18th International Conference on the Application of Computer
Science and Mathematics in Architecture and Civil Engineering
K. Gürlebeck and C. Könke (eds.)
Weimar, Germany, 07–09 July 2009

# ARTIFICIAL NEURONAL NETWORKS IN ENVIRONMENTAL ENGINEERING: THEORY AND APPLICATIONS

**T. Berthold**[*], **P. Milbradt**

[*]*Institute for Computer Science in Civil Engineering, Leibniz University of Hanover*
*Callinstraße 34, 30167 Hanover, Germany*
E-mail: berthold@bauinf.uni-hannover.de

**Keywords:** Artificial Neuronal Network, Backpropagation, Bathymetry, Generalized Approximation.

**Abstract.** *Models in the context of engineering can be classified in process based and data based models. Whereas the process based model describes the problem by an explicit formulation, the data based model is often used, where no such mapping can be found due to the high complexity of the problem. Artificial Neuronal Networks (ANN) is a data based model, which is able to "learn" a mapping from a set of training patterns. This paper deals with the application of ANN in time dependent bathymetric models. A bathymetric model is a geometric representation of the sea bed. Typically, a bathymetry is been measured and afterwards described by a finite set of measured data. Measuring at different time steps leads to a time dependent bathymetric model. To obtain a continuous surface, the measured data has to be interpolated by some interpolation method. Unlike the explicitly given interpolation methods, the presented time dependent bathymetric model using an ANN trains the approximated surface in space and time in an implicit way. The ANN is trained by topographic measured data, which consists of the location* $(x, y)$ *and time* $t$*. In other words the ANN is trained to reproduce the mapping* $h = f(x, y, t)$ *and afterwards it is able to approximate the topographic height for a given location and date. In a further step, this model is extended to take meteorological parameters into account. This leads to a model of more predictive character.*

## 1 INTRODUCTION

Models in the context of engineering can be classified in process based and data based models. Dependencies of values of interest, in the case of process based models, can be described by a set of equations. Those can be solved analytically or approximated by appropriate methods. Many (physical) processes are so extensive, that no appropriate description can be formulated. But often, there is some (measured) data, that describes the process in an implicit way, which can be used for a data based model. Artificial Neuronal Networks (ANN) is a data based model, which is able to "learn" a mapping from a set of training patterns. This paper deals with the application of ANN in time dependent bathymetric models.

A bathymetric model is a model of the underwater topography. The seabed is changed tremendously with time due to high pressures and velocities, that the water applies on it. In contrast to a topography, a bathymetry changes quickly in general. The evaluation of long-term morphological changes and the way they form the seabed is an essential prerequisite for planning, designing and realizing of sustainable concepts for coastal protection and development.

In this paper a generalized approximation method for a time dependent bathymetric model is presented, using ANN to learn bathymetries implicitly given by measured data. Therefor a short overview of the theory of ANN is given in section 2, followed by the modeling of bathymetries using ANN in section 3.

Another matter of interest is how to describe the sediment transport on the sea bed influenced by wind and tidal effects in order to predict the movement of a sea channel. The model is extended in section 4 to take meteorological effects into account.

Some application of the model is presented, regarding the domain around the Medem Channel.

## 2 ANN: THEORETICAL OVERVIEW

In the context of Artificial Neuronal Networks (ANN) the functioning of the human brain and its ability to abstract is tried to be carried to the computer as a simplified model. ANN are able to reproduce dependencies between parameters, that are given by a set of input and output patterns. That makes ANN an inductive method, because there is no (or less) need for a priori knowledge, as the physical context of a phenomenon. There are many applications for ANN: pattern recognition, approximation of functions in general and forecasting of timeseries or determination of wave overtopping[4] for coastal structures in civil engineering.

### 2.1 Modeling

As the information in the brain is mainly processed and influenced by neurons and synapses, these build the basic elements of ANN as well. A neuron can be understood as a "mini-processor", that produces an output signal, if the incoming signal exceeds a certain level. This output is sent to connected neurons via the nerv tracts. Between two neurons another cell is arranged, which is called *synapse*. A synapse influences the magnitude of the transmitted signal by decreasing or increasing it. That has an important effect on the signal processing and therefore establishes the basis for many learning algorithms. Many models for neurons, synapses and networks have been developed in the past. The properties can be summarized as follows:

A neuron $N$ is modeled as a node of the network and is connected with its *antecessor* and *successor* neurons by synapses. It is composed of its input signal $net^N$, the activation function $f_a^N$ and the activation value $a^N$ (output). The input is determined by summation over all activation values of its antecessors $N_i$ multiplied by the weights of the connecting synapses ($a^N = \sum_i a^{N_i} w^{N_i N}$). The activation value is determined through the activation function with respect to the input: $a^N = f_a^N(net^N)$. Figure 1 illustrates the properties of a neuron. It is useful to differentiate between input, output and hidden neurons. Input and output neurons are special neurons, that act as input and output "locations" in a network. A neuron, that does not have an activation function, as well as no antecessors is called bias neuron. It constantly sends the output 1.

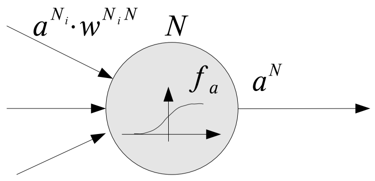


Figure 1: Properties of a neuron

The synapses connect the neurons and provide the propagation of the information passed from one neuron to its successors. A synapse is typically modeled as a weighted edge of the network.

## 2.2 Network Topology

Classification of ANN is based on the network topology. Basically, ANN are divided into recurrent networks and those without feedback connections. The latter are called feedforward-networks due to the directed information flow between input and output neurons. Figure 2 illustrates an example of a general network.

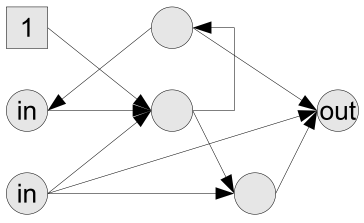


Figure 2: Example for a general network (including feedback connections). The input neurons are marked as "in", the output neurons as "out". The rectangle describes a bias neuron sending the output 1. Synapses are modelled as weighted edges (The weights are not illustrated here).

A class of feedforward networks, where the neurons are arranged in layers is called multi-layer perceptron. The first layer consists of input neurons and is called input layer. The output layer is set up by output neurons and is located at the last position. Between these layers an arbitrary number of hidden layers are placed. Each layer is completely connected with the subsequent one. This topology is depicted in figure 3.

The activation function influences the behavior of a network, too. There are different types of activation functions. The unit step function is typically used for theoretical purposes or networks, that are set up analytically. An output signal is sent, if the input signal exceeds the threshold $\Theta$:

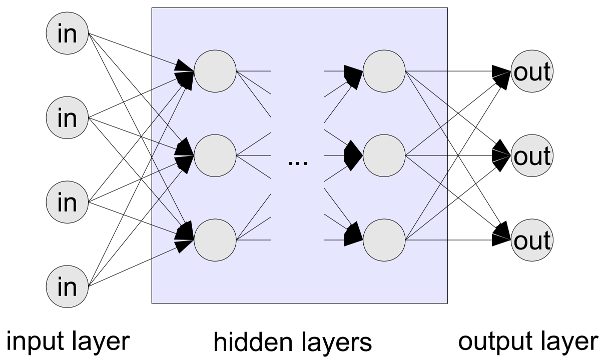


Figure 3: Network topology of a multilayer perceptron. Additionally a bias neuron could be included.

$$a^N = \begin{cases} 1 & net^N > \Theta \\ 0 & \text{else} \end{cases} \tag{1}$$

It reflects the properties of the biological neurons; they switch to the activated state, if the input exceeds a threshold. Some learning rules base on the derivation of the activation function, which is not defined for the unit step function. Therefore, sigmoid functions have been introduced as activation functions, like the logistic function or the hyperbolic tangent. For those functions, the derivation is defined in every point. In this paper, a multilayer perceptron is used and the logistic activation function is applied for the neurons. The logistic function and its derivation is defined as:

$$f_{act}(x) = \frac{1}{1+e^{-c\cdot x}} \qquad f'_{act}(x) = \frac{c\cdot e^{-cx}}{(1+e^{-cx})^2} \tag{2}$$

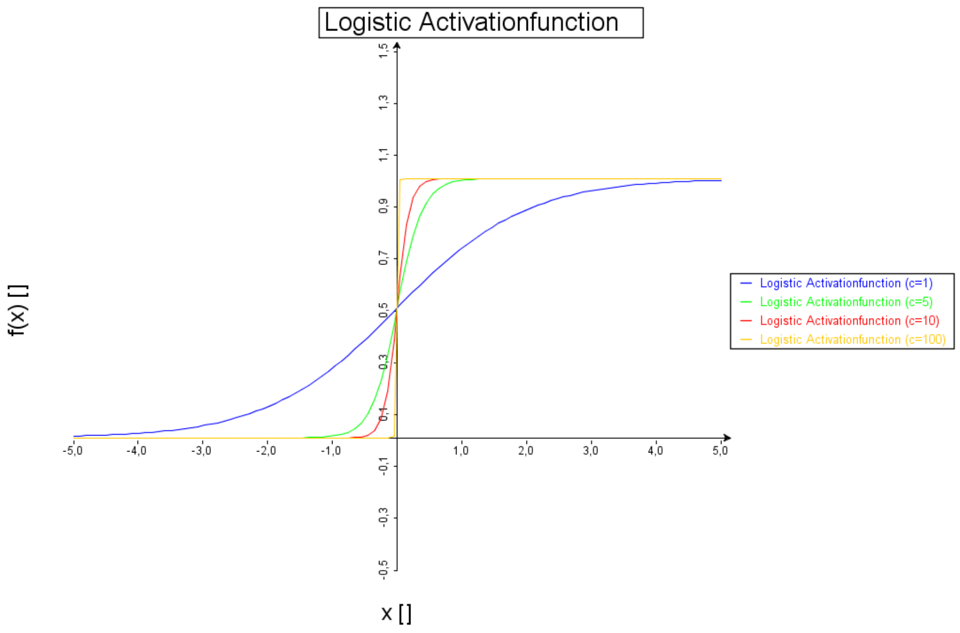
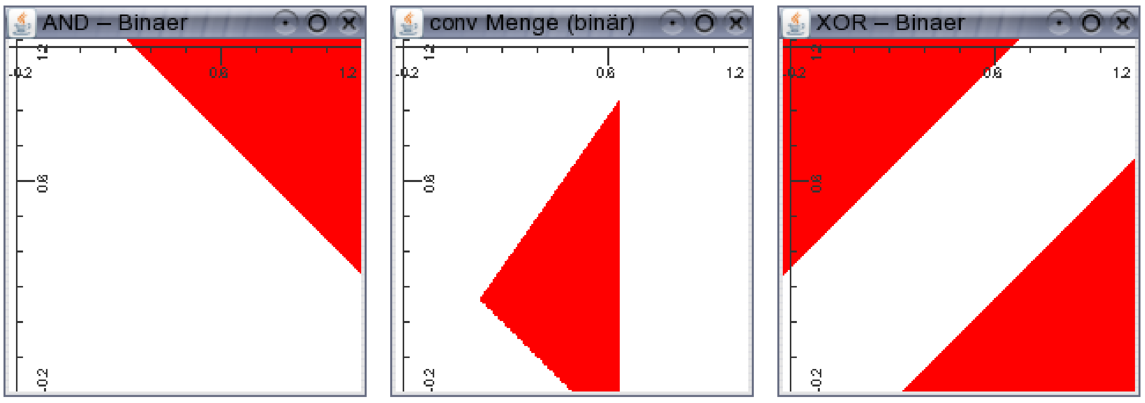


Figure 4: Logistic activation function with different values for $c$

For $c \to \infty$, the logistic function is equal to the unit step function (compare to figure 4).

## 2.3 Representativeness

An important point in the process of modeling a network is the capability of representing the desired mapping. That depends significantly on the chosen network topology. An overview of the representativeness of mappings subject to network topology is given in [1]. The author relates to binary perceptrons[1] only. Based on a layered feedforward network, he demonstrates the influence of the number of layers: A network with two layers[2] is able to linearly separate the space of input parameters only. By introducing another layer, the network is able to represent a logical combination of linearly separated sets. Using four layers, i.e. three layers of changeable synapses, leads to a network that can represent sets with an arbitrary shape by logical combination of convex sets. Figure 5 shows the influence of topology on the capability of representing a function of a network. Creating a binary perceptron with more than four layers does not lead to more flexibility in representability.



(a) Network with two layers, representing linearly separable sets (b) Network with three layers, representing a convex set (c) Network with three layers, representing the XOR-function

Figure 5: The influence of the topology on presentability considering a binary perceptron

## 2.4 Training process

The training process is an iterative process. At the end of the process, the network should approximate the given mapping well. In other words, the difference between the desired output and the output calculated by the network with respect to the current settings has to be minimized. Therefore, the error can be formulated as a function of the weights of the synapses, which change during the training process:

$$E = E(w_1, \dots, w_n) \tag{3}$$

During the training process, each training pattern is presented to the network, and the learning rule defines how to adapt the properties of the network with a view to correct the predicted output. In theory, there are three types of learning paradigms:

**Supervised learning** According to a given input pattern, the network determines the current output values. Presenting also the desired output pattern, the network is able to calculate the difference between the current and the correct output values and to adapt network properties.

[1] A binary perceptron is a model of a neuron with a binary activation function.

[2] That means, that the network contains one layer of changeable synapses.

**Reinforcement learning** As in the previous case, the network calculates the current output depending on the given input. In contrast, not the absolute value of the correct output parameters is told, but only the information wheter the predicted value is right or wrong.

**Unsupervised learning** The input patterns are presented only. The learning process takes place without any information about the desired output parameters.

In the case of approximating a bathymetry, it is useful to apply a supervised learning method, because the exact input and output patterns are known from the measured data. The backpropagation learning algorithm is often used in combination with a multilayer perceptron. This algorithm is a gradient descend method and works as follows:

1. Initialize each weight of the synapses randomly
2. Present a randomly chosen pattern of the training set to the network: Set the values of the input layer to the values of the input pattern
3. Determine the output values of the output neurons of the network
4. Adapt the weights of the synapses between neuron $N_i$ and $N_j$ as follows:

$$w^{N_iN_j} = w^{N_iN_j} + \Delta w \tag{4}$$

$$\Delta w^{N_iN_j} = \eta \delta^{N_j} a^{N_i} \tag{5}$$

$$\delta^{N_j} = \begin{cases} f'_{act}(e^{N_j}) \cdot (y_j - f_{act}(e^{N_j})) & N \text{ is output neuron} \\ f'_a(e^{N_j}) \cdot (\delta_k w^{N_jN_k}) & N\text{is hidden neuron} \end{cases} \tag{6}$$

5. Go to 2. and repeat until the error is less than a given tolerance or a maximum number of iteration steps is reached

## 3 BATHYMETRIC MODELS

A bathymetric model is typically based on measured data, which is obtained by an echo sounder. The echo sounder measures the depth at certain locations, which leads to a finite set of data. Assuming that there is only one measured value for each measuring location, the height $h$ can be interpreted as a function that depends on the location: $h = f(x, y)$. To get a continuous surface from the discrete distribution, the model has to provide an interpretation method. The interpretation method specifies how to calculate the value $h$ for a given location $(x, y)$. Interpretation methods can be divided into interpolation and approximation methods. An interpolation method $I$ has to hit the measured values $h_i$ at the related position $p_i$ exactly:

$$I(p_i) \overset{!}{=} h_i \tag{7}$$

In contrast, approximation is a more general method, where the goal is to find a function $A$, that nearly hits the measured data, but does not have to hit exactly:

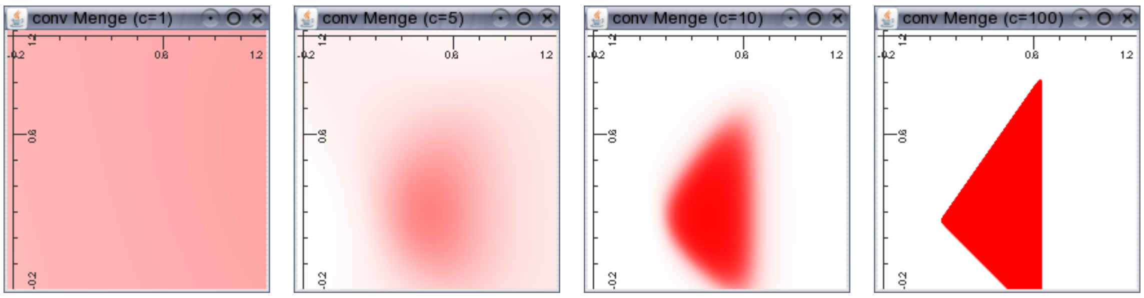
$$A(p_i) \approx h_i \tag{8}$$

### 3.1 ANN as bathymetric model

The idea of this paper is to use ANN as an approximation method for bathymetries. A multilayer perceptron is used as the model. It consists of two input neurons for the location $(x, y)$ and one output neuron for the height. The approximated mapping of the underwater surface is iteratively trained first, using the measured data. For this purpose the backpropagation learning algorithm is applied, as described in section 2.4. Afterwards, the ANN can be applied as a bathymetric model, using a requested location $(x, y)$ as input pattern. After propagation of this information through the network, the approximated value of the height can be read from the output neuron. A good approximation depends on the network topology and the given training set, as described in the following sections.

### 3.2 Network Topology

According to the demonstration of the binary perceptron model, the representation of bathymetries has to be considered. Contrary to the binary case, a bathymetry needs a model with continuous values in the height range. Therefore the logistic activation function is used (compare section 2.2). Observing the same topology and replacing the binary activation function by the logistic activation function, the represented mapping changes. The change is significantly influenced by the choice of the parameter $c$: A larger value leads to a better approximation of the binary function and therefore results in sharper edges, whereas smaller values lead to a smoother surface (see figures 6 and 4).



(a) $c = 1$ (b) $c = 5$ (c) $c = 10$ (d) $c = 100$

Figure 6: Influence of the parameter $c$ of the logistic activation function

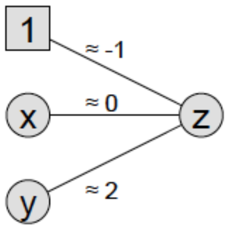
### 3.3 Representation of characteristic structures

Regarding a layered network with two input neurons for the location $(x, y)$ and one output neuron representing the height $h$ some typical structures in bathymetries will be shown.

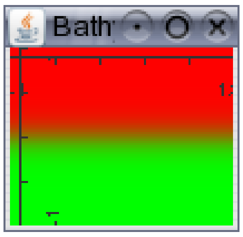
An ascent can be modeled without introducing another neuron, because the sigmoid activation function of the output neuron itself reproduces such a structure. The weight of the synapses that connects the input neuron for $x$ and the output neuron defines the dependency of the $x$-value. The influence of $y$ is given respectively. An offset can be taken into account by adding a bias neuron with appropriate synapses and weights. Figure 7 shows an example of an ascent with an according network topology.

The steepness of the ascent can be varied by increasing or decreasing the weights of the synapses.

Following the XOR-function in the binary case, a see channel may be emulated by two conditions in terms of one neuron each. The conditions are represented again by the weights



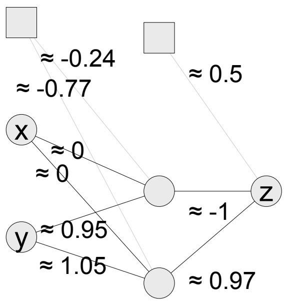
(a) Topology of the network (2-1) with trained synapses



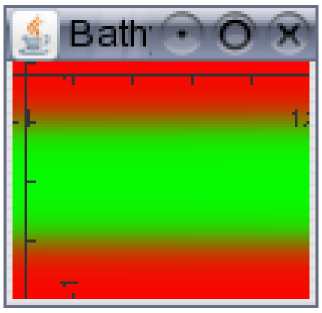
(b) Bathymetry to learn

Figure 7: Topology of a network, that is able to represent an ascent of the bathymetry

of the synapses. The second changeable layer of synapses connects the two conditions. Both, the network topology including the weights of the synapses and the learned bathymetry are visualized in figure 8.



(a) Topology of the network (2-2-1) with trained synapses



(b) Bathymetry to learn

Figure 8: Topology of a network, that is able to represent a simple see channel

Representing a hill like structure may be accomplished by $n$ neurons in the hidden layer, that forms a convex polygon in the binary case. The use of a logistic activation function leads to a similar shape, but with smooth edges (see figure 5(c), where $n = 3$). A mountain ridge may be modeled this way too by additionally adjusting the weights of the appropriate synapses (conditions).

### 3.4 Examples

First of all, an academic example will be presented. The function $h(x, y) = \cos(x) \cdot \sin(y)$ in the domain $[-\frac{1}{2}\pi, -\pi] \times [\frac{3}{2}\pi, \pi]$ will be considered (see figure 9).

The following examples reveal, that beside a sufficient network topology, the training data has to be representative, too. Two networks are considered with different topologies. The first one, depicted in figure 10(a), is a multi layer perceptron with two hidden layers (3 and 2 neurons). The second one has 16 neurons in the first hidden layer and 4 neurons in the second. Both where trained with 25 and 81 training patterns each. The training data is arranged in a grid in both cases.

Regarding to section 3.3, a network with insufficient topology is not able to map the desired bathymetry (see figure 10(a) and 10(b)). In figure 10(d) the result of the trained network with

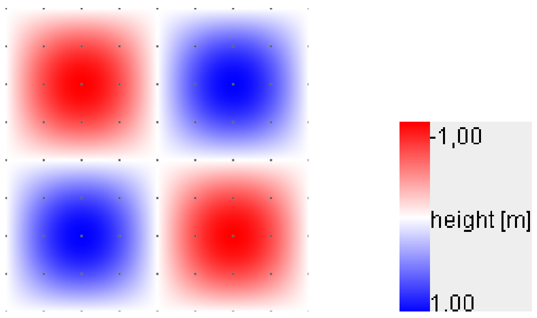
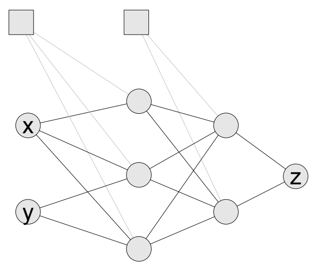
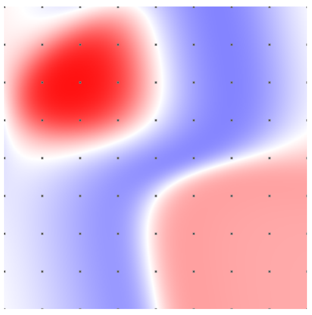


Figure 9: Given function $h(x, y) = \cos(x) \cdot \sin(y)$ in the domain $[-\frac{1}{2}\pi, -\pi] \times [\frac{3}{2}\pi, \pi]$

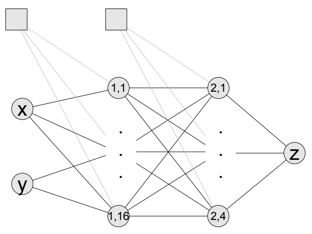
2-16-4-1 topology (illustrated in figure 10(c)) is shown. The $5 \times 5$ training patterns are marked by the dots. The given training patterns are approximated well, but the characteristic of the bathymetry is not reproduced correctly. In figure 10(e) 81 training patterns where used for the training process and the network results in a good approximation.



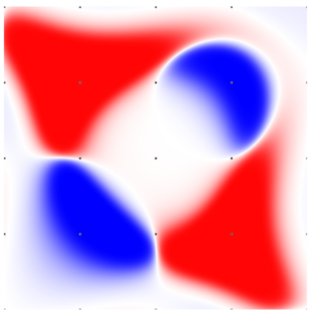
(a) Insufficient network topology (2-3-2-1)



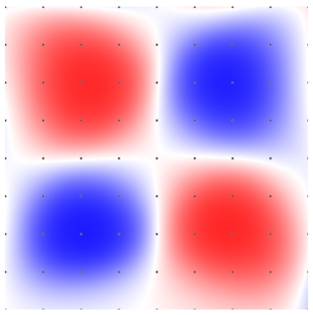
(b) Result trained by $9 \times 9$ patterns



(c) Sufficient network topology (2-16-4-1)



(d) Result trained by $5 \times 5$ patterns



(e) Result trained by $9 \times 9$ patterns

Figure 10: Results of the approximation of the function depicted in figure 9 represented by sufficient (10(b), 10(e)) and insufficient 10(d) sets of training data, using a sufficient 10(c) and an insufficient 10(a) network topology.

As a second example, real world data will be used as training data. Figure 11 illustrates the bathymetry around the Medem Channel in the German Bight. The data is taken from the research project "ImTG" [2], that deals with the identification of morphological trends and process velocities in the near shore zone. The data has been collected by the project partners.

For the ANN-bathymetric model a multilayer perceptron with two hidden layers each consisting of 30 neurons. The result of the approximation is shown in figure 12. The main structures, especially the Medem Channel are approximated well. The finer structures are not repre-

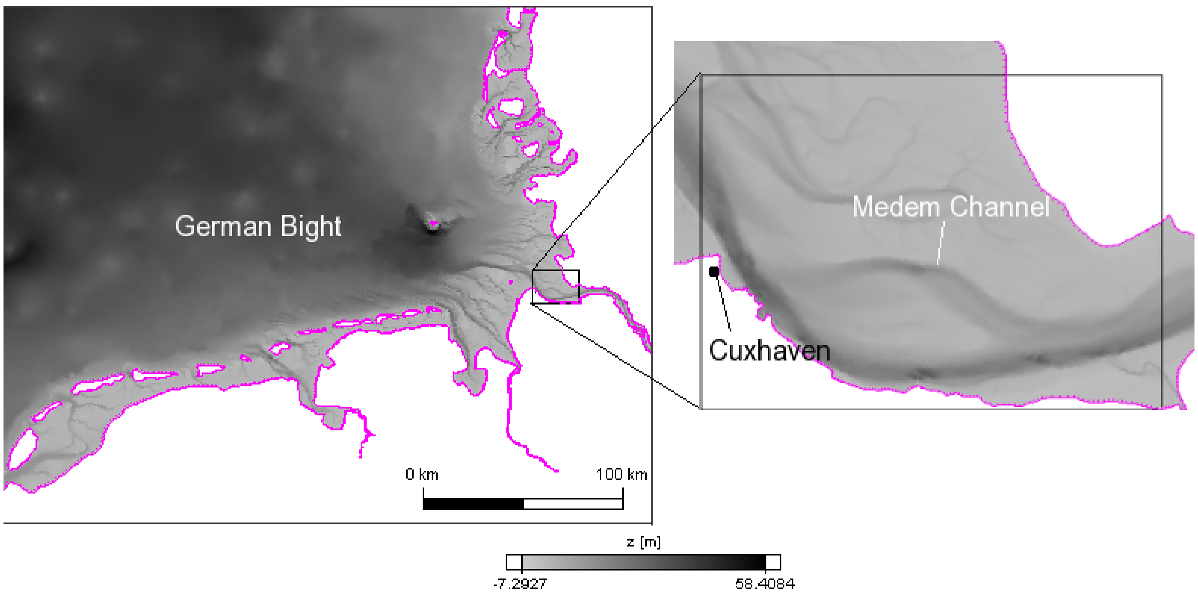


Figure 11: Bathymetric data around the Medem Channel is used as training set for the ANN.

sented in detail. Taking the thoughts of section 3.3 into account could lead to a better approximation in detail. A further approach could be a hierarchical representation of special regions.

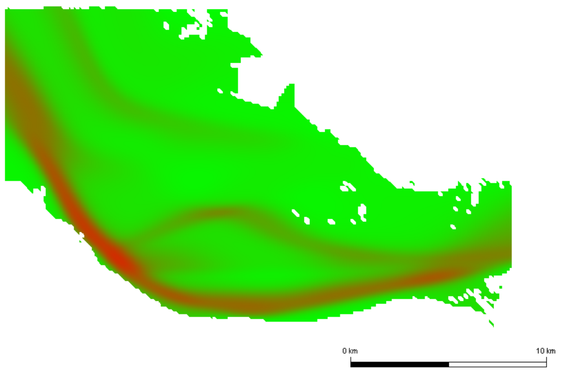
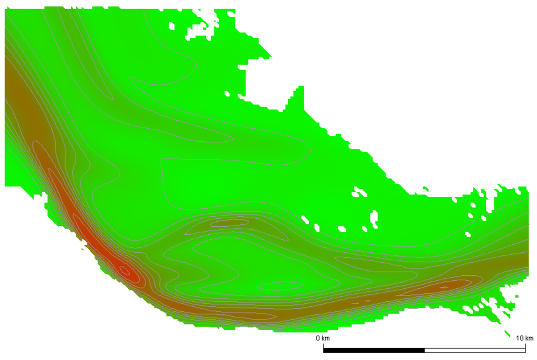


Figure 12: The results produced by an ANN-bathymetric model (2-30-30-1 topology)

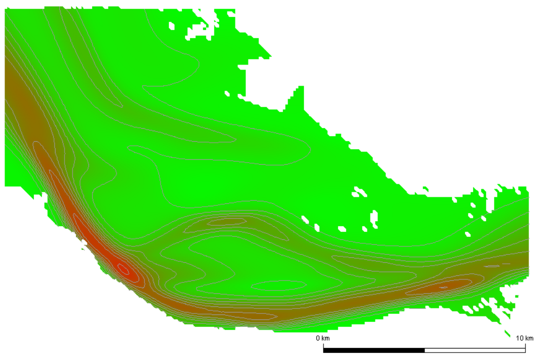
### 3.5 Time dependent bathymetric model

Measuring a bathymetry at different points in time leads to a time dependent bathymetry, which can be modeled by an ANN with three input neurons $(x, y, t)$ and one output neuron $h$. The domain around the Medem Channel is regarded for experimental purposes (see figure 11). Training data was taken from three points in time: 1990, 1991 and 1992. Again, a multilayer perceptron with a 3-30-30-1 topology was used. The results are shown in figure 13. The coarse structures are represented well. Even the movement of the Medem Channel is emulated by the ANN, but it can hardly be seen in the pictures.

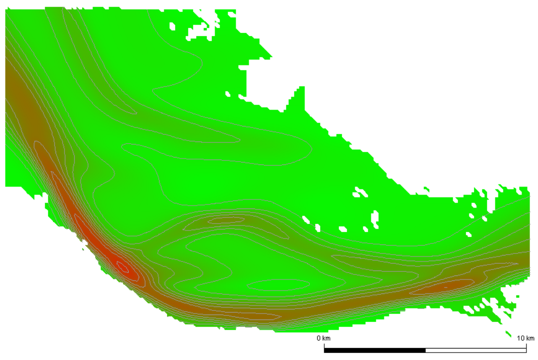
The time dependent bathymetric model using an ANN provides an approximation of a time dependent bathymetry. As in the 2D-case $(h = f(x, y))$, ANN are able to approximate the



(a) Bathymetry in 1990



(b) Bathymetry in 1991



(c) Bathymetry in 1992

Figure 13: Results of the time dependent ANN-bathymetric model (3-30-30-1 topology)

height at a given location (and time). A confident approximation is only achieved within the bounds of the training set and with representative training data. In other words, “extrapolation“[3] does not provide reliable results. Therefore, the point of interest should be contained in the following bounds:

$$x_p \in [x_{min}, x_{max}], y_p \in [y_{min}, y_{max}], t_p \in [t_{min}, t_{max}] \tag{9}$$

To get a model with a more predictive character, meteorological parameters will be introduced in the next section.

## 4 EXTENSION FOR METEOROLOGIC PARAMETERS

It is obvious that morphological changes to the bathymetry are basically caused by the movement of the water. The main influences on the water movement again are tidal effects and excitation by the wind. The tide itself can be described well due to the regular appearance of ebb and flood. The more complex influence of the wind on the wave formation and the resulting morphological changes of the sea bed can hardly be described. The system of effect and interaction between the meteorology, the water body and the bathymetry is depicted in figure 14.

[3] An extrapolation is not possible as such, since the ANN provides no interpolation method but an approximation method.

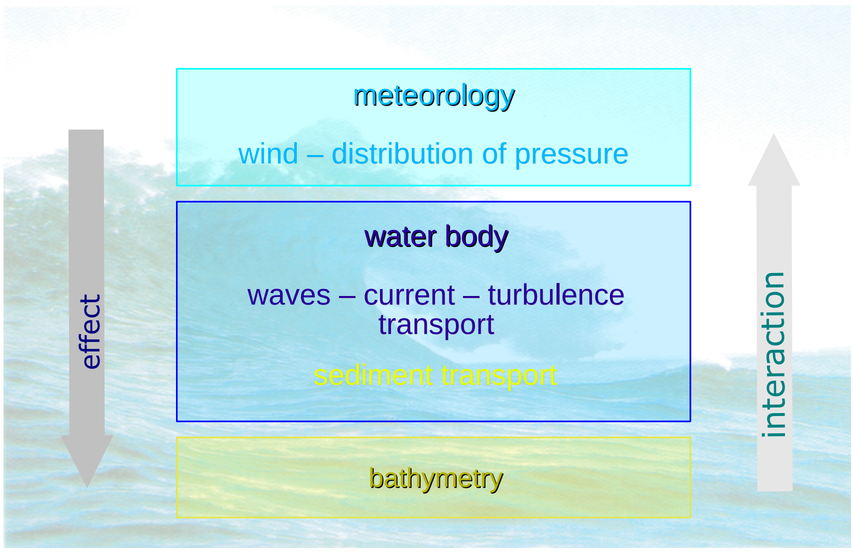


Figure 14: The complex context between wind, waves and bathymetry

## 4.1 Model

In this section a simple model is set up to describe the effects of the wind on the bathymetry. As a first step it is assumed, that the change of the height depends on the location $(x, y)$, the point in time $t$, the height $h$ before the wind takes effect, the wind field $V$, the tidal effects and the exposure duration $\Delta t$ of the wind (and the tide).

$$\Delta h = f(x, y, t, h, \Delta t, \int_t^{t+\Delta t} V(\tau)d\tau, \int_t^{t+\Delta t} \text{“tide“}(\tau)d\tau) \tag{10}$$

Assuming that the wind does not vary significantly with the location, the wind data can be regarded from one characteristic measurement station only. Wind data at a certain location is typically described by the direction $\Phi$ and the velocity $v$ at a certain point in time. In order to obtain adequate training patterns, the direction of the wind has to be quantized. Therefor the direction is divided into $n$ classes. The time dependent model developed in section 3.5 is now being extended: For each class of the direction of wind, an appropriate input neuron is added. The exposure duration is represented by another input neuron. The tidal effects will be trained implicity as before in the 3D-case. The output neuron now represents the change of height $\Delta h$ instead of the height itself. This leads to the following dependency of $\Delta h$:

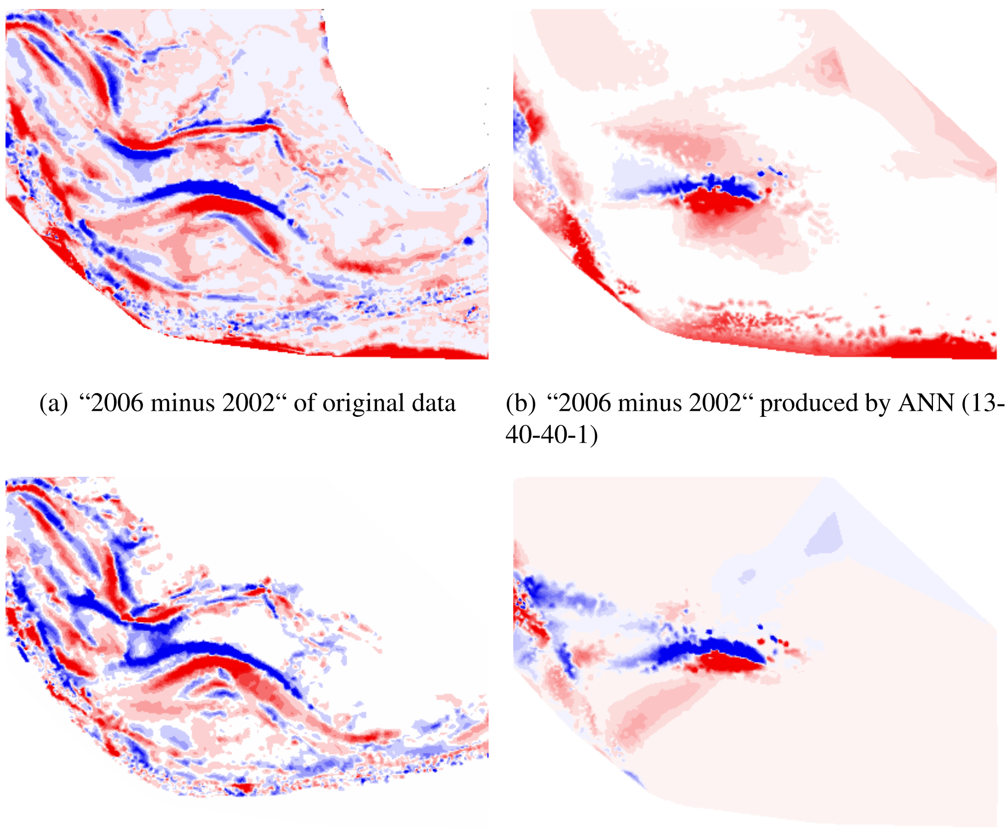
$$\Delta h = f(x, y, t, h, \Delta t, \int_t^{t+\Delta t} w_v^{\Phi_1}(\tau)d\tau, \ldots, \int_t^{t+\Delta t} w_v^{\Phi_n}(\tau)d\tau) \tag{11}$$

The idea is to get a model of more predictive character. Once the ANN is trained well, it is able to produce a prediction due to the influence of the wind coming from a given bathymetry at a certain point in time. First results are presented in the next section.

## 4.2 Example

Regarding to the example of the Medem Channel, wind data is needed for the extended model. Wind data was available again from the research project “ImTG“. The measuring station is located in Cuxhaven (see figure 11). The wind data was collected every hour and is available throughout 18 years (1990 – 2008). As a first approach, the training patterns were obtained by regarding the domain in the years 2002, 2006 and 2008. Out of this, input and output data was determined by calculating the differences in time and height of the layers, i.e. the bathymetry in the year 2008 minus the bathymetry in the year 2006 and "2006 minus 2002" respectively. The wind data was included from the given time series of wind. The data was quantized in 8 classes with respect to the direction and then integrated over time. For this example a 13-40-40-1network topology was used.

In figure 15 the difference of the layers are shown for the original data and the output produced by the model. It is obvious, that the model does not perform very well. But the movement of the Medem Channel is reproduced well. For this is the result of a first approach, better results are to be expected taking better training data into account. A more appropriate network topology could also lead to a better approximation.



(a) "2006 minus 2002" of original data

(b) "2006 minus 2002" produced by ANN (13-40-40-1)

(c) "2008 minus 2006" of original data

(d) "2008 minus 2006" produced by ANN (13-40-40-1)

Figure 15: Results of the extended model compared to the original data.

## 5 CONCLUSION & OUTLOOK

In this paper a generalized approximation method for a time dependent bathymetric model was presented. The model is based on an artificial neuronal network, which is trained by measured data describing a time dependent bathymetry. The representativeness of functions in the context of ANN have been showed in section 2 in general. These have been assigned to the representation of bathymetrical structures in section 3.3. The results for the approximation of a time dependent bathymetry shows a good approximation of the coarse structures. The approximation is expected to be improved by applying the thoughts of the representativeness of bathymetrical structures. Therefor, further investigations have to be done.

A hierarchical approach could also lead to better results. In this case, the domain can be devided into several subdomains, if the results are not well. Then for each subdomain, an additional neuronal network is introduced, which represents this certain subdomain. Another approach could be a hierarchical refinement of the network topology.

In section 4 the model has been extended to take meteorological parameters into account. The goal is to obtain a model with more predictive character. The model does not perform very well at this point in time. More investigations have to be done to improve the model. Especially, a more representative set of training data has to be applied.

## REFERENCES

[1] Andreas Zell, *Simulation Neuronaler Netze*. Addison-Wesley, 1994.

[2] P. Milbradt and C. Dorow, Identification of Morphological Tendencies and Velocities in Coastal Zones, *Science and Information Technologies for Sustainable Management of Aquatic Ecosystems*, Proceedings of HEIC, 2009.

[3] W. Kinnebrock, *Neuronale Netze – Grundlagen, Anwendungen, Beispiele*, R. Oldenburg Verlag, Munich, Vienna, 1994.

[4] C.-P. Tsai, K.-F. Daemrich and C.-L. Ho, *Determination of Wave Overtopping Using Neuronal Network.*