. Furthermore, the DL model faces some difficulties in estimating the correct FAT after an overflowable EE. Non-overflowable EEs are sometimes still flooded, resulting in FAT values in the hinterland which should remain dry. However, this occurs sporadically, and the results of the BLs indicate a satisfactory understanding of highly dynamic landscapes in terms of varying land roughness and elevation.

4. **How is the model performing when applying it to a real life example?**
   The Dutch dike ring 48 has been chosen to answer this research question. The final average error for model one to four was equal to 4.22, 2.61, 2.28, and 2.81 hours per cell, respectively. The largest errors occurred in the vicinity of the irregular-shaped waterway, which is as expected

as the training data consisted solely of WWs in the two orthogonal directions. The area behind the small gap in EE also suffered of relatively low performance. The final shape of all three simulations are however similar and comparable to the final output of the groundtruth, implying that a reasonable first approximation of the FAT can be made with the created model.

This leads to the final overall conclusion, which is that the surrogate model is able to create a first approximation of the flood, especially in the first 24 hours on landscapes which are rather flat without too many complexities. It is recommended to expand the current dataset with additional training data which consists of undershots, narrow flow paths, non-straight WWs, random flood location within the domain, variable discharges, and active dike breaches, for which for the latter two the exploration of an algorithm which has temporal awareness is recommended, such as an LSTM. This research has shown the potential of DL in the field of water management, and its applicability to quickly get a first estimation of the FAT in case of a source point flooding.

# Bibliography

Alves, B. (2022), ‘Statista’.
**URL:** *https://www-statista-com.tudelft.idm.oclc.org/statistics/1293207/global-number-of-deaths-due-to-flood/*

Arends, M. (2014), ‘Veiligheid nederland in kaart 2 : overstromingsrisico dijkringgebied 48, rijn en ijssel’.
**URL:** *https://puc.overheid.nl/rijkswaterstaat/doc/PUC_148299_31/*

Bank, D., Koenigstein, N. & Giryes, R. (2020), ‘Autoencoders’, *arXiv preprint arXiv:2003.05991* .

Bengio, Y. (2009), *Learning deep architectures for AI*, Now Publishers Inc.

Bengio, Y., Simard, P. & Frasconi, P. (1994), ‘Learning long-term dependencies with gradient descent is difficult’, *IEEE transactions on neural networks* **5**(2), 157–166.

Bilbao, I. & Bilbao, J. (2017), Overfitting problem and the over-training in the era of data: Particularly for artificial neural networks, *in* ‘2017 eighth international conference on intelligent computing and information systems (ICICIS)’, IEEE, pp. 173–177.

Chang, L.-C., Shen, H.-Y. & Chang, F.-J. (2014), ‘Regional flood inundation nowcast using hybrid som and dynamic neural networks’, *Journal of Hydrology* **519**, 476–489.

Chen, Y., Jiang, H., Li, C., Jia, X. & Ghamisi, P. (2016), ‘Deep feature extraction and classification of hyperspectral images based on convolutional neural networks’, *IEEE Transactions on Geoscience and Remote Sensing* **54**(10), 6232–6251.

Cho, K., Van Merriënboer, B., Bahdanau, D. & Bengio, Y. (2014), ‘On the properties of neural machine translation: Encoder-decoder approaches’, *arXiv preprint arXiv:1409.1259* .

Choubin, B., Moradi, E., Golshan, M., Adamowski, J., Sajedi-Hosseini, F. & Mosavi, A. (2019), ‘An ensemble prediction of flood susceptibility using multivariate discriminant analysis, classification and regression trees, and support vector machines’, *Science of the Total Environment* **651**, 2087–2096.

Collins, M., Knutti, R., Arblaster, J., Dufresne, J.-L., Fichefet, T., Friedlingstein, P., Gao, X., Gutowski, W. J., Johns, T., Krinner, G. et al. (2013), Long-term climate change: projections, commitments and irreversibility, *in* ‘Climate Change 2013-The Physical Science Basis: Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change’, Cambridge University Press, pp. 1029–1136.

Da Silva, I. N., Spatti, D. H., Flauzino, R. A., Liboni, L. H. B. & dos Reis Alves, S. F. (2017), ‘Artificial neural networks’, *Cham: Springer International Publishing* **39**.

De Bruijn, K., slager, k., Piek, R., Riedstra, D. & Slomp, R. (2018), *Leidraad voor het maken van overstromingssimulaties (in Dutch) Guidance document for modelling floods*.

*Deltares* (2022).
**URL:** *https://oss.deltares.nl/web/delft3dfm/manuals*

DWW, R., Grontmij & Bos, W. . (2005), ‘Veiligheid nederland in kaart, overstromingsrisico dijkring 48 rijn en ijssel’.

Feurer, M. & Hutter, F. (2019), Hyperparameter optimization, *in* ‘Automated machine learning’, Springer, Cham, pp. 3–33.

Gamboa, J. C. B. (2017), ‘Deep learning for time-series analysis’, *arXiv preprint arXiv:1701.01887* .

Glorot, X. & Bengio, Y. (2010), Understanding the difficulty of training deep feedforward neural networks, *in* 'Proceedings of the thirteenth international conference on artificial intelligence and statistics', JMLR Workshop and Conference Proceedings, pp. 249–256.

Goodfellow, I., Bengio, Y. & Courville, A. (2016), *Deep learning*, MIT press.

Guan, M., Wright, N. G. & Sleigh, P. A. (2014), '2d process-based morphodynamic model for flooding by noncohesive dyke breach', *Journal of Hydraulic Engineering* **140**(7), 04014022.

Guo, Z., Leitao, J. P., Simões, N. E. & Moosavi, V. (2021), 'Data-driven flood emulation: Speeding up urban flood predictions by deep convolutional neural networks', *Journal of Flood Risk Management* **14**(1), e12684.

Guo, Z., Moosavi, V. & Leitão, J. P. (2022), 'Data-driven rapid flood prediction mapping with catchment generalizability', *Journal of Hydrology* **609**, 127726.

Hirabayashi, Y., Mahendran, R., Koirala, S., Konoshima, L., Yamazaki, D., Watanabe, S., Kim, H. & Kanae, S. (2013), 'Global flood risk under climate change', *Nature climate change* **3**(9), 816–821.

Jacobs, R. A. (1988), 'Increased rates of convergence through learning rate adaptation', *Neural networks* **1**(4), 295–307.

Jhong, Y.-D., Chen, C.-S., Lin, H.-P. & Chen, S.-T. (2018), 'Physical hybrid neural network model to forecast typhoon floods', *Water* **10**(5), 632.

Jongman, B., Ward, P. J. & Aerts, J. C. (2012), 'Global exposure to river and coastal flooding: Long term trends and changes', *Global Environmental Change* **22**(4), 823–835.

Kabir, S., Patidar, S., Xia, X., Liang, Q., Neal, J. & Pender, G. (2020), 'A deep convolutional neural network model for rapid prediction of fluvial flood inundation', *Journal of Hydrology* **590**, 125481.

Klambauer, G., Unterthiner, T., Mayr, A. & Hochreiter, S. (2017), 'Self-normalizing neural networks', *Advances in neural information processing systems* **30**.

Kratzert, F., Klotz, D., Brenner, C., Schulz, K. & Herrnegger, M. (2018), 'Rainfall–runoff modelling using long short-term memory (lstm) networks', *Hydrology and Earth System Sciences* **22**(11), 6005–6022.

Kratzert, F., Klotz, D., Herrnegger, M., Sampson, A. K., Hochreiter, S. & Nearing, G. S. (2019), 'Toward improved predictions in ungauged basins: Exploiting the power of machine learning', *Water Resources Research* **55**(12), 11344–11354.

Kuijpers, Mols & Schouten (2005), 'Veiligheid nederland in kaart : overstromingsrisico dijkring 48 rijn en ijssel'.
**URL:** *https://puc.overheid.nl/rijkswaterstaat/doc/PUC_125892_31*

LeCun, Y. A., Bottou, L., Orr, G. B. & Müller, K.-R. (2012), Efficient backprop, *in* 'Neural networks: Tricks of the trade', Springer, pp. 9–48.

LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. (1998), 'Gradient-based learning applied to document recognition', *Proceedings of the IEEE* **86**(11), 2278–2324.

Lhomme, J., Sayers, P., Gouldby, B., Samuels, P., Wills, M. & Mulet-Marti, J. (2008), 'Recent development and application of a rapid flood spreading method'.

Li, J., Cheng, J.-h., Shi, J.-y. & Huang, F. (2012), Brief introduction of back propagation (bp) neural network algorithm and its improvement, *in* 'Advances in computer science and information engineering', Springer, pp. 553–558.

Liu, G., Shih, K. J., Wang, T.-C., Reda, F. A., Sapra, K., Yu, Z., Tao, A. & Catanzaro, B. (2018), 'Partial convolution based padding', *arXiv preprint arXiv:1811.11718* .

Löwe, R., Böhm, J., Jensen, D. G., Leandro, J. & Rasmussen, S. H. (2021), 'U-flood–topographic deep learning for predicting urban pluvial flood water depth', *Journal of Hydrology* **603**, 126898.

Merz, B., Kreibich, H. & Lall, U. (2013), ‘Multi-variate flood damage assessment: a tree-based data-mining approach’, *Natural Hazards and Earth System Sciences* **13**(1), 53–64.

Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R. & Muharemagic, E. (2015), ‘Deep learning applications and challenges in big data analytics’, *Journal of big data* **2**(1), 1–21.

Nearing, G. S. & Gupta, H. V. (2015), ‘The quantity and quality of information in hydrologic models’, *Water Resources Research* **51**(1), 524–538.

Paprotny, D., Kreibich, H., Morales-Nápoles, O., Wagenaar, D., Castellarin, A., Carisi, F., Bertin, X., Merz, B. & Schröter, K. (2021), ‘A probabilistic approach to estimating residential losses from different flood types’, *Natural Hazards* **105**(3), 2569–2601.

Poggio, T., Mhaskar, H., Rosasco, L., Miranda, B. & Liao, Q. (2017), ‘Why and when can deep-but not shallow-networks avoid the curse of dimensionality: a review’, *International Journal of Automation and Computing* **14**(5), 503–519.

Ronneberger, O., Fischer, P. & Brox, T. (2015), U-net: Convolutional networks for biomedical image segmentation, *in* ‘International Conference on Medical image computing and computer-assisted intervention’, Springer, pp. 234–241.

Rudy, S. & Sapsis, T. (2021), ‘Output-weighted and relative entropy loss functions for deep learning precursors of extreme events’, *arXiv preprint arXiv:2112.00825* .

Schröter, K., Kreibich, H., Vogel, K., Riggelsen, C., Scherbaum, F. & Merz, B. (2014), ‘How useful are complex flood damage models?’, *Water Resources Research* **50**(4), 3378–3395.

Sharma, S., Sharma, S. & Athaiya, A. (2017), ‘Activation functions in neural networks’, *towards data science* **6**(12), 310–316.

Soleymani, S., Dabouei, A., Kazemi, H., Dawson, J. & Nasrabadi, N. M. (2018), Multi-level feature abstraction from convolutional neural networks for multimodal biometric identification, *in* ‘2018 24th International Conference on Pattern Recognition (ICPR)’, IEEE, pp. 3469–3476.

Solomatine, D., See, L. M. & Abrahart, R. (2009), ‘Data-driven modelling: concepts, approaches and experiences’, *Practical hydroinformatics* pp. 17–30.

Spinewine, B., Delobbe, A., Elslander, L. & Zech, Y. (2004), Experimental investigation of the breach growth process in sand dikes, *in* ‘Second IAHR international conference on fluvial hydraulics, Napoli, Italy’, pp. 983–993.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. (2014), ‘Dropout: a simple way to prevent neural networks from overfitting’, *The journal of machine learning research* **15**(1), 1929–1958.

Teng, J., Jakeman, A. J., Vaze, J., Croke, B. F., Dutta, D. & Kim, S. (2017), ‘Flood inundation modelling: A review of methods, recent advances and uncertainty analysis’, *Environmental modelling & software* **90**, 201–216.

Tompson, J., Schlachter, K., Sprechmann, P. & Perlin, K. (2017), Accelerating eulerian fluid simulation with convolutional networks, *in* ‘International Conference on Machine Learning’, PMLR, pp. 3424–3433.

Wang, Y., Fang, Z., Hong, H. & Peng, L. (2020), ‘Flood susceptibility mapping using convolutional neural network frameworks’, *Journal of Hydrology* **582**, 124482.

Weiler, M., Hamprecht, F. A. & Storath, M. (2018), Learning steerable filters for rotation equivariant cnns, *in* ‘Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition’, pp. 849–858.

Willner, S. N., Levermann, A., Zhao, F. & Frieler, K. (2018), ‘Adaptation required to preserve future high-end river flood risk at present levels’, *Science advances* **4**(1), eaao1914.

Yu, Y., Si, X., Hu, C. & Zhang, J. (2019), ‘A review of recurrent neural networks: Lstm cells and network architectures’, *Neural computation* **31**(7), 1235–1270.

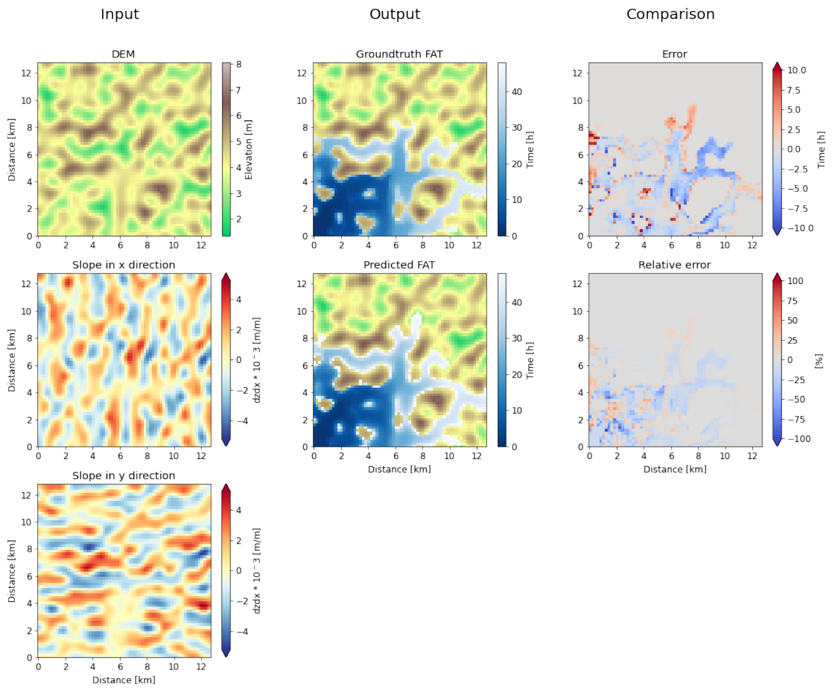
# A
# Recurrent Neural Networks

Timeseries are of great importance when dealing with processes such as rainfall-runoff modelling, drainage, melt water flow, and flooding. A simple way to integrate loops, is to use a hidden state which incorporates a form of memory of past timesteps as its own input for the next timestep (Gamboa 2017). This is referred to as a Recurrent Neural Network (RNN). One of the major disadvantages of a RNN are the vanishing and exploding gradients (Bengio et al. 1994). RNNs have a limited capability of memorizing information of more than a few time steps back. There was therefore a need for a type of network which could solve these problems, after which the Long Short-Term Memory (LSTM) was invented. An LSTM is in essence an more sophisticated RNN, where each cell includes a memory cell which can store information for a longer period of time. A set of gates is used to control when information enters the memory, when its output, and when it is forgotten (Yu et al. 2019). These gates are composed out of a different activation functions, which can all be tuned via training. Although LSTMs have great potential in forecasting time series, they have limited spatial awareness which is a crucial component for 2D or higher image prediction.

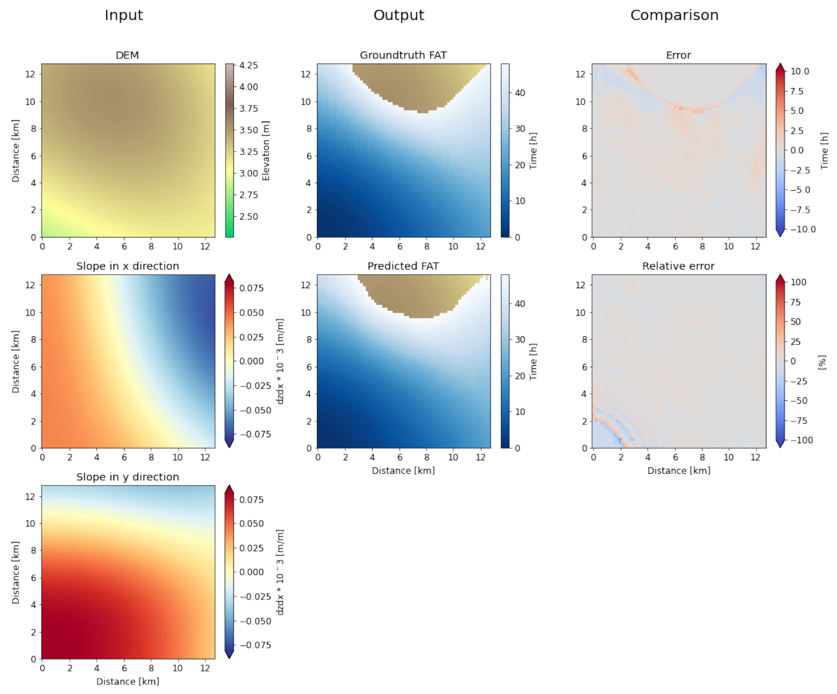
# B

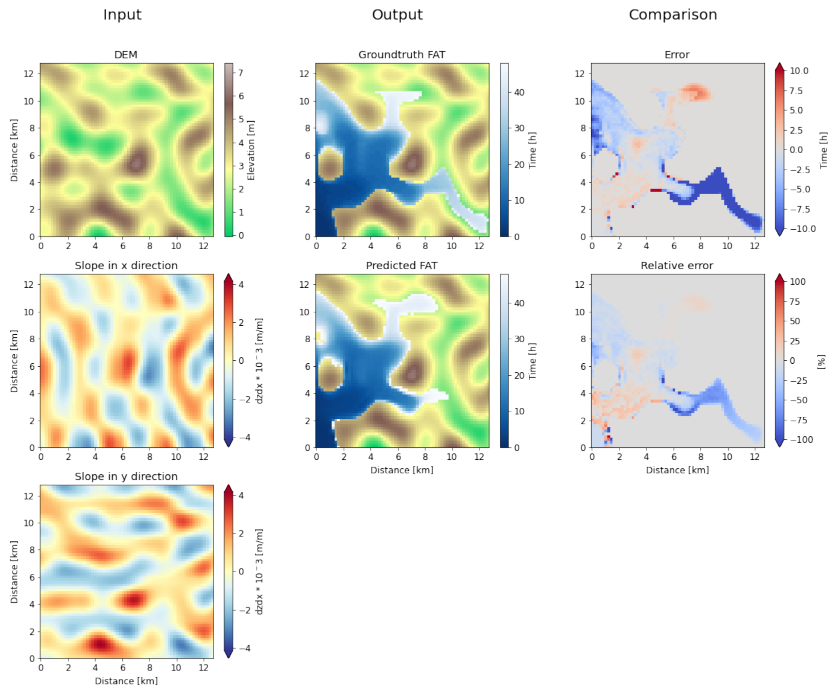
# Dataset examples

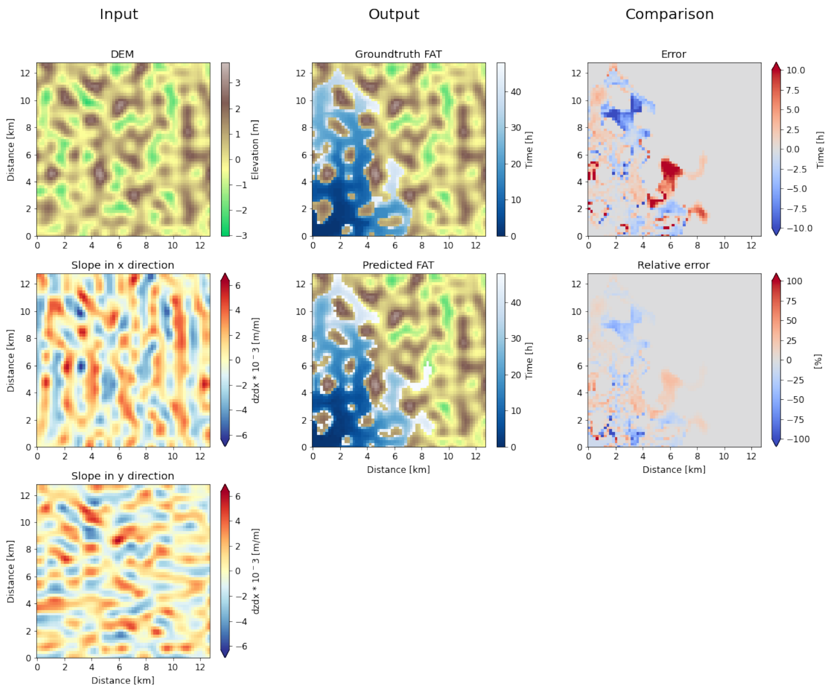
Some additional test dataset examples are shown underneath. The first six test samples are shown for each dataset, in the same type of fashion as showed in the result section.

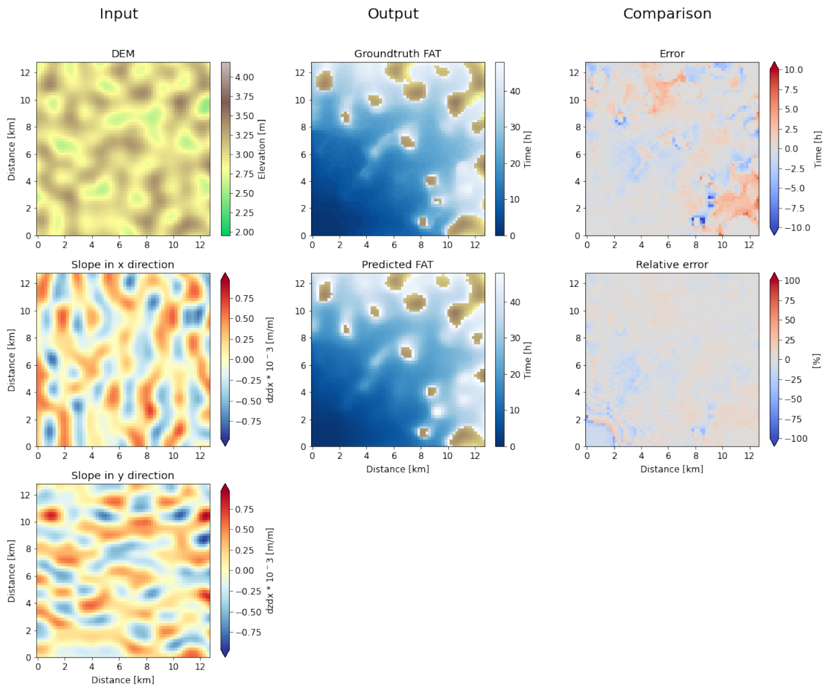


## B.1. Dataset 1


Input
Output
Comparison
DEM
Groundtruth FAT
Error
Slope in x direction
Predicted FAT
Relative error
Slope in y direction
Distance [km]
Elevation [m]
Time [h]
dzdx * 10^-3 [m/m]
[%]


Input
Output
Comparison
DEM
Groundtruth FAT
Error
Slope in x direction
Predicted FAT
Relative error
Slope in y direction
Distance [km]
Elevation [m]
Time [h]
dzdx * 10^-3 [m/m]
[%]


Input
Output
Comparison
DEM
Groundtruth FAT
Error
Slope in x direction
Predicted FAT
Relative error
Slope in y direction
Distance [km]
Elevation [m]
Time [h]
dzdx * 10^-3 [m/m]
[%]


Input
Output
Comparison
DEM
Groundtruth FAT
Error
Slope in x direction
Predicted FAT
Relative error
Slope in y direction
Distance [km]
Elevation [m]
Time [h]
dzdx * 10^-3 [m/m]
[%]

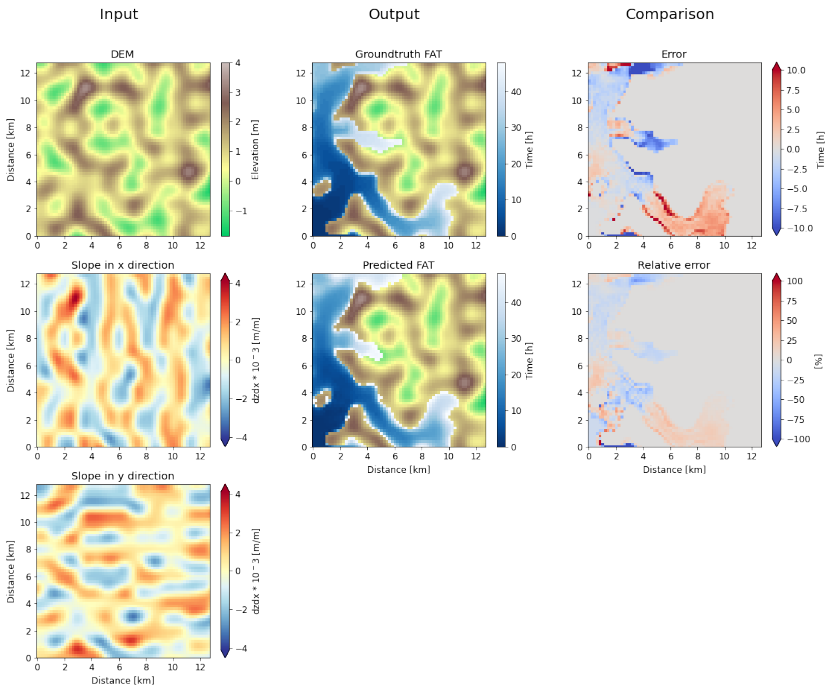
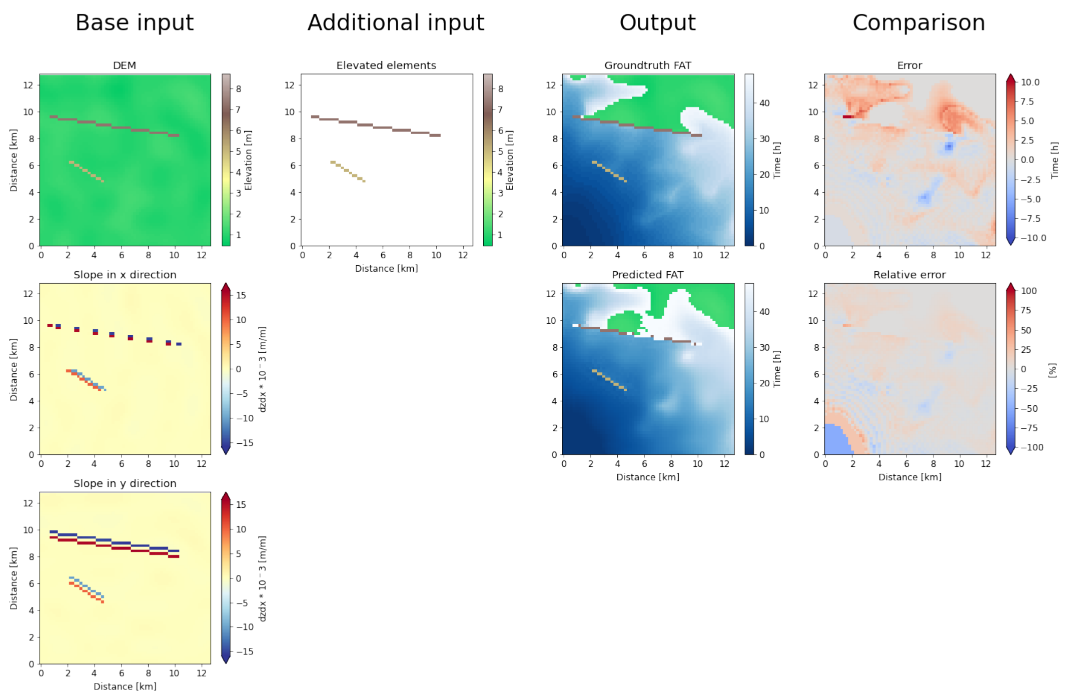
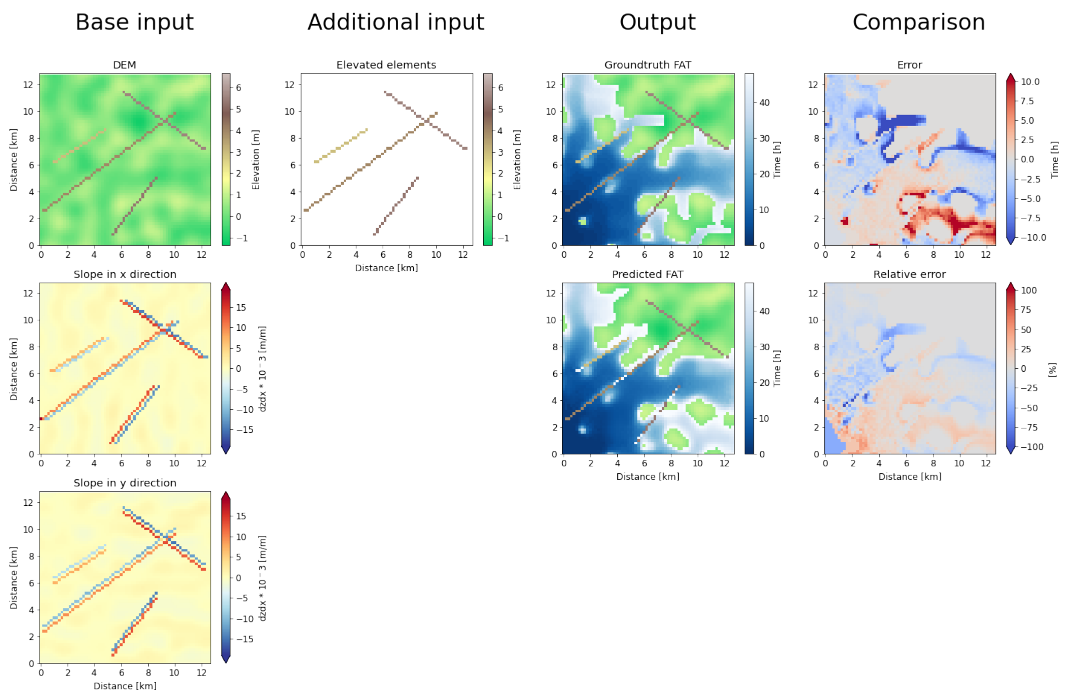
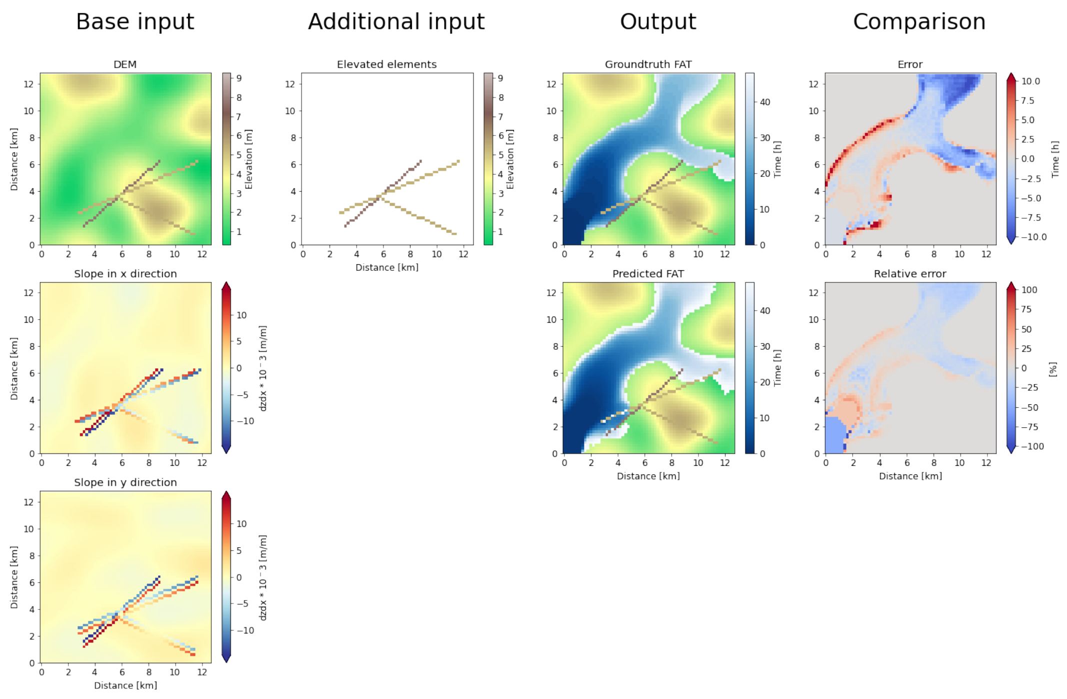


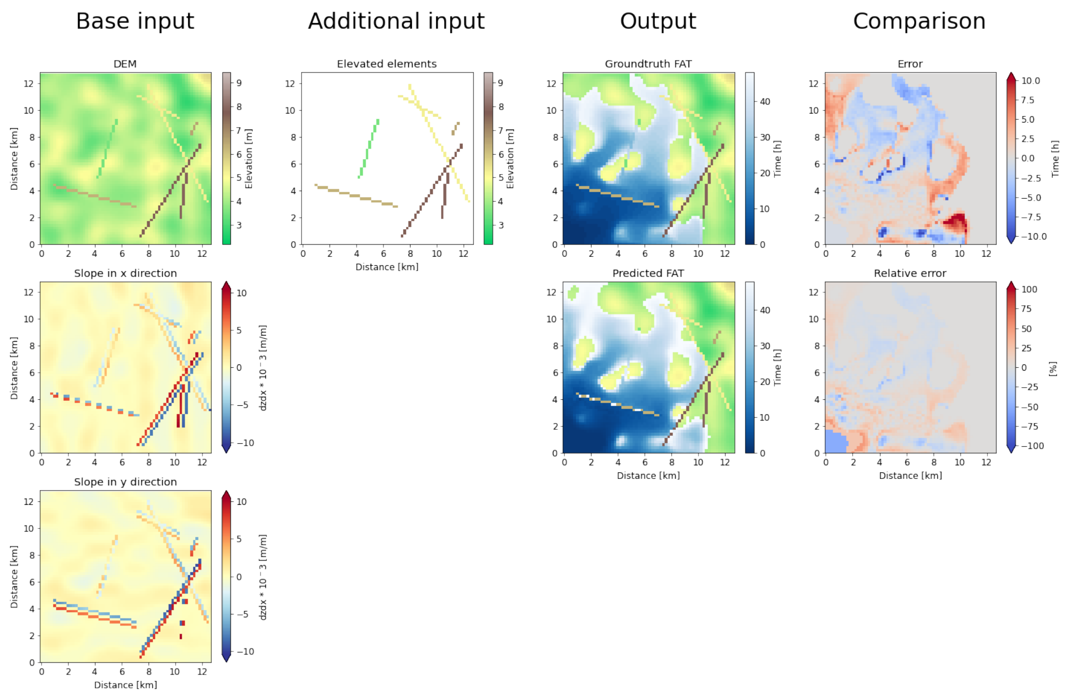
Figure B.1: First six flooding examples of dataset 1.

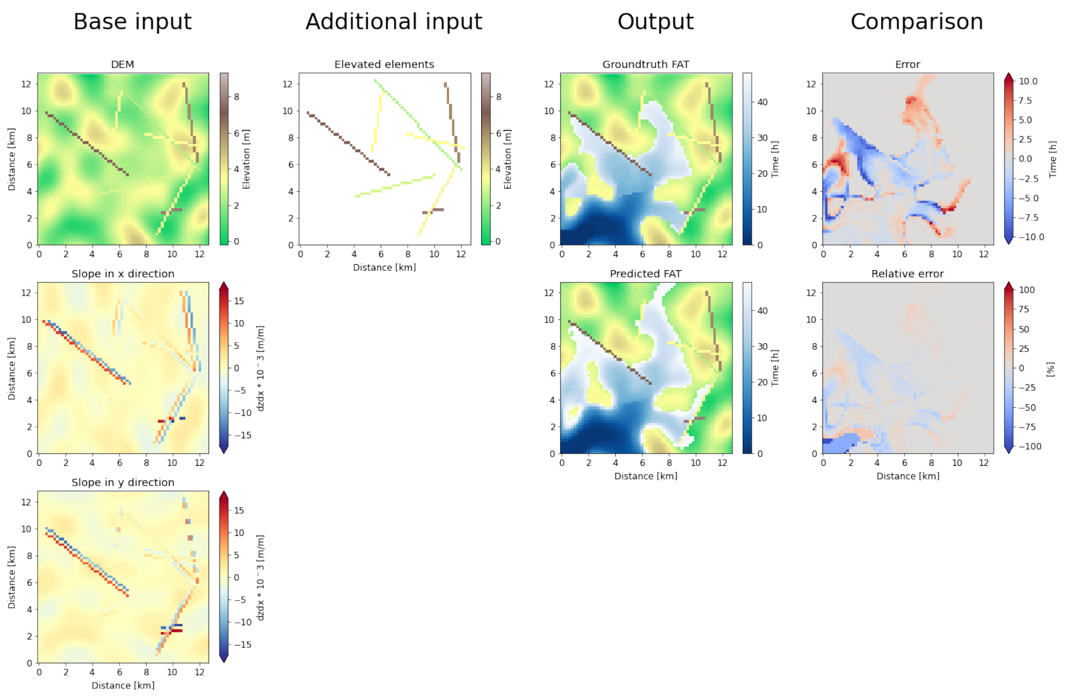
## B.2. Dataset 2


Base input
Additional input
Output
Comparison
DEM
Elevation [m]
Elevated elements
Elevation [m]
Groundtruth FAT
Time [h]
Error
Time [h]
Slope in x direction
dzdx * 10⁻3 [m/m]
Predicted FAT
Time [h]
Relative error
[%]
Slope in y direction
dzdx * 10⁻3 [m/m]
Distance [km]
Distance [km]


Base input
Additional input
Output
Comparison
DEM
Elevation [m]
Elevated elements
Elevation [m]
Groundtruth FAT
Time [h]
Error
Time [h]
Slope in x direction
dzdx * 10⁻3 [m/m]
Predicted FAT
Time [h]
Relative error
[%]
Slope in y direction
dzdx * 10⁻3 [m/m]
Distance [km]
Distance [km]


Base input
Additional input
Output
Comparison
DEM
Elevated elements
Groundtruth FAT
Error
Slope in x direction
Predicted FAT
Relative error
Slope in y direction
Distance [km]
Elevation [m]
Time [h]
dzdx * 10^-3 [m/m]
[%]


Base input
Additional input
Output
Comparison
DEM
Elevated elements
Groundtruth FAT
Error
Slope in x direction
Predicted FAT
Relative error
Slope in y direction
Distance [km]
Elevation [m]
Time [h]
dzdx * 10^-3 [m/m]
[%]



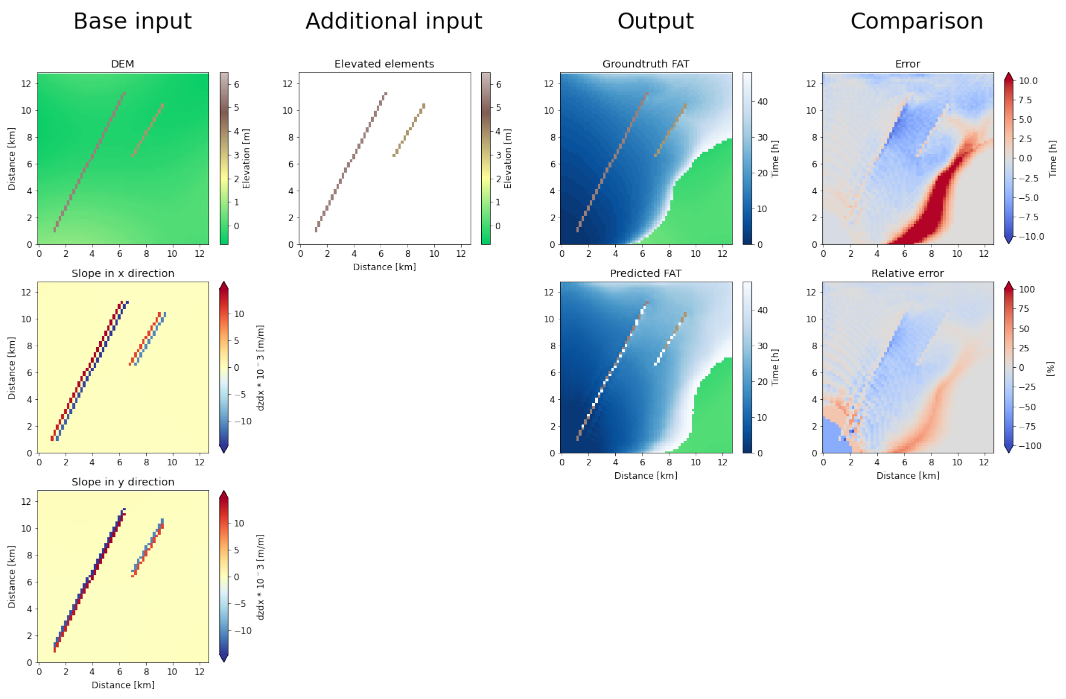
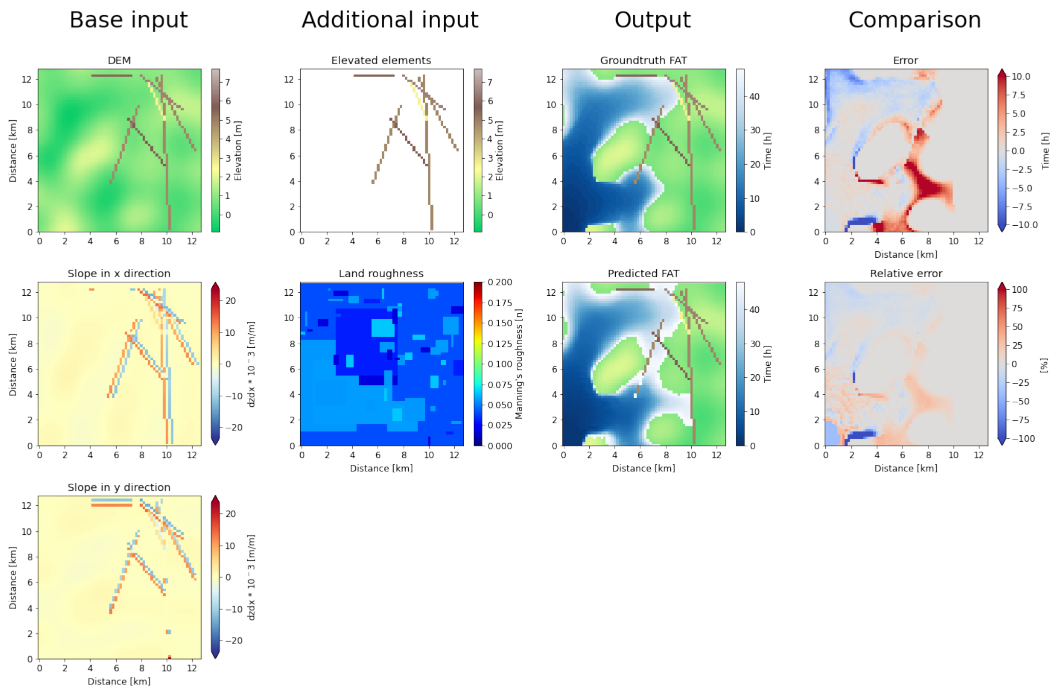
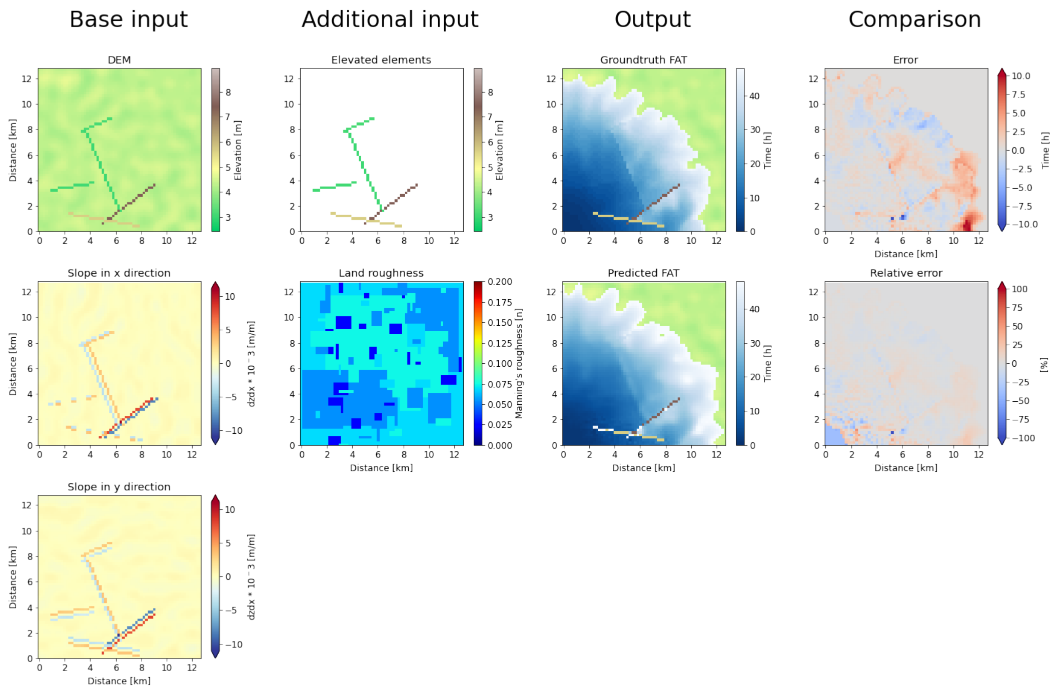
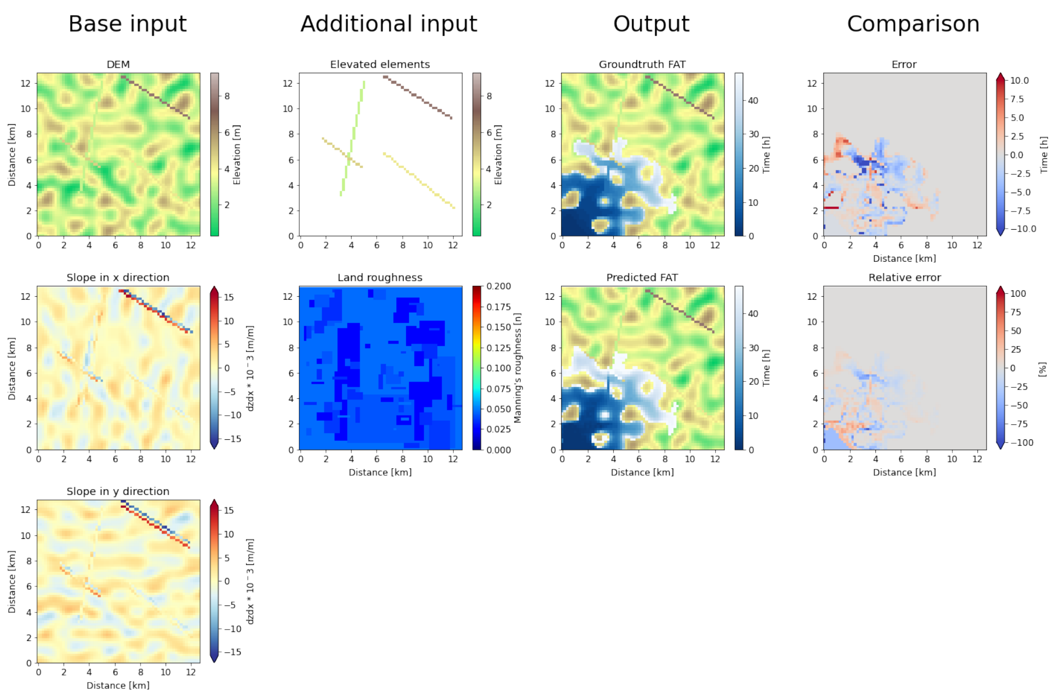


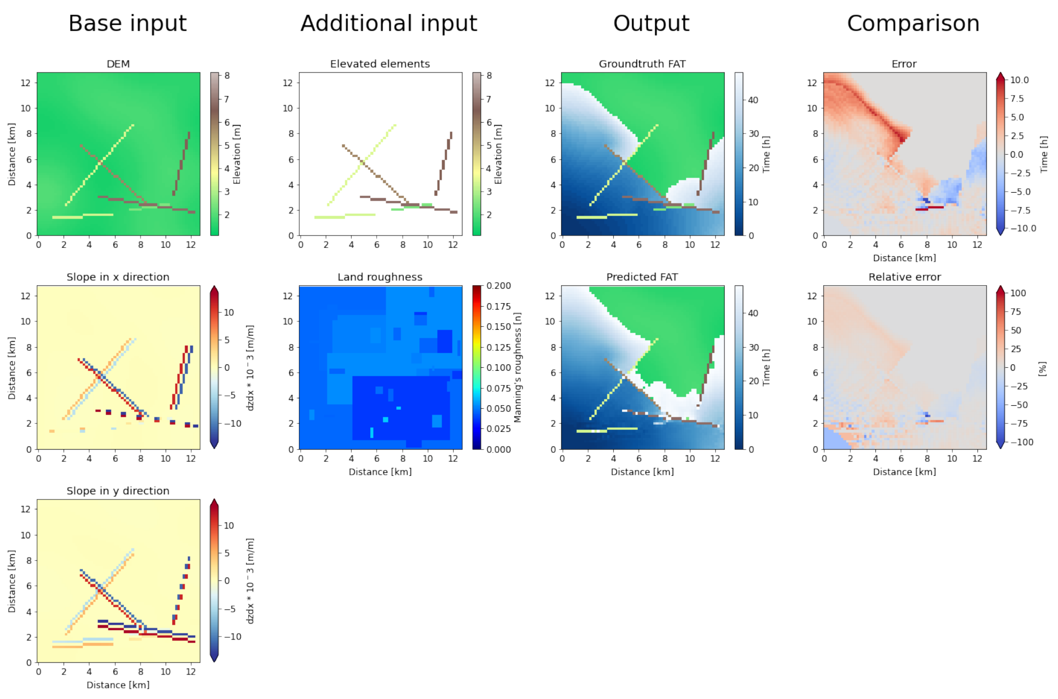
Figure B.2: First six flooding examples of dataset 2.

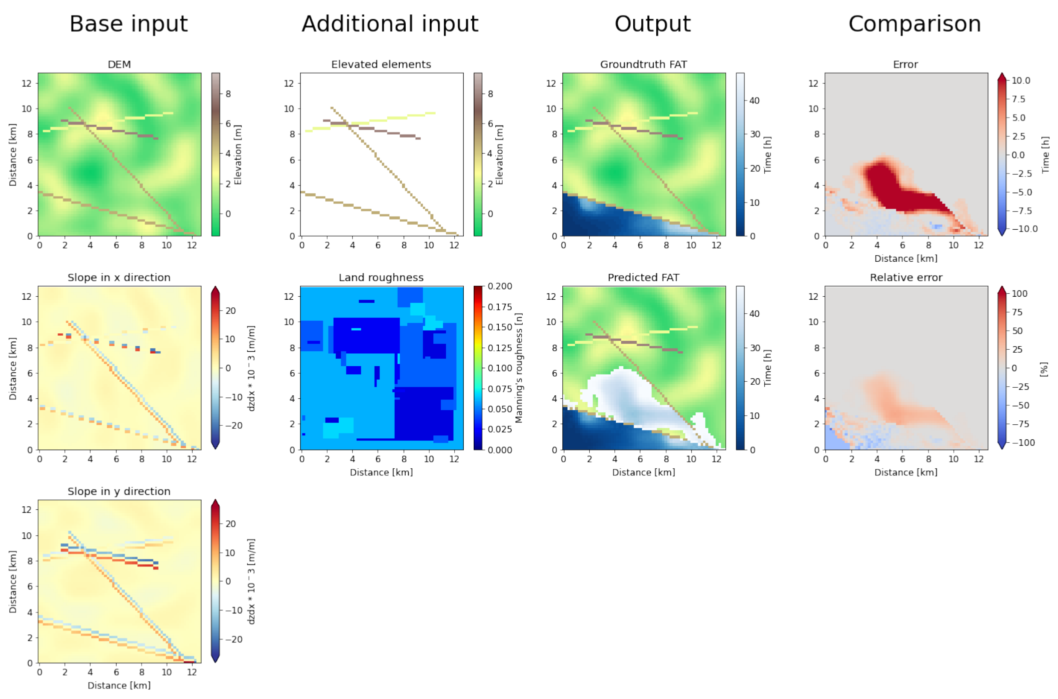
## B.3. Dataset 3


Base input
Additional input
Output
Comparison
DEM
Elevated elements
Groundtruth FAT
Error
Slope in x direction
Land roughness
Predicted FAT
Relative error
Slope in y direction
Distance [km]
Elevation [m]
Time [h]
dzdx * 10⁻3 [m/m]
Manning's roughness [n]
[%]


Base input
Additional input
Output
Comparison
DEM
Elevated elements
Groundtruth FAT
Error
Slope in x direction
Land roughness
Predicted FAT
Relative error
Slope in y direction
Distance [km]
Elevation [m]
Time [h]
dzdx * 10⁻3 [m/m]
Manning's roughness [n]
[%]


Base input
Additional input
Output
Comparison
DEM
Elevated elements
Groundtruth FAT
Error
Slope in x direction
Land roughness
Predicted FAT
Relative error
Slope in y direction
Distance [km]
Elevation [m]
Time [h]
dzdx * 10^-3 [m/m]
Manning's roughness [n]
[%]


Base input
Additional input
Output
Comparison
DEM
Elevated elements
Groundtruth FAT
Error
Slope in x direction
Land roughness
Predicted FAT
Relative error
Slope in y direction
Distance [km]
Elevation [m]
Time [h]
dzdx * 10^-3 [m/m]
Manning's roughness [n]
[%]



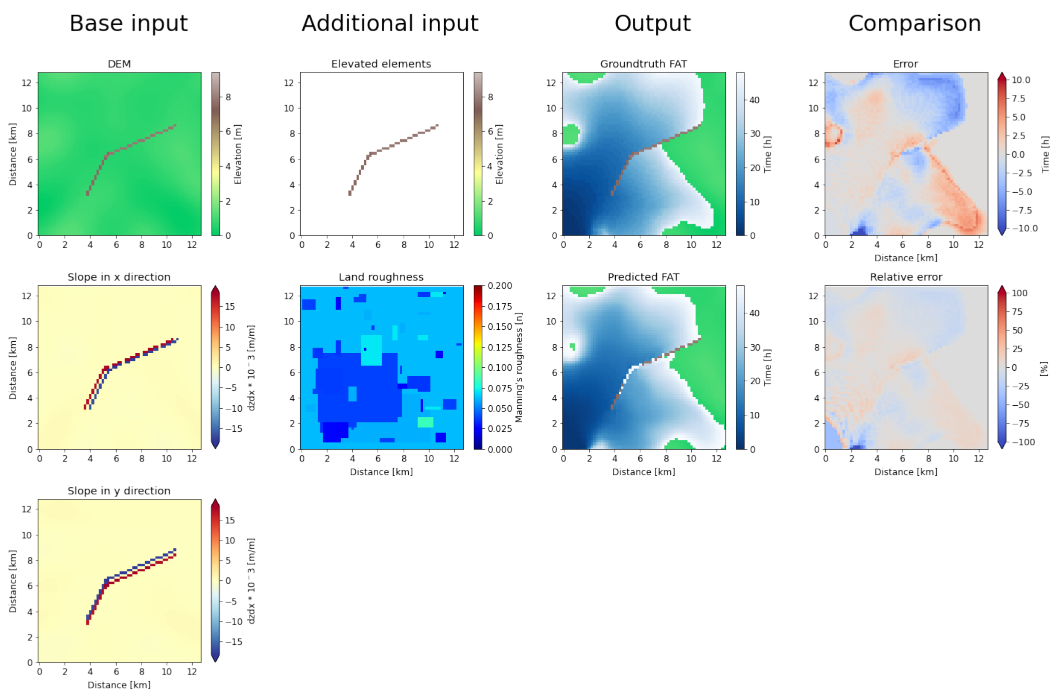
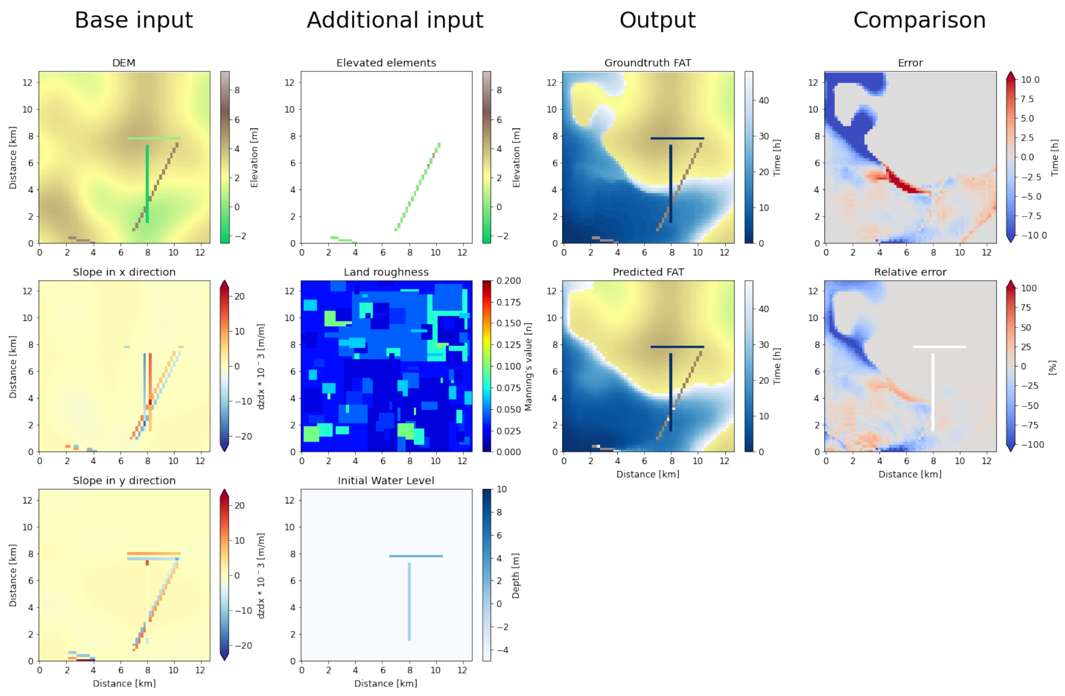
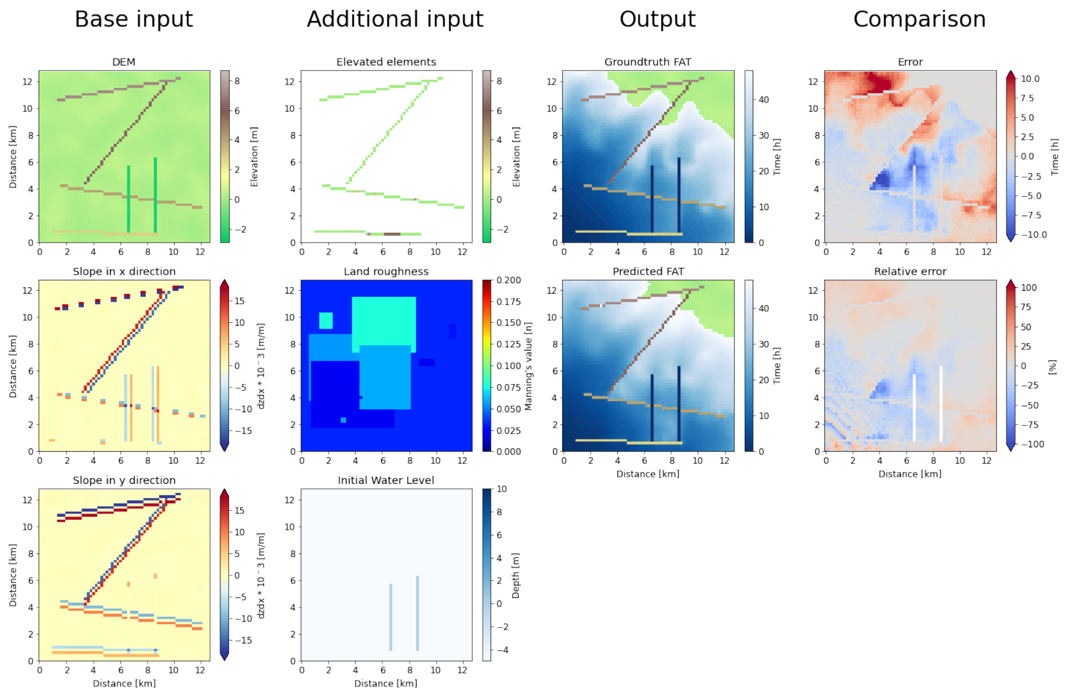
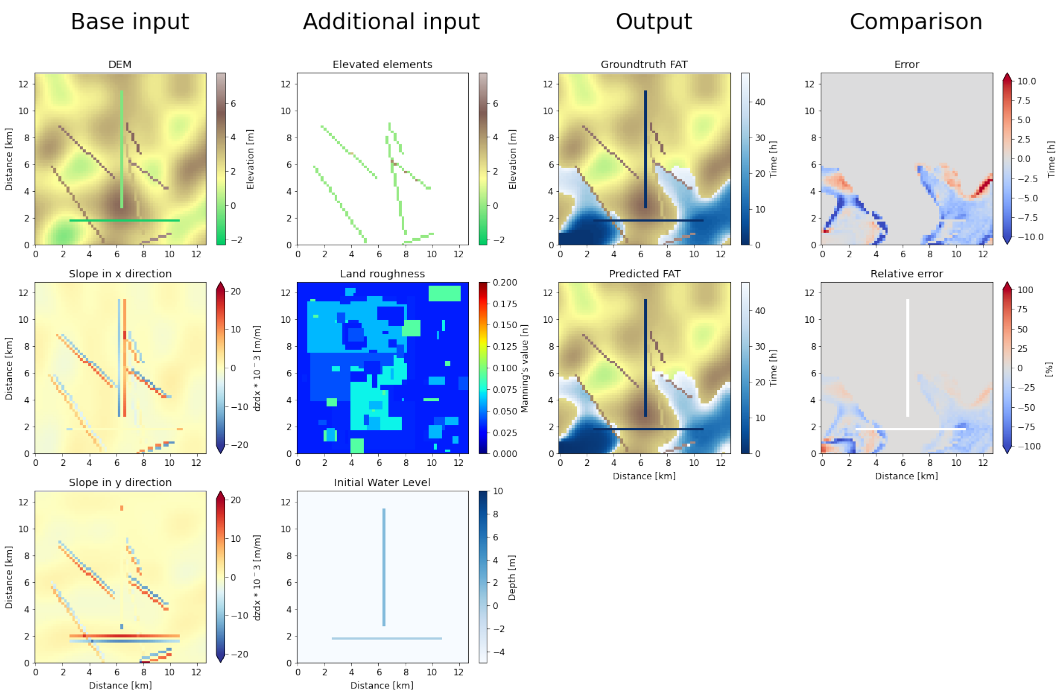


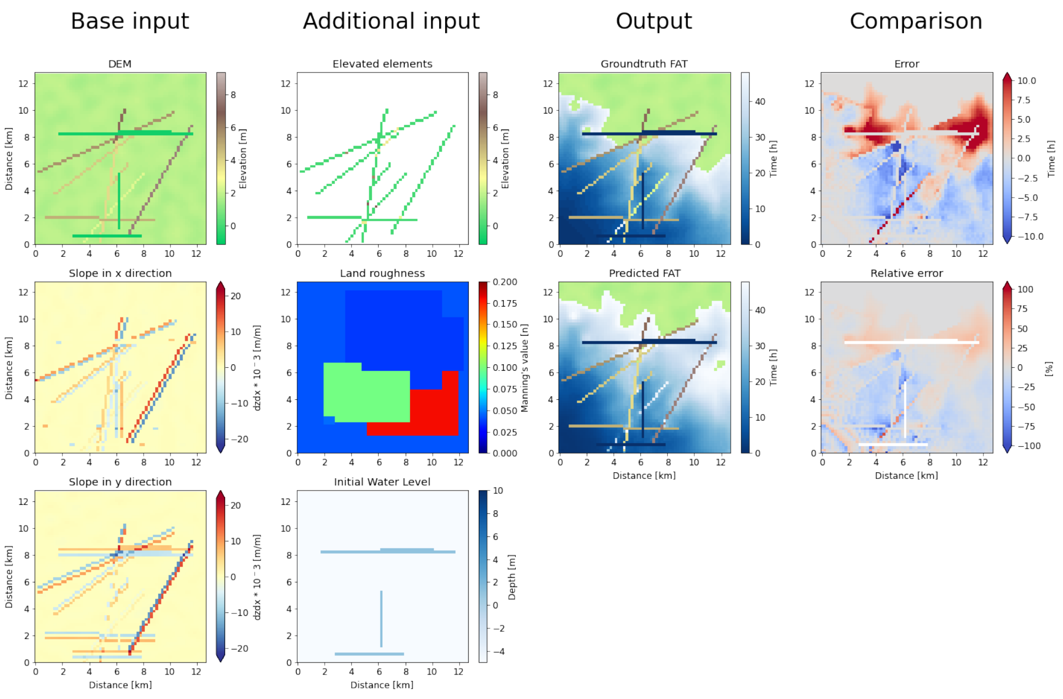
Figure B.3: First six flooding examples of dataset 3.

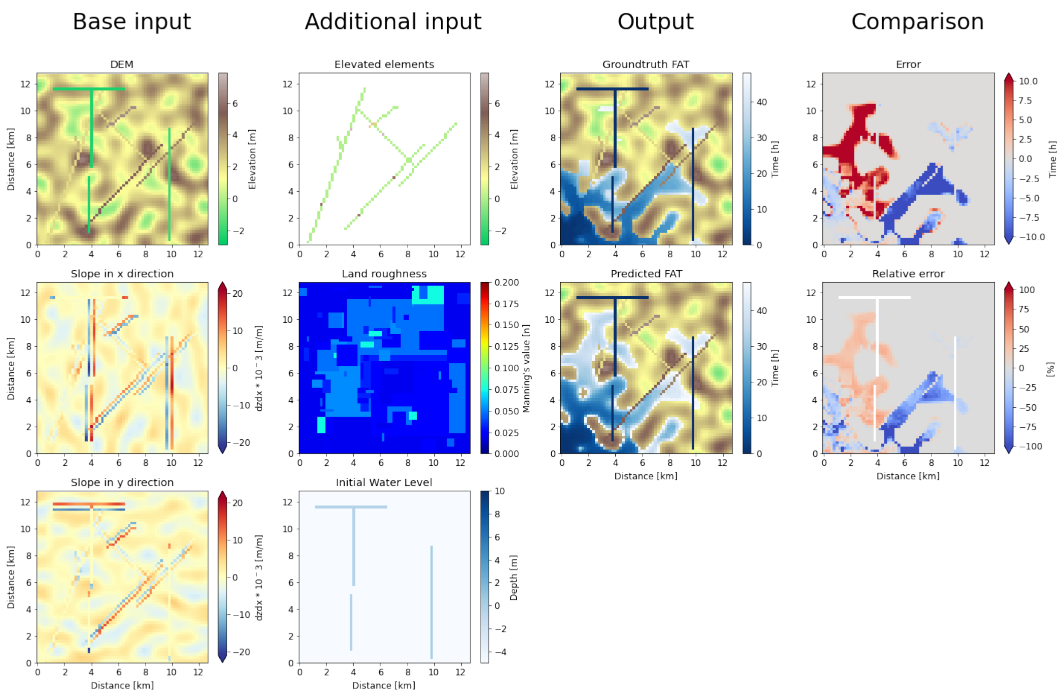
## B.4. Dataset 4


Base input
Additional input
Output
Comparison
DEM
Elevated elements
Groundtruth FAT
Error
Slope in x direction
Land roughness
Predicted FAT
Relative error
Slope in y direction
Initial Water Level
Distance [km]
Elevation [m]
Time [h]
dzdx * 10^3 [m/m]
Manning's value [n]
Depth [m]
[%]


Base input
Additional input
Output
Comparison
DEM
Elevated elements
Groundtruth FAT
Error
Slope in x direction
Land roughness
Predicted FAT
Relative error
Slope in y direction
Initial Water Level
Distance [km]
Elevation [m]
Time [h]
dzdx * 10^3 [m/m]
Manning's value [n]
Depth [m]
[%]


Base input
Additional input
Output
Comparison
DEM
Elevated elements
Groundtruth FAT
Error
Slope in x direction
Land roughness
Predicted FAT
Relative error
Slope in y direction
Initial Water Level
Distance [km]
Elevation [m]
dzdx * 10^-3 [m/m]
Manning's value [n]
Depth [m]
Time [h]
[%]


Base input
Additional input
Output
Comparison
DEM
Elevated elements
Groundtruth FAT
Error
Slope in x direction
Land roughness
Predicted FAT
Relative error
Slope in y direction
Initial Water Level
Distance [km]
Elevation [m]
dzdx * 10^-3 [m/m]
Manning's value [n]
Depth [m]
Time [h]
[%]



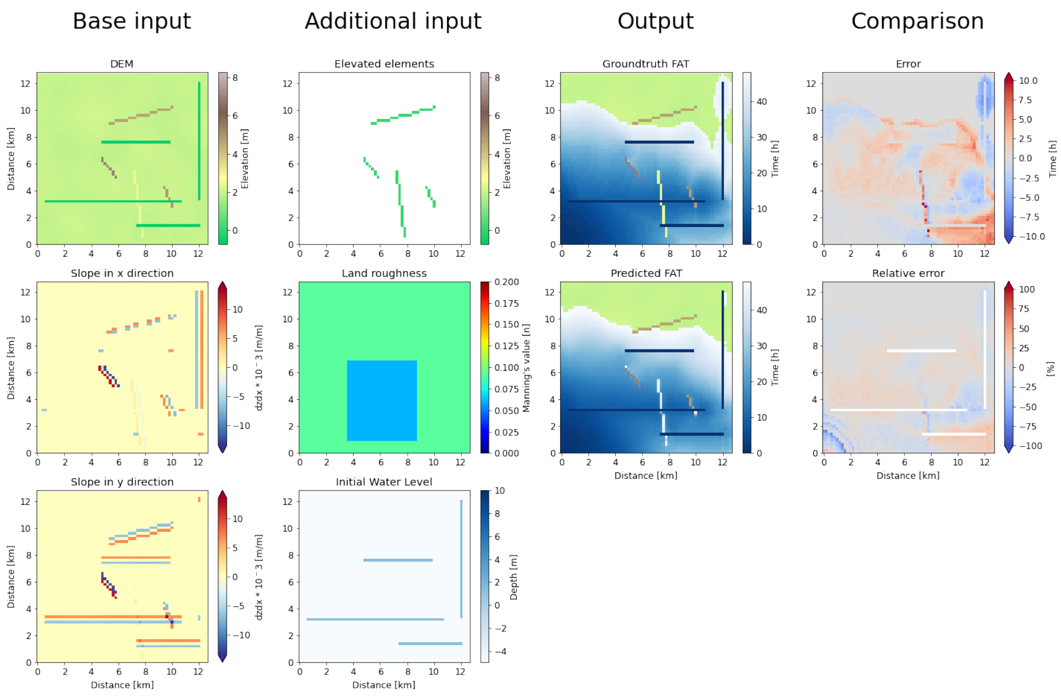


Figure B.4: First six flooding examples of dataset 4.

# C
# Location of dike ring 48

Figure C.1 shows the location of dike ring 48 in the Netherlands, and its grid.

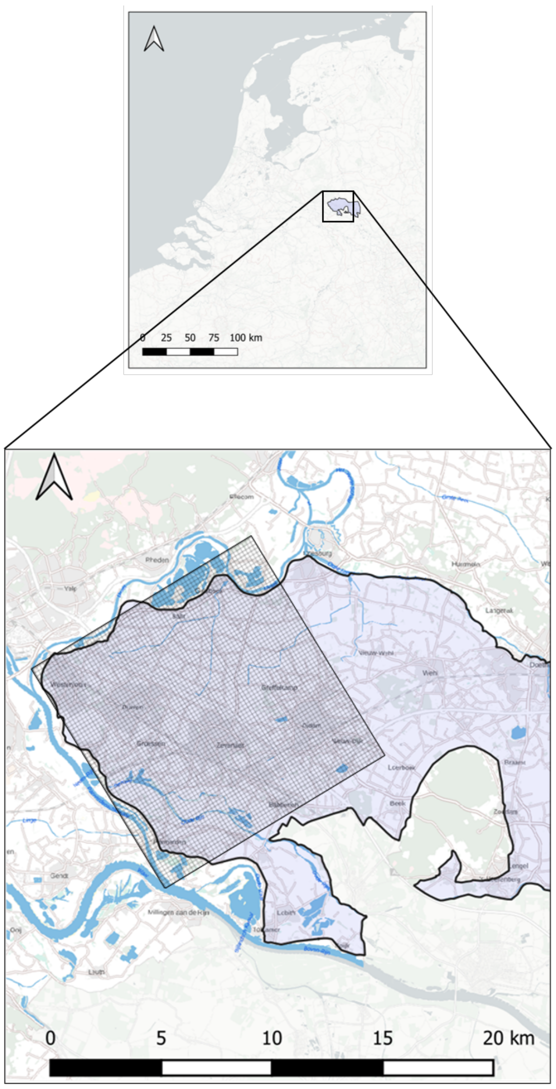


Figure C.1: The location of dike ring 48 in the Netherlands and the used grid

# D
# Variables correlation

The correlation matrix for each dataset is shown in following pages. The definition of the labels on the x and y axis are shown in Table D.1. The variables are quite self explanatory, with the only exception being the quantification of the hilliness, where relief is meant. Metrics such as the dDEM do not accurately represent the relief as a single large hill is not considered as a landscape with a strong relief. The average slope of a landscape is neither a good metric for the same reason, which also applies to the double derivative. Therefore, a custom relief quantification function has been created.

Each landscape is divided into height bins of one meter. First, a binary version of the landscape is created by setting a 1 for the cells which are below the threshold value. The rest of the landscape is assigned a 0. The number of patches with the value 1 is counted and added to a counter which starts at zero. The same process is repeated for the next bin, where previous binary bin map is subtracted. The number of patches is counted again and added to the counter. This process is repeated till all bins are done.

Table D.1: Definition of the labels for the correlation figures.

| Acronym | Explanation |
|---|---|
| mean_error | Domain wide error within a simulation. |
| dDEM | The difference between minimum and maximum DEM value within a simulation. |
| mean_DEM | The average DEM value within a simulation. |
| mean_dzdx | The average absolute slope of the DEM in x direction within a simulation. |
| mean_dzdy | The average absolute slope of the DEM in y direction within a simulation. |
| hilliness | The quantified degree of hilliness within a simulation. |
| mean_FATdx | The average absolute gradient of the FAT in x direction within a simulation. |
| mean_FATdy | The average absolute gradient of the FAT in y direction within a simulation. |
| flood_extent | The final percentage of flooded cells within a simulation. |
| n_elements | The number of elements within a simulation. |
| mean_roughness | The average roughness value within a simulation. |
| n_waterways | The number of WWs within a simulation. |
| error_flooded | The average error per flooded cell within a simulation. |

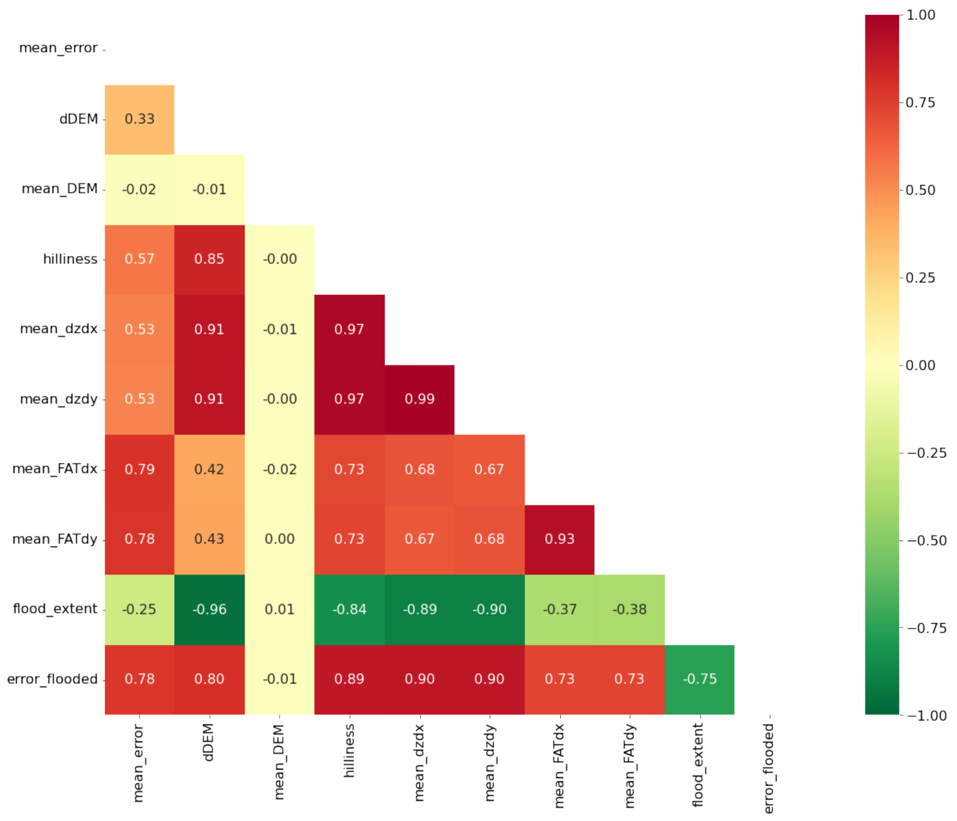


Figure D.1: Correlation coefficients between a range of variables in dataset 1.

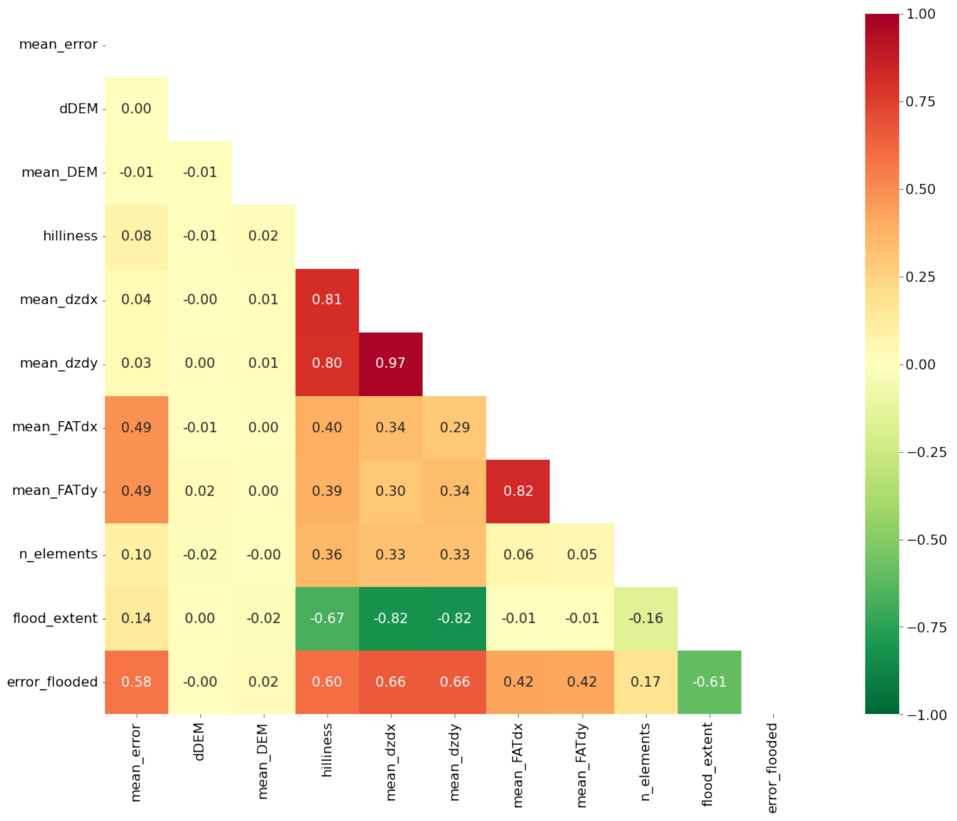


Figure D.2: Correlation coefficients between a range of variables in dataset 2.

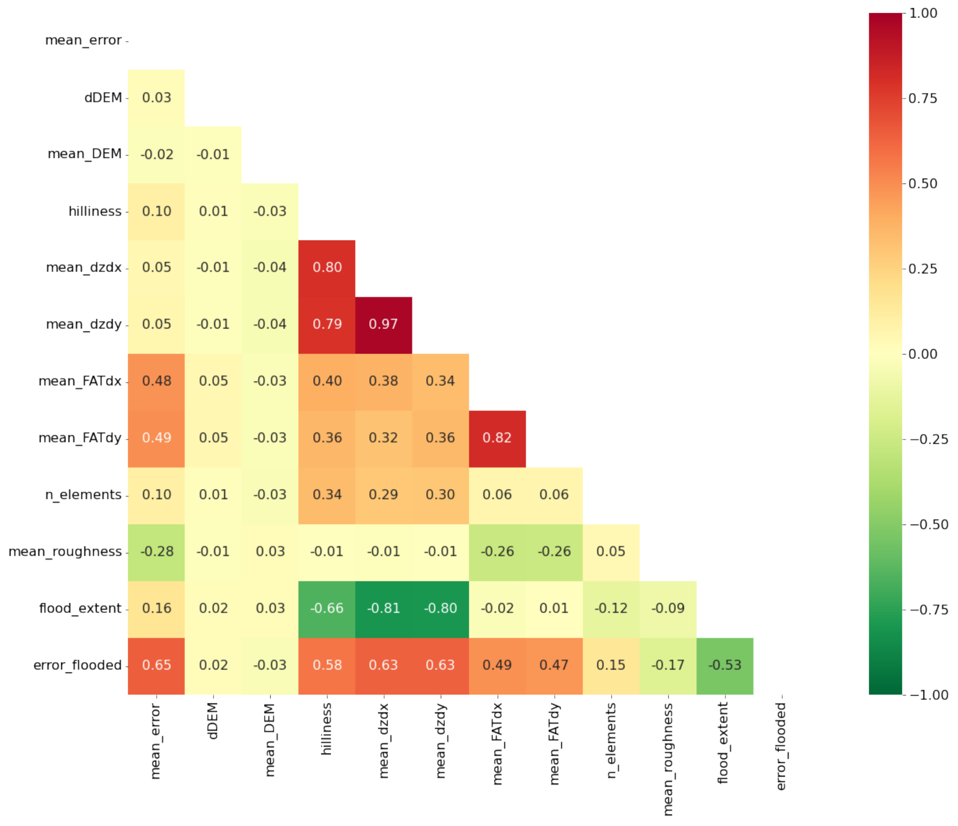


Figure D.3: Correlation coefficients between a range of variables in dataset 3.

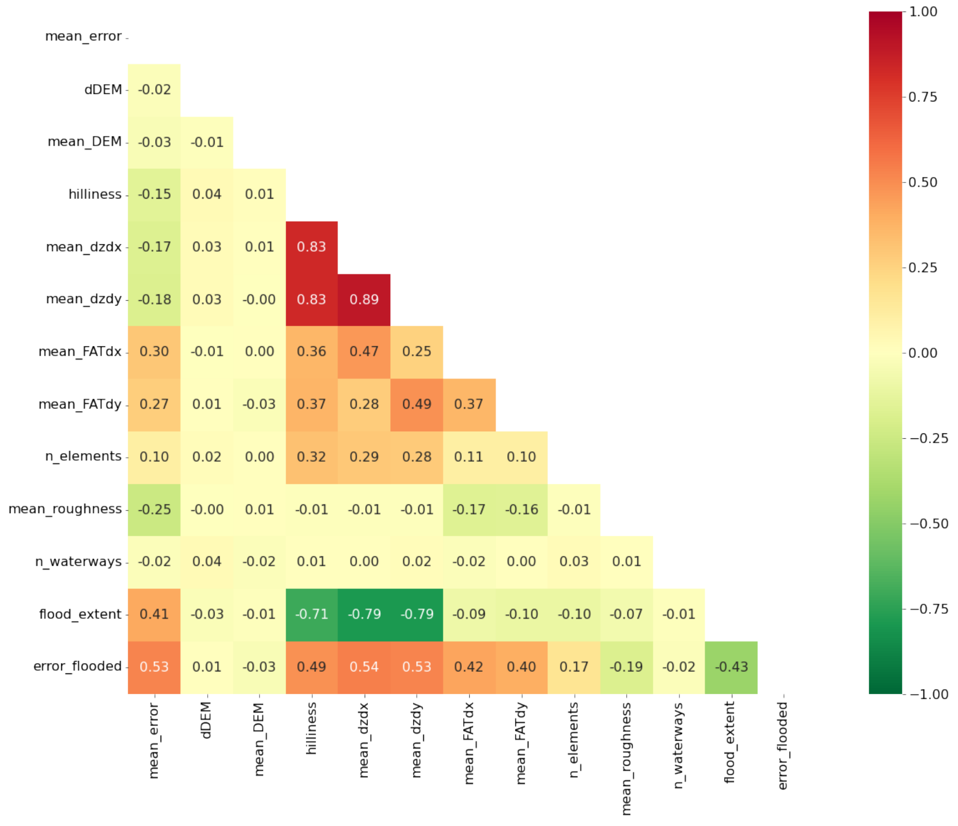


Figure D.4: Correlation coefficients between a range of variables in dataset 4.

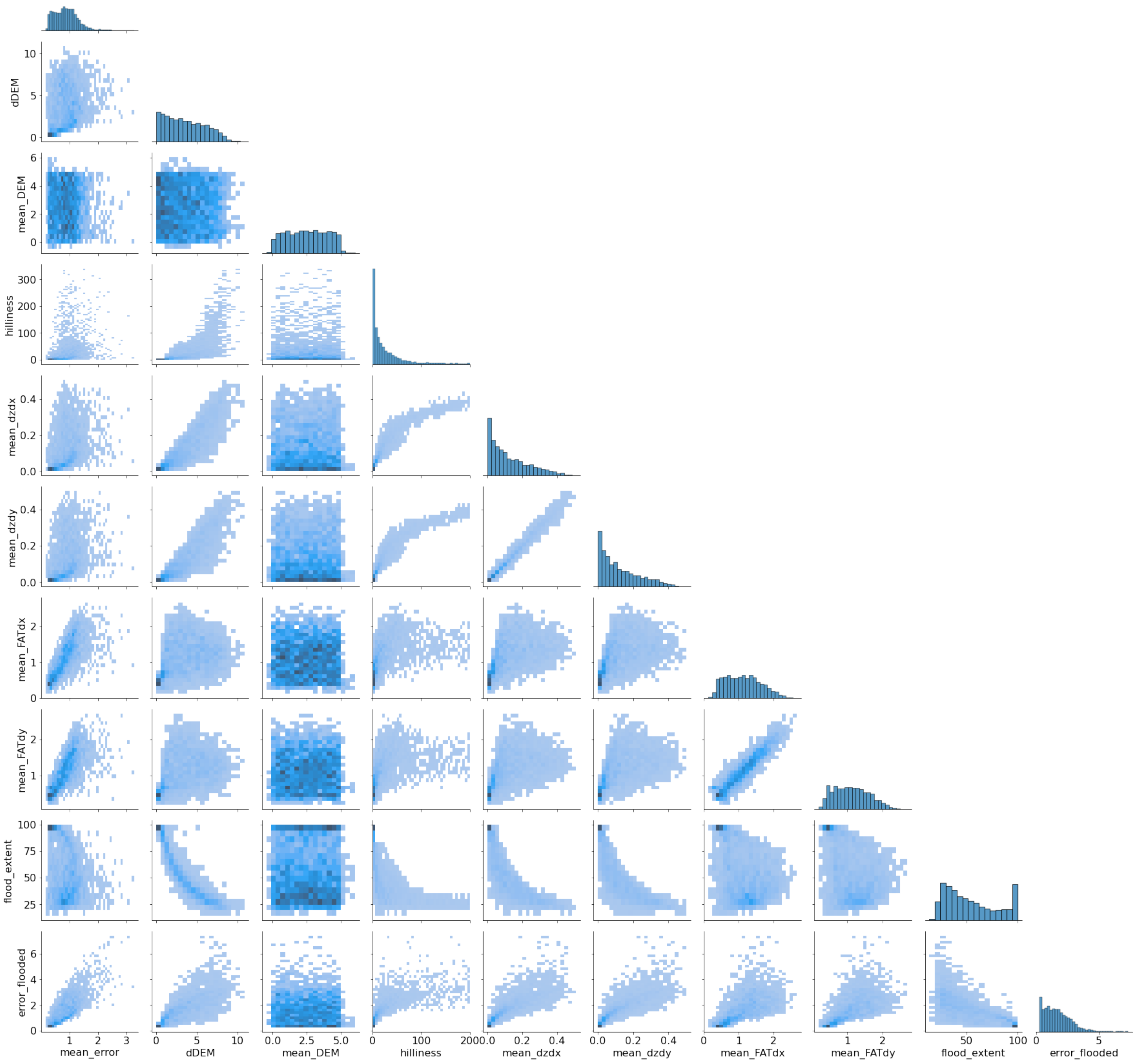


Figure D.5: Correlation histograms between a range of variables in dataset 1.

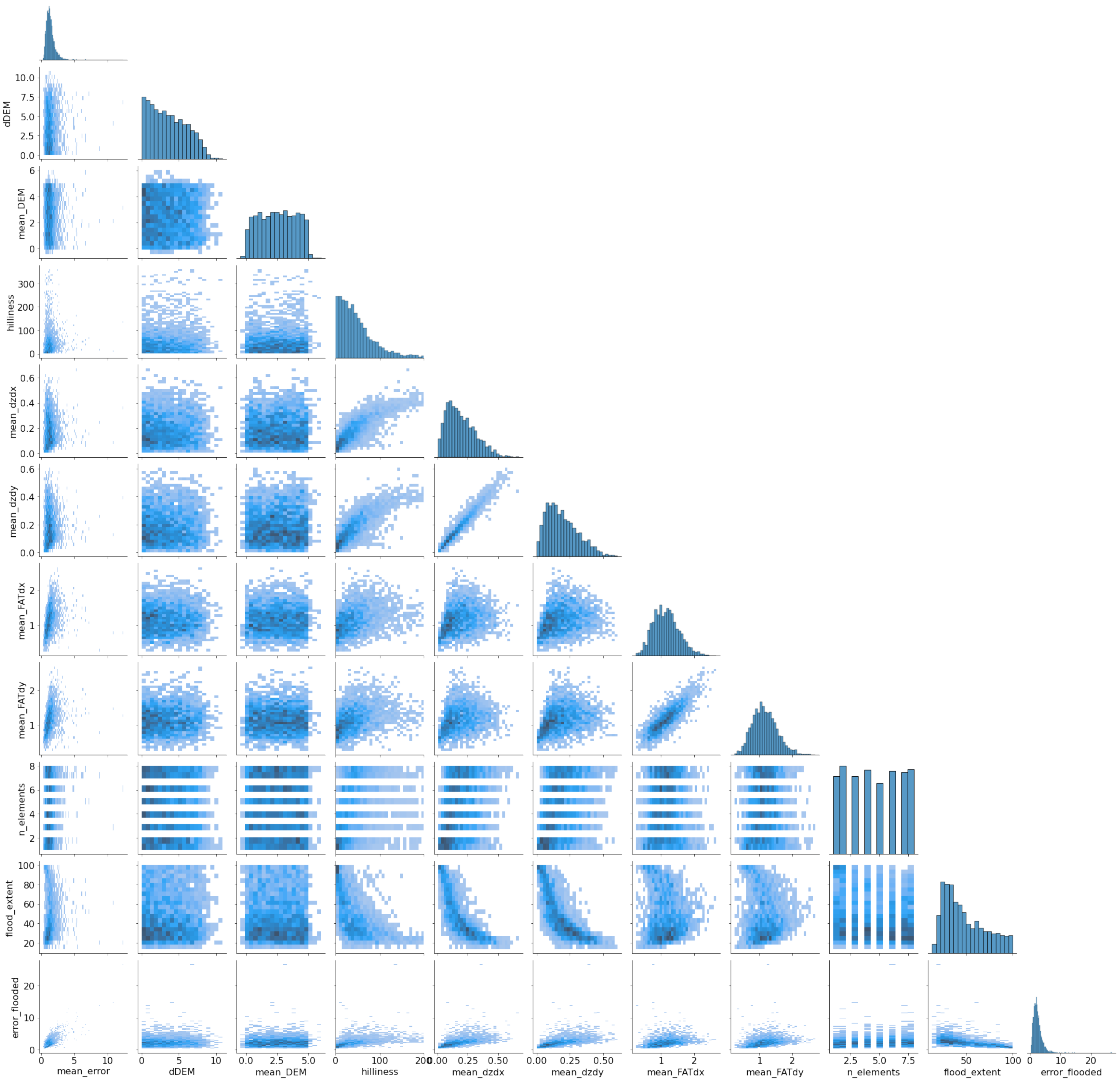


Figure D.6: Correlation histograms between a range of variables in dataset 2.

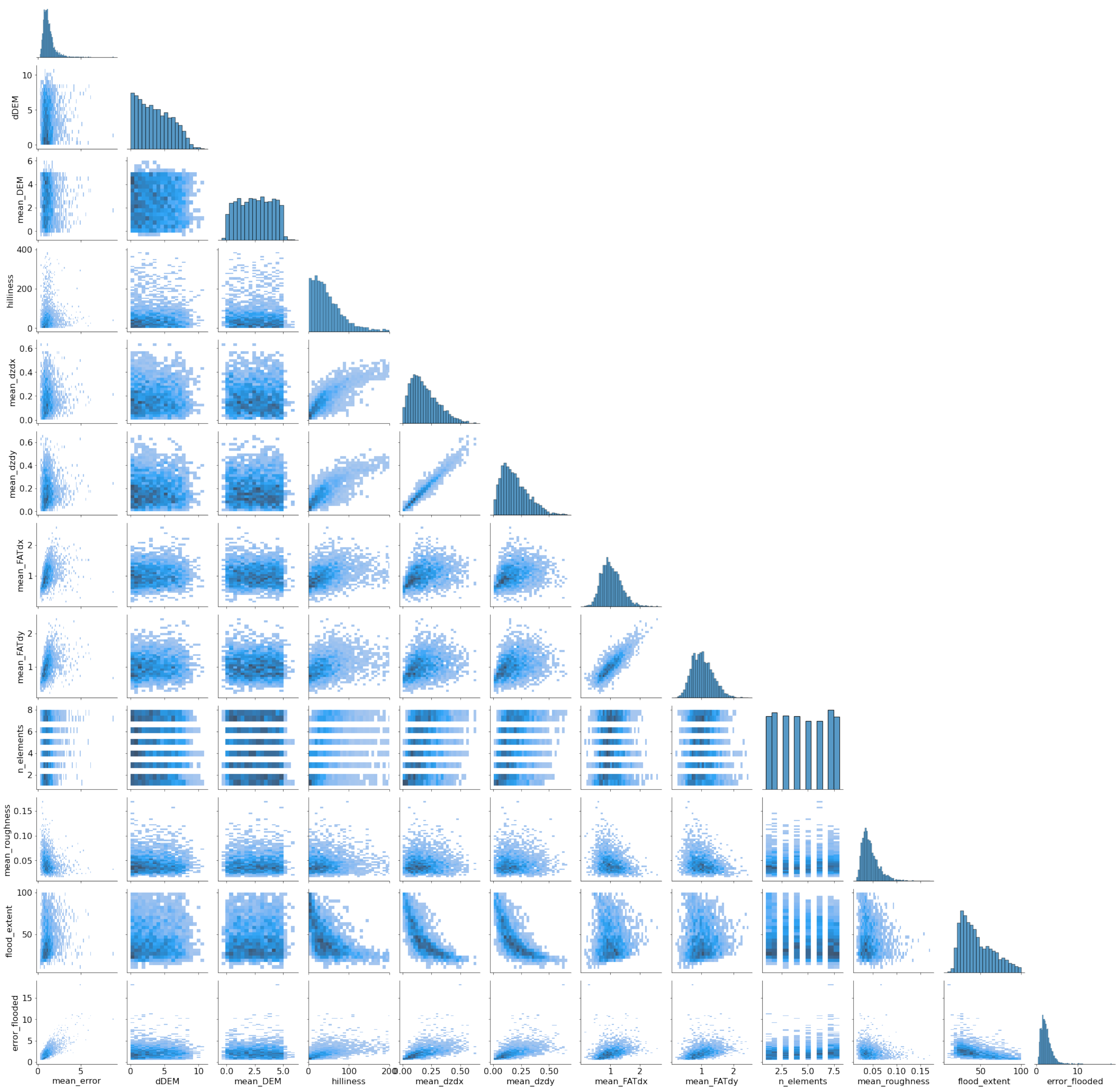


Figure D.7: Correlation histograms between a range of variables in dataset 3.

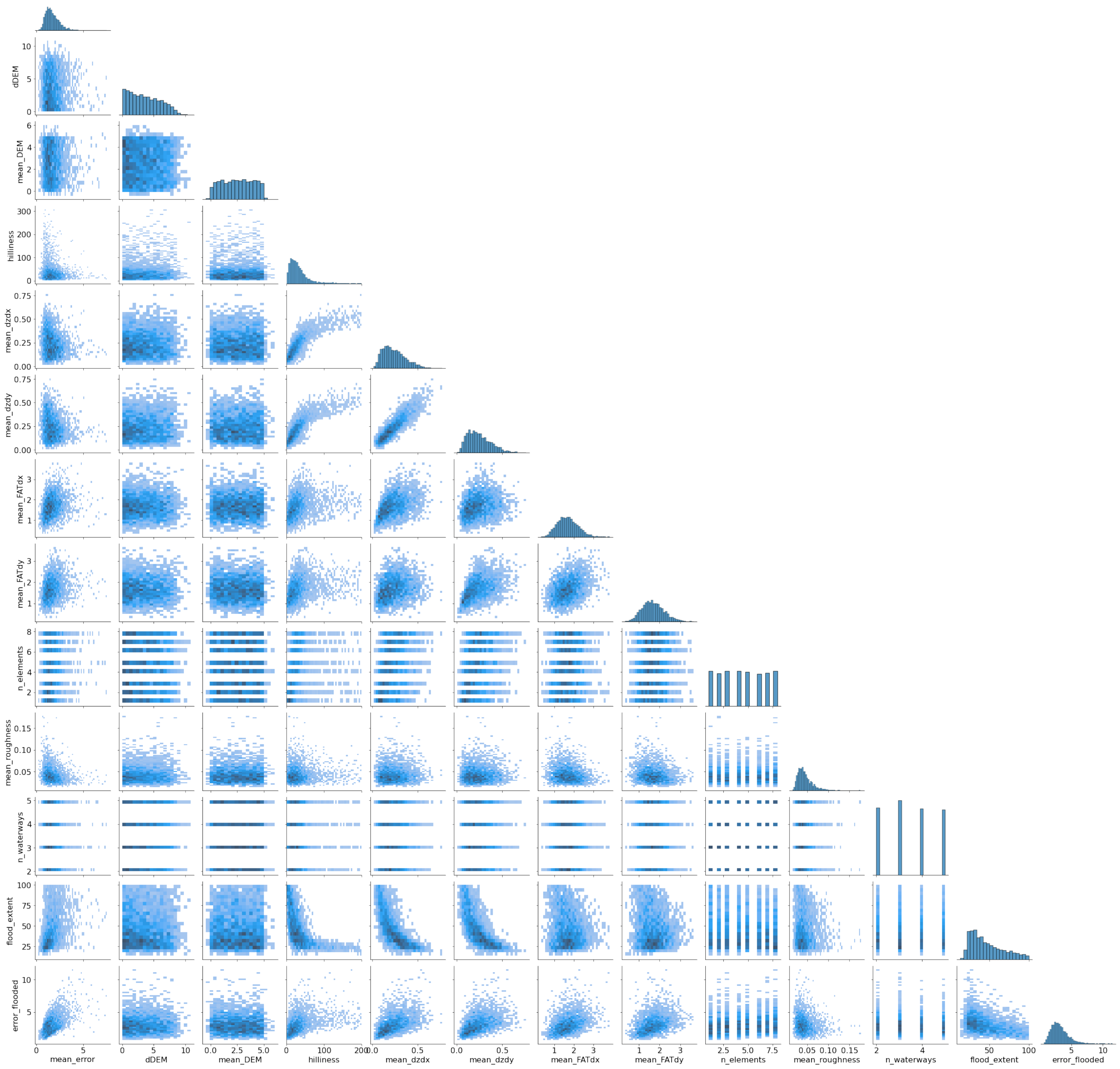


Figure D.8: Correlation histograms between a range of variables in dataset 4.

# E

# Performance in flooded and dry regions

The performance of each model divided into flooded and dry regions is shown in Table E.1.

Table E.1: Model's performance for flooded and dry regions.

| | Groundtruth flooded cells | | | | Groundtruth dry cells | | | | | | All cells |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Overprediction | | Underprediction | | Overprediction | | Underprediction | | Cortrectly predicted cells | | |
| | Error [h] | Frequency [%] | Error [h] | Frequency [%] | Error [h] | Frequency [%] | Error [h] | Frequency [%] | Error [h] | Frequency [%] | Flood extent [%] |
| Model 1 | 1.15 | 45.91 | 1.53 | 54.09 | 4.15 | 8.01 | - | - | 0.00 | 91.99 | 54.35 |
| Model 2 | 1.83 | 46.60 | 2.52 | 53.40 | 5.08 | 13.54 | - | - | 0.00 | 86.46 | 48.06 |
| Model 3 | 1.76 | 41.06 | 2.39 | 58.94 | 5.07 | 10.37 | - | - | 0.00 | 89.33 | 45.16 |
| Model 4 | 2.46 | 38.54 | 3.56 | 55.54 | 6.52 | 11.44 | - | - | 0.00 | 88.56 | 46.58 |

# F

# Confusion matrix over time

A confusion matrix between the groundtruth and predicted FAT values are shown in Figure F.1.

A confusion matrix for the binary classification of the flood extent over time has been plotted in Figure F.2, which forms the basis to compute the metrics mentioned in the main text.

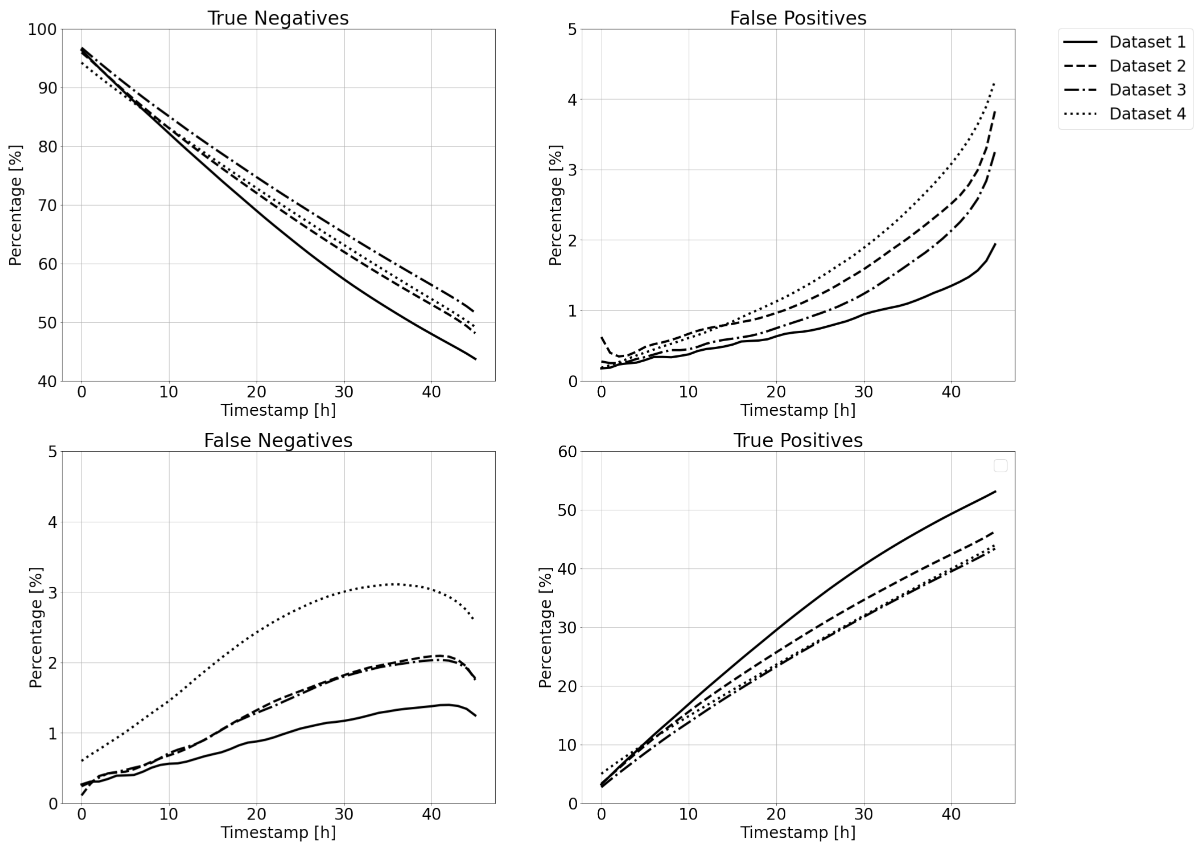


Figure F.2: Confusion matrix over time for each dataset

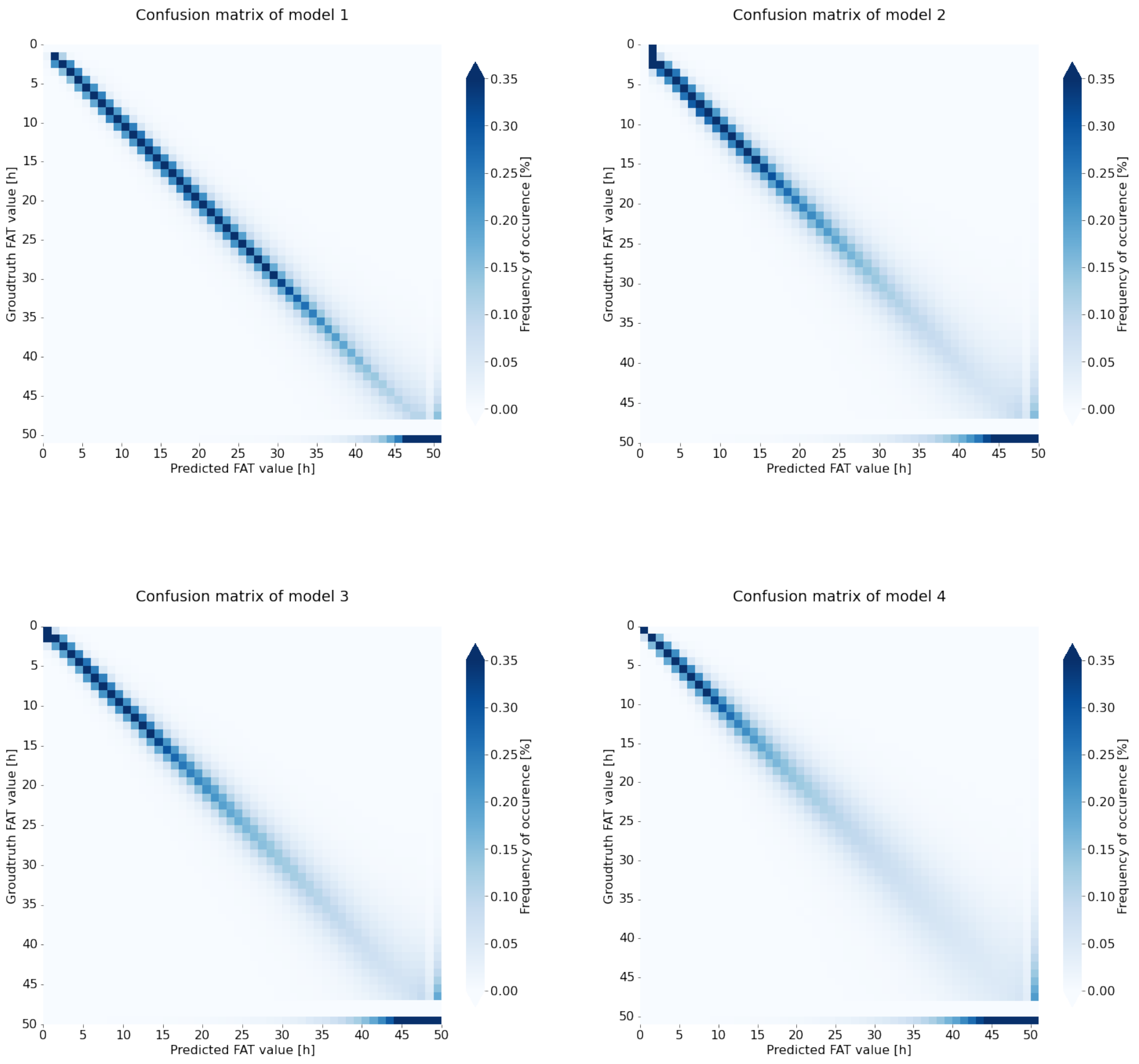


Figure F.1: Confusion matrix of the groundtruth and predicted FAT values per model

# G
# Overview BL input layers

Figure G.1 shows all input layers for the BLs.

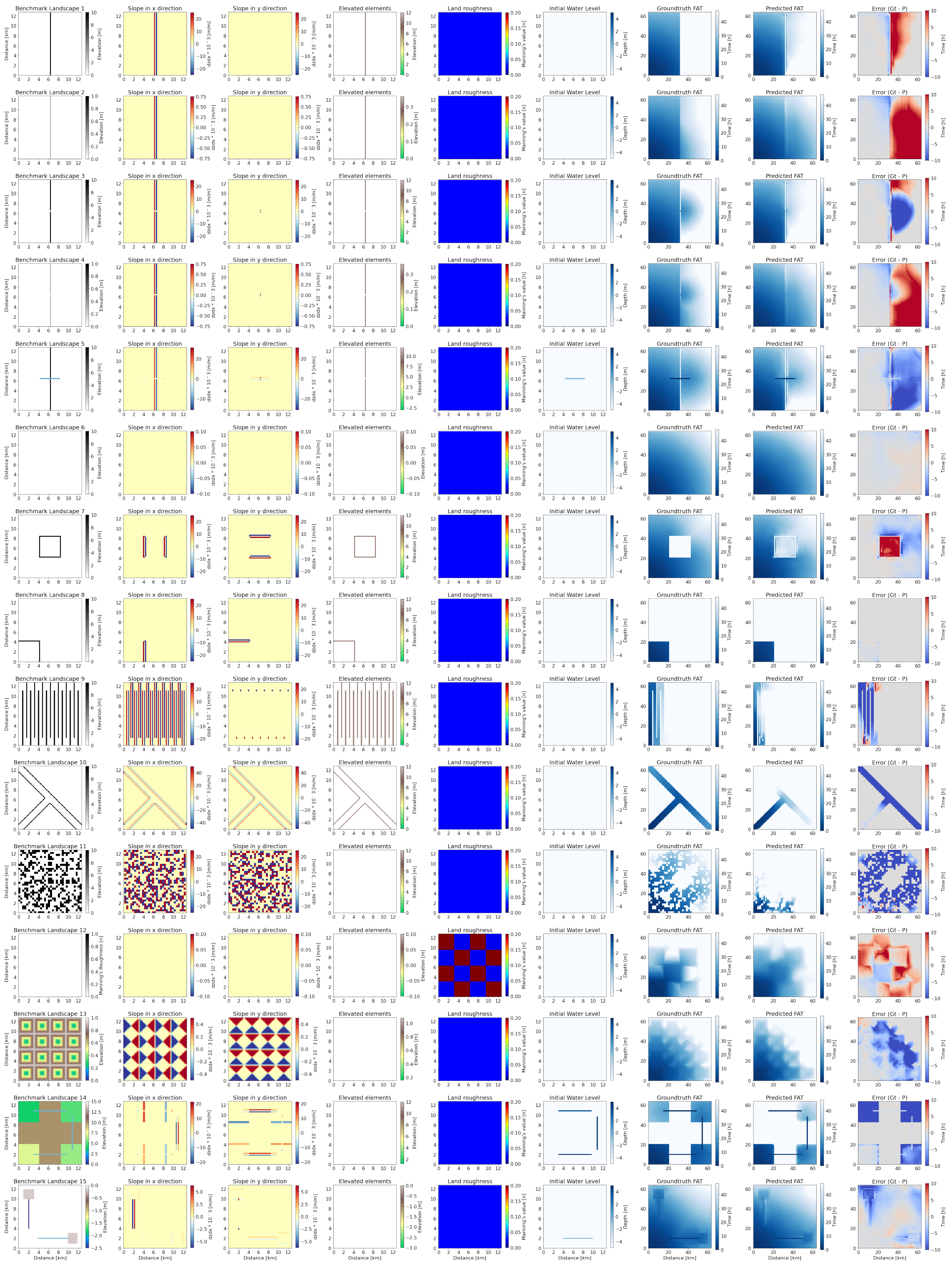


Figure G.1: Overview of all BL input layers